UNIVERSITY OF CALIFORNIA SAN DIEGO

Certifiable Robot Control under Uncertainty: Towards Safety, Stability, and Robustness

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Mathematics with a Specialization in Computational Science

by

Kehan Long

2025

University of California San Diego

2025

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

My Ph.D. journey at University of California, San Diego has been an invaluable and transformative period of my life, filled with learning, growth, and a deep sense of achievement. This journey has shaped not only my technical skills but also my approach to analyzing and solving complex problems. None of this would have been possible without the generous support and guidance of many remarkable individuals.

First and foremost, I wish to express my deepest gratitude to my Ph.D. advisors, Professor Nikolay Atanasov, Professor Jorge Cortés, and Professor Melvin Leok (committee chair). I am immensely thankful to Nikolay for welcoming me into his group and agreeing to serve as my co-advisor at the very beginning, as well as for his insightful guidance, unwavering patience, and consistent support throughout the years. His mentorship helped me identify and pursue the research topics that defined my doctoral work. I am grateful to Jorge for becoming my co-advisor beginning with my first published paper, and for his innovative ideas, rigorous standards, and meticulous attention to detail in every project we undertook together. I am equally thankful to Melvin for his invaluable advice, both in research and in life, and for his perspective that often helped me step back and see the bigger picture. I would also like to express my appreciation to Professor Alexander Cloninger and Professor Rayan Saab for serving on my committee and providing support during major milestones in my Ph.D. studies.

I am grateful to Dr. Georgios Fainekos, Dr. Bardh Hoxha, and Hideki Okamoto for their mentorship during my summer internship at Toyota Research Institute North America. My time in Ann Arbor with the team was both professionally enriching and personally enjoyable. The experience gave me valuable exposure to how research ideas can be translated into industrial applications, and it strengthened my interest in bridging academic research with real-world robotics challenges.

I would also like to extend my heartfelt thanks to my collaborators and co-authors, Sylvia Herbert, Lars Lindermann, Tania Morimoto, Pol Mestres, Yinzhuang Yi, Vikas Dhiman, Ki Myung Brian Lee, Aedan Mangan, Hardik Parwana, Zhirui Dai, Yiqi Zhao, Nikola Raicevic, Khoa

Tran, and Cheng Qian. I am equally grateful to my colleagues Yulun Tian, Arash Asgharivaskasi, Thai Duong, Shumon Koga, Tianyu Wang, Hanwen Cao, Zhichao Li, Qiaojun Feng, Sunghwan Kim, Dwait Bhatt, Mani Amani, Tianji Tang, Jason Stanley, and many others whose names I may have missed. Working alongside them has been instrumental to the development and implementation of my research and a deeply rewarding experience. Their insights, feedback, and shared enthusiasm were invaluable in overcoming challenges and pushing the boundaries of my work.

I have greatly enjoyed the vibrant atmosphere of Southern California, with its beautiful weather, diverse restaurants, and scenic hiking trails, as well as relaxing ski trips with friends during the winters. These experiences provided much-needed balance and renewal throughout my Ph.D. journey and helped me maintain perspective during challenging times.

Last but not least, my deepest love and gratitude go to my family. I am forever thankful to my parents for their unwavering support, both emotional and financial, throughout every stage of this journey. I am very grateful to my sister for sharing in my struggles and successes and for reminding me of the joy and warmth of family. I also cherish the memory of our family dog Jiujiu, whose playful spirit brought comfort and happiness.

I owe special thanks to my partner, Shuya, whose kindness and thoughtfulness continue to inspire me. Living with her has helped me grow into a more disciplined, organized, and balanced person. Her love and encouragement have been a constant source of motivation, and I am endlessly grateful for her presence in my life. I am thankful to our two cats, Ursa and Reiki, whose quiet companionship and gentle presence filled my days with joy and peace during my Ph.D. years.

Chapter 3, in full, is a reprint of the material as it appears in Learning Barrier Functions With Memory for Robust Safe Navigation, K. Long, C. Qian, J. Cortés, and N. Atanasov, IEEE Robotics and Automation Letters, 2021; Safe Control Synthesis With Uncertain Dynamics and Constraints, K. Long, V. Dhiman, M. Leok, J. Cortés, and N. Atanasov, IEEE Robotics and Automation Letters, 2022; and Safe Stabilizing Control for Polygonal Robots in Dynamic

Elliptical Environments, K. Long, K. Tran, M. Leok, and N. Atanasov, American Control Conference, 2024. The dissertation author was the primary researcher and author of these works.

Chapter 4, in full, is a reprint of the material as it appears in Safe and Stable Control Synthesis for Uncertain System Models via Distributionally Robust Optimization, K. Long, Y. Yi, J. Cortés, and N. Atanasov, American Control Conference, 2023; Sensor-Based Distributionally Robust Control for Safe Navigation in Dynamic Environments, K. Long, Y. Yi, Z. Dai, S. Herbert, J. Cortés, and N. Atanasov, International Journal of Robotics Research, 2025; and Neural Configuration-Space Barriers for Manipulation Planning and Control, K. Long, K. M. B. Lee, N. Raicevic, N. Attasseri, M. Leok, and N. Atanasov, under review, 2025. The dissertation author was the primary researcher and author of these works.

Chapter 5, in full, is a reprint of the material as it appears in Distributionally Robust Lyapunov Function Search Under Uncertainty, K. Long, Y. Yi, J. Cortés, and N. Atanasov, Learning for Dynamics and Control Conference, 2023; Distributionally Robust Policy and Lyapunov-Certificate Learning, K. Long, J. Cortés, and N. Atanasov, IEEE Open Journal of Control Systems, 2024; and Generalized Lyapunov Stability for Certified Control and Reinforcement Learning, K. Long, J. Cortés, and N. Atanasov, under review, 2025. The dissertation author was the primary researcher and author of these works.

VITA

2019	B.S. in Applied Mathematics, University of Illinois Urbana-Champaign
2024	M.A in Mathematics, University of California San Diego
2025	Ph.D. in Mathematics with a Specialization in Computational Science, University of California San Diego

PUBLICATIONS

K. Long, J. Cortés and N. Atanasov, "Generalized Lyapunov Stability for Certified Control and Reinforcement Learning," under review, 2025.
K. Long, K. M. B. Lee, N. Raicevic, N. Attasseri, M. Leok, N. Atanasov, "Neural Configuration-Space Barriers for Manipulation Planning and Control," under review, 2025.
K. Long, Y. Yi, Z. Dai, S. Herbert, J. Cortés and N. Atanasov, "Sensor-Based Distributionally Robust Control for Safe Navigation in Dynamic Environments," International Journal of Robotics Research (IJRR), 2025.
K. Long, J. Cortés and N. Atanasov, "Distributionally Robust Policy and Lyapunov-Certificate Learning," IEEE Open Journal of Control Systems (OJ-CSYS), 2024.
K. Long, K. Tran, M. Leok and N. Atanasov, "Safe Stabilizing Control for Polygonal Robots in Dynamic Elliptical Environments," American Control Conference (ACC), 2024.
K. Long, Y. Yi, J. Cortés and N. Atanasov, "Distributionally Robust Lyapunov Function Search Under Uncertainty," Learning for Dynamics and Control Conference (L4DC), 2023.
K. Long, Y. Yi, J. Cortés and N. Atanasov, "Safe and Stable Control Synthesis for Uncertain System Models via Distributionally Robust Optimization," American Control Conference (ACC), 2023.
K. Long, V. Dhiman, M. Leok, J. Cortés and N. Atanasov, "Safe Control Synthesis With Uncertain Dynamics and Constraints," IEEE Robotics and Automation Letters (RA-L), 7(3), pp. 7295-7302, 2022.
K. Long, C. Qian, J. Cortés and N. Atanasov "Learning Barrier Functions With Memory for Robust Safe Navigation,", IEEE Robotics and Automation Letters (RA-L), 6(3), pp. 4931-4938, 2021.

ABSTRACT OF THE DISSERTATION

Certifiable Robot Control under Uncertainty: Towards Safety, Stability, and Robustness

by

Kehan Long

Doctor of Philosophy in Mathematics with a Specialization in Computational Science

University of California San Diego, 2025

Professor Melvin Leok, Chair
Professor Nikolay A. Atanasov, Co-Chair
Professor Professor Jorge Cortés, Co-Chair

Autonomous robotic systems must operate safely and reliably in environments that involve significant uncertainty, including unmodeled dynamics, noisy sensor measurements, unpredictable obstacles, and learned representations. This dissertation develops a collection of methods that enable safe, stable, and certifiable control for such systems by integrating principles from control theory, robust and distributionally robust optimization, and learning-based techniques.

First, robust and probabilistic control synthesis methods are introduced that extend control barrier functions (CBFs) and control Lyapunov functions (CLFs) to handle bounded or stochastic model uncertainty and imperfect perception. These methods enable real-time safe navigation for mobile robots in unknown environments using onboard sensor data.

Second, a distributionally robust control framework is developed to address uncertainty in obstacle motion, localization, and sensor noise. Leveraging tools from distributionally robust optimization (DRO), the proposed methods ensure safety under worst-case distributional shifts. The framework is validated through sensor-based navigation and manipulation tasks in dynamic and cluttered environments.

Third, a set of methods is proposed for certifying the stability of learned control policies, including those trained via reinforcement learning. This includes distributionally robust Lyapunov function formulations that enable stability guarantees under unknown model perturbations using only limited uncertainty samples. Moreover, a generalized Lyapunov framework is proposed for certifying the stability of optimal control and reinforcement learning policies. By augmenting value functions with neural residual terms and enforcing a multi-step Lyapunov decrease condition, the framework enables practical verification of learned controllers and supports joint training of policies and stability certificates.

Together, these contributions advance the theoretical and algorithmic foundations of robust, interpretable, and adaptive robot autonomy, enabling reliable and safe operation of autonomous systems in complex and uncertain real-world environments.

Chapter 1

Introduction

1.1 Motivation

Enabling safe and reliable robotic autonomy remains a fundamental challenge at the intersection of robotics, control, and machine learning. Robots are increasingly being deployed in unstructured and dynamic environments such as autonomous driving, warehouse automation, surgical assistance, household tasks, and planetary exploration. In these settings, it is critical that robots operate reliably and robustly under uncertainty. Success in this area would enable robots to assist or even replace humans in high-risk or complex tasks, expanding the scope and impact of automation.

Unlike humans, who act based on intuition, experience, and learned heuristics, robots can systematically integrate multimodal sensor data, leverage prior knowledge of their own dynamics, and reason computationally about the consequences of their actions. This capacity opens up exciting possibilities for developing autonomous systems that not only achieve high performance but also offer interpretability, robustness, and formal guarantees. The long-term vision of my research is to develop autonomous robots that can match and ultimately exceed human-level agility, safety, and decision-making reliability in complex and uncertain environments.

However, realizing this vision requires addressing several core technical challenges. Robots must react in real time to unexpected changes in their surroundings, while accounting for uncertainty arising from inaccurate models, noisy sensors, imperfect localization, and even learned components such as neural network-based dynamics or perception modules. These forms of uncertainty make it difficult to synthesize control policies that are both high-performing and provably safe or stable, particularly in dynamic environments or safety-critical domains.

While classical control methods offer strong theoretical guarantees, such as those based on optimal control, control barrier functions (CBFs), and control Lyapunov functions (CLFs), they often assume perfect knowledge of dynamics and environmental structure. This limits their ability to scale or generalize to real-world variability. In contrast, learning-based methods, such as reinforcement learning, have demonstrated remarkable empirical success in complex and high-dimensional tasks. Yet, they often lack the theoretical guarantees and interpretability required for deployment in high-stakes settings, where reliability, safety, and accountability are non-negotiable.

This thesis aims to bridge these paradigms by developing a unified framework for safe, robust, and adaptive robot control under uncertainty. The approach integrates insights from control theory, robust and distributionally robust optimization, and machine learning to produce control strategies that are formally grounded, uncertainty-aware, and scalable to unknown environments. By combining classical structure with modern learning techniques, the resulting algorithms bring us closer to the goal of certifiable autonomy in real-world robotic systems.

1.2 Problem Statement

The central goal of this thesis is to develop safe, stable, and reliable control policies for robotic systems operating in dynamic and uncertain environments. This involves unifying control-theoretic formulations with modern learning-based approaches to achieve both real-time decision-making and performance guarantees.

The first objective is to extend classical CBFs and CLFs to handle model inaccuracies, unknown disturbances, and sensor-induced uncertainty. In particular, we aim to enable safe and stabilizing control synthesis for mobile robots with complex geometries navigating in unknown environments using only onboard sensors.

Second, we address the limitations of worst-case robust or probabilistic methods by developing a distributionally robust framework that accounts for ambiguity in the data-generating distribution. Using tools from distributionally robust optimization (DRO), we aim to quantify uncertainty in obstacle motion, localization, and sensor readings, and synthesize control policies that ensure safety even with distributional shifts.

Third, we seek to bridge the gap between classical control theory and learning-based control by developing scalable methods for analyzing and certifying the stability of neural control policies, including those trained via reinforcement learning (RL). We formulate a generalized Lyapunov stability framework that augments value functions with neural residuals and enforces multi-step decrease conditions. This enables not only post-hoc certification but also the joint training of control policies and Lyapunov certificates.

Together, these objectives advance the frontier of safe and reliable robotic autonomy by enabling real-time decision-making with guarantees and interpretability, supporting deployment in complex, real-world environments.

1.3 Related Work

This section provides a high-level overview of related work relevant to the core contributions of this thesis. The discussion is organized around three key themes that shaped the trajectory of my Ph.D. research: safe control synthesis, distributionally robust optimization, and neural stability certification. More detailed comparisons to closely related methods are deferred to the corresponding technical chapters.

Safe Control:

In the early 2000s, barrier certificates were introduced as formal tools to verify safety for nonlinear and hybrid systems [120, 121]. These methods established conditions under which a system would remain within a safe set indefinitely, without requiring complete knowledge of the control law. Building upon this foundation, control barrier functions (CBFs) were proposed as a synthesis tool for safety-critical control of nonlinear systems [144], offering a constructive method to design controllers that enforce forward invariance of safe sets.

A key breakthrough came from the observation that, for control-affine systems, both control Lyapunov function (CLF) and CBF conditions are affine in the control input. This structure allows safe and stabilizing controllers to be synthesized via real-time quadratic programming (QP) [7, 8]. The resulting CLF-CBF QP framework has been widely adopted due to its modularity and efficiency, and has demonstrated success in applications such as adaptive cruise control, robotic locomotion, and multi-robot coordination.

While the CLF-CBF QP formulation provides formal guarantees for deterministic systems, many robotics applications must contend with model uncertainty, imperfect state estimation, and noisy or incomplete sensory inputs. To address these challenges, robust and probabilistic extensions of CLF-CBF QPs have been proposed. These include formulations that account for bounded model uncertainty [46], probabilistic safety guarantees under stochastic disturbances [40], and observer-based approaches that incorporate state estimation uncertainty [37, 143].

Recent work has also focused on learning-based and sensor-driven synthesis of barrier functions, enabling robots to infer safety constraints from partial observations and data-driven priors. For example, CBFs have been learned from range data using support vector machines [134], synthesized from RGB-D and LiDAR inputs for vision-based navigation [1, 77], and constructed through differentiable pipelines for locomotion and obstacle avoidance tasks [95, 154]. These methods aim to bridge the gap between formal control-theoretic safety guarantees and the uncertainty and variability present in real-world environments. Much of this progress has evolved concurrently with the development of this thesis, reflecting a broader shift toward combining perception and learning with certified safety methods.

In parallel, there has been a surge in robust and probabilistic extensions of control barrier functions to explicitly handle uncertainty in the system model, external disturbances, and sensor noise. Jankovic [72] and Eman *et al.* [45] formulate robust CBF conditions using worst-case disturbance bounds and convex hulls, respectively. Nguyen and Sreenath [113] introduce a robust CLF-CBF QP that incorporates uncertainty into both safety and stability constraints. Clark [27] addresses stochastic systems with partial observability and derives sufficient conditions to ensure average-case safety.

Beyond worst-case guarantees, risk-sensitive formulations have been explored. Ahmadi *et al.* [2] propose a CVaR-based CBF framework to ensure safety with high probability under stochastic uncertainty. Relatedly, model predictive control (MPC) approaches have been proposed that embed safety constraints informed by probabilistic models. For instance, Hewing *et al.* [68] combine a nominal model with a Gaussian Process residual to account for modeling error, while Alcan and Kyrki [4] and Hassan *et al.* [6] use differential dynamic programming to embed safety constraints into trajectory optimization.

Other lines of work explore robust safety under disturbances and estimation errors using input-to-state safety (ISSf) concepts. Romdlony and Jayawardhana [127] introduce ISSf to characterize safety under bounded disturbances, which was later used by Kolathaya *et al.* [80] and Alan *et al.* [3] to enlarge safe sets and reduce conservatism. Measurement-robust formulations have also emerged, such as the work of Cosner *et al.* [29], which explicitly incorporates state estimation uncertainty in CBF design and demonstrates safety on physical systems.

Distributionally robust optimization.

Distributionally robust optimization (DRO) has emerged as a powerful framework to address uncertainty in optimization problems, especially in scenarios with limited or noisy data. Unlike classical robust methods that consider worst-case uncertainty, DRO explicitly models ambiguity in the distribution of uncertain parameters. To capture the potential discrepancy between empirical and true distributions, researchers have introduced several types of ambiguity sets, including moment-based sets [139], Kullback–Leibler divergence balls [74], and Wasserstein balls [47, 71, 147]. These formulations have been widely adopted in machine learning [88, 130], uncertainty quantification [20], and increasingly, in control theory [18, 19, 89, 150] and robotics [30, 124, 128].

Several works have also demonstrated the utility of DRO in safe planning and decision making under uncertainty. Ren and Majumdar [124] introduced a DRO-based reinforcement learning approach that trains policies against adversarial environments generated from a learned generative model. Hakobyan and Yang [56] proposed a distributionally robust risk map formulation for mobile robot safety, casting the resulting infinite-dimensional problem into a tractable semidefinite program. Lathrop *et al.* [84] developed a Wasserstein-safe RRT variant with finite-sample safety guarantees. Similarly, Summers [136] proposed a moment-based ambiguity formulation for motion planning under uncertainty. In the control domain, Aolaritei *et al.* [12] introduced Wasserstein tube MPC, enhancing robustness by constructing trajectory tubes around a nominal plan using Wasserstein balls. Bahari *et al.*[82] proposed a reinforcement learning-based DRO-MPC formulation, enabling robust control for stochastic systems. Finally, Hakobyan and Yang [57] developed a DRO variant of differential dynamic programming by leveraging Kantorovich duality to retain tractability.

A related line of research to our work is the application of conformal prediction in robot navigation tasks. Conformal prediction [132, 157] is a statistical tool for uncertainty quantification that provides valid prediction regions with a user-specified risk tolerance, making it particularly useful for ensuring safety in dynamic environments. Several recent works have explored the integration of conformal prediction into motion planning and control frameworks. Lindemann *et al.* [94] used conformal prediction to obtain prediction regions for a model predictive controller. Yang *et al.* [152] employed conformal prediction to quantify state estimation uncertainty and design a robust CBF controller based on the estimated uncertainty. An adaptive conformal prediction algorithm was developed by [42] to dynamically quantify prediction uncertainty and plan probabilistically safe paths around dynamic agents.

Neural Lyapunov Functions and Stability-Certified Policies:

Lyapunov theory provides a principled foundation for stabilizing control design in nonlinear systems. Classical methods for controller synthesis rely on constructing control Lyapunov functions (CLFs) that certify asymptotic stability. For linear systems, Lyapunov-stable controllers can be efficiently synthesized using linear matrix inequalities (LMIs) and linear quadratic regulator (LQR) techniques [10]. For polynomial systems, sum-of-squares (SOS) programming enables joint synthesis of controllers and certificates [73]. However, SOS-based approaches are limited in scalability and expressivity: not all valid Lyapunov functions are SOS-representable [69], and the computational cost grows rapidly with system dimension and polynomial degree.

To overcome these limitations, recent work has explored neural networks as flexible function approximators for Lyapunov functions and stabilizing controllers. Early efforts include learning Lyapunov functions to characterize regions of attraction [125], and jointly synthesizing controllers and certificates with formal verification using satisfiability modulo theories (SMT)[24, 158]. Subsequent works incorporated Lyapunov structure into network architectures[17, 50], or used optimization-based verification tools such as mixed-integer programming [33] and α, β-CROWN [142, 151] to ensure stability guarantees. Extensions to safety-critical control problems have been explored via joint Lyapunov-barrier functions [36] and distributionally robust Lyapunov formulations under model uncertainty [103]. See also [35] for a recent survey.

The relationship between reinforcement learning (RL) and Lyapunov stability has also received increasing attention. Early formulations focused on safe policy learning via Lyapunov constraints [26, 66, 118], including model-based RL methods that leverage Lyapunov verification to ensure safe exploration [15]. More recent works seek to integrate Lyapunov conditions directly into the learning pipeline, such as through regularization [66] and demonstration-driven training [109]. These methods aim to bridge the gap between classical control theory and modern policy learning by encoding stability guarantees into learned decision-making systems.

1.4 Outline and Contributions

The overarching goal of this thesis is to develop principled methods for safe and stable control of nonlinear robotic systems operating under uncertainty. This involves addressing three core challenges: (1) synthesizing safe control policies for robots in unknown and dynamic environments, (2) ensuring safety using onboard sensing in the presence of uncertainty and potential distributional shift, and (3) enabling stability guarantees for learned neural control policies.

To address these challenges, this thesis introduces a set of novel formulations and algorithms grounded in control theory, optimization, and machine learning. The work emphasizes the integration of formal guarantees, including Lyapunov stability, barrier safety, and distributionally robust optimization, with scalable learning-based methods. Below is a brief overview of the chapters.

Chapter 2:

This chapter provides the foundational background for the thesis, reviewing key concepts in Lyapunov theory, control barrier functions, distributionally robust optimization, and reinforcement learning. These tools form the theoretical basis for the control formulations and learning algorithms developed in subsequent chapters.

Chapter 3:

This chapter develops control synthesis methods that enable mobile robots to navigate safely in unknown environments using onboard sensing. The first contribution is a formulation for learning CBFs incrementally from sensor data with memory, allowing robots to maintain safety. The second contribution presents a convex reformulation of the control synthesis problem under bounded or stochastic uncertainty in both robot dynamics and barrier constraints. These formulations bridge formal safety guarantees with real-time implementation, enabling certifiably safe control in practice. The chapter emphasizes the integration of sensing, learning, and optimization to handle uncertainty in a structured and theoretically sound manner.

Chapter 4:

This chapter develops methods for synthesizing safe controllers under distributional uncertainty by leveraging tools from distributionally robust optimization (DRO). Specifically, it integrates control barrier functions (CBFs) with Wasserstein ambiguity sets to provide probabilistic safety guarantees from limited samples. The chapter introduces distributionally robust safety filters that account for various real-world uncertainty sources—such as sensor noise, localization drift, and imperfect neural representations—using only onboard observations. These methods are validated on both ground mobile robots and 6-DOF manipulators, where learned signed distance functions (SDFs) and configuration space distance functions (CDFs) enable geometry-aware control in cluttered environments.

Chapter 5:

This chapter presents novel formulations for learning Lyapunov functions and stabilizing control policies in the presence of model uncertainty. We begin by introducing the concept of distributionally robust Lyapunov functions (DR-LFs), which certify stability using only sampled disturbances, without requiring known uncertainty bounds. To construct these certificates, we develop both sum-of-squares and neural network-based formulations that enforce Lyapunov conditions over Wasserstein ambiguity sets, providing high-confidence guarantees under distributional uncertainty. The approach is further extended to support the joint synthesis of stabilizing neural controllers and DR-LFs, enabling certifiable control of uncertain nonlinear systems. Next, we present a new formulation for certifying the stability of control policies derived from reinforcement learning and optimal control. We leverage the concept of a generalized Lyapunov function, which relaxes the classical step-wise decrease condition to a multi-step, weighted criterion, enabling certification even when traditional Lyapunov functions fail. By augmenting value functions with residual terms and learning state-dependent weights, the proposed method supports both post hoc stability certification and joint training of neural controllers and certificates.

Chapter 6:

This chapter concludes the thesis by highlighting its central contributions to certifiable robot autonomy under uncertainty and outlining promising directions for future research. The work develops robust, probabilistic, and distributionally robust formulations of barrier and Lyapunov functions, demonstrating their effectiveness in enabling safe and stable control of mobile robots and manipulators in uncertain, dynamic environments. It shows how rigorous control-theoretic guarantees can be combined with neural representations and learning-based methods to handle compounded uncertainties and to certify the stability of reinforcement learning policies. At the same time, it acknowledges open challenges such as feasibility and conservatism in barrier formulations, scalability to high-dimensional systems, and broader notions of safety beyond collision avoidance. Looking ahead, the chapter charts three directions: unifying optimality and stability in learning-based control, extending safety to open-world and contact-rich settings, and developing scalable whole-body task and motion planning frameworks. Together, these advances and open problems point toward a future where robots operate not only with dexterity and agility, but also with reliability, interpretability, and potentially formal guarantees.

Chapter 2

Background

The goal of this chapter is to introduce foundational tools used throughout the thesis for safe and stable control of nonlinear robotic systems operating under uncertainty. In real-world robotics, safety refers to the ability of a system to avoid undesirable outcomes (such as collisions or violations of physical constraints), while stability ensures that the system's behavior remains well-behaved over time, ideally converging to a desired configuration or trajectory. These properties are crucial for deploying robotic systems in dynamic, unknown, and high-stakes environments.

We begin by reviewing classical notions of Lyapunov stability and control Lyapunov functions (CLFs), which formalize liveness through convergence guarantees. CLFs provide a constructive way to synthesize stabilizing controllers, especially for nonlinear and control-affine systems, by ensuring that a scalar-valued function decreases along system trajectories. Next, we discuss control barrier functions (CBFs), which encode safety by enforcing forward invariance of constraint sets—ensuring that, once a system enters a safe region, it remains inside for all future time. Together, CLFs and CBFs offer a unified framework for synthesizing controllers that guarantee both safety and stability through real-time optimization techniques, often formulated as quadratic programs.

To extend these guarantees under uncertainty, we introduce techniques from distributionally robust optimization (DRO). These tools are particularly useful when only limited or noisy data is available to characterize the environment, dynamics, or sensing uncertainties. DRO provides a way to make worst-case-optimal decisions by optimizing over an ambiguity set of possible distributions, enabling safety- and performance-critical decisions beyond empirical samples.

Finally, we provide a brief overview of optimal control and reinforcement learning (RL), with an emphasis on their connections to Lyapunov stability and safety-critical control. We focus particularly on model-based and constrained formulations that intersect with control-theoretic tools, highlighting their important roles in planning, learning, and decision-making.

Notations

Throughout the thesis, we adopt the following notations:

The sets of real, non-negative real, and natural numbers are denoted by $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, and $\mathbb{N}$, respectively. For $N \in \mathbb{N}$, we write $[N] := \{1, 2, \ldots N\}$. We denote the distribution and expectation of a random variable Y by $\mathbb{P}$ and $\mathbb{E}_{\mathbb{P}}(Y)$, respectively. The interior and boundary of a set $\mathcal{C} \subset \mathbb{R}^n$ are denoted by Int$(\mathcal{C})$ and $\partial\mathcal{C}$. We denote by $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ the identity matrix. For a vector $\mathbf{x}$, the notation $|\mathbf{x}|$ represents its element-wise absolute value, while $\|\mathbf{x}\|_1$, $\|\mathbf{x}\|$, and $\|\mathbf{x}\|_\infty$ denote its L_1, L_2, and L_∞ norms, respectively. For a matrix $\mathbf{X}$, we use $\|\mathbf{X}\|$ to denote the spectral norm. We use $\mathrm{vec}(\mathbf{X}) \in \mathbb{R}^{nm}$ to denote the vectorization of $\mathbf{X} \in \mathbb{R}^{n \times m}$, obtained by stacking its columns. We use $\mathbf{0}_n$ and $\mathbf{0}_{n \times n}$ to denote the zero vector and matrix of size n and $n \times n$.

We denote by ∇ the gradient and $\mathcal{L}_{\mathbf{f}} V = \nabla V \cdot \mathbf{f}$ the Lie derivative of a differentiable function V along a vector field $\mathbf{f}$. We use $\otimes$ to denote the Kronecker product and $\mathcal{GP}(\mu(\mathbf{x}), K(\mathbf{x}, \mathbf{x}'))$ to denote a Gaussian Process distribution with mean function $\mu(\mathbf{x})$ and covariance function $K(\mathbf{x}, \mathbf{x}')$. A continuous function $\alpha : [0, a) \to [0, \infty)$ is of class $\mathcal{K}$ if it is strictly increasing and $\alpha(0) = 0$. A continuous function $\alpha : \mathbb{R} \to \mathbb{R}$ is of extended class $\mathcal{K}_\infty$ if it is strictly increasing, $\alpha(0) = 0$, and $\lim_{r \to \infty} \alpha(r) = \infty$. The special orthogonal group of dimension p is denoted by SO(p), which is defined as the set of all $p \times p$ orthogonal matrices with determinant equal to 1: SO$(p) = \{\mathbf{R} \in \mathbb{R}^{p \times p} \mid \mathbf{R}^\top \mathbf{R} = \mathbf{I}_p, \det(\mathbf{R}) = 1\}$.

2.1 Lyapunov Theory

Lyapunov theory provides a fundamental framework for analyzing the stability of dynamical systems. In robotics, it is often used to certify that a control policy drives the system toward a desired equilibrium. This section introduces classical Lyapunov functions for autonomous systems and then extends to CLFs for control-affine systems.

2.1.1 Lyapunov Stability for Autonomous Systems

Consider an autonomous system of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{2.1}$$

where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state, and $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ is assumed to be locally Lipschitz to ensure existence and uniqueness of the solutions to (2.1) [78]. Suppose the origin $\mathbf{x} = \mathbf{0}$ is an equilibrium point, i.e., $\mathbf{f}(\mathbf{0}) = \mathbf{0}$.

A continuously differentiable function $V : \mathcal{X} \to \mathbb{R}$ is called a Lyapunov function if it satisfies:

$$V(\mathbf{0}) = 0, \quad V(\mathbf{x}) > 0, \quad \dot{V}(\mathbf{x}) < 0 \quad \forall \mathbf{x} \neq \mathbf{0}, \tag{2.2}$$

where $\dot{V}(\mathbf{x}) = \nabla V(\mathbf{x})^\top \mathbf{f}(\mathbf{x})$ denotes the time derivative along system trajectories.

If, in addition, $V(\mathbf{x}) \to \infty$ as $\|\mathbf{x}\| \to \infty$, then V is said to be radially unbounded, and the origin is globally asymptotically stable.

Sum-of-Squares Optimization.

Sum-of-squares (SOS) optimization provides a powerful method for certifying Lyapunov stability of polynomial systems. A polynomial $p(\mathbf{x})$ is SOS if there exist polynomials $\{s_i(\mathbf{x})\}$ such that $p(\mathbf{x}) = \sum_i s_i^2(\mathbf{x})$. This implies $p(\mathbf{x}) \geq 0$ for all $\mathbf{x}$, providing a tractable certificate of non-negativity.

If $\mathbf{f}$ and V are polynomial functions, the Lyapunov conditions:

$$V(\mathbf{x}) > 0, \quad \dot{V}(\mathbf{x}) < 0, \tag{2.3}$$

can be relaxed to:

$$V(\mathbf{x}) - \epsilon\|\mathbf{x}\|^2 \in \text{SOS}, \quad -\dot{V}(\mathbf{x}) - \epsilon\|\mathbf{x}\|^2 \in \text{SOS}, \tag{2.4}$$

for some small $\epsilon > 0$. These SOS constraints can be encoded as semidefinite programs, enabling efficient computation of polynomial Lyapunov functions for nonlinear systems [85, 115]. SOS programming is especially valuable for certifying stability for systems with algebraic structure, and has been widely applied in control and formal verification.

2.1.2 Control Lyapunov Functions (CLFs)

We now consider control-affine systems of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u} = [\mathbf{f}(\mathbf{x})\ \mathbf{G}(\mathbf{x})] \cdot \begin{bmatrix} 1 \\ \mathbf{u} \end{bmatrix} := \mathbf{F}(\mathbf{x})\underline{\mathbf{u}}, \tag{2.5}$$

where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ is the system state, $\mathbf{u} \in \mathbb{R}^m$ is the control input, and $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$, $\mathbf{G} : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are locally Lipschitz.

Definition 2.1.1. A continuously differentiable function $V : \mathcal{X} \to \mathbb{R}_{\geq 0}$ is called a *control Lyapunov function (CLF)* for the system (2.5) if:

1. $V(\mathbf{x})$ is positive definite, i.e., $V(\mathbf{x}) > 0$ for all $\mathbf{x} \neq \mathbf{0}$ and $V(\mathbf{0}) = 0$;

2. There exists a class $\mathcal{K}$ function α_V such that, for all $\mathbf{x} \in \mathcal{X} \setminus \{\mathbf{0}\}$, there exists a control input $\mathbf{u} \in \mathbb{R}^m$ satisfying:

$$\dot{V}(\mathbf{x}, \mathbf{u}) + \alpha_V(V(\mathbf{x})) \leq 0, \tag{2.6}$$

or equivalently,

$$\text{CLC}(\mathbf{x},\mathbf{u}) := \mathcal{L}_{\mathbf{f}}V(\mathbf{x}) + \mathcal{L}_{\mathbf{G}}V(\mathbf{x})\mathbf{u} + \alpha_V(V(\mathbf{x})) \leq 0. \tag{2.7}$$

The set of admissible inputs that satisfy the control Lyapunov condition (CLC) is defined as:

$$K_{\text{CLF}}(\mathbf{x}) = \left\{\mathbf{u} \in \mathbb{R}^m : \text{CLC}(\mathbf{x},\mathbf{u}) \leq 0\right\}, \tag{2.8}$$

The existence of a CLF guarantees that the system can be stabilized to the origin using a feedback controller $\mathbf{u}(\mathbf{x})$ chosen from $K_{\text{CLF}}(\mathbf{x})$.

2.2 Barrier Theory

Barrier theory provides tools to formally certify the forward invariance of a given safe set under the evolution of a dynamical system. In robotics, it is widely used to ensure that the system state remains within a safe region for all time. This section focuses on control barrier functions (CBFs) for control-affine systems.

2.2.1 Control Barrier Functions (CBFs)

Consider the control-affine system (2.5). Let the safe set be defined as the superlevel set of a continuously differentiable function $h : \mathcal{X} \to \mathbb{R}$:

$$\mathcal{C} := \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) \geq 0\}. \tag{2.9}$$

Definition 2.2.1. A continuously differentiable function $h : \mathcal{X} \to \mathbb{R}$ is called a *control barrier function (CBF)* for the system (2.5) if there exists an extended class $\mathcal{K}_\infty$ function α_h such that, for all $\mathbf{x} \in \mathcal{X}$, there exists a control input $\mathbf{u} \in \mathbb{R}^m$ satisfying:

$$\dot{h}(\mathbf{x},\mathbf{u}) + \alpha_h(h(\mathbf{x})) \geq 0, \tag{2.10}$$

or equivalently,

$$\mathrm{CBC}(\mathbf{x},\mathbf{u}) := \mathcal{L}_{\mathbf{f}}h(\mathbf{x}) + \mathcal{L}_{\mathbf{G}}h(\mathbf{x})\mathbf{u} + \alpha_h(h(\mathbf{x})) \geq 0. \tag{2.11}$$

The set of control inputs that render the set $\mathcal{C}$ forward invariant is defined as:

$$K_{\mathrm{CBF}}(\mathbf{x}) = \left\{\mathbf{u} \in \mathbb{R}^m : \mathrm{CBC}(\mathbf{x},\mathbf{u}) \geq 0\right\}. \tag{2.12}$$

If h is a valid CBF, then for any initial condition $\mathbf{x}(0) \in \mathcal{C}$, the system remains safe under any feedback control law $\mathbf{u}(\mathbf{x}) \in K_{\mathrm{CBF}}(\mathbf{x})$.

2.2.2 CLF-CBF Quadratic Program

We now describe how to synthesize feedback controllers that simultaneously ensure safety and stability for the control-affine system (2.5), using a quadratic program (QP) formulation.

Suppose we are given a nominal feedback controller $\mathbf{k}(\mathbf{x})$ that is not guaranteed to satisfy safety or stability constraints. Let V and h be a control Lyapunov function (CLF) and control barrier function (CBF), respectively, defined over the domain $\mathcal{X}$. Since both the CLF and CBF constraints are affine in the control input $\mathbf{u}$, we can formulate the following QP to modify the nominal control online:

$$\begin{aligned}(\mathbf{u}(\mathbf{x}),\delta) \in \underset{\mathbf{u}\in\mathbb{R}^m,\delta\in\mathbb{R}}{\arg\min} \quad & \|L(\mathbf{x})^\top(\mathbf{u}-\mathbf{k}(\mathbf{x}))\|^2 + \lambda\delta^2 \\ \text{s.t.} \quad & \mathrm{CLC}(\mathbf{x},\mathbf{u}) \leq \delta, \quad \mathrm{CBC}(\mathbf{x},\mathbf{u}) \geq 0,\end{aligned} \tag{2.13}$$

where $L(\mathbf{x})$ is a weighting matrix that penalizes deviation from the nominal controller, and $\delta \geq 0$ is a slack variable introduced to soften the CLF constraint and ensure feasibility. The scalar $\lambda > 0$ controls the penalty for relaxing the CLF constraint.

This QP formulation yields a control input that minimally deviates from the nominal controller while ensuring safety via the CBF constraint and encouraging stability via the (softened) CLF constraint. As such, it provides a practical and scalable approach to real-time safe control synthesis for nonlinear robotic systems.

2.3 Chance Constraints and Distributionally Robust Optimization

In many robotic control problems, safety constraints depend on uncertain quantities, such as obstacle locations or sensor measurements. Let $\boldsymbol{\xi}$ denote a random vector with (unknown) distribution $\mathbb{P}^*$ supported on $\Xi \subseteq \mathbb{R}^k$. Let $G : \mathbb{R}^m \times \Xi \to \mathbb{R}$ define an inequality constraint of the form

$$G(\mathbf{u}, \boldsymbol{\xi}) \leq 0, \tag{2.14}$$

where $\mathbf{u} \in \mathbb{R}^m$ is the decision variable (e.g., the control input). A natural formulation is the *chance-constrained program*:

$$\begin{aligned} \min_{\mathbf{u} \in \mathbb{R}^m} \quad & c(\mathbf{u}) \\ \text{s.t.} \quad & \mathbb{P}^*\big(G(\mathbf{u}, \boldsymbol{\xi}) \leq 0\big) \geq 1 - \epsilon, \end{aligned} \tag{2.15}$$

where $c : \mathbb{R}^m \to \mathbb{R}$ is a convex cost (e.g., the CLF–CBF QP objective) and $\epsilon \in (0, 1)$ is the allowable violation probability. The feasible set in (2.15) is generally nonconvex. A common convex approximation replaces the chance constraint with a bound on the *conditional value-at-risk* (CVaR) [112]:

$$\text{CVaR}_{1-\epsilon}^{\mathbb{P}^*}\big(G(\mathbf{u}, \boldsymbol{\xi})\big) \leq 0. \tag{2.16}$$

For a random variable Q with distribution $\mathbb{P}_q$, CVaR at level $1 - \epsilon$ is defined as

$$\text{CVaR}_{1-\epsilon}^{\mathbb{P}_q}(Q) := \inf_{s \in \mathbb{R}} \left[\epsilon^{-1} \mathbb{E}_{\mathbb{P}_q}\left[(Q + s)_+\right] - s\right], \tag{2.17}$$

where $(\cdot)_+ := \max\{\,\cdot, 0\,\}$. This reformulation produces a convex inner approximation of the chance-constrained feasible set [126].

Because CVaR is a coherent and convex risk measure, it has been widely adopted in control and robotics to deal with uncertainty. For instance, CVaR constraints have been used in stochastic model predictive control [16, 149], risk-sensitive reinforcement learning [32, 156], and motion planning under uncertainty [55, 129]. These works demonstrate that CVaR offers a tractable and conservative surrogate to chance constraints, balancing safety and computational efficiency.

The chance-constrained formulation (2.15) requires full knowledge of the distribution $\mathbb{P}^*$, while the CVaR reformulation (2.17) only requires computing expectations under $\mathbb{P}^*$. In practice, however, the true distribution is rarely available in robotics; instead, we typically have access to a finite set of samples $\{\boldsymbol{\xi}_i\}_{i=1}^N$ (e.g., from LiDAR measurements or localization data). This limitation motivates the use of *distributionally robust optimization (DRO)*, where constraints are enforced against a family of distributions consistent with the observed samples.

2.3.1 Wasserstein Ambiguity Sets

Let $\mathbb{P}_N := \frac{1}{N}\sum_{i=1}^N \delta_{\boldsymbol{\xi}_i}$ denote the empirical distribution of the samples. The p-Wasserstein distance between $\mu, \nu \in \mathcal{P}_p(\Xi)$, the set of Borel probability measures on Ξ with finite p-th moment, is defined as:

$$W_p(\mu, \nu) := \left(\inf_{\beta \in \mathbb{Q}(\mu,\nu)} \int_{\Xi \times \Xi} \eta(\boldsymbol{\xi}, \boldsymbol{\xi}')^p \, \mathrm{d}\beta(\boldsymbol{\xi}, \boldsymbol{\xi}') \right)^{\frac{1}{p}}, \tag{2.18}$$

where $\mathbb{Q}(\mu, \nu)$ is the set of couplings of μ and ν, and η is the ground metric (we use $\eta(\boldsymbol{\xi}, \boldsymbol{\xi}') = \|\boldsymbol{\xi} - \boldsymbol{\xi}'\|_1$).

The *Wasserstein ambiguity set* of radius $r > 0$ centered at $\mathbb{P}_N$ is:

$$\mathcal{M}_N^r := \left\{ \mu \in \mathcal{P}_p(\Xi) \;\middle|\; W_p(\mu, \mathbb{P}_N) \le r \right\}. \tag{2.19}$$

Distributionally robust optimization (DRO) replaces the chance constraint in (2.15) with its distributionally robust ambiguity set $\mathcal{M}_N^r$, yielding probabilistic guarantees against all distributions within a Wasserstein ball around the empirical distribution.

2.3.2 Distributionally Robust Chance-Constrained Programs

The chance-constrained program (2.15) enforces safety in probability under the true distribution $\mathbb{P}^*$. When only samples $\{\boldsymbol{\xi}_i\}_{i=1}^N$ are available, we can replace $\mathbb{P}^*$ with the Wasserstein ambiguity set (2.19), yielding the *distributionally robust chance-constrained program* (DRCCP):

$$\begin{aligned} \min_{\mathbf{u}\in\mathbb{R}^m} \quad & c(\mathbf{u}) \\ \text{s.t.} \quad & \inf_{\mathbb{P}\in\mathcal{M}_N^r} \mathbb{P}\big(G(\mathbf{u},\boldsymbol{\xi}) \le 0\big) \ge 1-\epsilon. \end{aligned} \tag{2.20}$$

The DRCCP (2.20) guarantees constraint satisfaction for all distributions in the Wasserstein ball $\mathcal{M}_N^r$ around the empirical distribution $\mathbb{P}_N$.

As in the nominal case (2.15), the feasible set of (2.20) is generally nonconvex. A tractable inner approximation is obtained by replacing the chance constraint with the CVaR [112, 126]:

$$\sup_{\mathbb{P}\in\mathcal{M}_N^r} \text{CVaR}_{1-\epsilon}^{\mathbb{P}}\big(G(\mathbf{u},\boldsymbol{\xi})\big) \le 0. \tag{2.21}$$

2.4 Optimal Control and Reinforcement Learning

Optimal Control (OC) and Reinforcement Learning (RL) both aim to synthesize feedback policies that optimize long-term performance for a dynamical system. They share the same mathematical foundation in dynamic programming and Bellman optimality. The key distinction lies in their information assumptions: OC methods typically require an explicit system model and cost function, while RL can learn policies directly from data, enabling application in settings where the model is unknown or difficult to obtain.

2.4.1 Optimal Control

Consider the discrete-time control system:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \quad \mathbf{x}_k \in \mathcal{X} \subseteq \mathbb{R}^n, \quad \mathbf{u}_k \in \mathcal{U} \subseteq \mathbb{R}^m, \tag{2.22}$$

where $\mathcal{X}$ is open and $\mathbf{f} : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ is locally Lipschitz. Given an initial state $\mathbf{x}_0$, the infinite-horizon discounted optimal control problem is:

$$\begin{aligned} J^*_\gamma(\mathbf{x}_0) \; = \; \min_{\boldsymbol{\pi}} \quad & J^{\boldsymbol{\pi}}_\gamma(\mathbf{x}_0) := \sum_{k=0}^{\infty} \gamma^k \, \ell\big(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)\big) \\ \text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)), \\ & \mathbf{x}_k \in \mathcal{X}, \quad \boldsymbol{\pi}(\mathbf{x}_k) \in \mathcal{U}, \end{aligned} \tag{2.23}$$

where $\ell : \mathcal{X} \times \mathcal{U} \to \mathbb{R}_{\geq 0}$ is the stage cost and $\gamma \in (0, 1)$ is the discount factor. The optimal policy $\boldsymbol{\pi}^*$ minimizes the cost-to-go $J^{\boldsymbol{\pi}(\mathbf{x})}_\gamma$ for all initial states, and the associated *value function* J^*_γ satisfies the Bellman optimality equation:

$$J^*_\gamma(\mathbf{x}) = \min_{\mathbf{u} \in \mathcal{U}} \left[\ell(\mathbf{x}, \mathbf{u}) + \gamma \, J^*_\gamma\big(\mathbf{f}(\mathbf{x}, \mathbf{u})\big) \right]. \tag{2.24}$$

Classical methods such as dynamic programming, Riccati equations (for linear–quadratic problems), and model predictive control (MPC) [123] approximate the solution of (2.24) to compute near-optimal policies.

2.4.2 Reinforcement Learning

Reinforcement Learning (RL) addresses optimal control problems of the form (2.23) when the system dynamics $\mathbf{f}$ or stage cost ℓ are partially or entirely unknown, but trajectories $(\mathbf{x}_k, \mathbf{u}_k, \ell_k)$ can be obtained through interaction with the real system or a simulator. The objective is to learn a control policy $\boldsymbol{\pi}$ that maximizes the expected discounted return:

$$\max_{\boldsymbol{\pi}} \; \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \, r(\mathbf{x}_k, \mathbf{u}_k)\right], \quad r = -\ell. \tag{2.25}$$

Over the past decade, RL has achieved notable breakthroughs in continuous control and robotics, with applications spanning agile locomotion, dexterous manipulation, and high-speed drone racing.

Model-Free RL.

Model-free methods learn a policy (and often a value function) directly from sampled experience without explicitly constructing a dynamics model. These methods are typically simple to implement and resilient to modeling errors, but they often require large amounts of data. Representative examples include value-based methods such as DQN [110] and policy-gradient algorithms such as DDPG [92], PPO [131], and SAC [53], all of which have been successfully deployed in both simulation and real-world robotic systems.

Model-Based RL.

Model-based methods learn (or exploit a known) system model to improve sample efficiency and enable planning over future trajectories. They can naturally incorporate safety constraints and task specifications, but are often sensitive to model inaccuracies. Recent advances such as TD-MPC [60, 61], DreamerV2 [54], and champion-level drone racing via learned dynamics models [76] demonstrate the potential of model-based approaches for high-speed, vision-based, and complex control tasks.

Chapter 3

Robust and Probabilistic Formulation of Safe Control

Ensuring safety in autonomous systems operating in complex, dynamic, and partially unknown environments remains a fundamental challenge in robotics. Robust operation requires the integration of rich environment representations with control synthesis techniques that can provide safety guarantees under uncertainty in both perception and dynamics. Figure 3.1 illustrates a representative scenario: a ground robot equipped with a LiDAR scanner navigating safely along a desired path in an unknown indoor environment. Traditional motion planning and control strategies often assume that obstacle locations and robot dynamics are known exactly, or that the robot's geometry can be approximated by simple shapes such as points or circles. While such assumptions simplify the control design process, they severely limit applicability in realistic scenarios where environments are only partially observable, obstacle geometry can be complex, and dynamics models may be uncertain.

The development of *control barrier functions* (CBFs) [8, 9] has provided a powerful framework for encoding safety constraints in real-time control synthesis. By combining CBFs with *control Lyapunov functions* (CLFs) [13, 133], one can formulate a quadratic program (QP) that ensures both safety and stability for control-affine systems. CBF-CLF-QP methods have been successfully applied to a variety of robotic systems, including mobile robots, legged platforms, and automotive systems [113, 141, 148]. However, a key limitation in most existing work is the

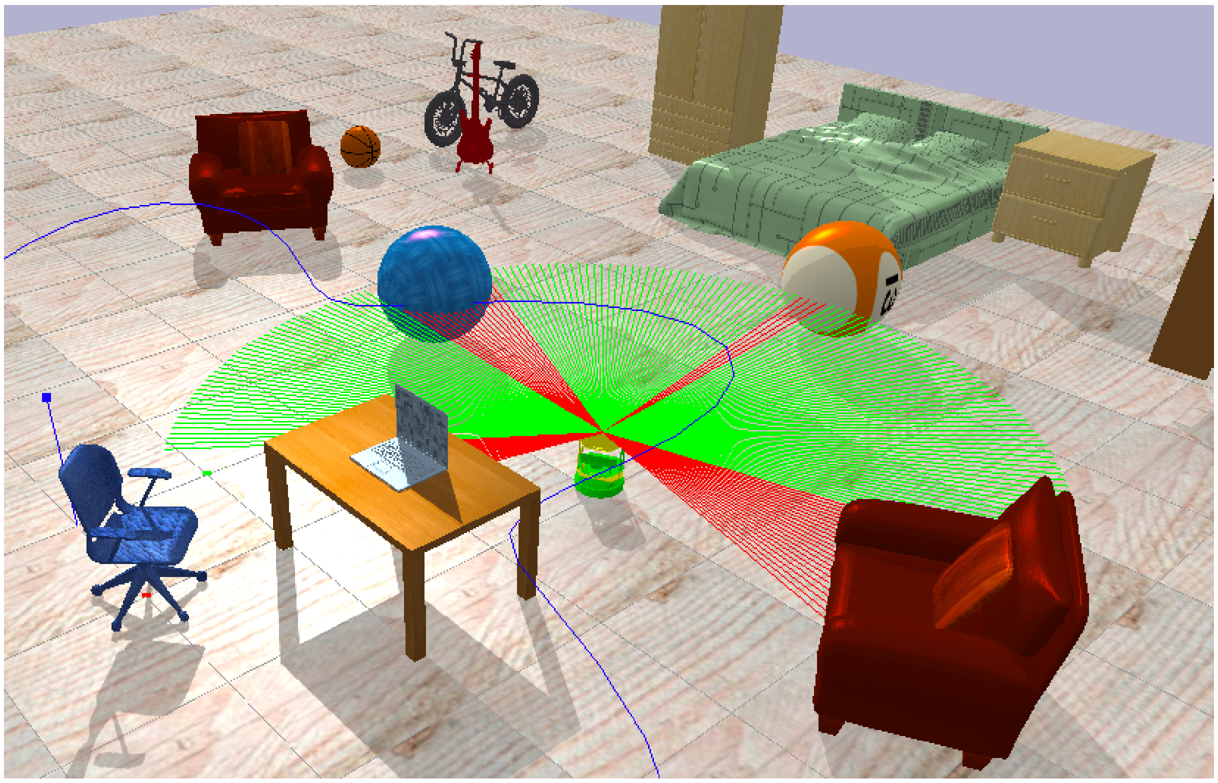


Figure 3.1. A ground robot equipped with a LiDAR scanner is navigating safely along a desired path (blue curve) in an unknown room.

reliance on *a priori* known barrier functions derived from exact knowledge of the environment and system model. When the robot operates in an unknown or changing environment, these barrier functions must be estimated online from sensor data. In this setting, both the barrier function itself and its gradient are subject to estimation errors, which can degrade safety if not explicitly accounted for in the control synthesis process.

A related challenge is the choice of environment representation for online estimation and control. Classical occupancy grid and mapping techniques, such as Octomap [70] or Voxblox [114], provide probabilistic or truncated signed distance information on a discretized spatial grid. While effective for mapping, these representations can be memory-intensive and do not naturally provide the continuous distance and gradient information required for CBF-based safety constraints. Implicit shape representations, such as signed distance functions (SDFs), offer a compelling alternative: they define the distance to an object surface at any query point, with the sign indicating inside/outside status. SDFs can be evaluated continuously and differentiated to provide exact surface normals, making them particularly suitable for integration into optimization-based control.

Recent advances in neural implicit modeling, such as DeepSDF [116] and occupancy networks [107], demonstrate that SDFs can be represented by compact neural networks trained from sparse observations. However, most of these methods are designed for offline learning of static shapes from dense datasets. For safe navigation in unknown environments, the robot must be able to construct and update SDFs online from streaming sensor measurements. Furthermore, to ensure persistent accuracy, the representation must avoid catastrophic forgetting of previously observed regions while adapting to new data.

Beyond the environment model, realistic safety constraints must also account for the geometry of both the robot and the obstacles. Point-robot and circular-robot assumptions, common in CBF-based obstacle avoidance [1, 36, 77, 154], are inadequate when the robot has a complex body shape or when obstacles cannot be well approximated by simple primitives. For example, in manipulation or cluttered navigation tasks, the robot and obstacles may be better modeled by polygons, ellipses, or unions of convex sets. Accurate distance computation between such shapes is essential for defining CBFs that are both safe and minimally conservative. While numerical optimization can be used to compute distances between general convex shapes, it is often too slow for real-time control loops, especially in dynamic environments where obstacle positions and orientations change over time.

An equally important source of complexity arises from uncertainty in the robot's dynamics model and in the estimated barrier functions. In practice, dynamics models are learned or identified from data, and may deviate from the true system due to unmodeled effects, changing payloads, or external disturbances. Similarly, barrier functions derived from noisy sensor data and learned environment models are subject to estimation error. Standard CBF-CLF-QP formulations, which assume exact knowledge of these quantities, cannot guarantee safety under such uncertainty. This motivates the development of uncertainty-aware formulations, where safety constraints are satisfied either in a probabilistic sense, given a distributional model of uncertainty, or in a robust worst-case sense, given deterministic error bounds.

This chapter addresses these challenges by presenting an integrated framework for *environment modeling* and *safe control synthesis under uncertainty*, grounded in three complementary contributions:

1. **Online neural signed distance field learning with replay memory:** We develop an incremental learning approach for continuous, differentiable SDF representations of obstacles from onboard range measurements, augmented with a replay memory to prevent forgetting of previously observed surfaces. The learned SDFs directly provide distance and gradient information for CBF construction and can adapt online to new observations.

2. **Uncertainty-aware safe control synthesis:** We extend the CBF–CLF framework to handle uncertainty in both the dynamics and the barrier functions. We present two formulations: a *probabilistic* approach using Gaussian process models and Cantelli's inequality to enforce safety with a specified risk tolerance, and a *robust* approach that enforces safety for all disturbances within known error bounds. Both lead to convex second-order cone programs (SOCPs) that are tractable in real time.

3. **Shape-aware control barrier functions for complex geometries:** We derive an analytic formula for computing the distance and gradient between a polygonal robot body and an elliptical obstacle in $\mathrm{SE}(2)$, enabling time-varying CBFs for dynamic environments with realistic robot and obstacle shapes. The analytic nature of the computation ensures both accuracy and computational efficiency.

These contributions build a unified control pipeline: first, construct a continuous and differentiable environment model from sensor data; second, translate this model into shape-aware CBF constraints; third, incorporate uncertainty into the constraints via probabilistic or robust formulations; and finally, solve a convex optimization problem to synthesize safe control inputs in real time.

By integrating learning-based environment representation, analytic shape modeling, and uncertainty-aware control synthesis, the framework presented in this chapter significantly broadens the applicability of CBF-based safe control to realistic scenarios. It enables autonomous systems to operate safely in environments that are only partially known, with dynamic obstacles and complex geometries, and under uncertainty in both sensing and dynamics. The methods have been validated in simulation for both mobile robots and articulated manipulators, demonstrating real-time feasibility and strong safety performance.

This chapter is based on the papers [98, 101, 102], which respectively address online learning of SDF-based barrier functions, robust and probabilistic CBF formulations under uncertainty, and analytic CBF design for polygonal robots in dynamic elliptical environments. Here, we present them in a unified narrative to highlight their complementarity and integration into a cohesive safe control framework.

In this chapter, we address the following problem:

Problem 1. Consider the estimated system dynamics

$$\dot{\mathbf{x}} = \tilde{\mathbf{f}}(\mathbf{x}) + \tilde{\mathbf{G}}(\mathbf{x})\mathbf{u},$$

together with an estimated barrier function $\tilde{h}(\mathbf{x})$ and its gradient $\nabla\tilde{h}(\mathbf{x})$ constructed from online sensor data. The objective is to design a feedback control law $\mathbf{u}(\mathbf{x})$ such that, for all $\mathbf{x} \in \mathcal{X}$,

$$\mathrm{CLC}(\mathbf{x}, \mathbf{u}) \leq \delta, \quad \mathrm{CBC}(\mathbf{x}, \mathbf{u}) \geq 0, \tag{3.1}$$

where CLC and CBC denote the control Lyapunov and control barrier constraints, respectively, and $\delta \geq 0$ specifies an allowable relaxation margin.

3.1 Environment Geometric Representation

Accurate and computationally efficient representations of the robot's environment are critical for safe navigation and control. The choice of representation affects not only collision checking and planning, but also the ability to reason about uncertainty in sensing and localization. In this work, we consider two complementary approaches to environment modeling: (i) learning continuous signed distance fields (SDFs) directly from sensor data, and (ii) deriving analytic distance functions for polygonal and elliptical shapes. Both approaches provide distance and gradient information that can be directly incorporated into optimization-based safe control frameworks.

3.1.1 Learning Signed Distance Fields from Sensor Data

The signed distance function (SDF) of a set Ω in a metric space measures the distance of a point $\mathbf{x}$ to the boundary $\partial\Omega$, with the sign indicating whether $\mathbf{x}$ lies inside or outside Ω. SDFs provide an implicit surface representation widely used in computer vision and graphics for surface reconstruction and rendering [93, 116]. Compared to other geometric representations, an SDF directly encodes both distance and gradient information to surfaces, which is essential for collision avoidance in autonomous navigation [58].

In this work, we approximate the SDFs of observed obstacles online and use them to define CBFs for control synthesis. The robot operates in a workspace $\mathbb{R}^w$ (with $w = 2$ or 3), which we partition into the *free space* $\mathcal{S} \subset \mathbb{R}^w$ and the *obstacle space* $\mathcal{O} = \cup_i \mathcal{O}_i$, where each $\mathcal{O}_i \subset \mathbb{R}^w$ denotes an individual obstacle. Thus, $\mathcal{S} = \mathbb{R}^w \setminus \mathcal{O}$. The robot is equipped with a range sensor (e.g., LiDAR) and follows a desired path while relying on noisy distance measurements for collision avoidance. A *path* is a piecewise-continuous function $r : [0, 1] \mapsto \text{Int}(\mathcal{S})$. At discrete times t_k, $k \in \mathbb{N}$, the sensor returns a set of points $\mathcal{P}_{k,i} = \{\mathbf{p}_{k,i,j}\}_j \subset \mathcal{F}(\mathbf{x}(t_k)) \cap \partial\mathcal{O}_i$, lying on the boundary $\partial\mathcal{O}_i$ of each obstacle $\mathcal{O}_i$ that is within the field of view $\mathcal{F}(\mathbf{x})$ of the robot at state $\mathbf{x}$.

For each obstacle i, the SDF $\varphi_i : \mathcal{Y} \to \mathbb{R}$ is defined as:

$$\varphi_i(\mathbf{y}) := \begin{cases} -d(\mathbf{y}, \partial\mathcal{O}_i), & \mathbf{y} \in \mathcal{O}_i, \\ d(\mathbf{y}, \partial\mathcal{O}_i), & \mathbf{y} \notin \mathcal{O}_i, \end{cases} \tag{3.2}$$

where d denotes the Euclidean distance between a point and a set. Our goal is to construct online approximations $\tilde{\varphi}_i$ of φ_i using the point cloud measurements $\{\mathcal{P}_{k,i}\}_k$.

Data Pre-processing

Given the point cloud $\mathcal{P}_{k,i} \subset \mathbb{R}^w$ on the surface of obstacle i at time t_k, we interpret $\mathcal{P}_{k,i}$ as samples on the zero-level set of the SDF. To normalize the data scale, we express point coordinates relative to the obstacle centroid, approximated as the sample mean:

$$\bar{\mathbf{p}}_{k,i} := \frac{1}{m} \sum_{j=1}^{m} \mathbf{p}_{k,i,j}.$$

The centered points $\mathbf{p}_{k,i,j} - \bar{\mathbf{p}}_{k,i}$ have a measured distance of 0 to $\partial v(\mathcal{O}_i)$.

To provide off-surface training points, we define a small positive constant $\delta > 0$ and construct truncated SDF points $\mathbf{q}_{k,i,j}$ along the LiDAR ray from the robot's position $v(\mathbf{x}_k)$ to $\mathbf{p}_{k,i,j}$:

$$c := \|v(\mathbf{x}_k) - \mathbf{p}_{k,i,j}\|, \quad \mathbf{q}_{k,i,j} := \frac{\delta}{c} v(\mathbf{x}_k) + \left(1 - \frac{\delta}{c}\right) \mathbf{p}_{k,i,j}.$$

Here, $\mathbf{q}_{k,i,j}$ is approximately at distance δ from $\partial v(\mathcal{O}_i)$. The training set at time t_k is:

$$\mathcal{D}_{k,i} := \{(\mathbf{p}_{k,i,j} - \bar{\mathbf{p}}_{k,i}, 0)\} \cup \{(\mathbf{q}_{k,i,j} - \bar{\mathbf{p}}_{k,i}, \delta)\}.$$

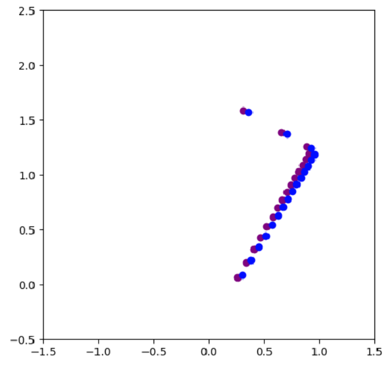
Loss Function

We approximate φ_i using a fully connected neural network $\tilde{\varphi}_i(\mathbf{y}; \boldsymbol{\theta}_k)$ with parameters $\boldsymbol{\theta}_k$. Since the true SDF satisfies the *Eikonal equation* $\|\nabla \varphi_i(\mathbf{y})\| = 1$ almost everywhere, we design a loss function combining a *distance loss* ℓ_i^D and an *Eikonal loss* ℓ_i^E:

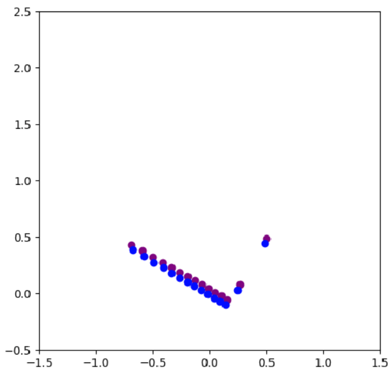
$$\begin{aligned}
\ell_i(\boldsymbol{\theta}_k; \mathcal{D}, \mathcal{D}') &:= \ell_i^D(\boldsymbol{\theta}_k; \mathcal{D}) + \lambda\, \ell_i^E(\boldsymbol{\theta}_k; \mathcal{D}'), \quad \lambda > 0, \\
\ell_i^D(\boldsymbol{\theta}_k; \mathcal{D}) &:= \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{p},d)\in\mathcal{D}} \big|\tilde{\varphi}_i(\mathbf{p}; \boldsymbol{\theta}_k) - d\big|, \\
\ell_i^E(\boldsymbol{\theta}_k; \mathcal{D}') &:= \frac{1}{|\mathcal{D}'|} \sum_{\mathbf{p}\in\mathcal{D}'} \left(\|\nabla \tilde{\varphi}_i(\mathbf{p}; \boldsymbol{\theta}_k)\| - 1\right).
\end{aligned} \tag{3.3}$$

The Eikonal training set $\mathcal{D}'$ is generated by mixing uniformly distributed points in $\mathcal{Y}$ with Gaussian samples centered at $\mathcal{D}$ points, with standard deviation equal to the distance to the k-th nearest neighbor ($k = |\mathcal{D}|/2$).

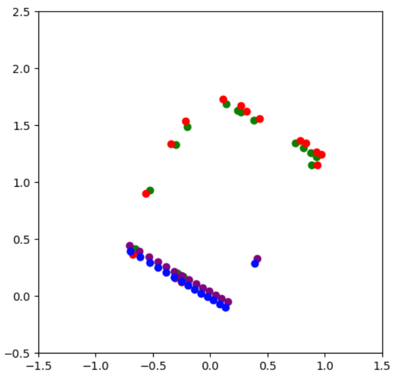
Using all accumulated data $\cup_{l=0}^{k} \mathcal{D}_{l,i}$ would yield the most accurate SDF estimates but is computationally impractical online. We address this in Sec. 3.1.1 via an incremental training strategy with replay memory.



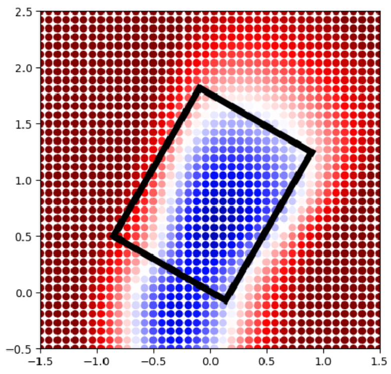
(a) Training data at $k = 0$



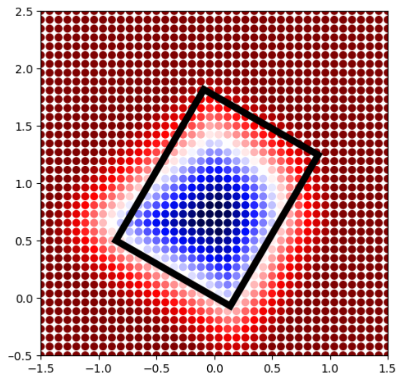
(b) IT data at $k = 70$



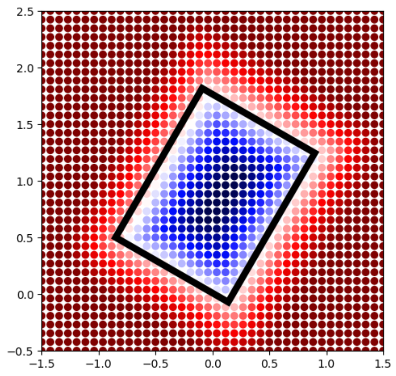
(c) ITRM data at $k = 70$



(d) Estimation at $k = 0$



(e) IT estimation at $k = 70$



(f) ITRM estimation at $k = 70$

Figure 3.2. Shape estimation with and without replay memory. The top row shows the training data used at time step $k = 0$ and $k = 70$ by the IT and ITRM approaches. The purple points are the observed LiDAR end points, while the blue points are the truncated SDF points along the LiDAR rays. In (c), the green and red points are boundary and truncated SDF points obtained from the replay memory. The bottom row shows the SDF estimation of the obstacle surface at time step $k = 0$ and $k = 70$ for the IT and ITRM approaches. The black rectangle shows the ground-truth obstacle boundary, while colored regions are level-sets of the SDF estimate. The white region denotes the estimated obstacle boundary. The blue (resp. red) region denotes negative (resp. positive) signed distance. IT and ITRM use the same data and lead to the same estimate at $k = 0$ because the replay memory set is empty. In (e), the SDF estimate of the top obstacle region at $k = 70$, without memory replay, degrades compared to (d). In (f), training with replay memory helps the neural network remember the overall obstacle shape.

We compare three online update strategies:

- Incremental Training (IT): Update $\tilde{\varphi}_i$ using only $\mathcal{D}_{k,i}$, initialized from $\boldsymbol{\theta}_{k-1}$. Fast but suffers from catastrophic forgetting.

- Batch Training (BT): Use all accumulated data $\cup_{l=0}^{k}\mathcal{D}_{l,i}$ at t_k. Avoids forgetting but training time grows with k.
- Incremental Training with Replay Memory (ITRM): Combine $\mathcal{D}_{k,i}$ with a balanced subset of previously stored replay samples.

The replay memory for an SDF φ with truncation parameter $\tau \geq 0$ is:

$$\mathcal{Q} := \{(\mathbf{q}, \varphi(\mathbf{q})) \in \mathbb{R}^w \times \mathbb{R} \mid |\varphi(\mathbf{q})| \leq \tau\}.$$

We sample level sets of $\tilde{\varphi}_i$ using Marching Cubes [106], extracting points $\mathbf{q}_0$ and $\mathbf{q}_\delta$ from the 0 and δ level sets. The replay memory at t_{k-1} is:

$$\mathcal{Q}_{k-1,i} := \{(\mathbf{q}_0, 0)\} \cup \{(\mathbf{q}_\delta, \delta)\}.$$

At t_k, ITRM trains on:

$$\mathcal{D} = \mathcal{D}_{k,i} \cup \bar{\mathcal{Q}}_{k-1,i}, \quad |\bar{\mathcal{Q}}_{k-1,i}| = |\mathcal{D}_{k,i}|.$$

This balances efficiency and accuracy, as illustrated in Fig. 3.2.

We use the Softplus activation $\ln(1 + e^x)$ to ensure $\tilde{\varphi}_i$ is continuously differentiable, enabling gradient computation via backpropagation for the Eikonal loss.

3.1.2 Analytic Distances for Polygon-Ellipse Geometry

In this section, we derive an analytic formula for computing the distance between a polygon and an ellipse, which also enables the formulation of CBFs to ensure safe autonomy.

We consider the mobile robot's body $S(\mathbf{x})$ to be described as a polygon, denoted by $\mathcal{P}(\tilde{\mathbf{q}}, \tilde{\mathbf{R}}(\tilde{\theta}), \{\tilde{\mathbf{p}}_i\}_{i=0}^{M-1})$. Here, $\tilde{\mathbf{q}}$ denotes the center of mass and $\tilde{\mathbf{R}}$ denotes the orientation in the inertial frame. In its fixed-body frame, $\{\tilde{\mathbf{p}}_i\}$ denotes the vertices of the robot with line segments $\tilde{\mathbf{d}}_i = \tilde{\mathbf{p}}_{[i+1]_M} - \tilde{\mathbf{p}}_i$ for $i = 0, 1, \ldots, M-1$ where $[\cdot]_M$ is the M-modulus.

For convenience, denote $\mathcal{E}$ and $\mathcal{P}$ as the bodies in the inertial frame, and we assume their intersection is empty. Now, denote $\mathcal{E}'$ and $\mathcal{P}'$ as the respective bodies in the body-fixed frame of the elliptical obstacle. As a result, $d(\mathcal{E}, \mathcal{P}) = d(\mathcal{E}', \mathcal{P}')$ by isometric transformation.

Furthermore, let $\tilde{\mathbf{p}}_i$ be a vertex in the robot's frame. Then in the inertial frame, it becomes $\mathbf{p}_i = \tilde{\mathbf{q}} + \tilde{\mathbf{R}}\tilde{\mathbf{p}}_i$. In the obstacle's frame, it is

$$\mathbf{p}'_i = \mathbf{R}^\top(\mathbf{p}_i - \mathbf{q}) = \mathbf{R}^\top\tilde{\mathbf{R}}\tilde{\mathbf{p}}_i + \mathbf{R}^\top(\tilde{\mathbf{q}} - \mathbf{q}), \tag{3.4}$$

In short, $\{\tilde{\mathbf{p}}_i\}$ are vertices in the robot's frame, $\{\mathbf{p}_i\}$ are vertices in the inertial frame, and $\{\mathbf{p}'_i\}$ are vertices in the obstacle's frame. The distance function is

$$d(\mathcal{E}', \mathcal{P}') := \min_{i \in [M-1]} d(\mathcal{E}', \mathbf{d}'_i), \tag{3.5}$$

which computes the distance between the ellipse $\mathcal{E}'$ and each line segment $\mathbf{d}'_i$. We write each segment as

$$l'_i(\tau) = (1 - \tau)\mathbf{p}'_i + \tau\mathbf{p}'_{[i+1]_M}, \tag{3.6}$$

for $\tau \in [0, 1]$. This further simplifies the function to

$$d(\mathcal{E}', \mathbf{d}'_i) = \min_{\tau \in [0,1]} d(\mathcal{E}', l'_i(\tau)). \tag{3.7}$$

Now, there are essentially two groups of computations for the distance in (3.7): one is the distance between the ellipse $\mathcal{E}'$ and the endpoints of $\mathbf{d}'_i$; the other is the distance between the ellipse $\mathcal{E}'$ and the infinite line $l'_i(\tau)$ for arbitrary τ with the caveat that the minimizing argument occurs at $\tau^* \in (0, 1)$. The two computations are detailed in the procedures which follow our next proposition.

Proposition 3.1.1. *Let $\mathcal{E}'$ be an ellipse and l'_i be a line segment in the frame of the ellipse. Denote*

τ^* as the argument of the minimum in (3.7). *Then, the distance*

$$d(\mathcal{E}', \mathbf{d}_i') = \begin{cases} \|\mathbf{p}_i' - \underline{\mathbf{p}_i'}\|, & \text{if } \tau^* = 0, \\ \|\mathbf{p}_{[i+1]_M}' - \underline{\mathbf{p}_{[i+1]_M}'}\|, & \text{if } \tau^* = 1, \\ \|l_i'(\tau^*) - \underline{l_i'(\tau^*)}\|, & \text{if } \tau^* \in (0,1), \end{cases} \tag{3.8}$$

*where $\underline{\mathbf{p}i'}$ and $\underline{\mathbf{p}[i+1]M'}$ are the points on the ellipse closest to $\mathbf{p}i'$ and $\mathbf{p}[i+1]M'$, respectively. These points are determined using **Procedure 1**. The terms $l_i'(\tau^*)$ and $\underline{l_i'(\tau^*)}$ (on the ellipse) are determined using **Procedure 2**.*

Procedure 1. Let $\mathbf{p}' = (p_x', p_y')$ be one of the endpoints for the line segment $\mathbf{d}_i'$. Recall that the ellipse is defined by its semi-axes along x-axis and y-axis, denoted by a and b, respectively. The points on the ellipse are parameterized by

$$x(t) = a\cos(t), \quad y(t) = b\sin(t), \tag{3.9}$$

for $0 \le t \le 2\pi$. The goal is to determine the point $(x(t), y(t))$ on the ellipse that is closest to the point $\mathbf{p}'$, so it is a minimization problem of the squared Euclidean distance:

$$d^2(t) = (p_x' - a\cos(t))^2 + (p_y' - b\sin(t))^2. \tag{3.10}$$

To find the minimum distance, we determine the critical point(s) by solving for $0 = \frac{d}{dt}d^2(t)$, which simplified to

$$0 = (b^2 - a^2)\cos t \sin t + ap_x' \sin t - bp_y' \cos t. \tag{3.11}$$

Using single-variable optimization, we substitute

$$\cos t = \lambda, \quad \sin t = \sqrt{1-\lambda}, \tag{3.12}$$

and this yields $bp'_y\lambda = \sqrt{1-\lambda^2}((b^2-a^2)\lambda + ap'_x)$, which is a quartic equation in λ. Furthermore, a monic quartic can be derived, which gives the following simplified coefficients:

$$0 = \lambda^4 + 2m\lambda^3 + (m^2 + n^2 - 1)\lambda^2 - 2m\lambda - m^2, \tag{3.13}$$

where

$$m = p'_x \frac{a}{b^2 - a^2}, \quad n = p'_y \frac{b}{b^2 - a^2}. \tag{3.14}$$

From this point, the real root(s) of the equation can be solved analytically following Cardano's and Ferrari's solution for the quartic equations [23]. Let $\underline{t}$ be the solution so that $\underline{\mathbf{p}'} = (x(\underline{t}), y(\underline{t}))$ is a point on the ellipse and is closest to $\mathbf{p}'$. Hence,

$$d(\mathcal{E}', \mathbf{d}'_i) = \|\mathbf{p}' - \underline{\mathbf{p}'}\| \tag{3.15}$$

where $\mathbf{p}'$ is either $\mathbf{p}'_i$ or $\mathbf{p}'_{[i+1]_M}$.

Procedure 2. We compute the distance between the ellipse $\mathcal{E}'$ and the infinite line $l'_i(\tau)$ whose minimizing point occurs at $\tau^* \in (0, 1)$. First, define the unit normal of the infinite line as

$$\hat{\mathbf{n}}'_i = \frac{1}{\|\mathbf{d}'_i\|}(-d'_{i,y}, d'_{i,x}). \tag{3.16}$$

Denote $\underline{l'_i(\tau^*)}$ as the point on the ellipse that is closest to the $l'_i(\tau^*)$. In fact, this point $\underline{l'_i(\tau^*)}$ must have a tangent line at the ellipse which is parallel to l'_i; which means the normal at $\underline{l'_i(\tau^*)}$ is $\pm\hat{\mathbf{n}}'_i$. Therefore, we compute the point on the ellipse up to a sign:

$$\underline{l'_i(\tau^*)} = \pm\frac{I_\epsilon^2 \hat{\mathbf{n}}'_i}{\|I_\epsilon \hat{\mathbf{n}}'_i\|}, \tag{3.17}$$

where $I_\epsilon = \text{diag}(a, b)$. The correct sign is chosen when we are looking at the sign of the constant

C in the line equation $Ax + By + C = 0$ of l_i'. In particular,

$$C = -\hat{\mathbf{n}}_i'^{\top} \mathbf{p}_i'. \tag{3.18}$$

If $C > 0$, then $\underline{l_i'(\tau^*)} = -\frac{I_\epsilon^2 \hat{\mathbf{n}}_i'}{\|I_\epsilon \hat{\mathbf{n}}_i'\|}$; otherwise, if $C < 0$, then $\underline{l_i'(\tau^*)} = \frac{I_\epsilon^2 \hat{\mathbf{n}}_i'}{\|I_\epsilon \hat{\mathbf{n}}_i'\|}$. Finally, we determine $l_i'(\tau^*)$ on the line segment $\mathbf{d}_i'$ using projection:

$$l_i'(\tau^*) = \mathbf{p}_i' + \text{proj}_{\mathbf{d}_i'}(\underline{l_i'(\tau^*)} - \mathbf{p}_i'). \tag{3.19}$$

Here, we are done with **Procedure 2**.

Next, we compute the partial derivatives of $d(\mathcal{E}', \mathcal{P}')$ with respect to either $(\mathbf{q}, \mathbf{R})$, the configuration of the obstacle, or $(\tilde{\mathbf{q}}, \tilde{\mathbf{R}})$, the configuration of the polygonal robot.

In general, both procedures above compute the distance using the Euclidean norm between two unique points: one point $\mathbf{p}'$ on a line segment of the robot, and the other $\underline{\mathbf{p}'}$ on the ellipse. This is, in fact, equivalent to the SDF of the ellipse evaluated at $\mathbf{p}'$ by the uniqueness of these two points. Therefore, let $\mathbf{p}' = l_i(\tau^*)$ for some $0 \le i < M$, then

$$d(\mathcal{E}', \mathcal{P}') = \psi_{\mathcal{E}}(\mathbf{p}') = \psi_{\mathcal{E}}(l_i(\tau^*)). \tag{3.20}$$

Then, its gradient with respect to $\mathbf{p}'$ is $\nabla \psi_{\mathcal{E}}(\mathbf{p}') = \frac{\mathbf{p}' - \underline{\mathbf{p}'}}{\|\mathbf{p}' - \underline{\mathbf{p}'}\|}$. However, note that $\mathbf{p}'$ is a point transformed from the polygonal robot's frame using (3.4), which depends on the configurations of the elliptical obstacle and the robot. Hence the partial derivatives can be computed as follows.

Proposition 3.1.2. *Let $\mathcal{E}'$ and $\mathcal{P}'$ be the elliptical obstacle and polygonal robot, respectively, in the obstacle's frame. Let $\mathbf{p}'$ and $\underline{\mathbf{p}'}$ be determined from Proposition 3.1.1, then*

$$\frac{\partial d}{\partial \mathbf{q}} = \left(\frac{\partial d}{\partial q_x}, \frac{\partial d}{\partial q_y} \right) = -\mathbf{R} \nabla \psi_{\mathcal{E}}(\mathbf{p}'), \tag{3.21}$$

$$\frac{\partial d}{\partial \mathbf{R}} = \nabla \psi_{\mathcal{E}}(\mathbf{p}') \otimes (\tilde{\mathbf{R}} \tilde{\mathbf{p}} + (\tilde{\mathbf{q}} - \mathbf{q})), \tag{3.22}$$

$$\frac{\partial d}{\partial \tilde{\mathbf{q}}} = \left(\frac{\partial d}{\partial \tilde{q}_x}, \frac{\partial d}{\partial \tilde{q}_y}\right) = \mathbf{R}\nabla\psi_{\mathcal{E}}(\mathbf{p}'), \tag{3.23}$$

$$\frac{\partial d}{\partial \tilde{\mathbf{R}}} = \mathbf{R}(\nabla\psi_{\mathcal{E}}(\mathbf{p}') \otimes \tilde{\mathbf{p}}). \tag{3.24}$$

Furthermore, (3.22) *and* (3.24) *are derivatives with respect to the rotation matrices; one may compute the derivatives with respect to the rotation angle as*

$$\begin{aligned} \frac{\partial d}{\partial \theta} &= \nabla\psi_{\mathcal{E}}(\mathbf{p}')^\top \left[\frac{\partial \mathbf{R}}{\partial \theta}^\top (\tilde{\mathbf{R}}\tilde{\mathbf{p}} + (\tilde{\mathbf{q}} - \mathbf{q}))\right] \\ &= tr\left[\frac{\partial d}{\partial \mathbf{R}}\frac{\partial \mathbf{R}}{\partial \theta}\right], \end{aligned} \tag{3.25}$$

$$\frac{\partial d}{\partial \tilde{\theta}} = \nabla\psi_{\mathcal{E}}(\mathbf{p}')^\top \left[\mathbf{R}^\top \frac{\partial \tilde{\mathbf{R}}}{\partial \tilde{\theta}}\tilde{\mathbf{p}}\right] = tr\left[\frac{\partial d}{\partial \tilde{\mathbf{R}}}\frac{\partial \tilde{\mathbf{R}}}{\partial \tilde{\theta}}^\top\right]. \tag{3.26}$$

Following both propositions above, we compute the distance function

$$\Phi(\mathbf{q}, \mathbf{R}, \tilde{\mathbf{q}}, \tilde{\mathbf{R}}) = d(\mathcal{E}, \mathcal{P}) = d(\mathcal{E}', \mathcal{P}') \tag{3.27}$$

for the elliptical obstacle $\mathcal{E}(\mathbf{q}, \mathbf{R}, a, b)$ and the polygonal robot $\mathcal{P}(\tilde{\mathbf{q}}, \tilde{\mathbf{R}}, \{\tilde{\mathbf{p}}_i\})$.

Proposition 3.1.1 shows that the ellipse–polygon distance can be computed exactly by considering only a finite set of cases: distances to the endpoints and to the line segment interior. This result not only provides a closed-form distance evaluation but also admits analytic gradients with respect to the robot and obstacle configurations, as shown in Proposition 3.1.2. These properties are critical for integrating the distance function (3.27) into control barrier functions, enabling safe robot autonomy with both geometric accuracy and computational efficiency.

Up to this point, we have assumed that the robot's geometry, state, and environment are perfectly known. In the next section, we turn to more realistic scenarios where the system dynamics and environment geometry are uncertain and must instead be estimated in real time from onboard sensors.

3.2 Safe Navigation Under Uncertainty

We focus on enforcing safety and stability for the control-affine system in (2.5) when the system dynamics $\mathbf{F}(\mathbf{x})$ and the barrier function $h(\mathbf{x})$ are *unknown* and need to be estimated from data.

As discussed in Sec. 3.1, the CBF is estimated online. Similarly, the system dynamics in (2.5) may also be uncertain and needs to be estimated. Our main goal is to develop techniques for safe and stable control synthesis with the estimated $F(\mathbf{x})$ and $h(\mathbf{x})$. We consider two scenarios, depending on whether probabilistic or worst-case error descriptions of the dynamics and barrier functions are available.

Safety and Stability with Gaussian Process Distributed System Dynamics and Barrier Function

When the system dynamics and barrier functions can be described as Gaussian Processes (GPs), we consider the following probabilistic control synthesis problem.

Problem 2 (**Safety and stability under Gaussian uncertainty**). Given an estimated distribution on the dynamics vec$(F(\mathbf{x})) \sim \mathcal{GP}(\text{vec}(\tilde{\mathbf{F}}(\mathbf{x})), K_F(\mathbf{x}, \mathbf{x}'))$ and an estimated distribution on the barrier function $h(\mathbf{x}) \sim \mathcal{GP}(\tilde{h}(\mathbf{x}), K_h(\mathbf{x}, \mathbf{x}'))$, design a feedback controller $\underline{\mathbf{k}}$ such that, for each $\mathbf{x} \in \mathcal{X}$:

$$\mathbb{P}(\text{CLC}(\mathbf{x}, \underline{\mathbf{k}}(\mathbf{x})) \leq \delta) \geq p, \quad \mathbb{P}(\text{CBC}(\mathbf{x}, \underline{\mathbf{k}}(\mathbf{x})) \geq 0) \geq p,$$

where $p \in (0, 1)$ is a user-specified risk tolerance.

Safety and Stability with Worst-Case Uncertainty in System Dynamics and Barrier Function

Many robotic systems require instead the guarantee that safety and stability hold under all possible error realizations, which motivates us to also consider the following problem.

Problem 3 (**Safety and stability under worst-case uncertainty**). Given estimated system dynamics $\tilde{\mathbf{F}}(\mathbf{x})$ with known error bound $e_F(\mathbf{x})$,

$$\|\mathbf{F}(\mathbf{x}) - \tilde{\mathbf{F}}(\mathbf{x})\| \leq e_F(\mathbf{x}),\ \forall \mathbf{x} \in \mathcal{X}, \tag{3.28}$$

and estimated barrier function $\tilde{h}(\mathbf{x})$ and gradient $\nabla \tilde{h}(\mathbf{x})$ with known error bounds $e_h(\mathbf{x})$ and $e_{\nabla h}(\mathbf{x})$, i.e., for all $\mathbf{x} \in \mathcal{X}$,

$$|h(\mathbf{x}) - \tilde{h}(\mathbf{x})| \leq e_h(\mathbf{x}),\ \|\nabla h(\mathbf{x}) - \nabla \tilde{h}(\mathbf{x})\| \leq e_{\nabla h}(\mathbf{x}), \tag{3.29}$$

design a feedback controller $\underline{\mathbf{k}}$ such that, for each $\mathbf{x} \in \mathcal{X}$:

$$\text{CLC}(\mathbf{x}, \underline{\mathbf{k}}(\mathbf{x})) \leq \delta, \quad \text{CBC}(\mathbf{x}, \underline{\mathbf{k}}(\mathbf{x})) \geq 0.$$

We rely on the estimated CBFs in Sec. 3.1 to formalize the synthesis of a controller that guarantees safety with respect to the exact obstacles, despite errors in the CBF approximation. Our analysis assumes error bounds are available, and we leave their actual computation for future work. In this regard, several recent works study the approximation power and error bounds of neural networks [14, 153].

3.2.1 Probabilistic Safe Control Formulation

This section presents our solution to Problem 2.

Inspired by the design (2.13) when the dynamics and the barrier function are known, we formulate the control synthesis problem via the following optimization problem:

$$\begin{aligned} &\min_{\underline{\mathbf{u}} \in \underline{\mathcal{U}}, \delta \in \mathbb{R}} \ \|L(\mathbf{x})^\top (\underline{\mathbf{u}} - \tilde{\underline{\mathbf{k}}}(\mathbf{x}))\|^2 + \lambda \delta^2, \\ &\text{s.t. } \mathbb{P}(\text{CLC}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta) \geq p, \quad \mathbb{P}(\text{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0) \geq p. \end{aligned} \tag{3.30}$$

The uncertainty in $\mathbf{F}$ and h affects the linearity in $\underline{\mathbf{u}}$ of the CLC and CBC conditions in the constraints of (3.30), making this optimization problem no longer a QP. Here, we justify that nevertheless the optimization can be solved efficiently. To show this, we start by analyzing the distributions of $\mathrm{CBC}(\mathbf{x}, \underline{\mathbf{u}})$ and $\mathrm{CLC}(\mathbf{x}, \underline{\mathbf{u}})$ in detail.

Proposition 3.2.1 (Mean and Variance for CBC). *Assume h is a CBF with a linear function α_h, i.e., $\alpha_h(z) = a \cdot z$ for $a \in \mathbb{R}_{\geq 0}$. Given independent distributions $h(\mathbf{x}) \sim \mathcal{GP}(\tilde{h}(\mathbf{x}), K_h(\mathbf{x}, \mathbf{x}'))$ and vec$(F(\mathbf{x})) \sim \mathcal{GP}(vec(\tilde{\mathbf{F}}(\mathbf{x})), K_F(\mathbf{x}, \mathbf{x}'))$, the mean and variance of CBC$(\mathbf{x}, \underline{\mathbf{u}})$ satisfy*

$$\mathbb{E}[\mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}})] = \mathbb{E}[\mathbf{p}(\mathbf{x})]^\top \underline{\mathbf{u}} \tag{3.31a}$$

$$\mathit{Var}[\mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}})] = \underline{\mathbf{u}}^\top \mathit{Var}[\mathbf{p}(\mathbf{x})]\underline{\mathbf{u}}, \tag{3.31b}$$

where $\mathbf{p}(\mathbf{x}) := \mathbf{F}^\top(\mathbf{x})[\nabla_{\mathbf{x}} h(\mathbf{x})] + \begin{bmatrix} ah(\mathbf{x}) & \mathbf{0}_m^\top \end{bmatrix}^\top \in \mathbb{R}^{m+1}$ and $\mathbb{E}[\mathbf{p}(\mathbf{x})]$, $\mathit{Var}[\mathbf{p}(\mathbf{x})]$ are computed in (3.37).

Proof. The control barrier condition can be written as:

$$\begin{aligned} \mathrm{CBC}(\mathbf{x}, \underline{\mathbf{u}}) &= [\nabla_{\mathbf{x}} h(\mathbf{x})]^\top \mathbf{f}(\mathbf{x}) + [\nabla_{\mathbf{x}} h(\mathbf{x})]^\top \mathbf{G}(\mathbf{x})\mathbf{u} + ah(\mathbf{x}) \\ &= \left[[\nabla_{\mathbf{x}} h(\mathbf{x})]^\top \mathbf{F}(\mathbf{x}) + \begin{bmatrix} ah(\mathbf{x}) \ \mathbf{0}_m^\top \end{bmatrix}\right] \underline{\mathbf{u}} = \mathbf{p}(\mathbf{x})^\top \underline{\mathbf{u}}. \end{aligned}$$

Note that $\nabla_{\mathbf{x}} h(\mathbf{x})$ is a GP because the gradient of a GP with differentiable mean function and twice-differentiable covariance function is also a GP, cf. [39, Lemma 6],

$$\nabla_{\mathbf{x}} h(\mathbf{x}) \sim \mathcal{GP}(\nabla_{\mathbf{x}} \tilde{h}(\mathbf{x}), \mathcal{H}_{\mathbf{x},\mathbf{x}'} K_h(\mathbf{x}, \mathbf{x}')),$$

where $\mathcal{H}_{\mathbf{x},\mathbf{x}'} K_h(\mathbf{x}, \mathbf{x}') = \left[\frac{\partial^2 K_h(\mathbf{x},\mathbf{x}')}{\partial \mathbf{x}_i, \partial \mathbf{x}'_j}\right]_{i=1,j=1}^{n,n}$ is finite for all $(\mathbf{x}, \mathbf{x}') \in \mathbb{R}^{2n}$. Since $\mathrm{vec}(\mathbf{ABC}) = (\mathbf{C}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{B})$ for appropriately sized matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, we can write

$$\begin{aligned} \mathit{Var}(F(\mathbf{x})\underline{\mathbf{u}}) &= \mathit{Var}((\underline{\mathbf{u}}^\top \otimes \mathbf{I}_n)\mathrm{vec}(\mathbf{F}(\mathbf{x}))) \\ &= (\underline{\mathbf{u}}^\top \otimes \mathbf{I}_n) K_F(\mathbf{x}, \mathbf{x})(\underline{\mathbf{u}} \otimes \mathbf{I}_n). \end{aligned} \tag{3.32}$$

For brevity, we let $K_F := K_F(\mathbf{x}, \mathbf{x}')$ and $K_h := K_h(\mathbf{x}, \mathbf{x}')$ and $\mathbf{p}_1 = \mathbf{F}^\top(\mathbf{x})[\nabla_\mathbf{x} h(\mathbf{x})]$. The term $[\nabla_\mathbf{x} h(\mathbf{x})]^\top \mathbf{F}(\mathbf{x})\underline{\mathbf{u}}$ is an inner product of two independent GPs, $\nabla_\mathbf{x} h(\mathbf{x})$ and $\mathbf{F}(\mathbf{x})\underline{\mathbf{u}}$. Thus, using [39, Lemma 5], (3.32), and that *Cov*$(\nabla_\mathbf{x} h(\mathbf{x}), \mathbf{F}(\mathbf{x})\underline{\mathbf{u}}) = 0$, $\mathbf{p}_1^\top \underline{\mathbf{u}}$ corresponds to a distribution with mean and variance:

$$\begin{aligned}
&\mathbb{E}[\mathbf{p}_1^\top \underline{\mathbf{u}}] = [\nabla_\mathbf{x} \tilde{h}(\mathbf{x})]^\top \tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}}, \\
&\text{Var}[\mathbf{p}_1^\top \underline{\mathbf{u}}] = [\nabla_\mathbf{x} \tilde{h}(\mathbf{x})]^\top (\underline{\mathbf{u}}^\top \otimes \mathbf{I}_n) K_F \\
&\qquad (\underline{\mathbf{u}} \otimes \mathbf{I}_n) \nabla_\mathbf{x} \tilde{h}(\mathbf{x}) + \underline{\mathbf{u}}^\top \tilde{\mathbf{F}}^\top(\mathbf{x}) \mathcal{H}_{\mathbf{x},\mathbf{x}'} K_h \tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}}.
\end{aligned} \tag{3.33}$$

To factorize $\underline{\mathbf{u}}$ from the variance expression, we apply the property $(\mathbf{A}\otimes\mathbf{B})(\mathbf{C}\otimes\mathbf{D}) = \mathbf{AC}\otimes\mathbf{BD}$ two times,

$$\begin{aligned}
(\underline{\mathbf{u}} \otimes \mathbf{I}_n)[\nabla_\mathbf{x} \tilde{h}(\mathbf{x})] &= (\underline{\mathbf{u}} \otimes \mathbf{I}_n)(1 \otimes [\nabla_\mathbf{x} \tilde{h}(\mathbf{x})]) \\
&= \underline{\mathbf{u}} \otimes \nabla_\mathbf{x} \tilde{h}(\mathbf{x}) = (\mathbf{I}_{m+1} \otimes \nabla_\mathbf{x} \tilde{h}(\mathbf{x}))\underline{\mathbf{u}}.
\end{aligned} \tag{3.34}$$

By substituting (3.34) in (3.33), we can factorize out $\underline{\mathbf{u}}$ to get,

$$\textit{Var}[\mathbf{p}_1] = (\mathbf{I}_{m+1} \otimes [\nabla_\mathbf{x} \tilde{h}(\mathbf{x})]^\top) K_F (\mathbf{I}_{m+1} \otimes \nabla_\mathbf{x} \tilde{h}(\mathbf{x})) + \tilde{\mathbf{F}}^\top(\mathbf{x}) \mathcal{H}_{\mathbf{x},\mathbf{x}'} K_h \tilde{\mathbf{F}}(\mathbf{x}). \tag{3.35}$$

Next, we write *Cov*$(h(\mathbf{x}), \mathbf{p}_1^\top \underline{\mathbf{u}})$ using [39, Lemma 5] and *Cov*$(h(\mathbf{x}), \mathbf{F}(\mathbf{x})\underline{\mathbf{u}}) = 0$,

$$\textit{Cov}(h(\mathbf{x}), \mathbf{p}_1^\top \underline{\mathbf{u}}) = \textit{Cov}(h(\mathbf{x}), \nabla_\mathbf{x} h(\mathbf{x}))\tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}} = \big[[\nabla_\mathbf{x} K_h]^\top \tilde{\mathbf{f}}(\mathbf{x}) \quad [\nabla_\mathbf{x} K_h]^\top \tilde{\mathbf{G}}(\mathbf{x})\big]\underline{\mathbf{u}}. \tag{3.36}$$

Using (3.33), (3.35) and (3.36), we write the mean and variance,

$$\begin{aligned}
&\mathbb{E}[\mathbf{p}(\mathbf{x})] = [\nabla_\mathbf{x} \tilde{h}(\mathbf{x})]^\top \tilde{\mathbf{F}}(\mathbf{x}) + a[\tilde{h}(\mathbf{x}) \quad \mathbf{0}_m^\top]^\top \\
&\textit{Var}[\mathbf{p}(\mathbf{x})] = \tilde{\mathbf{F}}^\top(\mathbf{x}) \mathcal{H}_{\mathbf{x},\mathbf{x}'} K_h \tilde{\mathbf{F}}(\mathbf{x}) + (\mathbf{I}_{m+1} \otimes \nabla_\mathbf{x} \tilde{h}(\mathbf{x})^\top) K_F (\mathbf{I}_{m+1} \otimes \nabla_\mathbf{x} \tilde{h}(\mathbf{x})) \\
&\quad + \begin{bmatrix} a^2 K_h + 2a[\nabla_\mathbf{x} K_h]^\top \tilde{\mathbf{f}}(\mathbf{x}) & a[\nabla_\mathbf{x} K_h]^\top \tilde{\mathbf{G}}(\mathbf{x}) \\ a\tilde{\mathbf{G}}(\mathbf{x})^\top [\nabla_\mathbf{x} K_h] & \mathbf{0}_{m\times m} \end{bmatrix},
\end{aligned} \tag{3.37}$$

from which the statement follows. □

Proposition 3.2.1 makes two points explicit. First, under independent GPs for h and F, the CBC is *affine* in $\underline{\mathbf{u}}$ in expectation and *quadratic* in $\underline{\mathbf{u}}$ in variance; equivalently, uncertainty enters as an ellipsoidal form $\underline{\mathbf{u}}^\top \mathit{Var}[\mathbf{p}(\mathbf{x})]\underline{\mathbf{u}}$. Second, the vector $\mathbf{p}(\mathbf{x})$ bundles all uncertainty channels that influence safety—geometry via h and dynamics via F—so that correlated or high-variance components directly shrink the feasible control set. Importantly, the result yields closed-form *sensitivities* of safety to $\underline{\mathbf{u}}$, which we exploit for a convex risk-aware reformulation later.

Next, we describe the distribution of $\mathrm{CLC}(\mathbf{x}, \underline{\mathbf{u}})$.

Proposition 3.2.2 (Gaussian distribution for CLC). *Given the distribution vec*$(F(\mathbf{x})) \sim \mathcal{GP}(\mathit{vec}(\tilde{\mathbf{F}}(\mathbf{x})), K_F(\mathbf{x}, \mathbf{x}'))$*, the CLC*$(\mathbf{x}, \underline{\mathbf{u}})$ *is Gaussian with mean and variance:*

$$\mathbb{E}[\mathit{CLC}(\mathbf{x}, \underline{\mathbf{u}})] = \mathbb{E}[\mathbf{q}(\mathbf{x})]^\top \underline{\mathbf{u}} \tag{3.38a}$$

$$\mathit{Var}[\mathit{CLC}(\mathbf{x}, \underline{\mathbf{u}})] = \underline{\mathbf{u}}^\top \mathit{Var}[\mathbf{q}(\mathbf{x})]\underline{\mathbf{u}}, \tag{3.38b}$$

where $\mathbf{q}(\mathbf{x}) := \mathbf{F}^\top(\mathbf{x})[\nabla_\mathbf{x} V(\mathbf{x})] + [\alpha_V(V(\mathbf{x})) \quad \mathbf{0}_m^\top]^\top \in \mathbb{R}^{m+1}$ *and* $\mathbb{E}[\mathbf{q}(\mathbf{x})]$, $\mathit{Var}[\mathbf{q}(\mathbf{x})]$ *are computed in* (3.39).

Proof. We can write the control Lyapunov condition as $\mathrm{CLC}(\mathbf{x}, \underline{\mathbf{u}}) = [\nabla_\mathbf{x} V(\mathbf{x})]^\top \mathbf{F}(\mathbf{x})\underline{\mathbf{u}} + \alpha_V(V(\mathbf{x})) = \mathbf{q}^\top(\mathbf{x})\underline{\mathbf{u}}$. We use the Kronecker product property $\mathrm{vec}(\mathbf{ABC}) = (\mathbf{C}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{B})$ to rewrite first term in $\mathbf{q}(\mathbf{x})$ as:

$$[\nabla_\mathbf{x} V(\mathbf{x})]^\top \mathbf{F}(\mathbf{x}) = (\mathbf{I}_{m+1} \otimes [\nabla_\mathbf{x} V(\mathbf{x})]^\top)\mathrm{vec}(F(\mathbf{x})).$$

Since $\mathrm{vec}(\mathbf{F}(\mathbf{x})) \sim \mathcal{GP}(\mathrm{vec}(\tilde{\mathbf{F}}(\mathbf{x})), K_F(\mathbf{x}, \mathbf{x}'))$ and $[\nabla_\mathbf{x} V(\mathbf{x})]$, $\alpha_V(V(\mathbf{x}))$ are known and deterministic and , we can express the distribution of $\mathbf{q}(\mathbf{x})$ as follows:

$$\mathbb{E}[\mathbf{q}(\mathbf{x})] = \tilde{\mathbf{F}}^\top(\mathbf{x})[\nabla_\mathbf{x} V(\mathbf{x})] + [\alpha_V(V(\mathbf{x})) \quad \mathbf{0}_m^\top]^\top \tag{3.39}$$

$$\text{Var}[\mathbf{q}(\mathbf{x})] = (\mathbf{I}_{m+1} \otimes [\nabla_{\mathbf{x}} V(\mathbf{x})]^\top) K_F (\mathbf{I}_{m+1} \otimes [\nabla_{\mathbf{x}} V(\mathbf{x})]).$$

The result follows from plugging (3.39) into CLC$(\mathbf{x}, \underline{\mathbf{u}})$. □

The CLC inherits the same affine/ellipsoidal structure in $\underline{\mathbf{u}}$, with uncertainty driven solely by $\mathbf{F}$ because V is known.

Next, we use the mean and variance of CBC$(\mathbf{x}, \underline{\mathbf{u}})$ and CLC$(\mathbf{x}, \underline{\mathbf{u}})$ obtained above to approximate the probabilistic safety and stability constraints in (3.30). To make (3.30) tractable, we seek convex reformulations that can be solved efficiently in real time. The following result shows that, using Cantelli's inequality, the probabilistic constraints can be converted into second-order cone (SOC) constraints, resulting in a standard SOCP formulation.

Proposition 3.2.3 (Probabilistic CLF-CBF SOCP). *Given a user-specified risk tolerance $p \in [0, 1)$, let $c(p) = \sqrt{\frac{p}{1-p}}$. The optimization problem* (3.30) *can be formulated as the following second-order cone program:*

$$\begin{aligned}
&\min_{\underline{\mathbf{u}} \in \underline{\mathcal{U}}, \delta \in \mathbb{R}, l \in \mathbb{R}} \; l \\
&\text{s.t. } \delta - \mathbb{E}[\mathbf{q}(\mathbf{x})]^\top \underline{\mathbf{u}} \geq c(p) \sqrt{\underline{\mathbf{u}}^\top \mathit{Var}[\mathbf{q}(\mathbf{x})] \underline{\mathbf{u}}}, \\
&\qquad \mathbb{E}[\mathbf{p}(\mathbf{x})]^\top \underline{\mathbf{u}} \geq c(p) \sqrt{\underline{\mathbf{u}}^\top \mathit{Var}[\mathbf{p}(\mathbf{x})] \underline{\mathbf{u}}}, \\
&l + 1 \geq \sqrt{\|2L(\mathbf{x})^\top (\underline{\mathbf{u}} - \tilde{\underline{\mathbf{k}}}(\mathbf{x}))\|^2 + (2\sqrt{\lambda}\delta)^2 + (l-1)^2}
\end{aligned} \tag{3.40}$$

where $\mathbf{p}$, $\mathbf{q}$ *are defined in Propositions 3.2.1 and 3.2.2, resp.*

Proof. To deal with the probabilistic constraints in (3.30), we employ Cantelli's inequality [22]. For any scalar $\gamma \geq 0$,

$$\mathbb{P}(\text{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq \mathbb{E}[\text{CBC}(\mathbf{x}, \underline{\mathbf{u}}))] - \gamma | \mathbf{x}, \underline{\mathbf{u}}) \geq 1 - \frac{\text{Var}[\text{CBC}(\mathbf{x}, \underline{\mathbf{u}})]}{\text{Var}[\text{CBC}(\mathbf{x}, \underline{\mathbf{u}})] + \gamma^2}.$$

Given this inequality, and since we want $\mathbb{P}(\text{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0) \geq p$, we choose $\gamma = \mathbb{E}[\text{CBC}(\mathbf{x}, \underline{\mathbf{u}})]$ and require the lower bound to be greater than or equal to p, i.e., $1 - \frac{\text{Var}[\text{CBC}(\mathbf{x},\underline{\mathbf{u}})]}{\text{Var}[\text{CBC}(\mathbf{x},\underline{\mathbf{u}})]+\gamma^2} \geq p$. The equation can be rearranged into

$$\mathbb{E}[\text{CBC}(\mathbf{x}, \underline{\mathbf{u}})] = \gamma \geq \sqrt{\frac{p}{1-p}\text{Var}[\text{CBC}(\mathbf{x}, \underline{\mathbf{u}})]},$$

which corresponds to the safety constraint in (3.40).

Next, we show that this is a second-order cone (SOC) constraint. By (3.31), given that $\tilde{h}$, $\nabla\tilde{h}$ and $\tilde{\mathbf{F}}$ are known and deterministic, the expectation $\mathbb{E}[\text{CBC}(\mathbf{x}, \underline{\mathbf{u}})] = \mathbb{E}[\mathbf{p}(\mathbf{x})]^\top \underline{\mathbf{u}}$ is affine in $\underline{\mathbf{u}}$. Since $\text{Var}[\mathbf{p}(\mathbf{x})]$ is positive semi-definite,

$$\sqrt{\text{Var}[\text{CBC}(\mathbf{x}, \underline{\mathbf{u}})]} = \sqrt{\underline{\mathbf{u}}^\top \text{Var}[\mathbf{p}(\mathbf{x})]\underline{\mathbf{u}}} = \|\mathbf{D}(\mathbf{x})\underline{\mathbf{u}}\| \tag{3.41}$$

where $\mathbf{D}(\mathbf{x})^\top\mathbf{D}(\mathbf{x}) = \text{Var}[\mathbf{p}(\mathbf{x})]$. Acccording to [5], the safety constraint in (3.40) is a valid SOC constraint.

For stability, the CLC condition can be constructed using a similar approach with Cantelli's inequality, resulting in (3.40). By (3.38), we know that the expectation is affine in $\underline{\mathbf{u}}$ and the variance is quadratic in terms of $\underline{\mathbf{u}}$, similar to (3.41). This shows that the CLC condition is also a valid SOC constraint.

Our last step is to reformulate the minimization of the objective function as a linear objective with an SOC constraint, resulting in the standard SOCP in (3.40). We introduce a new variable l so that the problem in (3.30) is equivalent to

$$\begin{aligned} &\min_{\underline{\mathbf{u}} \in \underline{\mathcal{U}}, \delta \in \mathbb{R}, l \in \mathbb{R}} \quad l \\ &\text{s.t. } \mathbb{P}(\text{CLC}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta) \geq p, \quad \mathbb{P}(\text{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0) \geq p, \\ &\quad \|L(\mathbf{x})^\top(\underline{\mathbf{u}} - \underline{\tilde{\mathbf{k}}}(\mathbf{x}))\|^2 + \lambda\delta^2 \leq l. \end{aligned} \tag{3.42}$$

The last constraint in (3.42) corresponds to a rotated second-order cone, $\mathcal{Q}^n_{rot} := \{(\mathbf{x}_r, y_r, z_r) \in \mathbb{R}^{n+2} \mid \|\mathbf{x}_r\|^2 \leq y_r z_r, y_r \geq 0, z_r \geq 0\}$, which can be converted into a standard SOC constraint [5], $\left\|\begin{bmatrix} 2\mathbf{x}_r & y_r - z_r \end{bmatrix}^\top\right\| \leq y_r + z_r$. Let $y_r = l$, $z_r = 1$ and consider the constraint $\|L(\mathbf{x})^\top(\underline{\mathbf{u}} - \tilde{\underline{\mathbf{k}}}(\mathbf{x}))\|^2 + \lambda\delta^2 \leq l$. Multiplying both sides by 4 and adding $(l-1)^2$, makes the constraint equivalent to

$$4\|L(\mathbf{x})^\top(\underline{\mathbf{u}} - \tilde{\underline{\mathbf{k}}}(\mathbf{x}))\|^2 + 4\lambda\delta^2 + (l-1)^2 \leq (l+1)^2.$$

Taking a square root on both sides, we end up with

$$\sqrt{\|2L(\mathbf{x})^\top(\underline{\mathbf{u}} - \tilde{\underline{\mathbf{k}}}(\mathbf{x}))\|^2 + (2\sqrt{\lambda}\,\delta)^2 + (l-1)^2} \;\leq\; l+1,$$

which is equivalent to the third constraint in (3.40). □

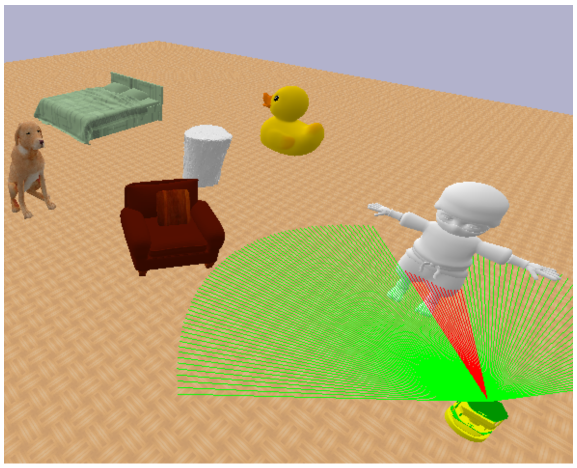
Remark 3.2.4 (Effects of risk-tolerance p and variance). *When $p = 0$, the probabilistic CLF-CBF-SOCP* (3.40) *reduces to the original CLF-CBF-QP* (2.13). *As p and/or Var$[\mathbf{p}(\mathbf{x})]$, Var$[\mathbf{q}(\mathbf{x})]$ increase, the feasible region of* (3.40) *gets smaller, and the optimal value worsens, cf. Fig. 3.3b for an illustration.*

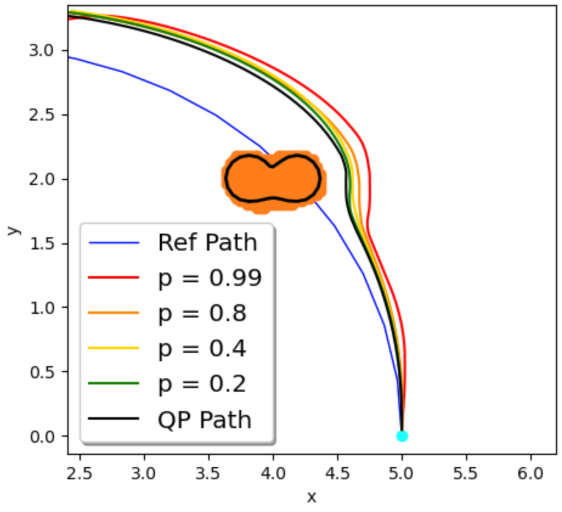
Proposition 3.2.3 establishes that probabilistic CLF-CBF constraints can be enforced within a convex optimization framework. This is significant for two reasons. First, it shows that risk-aware safety and stability constraints can be handled in real time using off-the-shelf SOCP solvers. Second, the reformulation makes explicit the trade-off between the risk-tolerance parameter p and the variance of the uncertainty: higher safety probability requires larger margins, shrinking the feasible control set (see Remark 3.2.4).

3.2.2 Robust Safe Control Formulation

In this section, we develop a solution to Problem 3.

Let $\tilde{\mathbf{F}}$ denote the estimated system dynamics, $\tilde{h}$, $\nabla\tilde{h}$ the estimated barrier function and its gradient, and let $e_F : \mathbb{R}^{n\times(m+1)} \mapsto \mathbb{R}_{\geq 0}$, $e_h : \mathbb{R} \mapsto \mathbb{R}_{\geq 0}$, and $e_{\nabla h} : \mathbb{R}^n \mapsto \mathbb{R}_{\geq 0}$ be


(a) Pybullet Simulator



(b) Probabilistic Trajectory

Figure 3.3. (a) is the Pybullet simulation environment where we conduct our experiments. (b) shows the results in a region of an environment, where the probabilistic ($p = 0.2, 0.4, 0.8, 0.99$) controller and QP controller both succeed. The ground-truth obstacle surface is shown in black while the estimated obstacles is shown in orange.

associated error bounds. For convenience, for each $\mathbf{x} \in \mathcal{X}$, we denote $D_F(\mathbf{x}) := \mathbf{F}(\mathbf{x}) - \tilde{\mathbf{F}}(\mathbf{x})$, $d_h(\mathbf{x}) := h(\mathbf{x}) - \tilde{h}(\mathbf{x})$ and $\mathbf{d}_{\nabla h}(\mathbf{x}) := \nabla h(\mathbf{x}) - \nabla \tilde{h}(\mathbf{x})$.

By (3.28) and (3.29), we have

$$\|D_F(\mathbf{x})\| \leq e_F(\mathbf{x}), \ |d_h(\mathbf{x})| \leq e_h(\mathbf{x}), \ \|\mathbf{d}_{\nabla h}(\mathbf{x})\| \leq e_{\nabla h}(\mathbf{x}). \tag{3.43}$$

Using this notation, we can rewrite $\mathrm{CBC}(\mathbf{x}, \underline{\mathbf{u}})$ as

$$\begin{aligned}\mathrm{CBC}(\mathbf{x}, \underline{\mathbf{u}}) &= [\nabla h(\mathbf{x})]^\top \mathbf{F}(\mathbf{x})\underline{\mathbf{u}} + \alpha_h(h(\mathbf{x})) \\ &= [\nabla \tilde{h}(\mathbf{x})]^\top \tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}} + \mathbf{d}_{\nabla h}^\top(\mathbf{x})\tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}} + [\nabla \tilde{h}(\mathbf{x})]^\top D_F(\mathbf{x})\underline{\mathbf{u}} \\ &\quad + \mathbf{d}_{\nabla h}^\top(\mathbf{x}) D_F(\mathbf{x})\underline{\mathbf{u}} + \alpha_h(\tilde{h}(\mathbf{x}) + d_h(\mathbf{x})).\end{aligned}$$

Let $\tilde{\mathbf{p}}(\mathbf{x}) := \tilde{\mathbf{F}}^\top(\mathbf{x})\nabla \tilde{h}(\mathbf{x})$. We group the error term in the expression for $\mathrm{CBC}(\mathbf{x}, \underline{\mathbf{u}})$ in the variable $d_{\mathrm{CBC}}(\mathbf{x}, \underline{\mathbf{u}}) := \mathrm{CBC}(\mathbf{x}, \underline{\mathbf{u}}) - \tilde{\mathbf{p}}(\mathbf{x})^\top \underline{\mathbf{u}}$.

Thus, $\mathrm{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0$ is satisfied if

$$\min_{D_F, \mathbf{d}_{\nabla h}, d_h} \mathrm{CBC}(\mathbf{x}, \underline{\mathbf{u}}) = \tilde{\mathbf{p}}(\mathbf{x})^\top \underline{\mathbf{u}} + \min_{D_F, \mathbf{d}_{\nabla h}, d_h} d_{\mathrm{CBC}}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0.$$

Let $\tilde{\mathbf{q}}(\mathbf{x}) := \tilde{\mathbf{F}}^\top(\mathbf{x}) \nabla V(\mathbf{x}) + [\alpha_V(V(\mathbf{x})) \quad \mathbf{0}_m^\top]^\top$ and $d_{\mathrm{CLC}}(\mathbf{x}, \underline{\mathbf{u}}) := [\nabla V(\mathbf{x})]^\top D_F(\mathbf{x}) \underline{\mathbf{u}}$, a robust version of the stability constraint $\mathrm{CLC}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta$ can be written as:

$$\max_{D_F} \mathrm{CLC}(\mathbf{x}, \underline{\mathbf{u}}) = \tilde{\mathbf{q}}(\mathbf{x})^\top \underline{\mathbf{u}} + \max_{D_F} d_{\mathrm{CLC}}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta. \tag{3.44}$$

This leads us to the following robust reformulation of the original control synthesis problem in (2.13),

$$\begin{aligned} &\min_{\underline{\mathbf{u}} \in \underline{\mathcal{U}}, \delta \in \mathbb{R}, l \in \mathbb{R}} \quad l \\ \text{s.t. } &\tilde{\mathbf{q}}(\mathbf{x})^\top \underline{\mathbf{u}} + \max_{D_F} \ d_{\mathrm{CLC}}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta \\ &\tilde{\mathbf{p}}(\mathbf{x})^\top \underline{\mathbf{u}} + \min_{D_F, d_h, \mathbf{d}_{\nabla h}} d_{\mathrm{CBC}}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0 \\ &l + 1 \geq \sqrt{\|2L(\mathbf{x})^\top (\underline{\mathbf{u}} - \underline{\tilde{\mathbf{k}}}(\mathbf{x}))\|^2 + (2\sqrt{\lambda}\delta)^2 + (l-1)^2}. \end{aligned} \tag{3.45}$$

Note that we used the same approach as in the proof of Proposition 3.2.3 to reformulate the original quadratic objective with a linear objective plus a SOC constraint. The second constraint in (3.45) requires solving $\min_{D_F, d_h, \mathbf{d}_{\nabla h}} d_{\mathrm{CBC}}(\mathbf{x}, \underline{\mathbf{u}})$ subject to (3.43). In general, this is a non-convex constrained quadratic program which does not have a closed-form expression of the minimizer as a function of $\underline{\mathbf{u}}$. Instead, we make the second constraint in (3.45) more conservative using the Cauchy-Schwarz inequality, which leads to a convex SOCP, whose optimal solution is guaranteed to be feasible for (3.45).

Proposition 3.2.5 (Robust CLF-CBF SOCP). *Let $\tilde{\mathbf{F}}$, $\tilde{h}$, $\nabla \tilde{h}$ denote estimates of the system dynamics and barrier function, with error bounds in* (3.43). *Then, the feasible set of the following*

SOCP is included in the feasible set of (3.45)*:*

$$\begin{aligned}
&\min_{\underline{\mathbf{u}}\in\underline{\mathcal{U}},\delta\in\mathbb{R},p\in\mathbb{R},q\in\mathbb{R},l\in\mathbb{R}} \; l \\
\text{s.t. } & \delta - \tilde{\mathbf{q}}(\mathbf{x})^\top\underline{\mathbf{u}} \geq e_F(\mathbf{x})\|\nabla V(\mathbf{x})\|\|\underline{\mathbf{u}}\|, \\
& p \geq e_{\nabla h}(\mathbf{x})\|\tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}}\|, \\
& q \geq \Big(e_F(\mathbf{x})\|\nabla\tilde{h}(\mathbf{x})\| + e_{\nabla h}(\mathbf{x})e_F(\mathbf{x})\Big)\|\underline{\mathbf{u}}\|, \\
& [\nabla\tilde{h}(\mathbf{x})]^\top\tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}} + \alpha_h(\tilde{h}(\mathbf{x}) - e_h(\mathbf{x})) \geq p + q, \\
& l + 1 \geq \sqrt{\|2L(\mathbf{x})^\top(\underline{\mathbf{u}} - \underline{\tilde{\mathbf{k}}}(\mathbf{x}))\|^2 + (2\sqrt{\lambda}\delta)^2 + (l-1)^2}
\end{aligned} \tag{3.46}$$

Proof. The stability constraint in (3.45) is reformulated using:

$$\max_{\|D_F(\mathbf{x})\|\leq e_F(\mathbf{x})} d_{\text{CLC}}(\mathbf{x},\underline{\mathbf{u}}) = e_F(\mathbf{x})\|\nabla V(\mathbf{x})\|\|\underline{\mathbf{u}}\|.$$

For the safety constraint in (3.45), note that

$$\begin{aligned}
\min_{D_F,d_h,\mathbf{d}_{\nabla h}} d_{\text{CBC}}(\mathbf{x},\underline{\mathbf{u}}) = \min_{D_F,\mathbf{d}_{\nabla h}} \Big(&\mathbf{d}_{\nabla h}^\top(\mathbf{x})\tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}} + [\nabla\tilde{h}(\mathbf{x})]^\top D_F(\mathbf{x})\underline{\mathbf{u}} + \\
&\mathbf{d}_{\nabla h}^\top(\mathbf{x})D_F(\mathbf{x})\underline{\mathbf{u}}\Big) + \min_{d_h}\alpha_h(\tilde{h}(\mathbf{x}) + d_h(\mathbf{x})).
\end{aligned} \tag{3.47}$$

Since $e_h(\mathbf{x}) \geq 0$ and α_h is an extended class $\mathcal{K}_\infty$ function,

$$\min_{|d_h(\mathbf{x})|\leq e_h(\mathbf{x})} \alpha_h(\tilde{h}(\mathbf{x}) + d_h(\mathbf{x})) = \alpha_h(\tilde{h}(\mathbf{x}) - e_h(\mathbf{x})). \tag{3.48}$$

Applying the Cauchy-Schwarz inequality on each term,

$$\begin{aligned}
\min_{D_F,d_h,\mathbf{d}_{\nabla h}} d_{\text{CBC}}(\mathbf{x},\underline{\mathbf{u}}) \geq& -\|\mathbf{d}_{\nabla h}\|\|\tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}}\| - \|\nabla\tilde{h}(\mathbf{x})\|\|D_F(\mathbf{x})\underline{\mathbf{u}}\| \\
&- \|\mathbf{d}_{\nabla h}(\mathbf{x})\|\|D_F(\mathbf{x})\underline{\mathbf{u}}\| + \alpha_h(\tilde{h}(\mathbf{x}) - e_h(\mathbf{x}))
\end{aligned}$$

$$\geq -e_{\nabla h}(\mathbf{x})\|\tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}}\| - e_F(\mathbf{x})\|\nabla\tilde{h}(\mathbf{x})\|\|\underline{\mathbf{u}}\| - e_{\nabla h}(\mathbf{x})e_F(\mathbf{x})\|\underline{\mathbf{u}}\| + \alpha_h(\tilde{h}(\mathbf{x}) - e_h(\mathbf{x})).$$

In the last step, we minimized each term independently, so the lower bound is not tight. We write the safety constraint as

$$\begin{aligned} &e_{\nabla h}(\mathbf{x})\|\tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}}\| + (e_F(\mathbf{x})\|\nabla\tilde{h}(\mathbf{x})\| + e_{\nabla h}(\mathbf{x})e_F(\mathbf{x}))\|\underline{\mathbf{u}}\| \\ &\leq [\nabla\tilde{h}(\mathbf{x})]^\top\tilde{\mathbf{F}}(\mathbf{x})\underline{\mathbf{u}} + \alpha_h(\tilde{h}(\mathbf{x}) - e_h(\mathbf{x})). \end{aligned} \tag{3.49}$$

Constraints of the form $\|\mathbf{A}\mathbf{z} - \boldsymbol{a}\| + \|\mathbf{B}\mathbf{z} - \boldsymbol{b}\| \leq \boldsymbol{c}^\top\mathbf{z}$ can be replaced by the set of constraints $\|\mathbf{A}\mathbf{z} - \boldsymbol{a}\| \leq p$, $\|\mathbf{B}\mathbf{z} - \boldsymbol{b}\| \leq q$, $p + q \leq \boldsymbol{c}^\top\mathbf{z}$ combined. Thus, (3.49) is equivalent to the second, third, and fourth constraints in (3.46) together. $\square$

Proposition 3.2.5 shows that, despite the apparent nonconvexity of the robust reformulation (3.45), the constraints can be upper bounded by convex inequalities, yielding a tractable SOCP. Intuitively, the worst-case effect of dynamics error D_F, barrier value error d_h, and gradient error $\mathbf{d}_{\nabla h}$ can all be captured by linear terms in $\underline{\mathbf{u}}$ scaled by their respective error bounds. The resulting constraints enforce a safety margin that grows proportionally with the magnitudes of e_F, e_h, and $e_{\nabla h}$, effectively shrinking the feasible control set. This makes the controller conservative but guarantees safety and stability under any admissible error realization.

Remark 3.2.6 (Effects of error bounds). *If there are no errors in either the dynamics or the barrier function ($e_F \equiv e_h \equiv e_{\nabla h} \equiv 0$), then the robust CLF-CBF SOCP* (3.46) *reduces to a CLF-CBF QP* (2.13). *If $e_F \equiv 0$ while $e_h(\mathbf{x}), e_{\nabla h}(\mathbf{x}) > 0$, the result in Proposition 3.2.5 recovers [101, Proposition 2]. As the error bounds $e_F, e_h, e_{\nabla h}$ increase, the feasible region of* (3.46) *gets smaller and the optimal solution worsens. Also, note that the choice of kernel function, $K_F(\mathbf{x}, \mathbf{x}) = \frac{e_F^2(\mathbf{x})}{c^2(p)}\mathbf{I}_{(m+1)n}$, reduces the inequality for stability in* (3.40) *to that in* (3.46).

Compared to Proposition 3.2.3, which leverages variance information to trade risk against performance, Proposition 3.2.5 removes this trade-off by requiring satisfaction for *all* admissible

Table 3.1. Empirical SDF estimation error $\mathcal{E}$ and dropout-network SDF estimation error averaged across 8 object instances under different LiDAR measurement noise standard deviation σ.

LiDAR Noise σ	SDF Empirical Error	SDF Dropout Error
0.01	0.0173	0.0132
0.02	0.0288	0.0184
0.05	0.0463	0.0242

errors. As a result, the robust SOCP is particularly suited to safety-critical scenarios where distributional assumptions are unreliable, but conservative error bounds are available. In practice, the choice between probabilistic and robust formulations depends on whether one prefers less conservatism with quantified risk or strict guarantees at the cost of performance.

3.3 Evaluation

In this section, we evaluate two key components: (1) the performance of our online CBF construction, using either a neural CBF learned from LiDAR data (Sec. 3.1) or an analytic CBF for ellipse–polygon environments (Sec. 3.1.2); and (2) the effectiveness of safe control synthesis under uncertainty in autonomous navigation and 2D manipulator tasks. All experiments are conducted in simulated environments containing obstacles that are *a priori* unknown to the robot.

3.3.1 Online CBF Estimation

The robot is equipped with a LiDAR scanner with a $270°$ field of view, 200 rays per scan, 3 meter range, and zero-mean Gaussian measurement noise with standard deviation $\sigma \in \{0.01, 0.02, 0.05\}$. The LiDAR scans are used to estimate the unsafe regions $\mathcal{O}_i$ in the environment and construct a CBF constraint for each. We rely on the concept of signed distance function (SDF) (e.g. Fig. 3.4b) to describe each $\mathcal{O}_i$. The SDF function $\varphi_i : \mathbb{R}^2 \mapsto \mathbb{R}$ of set $\mathcal{O}_i \subseteq \mathbb{R}^2$ is

$$\varphi_i(\mathbf{y}) := \begin{cases} -d(\mathbf{y}, \partial\mathcal{O}_i), & \mathbf{y} \in \mathcal{O}_i, \\ d(\mathbf{y}, \partial\mathcal{O}_i), & \mathbf{y} \notin \mathcal{O}_i, \end{cases} \tag{3.50}$$

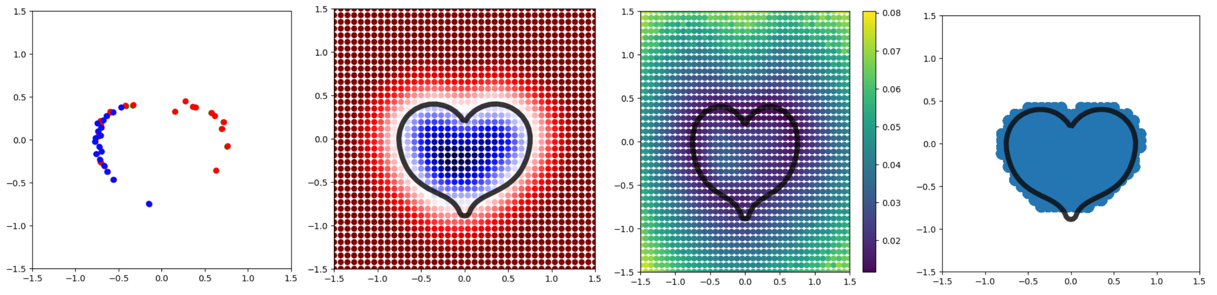

(a) Training data **(b)** Mean SDF **(c)** Variance **(d)** $\mathbb{P}(\tilde{\varphi} \leq 0) = 0.95$

Figure 3.4. Shape estimation with dropout neural network. (a) shows the training data. (b) shows the estimated mean SDF results. The black heart curve shows the ground-truth obstacle boundary, while colored regions are level-sets of the SDF estimate. The white region denotes the estimated obstacle boundary. The blue (resp. red) region denotes negative (resp. positive) signed distance. In (c), the variance of the SDF estimate is shown. In (d), we plot the estimated unsafe region with high probability, where $\mathbb{P}(\tilde{\varphi} \leq 0) = 0.95$.

where d denotes the Euclidean distance from a point $\mathbf{y} \in \mathbb{R}^2$ and the set boundary $\partial\mathcal{O}_i$. We employ incremental training with replay memory (ITRM) [101, Sec. IV] to estimate an SDF φ_i for each $\mathcal{O}_i$ from the LiDAR measurements. We use a 4-layer fully-connected neural network with parameters $\boldsymbol{\theta}$ and dropout layers to yield $\tilde{\varphi}_i(\mathbf{y}; \boldsymbol{\theta})$ with dropout rate 0.05 applied to each 512-neuron hidden layer.

Given $\mathbf{y} \in \mathbb{R}^2$, we obtain the predictive SDF mean $\hat{\varphi}_i(\mathbf{y})$ and standard deviation $\hat{\sigma}_i(\mathbf{y})$ by Monte-Carlo estimation with $T = 20$ stochastic forward passes through the dropout neural network model. When the TurtleBot moves along a circle of radius 2 while the object is placed at the center, we measure the accuracy of the online SDF method using the empirical SDF error, $\mathcal{E}_i = \frac{1}{m}\sum_{j=1}^{m} |\hat{\varphi}_i(\mathbf{y}_j)|$, where $\{\mathbf{y}_j\}_{j=1}^{m}$ are $m = 500$ points uniformly sampled on the surface of the object. In Fig. 3.4, we show the SDF estimation with measurement noise $\sigma = 0.01$.

Let $\mathbf{z} = [x, y] \in \mathcal{Z} \subset \mathbb{R}^2$ be the position part of $\mathbf{x}$. To account for the fact that the robot body is not a point mass, we subtract the robot radius $\rho = 0.177$ from each SDF estimate when defining each mean CBF: $\tilde{h}_i(\mathbf{x}) = \tilde{\varphi}_i(\mathbf{z}; \boldsymbol{\theta}) - \rho$. For variance $K_h(\mathbf{x}, \mathbf{x})$ in Sec. 3.2.1, we set $K_h^i(\mathbf{x}, \mathbf{x}) = \hat{\sigma}_i^2(\mathbf{z})$. We also take $\nabla\tilde{h}_i(\mathbf{x}) = \nabla\tilde{\varphi}_i(\mathbf{z}; \boldsymbol{\theta})$ and compute $\mathcal{H}_{\mathbf{x},\mathbf{x}'} K_h^i(\mathbf{x}, \mathbf{x}')$ by Monte-Carlo estimation using double back-propagation. We set the worst case error bounds $e_h(\mathbf{x})$, $e_{\nabla h}(\mathbf{x})$ in Sec. 3.2.2 as the 99.99% confidence bounds of a Gaussian random variable with standard deviation $\hat{\sigma}_i(\mathbf{z})$. If the robot observes multiple obstacles in the environment, we compute multiple CBFs $\tilde{h}_i(\mathbf{x})$ and their corresponding uncertainty, and add multiple CBCs to (2.13), (3.40), (3.46) for safe control synthesis.

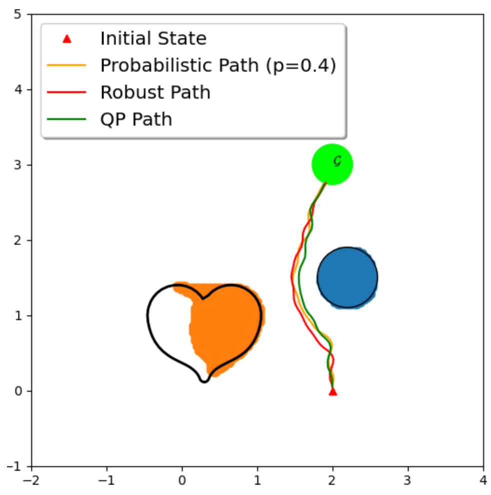
3.3.2 Safe Navigation

We evaluate safe navigation performance under different controller formulations and robot shape models, followed by an extension to safe manipulation. All experiments are performed in simulated environments containing obstacles unknown to the robot.

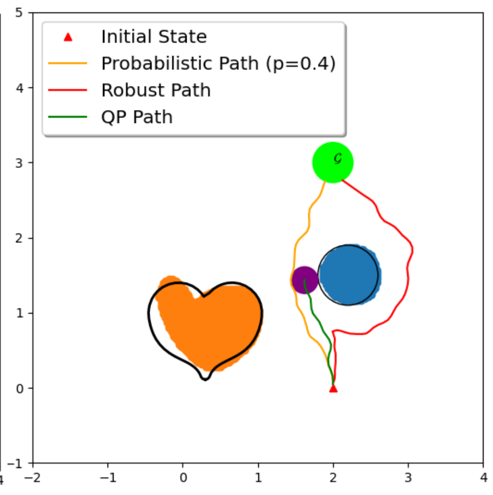
Navigation with Circular Robot Model.

We first consider a TurtleBot modeled as a circular robot and compare three controllers: (1) the original CLF-CBF-QP (2.13) (assumes perfect estimates), (2) the proposed probabilistic CLF-CBF-SOCP (3.40), and (3) the proposed robust CLF-CBF-SOCP (3.46). All controllers use $L(\mathbf{x}) = \mathrm{diag}([0, 10, 3])$, $\underline{\tilde{\mathbf{k}}}(\mathbf{x}) = [1, v_{\max}, 0]^\top$ with $v_{\max} = 0.65\,\mathrm{m/s}$, $\lambda = 1000$, $\alpha_V(V(\mathbf{x})) = 2V(\mathbf{x})$, and $\alpha_h(h_i(\mathbf{x})) = h_i(\mathbf{x})$.

Figures 3.5a and 3.5b show goal-reaching tasks with the CLF candidate $V(\mathbf{x}) = (x-2)^2 + (y-3)^2$ under two LiDAR noise levels. At $\sigma = 0.01$, all controllers succeed, with the SOCP formulations slightly more conservative than QP. At $\sigma = 0.02$, estimation variance increases and the QP controller collides with an obstacle, while both SOCP controllers avoid collisions. The robust SOCP takes a detour on the right side of the round obstacle due to larger error bounds making the left path infeasible.



(a) Noise: $\sigma = 0.01$



(b) Noise: $\sigma = 0.02$

Figure 3.5. Navigation to a goal point under different LiDAR noise levels. The QP controller collides at $\sigma = 0.02$, while both SOCP variants remain safe.

For safe trajectory tracking, we use the method from [101, Sec. VI] to construct a valid CLF $V(\mathbf{x})$ for path following. Table 3.2 reports success rates under different noise levels: the QP controller's success rate drops sharply with noise, while the robust and high-p probabilistic controllers maintain near-perfect performance.

Table 3.2. Success rate of trajectory tracking across 10 environments (10 runs each) under varying LiDAR noise σ.

LiDAR Noise σ	QP	Probabilistic (p)			Robust
		0.2	0.4	0.8	
0.01	0.82	0.98	1.0	1.0	1.0
0.02	0.65	0.92	0.97	1.0	1.0
0.05	0.37	0.72	0.89	0.96	1.0

Figure 3.6 shows a case (environment 8, $\sigma = 0.01$) where QP fails but both SOCP controllers remain safe. When obstacles lie near the reference path, robust SOCP keeps the largest clearance, followed by probabilistic SOCP, then QP.

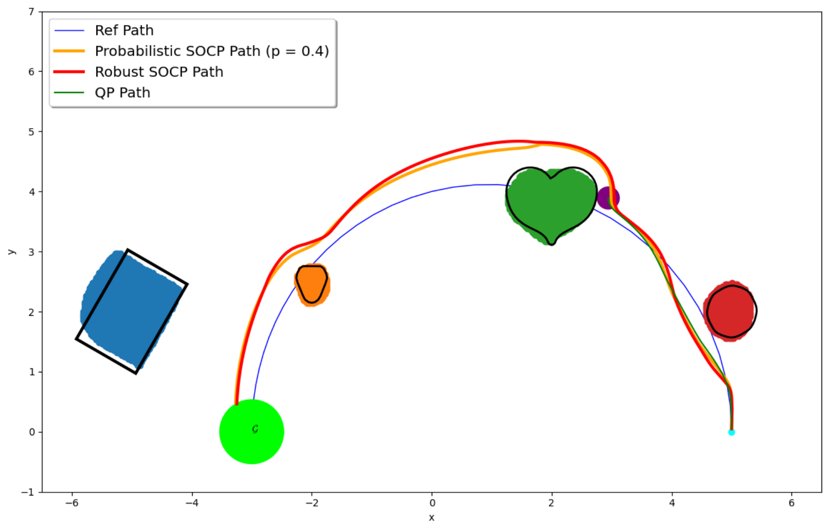


Figure 3.6. Trajectory tracking in environment 8 ($\sigma = 0.01$). QP collides, while both SOCP variants avoid obstacles.

Table 3.3 quantifies tracking performance via Fréchet distance [101, Sec. VI] under $\sigma = 0.02$. Robust SOCP is most conservative (largest distances), probabilistic SOCP offers a tunable safety–performance trade-off via p, and QP achieves closest tracking but fails in several environments.

Table 3.3. Fréchet distance between the reference path and the robot trajectories generated by the Probabilistic CLF-CBF-SOCP, Robust CLF-CBF-SOCP, and the CLF-CBF-QP controllers (smaller values indicate larger trajectory similarity, the value in the parentheses indicates the success rates while values without parentheses indicate the success rate is 1, and N/A indicates the robot collides with obstacles in all 10 realizations).

Env	QP	Probabilistic			Robust
		$p = 0.2$	$p = 0.4$	$p = 0.8$	
1	0.337	0.338	0.343	0.363	0.357
2	0.378	0.408	0.404	0.432	0.485
3	0.372	0.398	0.412	0.457	0.538
4	0.416	0.438	0.427	0.473	0.515
5	0.395	0.418	0.412	0.483	0.572
6	0.385 (0.8)	0.371	0.378	0.392	0.424
7	0.462 (0.5)	0.502	0.546	0.593	0.737
8	0.535 (0.2)	0.588	0.612	0.673	0.814
9	N/A	0.756 (0.8)	0.887 (0.9)	0.926	1.016
10	N/A	0.905 (0.4)	0.937 (0.8)	1.046	1.224

Computation times per control step are $0.00863\,\mathrm{s}$ (QP), $0.0109\,\mathrm{s}$ (probabilistic SOCP), and $0.0122\,\mathrm{s}$ (robust SOCP) on an Intel i7-9700K, confirming online feasibility.

Navigation with Polygon–Ellipse CBFs.

While the circular robot model is simple, it can be overly conservative in environments where robot orientation matters. We extend our framework to an $SE(2)$ CBF formulation for rigid-body polygon robots interacting with elliptical obstacles.

Figure 3.8 compares navigation results for a triangular robot vs. its circular approximation in a dynamic elliptical environment. The triangular robot successfully traverses a narrow passage between two moving obstacles and reaches the goal without collision. The circular approximation must take a longer route, showing that $SE(2)$ CBFs enable less conservative maneuvers.

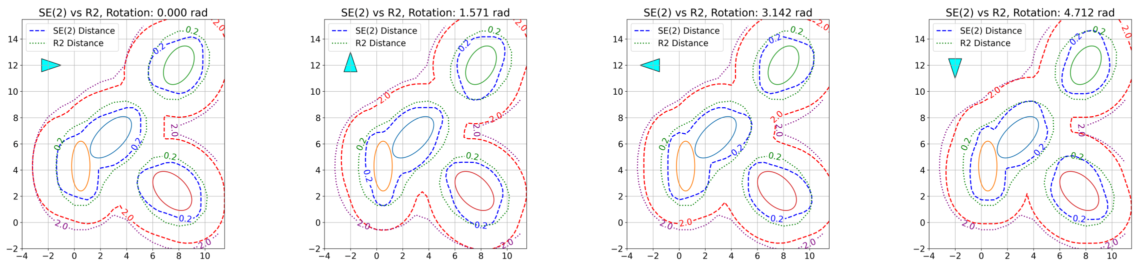
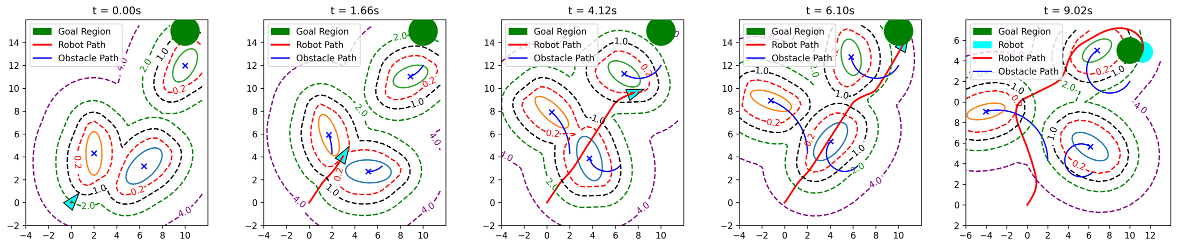


Figure 3.7. Comparative analysis of the $SE(2)$ and $\mathbb{R}^2$ signed distance functions for elliptical obstacles. The cyan triangle represents the rigid-body robot, with its orientation varying across the sequence. The importance of considering robot orientation in distance computations becomes evident: while the $SE(2)$ function accounts for this orientation, the $\mathbb{R}^2$ approximation treats the robot as an encapsulating circle with radius 1. Level sets at distances 0.2 and 2 are depicted for both functions.



(a) Initial Pose **(b)** Time t = 1.66 sec **(c)** Time t = 4.12 sec **(d)** Final Pose **(e)** Circular Robot

Figure 3.8. Safe navigation in a dynamical elliptical environment. (a) shows the initial pose of the triangular robot and the environment. (b) shows the triangular robot passing through the narrow space between two moving ellipses. (c) shows the robot adjusts its pose to avoid the moving obstacle. (d) shows the final pose of the robot that reaches the goal region. In (e), we plot the trajectory of navigating a circular robot in the same environment.

Safe Manipulation.

We further apply the polygon–ellipse CBF formulation to a 3-joint planar manipulator operating in a dynamic elliptical environment. The arm configuration is given by joint angles $\boldsymbol{\theta} = [\tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3]^\top$ with joint rates $\boldsymbol{\omega} = [\omega_1, \omega_2, \omega_3]^\top$ and dynamics

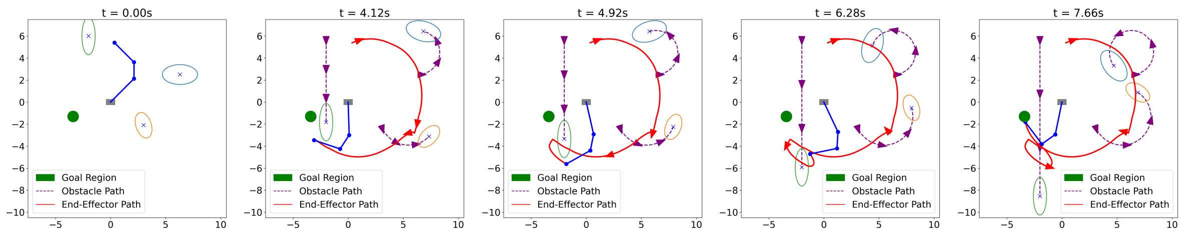
$$\dot{\boldsymbol{\theta}} = \boldsymbol{\omega}, \quad |\omega_i| \leq 3. \tag{3.51}$$

Each link is modeled as a line segment of length L_i with pose in $SE(2)$ determined by the forward kinematics; the overall arm shape is the union of all link shapes. The control Lyapunov function

$$V(\boldsymbol{\theta}) = (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \mathbf{Q}(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$$

drives the arm toward a desired joint configuration $\boldsymbol{\theta}^*$ corresponding to the goal end-effector position. Safety is enforced via CBFs $h_i(\boldsymbol{\theta})$ defined as the minimum $SE(2)$ distance between each link and the elliptical obstacles.

In Fig. 3.9, the arm begins in an extended pose and must reach the goal region while avoiding three moving elliptical obstacles. Throughout the motion, the controller adjusts all joints to maintain clearance, resulting in a smooth, collision-free trajectory of the end effector (red) to the goal (green).



(a) Initial Pose **(b)** Time t = 4.12 sec **(c)** Time t = 4.92 sec **(d)** Time t = 6.28 sec **(e)** Final Pose

Figure 3.9. Safe stabilization of a 3-joint robot arm. The green circle denotes the goal region, and the gray box denotes the base of the arm. The arm is shown in blue and the trajectory of its end-effector is shown in red. The trajectories of the moving elliptical obstacles are shown in purple.

Chapter 3, in full, is a reprint of the material as it appears in Learning Barrier Functions With Memory for Robust Safe Navigation, K. Long, C. Qian, J. Cortés, and N. Atanasov, IEEE Robotics and Automation Letters, 2021; Safe Control Synthesis With Uncertain Dynamics and Constraints, K. Long, V. Dhiman, M. Leok, J. Cortés, and N. Atanasov, IEEE Robotics and Automation Letters, 2022; and Safe Stabilizing Control for Polygonal Robots in Dynamic Elliptical Environments, K. Long, K. Tran, M. Leok, and N. Atanasov, American Control Conference, 2024. The dissertation author was the primary researcher and author of these works.

Chapter 4

Distributionally Robust Safety Filters for Robot Control

Safe autonomous operation in uncertain environments remains one of the most challenging problems in robotics. While probabilistic and robust formulations of safe control (Chapter 3) provide valuable theoretical foundations for handling uncertainty, they often suffer from practical limitations that hinder their deployment in real-world systems. Traditional probabilistic approaches require exact knowledge of uncertainty distributions, which are rarely available in practice. Meanwhile, robust control methods rely on fixed worst-case bounds that can be overly conservative or difficult to obtain for complex, multi-source uncertainties. These limitations become particularly pronounced in realistic robotic systems, where uncertainty manifests through multiple interconnected sources including localization errors, sensor noise, model mismatch, actuation delays, and geometric and state estimations.

The complexity of modern robotic systems compounds these challenges. Consider a manipulator operating in a cluttered workspace: uncertainty arises from vision-based object detection errors, kinematic calibration inaccuracies, and estimated collision geometry representations. These uncertainties do not act in isolation, they interact and propagate through the control system in ways that are difficult to predict or bound analytically. Ultimately, these combined effects manifest within safety constraint formulations, particularly in control barrier functions (CBFs), where their joint influence determines whether collision avoidance and other critical constraints are satisfied. While individual uncertainty sources can often be characterized separately through careful system identification, obtaining reliable bounds or accurate probabilistic descriptions for their combined effect in the final constraint expression remains a formidable challenge.

This fundamental difficulty motivates our adoption of distributionally robust optimization (DRO), a principled framework that circumvents the need for exact uncertainty models by reasoning directly over sets of plausible probability distributions. Rather than assuming knowledge of a single "true" distribution or requiring explicit bounds for each uncertainty source, DRO enforces safety constraints over an ambiguity set of distributions centered around empirically observed data. This approach naturally accommodates the complex, interacting nature of real-world uncertainties while providing meaningful probabilistic guarantees on constraint satisfaction.

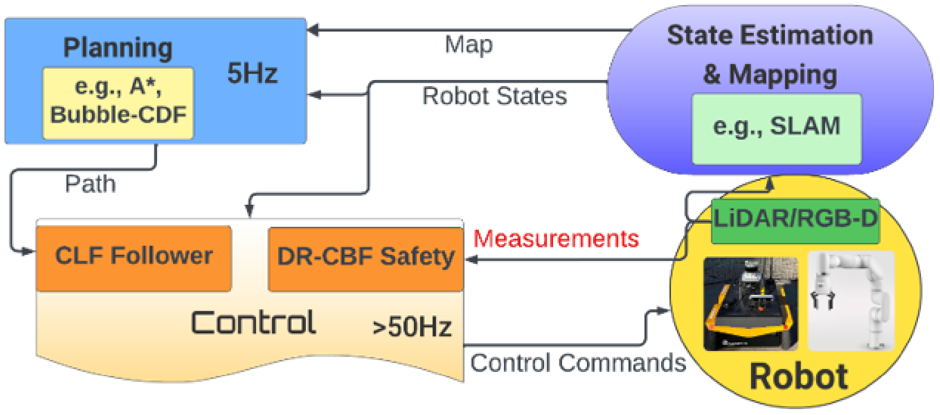


Figure 4.1. System overview

Figure 4.1 provides an overview of our proposed framework, which integrates distributionally robust optimization with control barrier functions to create a unified approach for safe control under uncertainty.

To address the limitations of existing uncertainty-aware formulations, this chapter builds upon the mathematical foundations of DRO, leveraging its ability to operate directly on sampled data without requiring explicit distributional assumptions. Our approach centers on Wasserstein ambiguity sets [47, 71], which provide a principled and geometrically intuitive means to capture distributional deviations under limited data availability. Unlike moment-based ambiguity sets that can be sensitive to outliers, or likelihood-based approaches that require distributional assumptions, Wasserstein sets offer robust performance guarantees while maintaining computational tractability. This formulation naturally accommodates uncertainty in sensor measurements (such as LiDAR point cloud noise and camera calibration errors), estimation errors from state filtering algorithms [43, 75], and perturbations in system dynamics, while providing high-probability guarantees on safety constraint satisfaction.

The integration of accurate geometric representations presents both opportunities and challenges for safe control synthesis. Traditional collision avoidance methods often rely on simplistic geometric abstractions such as spherical or cylindrical bounding volumes, which can introduce significant conservatism, particularly for manipulators operating in confined or articulated workspaces. Recent advances in neural implicit representations offer a promising alternative, enabling more expressive and compact modeling of complex robot and obstacle geometries. Signed distance functions [81,91,100] and configuration-space distance functions [90] have emerged as particularly powerful tools, providing differentiable geometric queries that integrate seamlessly with gradient-based optimization methods.

However, the learned nature of neural geometric representations introduces new sources of uncertainty that traditional robust control methods struggle to address explicitly. Neural network approximation errors, training data bias, and generalization limitations create compound uncertainty effects that are difficult to model analytically. This challenge further reinforces the value of our distributionally robust approach, which can capture and reason about such complex uncertainty interactions without requiring detailed characterization of each component.

This chapter synthesizes the research presented in [99, 104, 105], where we develop a distributionally robust control barrier constraint formulation that accounts for uncertainties from state estimation, system dynamics, sensor noise, and neural shape approximations. The central idea is to reason directly over an ambiguity set of distributions consistent with observed data, thereby avoiding the need to separately characterize or bound each uncertainty source. Despite the generality of this formulation, the resulting safe control synthesis problem can be reformulated

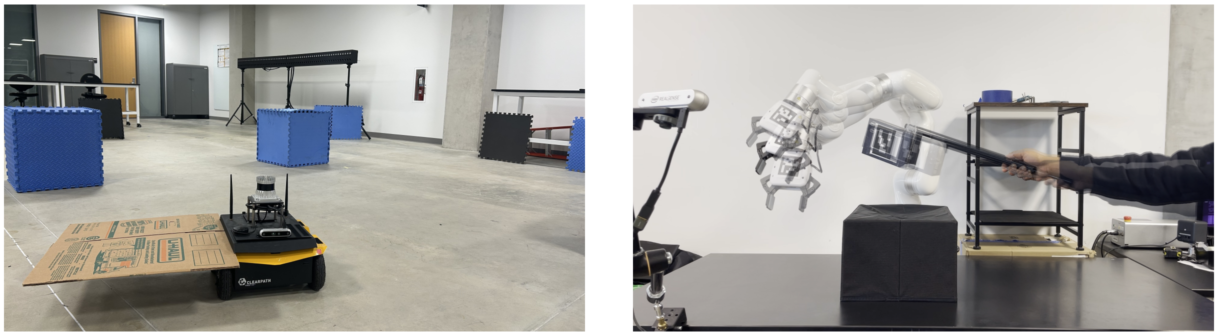

(a) Mobile Robot with Complex Geometry. **(b)** 6-DoF Manipulator Navigation

Figure 4.2. Experimental demonstrations of the distributionally robust safety filter framework for a ground mobile robot and a 6-DoF manipulator, both utilizing neural shape representations of their bodies.

as a convex quadratic program, ensuring real-time tractability on robotic platforms.

Building on these foundations, this chapter makes the following contributions:

1. **Distributionally robust control barrier constraints:** We propose a novel DRO formulation of CBF constraints that enforces safety against an ambiguity set of probability distributions centered on empirical data, rather than requiring exact knowledge of uncertainty models or conservative worst-case bounds.

2. **Integration with neural geometric representations:** We extend the DRO-CBF formulation to accommodate uncertainties arising from learned implicit shape models (e.g., neural SDFs and configuration-space distance functions), enabling shape-aware safe control despite neural approximation errors.

3. **Convex quadratic program reformulation:** We derive a tractable reformulation as a convex quadratic program, enabling real-time implementation on robotic platforms.

4. **Experimental validation on robots with complex geometry:** We demonstrate the proposed DRO safety filter on both a ground mobile robot and a 6-DoF manipulator, each leveraging neural shape representations (Fig. 4.2), showing robust safe control under uncertainty in cluttered environments.

In this chapter, we consider

Problem 4. Consider a robotic system with control-affine dynamics given by (2.5), operating in an uncertain environment and equipped with onboard sensors (e.g., LiDAR, RGB-D cameras). Design a control policy that drives the system to a desired goal state $\mathbf{x}_G \in \mathcal{X}$ while ensuring safety constraints of the form

$$\mathcal{B}(\mathbf{x}(t)) \subset \mathcal{F}(t), \quad \forall\, t \geq 0,$$

where $\mathcal{B}(\mathbf{x}(t))$ denotes the robot body at state $\mathbf{x}(t)$ and $\mathcal{F}(t)$ the estimated free workspace at time t. The policy must account for uncertainties arising from sensing, state estimation, and model mismatch, and should apply across different robot types (e.g., mobile robots and manipulators).

4.1 Distributionally Robust Safe Control

In this section, we present a distributionally robust control barrier function (DR-CBF) formulation that enables real-time safety guarantees in cluttered and dynamic environments by utilizing sensor data directly. This formulation is applicable to general control-affine systems (2.5).

We consider a CBF $h(\mathbf{x}, t)$ with super zero-level set $\mathcal{C}(t) = \{\mathbf{x} \in \mathcal{X} \mid h(\mathbf{x}, t) \geq 0\}$ that satisfies

$$\mathcal{C}(t) \subseteq \{\mathbf{x} \in \mathcal{X} \mid \mathcal{B}(\mathbf{x}(t)) \subseteq \mathcal{F}(t)\},$$

where $\mathcal{B}(\mathbf{x}(t))$ represents the robot's body at state $\mathbf{x}(t)$, and $\mathcal{F}(t)$ is the free space at time t. This establishes a connection between the CBF h and the environment geometry. To develop the DR-CBF formulation, we make the following assumption on the unknown CBF.

Assumption 4.1.1. *The CBF h has a uniform relative degree of 1 with respect to the system dynamics* (2.5)*, i.e., the time derivative of $h(\mathbf{x}, t)$ along* (2.5) *depends explicitly on the control input $\mathbf{u}$.*

Under Assumption 4.1.1, we can write the control barrier constraint associated with

$h(\mathbf{x}, t)$ as:

$$\begin{aligned}\mathrm{CBC}(\mathbf{x}, \mathbf{u}, t) &= [\nabla_{\mathbf{x}} h(\mathbf{x}, t)]^\top \mathbf{F}(\mathbf{x})\underline{\mathbf{u}} + \frac{\partial h(\mathbf{x}, t)}{\partial t} + \alpha_h(h(\mathbf{x}, t)) \\ &= \underbrace{[\nabla_{\mathbf{x}} h(\mathbf{x}, t)^\top \mathbf{F}(\mathbf{x}),\;\; \alpha_h(h(\mathbf{x}, t)),\;\; \frac{\partial h(\mathbf{x}, t)}{\partial t}]}_{\boldsymbol{\xi}^\top(\mathbf{x}, t)} \begin{bmatrix} \underline{\mathbf{u}} \\ 1 \\ 1 \end{bmatrix} \geq 0,\end{aligned} \tag{4.1}$$

where we define an uncertainty vector $\boldsymbol{\xi}(\mathbf{x}, t) \in \mathbb{R}^{m+3}$, containing the elements of the time derivative of the barrier function, for each $(\mathbf{x}, t) \in \mathcal{X} \times \mathbb{R}$.

Since the environment is unknown and both the sensor measurements and the robot state estimates are noisy, $\boldsymbol{\xi}$ cannot be determined exactly.

Our formulation addresses uncertainties arising from both the barrier function h and the state estimations $\mathbf{x}$. Unlike existing robust or probabilistic CBF approaches, which often focus on uncertainties in system dynamics [21, 28, 34, 40, 98, 143], few works tackle the challenges posed by state estimation errors. This stems from the nonlinearity of the dynamics model $\mathbf{F}$ and the barrier function h, which makes it difficult to propagate state estimation errors in the control barrier constraint (CBC) in (4.1).

Our formulation addresses this challenge by leveraging the power of distributionally robust optimization. Instead of explicitly propagating state estimation errors through the system dynamics and barrier functions, we assume access to M state samples $\{\mathbf{x}_j\}_{j=1}^{M}$ from a state estimation algorithm. These samples can be obtained, for example, from the Gaussian distributions provided by a Kalman filter [75], particles generated by a particle filter [43], or a graph-based localization algorithm [52]. The state samples, combined with estimates of h (e.g., obtained directly from the distance measurements $\boldsymbol{\eta}$), can be directly used to construct samples of the uncertainty vector $\{\boldsymbol{\xi}_i\}_{i=1}^{N}$, as detailed in [105]. By managing uncertainties in $\boldsymbol{\xi}$ using distributionally robust optimization, we ensure the satisfaction of constraint (4.1) without requiring explicit error propagation.

Before introducing the control synthesis formulation, we review the preliminaries of chance constraints and distributionally robust optimization.

4.1.1 Chance Constraints and Distributionally Robust Optimization

Consider a random vector $\boldsymbol{\xi}$ with (unknown) distribution $\mathbb{P}^*$ supported on the set $\Xi \subseteq \mathbb{R}^k$. Let $G : \mathbb{R}^m \times \Xi \to \mathbb{R}$ define an inequality constraint $G(\mathbf{u}, \boldsymbol{\xi}) \leq 0$ (e.g., the CBC in (4.1)). Consider, then, the chance-constrained program,

$$\begin{aligned} &\min_{\mathbf{u} \in \mathbb{R}^m} c(\mathbf{u}), \\ &\text{s.t. } \mathbb{P}^*(G(\mathbf{u}, \boldsymbol{\xi}) \leq 0) \geq 1 - \epsilon, \end{aligned} \tag{4.2}$$

where $c : \mathbb{R}^m \mapsto \mathbb{R}$ is a convex objective function (e.g., the objective function in (2.13)) and $\epsilon \in (0, 1)$ denotes a user-specified risk tolerance. Generally, the chance constraint in (4.2) leads to a non-convex feasible set. To address this, [112] propose a convex conditional value-at-risk (CVaR) approximation of the original chance constraint.

Value-at-risk (VaR) at confidence level $1 - \epsilon$ for $\epsilon \in (0, 1)$ is defined as $\text{VaR}_{1-\epsilon}^{\mathbb{P}_q}(Q) := \inf_{s \in \mathbb{R}}\{s \mid \mathbb{P}_q(Q \leq s) \geq 1 - \epsilon\}$ for a random variable Q with distribution $\mathbb{P}_q$. As VaR does not provide information about the right tail of the distribution and leads to intractable optimization in general, one can employ CVaR instead, defined as $\text{CVaR}_{1-\epsilon}^{\mathbb{P}_q}(Q) = \mathbb{E}_{\mathbb{P}_q}[Q \mid Q \geq \text{VaR}_{1-\epsilon}^{\mathbb{P}_q}(Q)]$. The resulting constraint

$$\text{CVaR}_{1-\epsilon}^{\mathbb{P}^*}(G(\mathbf{u}, \boldsymbol{\xi})) \leq 0 \tag{4.3}$$

creates a convex feasible set, which is a subset of the feasible set in the original chance-constrained problem (4.2). Additionally, CVaR can be written as the following convex program [126]:

$$\text{CVaR}_{1-\epsilon}^{\mathbb{P}^*}(G(\mathbf{u}, \boldsymbol{\xi})) := \inf_{s \in \mathbb{R}}[\epsilon^{-1}\mathbb{E}_{\mathbb{P}^*}[(G(\mathbf{u}, \boldsymbol{\xi}) + s)_+] - s]. \tag{4.4}$$

The formulations in (4.2) and (4.3) require knowledge of $\mathbb{P}^*$ to be utilized. However, in many robotics applications, usually only samples of the uncertainty $\boldsymbol{\xi}$ are available (e.g., obtained from LiDAR distance measurements). This motivates us to consider distributionally robust formulations [47, 147].

Assuming finitely many samples $\{\boldsymbol{\xi}_i\}_{i\in[N]}$ from the true distribution of $\mathbb{P}^*$ are available, we first describe a way of constructing an ambiguity set of distributions that agree with the empirical distribution. Let $\mathcal{P}_p(\Xi) \subseteq \mathcal{P}(\Xi)$ be the set of Borel probability measures with finite p-th moment with $p \geq 1$. The p-Wasserstein distance between two probability measures μ, ν in $\mathcal{P}_p(\Xi)$ is defined in :

$$W_p(\mu, \nu) := \left(\inf_{\beta \in \mathbb{Q}(\mu,\nu)} \left[\int_{\Xi\times\Xi} \eta(\boldsymbol{\xi}, \boldsymbol{\xi}')^p \mathrm{d}\beta(\boldsymbol{\xi}, \boldsymbol{\xi}') \right] \right)^{\frac{1}{p}}, \tag{4.5}$$

where $\mathbb{Q}(\mu, \nu)$ denotes the collection of all measures on $\Xi \times \Xi$ with marginals μ and ν on the first and second factors, and η denotes the metric in the space Ξ. Throughout the paper, we take $\eta(\boldsymbol{\xi}, \boldsymbol{\xi}') = \|\boldsymbol{\xi} - \boldsymbol{\xi}'\|_1$ and consider the ambiguity set corresponding to the 1-Wasserstein distance. We denote by $\mathbb{P}_N := \frac{1}{N}\sum_{i=1}^N \delta_{\boldsymbol{\xi}_i}$ the discrete empirical distribution of the available samples $\{\boldsymbol{\xi}_i\}_{i\in[N]}$, and define an ambiguity set, $\mathcal{M}_N^r := \{\mu \in \mathcal{P}_p(\Xi) \mid W_p(\mu, \mathbb{P}_N) \leq r\}$, as a ball of distributions with radius r centered at $\mathbb{P}_N$.

Remark 4.1.2. (Choice of Wasserstein ball radius): There is a connection between the sample size N and the Wasserstein radius r for constructing the ambiguity set $\mathcal{M}_N^r$. A distribution $\mathbb{P}$ is light-tailed if there exists an exponent ρ such that $A := \mathbb{E}_{\mathbb{P}}[\exp \|\boldsymbol{\xi}\|^\rho] = \int_\Xi \exp \|\boldsymbol{\xi}\|^\rho \mathbb{P}(d\boldsymbol{\xi}) < \infty$. If the true distribution $\mathbb{P}^*$ is light-tailed, the choice of $r = r_N(\bar{\epsilon})$ given in [47, Theorem 3.5] is

$$r_N(\bar{\epsilon}) = \begin{cases} \left(\frac{\log(c_1\bar{\epsilon}^{-1})}{c_2 N}\right)^{\frac{1}{\max\{k,2\}}} & \text{if } N \geq \frac{\log(c_1\bar{\epsilon}^{-1})}{c_2}, \\ \left(\frac{\log(c_1\bar{\epsilon}^{-1})}{c_2 N}\right)^{\frac{1}{\rho}} & \text{else,} \end{cases} \tag{4.6}$$

where c_1, c_2 are positive constants that depend on ρ, A and k, ensures that the ambiguity ball

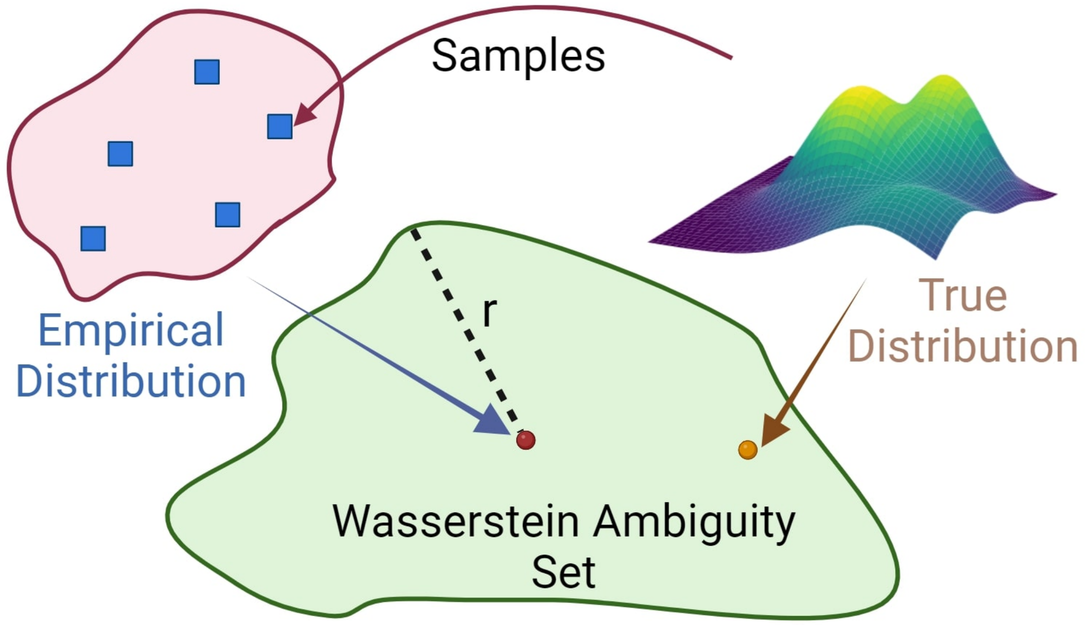


Figure 4.3. Wasserstein ambiguity set illustration. The figure shows the relationship between the samples, empirical distribution, true distribution, and the Wasserstein ambiguity set. The blue squares represent the available samples from the true distribution (yellow dot), which form the empirical distribution (red dot). The Wasserstein ambiguity set (green region) is constructed as a ball of distributions centered at the empirical distribution, with a radius r that depends on the sample size and the desired confidence level. The ambiguity set aims to contain the true distribution with high probability.

$\mathcal{M}_N^{r_N(\bar{\epsilon})}$ contains $\mathbb{P}^*$ with probability at least $1 - \bar{\epsilon}$. •

Fig. 4.3 provides an illustration of the Wasserstein ambiguity set and its relation to the samples, the empirical distribution, and the true distribution.

4.1.2 Distributionally Robust Safety Constraint

Consistently with our exposition of DRO in the previous section, we make the following assumption.

Assumption 4.1.3. *At each* $(\mathbf{x}, t) \in \mathcal{X} \times \mathbb{R}$, N *samples of the vector* $\boldsymbol{\xi}$ *in* (4.1) *can be obtained, denoted by* $\{\boldsymbol{\xi}_i\}_{i\in[N]}$.

The samples $\{\boldsymbol{\xi}_i\}_{i\in[N]}$ can be obtained using sensor measurements and state estimation (we discuss this in detail in Sec. 4.2). In many robotic systems, sensing and state estimation may operate at lower frequencies than the control loop. For example, LiDAR measurements or

SLAM-based localization may provide updates at about 10 Hz, while the control loop may require computations at 50 Hz. Our DR-CBF formulation addresses this challenge by incorporating samples of $\boldsymbol{\xi}$ derived from potentially delayed or uncertain sensor data and state estimations. By accounting for both the asynchrony and uncertainty inherent in sensing and estimation, our approach ensures probabilistic safe performance under realistic conditions.

Inspired by the CLF-CBF QP formulation in (2.13), we consider the following distributionally robust formulation to ensure safety with high probability:

$$
\begin{aligned}
(\mathbf{u}(\mathbf{x}, t), \delta) = \underset{\mathbf{u}\in\mathbb{R}^m, \delta\in\mathbb{R}}{\arg\min} \; & \|\mathbf{u} - \mathbf{k}(\mathbf{x})\|^2 + \lambda\delta^2, \\
\text{s.t.} \quad & \mathrm{CLC}(\mathbf{x}, \mathbf{u}) \leq \delta, && (4.7\text{a}) \\
& \inf_{\mathbb{P}\in\mathcal{M}_N^r} \mathbb{P}(\mathrm{CBC}(\mathbf{x}, \mathbf{u}, \boldsymbol{\xi})) \geq 0) \geq 1 - \epsilon, && (4.7\text{b})
\end{aligned}
$$

where $\mathcal{M}_N^r$ denotes the ambiguity set with radius r around the empirical distribution $\mathbb{P}_N$. The explicit time dependency of $\mathbf{u}$ on t stems from the random vector $\boldsymbol{\xi}(\mathbf{x}, t)$ in the CBF constraint. The formulation in (4.7) addresses the inherent uncertainty in the safety constraint without assuming a specific probabilistic model for $\boldsymbol{\xi}$. The Wasserstein radius r defines the acceptable deviation of the true distribution of $\boldsymbol{\xi}$ from the empirical distribution $\mathbb{P}_N$.

If a controller $\mathbf{u}^*(\mathbf{x}, t)$ satisfies (4.7b), the following result ensures that the closed-loop system satisfies a chance constraint under the true distribution.

Lemma 4.1.4. (Chance-constraint satisfaction under the true distribution): *Assume the distribution $\mathbb{P}^*$ of $\boldsymbol{\xi}$ is light-tailed and the Wasserstein radius $r_N(\bar{\epsilon})$ is set according to* (4.6). *If the controller $\mathbf{u}^*(\mathbf{x}, t)$ satisfies* (4.7b) *with $r = r_N(\bar{\epsilon})$, then*

$$
\mathbb{P}^*(\mathrm{CBC}(\mathbf{x}, \mathbf{u}^*(\mathbf{x}, t), \boldsymbol{\xi})) \geq 0) \geq (1 - \epsilon)(1 - \bar{\epsilon}). \tag{4.8}
$$

Proof. Consider the events $A := \{\mathbb{P}^* \in \mathcal{M}_N^{r_N(\bar{\epsilon})}\}$ and $B := \{\mathrm{CBC}(\mathbf{x}, \mathbf{u}^*(\mathbf{x}, t), \boldsymbol{\xi})) \geq 0\}$. From

[47, Theorem 3.4], we have $\mathbb{P}^*(A) \geq 1 - \bar{\epsilon}$. From (4.7b), we have that

$$\inf_{\mathbb{P} \in \mathcal{M}_N^{r_N(\bar{\epsilon})}} \mathbb{P}(B) \geq 1 - \epsilon. \tag{4.9}$$

Now, consider the probability of the event B under the true distribution $\mathbb{P}^*$:

$$\begin{aligned} \mathbb{P}^*(B) &\geq \mathbb{P}^*(B \cap A) = \mathbb{P}^*(B|A)\mathbb{P}^*(A) \qquad (4.10) \\ &\geq \left(\inf_{\mathbb{P} \in \mathcal{M}_N^{r_N(\bar{\epsilon})}} \mathbb{P}(B) \right) \mathbb{P}^*(A) \geq (1 - \epsilon)(1 - \bar{\epsilon}). \end{aligned}$$

□

Our previous work [97] presents a similar result but for a CLF in the context of stabilization. According to Lemma 4.1.4, the safety of the closed-loop system is guaranteed with high probability. However, the optimization problem in (4.7) is intractable [47, 71] due to the infimum over the Wasserstein ambiguity set. In Sec. 4.1.3, we discuss our approach to identify tractable reformulations of (4.7) and facilitate online safe control synthesis.

Building on the marginal safety guarantees in Lemma. 4.1.4, we also show that our formulation admits a Probably Approximately Correct (PAC)–style conditional guarantee [140], ensuring that with high confidence over the noisy observations, the controller satisfies the desired safety probability under the true distribution.

Lemma 4.1.5. (Calibration–conditional DRO safety guarantee): *Assume $\mathbb{P}^*$ is light-tailed and choose the radius $r_N(\bar{\epsilon})$ as in* (4.6). *If the control law $\mathbf{u}^*(\mathbf{x}, t)$ satisfies* (4.7b) *with $r = r_N(\bar{\epsilon})$, then*

$$\mathbb{P}^N\Big(\mathbb{P}^*\big(\mathrm{CBC}(\mathbf{x}, \mathbf{u}^*(\mathbf{x}, t), \boldsymbol{\xi}) \geq 0\big) \;\geq\; 1 - \epsilon\Big) \;\geq\; 1 - \bar{\epsilon}.$$

Unlike the marginal guarantee in Lemma 4.1.4, which only ensures average-case safety across sampled data, Lemma 4.1.5 provides a conditional guarantee that holds with high probability for the specific data realization. This stronger result is important for practical deployment, as it ensures that the controller is not only safe on average but also reliable given the particular noisy observations available.

4.1.3 Tractable Convex Reformulation

Next, we demonstrate how the samples $\{\boldsymbol{\xi}_i\}_{i\in[N]}$ from Assumption 4.1.3 can be used to obtain a tractable reformulation of (4.7).

Proposition 4.1.6. (Distributionally robust safe control synthesis): *Given samples* $\{\boldsymbol{\xi}_i\}_{i\in[N]}$ *of* $\boldsymbol{\xi}$ *as in* (4.1)*, if* $(\mathbf{u}^*, \delta^*, s^*, \{\beta_i^*\}_{i\in[N]})$ *is a solution to the quadratic program:*

$$\begin{aligned}
&\min_{\mathbf{u}\in\mathbb{R}^m,\delta\in\mathbb{R},s\in\mathbb{R},\beta_i\in\mathbb{R}} \|\mathbf{u}-\mathbf{k}(\mathbf{x})\|^2+\lambda\delta^2, \\
\text{s.t. }\ & \mathrm{CLC}(\mathbf{x},\mathbf{u})\le\delta, \\
& r\|\underline{\mathbf{u}}\|_\infty \le s\epsilon - \frac{1}{N}\sum_{i=1}^N \beta_i, \\
& \beta_i \ge s-[\underline{\mathbf{u}}\ \ 1\ \ 1]^\top\boldsymbol{\xi}_i,\ \beta_i\ge 0,\ \forall i\in[N],
\end{aligned} \tag{4.11}$$

then $(\mathbf{u}^*, \delta^*)$ *is also a solution to the distributionally robust chance-constrained program in* (4.7).

Proof. The safety constraint (4.7b) is equivalent to $\sup_{\mathbb{P}\in\mathcal{M}_N^r}\mathbb{P}(-\mathrm{CBC}(\mathbf{x},\mathbf{u},\boldsymbol{\xi})\ge 0)\le\epsilon$. Using the CVaR approximation of the chance constraint (2.16), we obtain a convex conservative approximation of (4.7b):

$$\sup_{\mathbb{P}\in\mathcal{M}_N^r} \mathrm{CVaR}_{1-\epsilon}^{\mathbb{P}}(-\mathrm{CBC}(\mathbf{x},\mathbf{u},\boldsymbol{\xi}))\le 0. \tag{4.12}$$

From (4.4), this is equivalent to

$$\sup_{\mathbb{P}\in\mathcal{M}_N^r}\inf_{s\in\mathbb{R}}[\frac{1}{\epsilon}\mathbb{E}_{\mathbb{P}}[(-\mathrm{CBC}(\mathbf{x},\mathbf{u},\boldsymbol{\xi})+s)_+]-s]\le 0. \tag{4.13}$$

Based on [71, Lemma V.8] and [47, Theorem 6.3], with the 1-Wasserstein distance, the following inequality is a sufficient condition for (4.13) to hold:

$$rL(\mathbf{u})+\inf_{s\in\mathbb{R}}\left[\frac{1}{N}\sum_{i=1}^{N}(-\text{CBC}(\mathbf{x},\mathbf{u},\boldsymbol{\xi}_i)+s)_+ - s\epsilon\right]\leq 0 \tag{4.14}$$

where $L(\mathbf{u})$ is the Lipschitz constant of $-\text{CBC}(\mathbf{x},\mathbf{u},\boldsymbol{\xi})$ in $\boldsymbol{\xi}$. Now, from (4.1), we have $\text{CBC}(\mathbf{x},\mathbf{u},\boldsymbol{\xi}) = [\underline{\mathbf{u}}\ \ 1\ \ 1]^\top\boldsymbol{\xi}$. Therefore, we can define the convex function $L:\mathbb{R}^m\mapsto\mathbb{R}_{>0}$ by

$$L(\mathbf{u}) = \|[\underline{\mathbf{u}}\ \ 1\ \ 1]^\top\|_\infty = \max\{1,\|\underline{\mathbf{u}}\|_\infty\} = \|\underline{\mathbf{u}}\|_\infty \tag{4.15}$$

The function $\boldsymbol{\xi}\mapsto -\text{CBC}(\mathbf{x},\mathbf{u},\boldsymbol{\xi})$ is Lipschitz in $\boldsymbol{\xi}$ with constant $L(\mathbf{u})$. This is because the Lipschitz constant of a differentiable affine function equals the dual norm of its gradient, and the dual norm of the L_1 norm is the L_∞ norm. Thus, the following is a conservative approximation of (4.7),

$$\begin{aligned}&\min_{\mathbf{u}\in\mathbb{R}^m,\delta\in\mathbb{R}}\ \|\mathbf{u}-\mathbf{k}(\mathbf{x})\|^2+\lambda\delta^2, \qquad (4.16)\\ &\text{s.t. } \text{CLC}(\mathbf{x},\mathbf{u})\leq\delta,\\ &rL(\mathbf{u})+\inf_{s\in\mathbb{R}}\left[\frac{1}{N}\sum_{i=1}^{N}(-\text{CBC}(\mathbf{x},\mathbf{u},\boldsymbol{\xi}_i)+s)_+ - s\epsilon\right]\leq 0.\end{aligned}$$

Lastly, as shown in [104, Proposition IV.1], the bi-level optimization in (4.16) can be rewritten as (4.11). □

Formally, the safe set $\mathcal{C}(t)$ is defined by a single CBF $h(\mathbf{x},t)$, but its value, gradient, and time derivative (defining $\boldsymbol{\xi}(\mathbf{x},t)$) are not directly known in practice. By leveraging observations of $\boldsymbol{\xi}$ derived from sensor measurements and state estimates, and applying Proposition 4.1.6, we are able to synthesize safe controllers with distributionally robust guarantees. Due to the convex reformulation of (4.7b), our approach inherently introduces some conservatism. However, in practice, this conservatism can be effectively managed by tuning the Wasserstein radius r and the safety probability ϵ. These parameters provide flexibility to adapt the formulation based on specific application requirements, sensor capabilities, and state estimation errors.

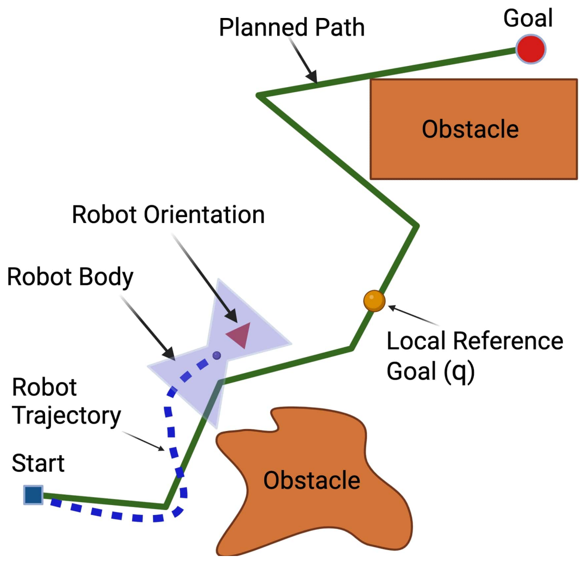
The Lipschitz continuity and regularity of distributionally robust controllers are characterized in [108]. Together with Lemma 4.1.4, this enables safe robot control with point-wise probabilistic guarantees in unknown dynamic environments.

Remark 4.1.7. (Uncertainty in System Dynamics): We have assumed that there is no uncertainty in the system dynamics $\mathbf{F}$ to simplify the presentation in Sec. 4.1.3. However, our approach can be extended if this is not the case as long as samples of $\mathbf{F}(\mathbf{x})$ are available. These samples can be combined with samples of robot state and $h(\mathbf{x}, t)$ to construct the uncertainty vector $\{\boldsymbol{\xi}_i\}_{i=1}^N$ and ensure the validity of Proposition 4.1.6. •

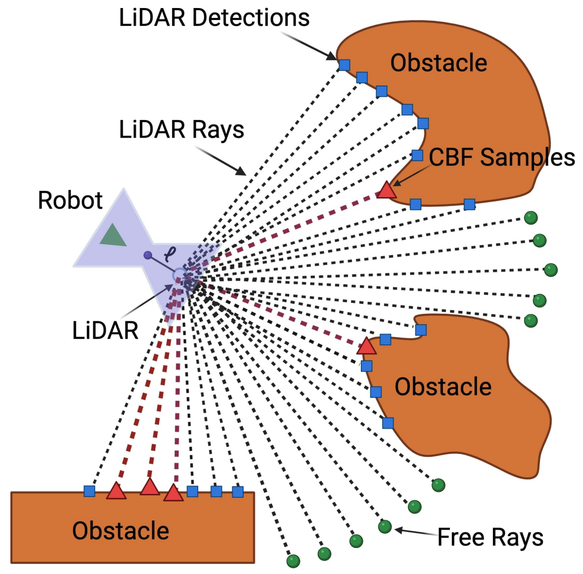
4.2 Application to Robot Navigation

This section demonstrates the practical application of our distributionally robust safety filter framework to autonomous robot navigation. We focus on differential-drive mobile robots operating in unknown and dynamic environments, where the robot must rely solely on onboard sensor measurements to maintain safety while tracking planned trajectories. The key challenge lies in handling the compound uncertainties arising from sensor noise, localization errors, and dynamic obstacles without requiring explicit probabilistic models or conservative worst-case bounds.

Figure 4.4 illustrates the key components of our navigation framework, which integrates path-following control with sensor-based safety constraints through a unified distributionally robust optimization formulation.



(a) CLF-based path following with dynamic reference point (yellow) along planned trajectory.



(b) Robot sensing environment with LiDAR. Red triangles indicate boundary points used for DR-CBF sampling.

Figure 4.4. Navigation system overview showing (a) Control Lyapunov Function-based path tracking with dynamic goal adjustment and (b) sensor-based distributionally robust control barrier function construction using LiDAR measurements for collision avoidance.

4.2.1 Robot Dynamics and Path Following

We consider a differential-drive robot with state $\mathbf{x} = [x, y, \theta]^\top \in \mathcal{X} \subseteq \mathbb{R}^2 \times [-\pi, \pi)$, control input $\mathbf{u} = [v, \omega]^\top \in \mathcal{U} \subset \mathbb{R}^2$, and dynamics:

$$\dot{\mathbf{x}} = \begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{4.17}$$

where v and ω represent linear and angular velocities, respectively. The position projection function is $\phi(\mathbf{x}) = [x, y]^\top$.

For path following, we employ a Control Lyapunov Function (CLF) approach with a moving reference point along the planned path $\gamma(s)$, where $s \in [0, 1]$ parameterizes the path.

Following the reference governor methodology, we introduce scalar dynamics:

$$\dot{s} = \frac{k}{1 + \|\phi(\mathbf{x}) - \gamma(s)\|}(1 - s^{\zeta}), \tag{4.18}$$

where $k > 0$ controls the reference progression speed, and $\zeta \in \mathbb{N}$ ensures s approaches but never exceeds 1.

The CLF is constructed to stabilize the robot to the moving reference point $\gamma(s)$:

$$V(\mathbf{x}, s) = \frac{1}{2}\left(k_v\|\gamma(s) - \phi(\mathbf{x})\|^2 + k_\omega \text{atan2}(e_v^{\perp}, e_v)^2\right), \tag{4.19}$$

where $k_v, k_\omega > 0$ are control gains, and the error terms are:

$$e_v = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}^{\top} (\gamma(s) - \phi(\mathbf{x})), \tag{4.20}$$

$$e_v^{\perp} = \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}^{\top} (\gamma(s) - \phi(\mathbf{x})). \tag{4.21}$$

This construction ensures asymptotic convergence to the planned path while accommodating the underactuated nature of the differential-drive system.

4.2.2 Sensor-Based Safety Constraints

The robot employs a 360° LiDAR sensor to perceive its environment and construct safety constraints. Rather than building explicit environment maps, our approach directly uses range measurements to formulate distributionally robust control barrier constraints.

Signed Distance Function as Control Barrier Function

For a robot with shape $\mathcal{B}_0$ centered at the origin, the oriented robot body at state $\mathbf{x}$ is given by:

$$\mathcal{B}(\mathbf{x}) = \{\mathbf{R}(\mathbf{x})\mathbf{p} + \phi(\mathbf{x}) : \mathbf{p} \in \mathcal{B}_0\}, \tag{4.22}$$

where $\mathbf{R}(\mathbf{x}) \in SO(2)$ is the rotation matrix corresponding to orientation θ.

The safety constraint is formulated using the distance between the robot body and obstacle set $\mathcal{O}(t)$:

$$h(\mathbf{x}, t) = d(\mathcal{B}(\mathbf{x}), \mathcal{O}(t)) = \inf_{\mathbf{q} \in \mathcal{O}(t)} d_{\mathcal{B}_0}\left(\mathbf{R}(\mathbf{x})^\top (\mathbf{q} - \phi(\mathbf{x}))\right), \tag{4.23}$$

where $d_{\mathcal{B}_0}(\cdot)$ is the signed distance function to the robot shape.

Since the obstacle set $\mathcal{O}(t)$ is unknown and time-varying, this function is generally non-smooth due to both the properties of the signed distance function and the infimum operation. We address this using nonsmooth control barrier function theory, where the barrier constraint is expressed in terms of the Clarke generalized gradient.

Distributionally Robust CBF Sample Selection

Rather than computing $h(\mathbf{x}, t)$ exactly, we construct CBF samples directly from LiDAR measurements. At each time step, the robot collects K range measurements $\{\eta_i\}_{i=1}^K$ corresponding to different angular directions. Each measurement yields a detected point $\mathbf{q}_i(t) \in \mathbb{R}^2$ in the global frame.

To account for both sensor noise and localization uncertainty, we consider M samples of the robot's estimated pose $\{\mathbf{x}^{(j)}\}_{j=1}^M$. For each pose sample, we transform the LiDAR measurements to obtain a comprehensive set of obstacle points:

$$\mathcal{P}(t) = \{\mathbf{q}_i^{(j)}(t) : i = 1, \ldots, K; j = 1, \ldots, M\}. \tag{4.24}$$

From this aggregated data, we select the N most critical samples based on the criterion:

$$\frac{\partial}{\partial t} d_{\mathcal{B}_0}\left(\mathbf{R}(\mathbf{x})^\top(\mathbf{q}_i(t) - \phi(\mathbf{x}))\right) + \alpha_h\left(d_{\mathcal{B}_0}\left(\mathbf{R}(\mathbf{x})^\top(\mathbf{q}_i(t) - \phi(\mathbf{x}))\right)\right), \tag{4.25}$$

where α_h is a class-$\mathcal{K}_\infty$ function. This criterion identifies points where safety constraints are most critical, considering both current proximity and relative motion.

Each selected sample yields a constraint vector:

$$\boldsymbol{\xi}_i = [\mathbf{F}^\top(\mathbf{x})\nabla_{\mathbf{x}} h_i(\mathbf{x}, t), \alpha_h(h_i(\mathbf{x}, t)), \frac{\partial h_i(\mathbf{x}, t)}{\partial t}], \tag{4.26}$$

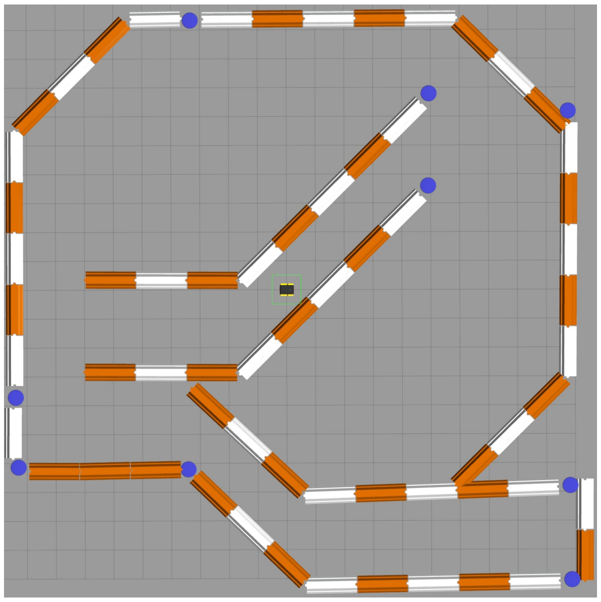
where $h_i(\mathbf{x}, t) = d_{\mathcal{B}_0}(\mathbf{R}(\mathbf{x})^\top(\mathbf{q}_i(t) - \phi(\mathbf{x})))$ and $\mathbf{F}(\mathbf{x}) = [\mathbf{f}(\mathbf{x}), \mathbf{g}(\mathbf{x})]$ combines the drift and input matrices from the robot dynamics.

The complete control synthesis problem combines the CLF constraint for path following with the DR-CBF constraints for safety. Using the tractable reformulation from Sec 4.1.3, this becomes a quadratic program and enables real-time control synthesis while providing formal guarantees on both path-following performance and collision avoidance under uncertainty.

4.2.3 Simulated and Experimental Validation

In this section, we evaluate our CLF-DR-CBF QP formulation through several simulation and real-world experiments.

We compare our approach with two other safe control strategies, the nominal CLF-CBF QP in (2.13) and a CLF-Gaussian Process (GP)-CBF second-order cone program (SOCP) [98]. The nominal CLF-CBF QP approach utilizes the closest LiDAR point to define a single CBF $h(\mathbf{x}, t)$ and its gradients at each time step. In the CLF-GP-CBF SOCP method, a real-time GP-SDF model [146] of the unknown environment is constructed using LiDAR data, from which the CBF, its gradient, and uncertainty information are determined. While the GP-SDF mapping process contributes to safety by continuously updating the environment representation, it also


(a) Static environment

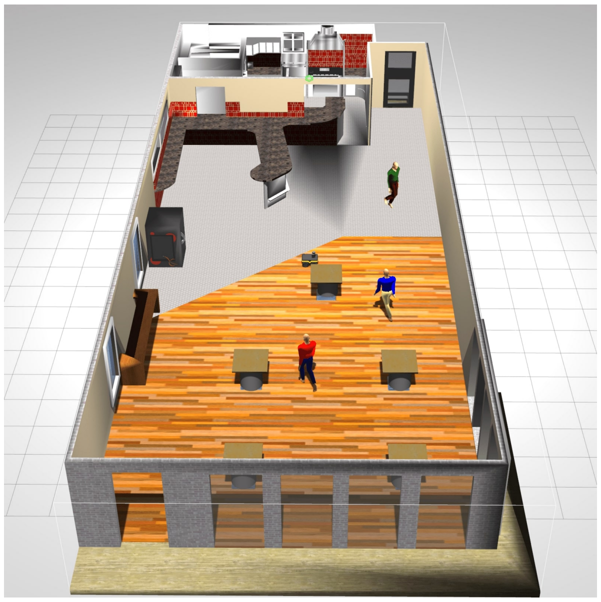

(b) Dynamic environment

Figure 4.5. Simulated environments in Gazebo.

incurs computational overhead due to the real-time update of the GP-SDF model. For a fair evaluation, we solve each of the optimization programs to generate control signals using the Interior Point Optimizer through the CasADi framework [11].

In the following simulations and experiments, a consistent set of parameters is utilized to ensure reproducibility of the results. The linear velocity is constrained in $[-1.2, 1.2]$ m/s and the angular velocity is limited within $[-1, 1]$ rad/s. The nominal control input $\mathbf{k}(\mathbf{x})$ is set to $[1.2, 0]^\top$, directing the robot to move forward at 1.2 m/s. While we use a constant nominal controller, note that our formulation supports more complex, state-dependent nominal controllers. The scaling factor is $\lambda = 50$. Table 4.1 summarizes other parameter values.

Table 4.1. Simulation and experiment parameters. The class $\mathcal{K}$ function α_V for CLF and the class $\mathcal{K}_\infty$ for CBF are assumed to be linear. The parameters k_v and k_ω are control gains for linear and angular velocities, respectively, ϵ the risk tolerance of the CLF-DR-CBF QP formulation, and N the DR-CBF sample size.

Parameters	α_V	α_h	k_v	k_ω	ϵ	N
Value	1.0	1.5	0.05	0.4	0.1	5

In all results, the A^* algorithm is employed for path planning, operating at a frequency of

5 Hz. Our CLF-DR-CBF QP formulation is used for real-time safe navigation, running at 50 Hz.

Simulated Static Environments

Table 4.2. Performance metrics for static environments under varying LiDAR and localization noise. The metrics include stuck rate and collision rate out of 1000 trials and tracking error (m) (mean ± std). The stuck rate reflects cases of infeasibility or the robot being trapped in local optima. The robot starts from the origin, and goals are placed at least 10 meters away in Figure 4.5a.

LiDAR Noise	Localization Noise	Method	Stuck Rate	Collision Rate	Tracking Error (m)
$\sigma = 0.001$	$\sigma = 0.01$	CLF-DR-CBF QP	0.0	0.0	1.33 ± 0.19
		CLF-GP-CBF SOCP	0.2	0.0	1.38 ± 0.31
		Nominal CLF-CBF QP	0.0	0.2	1.26 ± 0.20
	$\sigma = 0.05$	CLF-DR-CBF QP	0.1	0.0	1.88 ± 0.35
		CLF-GP-CBF SOCP	0.3	15.8	2.16 ± 0.47
		Nominal CLF-CBF QP	0.1	15.5	1.92 ± 0.34
$\sigma = 0.05$	$\sigma = 0.01$	CLF-DR-CBF QP	0.0	0.0	1.77 ± 0.38
		CLF-GP-CBF SOCP	25.1	0.2	1.97 ± 0.52
		Nominal CLF-CBF QP	1.7	4.4	1.82 ± 0.41
	$\sigma = 0.05$	CLF-DR-CBF QP	1.8	0.7	2.22 ± 0.57
		CLF-GP-CBF SOCP	31.3	13.9	2.58 ± 0.78
		Nominal CLF-CBF QP	3.3	25.8	2.23 ± 0.59
$\sigma = 0.1$	$\sigma = 0.01$	CLF-DR-CBF QP	1.2	0.0	2.31 ± 0.66
		CLF-GP-CBF SOCP	60.7	0.0	2.55 ± 0.79
		Nominal CLF-CBF QP	4.7	7.7	2.52 ± 0.72
	$\sigma = 0.05$	CLF-DR-CBF QP	6.6	1.8	2.61 ± 0.75
		CLF-GP-CBF SOCP	65.4	22.0	2.78 ± 0.91
		Nominal CLF-CBF QP	8.9	58.3	2.55 ± 0.69

We evaluate the performance of our CLF-DR-CBF QP formulation in static environments simulated in Gazebo (Fig. 4.5a), comparing it with two baseline approaches: Nominal CLF-CBF QP and CLF-GP-CBF SOCP [98].

Hypothesis: Our CLF-DR-CBF QP formulation ensures robust and safe navigation under varying levels of sensor and localization noise. Compared to baseline methods, our approach should achieve lower failure rates (stuck and collision) and demonstrate superior adaptability to noise conditions without compromising computational efficiency.

Setup: The simulations are conducted in a static Gazebo environment where the robot is tasked to achieve a predefined goal while avoiding obstacles. Gaussian noise with standard deviation σ (ranging from 0.001 to 0.1) is added to the LiDAR measurements, and localization noise with σ values up to 0.05 is introduced. For each noise level, 1000 trials are conducted with randomly placed goal locations at least 10 meters away from the robot's starting position. The evaluation metrics include stuck rate, collision rate, and average tracking error (mean $\pm$ std), as summarized in Table 4.2.

The stuck rate and collision rate together determine the success rate, representing the percentage of trials where the robot successfully reaches the goal without safety violations. The stuck rate captures two failure modes. First, the optimization program may become infeasible due to large uncertainties, particularly in the GP-CBF method, where the GP-SDF map's high variance at large LiDAR noise levels often renders the SOCP problem infeasible. Second, the robot may become trapped in a corner, unable to make further progress toward the goal.

Table 4.3. Computation time comparison between different control approaches (in milliseconds). The values represent the mean $\pm$ standard deviation of the computation time along the robot trajectory. The total computation time for each method is the sum of the GP map training time (if applicable), inference time, and control synthesis solver time. The CLF-DR-CBF QP and Nominal CLF-CBF QP methods have similar total computation time, as they do not require map updates. For these two methods, the inference time refers to processing the LiDAR data as CBF samples and corresponding gradients. The CLF-GP-CBF SOCP method has the highest total computation time due to the additional overhead of GP map training.

Method	Map Training	Inference	Solver	Total Time
CLF-DR-CBF QP	0	0.2	7.1 ± 2.2	7.3 ± 2.2
CLF-GP-CBF SOCP	8.6 ± 3.1	0.3	9.6 ± 2.8	18.5 ± 5.9
Nominal CLF-CBF QP	0	0.2	6.4 ± 2.3	6.6 ± 2.3

Results and discussion: The results in Table 4.2 highlight the robustness of the CLF-DR-CBF QP method, which consistently achieves low stuck and collision rates, even under high noise levels. In contrast, the GP-CBF and Nominal CLF-CBF QP methods exhibit performance degradation in challenging noise conditions.

As localization noise increases, the two baseline methods are more prone to collisions. This is because their formulations do not explicitly account for uncertainties in the robot state estimation. Our CLF-DR-CBF QP method remains robust by explicitly addressing localization uncertainties in its formulation. Similarly, at higher LiDAR noise levels, the GP-CBF method struggles due to significant variance in the GP-SDF estimation, often rendering its optimization program infeasible. The CLF-DR-CBF QP method, by directly using LiDAR measurements and incorporating distributionally robust constraints, avoids reliance on explicit map construction, exhibiting a higher probability of reliable performance at higher noise levels. Table 4.3 highlights the computational efficiency of the proposed method. The CLF-DR-CBF QP formulation achieves computation times comparable to the Nominal CLF-CBF QP method while significantly outperforming the GP-CBF method. This advantage stems from the CLF-DR-CBF QP method's direct use of LiDAR measurements without requiring computationally expensive GP map construction.

Overall, these results demonstrate the robustness and efficiency of the proposed CLF-DR-CBF QP formulation. By directly handling noisy sensor data and avoiding reliance on explicit map reconstruction, the method effectively balances computational efficiency and robust safety, making it suitable for real-world applications where sensing noise, localization noise, and computational constraints are significant challenges.

Simulated Dynamic Environments

Table 4.4. Performance metrics for dynamic environments over 1000 trials. The metrics include success rate, stuck rate, collision rate, and task completion time (mean ± std). The CLF-DR-CBF QP outperforms both baselines in terms of success rate and collision avoidance, demonstrating its robustness in dynamic settings.

Method	**Success (%)**	**Stuck (%)**	**Collision (%)**	**Time (s)**
CLF-DR-CBF QP	**93.2**	**5.1**	**1.7**	10.7 ± 2.2
CLF-GP-CBF SOCP	60.5	36.3	3.2	13.6 ± 2.9
Nominal CLF-CBF QP	61.7	8.5	29.8	10.1 ± 2.1

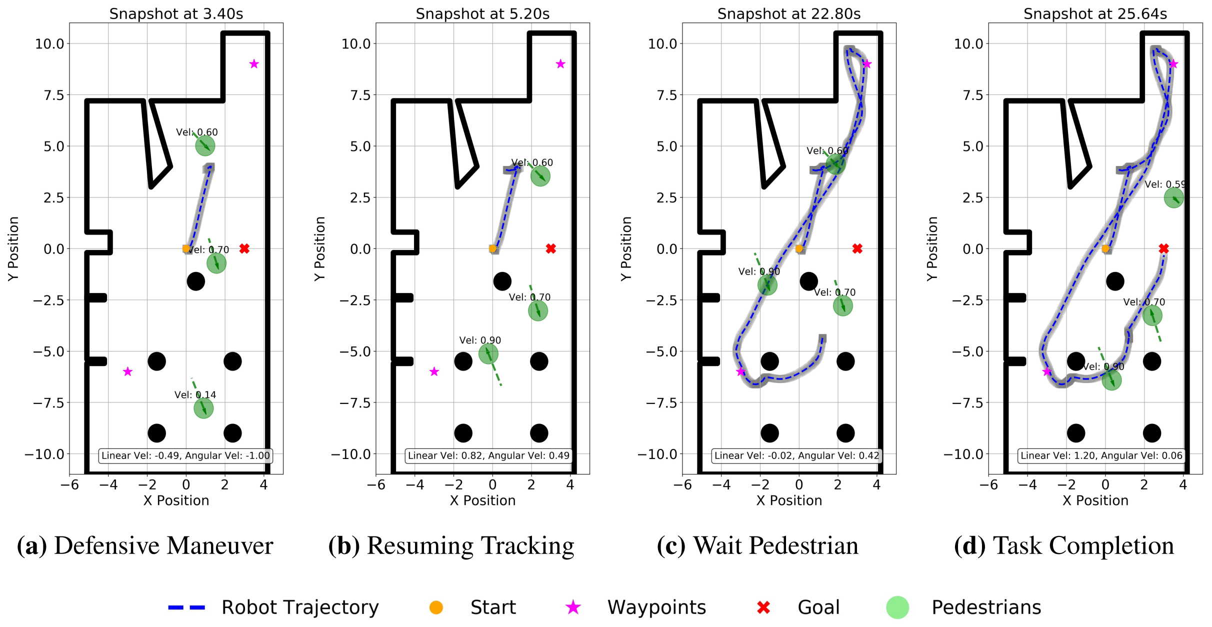


Figure 4.6. Snapshots showing safe robot navigation in a simulated dynamic environment with three pedestrians, as depicted in Fig. 4.5b. The ground-truth static environment (e.g., walls, table base) is plotted in black. Each pedestrian is represented by a light green circle, with trajectory over the past second and current velocity also displayed. (a) At $t = 3.4$s, the robot adjusts its trajectory due to an approaching pedestrian, adopting a defensive maneuver by rotating left (-1 rad/s) and moving backwards (-0.49m/s). (b) By $t = 5.2$s, as the pedestrian clears, the robot accelerates forward (0.89m/s) to track its planned path towards the first waypoint. (c) At $t = 22.8$s, facing another pedestrian crossing its planned path, the robot stops (-0.02m/s) to allow the pedestrian to pass. (d) The complete trajectory at $t = 25.6$s shows the robot successfully navigated to two waypoints and the final goal, ensuring safety in a dynamically changing environment.

The dynamic environment simulations are conducted in Gazebo (cf. Fig. 4.5b), designed to mimic real-world scenarios with pedestrians modeled using the social force model [67, 111].

Hypothesis: Our CLF-DR-CBF QP approach will outperform baseline methods in handling time-varying constraints under noisy conditions. Specifically, we expect our method to achieve higher success rates, lower collision rates, and efficient task completion times, due to its ability to incorporate sensor noise and localization uncertainty directly into the control formulation.

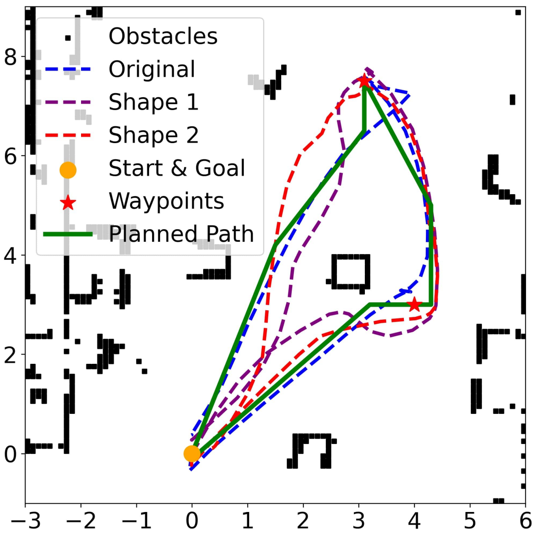
Setup: The robot starts at $(0, 0)$ with an initial orientation of $0°$, and the goal locations are randomly placed at least 6 meters away. Pedestrians are also randomly positioned in the environment, with velocities bounded by $B = 1$ m/s. Both the static and dynamic elements of the environment are unknown, and the robot relies on noisy LiDAR measurements (Gaussian noise with $\sigma = 0.05$) for collision avoidance. In all simulations below, the A^* planning algorithm operates independently of the pedestrian motion, and the real-time pedestrian avoidance relies on our CLF-DR-CBF QP formulation (or the two baseline approaches). We conducted 1000 trials for each method, measuring metrics such as success rate, stuck rate, collision rate, and average task completion time. Success is defined as reaching the goal while maintaining safety, while a trial is considered stuck if the robot fails to reach the goal within 20 seconds.

Results and discussion: Table 4.4 summarizes the quantitative results. The proposed CLF-DR-CBF QP approach achieves the highest success rate (93.2%) and the lowest collision rate (1.7%) among the three methods. By directly handling sensor noise through its distributionally robust formulation, the CLF-DR-CBF QP method ensures safety while maintaining efficient task completion times. The GP-CBF method exhibits a lower success rate and higher stuck rate due to its reliance on the GP-SDF map, which becomes computationally expensive and less reliable in dynamic environments. The Nominal CLF-CBF QP approach suffers from the highest collision rate (29.8%), highlighting its limitations in handling dynamic obstacles with sensor noise.

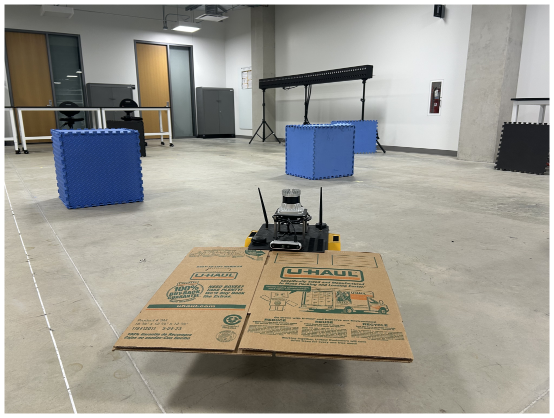
We next present some qualitative results in Fig. 4.6 in the same dynamic environment. The robot is tasked to sequentially visit two waypoints before reaching a designated goal at $(3, 0)$. In Fig. 4.6a, at $t = 3.4$s, the robot encounters a pedestrian on a collision course with its planned path to the first waypoint at the top right. With our CLF-DR-CBF QP controller, the robot employs a defensive maneuver. This adjustment shows the methodology's capability to anticipate potential hazards and react accordingly. As the pedestrian clears the immediate area, the robot resumes its path tracking towards the first waypoint by 5.2s (Fig. 4.6b). This behavior highlights the efficiency of our approach in balancing mission objectives with the need for safety. The challenge intensifies at $t = 22.8$s when another pedestrian intersects the robot's planned route (Fig. 4.6c). In response, the robot stops to allow the pedestrian to pass safely. Once the pedestrian has passed, the robot continues its journey towards the goal.

The successful completion of the task is shown in Fig. 4.6d, where the robot reaches its final goal after safely navigating past all dynamic obstacles at $t = 25.64$s. This simulation shows the CLF-DR-CBF QP controller's ability for robust path tracking and obstacle avoidance in a dynamic environment.

Real-World Experiments



(a) Comparison of trajectories for the three robot shapes.



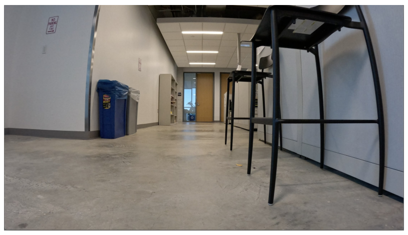
(b) Robot Shape 2

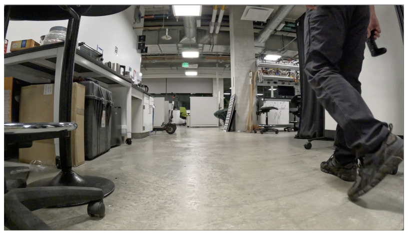
Figure 4.7. Robot shapes and trajectories in real-world experiments.

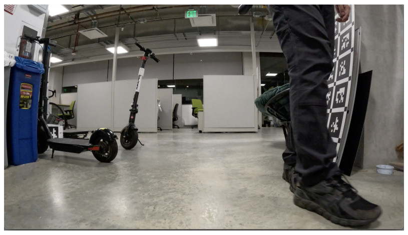
We carried out real-world experiments using a differential-drive ClearPath Jackal robot. The robot was equipped with an Intel i7-9700TE CPU with 32GB RAM, an Ouster OS1-32 LiDAR, and a UM7 9-axis IMU, and a velocity controller accepting linear and angular velocity.

Setup: The experiments took place in a lab environment, designed to test various challenging scenarios. The robot relied solely on noisy LiDAR measurements for navigation. We first tested our approach in an area of the lab with static obstacles, using three different robot shapes: the original shape, Shape 1 (Fig. 4.2a), and Shape 2 (Fig. 4.7b). For quantitative evaluation, we ran 50 trials per shape in environments populated with randomly placed obstacles (e.g., cubes and pyramids) and three randomly walking pedestrians.

Results and discussion: Figure 4.7a shows the trajectories of the robot with the three shapes navigating the same environment by following a pre-planned path. Notably, the planned


(a) Thin chair legs


(b) Narrow passage


(c) Pedestrian approaching

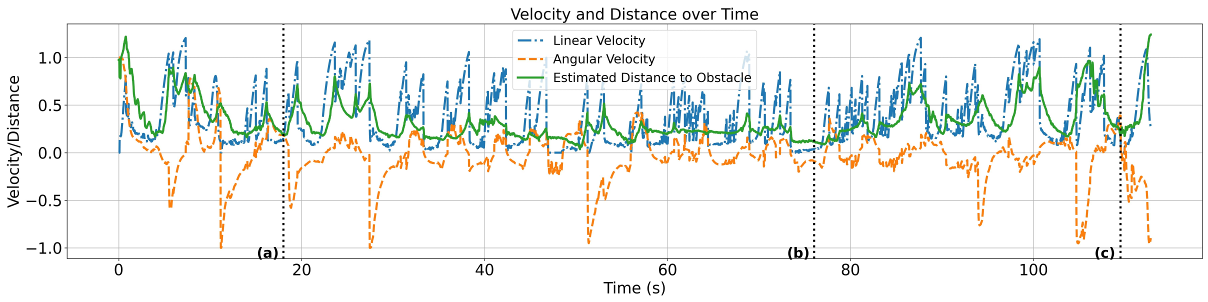


Figure 4.8. Evaluation of our CLF-DR-CBF QP approach in a real lab environment. The top row illustrates challenging scenarios, including (a) navigating around thin chair legs, (b) passing through a narrow passage with pedestrians, and (c) handling an approaching pedestrian. The bottom plot shows the robot's velocity profile and distance to obstacles over time, with 3 vertical dotted lines marking the specific time instances corresponding to the challenging scenarios in the top row.

path was generated assuming the nominal robot shape and did not account for the differences introduced by Shape 1 and Shape 2. Consequently, the original shape demonstrates a minimal deviation from the planned path, whereas Shape 1 exhibits the largest deviations due to its wider, asymmetrical design. Table 4.5 summarizes the results, showing the success, stuck, and collision rates for the three robot shapes under the proposed CLF-DR-CBF QP formulation. In our evaluation, a robot was considered "stuck" if it did not reach its goal within 60 seconds. When the robot shape becomes larger (Shapes 1 and 2), it is more likely to get stuck due to limited maneuverability in tight spaces. This is particularly evident for Shape 1, where the asymmetry makes the robot significantly wider on one side, further complicating its ability to bypass obstacles. Additionally, the larger size of Shapes 1 and 2 reduces their ability to navigate around obstacles within the required time, leading to reduced success rates. Despite these challenges, all three shapes achieved a 0% collision rate, demonstrating the effectiveness of our CLF-DR-CBF QP formulation in maintaining safe navigation.

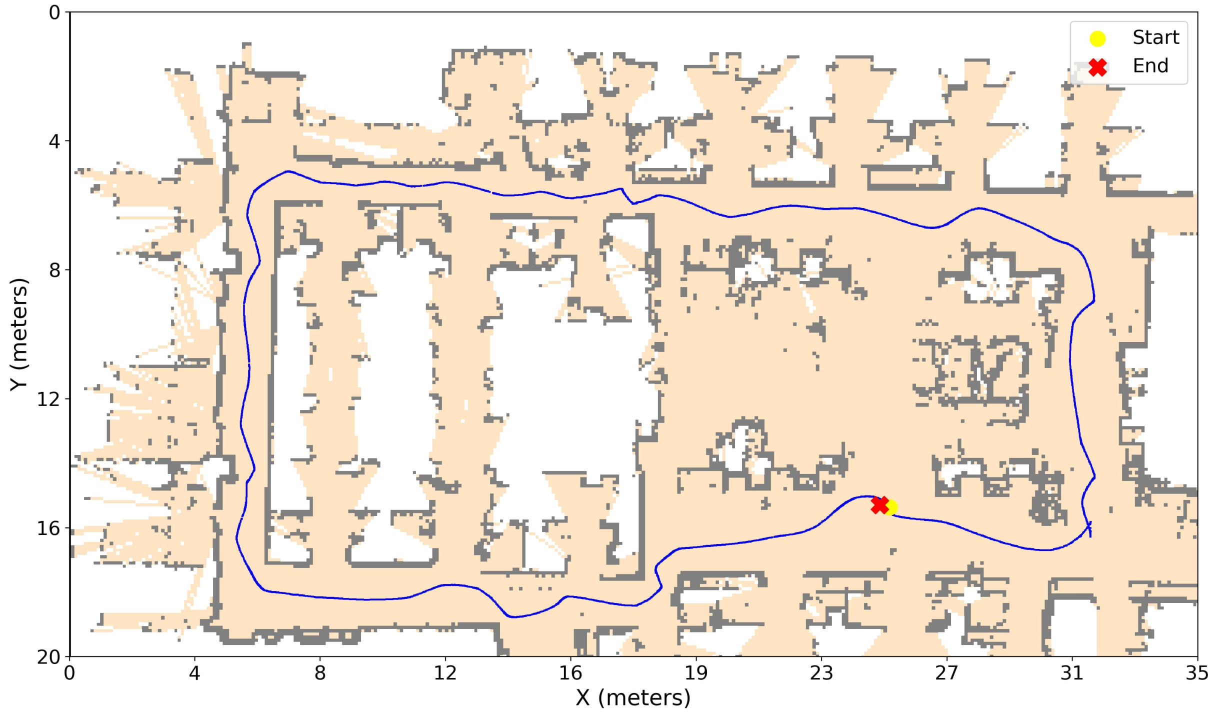
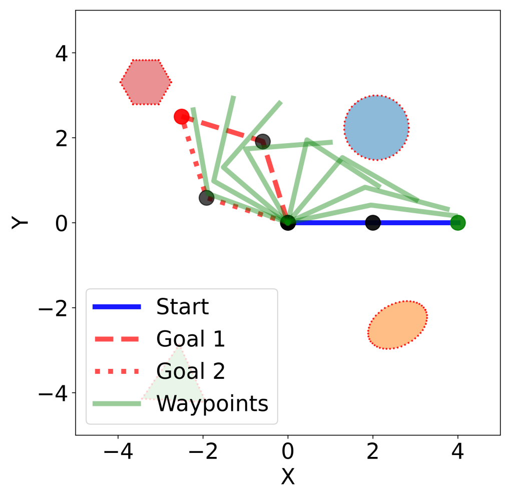


Figure 4.9. Robot trajectory (blue) and estimated occupancy map (yellow and gray) of the lab environment.

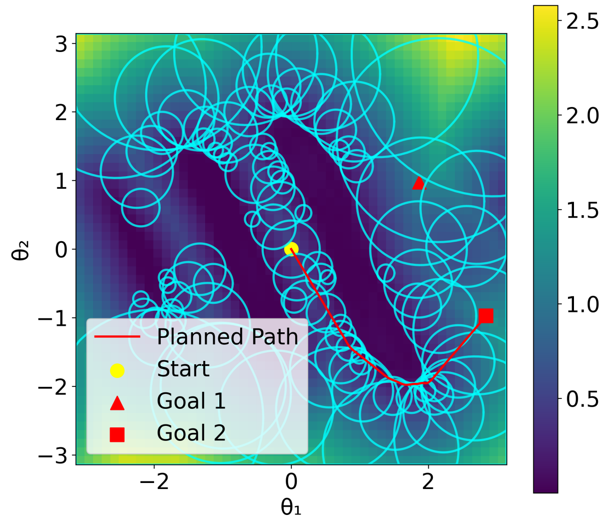
Table 4.5. Performance metrics for real-world experiments with different robot shapes. Metrics include success rate, stuck rate, and collision rate over 50 trials per shape.

Shape	**Success(%)**	**Stuck (%)**	**Collision(%)**
Original	100	0	0
Shape 1	84	16	0
Shape 2	90	10	0

Next, we demonstrated the performance of our CLF-DR-CBF QP formulation using the original robot shape in a full-lab navigation task. The robot successfully handled various real-world challenges, such as thin chair legs, narrow passages with pedestrians, and approaching pedestrians (Fig. 4.8). In contrast to the CLF-GP-CBF SOCP formulation, which relies on GP regression to construct CBFs and cannot handle dynamic environments effectively, our CLF-DR-CBF QP formulation maintains safety while solely depending on noisy LiDAR measurements. The bottom plot in Fig. 4.8 presents the distance to the obstacles and the robot's velocity profile over time, highlighting the robot's ability to maintain a safe distance while efficiently navigating towards its goal.



(a) Workspace visualization



(b) Bubble-CDF planning

Figure 4.10. Neural bubble-CDF planning on a planar 2-link robot arm. (a) The robot's initial configuration (solid blue) and two potential goal configurations (dashed red) are shown in a workspace with four obstacles, represented by their point-cloud surfaces. Intermediate waypoints of the planned trajectory appear in light green. (b) The corresponding 2D configuration space (θ_1 vs. θ_2), where the color map indicates the learned CDF value (darker regions are closer to collision). Cyan circles denote *safe bubbles* derived from the CDF barrier, forming a graph of collision-free regions. A smooth, optimized red trajectory connects the start configuration (yellow circle) to the closest goal (red square); another feasible goal is marked with a red triangle.

Fig. 4.9 depicts the estimated occupancy map and the executed trajectory using our CLF-DR-CBF QP controller. The robot successfully navigates through the cluttered environment, avoiding both static and dynamic obstacles, and reaches its desired goal position.

4.3 Application to Robot Manipulation

While the previous section demonstrated the effectiveness of our distributionally robust framework for mobile robot navigation in workspace coordinates, many robotic applications require safe control in high-dimensional configuration spaces. Robot manipulation presents a particularly challenging domain where traditional CBF approaches face significant computational and modeling difficulties due to the curse of dimensionality and complex kinematic constraints.

In manipulation tasks, robots must navigate through configuration space while avoiding self-collisions and obstacles, often with limited sensing capabilities and uncertain environment geometries. This configuration-space representation introduces additional complexity compared to workspace navigation, as distance computations are more expensive and collision detection requires sophisticated reasoning from workspace to configuration space.

To address these challenges, we extend our distributionally robust CBF framework to robot manipulation by combining configuration-space distance functions with efficient motion planning and real-time safety filtering. As illustrated in Fig. 4.10, our method leverages recent advances in bubble-based configuration-space modeling and integrates them with distributionally robust control to achieve safe manipulation under uncertainty.

4.3.1 Configuration-Space Distance Function

Efficient motion planning and control for robotic manipulators require accurate and scalable safety representations in configuration space. In this section, we first review the notion of a signed distance function (SDF) in the workspace, then introduce the environment and self-collision CDF and their key properties, and finally define a neural CDF barrier for planning and control applications.

Environment Configuration Space Distance Function

Motivated by successful applications in safe mobile robot navigation [59, 77, 101] we discussed in the previous section, SDFs have recently been used for motion planning and control of robot manipulators [81, 91, 100, 155]. An SDF f_s measures the distance from a workspace point $\mathbf{p} \in \mathbb{R}^3$ to the robot surface $\partial\mathcal{S}(\mathbf{q})$:

$$f_s(\mathbf{p}, \mathbf{q}) = \begin{cases} -\min\limits_{\mathbf{p}^* \in \partial\mathcal{S}(\mathbf{q})} \|\mathbf{p} - \mathbf{p}^*\|, & \text{if } \mathbf{p} \in \mathcal{S}(\mathbf{q}), \\ \min\limits_{\mathbf{p}^* \in \partial\mathcal{S}(\mathbf{q})} \|\mathbf{p} - \mathbf{p}^*\|, & \text{if } \mathbf{p} \notin \mathcal{S}(\mathbf{q}). \end{cases} \tag{4.27}$$

When represented using a differentiable model (e.g., neural network [81, 116]), this function enables efficient motion planning and collision avoidance by facilitating rapid computation of distances and gradients. While SDFs are widely applied to mobile robots to enforce workspace safety constraints, manipulators with high-dimensional joint spaces benefit from a configuration-centric perspective.

More recently, researchers [90] introduced a configuration distance function (CDF), which encodes the minimum joint-space distance (in radians) needed for a robot at configuration $\mathbf{q}$ to make contact with a point $\mathbf{p}$. Formally, given a robot SDF f_s, the robot CDF can be computed as

$$f_c(\mathbf{p}, \mathbf{q}) = \min_{\mathbf{q}'} \|\mathbf{q} - \mathbf{q}'\|, \quad \text{subject to} \quad f_s(\mathbf{p}, \mathbf{q}') = 0. \tag{4.28}$$

Similar to an SDF, this CDF representation satisfies an Eikonal equation with respect to $\mathbf{q}$:

$$\|\nabla_{\mathbf{q}} f_c(\mathbf{p}, \mathbf{q})\| = 1, \tag{4.29}$$

whenever it is differentiable in $\mathbf{q}$.

In practice, computing (4.28) requires identifying a set of zero-level configurations $\mathbf{q}'$ satisfying $f_s(\mathbf{p}, \mathbf{q}') = 0$. These configurations are typically obtained using numerical optimization methods such as Broyden-Fletcher-Goldfarb–Shanno (BFGS) algorithm [65]. However, due to the high dimensionality of the configuration space and the sparsity of $\mathbf{q}'$ samples, this direct formulation can lead to an overly smooth or inaccurate approximation of the true CDF. Therefore, [90] leverage the fact that the contact at a point $\mathbf{p}$ is primarily determined by a subset of joints preceding the contact link. Let k denote the robot link that comes into contact with $\mathbf{p}$, and let $\mathbf{q}_{:k}$ represent the joint angles influencing the motion of link k. The CDF is then refined by restricting the distance computation to these relevant joints:

$$f_c(\mathbf{p}, \mathbf{q}) = \min_{k=1,\dots,m} \min_{\mathbf{q}'} \|\mathbf{q}_{:k} - \mathbf{q}'_{:k}\|, \quad \text{s.t.} \quad f_s(\mathbf{p}, \mathbf{q}') = 0, \tag{4.30}$$

where m is the total number of robot joints.

Self-Collision Configuration Space Distance Function

While environment CDFs measure proximity to external obstacles, self-collisions between different parts of the robot pose a distinct challenge, especially for high-DoF manipulators where self-intersections are configuration-dependent and complex. To address this, we define the *Self-Collision Configuration Distance Function* (SCDF), which quantifies the minimum joint-space distance from a configuration $\mathbf{q}$ to the set of self-colliding configurations.

Let $\mathcal{C}_{\text{sc}} \subset \mathcal{Q}$ denote the closed set of joint configurations that result in self-collision. Then, for a given configuration $\mathbf{q} \in \mathcal{Q}$, we define the SCDF as:

$$f_{\text{sc}}(\mathbf{q}) := \min_{\mathbf{q}' \in \mathcal{C}_{\text{sc}}} \|\mathbf{q} - \mathbf{q}'\|. \tag{4.31}$$

Similar to Section 4.3.1, the SCDF in (4.31) represents the Euclidean distance from $\mathbf{q}$ to the closed set of self-colliding configurations $\mathcal{C}_{\text{sc}} \subset \mathbb{R}^m$. Note that $\mathcal{C}_{\text{sc}}$ may consist of multiple disconnected components, as different joint subsets can lead to distinct types of self-collision. Nevertheless, the SCDF inherits the Eikonal property and satisfies

$$\|\nabla_{\mathbf{q}} f_{\text{sc}}(\mathbf{q})\| = 1. \tag{4.32}$$

In practice, $\mathcal{C}_{\text{sc}}$ is often only sparsely sampled, and self-collisions typically involve only a subset of joints. As such, computing the full-joint distance in (4.31) can lead to overly conservative and inaccurate approximations. To improve accuracy and better exploit the robot's kinematic structure, we refine the SCDF by computing distance only over the subset of joints responsible for the self-collision.

Specifically, if a self-collision at $\mathbf{q}'$ occurs between Link i and Link j, we define the joint subvector $\mathbf{q}_{a:b}$, where $a = \min(i, j)$ and $b = \max(i, j)$, to include only joints that affect the relative motion of these links. The refined SCDF becomes:

$$f_{\text{sc}}(\mathbf{q}) := \min_{\mathbf{q}' \in \mathcal{C}_{\text{sc}}} \|\mathbf{q}_{a:b} - \mathbf{q}'_{a:b}\|. \tag{4.33}$$

For example, if the collision occurs between Link 3 and Link 5, then joints 3 through 5 may primarily determine their relative positions, so the distance is computed using $\mathbf{q}_{3:5}$.

Neural CDF Barrier

Building on the environment and self-collision CDF, we define a CDF barrier to ensure the robot remains in a collision-free region of the configuration space. Formally, a CDF barrier associated with environment CDF f_c, self-collision CDF f_{sc}, and obstacle set $\mathcal{O}(t)$ is defined as:

$$h(\mathbf{q}, t) := \min\{\inf_{\mathbf{p} \in \partial\mathcal{O}(t)} f_c(\mathbf{p}, \mathbf{q}), f_{sc}(\mathbf{q})\} \tag{4.34}$$

where $\partial\mathcal{O}(t)$ is the obstacle set boundary. Then, the time-varying *safe set* in configuration space induced by $h(\mathbf{q}, t)$ is

$$\mathcal{C}_{\text{safe}}(t) = \{\mathbf{q} \in \mathcal{Q} \mid h(\mathbf{q}, t) \geq 0\}. \tag{4.35}$$

In real-world settings, $\partial\mathcal{O}(t)$ must be inferred from point-cloud measurements $\mathcal{P}(t)$ from a depth camera or LiDAR. Because the points in $\mathcal{P}(t)$ are noisy samples from the boundary of $\mathcal{O}(t)$, we approximate the CDF barrier as

$$h(\mathbf{q}, t) \;\approx\; \min\{\min_{\mathbf{p} \in \mathcal{P}(t)} f_c(\mathbf{p}, \mathbf{q}), f_{sc}(\mathbf{q})\}. \tag{4.36}$$

Moreover, computing $f_c(\mathbf{p}, \mathbf{q})$ and $f_{sc}(\mathbf{q})$ exactly are difficult for high-DoF manipulators because it involves (infinitely) many potential contact configurations $\mathbf{q}'$ in (4.28) and self-collision configurations in (4.33). Similar to [90], we model f_c and f_{sc} by multi-layer perceptrons $\hat{f}_c(\mathbf{p}, \mathbf{q}; \boldsymbol{\theta_1})$ and $\hat{f}_{sc}(\mathbf{p}, \mathbf{q}; \boldsymbol{\theta_2})$ with learnable parameters $\boldsymbol{\theta_1}$ and $\boldsymbol{\theta_2}$, respectively. This representation reduces the storage requirements compared to volumetric or tabular encodings and supports parallel queries for a set of points $\mathbf{p}$ or configurations $\mathbf{q}$ with associated gradient computations.

However, learning-based representations inevitably introduce modeling errors. Combining the point-cloud approximation with the neural CDF approximation yields a practical CDF barrier:

$$\hat{h}(\mathbf{q}, t; \boldsymbol{\theta}) \ = \ \min\{ \min_{\mathbf{p} \in \mathcal{P}(t)} \hat{f}_c(\mathbf{p}, \mathbf{q};\, \boldsymbol{\theta_1}), \hat{f}_{sc}(\mathbf{q}; \boldsymbol{\theta_2})\} \tag{4.37}$$

where the noise in $\mathcal{P}(t)$, its lack of complete coverage of the obstacle boundary, and the approximation error in $\hat{f}_c$ and $\hat{f}_{sc}$ constitute multiple sources of uncertainty.

In the following sections, we leverage the fast parallel query capabilities of the neural CDF barrier to enable efficient sampling-based planning in configuration space, and then use the developed distributionally robust control strategy in Sec. 4.1 that tracks the planned trajectory while ensuring real-time safety despite uncertainties and dynamic obstacles.

4.3.2 Bubble-CDF Planning

Motion planning for high-DoF manipulators typically involves numerous collision checks [64], making global motion planning computationally expensive. To address this, we propose the **bubble-CDF planner**, a sampling-based approach that efficiently explores the configuration space while ensuring safety via neural CDF barriers. Unlike conventional RRT-based planners that rely on dense edge validation, our approach constructs configuration space bubbles as local safe regions, enabling rapid exploration and collision-free path generation.

The robot must move from an initial configuration $\mathbf{q}_0$ to any valid joint configuration $\mathbf{q}_{\text{G}}$ that achieves a desired end-effector pose $\mathbf{T}_{\text{ee}}(\mathbf{q}_{\text{G}}) = \mathbf{T}_{\text{G}} \in \text{SE}(3)$. Inverse kinematics (IK) provides K goal configurations $\{\mathbf{q}_{\text{G}}^i\}_{i\in[K]}$, each of which achieves the desired end-effector pose.

Our approach builds on the *Rapidly Exploring Bubble Graph (RBG)* algorithm [87], which constructs a configuration-space roadmap where each node represents a locally safe region (bubble) instead of a single configuration. The learned CDF barrier $\hat{h}(\mathbf{q}, t)$ is used to define the bubble radius dynamically, ensuring rapid, collision-free exploration without incremental edge validation. This substantially reduces the collision-checking overhead in high-dimensional spaces while preserving the efficiency of sampling-based planners.

Rapidly-exploring Bubble Graph (RBG)

Suppose $\mathbf{q} \in \mathcal{Q}$ is a safe configuration. We aim to identify a radius $r(\mathbf{q}) \geq 0$ such that any configuration $\mathbf{q}'$ satisfying $\|\mathbf{q} - \mathbf{q}'\| \leq r(\mathbf{q})$ is safe. We call this spherical region a *configuration-space bubble* and denote it as $\mathcal{B}(\mathbf{q}, r(\mathbf{q}))$. Given the neural CDF barrier $\hat{h}(\mathbf{q}, t)$ for fixed time t, we construct the bubble radius at configuration $\mathbf{q}$ as:

$$r(\mathbf{q}) = \hat{h}(\mathbf{q}, t) - \eta, \tag{4.38}$$

where $\eta > 0$ is a safety margin. All configurations $\mathbf{q}'$ within this radius satisfy the neural CDF barrier with margin $h(\mathbf{q}', t) \geq \eta$.

To explore a high-dimensional configuration space efficiently using these local safety certificates, RBG adapts the RRT algorithm [86] so that each vertex in the graph corresponds to a bubble rather than a single configuration. In RBG, the step size, corresponding to the distance the planned tree advances toward a sampled point, is dynamically set by the radius of each bubble. This obviates manual tuning of a fixed step size and reduces collision-checking overhead.

Algorithm 1 details how RBG constructs a bubble-based graph. RBG iteratively samples random configurations (biased toward goal configurations), identifies the nearest bubble in configuration space, and if the sample lies outside that bubble, creates a new bubble by querying the neural CDF barrier to determine its radius. The newly created bubble is then connected to any existing bubbles $\mathcal{B}(\mathbf{q}_i, r(\mathbf{q}_i))$ with overlapping safe regions, i.e., if

$$\|\mathbf{q}_{\text{new}} - \mathbf{q}_i\| \;\leq\; r(\mathbf{q}_{\text{new}}) + r(\mathbf{q}_i). \tag{4.39}$$

This process continues until the maximum number of bubbles $N_{\max}$ is reached or feasible paths to one goal configuration are identified.

Algorithm 1. Rapidly Exploring Bubble Graph

```
Require: Start configuration $\mathbf{q}_0$, goal configurations $\{\mathbf{q}_G^i\}_{i=1}^K$, Neural CDF barrier $\hat{h}(\mathbf{q}, t)$, safety margin $\eta$, Max. no. of bubbles $N_{\max}$, Min. radius $r_{\min}$.
Ensure: Graph $\mathcal{G}$ containing bubbles from start to goals
 1: function BuildBubbleGraph($\mathbf{q}_0, \{\mathbf{q}_G^i\}$)
 2:     $\mathcal{G}.V \leftarrow \{\mathcal{B}(\mathbf{q}_0, \hat{h}(\mathbf{q}_0, t) - \eta)\}, \quad \mathcal{G}.E \leftarrow \emptyset$
 3:     while $|\mathcal{G}.V| < N_{\max}$ do
 4:         $\mathbf{q}_{\text{rand}} \leftarrow$ SampleRandom()                                  ▷ With goal bias
 5:         $\mathcal{B}_{\text{near}} \leftarrow$ NearestBubble($\mathcal{G}, \mathbf{q}_{\text{rand}}$)
 6:         if $\mathbf{q}_{\text{rand}}$ outside $\mathcal{B}_{\text{near}}$ then
 7:             $\mathbf{q}_{\text{new}} \leftarrow$ ExtendToward($\mathcal{B}_{\text{near}}, \mathbf{q}_{\text{rand}}$)
 8:             $r_{\text{new}} \leftarrow \hat{h}(\mathbf{q}_{\text{new}}, t) - \eta$
 9:             if $r_{\text{new}} > r_{\min}$ then
10:                 $\mathcal{B}_{\text{new}} \leftarrow \mathcal{B}(\mathbf{q}_{\text{new}}, r_{\text{new}})$
11:                 UpdateConnections($\mathcal{G}, \mathcal{B}_{\text{new}}$)
12:     return $\mathcal{G}$
13: function UpdateConnections($\mathcal{G}, \mathcal{B}_{\text{new}}$)
14:     for $\mathcal{B}_i \in \mathcal{G}.V$ do
15:         if $\|\mathcal{B}_{\text{new}}.\mathbf{q} - \mathcal{B}_i.\mathbf{q}\| \leq r(\mathbf{q}_{\text{new}}) + r(\mathbf{q}_i)$ then
16:             $\mathcal{G}.E \leftarrow \mathcal{G}.E \cup (\mathcal{B}_{\text{new}}, \mathcal{B}_i)$
17:     $\mathcal{G}.V \leftarrow \mathcal{G}.V \cup \{\mathcal{B}_{\text{new}}\}$
```

Path Selection and Trajectory Optimization

Once the bubble-based graph $\mathcal{G}$ is constructed, we must extract and refine a collision-free path connecting the start configuration $\mathbf{q}_0$ to one of the K goal configurations $\{\mathbf{q}_G^i\}$. Since $\mathcal{G}$ may contain multiple feasible routes, we first select an optimal path according to a desired criterion (e.g., minimum total edge distance). We then solve a local trajectory optimization problem, fitting a continuous curve through the sequence of bubbles to ensure a smooth, dynamically feasible trajectory that remains collision-free.

Path Selection. We assign a cost to each edge in $\mathcal{G}$, for instance, the single-sided Hausdorff distance between two overlapping bubbles [87]. A shortest-path search (e.g., Dijkstra [41] or A* [63]) is run from the start bubble to every goal bubble. The path with the lowest overall cost is then passed to the optimization step, as summarized in Algorithm 2.

Trajectory Optimization. To obtain a smooth trajectory, we optimize a sequence of Bézier curves connecting the start configuration $\mathbf{q}_0$ to the selected goal configuration $\mathbf{q}_G^\star$ through the path of bubbles $\{\mathcal{B}_1, \dots, \mathcal{B}_n\}$. Each Bézier curve segment of degree d is defined by its control points:

$$\gamma_j(t) = \sum_{l=0}^{d} \binom{d}{l} t^l (1-t)^{d-l} \mathbf{c}_{j,l}, \tag{4.40}$$

where $\mathbf{c}_{j,l} \in \mathbb{R}^m$ is the l-th control point of the j-th curve segment. The k-th derivative of a Bézier curve is also a Bézier curve, and its squared norm integral can be expressed as:

$$\int_0^1 \|\gamma_j^{(k)}(t)\|^2 dt = \mathbf{c}_j^\top \mathbf{Q}_k \mathbf{c}_j, \tag{4.41}$$

where $\mathbf{c}_j$ is the vectorized form of all control points in segment j, and $\mathbf{Q}_k$ is a positive semidefinite matrix. This leads to the following convex quadratically constrained quadratic program:

$$\begin{aligned}
\min_{\{\mathbf{c}_{j,i}\}} \quad & \sum_{j=1}^{n} \sum_{k=1}^{K} w_k \int_0^1 \|\gamma_j^{(k)}(t)\|^2 dt \\
\text{subject to} \quad & \|\mathbf{c}_{j,l} - \mathbf{q}_j\|^2 \le r(\mathbf{q}_j)^2, \quad \forall j, l \\
& \gamma_j(1) = \gamma_{j+1}(0), \quad j = 1, \dots, n-1 \\
& \gamma_j^{(k)}(1) = \gamma_{j+1}^{(k)}(0), \quad j = 1, \dots, n-1, \ k = 1, 2 \\
& \gamma_1(0) = \mathbf{q}_0, \quad \gamma_n(1) = \mathbf{q}_{\mathrm{G}}^*, \\
& \gamma_1^{(i)}(0) = \mathbf{0}, \quad \gamma_n^{(i)}(1) = \mathbf{0}, \quad i = 1, \dots, d
\end{aligned} \tag{4.42}$$

where $\mathbf{c}_{j,i}$ are the control points of the j-th Bézier curve segment γ_j of degree d, and $\gamma_j^{(k)}$ denotes its k-th derivative. The objective minimizes a weighted sum of integrated squared derivatives, promoting smoothness. The first constraint ensures safety by keeping control points within their respective bubbles, which by the convex hull property of Bézier curves guarantees the entire trajectory remains collision-free. The remaining constraints enforce continuity of position and derivatives between segments, as well as boundary conditions including zero velocity at endpoints. Unlike traditional sampling-based planners that may not have safety guarantees or require additional collision checks for post-optimized trajectory, our formulation provides a

Algorithm 2. Path Selection and Trajectory Optimization

```
Require:
 1: Graph $\mathcal{G}$ with start bubble $\mathcal{B}_{\text{start}}$, goal bubbles $\{\mathcal{B}^i_{\text{goal}}\}_i$
 2: Edge cost function $\textsc{EdgeCost}(\mathcal{B}_i, \mathcal{B}_j)$
 3: Weights $\{w_k\}$ for trajectory cost
Ensure: Smooth collision-free Bézier trajectory from $\mathbf{q}_0$ to some $\mathbf{q}^i_{\text{G}}$
 4: function $\textsc{PathSelectAndOptimize}(\mathcal{G}, \mathcal{B}_{\text{start}}, \{\mathcal{B}^i_{\text{goal}}\})$
 5:     $\textit{bestCost} \leftarrow \infty, \quad \textit{bestPath} \leftarrow \textit{None}, \quad i_{\text{best}} \leftarrow -1$
 6:     for $i = 1 \to K$ do
 7:         $\textit{path}_i \leftarrow \textsc{ShortestPath}(\mathcal{G}, \mathcal{B}_{\text{start}}, \mathcal{B}^i_{\text{goal}}, \textsc{EdgeCost})$
 8:         if $\textit{path}_i$ exists then
 9:             $\textit{cost}_i \leftarrow \textsc{PathCost}(\textit{path}_i, \textsc{EdgeCost})$
10:             if $\textit{cost}_i < \textit{bestCost}$ then
11:                 $\textit{bestCost} \leftarrow \textit{cost}_i$, $\textit{bestPath} \leftarrow \textit{path}_i$, $i_{\text{best}} \leftarrow i$
12:     if $\textit{bestPath} = \textit{None}$ then
13:         return failure                                        ▷ No feasible path found
14:     curves $\leftarrow \textsc{BezierOpti}(\textit{bestPath}, \{w_k\})$          ▷ Eq. (4.42)
15:     return (curves, $i_{\text{best}}$)
```

convex optimization problem that simultaneously guarantees smoothness and safety without requiring additional collision checking.

Throughout this section, we have introduced the **bubble-CDF planner**, which leverages neural CDF barriers to efficiently explore the configuration space while ensuring safety. By constructing *configuration-space bubbles* around safe configurations and dynamically adjusting step sizes based on bubble radii, our approach significantly reduces collision-checking overhead. Additionally, the trajectory optimization formulation in (4.42) guarantees smooth, dynamically feasible paths without requiring post-processing for collision checks.

Thus far, we have focused on global planning in static environments, assuming a time-invariant and accurate neural CDF barrier $\hat{h}(\mathbf{q}, t)$. However, real-world deployments must contend with dynamic obstacles, CDF modeling errors, and sensor uncertainties. Those will be handled by the distributionally robust safe control approach, introduced in Sec. 4.1.

4.3.3 Simulated and Experimental Validation

This section evaluates our planning and control techniques, which utilize neural CDF barriers, in simulations on 2-DoF planar arm and 6-DoF xArm as well as in real-world experiments on 6-DoF xArm. We evaluate the computational efficiency and solution quality of the bubble-CDF planner and the robustness of the DR-CBF controller in ensuring safety under dynamic obstacles and uncertainty.

Baselines and Parameters

We compare our approach against the following baselines.

- **Planners:**
  - CDF-RRT: RRT with collision checks based on the neural CDF barrier.
  - SDF-RRT [159]: RRT with collision checks using a neural SDF barrier.
  - SDF-RRT-Connect [83]: A variant of SDF-RRT with a bi-directional search strategy.
  - SDF-Lazy-RRT [64]: A variant of SDF-RRT with reduced collision-checking overhead.
- **Controllers:**
  - PD Controller: A proportional-derivative controller $\bar{\mathbf{u}}(\mathbf{q}, t)$ focused solely on trajectory tracking.
  - PD + CBF-QP [25]: A PD controller $\bar{\mathbf{u}}(\mathbf{q}, t)$ augmented with a CBF-QP safety filter, where the CBF is constructed using the neural CDF barrier. However, this approach disregards uncertainty in the CDF model or sensor measurements, unlike ours.

The baseline planners are implemented using the Open Motion Planning Library (OMPL) [135], ensuring consistency across all algorithms. All controllers are implemented using Casadi,

Table 4.6. Planner parameters used for evaluation. **Goal bias** specifies the probability of sampling configurations directly toward the goal. **Step size** defines the maximum extension distance during tree expansion. **Safety margin** indicates the minimum clearance required for collision checking, and **collision check resolution** determines the granularity of edge validity checks, expressed as a fraction of the configuration space diagonal. For the 2-DoF planar robot, a resolution of 0.01 corresponds to 0.089 radians, while for the 6-DoF xArm, a resolution of 0.002 corresponds to 0.04 radians.

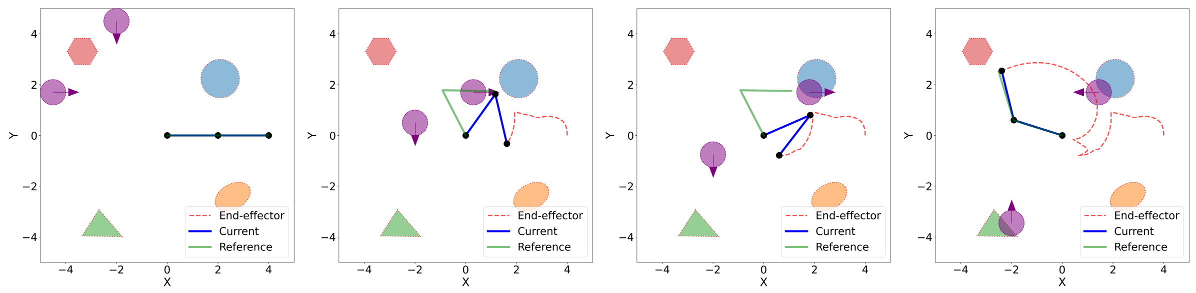
Planner	**Goal Bias**	**Step Size**	**Safety Margin**	**Col. Check Resolution**
Baselines	0.1	0.1	0.05	0.01 (2-DoF) / 0.002 (6-DoF)
Bubble-CDF	0.1	N/A	0.05	N/A

and the CBF-QP and our DR-CBF-QP problems are solved using the interior point optimizer, operating at a control frequency of 50 Hz.

Evaluation Metrics: For planners, we evaluate the number of collision checks and path length. For controllers, the evaluation metrics include success rate (percentage of trials where the robot safely reaches the goal) and tracking error (measured as the Fréchet distance between planned and executed paths).

Parameters: Tables 4.6 summarizes the parameters used for the planners.

2-DoF Planar Robot Simulation



(a) Initial configuration **(b)** Defensive maneuver **(c)** Resuming trajectory tracking **(d)** Goal reached

Figure 4.11. Snapshots of a 2-link arm navigating a dynamic environment with purple obstacles (velocity directions shown by arrows). The robot follows the planned path in Fig. 4.10, shown in red (end-effector trajectory), with the current local reference configuration $\gamma(s)$ shown in green. (a) *Initial configuration*; (b) *Defensive maneuver* to avoid a moving obstacle; (c) *Resuming trajectory tracking*; (d) *Goal reached*.

We begin by evaluating our bubble-CDF planner against baseline sampling-based global motion planning approaches on a 2-DoF planar robotic arm. The arm consists of two links, each

Table 4.7. Planning performance comparison on a 2-DoF planar robot.

Planner	**Col. Checks**	**Path Length**	**Time (s)**
CDF-RRT	2108.9 ± 438.7	3.64 ± 0.69	0.33 ± 0.06
SDF-RRT [159]	2118.3 ± 388.9	3.61 ± 0.69	0.29 ± 0.04
SDF-RRT-Connect	2144.6 ± 459.7	3.63 ± 0.74	0.33 ± 0.05
SDF-Lazy-RRT	1759.2 ± 299.3	3.75 ± 0.83	0.31 ± 0.06
Bubble-CDF	$\mathbf{153.8 \pm 62.2}$	3.72 ± 0.81	$\mathbf{0.13 \pm 0.04}$

2 meters long, with a fixed base located at the origin $(0, 0)$. The joint angles are constrained within the range $[-\pi, \pi)$, and the initial configuration is set to $(0, 0)$, placing the end-effector at $(4, 0)$, as illustrated in Fig. 4.10.

Setup: We assess the performance of the bubble-CDF planner and all baselines in 500 randomly generated environments. Each environment includes 4 static obstacles of varying sizes and positions, and the end-effector goal is randomly sampled to be at least 4 meters away from its initial position while ensuring reachability. The robot receives point-cloud observations of the obstacle surfaces. Due to the 2-DoF planar structure, most reachable end-effector positions correspond to two distinct goal configurations, and the planners are tasked with finding a feasible path from the initial configuration to either one of the two goal configurations.

Results and Discussion: Table 4.7 summarizes the performance metrics across all 500 environments. All planners succeed in finding the path. The results demonstrate the significant efficiency of the bubble-CDF planner in reducing the number of collision checks, reducing the planning time, while maintaining comparable path quality. The bubble-CDF planner requires 153.8 ± 62.2 collision checks, while even the best baseline (Lazy-RRT) requires at least $10\times$ more collision checks. As a result, bubble-CDF achieves the lowest planning time of all methods at 0.13 ± 0.04 seconds. This showcases the ability of our proposed bubble-CDF planner to explore the configuration space more efficiently. Despite this reduction, the bubble-based planner produces paths with similar lengths (3.72 ± 0.81) radians to those of the baselines.

Next, we present our evaluation of the proposed DR-CBF-QP formulation for safe control

Table 4.8. Control performance comparison on a 2-DoF planar robot.

Controller	**Static**		**Dynamic**	
	Success Rate	**Tra. Error**	**Success Rate**	**Tra. Error**
PD	0.874	0.068 ± 0.017	0.112	0.062 ± 0.027
PD + CBF-QP [25]	0.982	0.131 ± 0.049	0.638	0.217 ± 0.106
PD + DR-CBF-QP	$\mathbf{1.0}$	0.173 ± 0.074	$\mathbf{0.992}$	0.394 ± 0.172

synthesis on a planar robot.

Setup: The controllers are evaluated in both static (Fig. 4.10) and dynamic environments (Fig. 4.11). In both settings, point cloud data of the obstacle surfaces is provided to the robot. For dynamic environments, obstacle velocities are sampled from a normal distribution, $v \sim \mathcal{N}(0.5, 0.1)$ m/s, while the robot is provided with a nominal velocity of $v = 0.5$ m/s for control synthesis. Each controller is evaluated for 500 random trials in both static and dynamic environments.

In both scenarios, the objective is to track the bubble-CDF planned trajectory γ (Fig. 4.10) while avoiding collisions.

Results and Discussion: Table 4.8 presents the control performance across static and dynamic environments. The proposed DR-CBF-QP formulation achieves a 100% success rate in static environments and a high success rate (99.2%) in dynamic environments, outperforming both the PD controller and the standard CBF-QP approach. The PD controller, designed solely for trajectory tracking without obstacle avoidance, exhibits the lowest success rates in both static and dynamic environments. The baseline CBF-QP shows good performance in static scenarios, but its success rate drops significantly in dynamic environments (63.8%), which we attribute to disregarding uncertainties in dynamic obstacle velocities.

While the DR-CBF-QP formulation results in higher tracking errors compared to the baselines, this trade-off reflects its prioritization of safety. The controller adjusts the robot's trajectory to avoid collisions, especially in dynamic environments where safety margins must adapt to uncertainties in obstacle motion and CDF estimates.

Table 4.9. Planning performance comparison on a 6-DoF xArm robot in Pybullet Simulation.

Planner	**Col. Checks**	**Path Length**	**Time (s)**
CDF-RRT	3415.4 ± 803.6	3.11 ± 0.46	0.75 ± 0.23
SDF-RRT [159]	3251.8 ± 782.4	3.07 ± 0.49	0.73 ± 0.20
SDF-RRT-Connect	3437.4 ± 723.2	3.11 ± 0.43	0.71 ± 0.21
SDF-Lazy-RRT	2741.1 ± 737.9	3.15 ± 0.53	0.72 ± 0.22
Bubble-CDF	$\mathbf{278.5 \pm 80.7}$	3.04 ± 0.46	$\mathbf{0.15 \pm 0.06}$

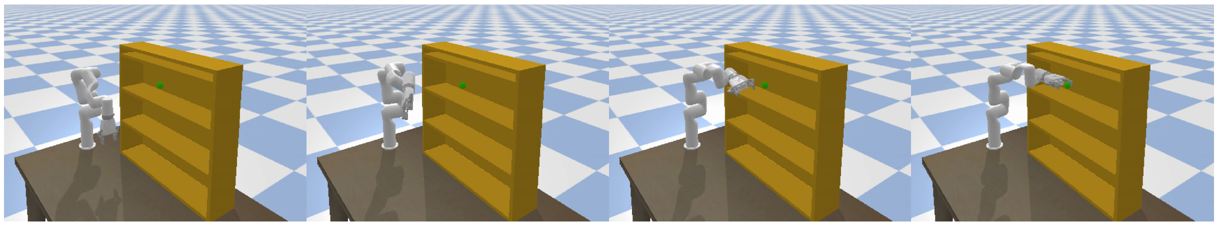
Qualitatively, the defensive maneuvers and adaptability of the DR-CBF-QP formulation are evident in Fig. 4.11. In dynamic environments, the robot effectively modifies its trajectory to avoid collisions, as shown in Fig. 4.11b, where it performs a defensive maneuver to bypass a moving obstacle. It then resumes tracking the planned trajectory (Fig. 4.11c) and successfully reaches the goal configuration (Fig. 4.11d).

6-DoF xArm Robot Simulation

Next, we present simulated experiments for a 6-DoF xArm robotic manipulator in Pybullet [31] with static and dynamic obstacles to further evaluate our proposed approach.

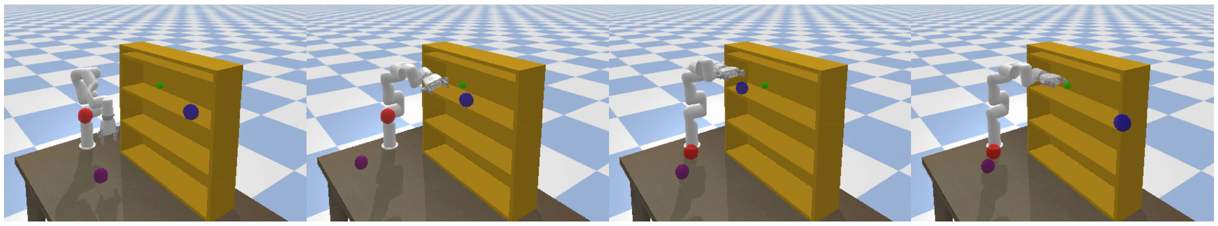
Setup: The simulation environment consists of a 6-DoF xArm robot positioned on a table next to a shelf, as illustrated in Fig. 4.12. A depth camera provides point-cloud observations of the obstacles. We conducted 50 randomized trials by varying the shelf position and selecting different random goal positions. For each trial, inverse kinematics was used to determine five feasible goal configurations, and motion planning was performed in a static environment using these configurations.

To evaluate controller performance, the best planned path from the bubble-CDF planner was selected for tracking in each trial. Additionally, three dynamic obstacles were introduced into the scene, with their positions and velocities varying but assumed to be known to the robot. The velocities of the dynamic obstacles were sampled from a normal distribution, $v \sim \mathcal{N}(0.2, 0.05)$ m/s.



(a) Initial Configuration **(b)** Planned Configuration 1 **(c)** Planned Configuration 2 **(d)** Goal Configuration

Figure 4.12. Bubble-CDF planning for a 6-DoF xArm robot in a static environment, targeting an end-effector goal represented by a green sphere. (a) The initial configuration of the xArm. (b, c) Intermediate configurations illustrating the planned path as the robot avoids obstacles while progressing toward the goal. (d) The final goal configuration reached by the robot.



(a) Start of Execution **(b)** Dynamic Obstacle Approaching **(c)** Defensive Maneuver **(d)** Goal Reached

Figure 4.13. Snapshots of safe control execution on a 6-DoF xArm robot in an environment with dynamic obstacles. (a) The control execution begins at the initial configuration. (b) A dynamic obstacle (blue) approaches the robot from right. (c) The robot executes a defensive maneuver, moving upward to avoid the obstacle. (d) The robot successfully resumes tracking and reaches the goal configuration.

Results and Discussion: Table 4.9 summarizes the planning results for the 6-DoF xArm robot. Similar to the 2-DoF case, the bubble-CDF planner significantly outperforms baseline methods in terms of collision checks and planning time. Path lengths across all planners remain comparable, indicating that the bubble-CDF approach does not compromise solution quality.

Figure 4.12 illustrates the planned path generated by the bubble-CDF planner in a static environment for a specific end-effector goal inside the shelf. Among the five possible goal configurations derived from inverse kinematics, the planner selects the shortest collision-free path.

Table 4.10 quantitatively compares the control performance of different controllers under both static and dynamic environments. The baseline PD controller achieves a success rate of 68% in the static environment but struggles significantly in dynamic scenarios, with a success rate dropping to 18%, highlighting its inability to effectively react to moving obstacles.

Table 4.10. Control performance comparison on a 6-DoF xArm robot in Pybullet Simulation.

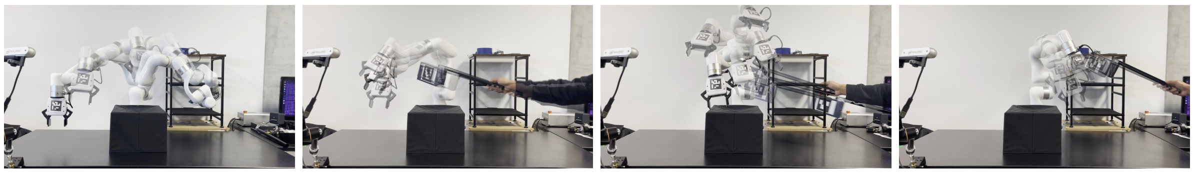
Controller	**Static**		**Dynamic**	
	Success Rate	**Tra. Error**	**Success Rate**	**Tra. Error**
PD	0.68	0.053 ± 0.014	0.18	0.051 ± 0.013
PD + CBF-QP [25]	0.88	0.115 ± 0.046	0.70	0.327 ± 0.104
PD + DR-CBF-QP	1.0	0.143 ± 0.055	1.0	0.516 ± 0.221

Table 4.11. Planning performance comparison on a real 6-DoF xArm robot.

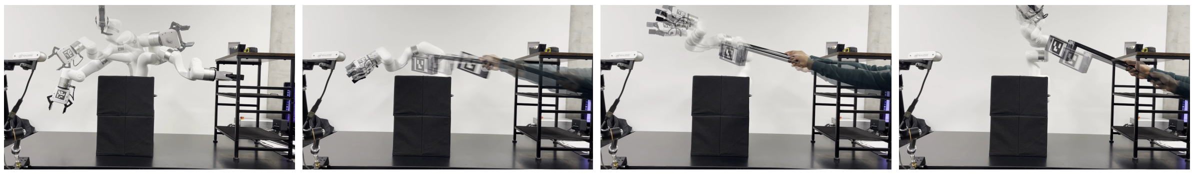
Planner	**Col. Checks**	**Path Length**	**Time (s)**
CDF-RRT	5617.4 ± 1333.6	3.71 ± 0.71	1.02 ± 0.33
SDF-RRT [159]	5911.9 ± 1487.7	3.72 ± 0.68	1.04 ± 0.33
SDF-RRT-Connect	5801.5 ± 1265.4	3.75 ± 0.63	1.05 ± 0.36
SDF-Lazy-RRT	4765.8 ± 1331.4	3.84 ± 0.69	0.96 ± 0.32
Bubble-CDF	$\mathbf{634.8 \pm 225.1}$	3.81 ± 0.66	$\mathbf{0.22 \pm 0.08}$

The addition of a CBF-QP safety filter improves the success rate to 88% in static environments. However, in dynamic scenarios, its success rate remains limited at 70%, suggesting that while CBF constraints enhance obstacle avoidance, they do not account for errors in CDF modeling and point-cloud observations. In contrast, our proposed PD + DR-CBF-QP achieves a 100% success rate in both static and dynamic environments, demonstrating its robustness against dynamic obstacles and uncertainty. The increased tracking error reflects the adaptive nature of our approach, which prioritizes safety by dynamically adjusting the trajectory to avoid obstacles, even if it deviates from the originally planned path.

Figure 4.13 illustrates the execution of a planned path by the xArm robot in a dynamic environment. The robot begins at the initial configuration (Fig. 4.13a) and dynamically reacts to an approaching obstacle (blue) as shown in Fig. 4.13b. By executing a defensive maneuver (Fig. 4.13c), the robot moves upward to maintain safety and avoid a collision. After the obstacle passes, the robot resumes trajectory tracking and successfully reaches the goal configuration (Fig. 4.13d).



(a) Planned configurations **(b)** Response to dynamic obstacle **(c)** Response to dynamic obstacle **(d)** Response to dynamic obstacle



(e) Planned configurations **(f)** Response to dynamic obstacle **(g)** Response to dynamic obstacle **(h)** Response to dynamic obstacle

Figure 4.14. Bubble-CDF planner and DR-CBF control applied to two real-world setups for a 6-DoF xArm robot. The top row (a-d) represents Setup 1, with the robot navigating a cluttered environment featuring a combination of static and dynamic obstacles. Similarly, the bottom row (e-h) depicts Setup 2, showcasing the planner's adaptability in a different obstacle layout. For both setups: (a, e) illustrate the bubble-CDF planned configuration in static environments, while (b-d, f-h) demonstrate the robot's real-time adaptive responses to dynamic obstacles.

6-DoF xArm Robot Experiments

To further validate the efficiency of the bubble-CDF planner and the robustness of the DR-CBF controller, we conducted real-world experiments on a 6-DoF xArm manipulator in several cluttered and dynamic environments.

Setup: The experimental environment consists of a 6-DoF xArm robot operating on a table in a cluttered workspace with static obstacles, as shown in Fig. 4.14. A depth camera provides point-cloud observations of the scene. We conducted 20 randomized trials by varying obstacle placements and selecting different goal configurations. Planning was performed assuming the static environment. During execution, the robot followed the planned path while dynamic obstacles were introduced. The velocity of the moving objects is estimated using ArUco markers.

Results and Discussion: Table 4.11 presents a quantitative comparison of planning performance across different methods. The bubble-CDF planner significantly reduces the number of collision checks compared to all baselines, achieving a nearly tenfold reduction. This improvement stems from the use of configuration-space bubbles, which certify local collision-free regions and minimize the need for frequent edge-based collision checking. Consistent with simulation results, the generated path lengths remain comparable across all methods, demonstrating that our approach maintains solution quality while substantially improving efficiency.

Figure 4.14 shows snapshots of the bubble-CDF planner and DR-CBF controller in two different workspace arrangements. The robot exhibits similar behavior to the simulation, following the nominal planned trajectory (Figs. 4.14a, 4.14e) but stopping and avoiding dynamic obstacles as they are presented.

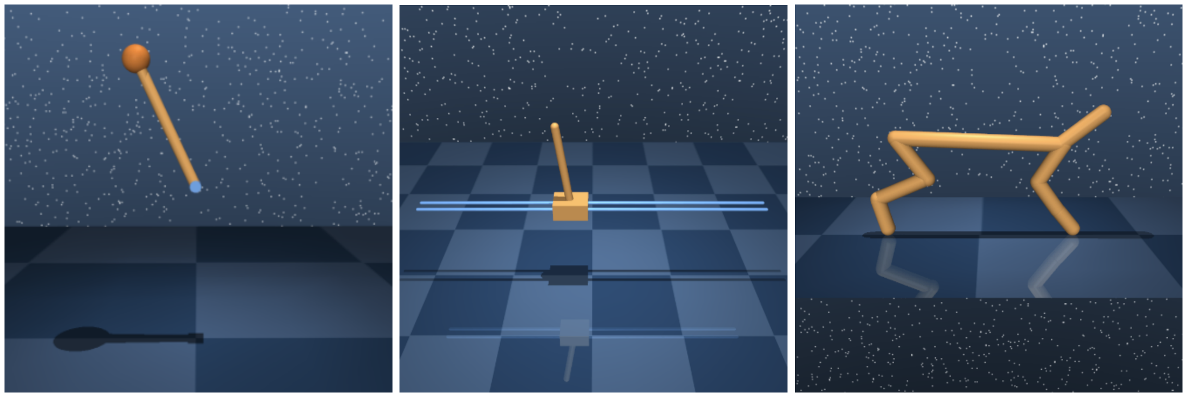
Chapter 4, in full, is a reprint of the material as it appears in Safe and Stable Control Synthesis for Uncertain System Models via Distributionally Robust Optimization, K. Long, Y. Yi, J. Cortés, and N. Atanasov, American Control Conference, 2023; Sensor-Based Distributionally Robust Control for Safe Navigation in Dynamic Environments, K. Long, Y. Yi, Z. Dai, S. Herbert, J. Cortés, and N. Atanasov, International Journal of Robotics Research, 2025; and Neural Configuration-Space Barriers for Manipulation Planning and Control, K. Long, K. M. B. Lee, N. Raicevic, N. Attasseri, M. Leok, and N. Atanasov, under review, 2025. The dissertation author was the primary researcher and author of these works.

Chapter 5

Stability Certification and Lyapunov-Stable Policy Learning

While the previous chapter (Chapter 4) established distributionally robust optimization as a powerful framework for ensuring safety under uncertainty, safety guarantees alone are insufficient for autonomous robotic systems. Beyond avoiding harm, robots must also demonstrate stability—the fundamental property that desired outcomes will eventually be achieved and maintained despite disturbances and uncertainties. Stability certification ensures that a robotic system will converge to target configurations, maintain desired trajectories, and recover from perturbations in a predictable manner. Without such guarantees, even a perfectly safe robot may fail to accomplish its intended tasks, oscillating indefinitely around targets or exhibiting unpredictable long-term behavior that undermines mission objectives.

The challenge of stability analysis becomes particularly acute for modern neural network-based controllers. While deep reinforcement learning and imitation learning policies can achieve remarkable performance on complex control tasks, their black-box nature resists traditional Lyapunov-based stability analysis that forms the theoretical foundation of classical control theory. Traditional Lyapunov methods rely on finding scalar energy-like functions that decrease along system trajectories, but constructing such functions for high-dimensional neural network policies with complex nonlinear activation patterns becomes computationally intractable. This challenge is compounded by the inherent uncertainties present in real-world robotic systems, where model mismatch, environmental variations, and measurement noise can destabilize controllers that appear stable under idealized training conditions.



(a) Inverted Pendulum **(b)** Cart Pole **(c)** Half Cheetah

Figure 5.1. DeepMind Control Suite environments and tasks used for stability analysis and neural policy learning validation.

This chapter addresses the fundamental challenge of stability certification under uncertainty through two complementary theoretical developments. First, we extend our distributionally robust optimization framework to Lyapunov stability analysis, developing probabilistic notions of stability that can handle uncertain systems without requiring exact knowledge of uncertainty distributions. Building upon the Wasserstein ambiguity sets and mathematical foundations established in the previous chapter, we formulate distributionally robust chance-constrained formulations for Lyapunov function synthesis. This approach naturally accommodates the complex, interacting uncertainties present in real robotic systems while providing finite-sample stability guarantees that account for the limited data available during system identification and validation.

Our distributionally robust Lyapunov theory extends traditional stability analysis by reasoning over families of plausible uncertainty distributions rather than assuming perfect knowledge of system dynamics. We develop both sum-of-squares programming approaches for polynomial systems and neural network-based methods for more general nonlinear dynamics. The sum-of-squares formulation provides theoretical guarantees through convex optimization, while the neural network approach offers greater expressiveness and scalability to high-dimensional systems. Both methods reformulate the classical Lyapunov conditions to account for distributional uncertainty, ensuring that stability certificates remain valid under distribution shifts and model mismatch.

However, our investigations revealed a fundamental limitation when attempting to jointly synthesize neural policies and their corresponding Lyapunov certificates for challenging control tasks. The optimization landscape becomes prohibitively complex, and policies derived directly from Lyapunov function gradients often exhibit poor performance characteristics. For instance, in the inverted pendulum task shown in Figure 5.1 with tight control constraints, synthesizing a Lyapunov function that certifies stability over the entire state space is nontrivial. Classical Lyapunov-based methods typically fail in such settings, only managing to certify stability for small regions near equilibrium. Moreover, policies derived directly from Lyapunov functions often fail to stabilize the system effectively, becoming trapped in local minima or failing to generate sufficient control authority to achieve stabilization.

This observation motivates the second major contribution of this chapter: a generalized Lyapunov theory specifically designed for certifying the stability of high-performance neural policies learned through reinforcement learning. Rather than attempting the computationally prohibitive joint synthesis of policies and certificates, we develop relaxed notions of Lyapunov functions that can provide meaningful stability guarantees for existing neural controllers post-hoc. This approach recognizes that reinforcement learning algorithms excel at discovering effective control strategies for challenging tasks like those shown in Figure 5.1, and focuses on developing certificate construction methods that can validate the stability properties of these learned policies.

The generalized Lyapunov framework introduces several key theoretical innovations that make stability certification tractable for neural policies operating in high-dimensional state spaces. By relaxing the strict Lyapunov decrease conditions and incorporating distributional robustness directly into the certificate construction process, we can provide stability guarantees for neural controllers under uncertainty. This framework proves particularly effective for challenging control tasks where traditional methods fail to provide meaningful certificates, successfully validating the stability of learned policies across their entire operating domains while accounting for the distributional uncertainties inherent in reinforcement learning. The chapter presents a systematic progression from distributionally robust extensions of classical Lyapunov theory to specialized methods for neural policy certification. We demonstrate how the same mathematical tools that enabled robust safety guarantees can be adapted to address stability concerns, while introducing novel theoretical developments that specifically target the unique challenges posed by neural network controllers. Through validation on both classical control systems and modern reinforcement learning benchmarks, we show that distributionally robust stability certificates can bridge the gap between the high performance of learned neural policies and the theoretical rigor required for safety-critical deployment.

This chapter is based on the papers [96, 97, 103], which collectively advance stability certification for uncertain and learned controllers through the following contributions:

1. **Distributionally robust Lyapunov stability:** We extend Lyapunov theory to uncertain systems using distributionally robust optimization, deriving chance-constrained stability conditions that provide finite-sample probabilistic guarantees without requiring exact knowledge of uncertainty models [103].

2. **Generalized Lyapunov certificates:** We introduce relaxed Lyapunov conditions that enable stability certification for both optimal control policies in linear systems and reinforcement learning policies in nonlinear systems, addressing cases where traditional Lyapunov theory is intractable.

3. **Joint learning of policies and certificates:** We extend both the distributionally robust and generalized Lyapunov frameworks to the joint learning of control policies and certificates, and demonstrate that our approach can learn provably Lyapunov-stable neural policies.

Together, these contributions establish a principled connection between Lyapunov theory, distributionally robust optimization, optimal control, and reinforcement learning, providing new tools for certifying the stability of neural policies for robotic systems.

In this chapter, we consider

Problem 5. Consider a general system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}),$$

or its discrete-time counterpart $\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t)$. The objective is to jointly design a control policy $\pi(\mathbf{x})$ and a stability certificate $V(\mathbf{x})$ such that the closed-loop system is provably stable for all $\mathbf{x}$ in the domain of interest, even in the presence of model uncertainty.

5.1 Distributionally Robust Lyapunov Theory

We start with reviewing some prelims about Lyapunov theory. Consider a dynamical system, $\dot{\mathbf{x}} = f(\mathbf{x})$, with state $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$. Assume $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ is locally Lipschitz and the origin $\mathbf{x} = \mathbf{0}$ is the desired equilibrium , i.e., $f(\mathbf{0}) = \mathbf{0}$. A valid Lyapunov function, ensuring the stability of the origin, satisfies (2.2). If the LF is also radially unbounded ($V(\mathbf{x}) \to \infty$ as $\|\mathbf{x}\| \to \infty$), then its existence implies global asymptotic stability. The second and third conditions in (2.2) are implied by

$$V(\mathbf{x}) - \epsilon\|\mathbf{x}\|_2^2 \geq 0 \text{ and } -\dot{V}(\mathbf{x}) - \epsilon\|\mathbf{x}\|_2^2 \geq 0, \ \forall \mathbf{x} \neq \mathbf{0}, \tag{5.1}$$

for some $\epsilon \in \mathbb{R}_{>0}$. A natural way of imposing non-negativity is by using SOS polynomials. A polynomial $\eta(\mathbf{x})$ of degree $2d$ is called an SOS polynomial if and only if there exist polynomials $s_1(\mathbf{x}), \ldots, s_p(\mathbf{x})$ of degree at most d such that $\eta(\mathbf{x}) = \sum_{i=0}^{p} s_i(\mathbf{x})^2$. Based on the positive-definiteness property of SOS polynomials, [115, 117] proposed the following SOS conditions, which are sufficient to imply (2.2),

$$V(\mathbf{x}) = \sum_{k=0}^{2d} c_k \mathbf{x}^k,\ c_0 = 0; \quad V(\mathbf{x}) - \epsilon\|\mathbf{x}\|_2^2 \in \text{SOS}(\mathbf{x}); \ \ -\dot{V}(\mathbf{x}) - \epsilon\|\mathbf{x}\|_2^2 \in \text{SOS}(\mathbf{x}), \quad (5.2)$$

where $\text{SOS}(\mathbf{x})$ denotes the set of SOS polynomials in variable $\mathbf{x}$. By fixing a polynomial degree d, one can search for an SOS LF using a semidefinite program [85] enforcing (5.2).

We aim to analyze Lyapunov stability for a dynamical system subject to model uncertainty:

$$\dot{\mathbf{x}} = f(\mathbf{x}) + \sum_{i=1}^{m} d_i(\mathbf{x})\xi_i = f(\mathbf{x}) + d(\mathbf{x})\boldsymbol{\xi}, \quad (5.3)$$

where $d_i : \mathbb{R}^n \mapsto \mathbb{R}^n$ is locally Lipschitz. We assume that $d(\mathbf{x}) = [d_1(\mathbf{x}), \ldots, d_m(\mathbf{x})] \in \mathbb{R}^{n\times m}$ is known or estimated from state-control trajectories [44, 62]. We do not assume any known error bounds or distribution for the parameter $\boldsymbol{\xi} \in \Xi \subseteq \mathbb{R}^m$. Instead, we consider a finite data set of samples $\{\boldsymbol{\xi}_i\}_{i=1}^N$ that may be used for LF synthesis. The uncertainty model in (5.3) captures the commonly considered additive disturbance, which in our formulation corresponds to $m = n$ and $d(\mathbf{x}) = \boldsymbol{I}_n$. The matrix $d(\mathbf{x})$ allows specifying particular system modes affected by the disturbance $\boldsymbol{\xi}$ depending on the state $\mathbf{x}$.

Problem 6 (Lyapunov Function Search For Uncertain Systems). Given a finite set of uncertainty samples $\{\boldsymbol{\xi}_i\}_{i=1}^N$ from the uncertain system in (5.3), obtain a Lyapunov function $V : \mathbb{R}^n \mapsto \mathbb{R}$ that can be used to verify the stability of the origin while taking the uncertainty into account.

We present an SOS approach (Sec. 5.1.1) and a neural network approach (Sec. 5.1.2) to address Problem 6. Our methodology is based on finding a function $V : \mathbb{R}^n \mapsto \mathbb{R}$ that satisfies the Lyapunov conditions in (2.2). The uncertainty in the dynamical system (5.3) appears in the term $\dot{V}(\mathbf{x})$, which presents a challenge for ensuring that the condition $\dot{V}(\mathbf{x}) < 0$, $\forall \mathbf{x} \neq \mathbf{0}$ is satisfied.

5.1.1 Sum-of-Squares Search

We first introduce our SOS approach for LF synthesis under model uncertainty. The Lyapunov conditions in (5.1), taking the uncertainty in (5.3) into account, become:

$$V(\mathbf{0}) = 0;\ \forall \mathbf{x} \neq \mathbf{0},\ V(\mathbf{x}) - \epsilon\|\mathbf{x}\|_2^2 \geq 0 \text{ and } \mathbb{P}^*(-\dot{V}(\mathbf{x}, \boldsymbol{\xi}) - \epsilon\|\mathbf{x}\|_2^2 \geq 0) \geq 1 - \beta, \tag{5.4}$$

where $\mathbb{P}^*$ denotes the true distribution of $\boldsymbol{\xi}$.

To simplify the presentation, let $G(\mathbf{x}, \boldsymbol{\xi}) = \dot{V}(\mathbf{x}, \boldsymbol{\xi}) + \epsilon\|\mathbf{x}\|_2^2 = \nabla V(\mathbf{x})^\top (f(\mathbf{x}) + d(\mathbf{x})\boldsymbol{\xi}) + \epsilon\|\mathbf{x}\|_2^2$, so that the chance-constraint in (5.4) becomes $\mathbb{P}^*(-G(\mathbf{x}, \boldsymbol{\xi}) \geq 0) \geq 1 - \beta$, $\forall \mathbf{x} \neq \mathbf{0}$. Based on the discussion in Sec. 2.3, the CVaR approximation provides a sufficient condition for enforcing the chance constraint:

$$\inf_{t \in \mathbb{R}} \left[\beta^{-1} \mathbb{E}_{\mathbb{P}^*}[(G(\mathbf{x}, \boldsymbol{\xi}) + t)_+] - t\right] \leq 0, \quad \text{for all } \ \mathbf{x} \neq \mathbf{0}. \tag{5.5}$$

If the true distribution $\mathbb{P}^*$ were known, this formulation could be used to deal with the uncertainty. However, we are only provided with samples $\{\boldsymbol{\xi}_i\}_{i=1}^N$ from $\mathbb{P}^*$. We thus rewrite the condition by multiplying by β on both sides and using the empirical expectation to approximate the true expectation,

$$\inf_{t \in \mathbb{R}} \left[\frac{1}{N} \sum_{i=1}^{N} (G(\mathbf{x}, \boldsymbol{\xi}_i) + t)_+ - t\beta\right] \leq 0, \quad \forall \mathbf{x} \neq \mathbf{0}. \tag{5.6}$$

Due to the infimum term in the constraint, one cannot directly write (5.6) as an SOS condition, as in (5.2). The following result provides an alternative SOS condition that ensures (5.6) holds.

Proposition 5.1.1 (CC-SOS Condition)**.** *Assume $\beta \leq \frac{1}{N}$, the constraint in* (5.6) *is equivalent to:*

$$\max_i \beta(\dot{V}(\mathbf{x}, \boldsymbol{\xi}_i) + \epsilon\|\mathbf{x}\|_2^2) \leq 0, \quad \forall \mathbf{x} \neq \mathbf{0}, \tag{5.7}$$

Furthermore, if f and d_i are polynomials, the following N SOS conditions are sufficient for (5.7)*,*

$$-\dot{V}(\mathbf{x}, \boldsymbol{\xi}_i) - \epsilon\|\mathbf{x}\|_2^2 \in \mathrm{SOS}(\mathbf{x}), \quad \forall i = 1, 2 \ldots, N. \tag{5.8}$$

Proof. Denote by t^* the value when the infimum is attained in (5.6). Without loss of generality, we assume that for a given $\mathbf{x}$, $G(\mathbf{x}, \boldsymbol{\xi}_i) \geq G(\mathbf{x}, \boldsymbol{\xi}_j)$, for all $1 \leq i < j \leq N$. Observe that for each $\mathbf{x} \neq \mathbf{0}$, the function $\frac{1}{N}\sum_{i=1}^{N}(G(\mathbf{x}, \boldsymbol{\xi}_i) + t)_+ - t\beta$ is piecewise-linear in t with $N+1$ intervals and N breakpoints, given by $\{-G(\mathbf{x}, \boldsymbol{\xi}_i)\}_{i=1}^{N}$ and the slope for the i-th interval is $\frac{i-1}{N} - \beta$. Thus, the optimal solution is $t^* = -G(\mathbf{x}, \boldsymbol{\xi}_k)$, where k satisfies $\frac{k-1}{N} - \beta < 0$ and $\frac{k}{N} - \beta \geq 0$. The constraint in (5.6) can be rewritten as $\frac{1}{N}\sum_{i=1}^{k}(G(\mathbf{x}, \boldsymbol{\xi}_i) - G(\mathbf{x}, \boldsymbol{\xi}_k)) + \beta G(\mathbf{x}, \boldsymbol{\xi}_k) \leq 0, \quad \forall \mathbf{x} \neq \mathbf{0}$. Since $\beta \leq \frac{1}{N}$, only the first interval has negative slope and this constraint can be written as (5.7). Inspired by the SOS formulation in (5.2), (5.7) is implied by the N SOS constraints in (5.8). □

Using Proposition 5.1.1, we propose a chance-constrained (CC)-SOS formulation to search for a valid Lyapunov function for the uncertain system in (5.3):

$$V(\mathbf{x}) = \sum_{k=0}^{2d} c_k \mathbf{x}^k,\ c_0 = 0;\ V(\mathbf{x}) - \epsilon\|\mathbf{x}\|_2^2 \in \mathrm{SOS}(\mathbf{x});\ -\dot{V}(\mathbf{x}, \boldsymbol{\xi}_i) - \epsilon\|\mathbf{x}\|_2^2 \in \mathrm{SOS}(\mathbf{x}), \tag{5.9}$$

for all $i \in [N]$. Note that by using CVaR approximations in (5.6) and assuming $\beta \leq \frac{1}{N}$, the CC-SOS formulation becomes equivalent to the formulation that is robust against the provided samples $\{\boldsymbol{\xi}_i\}_{i=1}^{N}$, as shown in (5.8). This CC-SOS formulation overcomes the lack of knowledge of the true uncertainty distribution $\mathbb{P}^*$ by using the available samples $\boldsymbol{\xi}_i$ to conservatively approximate the probabilistic constraint in (5.4) with N SOS conditions. Nonetheless, the test-time validity of a Lyapunov function satisfying (5.9) is not guaranteed because the CC-SOS condition does not account for the error between the empirical $\hat{\mathbb{P}}_N$ and the true $\mathbb{P}^*$ distributions. Moreover, the distribution $\mathbb{P}^*$ that generates the uncertainty samples may change at deployment time. This motivates the following distributionally robust chance-constrained formulation:

$$V(\mathbf{0}) = 0;\ \forall \mathbf{x} \neq \mathbf{0},\ V(\mathbf{x}) - \epsilon\|\mathbf{x}\|_2^2 \geq 0 \text{ and } \inf_{\mathbb{P} \in \mathcal{M}_N^r} \mathbb{P}(-\dot{V}(\mathbf{x}, \boldsymbol{\xi}) - \epsilon\|\mathbf{x}\|_2^2 \geq 0) \geq 1 - \beta, \quad (5.10)$$

where $\mathcal{M}_N^r$ denotes the Wasserstein ambiguity set around the empirical distribution $\hat{\mathbb{P}}_N$ with user-defined radius r. Based on the discussion in Sec. 2.3, the following constraint is a sufficient condition for the distributionally robust chance constraint in (5.10) to hold,

$$\sup_{\mathbb{P} \in \mathcal{M}_N^r} \inf_{t \in \mathbb{R}} \left[\mathbb{E}_{\mathbb{P}}[G(\mathbf{x}, \boldsymbol{\xi}) + t)_+] - t\beta\right] \leq 0, \quad \forall \mathbf{x} \neq \mathbf{0}. \quad (5.11)$$

As before, (5.11) is not amenable to a SOS formulation. The following result presents SOS conditions which are sufficient to ensure that (5.11) holds.

Proposition 5.1.2 (DRCC-SOS Condition)**.** *Assume $\beta \leq \frac{1}{N}$, consider the* 1*-Wasserstein distance with L_1 norm as the metric d. The following is a sufficient condition for* (5.11) *to hold,*

$$r \max_{1 \leq j \leq m} |\nabla V(\mathbf{x})^\top d_j(\mathbf{x})| + \max_i \beta(\dot{V}(\mathbf{x}, \boldsymbol{\xi}_i) + \epsilon\|\mathbf{x}\|_2^2) \leq 0, \quad \forall \mathbf{x} \neq \mathbf{0}, \quad (5.12)$$

where $\nabla V(\mathbf{x})^\top d_j(\mathbf{x})$ denotes the j-th element of the row vector. If $\Xi = \mathbb{R}^m$, then (5.12) *is equivalent to* (5.11)*. Also, if f and d_i are polynomials,* (5.12) *is implied by the following SOS conditions,*

$$\pm r \nabla V(\mathbf{x})^\top d_j(\mathbf{x}) - \beta(\dot{V}(\mathbf{x}, \boldsymbol{\xi}_i) - \epsilon\|\mathbf{x}\|_2^2) \in \textit{SOS}(\mathbf{x}),\ \ \forall i = 1, 2 \ldots, N,\ \ \forall j = 1, 2 \ldots, m. \quad (5.13)$$

Proof. Based on [71, Lemma V.8] and [47, Theorem 6.3], the supremum over the Wasserstein ambiguity set, i.e., condition (5.11), can be written conservatively as the sample average $\inf_{t \in \mathbb{R}} \left[\frac{1}{N}\sum_{i=1}^N (G(\mathbf{x}, \boldsymbol{\xi}_i) + t)_+ - t\beta\right]$ and a regularization term $rL_G(\mathbf{x})$, where $L_G(\mathbf{x}) : \mathcal{X} \mapsto \mathbb{R}_{>0}$ is the Lipschitz constant of $G(\mathbf{x}, \boldsymbol{\xi})$ in $\boldsymbol{\xi}$. If $\Xi = \mathbb{R}^m$, then (5.11) is equivalent to the sample average plus $rL_G(\mathbf{x})$. Since the Lipschitz constant of a differentiable affine function equals the dual norm of its gradient, and the dual norm of the L_1 norm is the L_∞ norm, we can define the convex function $L_G : \mathcal{X} \mapsto \mathbb{R}_{>0}$ as $L_G(\mathbf{x}) = \|\nabla V(\mathbf{x})^\top d(\mathbf{x})\|_\infty = \max_{1\leq j\leq m} |\nabla V(\mathbf{x})^\top d_j(\mathbf{x})|$, which satisfies the property that $\boldsymbol{\xi} \mapsto G(\mathbf{x}, \boldsymbol{\xi})$ is Lipschitz in $\boldsymbol{\xi}$ with Lipschitz constant $L_G(\mathbf{x})$. With the assumption that $\beta \leq \frac{1}{N}$, we use Proposition 5.1.1 and conclude that (5.12) is a sufficient condition for (5.11) and they are equivalent if $\Xi = \mathbb{R}^m$. Finally, inspired by the SOS relaxations of (2.2) to (5.2), we can relax (5.12) to the $2Nm$ SOS constraints in (5.13). □

Based on Proposition 5.1.2, we propose a DRCC-SOS formulation to find a Lyapunov function,

$$V(\mathbf{x}) = \sum_{k=0}^{2d} c_k \mathbf{x}^k,\ c_0 = 0;\ \ V(\mathbf{x}) - \epsilon\|\mathbf{x}\|_2^2 \in \text{SOS}(\mathbf{x}); \tag{5.14}$$

$$\pm\, r[\nabla V(\mathbf{x})]^\top d_j(\mathbf{x}) - \beta(\dot{V}(\mathbf{x}, \boldsymbol{\xi}_i) - \epsilon\|\mathbf{x}\|_2^2) \in \text{SOS}(\mathbf{x}),\ \ \forall i = 1, 2 \ldots, N,\ \ \forall j = 1, 2 \ldots, m.$$

The next result identifies conditions under which the resulting Lyapunov function solves Problem 6.

The DRCC-SOS formulation (5.14) provides a stability guarantee if there is no uncertainty distributional shift, i.e., $\mathbb{P}^*$ does not shift outside of $\mathcal{M}_N^{r^*}$ at deployment time. However, similar to other SOS approaches, the formulation is restricted to polynomial systems and the non-existence of an SOS LF does not imply the non-existence of other valid LFs. This motivates us to consider next a more general candidate LF candidate, represented as a neural network.

5.1.2 Neural Network Search

We propose a neural network approach that encourages the satisfaction of Lyapunov conditions by minimizing a loss function that quantifies their violation. Consider a neural network Lyapunov function (NN-LF) representation of the form $V_{\boldsymbol{\theta}}(\mathbf{x}) := \|\phi_{\boldsymbol{\theta}}(\mathbf{x}) - \phi_{\boldsymbol{\theta}}(\mathbf{0})\|^2 + \hat{\alpha}\|\mathbf{x}\|^2$, where $\phi_{\boldsymbol{\theta}} : \mathbb{R}^m \mapsto \mathbb{R}$ is a fully-connected neural network with parameters $\boldsymbol{\theta}$ and $\tanh$ activations, and $\hat{\alpha}$ is a user-chosen parameter [50]. By construction, this function is positive definite and

$V_{\boldsymbol{\theta}}(\mathbf{0}) = 0$. We obtain a training set $\mathcal{D}_{\text{LF}} := \{\mathbf{x}_i\}_{i=1}^{M}$ by sampling uniformly from the domain of interest $\mathcal{X}_\delta$ and then minimize the following empirical loss function:

$$\ell_{\text{LF}}(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M}(\dot{V}_{\boldsymbol{\theta}}(\mathbf{x}_i) + \gamma\|\mathbf{x}_i\|)_+, \tag{5.15}$$

where γ is user-defined. This loss encourages a decrease of $V_{\boldsymbol{\theta}}$ along the system trajectories.

To deal with the model uncertainty in (5.3), we develop chance-constrained (CC) NN-LF and distributionally robust chance-constrained (DRCC) NN-LF formulations. In both cases, we also have the offline uncertainty training set $\mathcal{D}_\xi := \{\boldsymbol{\xi}_i\}_{i=1}^{N}$. For the CC-NN-LF formulation, we require

$$\mathbb{P}^*(\dot{V}_{\boldsymbol{\theta}}(\mathbf{x}, \boldsymbol{\xi}) + \gamma\|\mathbf{x}\| \le 0) \ge 1 - \beta, \quad \forall \mathbf{x} \in \mathcal{X}_\delta. \tag{5.16}$$

However, we are only given samples $\mathcal{D}_\xi$ from $\mathbb{P}^*$. Assuming $\beta \le \frac{1}{N}$, similarly to Proposition 5.1.1, we approximate (5.16) conservatively as,

$$\forall \mathbf{x}_i \in \mathcal{D}_{\text{LF}}, \;\; \max_j(\dot{V}_{\boldsymbol{\theta}}(\mathbf{x}_i, \boldsymbol{\xi}_j) + \gamma\|\mathbf{x}_i\|) \le 0.$$

Thus, to aim for the satisfaction of (5.16) for the training set $\mathcal{D}_{\text{LF}}$, we construct the loss function,

$$\ell_{\text{CC-LF}}(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M}(\max_j(\dot{V}_{\boldsymbol{\theta}}(\mathbf{x}_i, \boldsymbol{\xi}_j) + \gamma\|\mathbf{x}_i\|))_+. \tag{5.17}$$

For the DRCC-NN-LF formulation, to account for errors between the empirical distribution $\hat{\mathbb{P}}_N$ and the true distribution $\mathbb{P}^*$ as well as possible distribution shift during deployment, we require:

$$\inf_{\mathbb{P}\in\mathcal{M}_N^r} \mathbb{P}(\dot{V}_{\boldsymbol{\theta}}(\mathbf{x}, \boldsymbol{\xi}) + \gamma\|\mathbf{x}\| \le 0) \ge 1 - \beta, \quad \forall \mathbf{x} \in \mathcal{X}_\delta. \tag{5.18}$$

Note that (5.18) can be tightened in terms of the CVaR approximation as:

$$\sup_{\mathbb{P}\in\mathcal{M}_N^r}\inf_{t\in\mathbb{R}}\left[\mathbb{E}_{\mathbb{P}}(\dot{V}_{\boldsymbol{\theta}}(\mathbf{x},\boldsymbol{\xi})+\gamma\|\mathbf{x}\|+t)_+ - t\beta\right]\leq 0,\ \ \forall\mathbf{x}\in\mathcal{X}_\delta. \tag{5.19}$$

Next, using the uncertainty set $\mathcal{D}_\xi$ and the training dataset $\mathcal{D}_{\text{LF}}$ and assuming $\beta\leq\frac{1}{N}$, similarly to Proposition. 5.1.2, we rewrite the inequality conservatively as (equivalently if $\Xi=\mathbb{R}^m$), $\forall\mathbf{x}_i\in\mathcal{D}_{\text{LF}}$, $r\|\nabla V(\mathbf{x}_i)^\top d(\mathbf{x}_i)\|_\infty+\beta\max_j(\dot{V}_{\boldsymbol{\theta}}(\mathbf{x}_i,\boldsymbol{\xi}_j))+\gamma\|\mathbf{x}_i\|\leq 0$. Thus, we design the following empirical loss function for the DRCC-NN-LF formulation,

$$\ell_{\text{DRCC-LF}}(\boldsymbol{\theta})=\frac{1}{M}\sum_{i=1}^{M}(r\|\nabla V(\mathbf{x}_i)^\top d(\mathbf{x}_i)\|_\infty+\beta\max_j(\dot{V}_{\boldsymbol{\theta}}(\mathbf{x}_i,\boldsymbol{\xi}_j))+\gamma\|\mathbf{x}_i\|)_+. \tag{5.20}$$

The neural network approach, with the novel loss function designs in (5.17) and (5.20), overcomes the issues noted above for the SOS approach. In particular, we do not require the dynamics to be described by polynomials and avoid scalability problems.

Evaluation Results

We apply the SOS approach (Sec. 5.1.1) and the neural network approach (Sec. 5.1.2) to synthesize LFs for a polynomial system and a pendulum system under model uncertainty.

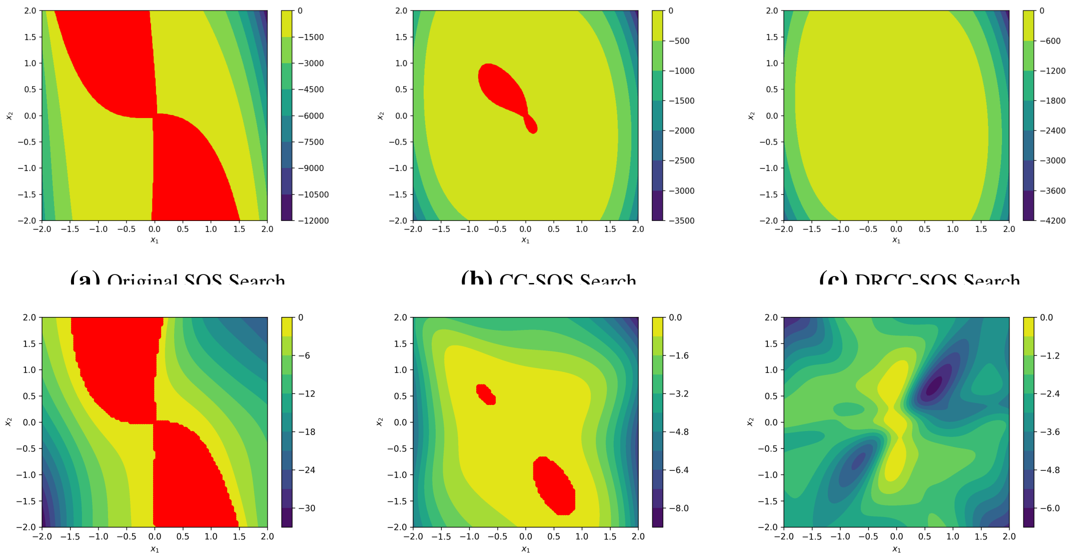
Third-degree Polynomial System: Consider a two-dimensional polynomial system:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}-\frac{1}{2}x_1^3-\frac{3}{2}x_1^2-x_2\\ 6x_1-x_2\end{bmatrix}+\sum_{i=1}^{2}d_i(\mathbf{x})\xi_i, \tag{5.21}$$

with two cases for the model uncertainty:

- Case 1: $r=0.25,\ d_1(\mathbf{x})=-[x_1,x_2]^\top,\ d_2(\mathbf{x})=-[x_2,0]^\top,\ \boldsymbol{\xi}\sim[\mathcal{N}(5,1),\mathcal{N}(3,1)]^\top$.
- Case 2: $r=0.15,\ d_1(\mathbf{x})=-[(x_1^3+x_2),x_2]^\top,\ d_2(\mathbf{x})=-[x_2,x_1]^\top,\ \boldsymbol{\xi}\sim[\mathcal{N}(6,1),\mathcal{N}(0,1)]^\top$.

Suppose that 9 samples $\{\boldsymbol{\xi}_i\}_{i=1}^{9}$ are available offline and set the confidence level to $\beta=0.1$.



(a) Original SOS Search **(b)** CC-SOS Search **(c)** DRCC-SOS Search

(d) Original NN Search **(e)** CC-NN Search **(f)** DRCC-NN Search

Figure 5.2. Results from SOS and NN formulations to design LF certificates for the polynomial system with Case 2 perturbations and online uncertainty $\boldsymbol{\xi}^* = [1.9, 3.0]^\top$. The plots display the value of $\dot{V}$ over the domain, where the red areas indicate positive values (violation of the LF derivative requirements).

We compare the SOS search results with polynomial degree of 4 for the original SOS formulation in (5.2), the CC-SOS formulation in (5.9), and the DRCC-SOS formulation in (5.14). We also include results from the NN formulation in (5.15), the CC-NN formulation in (5.17), and the DRCC-NN formulation in (5.20). For the neural network approach discussed in Sec. 5.1.2, we parametrize $V_{\boldsymbol{\theta}}(\mathbf{x}) = |\phi_{\boldsymbol{\theta}}(\mathbf{x}) - \phi_{\boldsymbol{\theta}}(\mathbf{0})| + \hat{\alpha}\|\mathbf{x}\|$, where $\phi_{\boldsymbol{\theta}}(\mathbf{x})$ is a fully connected three-layer neural network with 2-D input, two 16-D hidden layers, and 1-D output, with $\tanh$ activations. We train the network with the ADAM optimizer [79] with learning rate 0.005 and Xavier initializer, and set the parameter $\hat{\alpha} = 0.05$.

We report qualitative results in Fig. 5.2 for Case 2 with online uncertainty $\boldsymbol{\xi}^* = [1.9, 3.0]^\top$. We uniformly sample $\{\mathbf{x}_i\}_{i=1}^{5000}$ states in the region $x_1, x_2 \in [-2, 2]$. For the first-column plots, the resulting LFs from the baseline SOS and NN formulation fail to satisfy the Lyapunov condition for uncertain systems of the form (5.21), and the violation area is large since neither formulation takes uncertainty into account. For the second-column plots, the resulting LF from the CC-SOS or CC-NN formulation is less sensitive to uncertainty, since both take offline uncertainty samples

Table 5.1. Comparison of Cases 1 and 2 under different online true distributions. Here, "vio. rate" denotes violation rate: (validations with $\dot{V} > 0$)/(total validations), and "vio. area" denotes average violation area over all simulations: (data points with $\dot{V} > 0$)/(total data points). 5000 realizations of the online true uncertainty $\boldsymbol{\xi}^*$ are sampled from uniform and Gaussian distributions: $\boldsymbol{\xi}^* \sim [\mathcal{U}(1,4), \mathcal{U}(1,2)]^\top$ and $\boldsymbol{\xi}^* \sim [\mathcal{N}(4,1.5), \mathcal{N}(1,1.5)]^\top$ for Case 1, $\boldsymbol{\xi}^* \sim [\mathcal{U}(5,7), \mathcal{U}(-1,1)]^\top$ and $\boldsymbol{\xi}^* \sim [\mathcal{N}(7,1), \mathcal{N}(1,1)]^\top$ for Case 2.

Formulations	Case 1 Uniform		Case 1 Gaussian		Case 2 Uniform		Case 2 Gaussian	
	vio. rate	vio. area	vio. rate	vio. area	vio. rate	vio. area	vio. rate	vio. area
SOS	14.28%	0.94%	12.14%	1.53%	100%	15.52%	100%	18.55%
CC-SOS	11.78%	0.89%	8.30%	1.24%	0.00%	0.00%	5.10%	0.04%
DRCC-SOS	0.02%	0.00%	5.24%	0.80%	0.00%	0.00%	1.64%	0.01%
NN	31.80%	1.95%	16.66%	1.65%	100%	17.10%	100%	19.53%
CC-NN	1.82%	0.01%	6.24%	0.72%	0.00%	0.00%	1.26%	0.01%
DRCC-NN	0.00%	0.00%	3.22%	0.38%	0.00%	0.00%	0.72%	0.00%

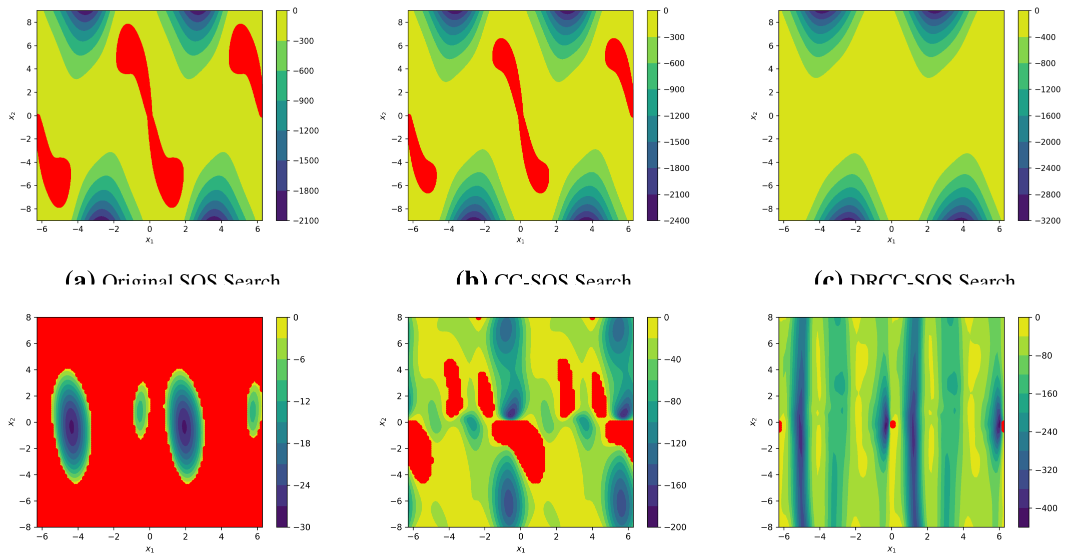
into account. However, the resulting V still fails to satisfy the Lyapunov condition for (5.21). The LF resulting from our DRCC-SOS and DRCC-NN formulations in the last column satisfies the Lyapunov conditions for (5.21), even with out-of-distribution uncertainty. Table 5.1 shows quantitative results. We report the violation rate and average violation area for each of the 6 formulations: in all cases, the DRCC formulations outperform the CC and baseline formulations (no uncertainty considered) using either the SOS or the neural network approach in terms of violation rate and mean violation area.

Pendulum: Consider a pendulum with angle θ and angular velocity $\dot{\theta}$ following dynamics:

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ \frac{-mgl\sin\theta - b\dot{\theta}}{ml^2} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ -\frac{0.05b\dot{\theta}}{ml^2} & -\frac{0.05mgl\sin\theta}{ml^2} \end{bmatrix} \boldsymbol{\xi}, \tag{5.22}$$

where $g = 9.81$ is the gravity acceleration, $m = 1.0$ is the ball mass, $l = 0.5$ is the length, $b = 0.1$ is the damping, and $d(\mathbf{x}) = [d_1(\mathbf{x}), d_2(\mathbf{x})]$ is the perturbation matrix with d_1 and d_2 representing perturbations in damping and length, respectively.

We use 3 offline uncertainty samples $\{\boldsymbol{\xi}_i\}_{i=1}^3$ with $\boldsymbol{\xi}_i \sim [\mathcal{N}(0,1), \mathcal{N}(0,1)]^\top$, and set the confidence level $\beta = 0.1$. The SOS search polynomial is set to have a degree 4 with Wasserstein radius $r = 0.03$. The neural network $\phi_{\boldsymbol{\theta}}$ consists of a fully connected four-layer architecture,



(a) Original SOS Search **(b)** CC-SOS Search **(c)** DRCC-SOS Search

(d) Original NN Search **(e)** CC-NN Search **(f)** DRCC-NN Search

Figure 5.3. Results from SOS and NN formulations to design LF certificates for a pendulum with perturbation in the damping and length and online uncertainty $\boldsymbol{\xi}^* = [-3.6, 1.4]^\top$. The plots display the value of $\dot{V}$ over the domain, where the red areas indicate positive values (violation of the LF derivative requirement).

featuring a 3-D input, three 64-D hidden layers, and a 2-D output. The network employs $\tanh$ activations, and the pendulum state θ is rewritten as two states, $\sin\theta$ and $\cos\theta$. The Wasserstein radius is set to $r = 0.12$. We train the network with the ADAM optimizer with learning rate 0.002 and Xavier initializer, and set the parameter $\hat{\alpha} = 0.5$.

We compare the qualitative results between the SOS-based approaches and the NN-based approaches in Fig. 5.3 with the online uncertainty $\boldsymbol{\xi}^* = [-3.6, 1.4]^\top$. Similar to Fig. 5.2, only the DRCC-SOS and DRCC-NN formulations meet the Lyapunov conditions within the domain of interest. The derivative violations observed near the small neighborhood of the equilibrium in the DRCC-NN formulation are a common issue in neural network-based Lyapunov functions, as reported in previous studies [24, 50].

The approaches presented thus far focus on synthesizing distributionally robust Lyapunov functions for closed-loop systems, whether through sum-of-squares programming or neural network parametrization. While these methods provide stability certificates robust to model uncertainty, they assume the control policy is fixed. In many practical scenarios, however, we have the flexibility to design both the control policy and its corresponding stability certificate simultaneously. This joint design paradigm can potentially yield superior performance by allowing the policy and certificate to be co-optimized.

5.1.3 Joint Synthesis of Lyapunov-Stable Policies and Distributionally Robust Certificates

In this section, we present our results from [97]. We aim to synthesize a Lyapunov-stable controller for the continuous-time system with model uncertainty:

$$\dot{\mathbf{x}} = \bar{\mathbf{f}}(\mathbf{x}, \mathbf{u}, \boldsymbol{\xi}) := \mathbf{f}(\mathbf{x}, \mathbf{u}) + \mathbf{W}(\mathbf{x}, \mathbf{u})\boldsymbol{\xi}, \tag{5.23}$$

where $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n$ denotes the state vector, $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^m$ is the control input, and the functions $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and $\mathbf{W} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n\times k}$ are locally Lipschitz. The set $\mathcal{X} \subset \mathbb{R}^n$ represents the domain of interest, which includes the origin, and $\mathcal{U} \subset \mathbb{R}^m$ denotes the set of admissible control inputs. We assume $\mathcal{X}$ and $\mathcal{U}$ are compact.

The uncertainty structure is characterized by the matrix $\mathbf{W}$, where each column represents a specific direction of uncertainty (e.g., variations in mass or friction parameters). The vector $\boldsymbol{\xi} = [\xi_1, \xi_2, \dots, \xi_k]^\top$ captures the magnitude of uncertainty along these directions, with its true distribution $\mathbb{P}^*$ supported on a compact set $\Xi \subset \mathbb{R}^k$. We assume that $\mathbf{W}(\mathbf{0}_n, \mathbf{0}_m) = \mathbf{0}_{n\times k}$, ensuring the origin remains the desired equilibrium point regardless of the uncertainty realization.

Problem 7 (**Distributionally robust Lyapunov function and controller learning**). Consider the system (5.23) with nominal dynamics $\mathbf{f}$ and uncertainty structure $\mathbf{W}$. Given finitely many samples $\{\boldsymbol{\xi}_i\}_{i\in[N]}$ of the uncertainty $\boldsymbol{\xi}$, design a control policy $\boldsymbol{\pi} : \mathbb{R}^n \to \mathbb{R}^m$ and a corresponding Lyapunov certificate $V : \mathbb{R}^n \to \mathbb{R}$ such that the closed-loop system achieves global asymptotic stability with high confidence, accounting for potential distribution shift between the training samples and the true uncertainty distribution.

To address Problem 7, we develop a distributionally robust formulation that accounts for potential discrepancies between the empirical distribution $\mathbb{P}_N$ and the true distribution $\mathbb{P}^*$ of the system uncertainty $\boldsymbol{\xi}$ at runtime. Our approach seeks a pair $(V^*, \boldsymbol{\pi}^*)$ satisfying the distributionally robust Lyapunov derivative constraint:

$$\inf_{\mathbb{P}\in\mathcal{M}_N^r} \mathbb{P}(\sup_{\mathbf{x}\in\mathcal{X}}(\dot{V}^*(\mathbf{x}, \boldsymbol{\pi}^*(\mathbf{x}), \boldsymbol{\xi}) + \gamma\|\mathbf{x}\|) \leq 0) \geq 1 - \epsilon. \tag{5.24}$$

Compared with the chance-constrained formulation in (2.15), this distributionally robust formulation requires only finite samples instead of the true distribution $\mathbb{P}^*$, while offering robust constraint satisfaction guarantees against potential distribution shifts within the constructed ambiguity set.

We now characterize the stability properties of the closed-loop system. The following results establish that satisfying constraint (5.24) ensures global asymptotic stability with high probability.

Lemma 5.1.3 (Chance-constraint satisfaction under the true distribution). *Assume the distribution $\mathbb{P}^*$ of $\boldsymbol{\xi}$ in* (5.23) *is light-tailed and the Wasserstein radius $r_N(\bar{\epsilon})$ is set according to* (4.6). *If the controller $\boldsymbol{\pi}^*(\mathbf{x})$ and Lyapunov function $V^*(\mathbf{x})$ pair satisfies* (5.24) *with $r = r_N(\bar{\epsilon})$, then,*

$$\mathbb{P}^*(\sup_{\mathbf{x}\in\mathcal{X}}(\dot{V}^*(\mathbf{x}, \boldsymbol{\pi}^*(\mathbf{x}), \boldsymbol{\xi}) + \gamma\|\mathbf{x}\|) \leq 0) \geq (1 - \epsilon)(1 - \bar{\epsilon}). \tag{5.25}$$

Proof. Let $A := \{\mathbb{P}^* \in \mathcal{M}_N^{r_N(\bar{\epsilon})}\}$ and $B := \{\sup_{\mathbf{x}\in\mathcal{X}}(\dot{V}^*(\mathbf{x}, \boldsymbol{\pi}^*(\mathbf{x}), \boldsymbol{\xi}) + \gamma\|\mathbf{x}\|) \leq 0\}$. From [47, Theorem 3.4], $\mathbb{P}^*(A) \geq 1 - \bar{\epsilon}$. From (5.24), $\inf_{\mathbb{P}\in\mathcal{M}_N^{r_N(\bar{\epsilon})}} \mathbb{P}(B) \geq 1 - \epsilon$. Therefore:

$$\begin{aligned} \mathbb{P}^*(B) &\geq \mathbb{P}^*(B \cap A) = \mathbb{P}^*(B|A)\mathbb{P}^*(A) \\ &\geq \left(\inf_{\mathbb{P}\in\mathcal{M}_N^{r_N(\bar{\epsilon})}} \mathbb{P}(B)\right)\mathbb{P}^*(A) \geq (1 - \epsilon)(1 - \bar{\epsilon}) \end{aligned}$$ □

Lemma 5.1.4 (Global asymptotic stability in probability). *Let $V^* : \mathbb{R}^n \to \mathbb{R}$ be positive* *definite with $V^*(\mathbf{0}_n) = 0$, and let V^* and controller $\boldsymbol{\pi}^* : \mathbb{R}^n \mapsto \mathbb{R}^m$ satisfy* (5.25)*. Then the origin of the closed-loop system $\dot{\mathbf{x}} = \bar{\mathbf{f}}(\mathbf{x}, \boldsymbol{\pi}^*(\mathbf{x}), \boldsymbol{\xi})$ is globally asymptotically stable with probability at least $(1-\epsilon)(1-\bar{\epsilon})$.*

Proof. Define $\mathcal{A} := \{\boldsymbol{\xi} \mid \sup_{\mathbf{x}\in\mathcal{X}}(\dot{V}^*(\mathbf{x}, \boldsymbol{\pi}^*(\mathbf{x}), \boldsymbol{\xi}) + \gamma\|\mathbf{x}\|) \leq 0\}$ and $\mathcal{B} := \{\boldsymbol{\xi} \mid$ the system is asymptotically stable at $\mathbf{x} = \mathbf{0}_n\}$. For $\boldsymbol{\xi} \in \mathcal{A}$, we have $\dot{V}^*(\mathbf{x}, \boldsymbol{\pi}^*(\mathbf{x}), \boldsymbol{\xi}) \leq -\gamma\|\mathbf{x}\|$, so V^* decreases along trajectories, ensuring convergence to equilibrium. Thus $\mathcal{A} \subseteq \mathcal{B}$ and $\mathbb{P}^*(\mathcal{B}) \geq \mathbb{P}^*(\mathcal{A}) \geq (1-\epsilon)(1-\bar{\epsilon})$. □

Remark 5.1.5 (Extensions and connections). Under additional conditions, exponential stability can be established by replacing constraint (5.24) with

$$\inf_{\mathbb{P}\in\mathcal{M}_N^r} \mathbb{P}(\sup_{\mathbf{x}\in\mathcal{X}}(\dot{V}^*(\mathbf{x}, \boldsymbol{\pi}^*(\mathbf{x}), \boldsymbol{\xi}) + \gamma\|\mathbf{x}\| + \alpha V^*(\mathbf{x})) < 0) \geq 1 - \epsilon \tag{5.26}$$

for some $\alpha > 0$, and requiring V^* to satisfy appropriate bounds. Our stability notion differs from typical stochastic systems where uncertainty varies over time; since $\boldsymbol{\xi}$ remains fixed in our setting, we achieve stronger convergence guarantees.

Directly optimizing constraint (5.24) is challenging due to the infimum over probability measures and supremum over state space. We leverage distributionally robust optimization techniques to derive a tractable sufficient condition.

Define $h : \mathbb{R}^k \to \mathbb{R}$ as:

$$h(\boldsymbol{\xi}) := \sup_{\mathbf{x}\in\mathcal{X}} \left(\dot{V}(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}), \boldsymbol{\xi}) + \gamma\|\mathbf{x}\|\right), \tag{5.27}$$

where $\dot{V}(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}), \boldsymbol{\xi}) = \nabla V(\mathbf{x})^\top(\mathbf{f}(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \mathbf{W}(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))\boldsymbol{\xi})$. Since $h(\boldsymbol{\xi})$ is the supremum of affine functions in $\boldsymbol{\xi}$, it is convex in $\boldsymbol{\xi}$.

Proposition 5.1.6 (Distributionally Robust Stability Condition). *Let $\{\boldsymbol{\xi}_i\}_{i\in[N]}$ be samples ordered such that $h(\boldsymbol{\xi}_i) \geq h(\boldsymbol{\xi}_k)$ for $1 \leq i < k \leq N$. For $\epsilon \in (0,1)$, let $j \in [N]$ satisfy*

$\frac{j-1}{N} - \epsilon < 0$ *and* $\frac{j}{N} - \epsilon \geq 0$. *Then:*

$$\frac{r}{\epsilon} \sup_{\mathbf{x} \in \mathcal{X}} \|\mathbf{W}^\top(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) \nabla V(\mathbf{x})\| + \frac{1}{N\epsilon} \sum_{i=1}^{j-1} (h(\boldsymbol{\xi}_i) - h(\boldsymbol{\xi}_j)) + h(\boldsymbol{\xi}_j) \leq 0 \tag{5.28}$$

is sufficient for (5.24) *with 1-Wasserstein distance. If* $\epsilon \leq \frac{1}{N}$, *this simplifies to:*

$$\frac{r}{\epsilon} \sup_{\mathbf{x} \in \mathcal{X}} \|\mathbf{W}^\top(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) \nabla V(\mathbf{x})\| + \max_i h(\boldsymbol{\xi}_i) \leq 0. \tag{5.29}$$

This reformulation provides a tractable condition for synthesizing the controller-certificate pair $(V^*, \boldsymbol{\pi}^*)$, enabling practical implementation through neural network optimization as detailed in the next section.

Having established the theoretical foundation and tractable reformulations, we now present a neural network-based approach to learn the distributionally robust controller-certificate pair $(V^*, \boldsymbol{\pi}^*)$ that satisfies constraint (5.28). Our approach leverages the universal approximation capabilities of neural networks to parametrize both the Lyapunov function and controller, enabling practical implementation of the distributionally robust stability framework.

We begin by establishing the necessary assumptions and then detail our neural network architecture and training methodology.

Assumption 5.1.7 (Lipschitz continuity and boundedness). *The nominal dynamics* $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ *and perturbation function* $\mathbf{W} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n \times k}$ *are Lipschitz continuous on* $\mathcal{X} \times \mathcal{U}$ *with constants* L_f *and* L_W, *respectively. Moreover, both functions are uniformly bounded on* $\mathcal{X} \times \mathcal{U}$ *with bounds* B_f *and* B_W.

This assumption ensures that the system dynamics vary smoothly across the state and control spaces, which is essential for our neural network-based stability certificates to generalize from discrete training samples to the continuous state space. We parametrize the Lyapunov function and controller using neural networks with specific structural constraints that ensure the necessary mathematical properties.

Lyapunov Function Parametrization: We define the neural network Lyapunov function as:

$$V_{\boldsymbol{\theta}_1}(\mathbf{x}) := \|\phi_{\boldsymbol{\theta}_1}(\mathbf{x}) - \phi_{\boldsymbol{\theta}_1}(\mathbf{0}_n)\|^2 + \hat{\alpha}\|\mathbf{x}\|^2, \tag{5.30}$$

where $\phi_{\boldsymbol{\theta}_1} : \mathbb{R}^n \mapsto \mathbb{R}$ is a fully-connected neural network with parameters $\boldsymbol{\theta}_1$, $\tanh$ activations, and $\hat{\alpha} > 0$ is a regularization parameter. This construction ensures:

- **Positive definiteness:** $V_{\boldsymbol{\theta}_1}(\mathbf{x}) > 0$ for all $\mathbf{x} \neq \mathbf{0}_n$
- **Zero at origin:** $V_{\boldsymbol{\theta}_1}(\mathbf{0}_n) = 0$
- **Continuous differentiability:** The $\tanh$ activations ensure smooth gradients

Controller Parametrization: The neural network controller is defined as:

$$\boldsymbol{\pi}_{\boldsymbol{\theta}_2}(\mathbf{x}) := \varphi_{\boldsymbol{\theta}_2}(\mathbf{x}) - \varphi_{\boldsymbol{\theta}_2}(\mathbf{0}_n), \tag{5.31}$$

where $\varphi_{\boldsymbol{\theta}_2} : \mathbb{R}^n \mapsto \mathbb{R}^m$ is a neural network with parameters $\boldsymbol{\theta}_2$ and $\tanh$ activations. This ensures $\boldsymbol{\pi}_{\boldsymbol{\theta}_2}(\mathbf{0}_n) = \mathbf{0}_m$, maintaining the equilibrium at the origin.

Due to neural network approximation limitations near the equilibrium, we adopt a δ-accurate stability framework that ensures stability outside a small neighborhood of the origin.

Definition 5.1.8 (Distributionally robust δ-accurate Lyapunov function). A Lyapunov function $V(\mathbf{x})$ for system (5.23) is distributionally robust δ-accurate if it is positive definite, $V(\mathbf{0}_n) = 0$, and satisfies:

$$\inf_{\mathbb{P} \in \mathcal{M}_N^r} \mathbb{P}\big(\sup_{\mathbf{x} \in \mathcal{X}_\delta} (\dot{V}(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}), \boldsymbol{\xi}) + \gamma\|\mathbf{x}\|) \leq 0\big) \geq 1 - \epsilon, \tag{5.32}$$

where $\mathcal{X}_\delta := \mathcal{X} \setminus \overline{B}(\mathbf{0}_n; \delta)$ excludes a δ-radius ball around the origin.

This framework ensures ultimate boundedness of trajectories within the ball $\overline{B}(\mathbf{0}_n; \delta)$, with $\delta > 0$ chosen arbitrarily small.

We construct training datasets $\mathcal{D}_{\text{LF}} := \{\mathbf{x}_i\}_{i\in[M]}$ by uniform sampling from $\mathcal{X}_\delta$ and use the uncertainty samples $\mathcal{D}_{\boldsymbol{\xi}} := \{\boldsymbol{\xi}_i\}_{i\in[N]}$ from offline system measurements.

Nominal Loss (Baseline): For systems without uncertainty, we use:

$$\ell_{\text{Nominal}}(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M}(\dot{V}_{\boldsymbol{\theta}_1}(\mathbf{x}_i, \boldsymbol{\pi}_{\boldsymbol{\theta}_2}(\mathbf{x}_i)) + \gamma\|\mathbf{x}_i\|)_+ \tag{5.33}$$

Distributionally Robust Loss: For uncertain systems with $\epsilon \le \frac{1}{N}$, we use:

$$\begin{aligned}\ell_{\text{DR}}(\boldsymbol{\theta}) = \Bigg(&\frac{r}{\epsilon}\max_{\mathbf{x}_i\in\mathcal{D}_{\text{LF}}}\|\mathbf{W}^\top(\mathbf{x}_i, \boldsymbol{\pi}_{\boldsymbol{\theta}_2}(\mathbf{x}_i))\nabla V_{\boldsymbol{\theta}_1}(\mathbf{x}_i)\| \\ &+\max_j\sum_{i=1}^{M}(\dot{V}_{\boldsymbol{\theta}_1}(\mathbf{x}_i, \boldsymbol{\pi}_{\boldsymbol{\theta}_2}(\mathbf{x}_i), \boldsymbol{\xi}_j) + \gamma\|\mathbf{x}_i\|)\Bigg)_+\end{aligned} \tag{5.34}$$

The condition $\epsilon \le \frac{1}{N}$ is practical for limited data scenarios and ensures stronger stability guarantees by requiring the Lyapunov condition to hold for all uncertainty samples.

We establish that sufficient sampling density ensures the learned neural networks provide the desired stability guarantees.

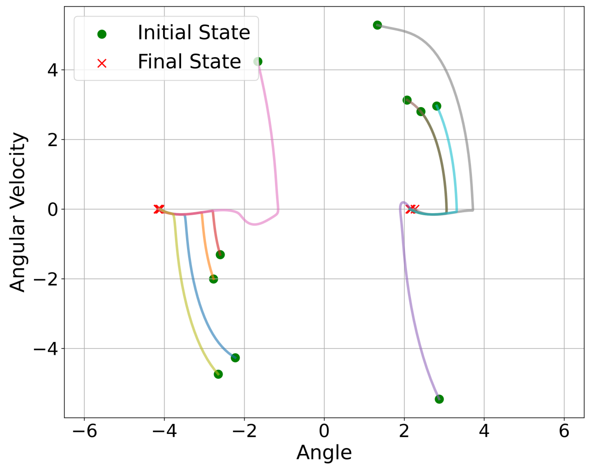
Lemma 5.1.9 (Coverage lemma [50]**).** *For any $\delta > 0$ and $c > 0$, there exists $M(\delta, c) \in \mathbb{N}$ such that for all $M \ge M(\delta, c)$, a uniformly sampled dataset $\mathcal{D}_{LF} \subset \mathcal{X}_\delta$ ensures that any $\mathbf{x} \in \mathcal{X}_\delta$ is within distance $c\|\mathbf{x}_i\|$ of some $\mathbf{x}_i \in \mathcal{D}_{LF}$.*

Proposition 5.1.10 (Distributionally robust neural Lyapunov-stable control). *Let $\mathcal{D}_{LF} \subset \mathcal{X}_\delta$ with $|\mathcal{D}_{LF}| \ge M(\delta, c)$, and let $\boldsymbol{\theta}^*$ achieve $\ell_{DR}(\boldsymbol{\theta}^*) = 0$. If $c > 0$ is sufficiently small such that:*

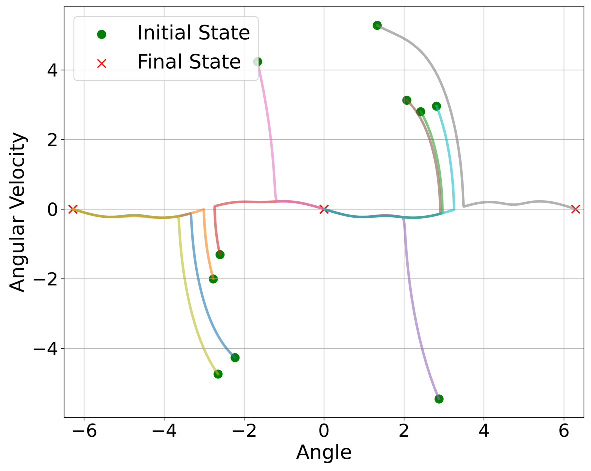
$$\gamma > ((L_W(L_\pi + 1))B_V + L_{\nabla V}B_W)\,B_\xi\frac{r}{\epsilon}c + (L_f(L_\pi + 1)B_V + L_{\nabla V}B_f)c, \tag{5.35}$$

then $\boldsymbol{\pi}_{\boldsymbol{\theta}_2^*}$ stabilizes system (5.23) *to $\overline{B}(\mathbf{0}_n; \delta)$ with high probability, as certified by the distributionally robust δ-accurate Lyapunov function $V_{\boldsymbol{\theta}_1^*}$.*

Evaluation:



(a) Baseline Trajectories



(b) DR Trajectories

Figure 5.4. Comparison of trajectories for the inverted pendulum system with test case parameters: mass $m = 1.1$, length $l = 1.0$, and damping $b = 0.18$. The 10 random sampled initial states are marked as green dots, while the final states are marked as red crosses. The states $(\theta, \dot{\theta}) = (2k\pi, 0)$ for $k \in \mathbb{N}$ are stable equilibrium states, while the points $((2k-1)\pi, 0)$ are unstable equilibrium points, corresponding to the upside-down position of the pendulum. In (a), the baseline controller, trained using the average mass and damping from offline observations, fails to stabilize the pendulum to upright. In (b), the distributionally robust (DR) controller successfully stabilizes the pendulum to an upright position for every initial state, demonstrating improved robustness to distributional shifts in the system parameters.

We then evaluates our approach for synthesizing distributionally robust Lyapunov-stable controllers. We focus on the classic Inverted Pendulum control problem. For the system, we consider multiple instances with varying characteristics, each defined by its own physical parameters such as mass, length, and friction. These instances are used for training our distributionally robust Lyapunov-stable controller.

During testing, we assume the physical parameters of each system may be drawn from distributions different from those of the training instances. This setup aims to assess the performance of the learned controller under realistic scenarios, where distributional shifts in the system parameters are common. By doing so, we effectively evaluate the robustness and adaptability of our controller in the presence of distributional uncertainty in the system models.

The inverted pendulum is a standard nonlinear control problem for testing control methods. The system consists of two state variables, angular position θ and angular velocity $\dot{\theta}$, and one control input u. The system dynamics are:

$$\underbrace{\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix}}_{\dot{\mathbf{x}}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) := \begin{bmatrix} \dot{\theta} \\ \frac{mgl \sin\theta - b\dot{\theta}}{ml^2} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{ml^2} \end{bmatrix} u \tag{5.36}$$

where $g = 9.81$ m/s^2 is the gravity acceleration, $m = 1.0$ kg is the mass, $l = 1.0$ m is the length, and $b = 0.13$ N·m·s/rad is the damping coefficient. The equilibrium state is defined as $[\theta, \dot{\theta}] = [0, 0]$, which corresponds to the upright equilibrium position.

For training both the baseline and the DR controllers, we generate a training dataset by uniformly sampling $\{\mathbf{x}_i\}_{i\in[3600]}$ from the box region defined by $0 \le \theta \le 2\pi$ and $-8 \le \dot{\theta} \le 8$.

We assume uncertainties in mass and damping, represented by ξ_1 and ξ_2, respectively. The uncertain system dynamics are:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\theta} \\ \frac{(m+\xi_1)gl\sin\theta - (b+\xi_2)\dot{\theta}}{(m+\xi_1)l^2} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{(m+\xi_1)l^2} \end{bmatrix} u. \tag{5.37}$$

Through first-order Taylor expansion around the nominal parameters, we derive the perturbation matrices $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2]$ for the inverted pendulum subject to model uncertainty:

$$\mathbf{w}_1(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} 0 \\ \frac{b\dot{\theta} - u}{m^2 l^2} \end{bmatrix}, \quad \mathbf{w}_2(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} 0 \\ -\frac{\dot{\theta}}{ml^2} \end{bmatrix}. \tag{5.38}$$

We assume that a set of offline samples $\{\boldsymbol{\xi}_i\}_{i=1}^5$ of the uncertainties $\boldsymbol{\xi} = [\xi_1, \xi_2]^\top$ are available for training the controller and certificate pairs. We take $\xi_1 \sim \mathcal{U}(-0.04, 0.08)$, $\xi_2 \sim \mathcal{N}(0.0, 0.02)$, where $\mathcal{U}$ and $\mathcal{N}$ denote uniform and normal distributions, respectively. However, during test time, the test uncertainty parameters are set to $\xi_1 = 0.1$ and $\xi_2 = 0.05$. These test values represent a significant deviation from the training distributions, allowing us to evaluate the controllers' robustness to distributional shifts in parameters.

To train the baseline controller, we compute the average mass $\tilde{m}$ and the average damping $\tilde{b}$ from the five offline samples. These average values are then used in the baseline system dynamics and the loss function (5.33) to learn the baseline controller and Lyapunov function pair.

In contrast, for training the DR controller, we set the Wasserstein radius to $r = 0.01$ and the risk tolerance to $\epsilon = 0.1$. The 5 uncertainty samples $\{\boldsymbol{\xi}_i\}_{i=1}^{5}$ are then used in the loss function (5.20) to learn the DR controller and its corresponding Lyapunov function. Fig. 5.4 illustrates the performance of the baseline and DR controllers with the test uncertainty parameters $\xi_1 = 0.1$ and $\xi_2 = 0.05$. To assess the controllers' performance, we randomly sample 10 initial states within the box region defined by $0 \leq \theta \leq 2\pi$ and $-6 \leq \dot{\theta} \leq 6$. Each controller is applied to the system and the resulting trajectories are simulated. In Fig. 5.4a, we observe that the baseline controller, trained using only the average mass and damping values, fails to stabilize the inverted pendulum system to the desired upright equilibrium. For all the 10 initial states, the baseline controller's trajectories converge to states that are not the desired equilibrium. This can be attributed to the increased damping ($\xi_2 = 0.05$) and the heavier pendulum mass ($\xi_1 = 0.1$) in the test scenario. The baseline controller, designed based on the average parameter values, lacks the necessary robustness to compensate for the increased damping and mass.

On the other hand, Fig. 5.4b shows that all trajectories converge to the desired equilibrium under the DR controller. Despite the presence of distributional shift in the model uncertainty, the DR controller successfully stabilizes the inverted pendulum in an upright position. This robustness can be attributed to the DR controller's training process, which explicitly takes into account the distributional information of the uncertainty and optimizes for worst-case performance within the constructed ambiguity set.

We have presented an approach for jointly synthesizing distributionally robust controllers and Lyapunov certificates for nonlinear systems with model uncertainty. Our key contribution is a distributionally robust Lyapunov stability formulation that leverages finite samples of uncertainty parameters to stabilize uncertain systems with theoretical guarantees. Through theoretical analysis and experimental validation on the inverted pendulum, we demonstrated that the learned controller achieves asymptotic stability with high probability, even under out-of-distribution model uncertainties.

However, we observe that co-training Lyapunov functions and policies becomes challenging under tight control constraints. Classical Lyapunov methods typically fail in such settings, certifying stability only in small regions near equilibrium due to insufficient control authority to guarantee negative definiteness of the Lyapunov derivative across the full state space. Policies derived directly from Lyapunov functions often become trapped in local minima or fail to generate adequate control for global stabilization.

In contrast, reinforcement learning algorithms [53, 131] demonstrate remarkable flexibility under constraints, discovering creative strategies that exploit system dynamics—such as strategic pendulum swinging to leverage gravity before stabilization. This suggests the potential value of combining Lyapunov guarantees with the flexibility of learned policies.

5.2 Generalized Lyapunov Theory for Neural Policy Certification

The limitations of classical Lyapunov methods under control constraints motivate exploring alternative stability certification approaches that can handle the practical flexibility of reinforcement learning policies. While the distributionally robust approach presented in the previous section addresses model uncertainty, it still relies on the restrictive requirement of strict pointwise decrease in the Lyapunov function. This constraint often makes it difficult to certify stability for policies learned through optimal control or reinforcement learning, particularly when control authority is limited or when the value function naturally exhibits non-monotonic behavior due to discounting effects.

Building on prior work on generalized Lyapunov functions [48, 49], we develop a comprehensive framework for certifying the stability of learned control policies. Our key insight is that value functions from optimal control and RL, while not directly satisfying classical

Lyapunov conditions, can be systematically augmented with learned residual terms to form valid stability certificates. We replace the strict stepwise decrease requirement with a more flexible multi-step weighted decrease criterion, enabling certification of a broader class of control policies.

Our main contributions in this section include:

- **Theoretical analysis** showing how discounted value functions from LQR can be augmented to satisfy generalized Lyapunov conditions via a new set of linear matrix inequalities that significantly reduce conservatism compared to classical approaches.
- **A practical framework** for certifying nonlinear RL policies by augmenting their value functions with neural network residual terms and learning state-dependent multi-step weights.
- **Joint synthesis methodology** that simultaneously optimizes control policies and their stability certificates, yielding larger certified regions of attraction than classical one-step Lyapunov methods.

This generalized framework offers several key advantages: it allows temporary increases in the Lyapunov function over individual time steps provided the weighted average decreases over a finite horizon, it directly leverages value functions learned by RL algorithms as building blocks for stability certificates, and it enables joint optimization of both control policies and their corresponding certificates. This approach bridges the gap between classical control theory's stability guarantees and modern learning-based control methods' practical effectiveness.

We consider the discrete-time system:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \quad \mathbf{x}_k \in \mathcal{X} \subseteq \mathbb{R}^n, \quad \mathbf{u}_k \in \mathcal{U} \subseteq \mathbb{R}^m, \tag{5.39}$$

where $\mathcal{X}$ is open and $\mathbf{f} : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ is locally Lipschitz. A control policy $\boldsymbol{\pi} : \mathcal{X} \to \mathcal{U}$ for the system can be obtained by solving an infinite-horizon discounted optimal control problem:

$$\begin{aligned} J^*_\gamma(\mathbf{x}_0) = \min_{\boldsymbol{\pi}} \; J^{\boldsymbol{\pi}}_\gamma(\mathbf{x}_0) &:= \sum_{k=0}^{\infty} \gamma^k \, \ell(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)), \\ \text{s.t.} \;\; \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)), &\quad \mathbf{x}_k \in \mathcal{X}, \quad \boldsymbol{\pi}(\mathbf{x}_k) \in \mathcal{U}, \end{aligned} \tag{5.40}$$

where $\gamma \in (0,1)$ is a discount factor and $\ell(\mathbf{x}, \mathbf{u})$ is a stage cost (or reward in the case of maximization), specifying the performance criterion.

Optimal control and reinforcement learning methods usually solve a problem like (5.40) to obtain a policy $\boldsymbol{\pi}$ and associated value function $J^{\boldsymbol{\pi}}_\gamma$. We consider a deterministic problem in our theoretical development for simplicity. Note that, while reinforcement learning methods work with stochastic control policies during training, only the mode of the final trained policy is used at test time, allowing us to treat it as deterministic.

Given a policy $\boldsymbol{\pi}$, we are interested in certifying whether it stabilizes the closed-loop system:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)). \tag{5.41}$$

We assume $\boldsymbol{\pi}(\mathbf{0}_n) = \mathbf{0}_m$, $\mathbf{f}(\mathbf{0}_n, \mathbf{0}_m) = \mathbf{0}_n$, $\ell(\mathbf{0}_n, \mathbf{0}_m) = 0$, and $\mathbf{0}_n \in \mathcal{X}$, so that the origin is an equilibrium point of the system. Stability can be certified by identifying a Lyapunov function.

Definition 5.2.1 (Lyapunov Function). Consider the closed-loop system (5.41). A continuous function $V : \mathcal{X} \to \mathbb{R}_{\geq 0}$ is a *Lyapunov function* if it satisfies:

$$V(\mathbf{0}_n) = 0, \quad V(\mathbf{x}) > 0 \quad \forall \mathbf{x} \in \mathcal{X} \setminus \{\mathbf{0}_n\}, \tag{5.42a}$$

$$V\big(\mathbf{f}(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))\big) - V(\mathbf{x}) < 0 \quad \forall \mathbf{x} \in \mathcal{X} \setminus \{\mathbf{0}_n\}. \tag{5.42b}$$

Lyapunov functions certify the asymptotic stability of the system, as stated in the following result.

Theorem 5.2.2 (Asymptotic Stability via a Lyapunov Function [78, Theorem 3.3]**).** *If there* *exists a Lyapunov function V as in Definition 5.2.1, then the origin $\mathbf{x} = \mathbf{0}_n$ is an asymptotically stable equilibrium of the system* (5.41).

When $\boldsymbol{\pi}$ arises from (5.40), the corresponding value function $J_\gamma^{\boldsymbol{\pi}}$ is also available. We are interested in whether $J_\gamma^{\boldsymbol{\pi}}$ can itself certify the stability of (5.41) or assist in constructing a certificate.

5.2.1 Lyapunov Stability Analysis for Linear Quadratic Problems

To understand the relationship between a discounted value function $J_\gamma^{\boldsymbol{\pi}}$ obtained from (5.40) and a Lyapunov function V certifying stability of the closed-loop system (5.41), we first study a simple setting with a linear system and quadratic stage cost.

Consider the discrete-time linear system:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \tag{5.43}$$

where $\mathbf{x}_k \in \mathbb{R}^n$, $\mathbf{u}_k \in \mathbb{R}^m$, and $\mathbf{A}$, $\mathbf{B}$ are known constant matrices.

Choosing the stage cost in (5.40) as $\ell(\mathbf{x}, \mathbf{u}) = \mathbf{x}^\top \mathbf{Q}\mathbf{x} + \mathbf{u}^\top \mathbf{R}\mathbf{u}$ with $\mathbf{Q} = \mathbf{C}^\top \mathbf{C} \succeq 0$ and $\mathbf{R} \succ 0$ leads to the discounted LQR problem [10]. The optimal value function $J_\gamma^*(\mathbf{x}) = \mathbf{x}^\top \mathbf{P}_\gamma \mathbf{x}$ is quadratic, where $\mathbf{P}_\gamma$ solves the discounted Algebraic Riccati Equation (ARE):

$$\mathbf{P}_\gamma = \mathbf{A}^\top \big(\mathbf{P}_\gamma - \mathbf{P}_\gamma \mathbf{B}\big(\gamma \mathbf{B}^\top \mathbf{P}_\gamma \mathbf{B} + \mathbf{R}\big)^{-1} \mathbf{B}^\top \mathbf{P}_\gamma\big)\mathbf{A} + \mathbf{Q}. \tag{5.44}$$

The corresponding optimal feedback gain is $\mathbf{K}_\gamma^\star = -(\gamma \mathbf{B}^\top \mathbf{P}_\gamma \mathbf{B} + \mathbf{R})^{-1} \mathbf{B}^\top \mathbf{P}_\gamma \mathbf{A}$, which yields the policy $\boldsymbol{\pi}_\gamma^*(\mathbf{x}_k) = \mathbf{K}_\gamma^\star \mathbf{x}_k$.

Under this policy, the closed-loop system evolves as

$$\mathbf{x}_{k+1} = \mathbf{F}_\gamma^* \mathbf{x}_k, \quad \text{where} \quad \mathbf{F}_\gamma^* = \mathbf{A} + \mathbf{B}\mathbf{K}_\gamma^\star. \tag{5.45}$$

Although J_γ^* satisfies the discounted Bellman equation, it is well understood that it does not guarantee Lyapunov decrease due to $\gamma \in (0, 1)$ [38]. Moreover, verifying stability by directly analyzing the eigenvalues of $\mathbf{F}_\gamma^*$ and solving the discounted ARE (5.44) is nontrivial due to its nonlinear dependence on γ. This motivates the idea [119] of constructing valid Lyapunov functions by augmenting J_γ^* with a residual term. To formalize this, we first state standard assumptions that ensure the existence of a stabilizing policy and a well-defined value function.

Assumption 5.2.3. *The pair* $(\mathbf{A}, \mathbf{B})$ *is stabilizable and the pair* $(\mathbf{A}, \mathbf{C})$ *is detectable.*

The following result shows the existence of a Lyapunov certificate derived from the value function.

Theorem 5.2.4 (Lyapunov Stability via LMIs for Discounted LQR [119]**).** *Suppose Assumption 5.2.3 holds. Consider an optimal policy* $\boldsymbol{\pi}_\gamma^*$ *and value function* J_γ^* *obtained from the LQR problem with discount* γ*. Let* $\mathbf{P}$ *denote the solution to the undiscounted ARE in* (5.44) *(with* $\gamma = 1$*). If there exist symmetric positive definite matrices* $\mathbf{S}_0, \mathbf{S}_1 \succ 0$ *and scalars* $\varpi, \alpha > 0$ *such that:*

$$\begin{bmatrix} \mathbf{A}^\top \mathbf{S}_0 \mathbf{A} - \mathbf{S}_0 + \mathbf{S}_1 - \varpi \mathbf{Q} & \mathbf{A}^\top \mathbf{S}_0 \mathbf{B} \\ \mathbf{B}^\top \mathbf{S}_0 \mathbf{A} & \mathbf{B}^\top \mathbf{S}_0 \mathbf{B} - \varpi \mathbf{R} \end{bmatrix} \preceq 0, \quad \alpha \mathbf{P} \preceq \mathbf{S}_1, \tag{5.46}$$

then the function $V(\mathbf{x}) := J_\gamma^*(\mathbf{x}) + \frac{1}{\varpi}\mathbf{x}^\top \mathbf{S}_0 \mathbf{x}$ *is a Lyapunov function for the closed-loop system* (5.45)*, and certifies global exponential stability for any discount factor satisfying* $\gamma > \frac{\varpi}{\varpi + \alpha}$.

Theorem 5.2.4 shows that while the discounted value function J_γ^* may not satisfy the Lyapunov condition on its own, it can be modified into a valid certificate when γ is sufficiently large. The required residual term and a computable lower bound on γ are obtained by solving a set of LMIs. However, this bound is often conservative compared to the true stability threshold γ^* obtained by analyzing the closed-loop dynamics via the discounted ARE, as illustrated in the following example.

Example 5.2.5 ([119, Example 1])**.** *Consider the scalar system $x_{k+1} = 2x_k + u_k$ with stage cost $\ell(x,u) = x^2 + u^2$. The optimal policy is $u_k = K^\star_\gamma x_k$, where*

$$K^\star_\gamma = -2\big(1 + 2\big(5\gamma - 1 + \sqrt{(5\gamma-1)^2 + 4\gamma}\big)^{-1}\big)^{-1}.$$

The closed-loop multiplier is $F^*_\gamma = 2 + K^\star_\gamma$, and the origin is globally exponentially stable if and only if $|F^*_\gamma| < 1$, which is equivalent to $\gamma > \gamma^* = 1/3$. However, applying the LMIs from Theorem 5.2.4 yields a feasible solution for $\gamma > 0.8090$, which is significantly more conservative.

Example 5.2.5 highlights the limitation of classical Lyapunov analysis and motivates the development of an alternative formulation with less conservative conditions for certifying stability.

5.2.2 Generalized Lyapunov Stability and Applications to Linear Systems

Building on the observation from Example 5.2.5, we introduce a generalized notion of Lyapunov stability [48, Definition 2.2] that relaxes the classical pointwise decrease condition.

Definition 5.2.6 (Generalized Lyapunov Function). Consider the closed-loop system in (5.41). A continuous function $V : \mathcal{X} \to \mathbb{R}_{\geq 0}$ is a *generalized Lyapunov function* if it satisfies (5.42a) and there exist an integer $M \in \mathbb{N}_{>0}$ and state-dependent non-negative weights $\sigma_1(\mathbf{x}), \ldots, \sigma_M(\mathbf{x})$ such that

$$\frac{1}{M}\sum_{i=1}^{M} \sigma_i(\mathbf{x}) \geq 1, \tag{5.47}$$

and, for any $\mathbf{x} \in \mathcal{X} \setminus \{\mathbf{0}_n\}$, the following generalized decrease condition holds:

$$\frac{1}{M}\sum_{i=1}^{M} \sigma_i(\mathbf{x}) V(\mathbf{x}_i) - V(\mathbf{x}) < 0, \tag{5.48}$$

where $\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \boldsymbol{\pi}(\mathbf{x}_i))$ with $\mathbf{x}_0 = \mathbf{x}$.

Condition (5.48) generalizes the decrease requirement in (5.42b) by allowing temporary increases in V over individual steps, provided that its weighted average decreases over a finite horizon of length M. We state the stability guarantee provided by a generalized Lyapunov function in the next theorem.

Theorem 5.2.7 (Asymptotic Stability via a Generalized Lyapunov Function). *Consider the closed-loop system in* (5.41)*, and assume the policy* $\boldsymbol{\pi}$ *is Lipschitz on* $\mathcal{X}$*. If there exists a generalized Lyapunov function* $V : \mathcal{X} \to \mathbb{R}_{\geq 0}$ *as in Definition 5.2.6, then* $\mathbf{x} = \mathbf{0}_n$ *is an asymptotically stable equilibrium.*

Returning to the LQR setting in Section 5.2.1, we consider certifying the stability of (5.45) via a generalized Lyapunov function. Building on Theorem 5.2.4, we augment J_γ^* with a quadratic residual and require the composite function to satisfy (5.48), leading to a new set of LMIs for stability certification.

Theorem 5.2.8 (Generalized Lyapunov Stability for Discounted LQR via LMIs). *Suppose Assumption 5.2.3 holds. Consider an optimal policy* $\boldsymbol{\pi}_\gamma^*$ *and value function* J_γ^* *obtained from the LQR problem with discount* γ*. Let* $\mathbf{P}$ *denote the solution to the undiscounted ARE in* (5.44) *(with* $\gamma = 1$*). If there exist symmetric positive definite matrices* $\mathbf{S}_0, \mathbf{S}_1, \ldots, \mathbf{S}_M$*, scalars* $\varpi, \alpha_1, \ldots, \alpha_M > 0$*, and weights* $\sigma_1, \ldots, \sigma_M \geq 0$ *satisfying* $\sum_{i=1}^M \sigma_i \geq M$*, such that:*

$$\begin{bmatrix} \frac{\sigma_1}{M}\mathbf{A}^\top\mathbf{S}_0\mathbf{A} - \mathbf{S}_0 + \mathbf{S}_1 - \varpi\mathbf{Q} & \frac{\sigma_1}{M}\mathbf{A}^\top\mathbf{S}_0\mathbf{B} \\ \frac{\sigma_1}{M}\mathbf{B}^\top\mathbf{S}_0\mathbf{A} & \frac{\sigma_1}{M}\mathbf{B}^\top\mathbf{S}_0\mathbf{B} - \varpi\mathbf{R} \end{bmatrix} \preceq 0, \tag{5.49a}$$

$$\begin{bmatrix} \frac{\sigma_{i+1}}{M}\mathbf{A}^\top\mathbf{S}_0\mathbf{A} - \mathbf{S}_{i+1} - \varpi\mathbf{Q} & \frac{\sigma_{i+1}}{M}\mathbf{A}^\top\mathbf{S}_0\mathbf{B} \\ \frac{\sigma_{i+1}}{M}\mathbf{B}^\top\mathbf{S}_0\mathbf{A} & \frac{\sigma_{i+1}}{M}\mathbf{B}^\top\mathbf{S}_0\mathbf{B} - \varpi\mathbf{R} \end{bmatrix} \preceq 0, \quad \forall i = 1, \ldots, M-1, \tag{5.49b}$$

$$\alpha_i\mathbf{P} \preceq \mathbf{S}_i, \quad \forall i = 1, \ldots, M, \tag{5.49c}$$

then the function

$$V(\mathbf{x}) := J_\gamma^*(\mathbf{x}) + \frac{1}{\varpi}\mathbf{x}^\top \mathbf{S}_0 \mathbf{x} \tag{5.50}$$

is a generalized Lyapunov function in the sense of Definition 5.2.6 (with constant weights), and certifies that the origin is globally exponentially stable for the closed-loop system (5.45)*, provided that:*

$$\gamma > \max\left(\frac{\sigma_M \varpi}{M\alpha_M}, \ldots, \frac{\sigma_2 \varpi}{M\alpha_2}, \frac{\sigma_1 \varpi}{M(\varpi + \alpha_1)}\right). \tag{5.51}$$

Remark 5.2.9 (Feasibility of Multi-Step LMIs). *Note that when $\sigma_1 = M$ and $\sigma_i = 0$ for all $2 \leq i \leq M$, Theorem 5.2.8 recovers the classical Lyapunov result in Theorem 5.2.4. Therefore, if the original one-step LMIs* (5.46) *are feasible, then feasible solutions for* (5.49) *also exist.*

Remark 5.2.10 (Choice of σ_i Weights). *While the multi-step formulation enables optimizing the weights $\sigma_1, \ldots, \sigma_M$ to minimize the certified lower bound on γ in* (5.51)*, finding globally optimal weights is challenging due to its non-convexity nature. In practice, heuristics such as grid search for small M and random sampling with local refinements for larger M are sufficient to achieve noticeable improvements over the one-step baseline.*

Example 5.2.11 (Example 5.2.5 Revisited). *We illustrate the benefits of the generalized multi-step LMIs from Theorem 5.2.8 in Example 5.2.5. We first focus on the case $M = 2$. Figure 5.5 shows how the certified bound on γ varies with the choice of weights σ_1 and σ_2, constrained by $\sigma_1 + \sigma_2 = 2$. Since the undiscounted value function ($\gamma = 1$) satisfies the Lyapunov condition, this bound is capped at $\gamma = 1$. The classical Lyapunov setting $(\sigma_1, \sigma_2) = (2, 0)$ yields a conservative bound of $\gamma > 0.8090$, whereas optimizing over σ_1 and σ_2 improves the bound to $\gamma > 0.6229$ at $(\sigma_1, \sigma_2) = (1.54, 0.46)$. Another important observation is that increasing M significantly reduces the certified lower bound on γ, progressively approaching the true stability threshold $\gamma^\star = 1/3$, as shown in Figure 5.6.*

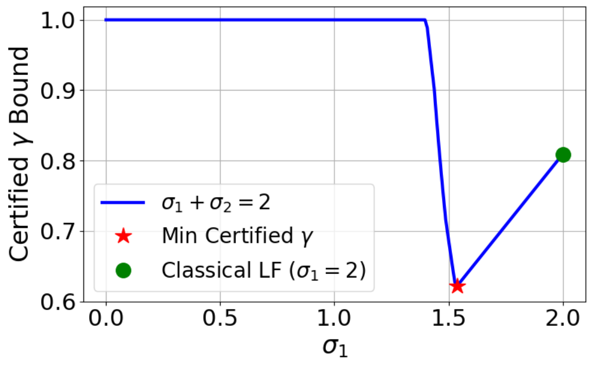


Figure 5.5. Certified γ bound for $M = 2$.

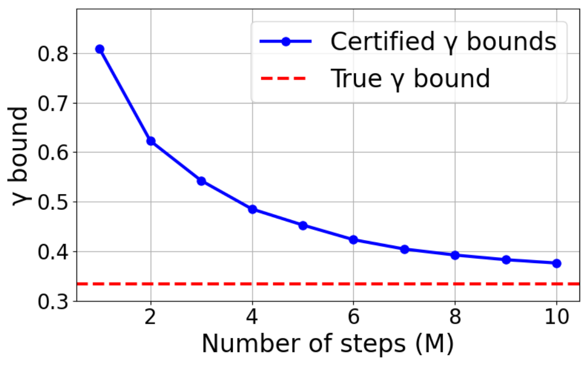


Figure 5.6. Certified γ bound versus M.

5.2.3 Nonlinear Systems and Stability Certification of RL Policies

Inspired by the treatment of linear systems described in Section 5.2.2, we formulate an approach to certify stability of policies for nonlinear systems obtained by RL using a generalized Lyapunov function. The key observation from Theorem 5.2.8 is that augmenting the optimal value function with a residual term can result in a valid generalized Lyapunov function, as in (5.50). Here, we use the same idea to form generalized Lyapunov functions for nonlinear systems with unknown dynamics, which is a problem considered by RL. The value function and policy obtained by an RL algorithm are typically parameterized by neural networks. Let $\boldsymbol{\pi}_{\text{RL}}$ be a pre-trained RL policy (e.g., obtained by [53, 61, 131]), and let $J_\gamma^{\boldsymbol{\pi}_{\text{RL}}}$ denote its corresponding learned value function. We consider a generalized Lyapunov candidate as:

$$V(\mathbf{x};\boldsymbol{\theta}_1) = J_\gamma^{\boldsymbol{\pi}_{\text{RL}}}(\mathbf{x}) + \varphi(\mathbf{x};\boldsymbol{\theta}_1), \tag{5.52}$$

where $\varphi(\mathbf{x};\boldsymbol{\theta}_1)$ is a neural residual correction. To allow more flexibility in the generalized Lyapunov condition (5.48), we introduce a step-weighting network $\sigma(\mathbf{x};\boldsymbol{\theta}_2) \in \mathbb{R}^M_{\geq 0}$ that outputs non-negative weights over a rollout horizon of length M. The weights are required to satisfy $\sum_{i=1}^M \sigma_i(\mathbf{x};\boldsymbol{\theta}_2) \geq M$. These weights are then used to construct the generalized Lyapunov decrease condition:

$$F(\mathbf{x}_k) := \frac{1}{M}\sum_{i=1}^{M}\sigma_i(\mathbf{x}_k;\boldsymbol{\theta}_2)\,V(\mathbf{x}_{k+i};\boldsymbol{\theta}_1) - (1-\bar{\alpha})\,V(\mathbf{x}_k;\boldsymbol{\theta}_1), \tag{5.53}$$

where $\bar{\alpha} \in (0,1)$ is a user-specified decay parameter. We train the networks $\varphi(\cdot;\boldsymbol{\theta}_1)$ and $\sigma(\cdot;\boldsymbol{\theta}_2)$ jointly by minimizing a loss that penalizes violations of the generalized Lyapunov condition:

$$\mathcal{L}(\boldsymbol{\theta}_1,\boldsymbol{\theta}_2) := \frac{1}{N}\sum_{i=1}^{N}\mathrm{ReLU}\big(F(\mathbf{x}_k^{(i)})\big), \tag{5.54}$$

where $\{\mathbf{x}_k^{(i)}\}_{i=1}^{N}$ are sampled initial states.

Remark 5.2.12 (Network Architecture). *In practice, the learned value function $J_\gamma^{\boldsymbol{\pi}_{RL}}$ may not be optimal, hence, may not satisfy $J_\gamma^{\boldsymbol{\pi}_{RL}}(\mathbf{0}_n) = 0$. To ensure that the generalized Lyapunov candidate satisfies $V(\mathbf{0}_n;\boldsymbol{\theta}_1) = 0$ and is positive definite, we modify* (5.52) *as:*

$$V(\mathbf{x};\boldsymbol{\theta}_1) := \left|J_\gamma^{\boldsymbol{\pi}_{RL}}(\mathbf{x}) - J_\gamma^{\boldsymbol{\pi}_{RL}}(\mathbf{0}_n)\right| + |\varphi(\mathbf{x};\boldsymbol{\theta}_1) - \varphi(\mathbf{0}_n;\boldsymbol{\theta}_1)| + \beta\|\mathbf{x}\|^2, \tag{5.55}$$

where $\beta > 0$ is a small constant (the term $\beta\|\mathbf{x}\|^2$ enforces strict positive definiteness for $\mathbf{x} \neq \mathbf{0}_n$ and improves numerical stability near the origin.) The step-weighting network $\sigma(\mathbf{x};\boldsymbol{\theta}_2) \in \mathbb{R}^M_{\geq 0}$ ends with a softmax layer scaled by M, ensuring the output weights satisfy $\frac{1}{M}\sum_{i=1}^{M}\sigma_i(\mathbf{x};\boldsymbol{\theta}_2) = 1$.

Remark 5.2.13 (Equilibrium Behavior). *In practice, RL policies often converge to a neighborhood near the origin rather than the origin itself. Thus, we train and verify the generalized Lyapunov condition over the set $\mathcal{X} \setminus \mathcal{B}(\mathbf{0}_n;\delta)$, where $\mathcal{B}(\mathbf{0}_n;\delta)$ denotes a ball of radius δ around the origin.*

Training and Evaluation Setup. We evaluate our method on two standard RL control benchmarks from Gymnasium [138] and the DeepMind Control Suite [137]. RL policies $\boldsymbol{\pi}_{\mathrm{RL}}$ and their corresponding value functions $J_\gamma^{\boldsymbol{\pi}_{\mathrm{RL}}}$ are trained using implementations from [61, 122].

We collect a total of N rollout trajectories by simulating the closed-loop system under $\boldsymbol{\pi}_{\mathrm{RL}}$, with each trajectory $\tau^{(i)} = \{\mathbf{x}_k^{(i)}\}_{k=0}^{M}$ starting from a randomly sampled initial state $\mathbf{x}_0^{(i)}$.

Inverted Pendulum Swingup. We consider the inverted pendulum environment from [138] with parameters $m = 1$, $l = 1$, $g = 10$, and control limits $|u| \leq \frac{mgl}{5} = 2$. The state space is $[-\pi, \pi) \times [-8, 8]$. Due to tight torque limits, the pendulum must swing back and forth to build momentum before reaching the upright position. This makes it challenging to synthesize a policy with a Lyapunov certificate valid over the entire state space. Prior work [97, 151] has shown that regions where stability can be verified are typically restricted to small neighborhoods near the upright position, failing to cover the full swing-up trajectories. In contrast, modern RL policies can discover effective swing-up behaviors but lack stability guarantees. Our method bridges this gap: we take a trained SAC policy [53] and apply our generalized Lyapunov function training with $M = 15$. Figure 5.7 shows the generalized Lyapunov function values along several trajectories from different initial states. As expected, the function exhibits non-monotonic behavior but shows an overall decline over the planning horizon. Figure 5.8 visualizes the Lyapunov function across the state space. Figure 5.9 plots the residual $F(\mathbf{x}_k)$ defined in (5.53), verifying that the generalized decrease condition is satisfied throughout the domain.

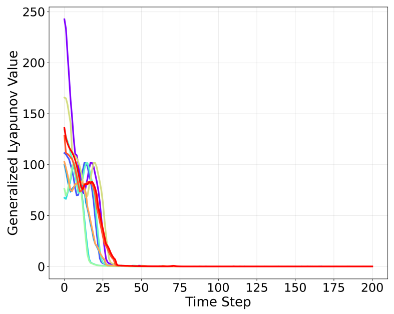


Figure 5.7. Generalized Lyapunov function values along trajectories.

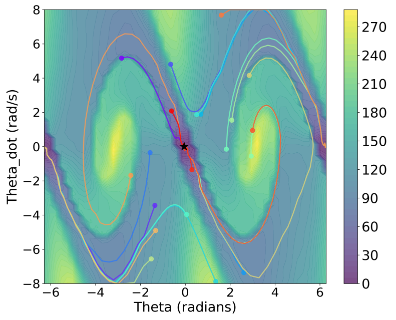


Figure 5.8. Generalized Lyapunov function value over the state space.

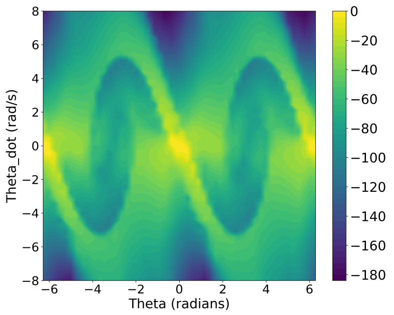


Figure 5.9. Generalized Lyapunov decrease condition.

Cartpole Swingup. We consider the cartpole swing-up task from [137]. The system state is represented as $[x, \cos(\theta), \sin(\theta), \dot{x}, \dot{\theta}]$, where x is the cart position, θ is the pole angle, and the remaining terms are their velocities. The goal is to swing the pole upright $\theta = 0$ and stabilize the position at $x = 0$. We use a TD-MPC policy [61] and apply our certificate training (5.54) with $M = 20$. Figure 5.10 visualize the generalized Lyapunov condition across three representative 2D slices of the state space, with the remaining two states fixed at zero. The generalized decrease condition is satisfied throughout these slices, suggesting asymptotic stability of the RL policy.

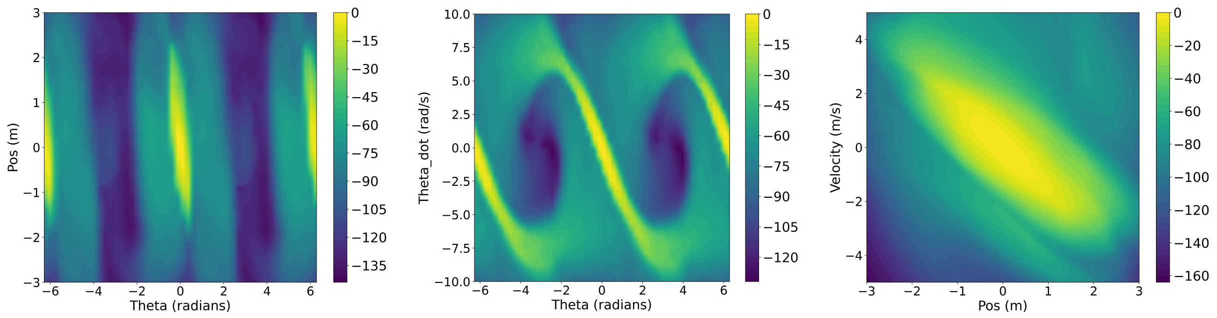


Figure 5.10. Generalized Lyapunov decrease condition for the cartpole swing-up using a TD-MPC policy.

In addition to qualitative visualizations, we quantitatively evaluate our learned certificates by sampling N_{test} states from the full state space and checking whether (5.53) is satisfied. For each environment, we train certificates for multiple RL policies (the PPO policy fails to stabilize the cartpole from some initial states). As shown in Table 5.2, the condition holds for all test states in both environments.

Table 5.2. Percentage of sampled test states satisfying $F(\mathbf{x}_k) \leq 0$ under different RL policies.

Environment	**RL Methods**	M	N_{test}	**% Satisfying** $F(\mathbf{x}_k) \leq 0$
Inverted Pendulum	PPO, SAC, TD-MPC	15	10,000	100%
Cartpole	SAC, TD-MPC	20	1,000,000	100%

5.2.4 Joint Synthesis of Stable Neural Policies and Generalized Certificates

So far, we addressed the certification of stability for fixed pre-trained control policies. In this section, we consider *joint synthesis* of neural controllers and Lyapunov certificates and employ formal verification inspired by [24, 33, 145, 151]. We demonstrate that our generalized Lyapunov approach integrates naturally into this setting by replacing the standard one-step decrease condition with the proposed multi-step, weighted formulation in (5.48).

Definition 5.2.14 (Region of Attraction). The *region of attraction (ROA)* $\mathcal{R} \subseteq \mathbb{R}^n$ of the (locally asymptotically stable) origin for the discrete-time system (5.41) is the set of all points from which the system trajectory converges to the origin, i.e., $\mathbf{x}_0 \in \mathcal{R}$ implies $\lim_{k\to\infty} \mathbf{x}_k = \mathbf{0}_n$.

In general, precise characterizations of $\mathcal{R}$ are challenging to obtain and one instead looks for suitable approximations. Here, we seek to jointly learn a policy $\boldsymbol{\pi}(\mathrm{x}; \boldsymbol{\phi})$ and a certificate function $V(\mathrm{x}; \boldsymbol{\theta}_1)$ such that the closed-loop system $\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \boldsymbol{\pi}_\phi(\mathbf{x}_k))$ is provably asymptotically stable and obtain at the same time an inner approximation $\mathcal{S}$ of the ROA $\mathcal{R}$. [151] formalized it as a constrained optimization that maximizes the volume of a Lyapunov sublevel set $\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^n : V(\mathbf{x}) \leq \rho\}$ under Lyapunov constraints

$$\max_{\boldsymbol{\theta}_1, \boldsymbol{\phi}} \quad \mathrm{Vol}(\mathcal{S}) \tag{5.56a}$$

$$\text{s.t.} \quad V(\mathbf{0}_n; \boldsymbol{\theta}_1) = 0, \quad V(\mathbf{x}; \boldsymbol{\theta}_1) > 0 \quad \forall \mathbf{x} \in \mathcal{S} \setminus \{\mathbf{0}_n\}, \tag{5.56b}$$

$$V(\mathbf{x}_{k+1}; \boldsymbol{\theta}_1) - (1 - \bar{\alpha}) V(\mathbf{x}_k; \boldsymbol{\theta}_1) \leq 0 \quad \forall \mathbf{x}_k \in \mathcal{S}, \tag{5.56c}$$

where $\bar{\alpha} \in (0, 1)$ is a user-specified decay parameter. To utilize a generalized Lyapunov function, we replace (5.56c) with the generalized M-step condition for each $\mathbf{x}_k \in \mathcal{S}$, with $\sum_{i=1}^{M} \sigma_i(\mathbf{x}_k) \geq M$:

$$F(\mathbf{x}_k) := \frac{1}{M} \sum_{i=1}^{M} \sigma_i(\mathbf{x}_k) V(\mathbf{x}_{k+i}; \boldsymbol{\theta}_1) - (1 - \bar{\alpha}) V(\mathbf{x}_k; \boldsymbol{\theta}_1) \leq 0, \;\; \sigma_i(\mathbf{x}_k) \geq \underline{\sigma} > 0, \tag{5.57}$$

where $\underline{\sigma} \in \mathbb{R}_{>0}$ is a uniform lower bound on the weights. The condition (5.57) enables learning certificates that tolerate non-monotonic behavior along a trajectory, but it no longer guarantees that $\mathcal{S}$ is *forward invariant*. However, as Theorem 5.2.15 shows, $\mathcal{S}$ still defines a valid inner approximation of the ROA and guarantees asymptotic stability of the origin.

Theorem 5.2.15 (**Asymptotic Stability Relative to $\mathcal{S}$**). *If the optimization problem* (5.56) *is solved with the generalized condition* (5.57) *instead of* (5.56c)*, then $\mathcal{S} \subset \mathcal{R}$.*

Training Formulation. Following [151], we reformulate the problem (5.56) into a learning objective by sampling states $\mathbf{x}_k \in \mathcal{X}$ and penalizing violations of the Lyapunov conditions using soft constraints. For each $\mathbf{x}_k \in \mathcal{D}$, we simulate M forward steps under the closed-loop dynamics and compute the generalized Lyapunov residual $F(\mathbf{x}_k)$ as defined in (5.57).

To enforce stability and domain constraints, we define the stability loss:

$$\mathcal{L}_{\text{stab}}(\mathbf{x}_k) := \text{ReLU}\left(\min\left\{\text{ReLU}(F(\mathbf{x}_k)) + c_0\mathcal{H}(\mathbf{x}_k), \rho - V(\mathbf{x}_k, \boldsymbol{\theta}_1)\right\}\right), \tag{5.58}$$

where $\mathcal{H}(\mathbf{x}_k) := \sum_{i=1}^{M} \left\|\text{ReLU}(\mathbf{x}_{k+i} - \mathbf{x}_{\text{up}}) + \text{ReLU}(\mathbf{x}_{\text{lo}} - \mathbf{x}_{k+i})\right\|_1$ penalizes violations of the bounded domain $\mathcal{X} = \{\mathbf{x} \mid \mathbf{x}_{\text{lo}} \leq \mathbf{x} \leq \mathbf{x}_{\text{up}}\}$. The inner $\min$ ensures that the generalized Lyapunov decrease condition is only enforced inside the certified region $\mathcal{S}$.

To expand the certifiable region, [151] proposed a surrogate region loss $\mathcal{L}_{\text{region}} := \sum_{j=1}^{N} \text{ReLU}\left(V(\mathbf{x}_j; \boldsymbol{\theta}_1)/\rho - 1\right)$, where the candidate states $\mathbf{x}_j$ are obtained via random boundary sampling or projected gradient descent (PGD) to minimize $V(\cdot; \boldsymbol{\theta}_1)$. PGD is also used for falsification by maximizing stability violations, generating additional training states for $\mathcal{D}$.

The final training objective is $\mathcal{L}(\boldsymbol{\theta}_1, \boldsymbol{\phi}) := \sum_{\mathbf{x}_k \in \mathcal{D}} \mathcal{L}_{\text{stab}}(\mathbf{x}_k) + c_1\mathcal{L}_{\text{region}} + c_2\|\boldsymbol{\theta}_1, \boldsymbol{\phi}\|_1$, where $\mathcal{D}$ contains states sampled randomly and generated by falsification.

Remark 5.2.16. *Although the theoretical framework (Theorem 5.2.7) supports trainable weights $\sigma_i(\mathbf{x}_k)$, including neural network parameterizations (see Section 5.2.3), certifying stability under jointly learned controllers, Lyapunov functions, and weights remains computationally challenging with existing tools [51, 142]. In practice, we fix σ_i during training. For small values of M, we select the best-performing weight configuration via a simple grid search.*

Verification Formulation. After training, we verify that the generalized decrease condition holds within $\mathcal{X}$ using the α-β-CROWN verifier [142]. We formally certify Lyapunov stability of the origin if the following holds for all $\mathbf{x}_k \in \mathcal{X}$:

$$\big(-F(\mathbf{x}_k) \geq 0 \ \wedge \ \bigwedge_{i=1}^{M} \mathbf{x}_{k+i} \in \mathcal{X}\big) \ \vee \ (V(\mathbf{x}_k) \geq \rho)\,. \tag{5.59}$$

As Theorem 5.2.15 implies, if (5.59) holds for all $\mathbf{x}_k \in \mathcal{X}$, then $\mathcal{S}$ is an inner approximation of $\mathcal{R}$.

Evaluation. We evaluate our generalized Lyapunov synthesis approach on three systems presented in [151]: inverted pendulum, path tracking, and 2D quadrotor (with a 6D state space).

Figures 5.11 and 5.12 show the certified stability regions $\mathcal{S}$ for different horizon lengths M. We use fixed step weights $(\sigma_1, \sigma_2, \dots)$ selected via grid search: for the inverted pendulum, $(0.4, 1.6)$ for M=2 and $(0.3, 1.5, 1.2)$ for M=3; for path tracking, $(0.4, 1.6)$ and $(1.2, 1.2, 0.6)$, respectively. As shown in the figures, multi-step training and verification yields consistently larger certifiable ROAs.

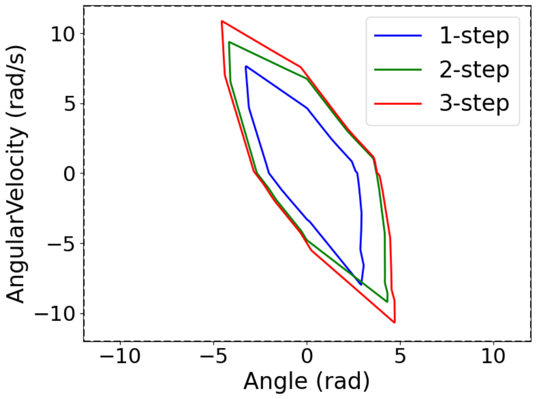


Figure 5.11. Certified ROAs for inverted pendulum.

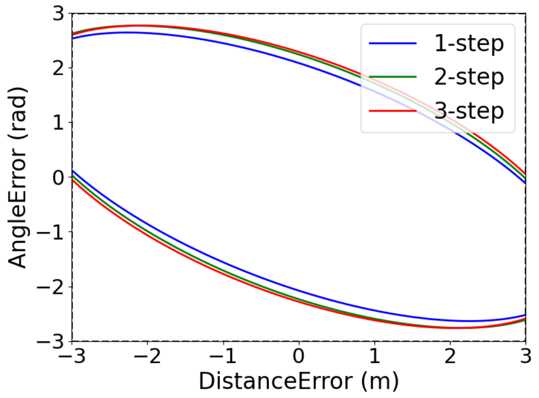


Figure 5.12. Certified ROAs for path-tracking.

Table 5.3 presents the quantitative results, including the volume of $\mathcal{S}$ and verification time. The volume is estimated via Monte Carlo integration: we sample $n = 10^6$ points in $\mathcal{X}$ and compute the fraction satisfying $V(\mathbf{x}) \leq \rho$, averaged over 10 trials and scaled by the total domain volume. While our generalized stability formulation leads to consistently larger certified regions, it incurs longer formal verification time due to the need to bound multiple intermediate states and their weighted combinations, leading to more complex bound computations and more branching decisions.

Table 5.3. Certified region volume (left) and verification time (right) under different M-step Lyapunov training.

System	$M = 1$	$M = 2$	$M = 3$
Inverted Pendulum	42.87	76.68	89.59
Path Tracking	21.81	23.42	23.93
2D quadrotor	103.26	109.45	113.53

(a) Certified ROA volume ($\mathcal{S}$).

System	$M = 1$	$M = 2$	$M = 3$
Inverted Pendulum	11.54	22.42	39.87
Path Tracking	8.53	19.61	36.26
2D quadrotor	2209.75	3824.19	5657.86

(b) Verification time (seconds).

This chapter has presented a comprehensive framework for stability certification under uncertainty, bridging the gap between classical control theory's theoretical rigor and modern learning-based control's practical effectiveness. We have addressed the fundamental challenge of providing stability guarantees for uncertain systems with neural network controllers. Our distributionally robust formulations successfully extend classical Lyapunov analysis to handle model uncertainties without requiring exact knowledge of uncertainty distributions, while our generalized Lyapunov framework enables certification of high-performance reinforcement learning policies that would otherwise resist traditional stability certificate construction.

By replacing strict pointwise decrease requirements with flexible multi-step weighted criteria and incorporating distributional robustness directly into certificate construction, this work establishes a principled pathway for deploying neural network controllers in real-world robotic applications while maintaining the performance advantages that make learning-based control so compelling. The integration of sum-of-squares programming, neural network optimization, and formal verification techniques creates a versatile toolkit that can adapt to different system complexities and uncertainty structures, ultimately enabling the confident deployment of intelligent autonomous systems in uncertain real-world environments.

Chapter 5, in full, is a reprint of the material as it appears in Distributionally Robust Lyapunov Function Search Under Uncertainty, K. Long, Y. Yi, J. Cortés, and N. Atanasov, Learning for Dynamics and Control Conference, 2023; Distributionally Robust Policy and

Lyapunov-Certificate Learning, K. Long, J. Cortés, and N. Atanasov, IEEE Open Journal of Control Systems, 2024; and Generalized Lyapunov Stability for Certified Control and Reinforcement Learning, K. Long, J. Cortés, and N. Atanasov, under review, 2025. The dissertation author was the primary researcher and author of these works.

Chapter 6

Conclusions and Future Work

This thesis has investigated the problem of **certifiable robot autonomy under uncertainty**, with a focus on learning- and optimization-based methods that provide formal guarantees. The key contributions span both theoretical and algorithmic developments, including robust and probabilistic formulations of safe control, distributionally robust safety filters for robot control, and neural stability certificates with Lyapunov-stable policies. Through extensive simulations and experiments on ground mobile robots and 6-dimensional manipulators, this work demonstrates that robots can achieve safe, efficient, and theoretically grounded autonomy in complex and dynamic environments while handling various sources of uncertainty.

6.1 Summary of Contributions

This thesis merges control-theoretic rigor with learning-based representations to address the fundamental challenge of safe, stable, and efficient robotic control under uncertainty. In particular, I extend classical certificate functions (e.g., CBFs, CLFs) to robust, probabilistic, and distributionally robust formulations, and demonstrate their efficacy across various robotic systems in complex and dynamic environments. I also presented results

6.1.1 Safe Control under Uncertainty

Ensuring safe control under uncertainty remains a core challenge for autonomous robot systems. Control barrier functions (CBFs) provide a principled, **real-time** method for enforcing safety in control-affine systems by solving a quadratic program that adjusts nominal inputs to maintain forward invariance of a safe set. Due to their computational efficiency, CBFs have been successfully deployed across robotic domains. However, most early work assumes that the CBF, system dynamics, and robot states are precisely known, which is often invalid in real-world deployments with noisy onboard sensors. My research addresses these limitations by systematically incorporating sensor noise, dynamic inaccuracies, and state estimation errors into the CBF framework. A central challenge in safe control synthesis is obtaining accurate, efficient geometric representations of both the environment and the robot itself.

In [101], I tackled this by using neural networks to learn a signed distance function (SDF) of obstacles from onboard sensor data. Unlike conventional approaches that assume a priori known barriers, my method incrementally learns the SDF using range measurements. By explicitly taking the estimation error bounds of the learned SDF and its gradient into account, I formulated a robust control barrier constraint, which led to a novel **second-order cone programming** formulation for safe control synthesis. Building on this, I extended my research to consider uncertainty not only in environment perception but also in neural system dynamics approximations [98]. I developed both probabilistic and robust formulations of CBF constraints, allowing the control synthesis problem to handle uncertainty in both barrier function and system dynamics estimates.

Although other recent work has also considered uncertainty in CBF-based safety filters, they often address a single source (e.g., model or barrier error) in isolation. In contrast, real-world robots face **compounded uncertainties** from various sources (e.g., localization, sensor, geometry estimations), which interact in nonlinear ways and are hard to model explicitly. This motivates my use of distributionally robust optimization (DRO), which handles multiple uncertainties without requiring precise bounds or distributions. DRO circumvents the need for explicit uncertainty models by operating directly on sampled data, such as LiDAR hits or states from standard estimators. By enforcing a chance constraint over a Wasserstein ambiguity set, it avoids restrictive distributional assumptions. At the same time, simplistic robot shapes (e.g., circles or spheres) often over-constrain feasible motions, particularly for manipulators. Recent work instead uses neural SDFs and configuration-space distance functions to capture complex geometries more accurately, but these **neural representations** introduce uncertainties that are difficult to quantify, reinforcing the need for a distributionally robust approach.

In my work [99, 104, 105], I propose a novel **distributionally robust control barrier constraint** that systematically accounts for uncertainties in state estimation, system dynamics, sensor measurements, and neural shape representations. Critically, the resulting safe control synthesis problem can be reformulated as a quadratic program. I validate this approach on both ground mobile robots and 6-dimensional manipulators, demonstrating safe, efficient control under uncertainty in cluttered, dynamic environments.

6.1.2 Stability Certification for Neural Policies

The third major contribution of my research connects reinforcement learning (RL) with formal stability guarantees. While modern RL policies achieve impressive performance, they typically lack stability certificates, which limits their applicability in safety-critical domains. Most prior work assumes deterministic models, leaving open the problem of analyzing Lyapunov stability and synthesizing stable controllers in the presence of model uncertainty.

To address this challenge, I combine Lyapunov-based stability principles with distributionally robust optimization (DRO), enabling the synthesis of neural controllers and certificates that remain valid under uncertainty. In [103], I introduce the concept of a distributionally robust Lyapunov function (DR-LF) for closed-loop systems with parametric uncertainty. The DR-LF search is formulated both as a sum-of-squares (SOS) program and as a neural network optimization problem, providing a scalable framework for certifying stability in probability. Building on this work, [97] extends the approach to jointly learn neural stabilizing controllers and Lyapunov certificates for uncertain nonlinear systems. However, we find that joint training is challenging for underactuated systems with limited control bounds, whereas RL with carefully designed rewards can still produce high-performance policies. This insight motivates our subsequent work.

In [96], I study stability certification for closed-loop systems under policies obtained from optimal control and RL. We make two key observations: first, a policy's value function can be augmented with a residual term to form a valid stability certificate; second, the classical step-wise Lyapunov decrease condition can be relaxed to a multi-step weighted criterion. This leads to the concept of a generalized Lyapunov function, where the value function is augmented with a residual neural network and the generalized decrease condition is enforced by optimizing state-dependent weights. This relaxation makes it possible to certify the stability of modern RL policies and also supports joint training of controllers and certificates using a multi-step Lyapunov loss, resulting in substantially larger certified regions of attraction.

6.2 Limitations and Open Challenges

While the methods presented in this thesis advance the state of the art in stability certification and uncertainty-aware safety-critical control, several limitations remain that open opportunities for future research.

1. **Pointwise Probabilistic Safety Guarantees.** The probabilistic safety guarantees provided in this work are primarily pointwise, ensuring that individual states satisfy safety constraints with a specified probability. Extending these notions to *trajectory-wise* probabilistic safety, where the entire execution satisfies the safety requirement with high probability, remains an open challenge.

2. **Feasibility and Myopic Behavior in CBF Approaches.** Control Barrier Function (CBF) formulations can suffer from feasibility issues, especially in scenarios with multiple safety constraints or under limited control bounds. Besides, the inherently myopic nature of many CBF-based controllers often leads to conservative actions that maintain safety but sacrifice long-term performance. Designing controllers that balance safety with performance over extended horizons remains an unresolved problem.

3. **Generality Across Systems and High-Dimensional Applications.** Constructing valid CBFs or stability certificates for arbitrary nonlinear and high-dimensional systems remains a difficult task. The methods developed here have been demonstrated on representative systems, but scaling them to robots with many degrees of freedom or to domains with complex, hybrid dynamics is still challenging.

4. **Safety Notions Beyond Collision Avoidance.** Many real-world robotic tasks require safety concepts that are difficult to formalize, such as ergonomic constraints in human–robot collaboration, thermal limits, energy budgets, or mission-specific operational envelopes. Extending safety-aware control frameworks to handle such abstract or task-dependent notions remains largely unexplored.

5. **Semantic and Language-Conditioned Safety.** In principle, safe sets or CBFs should be definable not only from geometric and physical constraints, but also from semantic information or high-level task descriptions. Extending safety-aware control to incorporate semantic cues, natural language instructions, and context-dependent safety specifications is an open and exciting research direction.

6. **Integration with High-Level Task and Motion Planning.** Most of the presented approaches operate at the control or local planning level, without tight integration with symbolic task planners or high-level decision-making frameworks. This limits their applicability in long-horizon, multi-stage tasks where safety must be coordinated across task execution, sequencing, and re-planning.

6.3 Future Work

Building on the foundations established in this thesis, I identify three primary research directions that I am particularly excited to pursue and that I believe are worthy of broader investigation by the community.

6.3.1 Optimality and Stability

While optimal control and reinforcement learning (RL) optimize long-term performance, they typically lack formal stability guarantees, or more broadly, a principled explanation of why the resulting policy behaves well, even within a region of interest. Lyapunov-based methods provide a foundation for formal stability certification, but constructing and scaling them to complex, high-dimensional systems remains challenging. Our work on generalized Lyapunov functions [96] bridges this gap by augmenting an RL policy's value function with a residual neural network and verifying stability through relaxed, multi-step decrease conditions.

A promising direction is to use certificate structures not only for post hoc verification, but also as part of the policy learning process. The objective is to embed stability constraints directly into training, so that the learned value function simultaneously serves as both the optimal value function and a stability certificate. Achieving this vision will require new multi-objective optimization formulations that balance performance and stability, curriculum learning strategies that progressively tighten stability requirements, and theoretical analyses linking value function structure to Lyapunov properties.

Several key research questions arise:

- How can reward functions be designed to naturally promote stability in the resulting policy?
- What approaches most effectively balance the objectives of optimality and stability during training?
- How can these methods be generalized to uncertain or stochastic systems?
- Can concepts such as robustness or input-to-state stability be leveraged to reason about the sim-to-real gap?

Pursuing these questions has the potential to influence a broad class of control and RL algorithms, improving not only performance but also interpretability and formal certification.

Such developments would be particularly impactful in safety-critical domains, including household robotics, aerospace systems, and human–robot collaboration, where both stability and explainability are essential for deployment.

6.3.2 Open-World and Contact-Rich Safety

Many existing safety filters focus on collision avoidance or other well-defined geometric constraints. While important, these approaches address only part of the safety challenges robots face in realistic environments. Two additional dimensions are particularly critical. First, **open-world safety** refers to the ability to operate in environments with incomplete or evolving knowledge, where safety depends on both geometric constraints and semantics. This includes behaviors such as adapting pouring strategies to avoid spills, slowing down when handling hazardous materials, or seeking clarification when task objectives are ambiguous. Second, **contact-rich safety** extends beyond geometric clearance to include safe physical interactions, such as regulating force, pressure, and compliance when manipulating fragile objects, handling deformable materials, or operating soft robots in delicate settings such as surgical procedures within human vessels.

A key direction is to develop unified safety frameworks that combine physical interaction constraints with contextual and semantic understanding. For contact-rich scenarios, this means incorporating force limits, pressure constraints, compliance requirements, and tactile feedback. For open-world settings, it requires safety certificates that adapt online to evolving tasks and environments, potentially leveraging real-time perception, semantic mapping, and language-based specifications.

Technical challenges include handling uncertainty from force and tactile sensors, modeling hybrid and discontinuous contact dynamics, and maintaining real-time performance. In open-world contexts, safety constraints often depend on high-level semantic interpretation, requiring continual updates to safety representations from new objects, tasks, or human feedback.

Addressing these challenges will involve integrating force-aware CBFs or alternative safety methods such as Hamilton–Jacobi reachability with tactile sensing and uncertainty-aware optimization; extending neural geometric representations to capture deformability, compliance, and semantic attributes; and creating frameworks that can handle both collision avoidance and safe physical interaction in dynamic, partially known environments.

Key research questions include:

- How can safety certificates be designed to handle deformable objects, semantic information, and evolving task constraints in open-world environments?
- What are effective formulations for unified safe control that incorporate force, pressure, and compliance constraints alongside collision avoidance?
- How can tactile sensing and force feedback be integrated into uncertainty-aware safety filters while maintaining real-time performance?
- What neural representations can jointly capture geometry, compliance, and semantic attributes of interacting objects for use in safety-critical planning and control?

Pursuing these directions will extend safety-aware control beyond simple collision avoidance, enabling robots to operate reliably in complex, dynamic, and human-centered environments. This has direct implications for applications such as surgical robotics, household assistance, manufacturing, and collaborative robotics, where both contextual understanding and safe physical interaction are essential.

6.3.3 Whole-Body Task and Motion Planning

Traditional sampling-based motion planning methods struggle in high-dimensional configuration spaces due to the exponential growth in possible states. This limitation is especially acute for whole-body robots and manipulators operating in cluttered or dynamic environments, where planning must account for complex geometry, task constraints, and real-time execution. Our recent work on neural configuration-space barriers [99] addresses this challenge by introducing configuration-space bubbles, where each graph node represents a set of configurations rather than a single point. This set-based representation enables more efficient exploration and planning while leveraging raw sensor data.

A promising direction is to extend this framework toward sensor-driven, convex decomposition of safe configuration spaces, enabling seamless integration with motion planning and high-level task planning. By providing compact, certifiably safe representations of feasible motion, these decompositions can serve as building blocks for planners to generate action sequences that are both goal-directed and safety-aware. The goal is to achieve scalable, perception-informed planning for high-dimensional robots in dynamic, unstructured environments.

The technical approach will involve developing hierarchical decompositions that capture connectivity at multiple spatial and geometric scales, creating neural representations that can efficiently compute set-valued distance functions for complex robot geometries, and designing optimization-based planners that exploit these representations for real-time performance. Establishing theoretical connections between local convex approximations and global motion planning will be important, and linking geometric safe motion planners to task-level reasoning will be essential for reliable whole-body autonomy.

Key research challenges include:

- Maintaining local convexity in safe set decompositions while accurately capturing complex robot and environment geometry, including dynamic obstacles that require rapid updates.
- Scaling to robots with many degrees of freedom while ensuring real-time performance and global completeness guarantees.
- Integrating hierarchical task planning with motion planning so that high-level action sequences remain geometrically feasible and satisfy safety constraints.
- Generalizing the C-space decomposition formulation to account for variations in grasped object geometry and semantic properties, which can alter the feasible configuration space.

Pursuing this research will enable whole-body robots to plan and execute complex tasks with reliability in domains such as warehouse automation, household assistance, manufacturing, and space operations, where scalability, safety, and adaptability are all essential.

6.4 Concluding Remarks

This thesis demonstrates that certifiable robot autonomy under uncertainty is achievable through the careful integration of control theory, optimization, and machine learning. The developed methods bridge the gap between the strong guarantees of classical control and the impressive performance of modern learning-based approaches. The nine papers comprising this thesis establish both theoretical foundations and practical algorithms that advance the state of the art in safe robot control. The open-source implementations and datasets developed provide a foundation for future research and practical deployment.

As robotic systems become increasingly prevalent in society, the importance of providing safety, stability, and reliability guarantees cannot be overstated. This work contributes to ensuring that robotic systems are not only capable and intelligent, but also safe, reliable, and worthy of human trust. The future directions outlined provide a roadmap for addressing remaining challenges while opening new directions that will benefit from certifiable robot autonomy under uncertainty.

At a broader level, the deployment of robots and policies should be accompanied by some form of **explainability, interpretability, or certificate**, rather than relying purely on large, opaque neural policies. I believe the future of robotics lies in leveraging the rigor and insights of control theory and optimization together with the scalability and representational power of modern learning approaches, including reinforcement learning, diffusion policy, vision-language-action models, and other emerging large-scale models. Such principled integration promises robotic systems that deliver consistently high performance while remaining transparent, trustworthy, and aligned with human expectations.

Bibliography

[1] Hossein Abdi, Golnaz Raja, and Reza Ghabcheloo. Safe control using vision-based control barrier function (V-CBF). In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 782–788, 2023.

[2] Mohamadreza Ahmadi, Xiaobin Xiong, and Aaron D. Ames. Risk-averse control via CVaR barrier functions: Application to bipedal robot locomotion. *IEEE Control Systems Letters*, 6:878–883, 2022.

[3] Anil Alan, Andrew J. Taylor, Chaozhe R. He, Gábor Orosz, and Aaron D. Ames. Safe controller synthesis with tunable input-to-state safe control barrier functions. *IEEE Control Systems Letters*, 6:908–913, 2022.

[4] Gokhan Alcan and Ville Kyrki. Differential dynamic programming with nonlinear safety constraints under system uncertainties. *IEEE Robotics and Automation Letters*, 7(2):1760–1767, 2022.

[5] Farid Alizadeh and Donald Goldfarb. Second-order cone programming. *Mathematical programming*, 95(1):3–51, 2003.

[6] Hassan Almubarak, Kyle Stachowicz, Nader Sadegh, and Evangelos A. Theodorou. Safety embedded differential dynamic programming using discrete barrier states. *IEEE RAL*, 7(2):2755–2762, 2022.

[7] Aaron Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *European Control Conference (ECC)*, pages 3420–3431, 2019.

[8] Aaron Ames, Xiangru Xu, Jessy Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2016.

[9] Aaron D Ames, Kevin Galloway, Koushil Sreenath, and Jessy W Grizzle. Rapidly exponentially stabilizing control Lyapunov functions and hybrid zero dynamics. *IEEE Transactions on Automatic Control*, 59(4):876–891, 2014.

[10] Brian DO Anderson and John B Moore. *Optimal control: linear quadratic methods*. Courier Corporation, 2007.

[11] Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[12] Liviu Aolaritei, Marta Fochesato, John Lygeros, and Florian Dörfler. Wasserstein tube MPC with exact uncertainty propagation. In *IEEE Conference on Decision and Control (CDC)*, pages 2036–2041, 2023.

[13] Zvi Artstein. Stabilization with relaxed controls. *Nonlinear Analysis-theory Methods & Applications*, 7:1163–1173, 1983.

[14] Bolton Bailey, Ziwei Ji, Matus Telgarsky, and Ruicheng Xian. Approximation power of random neural networks. *ArXiv*, abs/1906.07709, 2019.

[15] Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *Advances in neural information processing systems*, volume 30, page 908–919, 2017.

[16] Lars Blackmore, Masahiro Ono, and Brian C Williams. Chance-constrained optimal path planning with obstacles. *IEEE Transactions on Robotics*, 27(6):1080–1094, 2011.

[17] Nicholas Boffi, Stephen Tu, Nikolai Matni, Jean-Jacques Slotine, and Vikas Sindhwani. Learning stability certificates from data. In *Conference on Robot Learning*, pages 1341–1350. PMLR, 2021.

[18] D. Boskos, J. Cortés, and S. Martinez. Data-driven ambiguity sets with probabilistic guarantees for dynamic processes. *IEEE Transactions on Automatic Control*, 66(7):2991–3006, 2021.

[19] D. Boskos, J. Cortés, and S. Martínez. High-confidence data-driven ambiguity sets for time-varying linear systems. *IEEE Transactions on Automatic Control*, 69(2):797–812, 2024.

[20] F. Boso, D. Boskos, J. Cortés, S. Martínez, and D. M. Tartakovsky. Dynamics of data-driven ambiguity sets for hyperbolic conservation laws with uncertain inputs. *SIAM Journal on Scientific Computing*, 43(3):A2102–A2129, 2021.

[21] Joseph Breeden and Dimitra Panagou. Robust control barrier functions under high relative degree and input constraints for satellite trajectories. *Automatica*, 155:111109, 2023.

[22] F. P. Cantelli. Sui confini della probabilità. *Atti del Congresso Internazionale dei Matematici*, 6:47–60, 1929.

[23] Hieronymus Cardano. *Artis magnae, sive de regulis algebraicis, liber unus*. Joh. Petreius, 2011.

[24] Ya-Chien Chang, Nima Roohi, and Sicun Gao. Neural Lyapunov control. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[25] Xuemin Chi, Yiming Li, Jihao Huang, Bolun Dai, Zhitao Liu, and Sylvain Calinon. Safe dynamic motion generation in configuration space using differentiable distance fields. *arXiv preprint arXiv:2412.16456*, 2024.

[26] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A Lyapunov-based approach to safe reinforcement learning. In *Advances in neural information processing systems*, volume 31, 2018.

[27] Andrew Clark. Control barrier functions for complete and incomplete information stochastic systems. In *ACC*, pages 2928–2935, 2019.

[28] Andrew Clark. Control barrier functions for complete and incomplete information stochastic systems. In *2019 American Control Conference (ACC)*, pages 2928–2935, 2019.

[29] Ryan K Cosner, Andrew W Singletary, Andrew J Taylor, Tamas G Molnar, Katherine L Bouman, and Aaron D Ames. Measurement-robust control barrier functions: Certainty in safety with uncertainty in state. *arXiv*, abs/2104.14030, 2021.

[30] Jeremy Coulson, John Lygeros, and Florian Dörfler. Distributionally robust chance constrained data-enabled predictive control. *IEEE Transactions on Automatic Control*, 67(7):3289–3304, 2021.

[31] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021.

[32] Tianxiang Cui, Shusheng Ding, Huan Jin, and Yongmin Zhang. Portfolio constructions in cryptocurrency market: A cvar-based deep reinforcement learning approach. *Economic Modelling*, 119:106078, 2023.

[33] Hongkai Dai, Benoit Landry, Lujie Yang, Marco Pavone, and Russ Tedrake. Lyapunov-stable neural-network control. In *Proceedings of Robotics: Science and Systems*, Virtual, July 2021.

[34] Ersin Das and Joel W Burdick. Robust control barrier functions using uncertainty estimation with application to mobile robots. *arXiv preprint arXiv:2401.01881*, 2024.

[35] Charles Dawson, Sicun Gao, and Chuchu Fan. Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control. *IEEE Transactions on Robotics*, 39(3):1749–1767, 2023.

[36] Charles Dawson, Zengyi Qin, Sicun Gao, and Chuchu Fan. Safe nonlinear control using robust neural Lyapunov-barrier functions. In *Conference on Robot Learning*, volume 164, pages 1724–1735. PMLR, 2022.

[37] Ersin Daş and Richard M. Murray. Robust safe control synthesis with disturbance observer-based control barrier functions. In *IEEE Conference on Decision and Control (CDC)*, pages 5566–5573, 2022.

[38] Daniela Pucci De Farias and Benjamin Van Roy. The linear programming approach to approximate dynamic programming. *Operations research*, 51(6):850–865, 2003.

[39] Vikas Dhiman*, Mohammad Javad Khojasteh*, Massimo Franceschetti, and Nikolay Atanasov. Control barriers in bayesian learning of system dynamics. *IEEE Transactions on Automatic Control*, 2021.

[40] Vikas Dhiman, Mohammad Javad Khojasteh, Massimo Franceschetti, and Nikolay Atanasov. Control barriers in bayesian learning of system dynamics. *IEEE Transactions on Automatic Control*, 68(1):214–229, 2023.

[41] E W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[42] Anushri Dixit, Lars Lindemann, Skylar X Wei, Matthew Cleaveland, George J Pappas, and Joel W Burdick. Adaptive conformal prediction for motion planning among dynamic agents. In *Learning for Dynamics and Control Conference*, pages 300–314. PMLR, 2023.

[43] Petar M Djuric, Jayesh H Kotecha, Jianqui Zhang, Yufei Huang, Tadesse Ghirmai, Mónica F Bugallo, and Joaquin Miguez. Particle filtering. *IEEE signal processing magazine*, 20(5):19–38, 2003.

[44] Thai Duong and Nikolay Atanasov. Adaptive control of se(3) hamiltonian dynamics with learned disturbance features. *IEEE Control Systems Letters*, 6:2773–2778, 2022.

[45] Yousef Emam, Paul Glotfelter, and Magnus Egerstedt. Robust barrier functions for a fully autonomous, remotely accessible swarm-robotics testbed. In *IEEE Conference on Decision and Control*, pages 3984–3990, 2019.

[46] Yousef Emam, Paul Glotfelter, Sean Wilson, Gennaro Notomista, and Magnus Egerstedt. Data-driven robust barrier functions for safe, long-term operation. *IEEE Transactions on Robotics*, 38(3):1671–1685, 2022.

[47] Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the Wasserstein metric: performance guarantees and tractable reformulations. *Mathematical Programming*, 171:115–166, 2018.

[48] Annika Fürnsinn, Christian Ebenbauer, and Bahman Gharesifard. Relaxed feasibility and stability criteria for flexible-step MPC. *IEEE Control Systems Letters*, 7:2851–2856, 2023.

[49] Annika Fürnsinn, Christian Ebenbauer, and Bahman Gharesifard. Flexible-step model predictive control based on generalized Lyapunov functions. *Automatica*, 175:112215, 2025.

[50] Nathan Gaby, Fumin Zhang, and Xiaojing Ye. Lyapunov-net: A deep neural network architecture for Lyapunov function approximation. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 2091–2096, 2022.

[51] Sicun Gao, Soonho Kong, and Edmund M Clarke. dreal: An smt solver for nonlinear theories over the reals. In *International conference on automated deduction*, pages 208–214. Springer, 2013.

[52] Giorgio Grisetti, Rainer Kümmerle, Cyrill Stachniss, and Wolfram Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

[53] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[54] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2021.

[55] Astghik Hakobyan, Gyeong Chan Kim, and Insoon Yang. Risk-aware motion planning and control using cvar-constrained optimization. *IEEE Robotics and Automation letters*, 4(4):3924–3931, 2019.

[56] Astghik Hakobyan and Insoon Yang. Distributionally robust risk map for learning-based motion planning and control: A semidefinite programming approach. *IEEE Transactions on Robotics*, 39(1):718–737, 2022.

[57] Astghik Hakobyan and Insoon Yang. Distributionally robust differential dynamic programming with Wasserstein distance. *IEEE Control Systems Letters*, 7:2329–2334, 2023.

[58] Luxin Han, Fei Gao, Boyu Zhou, and Shaojie Shen. Fiesta: Fast incremental euclidean distance fields for online motion planning of aerial robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[59] Luxin Han, Fei Gao, Boyu Zhou, and Shaojie Shen. Fiesta: Fast incremental euclidean distance fields for online motion planning of aerial robots. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4423–4430, 2019.

[60] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *The Twelfth International Conference on Learning Representations*, 2024.

[61] Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *Proceedings of the 39th International Conference on Machine Learning*, pages 8387–8406. PMLR, 2022.

[62] James Harrison, Apoorva Sharma, and Marco Pavone. Meta-learning priors for efficient online bayesian regression. In Marco Morales, Lydia Tapia, Gildardo Sánchez-Ante, and Seth Hutchinson, editors, *Algorithmic Foundations of Robotics XIII*, pages 318–337, Cham, 2020. Springer International Publishing.

[63] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[64] Kris Hauser. Lazy collision checking in asymptotically-optimal motion planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2951–2957, 2015.

[65] John D Head and Michael C Zerner. A broyden—fletcher—goldfarb—shanno optimization procedure for molecular geometries. *Chemical physics letters*, 122(3):264–270, 1985.

[66] Bilal Hejase and Umit Ozguner. Lyapunov stability regulation of deep reinforcement learning control with application to automated driving. In *2023 American Control Conference (ACC)*, pages 4437–4442. IEEE, 2023.

[67] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical Review E*, 51(5):4282, 1995.

[68] Lukas Hewing, Juraj Kabzan, and Melanie N. Zeilinger. Cautious model predictive control using gaussian process regression. *IEEE Transactions on Control Systems Technology*, 28(6):2736–2743, 2020.

[69] David Hilbert. Ueber die darstellung definiter formen als summe von formenquadraten. *Mathematische Annalen*, 32:342–350, September 1888.

[70] Armin Hornung, Kai Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206, 2013.

[71] Ashish Ranjan Hota, Ashish Kumar Cherukuri, and John Lygeros. Data-driven chance constrained optimization under Wasserstein ambiguity sets. In *American Control Conference (ACC)*, pages 1501–1506, 2019.

[72] Mrdjan Jankovic. Robust control barrier functions for constrained stabilization of nonlinear systems. *Automatica*, 96:359, 2018.

[73] Z. Jarvis-Wloszek, R. Feeley, Weehong Tan, Kunpeng Sun, and A. Packard. Some controls applications of sum of squares programming. In *42nd IEEE International Conference on Decision and Control*, volume 5, pages 4676–4681, 2003.

[74] Ruiwei Jiang and Yongpei Guan. Data-driven chance constrained stochastic program. *Mathematical Programming*, 158:291–327, 2016.

[75] R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[76] Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, 2023.

[77] Shaghayegh Keyumarsi, Made Widhi Surya Atman, and Azwirman Gusrialdi. Lidar-based online control barrier function synthesis for safe navigation in unknown environments. *IEEE Robotics and Automation Letters*, 9(2):1043–1050, 2024.

[78] Hassan Khalil. *Nonlinear Systems*. Prentice Hall, 1996.

[79] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint: 1412.6980*, 2015.

[80] Shishir Kolathaya and Aaron D. Ames. Input-to-state safety with control barrier functions. *IEEE CSL*, 3(1):108–113, 2019.

[81] Mikhail Koptev, Nadia Figueroa, and Aude Billard. Neural joint space implicit signed distance functions for reactive robot manipulator control. *IEEE Robotics and Automation Letters*, 8(2):480–487, 2023.

[82] Arash Bahari Kordabad, Rafael Wisniewski, and Sebastien Gros. Safe reinforcement learning using Wasserstein distributionally robust mpc and chance constraint. *IEEE Access*, 10:130058–130067, 2022.

[83] J.J. Kuffner and S.M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, volume 2, pages 995–1001 vol.2, 2000.

[84] Paul Lathrop, Beth Boardman, and Sonia Martínez. Distributionally safe path planning: Wasserstein safe RRT. *IEEE Robotics and Automation Letters*, 7(1):430–437, 2021.

[85] Monique Laurent. *Sums of Squares, Moment Matrices and Optimization Over Polynomials*, pages 157–270. Springer New York, New York, NY, 2009.

[86] Steven M LaValle and James J Kuffner. Rapidly-exploring random trees: Progress and prospects. *Algorithmic and computational robotics*, pages 303–307, 2001.

[87] Ki Myung Brian Lee, Zhirui Dai, Cedric Le Gentil, Lan Wu, Nikolay Atanasov, and Teresa Vidal-Calleja. Safe bubble cover for motion planning on distance fields. *arXiv preprint arXiv:2408.13377*, 2024.

[88] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[89] Bin Li, Yuan Tan, Ai-Guo Wu, and Guang-Ren Duan. A distributionally robust optimization based method for stochastic model predictive control. *IEEE Transactions on Automatic Control*, 67(11):5762–5776, 2021.

[90] Yiming Li, Xuemin Chi, Amirreza Razmjoo, and Sylvain Calinon. Configuration space distance fields for manipulation planning. In *Robotics: Science and Systems (RSS)*, 2024.

[91] Yiming Li, Yan Zhang, Amirreza Razmjoo, and Sylvain Calinon. Representing robot geometry as distance fields: Applications to whole-body manipulation. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 15351–15357, 2024.

[92] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[93] Chen-Hsuan Lin, Chaoyang Wang, and Simon Lucey. Sdf-srn: Learning signed distance 3d object reconstruction from static images. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[94] Lars Lindemann, Matthew Cleaveland, Gihyun Shim, and George J. Pappas. Safe planning in dynamic environments using conformal prediction. *IEEE Robotics and Automation Letters*, 8(8):5116–5123, 2023.

[95] Jinze Liu, Minzhe Li, Jiunn-Kai Huang, and Jessy W Grizzle. Realtime safety control for bipedal robots to avoid multiple obstacles via CLF-CBF constraints. *arXiv preprint arXiv:2301.01906*, 2023.

[96] Kehan Long, Jorge Cortés, and Nikolay Atanasov. Certifying stability of reinforcement learning policies using generalized lyapunov functions. *arXiv preprint arXiv:2505.10947*, 2025.

[97] Kehan Long, Jorge Cortés, and Nikolay Atanasov. Distributionally robust policy and lyapunov-certificate learning. *IEEE Open Journal of Control Systems*, 3:375–388, 2024.

[98] Kehan Long, Vikas Dhiman, Melvin Leok, Jorge Cortés, and Nikolay Atanasov. Safe control synthesis with uncertain dynamics and constraints. *IEEE Robotics and Automation Letters*, 7(3):7295–7302, 2022.

[99] Kehan Long, Ki Myung Brian Lee, Nikola Raicevic, Niyas Attasseri, Melvin Leok, and Nikolay Atanasov. Neural configuration-space barriers for manipulation planning and control. *arXiv preprint arXiv:2503.04929*, 2025.

[100] Kehan Long, Hardik Parwana, Georgios Fainekos, Bardh Hoxha, Hideki Okamoto, and Nikolay Atanasov. Neural configuration distance function for continuum robot control. *arXiv preprint arXiv:2409.13865*, 2024.

[101] Kehan Long, Cheng Qian, Jorge Cortés, and Nikolay Atanasov. Learning barrier functions with memory for robust safe navigation. *IEEE Robotics and Automation Letters*, 6(3):4931–4938, 2021.

[102] Kehan Long, Khoa Tran, Melvin Leok, and Nikolay Atanasov. Safe stabilizing control for polygonal robots in dynamic elliptical environments. In *2024 American Control Conference (ACC)*, pages 312–317, 2024.

[103] Kehan Long, Yinzhuang Yi, Jorge Cortes, and Nikolay Atanasov. Distributionally robust Lyapunov function search under uncertainty. In *Learning for Dynamics and Control Conference*, pages 864–877. PMLR, 2023.

[104] Kehan Long, Yinzhuang Yi, Jorge Cortés, and Nikolay Atanasov. Safe and stable control synthesis for uncertain system models via distributionally robust optimization. In *American Control Conference (ACC)*, pages 4651–4658, 2023.

[105] Kehan Long, Yinzhuang Yi, Zhirui Dai, Sylvia Herbert, Jorge Cortés, and Nikolay Atanasov. Sensor-based distributionally robust control for safe robot navigation in dynamic environments. *The International Journal of Robotics Research*, 2025.

[106] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *COMPUTER GRAPHICS*, 21(4):163–169, 1987.

[107] Lars M. Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4455–4465, 2019.

[108] Pol Mestres, Kehan Long, Nikolay Atanasov, and Jorge Cortés. Feasibility analysis and regularity characterization of distributionally robust safe stabilizing controllers. *IEEE Control Systems Letters*, 8:91–96, 2024.

[109] Mayank Mittal, Marco Gallieri, Alessio Quaglino, Seyed Sina Mirrazavi Salehian, and Jan Koutník. Neural Lyapunov model predictive control: Learning safe global controllers from sub-optimal examples. *arXiv preprint arXiv:2002.10451*, 2020.

[110] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[111] Mehdi Moussaïd, Niriaska Perozo, Simon Garnier, Dirk Helbing, and Guy Theraulaz. The walking behaviour of pedestrian social groups and its impact on crowd dynamics. *PloS one*, 5(4):e10047, 2010.

[112] Arkadi Nemirovski and Alexander Shapiro. Convex approximations of chance constrained programs. *SIAM J. Optim.*, 17:969–996, 2006.

[113] Quan Nguyen and Koushil Sreenath. Robust safety-critical control for dynamic robotics. *IEEE Transactions on Automatic Control*, 2021.

[114] Helen Oleynikova, Zachary Taylor, Marius Fehr, Roland Siegwart, and Juan Nieto. Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1366–1373, 2017.

[115] Antonis Papachristodoulou and Stephen Prajna. On the construction of Lyapunov functions using the sum of squares decomposition. In *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, volume 3, pages 3482–3487 vol.3, 2002.

[116] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[117] Pablo Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. California Institute of Technology, 2000.

[118] Theodore J Perkins and Andrew G Barto. Lyapunov design for safe reinforcement learning. *Journal of Machine Learning Research*, 3(Dec):803–832, 2002.

[119] Romain Postoyan, Lucian Buşoniu, Dragan Nešić, and Jamal Daafouz. Stability analysis of discrete-time infinite-horizon optimal control with discounted cost. *IEEE Transactions on Automatic Control*, 62(6):2736–2749, 2017.

[120] S. Prajna. Barrier certificates for nonlinear model validation. In *Conference on Decision and Control (CDC)*, pages 2884–2889, 2003.

[121] Stephen Prajna and Ali Jadbabaie. Safety verification of hybrid systems using barrier certificates. In *Hybrid Systems: Computation and Control*, pages 477–492. Springer Berlin Heidelberg, 2004.

[122] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[123] James B Rawlings. Tutorial overview of model predictive control. *IEEE control systems magazine*, 20(3):38–52, 2000.

[124] Allen Z. Ren and Anirudha Majumdar. Distributionally robust policy learning via adversarial environment generation. *IEEE Robotics and Automation Letters*, 7(2):1379–1386, 2022.

[125] Spencer M Richards, Felix Berkenkamp, and Andreas Krause. The Lyapunov neural network: Adaptive stability certification for safe learning of dynamical systems. In *Conference on Robot Learning*, pages 466–476. PMLR, 2018.

[126] Ralph Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2:21–41, 2000.

[127] Muhammad Zakiyullah Romdlony and Bayu Jayawardhana. On the new notion of input-to-state safety. In *IEEE CDC*, pages 6403–6409, 2016.

[128] Kanghyun Ryu and Negar Mehr. Integrating predictive motion uncertainties with distributionally robust risk-aware control for safe robot navigation in crowds. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2410–2417, 2024.

[129] Sleiman Safaoui and Tyler H Summers. Distributionally robust cvar-based safety filtering for motion planning in uncertain environments. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 103–109. IEEE, 2024.

[130] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[131] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[132] Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(3), 2008.

[133] Eduardo D. Sontag. A 'universal' construction of Artstein's theorem on nonlinear stabilization. *Systems & Control Letters*, 13(2):117–123, 1989.

[134] M. Srinivasan, Amogh Dabholkar, Samuel D. Coogan, and P. Vela. Synthesis of control barrier functions using a supervised machine learning approach. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7139–7145, 2020.

[135] Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012. https://ompl.kavrakilab.org.

[136] Tyler Summers. Distributionally robust sampling-based motion planning under uncertainty. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6518–6523, 2018.

[137] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[138] Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

[139] Bart P. G. Van Parys, Daniel Kuhn, Paul J. Goulart, and Manfred Morari. Distributionally robust control of constrained stochastic systems. *IEEE Transactions on Automatic Control*, 61(2):430–442, 2016.

[140] Vladimir Vovk. Conditional validity of inductive conformal predictors. In *Asian conference on machine learning*, pages 475–490. PMLR, 2012.

[141] Li Wang, Aaron Ames, and Magnus Egerstedt. Safety barrier certificates for collisions-free multirobot systems. In *IEEE Transactions on Robotics*, pages 1–14, 2017.

[142] Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J Zico Kolter. Beta-CROWN: Efficient bound propagation with per-neuron split constraints for complete and incomplete neural network verification. *Advances in Neural Information Processing Systems*, 34, 2021.

[143] Yujie Wang and Xiangru Xu. Disturbance observer-based robust control barrier functions. In *American Control Conference (ACC)*, pages 3681–3687, 2023.

[144] Peter Wieland and Frank Allgöwer. Constructive safety using control barrier functions. In *IFAC Proceedings Volumes*, pages 462–467, 2007.

[145] Junlin Wu, Andrew Clark, Yiannis Kantaros, and Yevgeniy Vorobeychik. Neural Lyapunov control for discrete-time systems. In *Advances in neural information processing systems*, volume 36, pages 2939–2955, 2023.

[146] Lan Wu, Ki Myung Brian Lee, Liyang Liu, and Teresa Vidal-Calleja. Faithful Euclidean distance field from log-gaussian process implicit surfaces. *IEEE Robotics and Automation Letters*, 6(2):2461–2468, 2021.

[147] Weijun Xie. On distributionally robust chance constrained programs with Wasserstein distance. *Math. Program.*, 186:115–155, 2021.

[148] Xiangru Xu, Thomas Waters, Daniel Pickem, Paul Glotfelter, Magnus Egerstedt, Paulo Tabuada, Jessy W Grizzle, and Aaron D Ames. Realizing simultaneous lane keeping and adaptive speed regulation on accessible mobile robot testbeds. In *IEEE Conference on Control Technology and Applications (CCTA)*, pages 1769–1775, 2017.

[149] Shuhao Yan, Paul Goulart, and Mark Cannon. Stochastic model predictive control with discounted probabilistic constraints. In *2018 European Control Conference (ECC)*, pages 1003–1008. IEEE, 2018.

[150] Insoon Yang. Wasserstein distributionally robust stochastic control: A data-driven approach. *IEEE Transactions on Automatic Control*, 66(8):3863–3870, 2020.

[151] Lujie Yang, Hongkai Dai, Zhouxing Shi, Cho-Jui Hsieh, Russ Tedrake, and Huan Zhang. Lyapunov-stable neural control for state and output feedback: A novel formulation. In *Forty-first International Conference on Machine Learning*, 2024.

[152] Shuo Yang, George J Pappas, Rahul Mangharam, and Lars Lindemann. Safe perception-based control under stochastic sensor uncertainty using conformal prediction. In *IEEE Conference on Decision and Control (CDC)*, pages 6072–6078, 2023.

[153] Dmitry Yarotsky. Error bounds for approximations with deep relu networks. *Neural Networks*, 94:103–114, 2017.

[154] Hongzhan Yu, Chiaki Hirayama, Chenning Yu, Sylvia Herbert, and Sicun Gao. Sequential neural barriers for scalable dynamic obstacle avoidance. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11241–11248, 2023.

[155] Mingxin Yu, Chenning Yu, M-Mahdi Naddaf-Sh, Devesh Upadhyay, Sicun Gao, and Chuchu Fan. Efficient motion planning for manipulators with control barrier function-induced neural controller. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14348–14355, 2024.

[156] Qiyuan Zhang, Shu Leng, Xiaoteng Ma, Qihan Liu, Xueqian Wang, Bin Liang, Yu Liu, and Jun Yang. Cvar-constrained policy optimization for safe reinforcement learning. *IEEE transactions on neural networks and learning systems*, 36(1):830–841, 2024.

[157] Yiqi Zhao, Xinyi Yu, Jyotirmoy V Deshmukh, and Lars Lindemann. Conformal predictive programming for chance constrained optimization. *arXiv preprint arXiv:2402.07407*, 2024.

[158] Ruikun Zhou, Thanin Quartz, Hans De Sterck, and Jun Liu. Neural Lyapunov control of unknown nonlinear systems with stability guarantees. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

[159] Xiankun Zhu, Yucheng Xin, Shoujie Li, Houde Liu, Chongkun Xia, and Bin Liang. Efficient collision detection framework for enhancing collision-free robot motion. *arXiv preprint arXiv:2409.14955*, 2024.