# Safe Control Synthesis with Uncertain Dynamics and Constraints

Kehan Long[1] Vikas Dhiman[2] Melvin Leok[1] Jorge Cortés[1] Nikolay Atanasov[1]

***Abstract*— This paper considers safe control synthesis for dynamical systems in the presence of uncertainty in the dynamics model and the safety constraints that the system must satisfy. Our approach captures probabilistic and worst-case model errors and their effect on control Lyapunov function (CLF) and control barrier function (CBF) constraints in the control-synthesis optimization problem. We show that both the probabilistic and robust formulations lead to second-order cone programs (SOCPs), enabling safe and stable control synthesis that can be performed efficiently online. We evaluate our approach in PyBullet simulations of an autonomous robot navigating in unknown environments and compare the performance with a baseline CLF-CBF quadratic programming approach.**

## I. Introduction

Autonomous robotic systems are increasingly employed in warehouse and home automation, transportation, and security applications. A crucial aspect of successfully deploying such systems is the satisfaction of safety and stability requirements, even in the presence of uncertainty in the system model or constraints. The notion of safety in the context of program correctness was first introduced in the 1970's [1], [2]. Around the same time, Artstein [3] introduced control Lyapunov functions (CLFs) to enforce stability in the context of nonlinear system control. The seminal work of Sontag [4] established a universal formula for constructing feedback control laws that stabilize nonlinear systems. In the 2000's, barrier certificates were proposed to formally prove the safety of closed-loop nonlinear and hybrid systems [5], [6]. Control barrier functions (CBFs) were developed to support task-independent safe control synthesis, serving as a barrier certificate for a closed-loop nonlinear system [7].

A key observation is that, for control-affine systems, the CLF and CBF conditions are linear in the control input, allowing a formulation of safe and stable control synthesis as a quadratic program (QP) [8]–[11]. CLF-CBF-QP techniques have been successfully employed in a variety systems, including aerial robots [12], walking robots [13], and automotive systems [14]. Most existing work, however, assumes complete knowledge of the system dynamics and control barrier functions. In reality, the dynamics model and safety constraints are obtained using noisy sensor data and simplifying assumptions, leading to uncertainty and errors that should be captured when ensuring safety and stability.

We gratefully acknowledge support from NSF RI IIS-2007141.

[1] The authors are with the Contextual Robotics Institute, University of California San Diego, La Jolla, CA 92093, USA. {k3long,mleok,cortes,natanasov}@ucsd.edu

[2] V. Dhiman is with Department of Electrical and Computer Engineering, University of Maine, Bangor, ME 04469, USA. {vikas.dhiman}@maine.edu

Capturing system-model and barrier-function estimation errors impacts the formulation of CLF and CBF constraints, and no longer give rise to QPs. Our main contribution is to show that such uncertainty-aware stability and safety constraints can still be formulated as convex constraints under two different models of uncertainty: probabilistic and worst-case. To capture probabilistic uncertainty, we specifically consider *Gaussian Process* (GP) regression [15] as an example approach for modeling a probability distribution over a function space. When the estimated barrier function and system dynamics are described by a GP, we aim to ensure probabilistic safety and stability up to a user-specified risk tolerance. We compute the distribution of the CLF and CBF constraints, and use Cantelli's inequality [16] to bound the computed means with a margin dependent on the variances and the desired risk-tolerance. The control input appears linearly in the mean and quadratically in the variance of the CLF and CBF constraints. This allows us to restate the probabilistic constraints as second-order cone constraints, leading to a second-order cone program (SOCP), which is convex and can be solved efficiently online.

Alternatively, when *worst-case error bounds* on the system dynamics, barrier function and its gradient are given, the goal is to formulate a robust safe control synthesis problem. Under worst-case disturbances, we show that the input appears both linearly and within a norm term in the CLF and CBF constraints. Similar to the probabilistic formulation, the original QP problem can be reformulated as a convex SOCP for safe control synthesis.

We demonstrate our safe control synthesis techniques in mobile robot navigation simulations. We consider a robot tasked to follow a desired path in an unknown environment, relying on online noisy obstacle sensing and offline dynamic model estimation to ensure safety and stability. We show that both the probabilistic and the robust CLF-CBF-SOCP formulation allows the robot to safely track the deisred path.

In summary, we make the following **contributions**. First, we formulate novel probabilistic safety and stability constraints by considering stochastic uncertainty in the barrier functions and system dynamics. Second, we formulate novel robust safety and stability constraints by considering worst-case error bounds in the barrier functions and system dynamics. Finally, we show that either the probabilistic or the worst-case formulations lead to a (convex) SOCP, enabling efficient synthesis of safe and stable control.

## II. Related Work

CLF-CBF-QP optimization techniques [11] have gained significantly popularity for enforcing stability and safety for

robotic systems. While the original formulation does not take uncertainty into account, several recent works address uncertainty due to unmodeled dynamics, input disturbances, and barrier function estimation separately.

For system dynamics uncertainty, Jankovic [17] considered worst-case disturbance bounds on the dynamics and proposed robust CBF formulations with a modified QP. Eman *et al.* [18] utilized convex hulls to model disturbances in a CBF-based safety framework. Clark [19] considered stochastic control systems with incomplete information and derived sufficient conditions for ensuring safety on average. Nguyen and Sreenath [20] formulated a robust CLF-CBF-QP by introducing additional robust constraints to guarantee stability and safety under model uncertainty. Ahmadi *et al.* [21] introduced a conditional value-at-risk (CVaR) barrier function to ensure safety for discrete-time systems subject to stochastic uncertainty. In our previous work [22], we proposed Bayesian learning methods to obtain the distribution of system dynamics online and enable safe control synthesis while taking system dynamics uncertainty into account.

For input disturbances, the concept of *input-to-state safety* (ISSf) was introduced by Romdlony and Jayawardhana in [23]. The ISSf-CBF concept was extended in [24], which focused on enlarging the safe set by modifying CBF. Recently, Alan *et al.* [25] introduced the tunable ISSf-CBF concept for safe controller synthesis while reducing conservatism. In addition, Cosner *et al.* in [26] introduced measurement-robust CBFs to account for uncertainty in state estimation and conducted experiments on a Segway.

For barrier function uncertainty, Srinivasan *et al.* [27] estimated barrier functions online using a Support Vector Machine approach and solved the CLF-CBF-QP to generate safe control inputs. Our previous work [28] computed worst-case error bounds of barrier function constraints and formulated a robust CLF-CBF-SOCP that enforces safety and stability. Zhang *et al.* [29] constructed robust output CBFs from safe expert demonstrations while considering worst-case error bounds in measurement map and system dynamics.

In this work, we provide a unified formulation for safe control synthesis while taking either probabilistic or worst-case uncertainty of the system dynamics and barrier function simultaneously into account.

## III. Problem Formulation

Consider a robot with dynamics model:

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

where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ is the robot state and $\underline{\mathbf{u}} \in \underline{\mathcal{U}} \subseteq \mathbb{R}^{m+1}$ is the control input.[1] We assume $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ and $g : \mathbb{R}^n \mapsto \mathbb{R}^{n\times m}$ are continuously differentiable. The admissible control input space is given by $\underline{\mathcal{U}} := \{\underline{\mathbf{u}} \in \mathbb{R}^{m+1} \,|\, \underline{\mathbf{u}}^\top \boldsymbol{e}_1 = 1,\ \mathbf{A}\underline{\mathbf{u}} \leq \boldsymbol{b}\}$, where $\boldsymbol{e}_1 \in \mathbb{R}^{m+1}$ is the basis vector with 1 at the first entry and 0 elsewhere, $\mathbf{A} \in \mathbb{R}^{k\times(m+1)}$ and $\boldsymbol{b} \in \mathbb{R}^k$.

**Definition III.1.** A continuously differentiable function $V : \mathbb{R}^n \mapsto \mathbb{R}_{\geq 0}$ is a *control Lyapunov function (CLF)* for the system (1) if there exists a class $\mathcal{K}$ function $\alpha_V$ such that:

$$\inf_{\underline{\mathbf{u}}\in\underline{\mathcal{U}}} \mathit{CLC}(\mathbf{x}, \underline{\mathbf{u}}) \leq 0, \quad \forall \mathbf{x} \in \mathcal{X}, \tag{2}$$

where the *control Lyapunov condition (CLC)* is:

$$\begin{aligned} \mathit{CLC}(\mathbf{x}, \underline{\mathbf{u}}) &\triangleq \mathcal{L}_f V(\mathbf{x}) + \mathcal{L}_g V(\mathbf{x})\mathbf{u} + \alpha_V(V(\mathbf{x})) \\ &= [\nabla_{\mathbf{x}} V(\mathbf{x})]^\top F(\mathbf{x})\underline{\mathbf{u}} + \alpha_V(V(\mathbf{x})). \end{aligned} \tag{3}$$

A CLF $V$ may be used to encode a variety of control objectives, including path following [28], adaptive cruise control [14], and bipedal robot walking [13].

To define safety requirements for the control objective, consider a continuously differentiable function $h : \mathbb{R}^n \mapsto \mathbb{R}$, which implicitly defines a (closed) safe set of system states $\mathcal{S} \triangleq \{\mathbf{x} \in \mathcal{X} \mid h(\mathbf{x}) \geq 0\}$. The following definition is a useful tool to ensure that $\mathcal{S}$ is forward invariant, i.e., the robot state remains in $\mathcal{S}$ throughout its evolution.

**Definition III.2.** A continuously differentiable function $h : \mathbb{R}^n \mapsto \mathbb{R}$ is a *control barrier function (CBF)* on $\mathcal{X} \subseteq \mathbb{R}^n$ for the system (1) if there exists an extended class $\mathcal{K}_\infty$ function $\alpha_h$ such that:

$$\sup_{\underline{\mathbf{u}}\in\underline{\mathcal{U}}} \mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0, \quad \forall \mathbf{x} \in \mathcal{X}, \tag{4}$$

where the *control barrier condition (CBC)* is:

$$\begin{aligned} \mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}}) &\triangleq \mathcal{L}_f h(\mathbf{x}) + \mathcal{L}_g h(\mathbf{x})\mathbf{u} + \alpha_h(h(\mathbf{x})) \\ &= [\nabla_{\mathbf{x}} h(\mathbf{x})]^\top F(\mathbf{x})\underline{\mathbf{u}} + \alpha_h(h(\mathbf{x})). \end{aligned} \tag{5}$$

According to [9], [11], any Lipschitz-continuous controller $\underline{\mathbf{k}} : \mathcal{X} \mapsto \underline{\mathcal{U}}$ that satisfies $\mathit{CBC}(\mathbf{x}, \underline{\mathbf{k}}(\mathbf{x})) \geq 0$ for all $\mathbf{x} \in \mathcal{X}$ renders the set $\mathcal{S}$ forward invariant for the system (1).

### A. Safety and Stability with Known System Dynamics and Barrier Function

When the system dynamics $F(\mathbf{x})$ and the barrier function $h(\mathbf{x})$ are precisely known, one can combine CLF and CBF constraints to synthesize a safe controller via the following quadratic program:

$$\begin{aligned} \min_{\underline{\mathbf{u}}\in\underline{\mathcal{U}},\delta\in\mathbb{R}}\ & \|L(\mathbf{x})^\top(\underline{\mathbf{u}} - \tilde{\underline{\mathbf{k}}}(\mathbf{x}))\|^2 + \lambda\delta^2, \\ \text{s.t. } & \mathit{CLC}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta,\ \mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0. \end{aligned} \tag{6}$$

The term $\tilde{\underline{\mathbf{k}}}(\mathbf{x})$ is a baseline controller and may be used to specify additional control requirements, such as desirable velocity or orientation. This term may be set to $\tilde{\underline{\mathbf{k}}}(\mathbf{x}) \equiv \boldsymbol{e}_1$ if minimum control effort is the main objective. The term $L(\mathbf{x})$ is a weighting matrix penalizing deviation from the baseline

[1] **Notation**: We denote by $\mathbb{R}_{\geq 0}$ the set of non-negative reals and $\partial\mathcal{A}$ the boundary of a set $\mathcal{A} \subset \mathbb{R}^n$. For a vector $\mathbf{x}$ and a matrix $\mathbf{X}$, we use $\|\mathbf{x}\|$ and $\|\mathbf{X}\|$ to denote the Euclidean norm and the spectral norm. We use $\text{vec}(\mathbf{X}) \in \mathbb{R}^{nm}$ to denote the vectorization of $\mathbf{X} \in \mathbb{R}^{n\times m}$, obtained by stacking its columns, and $\text{diag}(\mathbf{x}) \in \mathbb{R}^{n\times n}$ to denote a diagonal matrix whose diagonal is $\mathbf{x} \in \mathbb{R}^n$. We denote by $\nabla$ the gradient and $\mathcal{L}_f V = \nabla V \cdot f$ the Lie derivative of a differentiable function $V$ along a vector field $f$. 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 of class $\mathcal{K}$ and $\lim_{r\to\infty} \alpha(r) = \infty$.

controller. The term $\delta \geq 0$ is a slack variable that relaxes the CLF constraints to ensure the feasibility of the QP, controlled by the scaling factor $\lambda > 0$. The QP formulation in (6) modifies the baseline controller $\tilde{\underline{\mathbf{k}}}(\mathbf{x})$ online to ensure safety and stability via the CBF and CLF constraints.

### *B. Safety and Stability with Estimated System Dynamics and Barrier Function*

Our work focuses on enforcing safety and stability for the control-affine system (1) when the system dynamics $F(\mathbf{x})$ and the barrier function $h(\mathbf{x})$ are *unknown* and need to be estimated from data. We consider two scenarios, depending on whether probabilistic or worst-case error descriptions of the dynamics and barrier functions are available.

**Problem 1 (Safety and stability under Gaussian uncertainty).** Given an estimated distribution on the unknown system dynamics $\text{vec}(F(\mathbf{x})) \sim \mathcal{GP}(\text{vec}(\tilde{F}(\mathbf{x})), K_F(\mathbf{x}, \mathbf{x}'))$ 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 $\mathbf{x} \in \mathcal{X}$:

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

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

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

$$\|F(\mathbf{x}) - \tilde{F}(\mathbf{x})\| \leq e_F(\mathbf{x}), \ \forall \mathbf{x} \in \mathcal{X}, \tag{7}$$

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 $\mathbf{x} \in \mathcal{X}$,

$$|h(\mathbf{x}) - \tilde{h}(\mathbf{x})| \leq e_h(\mathbf{x}), \ \|\nabla h(\mathbf{x}) - \nabla \tilde{h}(\mathbf{x})\| \leq e_{\nabla h}(\mathbf{x}), \tag{8}$$

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

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

## IV. Probabilistic Safe Control

This section presents our solution to Problem 1. Inspired by the design (6) when the dynamics and the barrier function are known, we formulate the control synthesis problem via the following optimization problem:

$$\begin{aligned} &\min_{\underline{\mathbf{u}} \in \underline{\mathcal{U}}, \delta \in \mathbb{R}} \ \|L(\mathbf{x})^\top (\underline{\mathbf{u}} - \tilde{\underline{\mathbf{k}}}(\mathbf{x}))\|^2 + \lambda \delta^2, \\ &\text{s.t. } \mathbb{P}(\mathit{CLC}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta) \geq p, \quad \mathbb{P}(\mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0) \geq p. \end{aligned} \tag{9}$$

The uncertainty in $F$ and $h$ affects the linearity in $\underline{\mathbf{u}}$ of the CLC and CBC conditions in the constraints of (9), 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 $\mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}})$ and $\mathit{CLC}(\mathbf{x}, \underline{\mathbf{u}})$ in detail.

**Proposition IV.1 (Distribution for CBC).** *Assume $h$ is a CBF with a linear function $\alpha_h$, 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 $\mathit{vec}(F(\mathbf{x})) \sim \mathcal{GP}(\mathit{vec}(\tilde{F}(\mathbf{x})), K_F(\mathbf{x}, \mathbf{x}'))$, the mean and variance of $\mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}})$ satisfy*

$$\mathbb{E}[\mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}})] = \mathbb{E}[\mathbf{p}(\mathbf{x})]^\top \underline{\mathbf{u}} \tag{10a}$$

$$\mathit{Var}[\mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}})] = \underline{\mathbf{u}}^\top \mathit{Var}[\mathbf{p}(\mathbf{x})] \underline{\mathbf{u}}, \tag{10b}$$

*where*

$$\mathbf{p}(\mathbf{x}) := F^\top(\mathbf{x})[\nabla_{\mathbf{x}} h(\mathbf{x})] + \begin{bmatrix} a h(\mathbf{x}) & \mathbf{0}_m^\top \end{bmatrix}^\top \in \mathbb{R}^{m+1} \tag{11}$$

*and $\mathbb{E}[\mathbf{p}(\mathbf{x})]$, $\mathit{Var}[\mathbf{p}(\mathbf{x})]$ are computed in* (18)*.*

*Proof.* The control barrier condition can be written as:

$$\begin{aligned} \mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}}) &= [\nabla_{\mathbf{x}} h(\mathbf{x})]^\top f(\mathbf{x}) + [\nabla_{\mathbf{x}} h(\mathbf{x})]^\top g(\mathbf{x}) \mathbf{u} + a h(\mathbf{x}) \\ &= \Bigg[ \underbrace{[\nabla_{\mathbf{x}} h(\mathbf{x})]^\top F(\mathbf{x})}_{\mathbf{p}_1^\top} + \begin{bmatrix} a h(\mathbf{x}) & \mathbf{0}_m^\top \end{bmatrix} \Bigg] \underline{\mathbf{u}} \\ &= \mathbf{p}(\mathbf{x})^\top \underline{\mathbf{u}}. \end{aligned} \tag{12}$$

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. [22, 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 $\text{vec}(\mathbf{ABC}) = (\mathbf{C}^\top \otimes \mathbf{A})\text{vec}(\mathbf{B})$ for appropriately sized matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, we can write

$$\begin{aligned} \mathit{Var}(F(\mathbf{x})\underline{\mathbf{u}}) &= \mathit{Var}((\underline{\mathbf{u}}^\top \otimes \mathbf{I}_n)\text{vec}(F(\mathbf{x}))) \\ &= (\underline{\mathbf{u}}^\top \otimes \mathbf{I}_n) K_F(\mathbf{x}, \mathbf{x}) (\underline{\mathbf{u}} \otimes \mathbf{I}_n). \end{aligned} \tag{13}$$

We abbreviate $K_F := K_F(\mathbf{x}, \mathbf{x}')$ and $K_h := K_h(\mathbf{x}, \mathbf{x}')$ for conciseness in the rest of this section. The term $[\nabla_{\mathbf{x}} h(\mathbf{x})]^\top F(\mathbf{x}) \underline{\mathbf{u}}$ is an inner product of two independent GPs, $\nabla_{\mathbf{x}} h(\mathbf{x})$ and $F(\mathbf{x})\underline{\mathbf{u}}$. Thus, using [22, Lemma 5], (13), and that $\mathit{Cov}(\nabla_{\mathbf{x}} h(\mathbf{x}), F(\mathbf{x})\underline{\mathbf{u}}) = 0$, $\mathbf{p}_1^\top \underline{\mathbf{u}}$ corresponds to a distribution with mean and variance:

$$\begin{aligned} &\mathbb{E}[\mathbf{p}_1^\top \underline{\mathbf{u}}] = [\nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})]^\top \tilde{F}(\mathbf{x}) \underline{\mathbf{u}}, \\ &\mathit{Var}[\mathbf{p}_1^\top \underline{\mathbf{u}}] = [\nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})]^\top (\underline{\mathbf{u}}^\top \otimes \mathbf{I}_n) K_F \\ &\qquad (\underline{\mathbf{u}} \otimes \mathbf{I}_n) \nabla_{\mathbf{x}} \tilde{h}(\mathbf{x}) + \underline{\mathbf{u}}^\top \tilde{F}^\top(\mathbf{x}) \mathcal{H}_{\mathbf{x}, \mathbf{x}'} K_h \tilde{F}(\mathbf{x}) \underline{\mathbf{u}}. \end{aligned} \tag{14}$$

To factorize $\underline{\mathbf{u}}$ from the variance expression, we apply the property $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}$ for any appropriately sized matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$,

$$\begin{aligned} (\underline{\mathbf{u}} \otimes \mathbf{I}_n)[\nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})] &= (\underline{\mathbf{u}} \otimes \mathbf{I}_n)(1 \otimes [\nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})]) \\ &= \underline{\mathbf{u}} \otimes \nabla_{\mathbf{x}} \tilde{h}(\mathbf{x}) = (\mathbf{I}_{m+1} \otimes \nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})) \underline{\mathbf{u}}. \end{aligned} \tag{15}$$

By substituting (15) in (14), we can factorize out $\underline{\mathbf{u}}$ to get,

$$\begin{aligned} \mathit{Var}[\mathbf{p}_1] = (\mathbf{I}_{m+1} \otimes [\nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})]^\top) K_F (\mathbf{I}_{m+1} \otimes \nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})) \\ + \tilde{F}^\top(\mathbf{x}) \mathcal{H}_{\mathbf{x}, \mathbf{x}'} K_h \tilde{F}(\mathbf{x}). \end{aligned} \tag{16}$$

Next, we write $\mathit{Cov}(h(\mathbf{x}), \mathbf{p}_1^\top \underline{\mathbf{u}})$ using [22, Lemma 5] and $\mathit{Cov}(h(\mathbf{x}), F(\mathbf{x})\underline{\mathbf{u}}) = 0$,

$$\mathit{Cov}(h(\mathbf{x}), \mathbf{p}_1^\top \underline{\mathbf{u}}) = \mathit{Cov}(h(\mathbf{x}), \nabla_{\mathbf{x}} h(\mathbf{x})) \tilde{F}(\mathbf{x}) \underline{\mathbf{u}}$$

$$= [\nabla_{\mathbf{x}} K_h]^\top \tilde{F}(\mathbf{x})\underline{\mathbf{u}}$$
$$= \begin{bmatrix}[\nabla_{\mathbf{x}} K_h]^\top \tilde{f}(\mathbf{x}) & [\nabla_{\mathbf{x}} K_h]^\top \tilde{g}(\mathbf{x})\end{bmatrix}\underline{\mathbf{u}}. \quad (17)$$

We can now write mean and variance of $\mathbf{p}(\mathbf{x})$ explicitly by using (14), (16) and (17),

$$\begin{aligned}\mathbb{E}[\mathbf{p}(\mathbf{x})] &= [\nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})]^\top \tilde{F}(\mathbf{x}) + a[\tilde{h}(\mathbf{x}) \quad \mathbf{0}_m^\top]^\top \\ \textit{Var}[\mathbf{p}(\mathbf{x})] &= \tilde{F}^\top(\mathbf{x})\mathcal{H}_{\mathbf{x},\mathbf{x}'} K_h \tilde{F}(\mathbf{x}) \\ &+ (\mathbf{I}_{m+1} \otimes \nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})^\top) K_F (\mathbf{I}_{m+1} \otimes \nabla_{\mathbf{x}} \tilde{h}(\mathbf{x})) \quad (18) \\ &+ \begin{bmatrix} a^2 K_h + 2a[\nabla_{\mathbf{x}} K_h]^\top f(\mathbf{x}) & a[\nabla_{\mathbf{x}} K_h]^\top g(\mathbf{x}) \\ a g(\mathbf{x})^\top [\nabla_{\mathbf{x}} K_h] & \mathbf{0}_{m\times m} \end{bmatrix}\end{aligned}$$

The statement now follows by using (18) in (12). □

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

**Proposition IV.2 (Gaussian distribution for CLC).** *Given the distribution* $\textit{vec}(F(\mathbf{x})) \sim \mathcal{GP}(\textit{vec}(\tilde{F}(\mathbf{x})), K_F(\mathbf{x}, \mathbf{x}'))$, *the* $\textit{CLC}(\mathbf{x}, \underline{\mathbf{u}})$ *is Gaussian with mean and variance:*

$$\mathbb{E}[\textit{CLC}(\mathbf{x}, \underline{\mathbf{u}})] = \mathbb{E}[\mathbf{q}(\mathbf{x})]^\top \underline{\mathbf{u}} \quad (19a)$$
$$\textit{Var}[\textit{CLC}(\mathbf{x}, \underline{\mathbf{u}})] = \underline{\mathbf{u}}^\top \textit{Var}[\mathbf{q}(\mathbf{x})]\underline{\mathbf{u}}, \quad (19b)$$

*where*

$$\mathbf{q}(\mathbf{x}) := F^\top(\mathbf{x})[\nabla_{\mathbf{x}} V(\mathbf{x})] + [\alpha_V(V(\mathbf{x})) \quad \mathbf{0}_m^\top]^\top \in \mathbb{R}^{m+1} \quad (20)$$

*and* $\mathbb{E}[\mathbf{q}(\mathbf{x})]$, $\textit{Var}[\mathbf{q}(\mathbf{x})]$ *are computed in* (21).

*Proof.* We can write the control Lyapunov condition as

$$\textit{CLC}(\mathbf{x}, \underline{\mathbf{u}}) = [\nabla_{\mathbf{x}} V(\mathbf{x})]^\top F(\mathbf{x})\underline{\mathbf{u}} + \alpha_V(V(\mathbf{x})) = \mathbf{q}^\top(\mathbf{x})\underline{\mathbf{u}}.$$

We can use the Kronecker product property $\text{vec}(\mathbf{ABC}) = (\mathbf{C}^\top \otimes \mathbf{A})\text{vec}(\mathbf{B})$ to rewrite first term in $\mathbf{q}(\mathbf{x})$ as:

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

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

$$\begin{aligned}\mathbb{E}[\mathbf{q}(\mathbf{x})] &= \tilde{F}^\top(\mathbf{x})[\nabla_{\mathbf{x}} V(\mathbf{x})] + [\alpha_V(V(\mathbf{x})) \quad \mathbf{0}_m^\top]^\top \quad (21) \\ \textit{Var}[\mathbf{q}(\mathbf{x})] &= (\mathbf{I}_{m+1} \otimes [\nabla_{\mathbf{x}} V(\mathbf{x})]^\top) K_F (\mathbf{I}_{m+1} \otimes [\nabla_{\mathbf{x}} V(\mathbf{x})]).\end{aligned}$$

The result follows from plugging (21) into $\textit{CLC}(\mathbf{x}, \underline{\mathbf{u}})$. □

Using the distributions of $\textit{CBC}(\mathbf{x}, \underline{\mathbf{u}})$ and $\textit{CLC}(\mathbf{x}, \underline{\mathbf{u}})$, problem (9) can be formulated as a convex SOCP as follows.

**Proposition IV.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* (9) *can be formulated as the following second-order cone program:*

$$\begin{aligned}\min_{\underline{\mathbf{u}} \in \underline{\mathcal{U}}, \delta \in \mathbb{R}, l \in \mathbb{R}} \quad & l \\ \text{s.t. } \delta - \mathbb{E}[\mathbf{q}(\mathbf{x})]^\top \underline{\mathbf{u}} &\geq c(p)\sqrt{\underline{\mathbf{u}}^\top \textit{Var}[\mathbf{q}(\mathbf{x})]\underline{\mathbf{u}}}, \\ \mathbb{E}[\mathbf{p}(\mathbf{x})]^\top \underline{\mathbf{u}} &\geq c(p)\sqrt{\underline{\mathbf{u}}^\top \textit{Var}[\mathbf{p}(\mathbf{x})]\underline{\mathbf{u}}}, \quad (22) \\ l + 1 &\geq \left\| \begin{bmatrix} 2L(\mathbf{x})^\top(\underline{\mathbf{u}} - \tilde{\underline{\mathbf{k}}}(\mathbf{x})) \\ 2\sqrt{\lambda}\delta \\ l - 1 \end{bmatrix} \right\|,\end{aligned}$$

*where* $\mathbf{p}(\mathbf{x})$ *and* $\mathbf{q}(\mathbf{x})$ *are defined in* (11) *and* (20).

*Proof.* To deal with the probabilistic constraints in (9), we employ Cantelli's inequality [16]. For any scalar $\gamma \geq 0$,

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

Given this inequality, and since we want $\mathbb{P}(\textit{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0) \geq p$, we choose $\gamma = \mathbb{E}[\textit{CBC}(\mathbf{x}, \underline{\mathbf{u}})]$ and require the lower bound to be greater than or equal to $p$, i.e.,

$$1 - \frac{\textit{Var}[\textit{CBC}(\mathbf{x}, \underline{\mathbf{u}})]}{\textit{Var}[\textit{CBC}(\mathbf{x}, \underline{\mathbf{u}})] + \gamma^2} \geq p.$$

The equation can be rearranged into

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

which corresponds to the safety constraint in (22).

Next, we show that this constraint is a valid second-order cone (SOC) constraint. By (10), given that $\tilde{h}$, $\nabla\tilde{h}$ and $\tilde{F}$ are known and deterministic, the expectation $\mathbb{E}[\textit{CBC}(\mathbf{x}, \underline{\mathbf{u}})] = \mathbb{E}[\mathbf{p}(\mathbf{x})]^\top \underline{\mathbf{u}}$ is affine in $\underline{\mathbf{u}}$. Since $\textit{Var}[\mathbf{p}(\mathbf{x})]$ is positive semi-definite, we can write

$$\sqrt{\textit{Var}[\textit{CBC}(\mathbf{x}, \underline{\mathbf{u}})]} = \sqrt{\underline{\mathbf{u}}^\top \textit{Var}[\mathbf{p}(\mathbf{x})]\underline{\mathbf{u}}} = \|\mathbf{D}(\mathbf{x})\underline{\mathbf{u}}\| \quad (23)$$

where $\mathbf{D}(\mathbf{x})^\top \mathbf{D}(\mathbf{x}) = \textit{Var}[\mathbf{p}(\mathbf{x})]$. Acccording to [30], the safety constraint in (22) is a valid SOC constraint.

For stability, the CLC condition can be constructed using a similar approach with Cantelli's inequality, resulting in (22). By (19), we know that the expectation is affine in $\underline{\mathbf{u}}$ and the variance is quadratic in terms of $\underline{\mathbf{u}}$, similar to (23). This shows that the CLC condition is also a valid SOC constraint.

Our last step is to show that the minimization of the objective function can be reformulated with a linear objective and an additional SOC constraint, resulting in the standard SOCP in (22). We introduce a new variable $l$ so that the problem in (9) is equivalent to

$$\begin{aligned}\min_{\underline{\mathbf{u}} \in \underline{\mathcal{U}}, \delta \in \mathbb{R}, l \in \mathbb{R}} \quad & l \\ \text{s.t. } \mathbb{P}(\textit{CLC}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta) \geq p, \quad & \mathbb{P}(\textit{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0) \geq p, \\ \|L(\mathbf{x})^\top(\underline{\mathbf{u}} - \tilde{\underline{\mathbf{k}}}(\mathbf{x}))\|^2 + \lambda\delta^2 &\leq l. \quad (24)\end{aligned}$$

The last constraint in (24) corresponds to a rotated second-order cone, $\mathcal{Q}^n_{rot} := \{(\mathbf{x}_r, y_r, z_r) \in \mathbb{R}^{n+2} \,|\, \|\mathbf{x}_r\|^2 \leq y_r z_r, y_r \geq 0, z_r \geq 0\}$, which can be converted into a standard SOC constraint [30],

$$\left\| \begin{bmatrix} 2\mathbf{x}_r \\ y_r - z_r \end{bmatrix} \right\| \leq y_r + z_r.$$

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 $\sqrt{\|2L(\mathbf{x})^\top(\underline{\mathbf{u}} - \underline{\tilde{\mathbf{k}}}(\mathbf{x}))\|^2 + (2\sqrt{\lambda}\delta)^2 + (l-1)^2} \leq l+1$, which is equivalent to the third constraint in (22). □

*Remark* IV.4 (**Risk-tolerance** $p$). When $p = 0$, the probabilistic CLF-CBF-SOCP (22) reduces to the original CLF-CBF-QP (6).

## V. Robust Safe Control

In this section, we develop a solution to Problem 2. Let $\tilde{F}$ 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 associated error bounds. For convenience, for each $\mathbf{x} \in \mathcal{X}$, we denote $D_F(\mathbf{x}) := F(\mathbf{x}) - \tilde{F}(\mathbf{x})$, $d_h(\mathbf{x}) := h(\mathbf{x}) - \tilde{h}(\mathbf{x})$ and $\mathbf{d}_{\nabla h}(\mathbf{x}) := \nabla h(\mathbf{x}) - \nabla\tilde{h}(\mathbf{x})$. By (7) and (8), we have

$$\|D_F(\mathbf{x})\| \leq e_F(\mathbf{x}),\ |d_h(\mathbf{x})| \leq e_h(\mathbf{x}),\ \|\mathbf{d}_{\nabla h}(\mathbf{x})\| \leq e_{\nabla h}(\mathbf{x}). \tag{25}$$

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

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

Let $\tilde{\mathbf{p}}(\mathbf{x}) := \tilde{F}^\top(\mathbf{x})\nabla\tilde{h}(\mathbf{x})$. We group the error term in the expression for $\mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}})$ in the variable $d_{\mathit{CBC}}(\mathbf{x}, \underline{\mathbf{u}}) := \mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}}) - \tilde{\mathbf{p}}(\mathbf{x})^\top\underline{\mathbf{u}}$. Thus, $\mathit{CBC}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0$ is satisfied if

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

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

$$\max_{D_F} \mathit{CLC}(\mathbf{x}, \underline{\mathbf{u}}) = \tilde{\mathbf{q}}(\mathbf{x})^\top\underline{\mathbf{u}} + \max_{D_F} d_{\mathit{CLC}}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta. \tag{26}$$

This leads us to the following robust reformulation of the original control synthesis problem in (6),

$$\begin{aligned}
\min_{\underline{\mathbf{u}}\in\underline{\mathcal{U}}, \delta\in\mathbb{R}, l\in\mathbb{R}}\ & l \\
\text{s.t. } & \tilde{\mathbf{q}}(\mathbf{x})^\top\underline{\mathbf{u}} + \max_{D_F}\ d_{\mathit{CLC}}(\mathbf{x}, \underline{\mathbf{u}}) \leq \delta \\
& \tilde{\mathbf{p}}(\mathbf{x})^\top\underline{\mathbf{u}} + \min_{D_F, d_h, \mathbf{d}_{\nabla h}} d_{\mathit{CBC}}(\mathbf{x}, \underline{\mathbf{u}}) \geq 0 \\
& \left\| \begin{bmatrix} 2L(\mathbf{x})^\top(\underline{\mathbf{u}} - \underline{\tilde{\mathbf{k}}}(\mathbf{x})) \\ 2\sqrt{\lambda}\delta \\ l-1 \end{bmatrix} \right\| \leq l+1.
\end{aligned} \tag{27}$$

Note that we used the same approach as in the proof of Proposition IV.3 to reformulate the original quadratic objective with a linear objective plus a SOC constraint. The difficulty in solving (27) arises from the complexity of the constraints. The next result considers a restriction of the feasible set that gives rise to a convex SOCP formulation.

**Proposition V.1** (**Robust-CLF-CBF-SOCP**). *Let $\tilde{F}$, $\tilde{h}$, $\nabla\tilde{h}$ denote estimates about the system dynamics and barrier function, with error bounds described by* (25)*. Consider the following optimization problem*

$$\begin{aligned}
\min_{\underline{\mathbf{u}}\in\underline{\mathcal{U}}, \delta\in\mathbb{R}, p\in\mathbb{R}, q\in\mathbb{R}, l\in\mathbb{R}}\ & l \\
\text{s.t. } & \delta - \tilde{\mathbf{q}}(\mathbf{x})^\top\underline{\mathbf{u}} \geq e_F(\mathbf{x})\|\nabla V(\mathbf{x})\|\|\underline{\mathbf{u}}\|, \\
& p \geq e_{\nabla h}(\mathbf{x})\|\tilde{F}(\mathbf{x})\underline{\mathbf{u}}\|, \\
& q \geq \Big(e_F(\mathbf{x})\|\nabla\tilde{h}(\mathbf{x})\| + e_{\nabla h}(\mathbf{x})e_F(\mathbf{x})\Big)\|\underline{\mathbf{u}}\|, \\
& [\nabla\tilde{h}(\mathbf{x})]^\top\tilde{F}(\mathbf{x})\underline{\mathbf{u}} + \alpha_h(\tilde{h}(\mathbf{x}) - e_h(\mathbf{x})) \geq p + q, \\
& l+1 \geq \left\| \begin{bmatrix} 2L(\mathbf{x})^\top(\underline{\mathbf{u}} - \underline{\tilde{\mathbf{k}}}(\mathbf{x})) \\ 2\sqrt{\lambda}\delta \\ l-1 \end{bmatrix} \right\|.
\end{aligned} \tag{28}$$

*Then, problem* (28) *is a SOCP and its feasible set is included in the feasible set of* (27)*,*

*Proof.* The stability constraint directly follows from the fact:

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

For the safety constraint, note that

$$\begin{aligned}
&\min_{D_F, d_h, \mathbf{d}_{\nabla h}} d_{\mathit{CBC}}(\mathbf{x}, \underline{\mathbf{u}}) \\
&= \min_{D_F, \mathbf{d}_{\nabla h}} \Big(\mathbf{d}_{\nabla h}^\top(\mathbf{x})\tilde{F}(\mathbf{x})\underline{\mathbf{u}} + [\nabla\tilde{h}(\mathbf{x})]^\top D_F(\mathbf{x})\underline{\mathbf{u}} + \\
&\qquad \mathbf{d}_{\nabla h}^\top(\mathbf{x})D_F(\mathbf{x})\underline{\mathbf{u}}\Big) + \min_{d_h} \alpha_h(\tilde{h}(\mathbf{x}) + d_h(\mathbf{x})).
\end{aligned} \tag{29}$$

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

$$\min_{|d_h(\mathbf{x})| \leq e_h(\mathbf{x})} \alpha_h(\tilde{h}(\mathbf{x}) + d_h(\mathbf{x})) = \alpha_h(\tilde{h}(\mathbf{x}) - e_h(\mathbf{x})). \tag{30}$$

Applying the Cauchy-Schwarz inequality on each term,

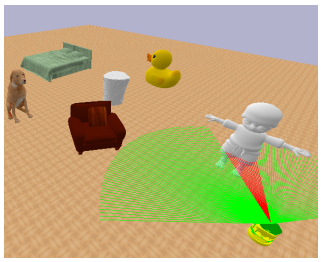
$$\begin{aligned}
&\min_{D_F, d_h, \mathbf{d}_{\nabla h}} d_{\mathit{CBC}}(\mathbf{x}, \underline{\mathbf{u}}) \geq -\|\mathbf{d}_{\nabla h}\|\|\tilde{F}(\mathbf{x})\underline{\mathbf{u}}\| \\
&\qquad - \|\nabla\tilde{h}(\mathbf{x})\|\|D_F(\mathbf{x})\underline{\mathbf{u}}\| - \|\mathbf{d}_{\nabla h}(\mathbf{x})\|\|D_F(\mathbf{x})\underline{\mathbf{u}}\|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

Note that in the second inequality we have considered the minimization of each term independently, and hence the resulting lower bound might not be tight in general. Thus, we rewrite the safety constraint as

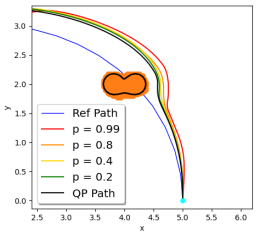
$$\begin{aligned}
&e_{\nabla h}(\mathbf{x})\|\tilde{F}(\mathbf{x})\underline{\mathbf{u}}\| + (e_F(\mathbf{x})\|\nabla\tilde{h}(\mathbf{x})\| + e_{\nabla h}(\mathbf{x})e_F(\mathbf{x}))\|\underline{\mathbf{u}}\| \\
&\leq [\nabla\tilde{h}(\mathbf{x})]^\top\tilde{F}(\mathbf{x})\underline{\mathbf{u}} + \alpha_h(\tilde{h}(\mathbf{x}) - e_h(\mathbf{x})).
\end{aligned} \tag{31}$$

Finally, from Lemma A.1, the second, third, and fourth constraints in (28) combined together are equivalent to (31). □

*Remark* V.2 (**Connection with the literature**). The result in Proposition V.1 recovers [28, Proposition 2] for the case when there is no uncertainty in the system dynamics ($e_F = 0$). If there are no errors in both the dynamics and the barrier function ($e_F = e_h = e_{\nabla h} = 0$), then the robust-CLF-CBF-SOCP (28) reduces to the original CLF-CBF-QP (6).



(a) Pybullet Simulator




(b) Probabilistic Trajectory

Fig. 1: (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.

## VI. Evaluation

In this section, we present an approach to estimate the unknown dynamics of a mobile robot, and construct CBF constraints online. Then, we evaluate our safe control synthesis using the estimated robot dynamics and CBFs in autonomous navigation tasks in 10 simulated environments, containing obstacles a priori unknown to the robot.

### A. System Dynamics Estimation

We consider a Turtlebot robot simulated in the PyBullet simulator [31] (see Fig. 1a). We first present a learning approach to model the unknown dynamics of the TurtleBot using training data collected from the PyBullet simulator. The robot state and input are $\mathbf{x} := [x, y, \mu]^\top \in \mathbb{R}^2 \times [-\pi, \pi)$ and $\underline{\mathbf{u}} := [1, v, \omega]^\top \in \{1\} \times \mathbb{R}^2$, respectively. We collect a dataset $\mathcal{D} = \{t^{(i)}_{0:N}, \mathbf{x}^{(i)}_{0:N}, \underline{\mathbf{u}}^{(i)}_{0:N}\}_{i=1}^{D}$ of $D = 40000$ state sequences $\mathbf{x}^{(i)}_{0:N}$ obtained by applying random control inputs $\underline{\mathbf{u}}^{(i)}_{0:N}$ to the robot with initial condition $\mathbf{x}^{(i)}_0$ at time intervals of $\tau = 0.02$ seconds. For each trajectory $i$, a constant control input is applied for $N = 5$ time steps.

We employ a neural ODE network [32] to approximate the unknown robot dynamics $F$ with a neural network $F_{\boldsymbol{\theta}}$ based on the dataset $\mathcal{D}$. A forward pass through the ODE network is obtained using an ODE solver:

$$\{\tilde{\mathbf{x}}^i_1, \tilde{\mathbf{x}}^i_2, \cdots, \tilde{\mathbf{x}}^i_N\} = \text{ODESolve}(\mathbf{x}^i_0, F_{\boldsymbol{\theta}}(\cdot)\underline{\mathbf{u}}^i, t^i_1, \cdots, t^i_N).$$

We use a loss function,

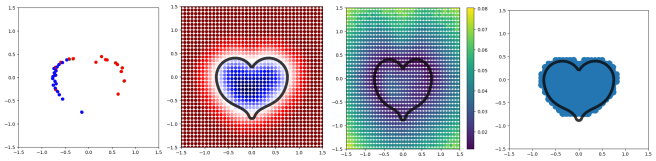
$$\begin{aligned} &\min_{\boldsymbol{\theta}} \sum_{i=1}^{D}\sum_{j=1}^{N} \ell(\mathbf{x}^{(i)}_j, \tilde{\mathbf{x}}^{(i)}_j), \\ &\text{s.t. } \dot{\tilde{\mathbf{x}}}^{(i)}(t) = F_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}^{(i)}(t))\underline{\mathbf{u}}^{(i)}(t), \quad \tilde{\mathbf{x}}^{(i)}(j\tau) = \tilde{\mathbf{x}}^{(i)}_j, \\ &\quad \underline{\mathbf{u}}^{(i)}(t) \equiv \underline{\mathbf{u}}^{(i)}_j \quad \text{for } t \in [j\tau, (j+1)\tau), \end{aligned} \tag{32}$$

where $\ell(\mathbf{x}, \tilde{\mathbf{x}}) = \|[x, y, \cos\mu, \sin\mu]^\top - [\tilde{x}, \tilde{y}, \cos\tilde{\mu}, \sin\tilde{\mu}]^\top\|^2$. To update the weights $\boldsymbol{\theta}$, the gradient of the loss function is back-propagated by solving another ODE with adjoint states backwards in time. Please refer to [32] for details.

Gal and Ghahramani [33] showed that introducing dropout layers in a neural network is approximately equivalent to performing deep Gaussian Process regression. We use a 6-layer fully-connected neural network with $\tanh$ activations and 800 neurons in each layer to model $F_{\boldsymbol{\theta}}$, and apply dropout to each hidden layer with rate 0.05. Given a query state $\mathbf{x} \in \mathcal{X}$, Monte-Carlo estimates of the predictive mean $\tilde{F}_{\boldsymbol{\theta}}(\mathbf{x})$ and element-wise standard deviation $\tilde{\Sigma}(\mathbf{x})$ of the dynamics are obtained with $T = 100$ stochastic forward passes through the dropout neural network model. We use $\tilde{F}_{\boldsymbol{\theta}}(\mathbf{x})$ for the mean of system dynamics and $K_F(\mathbf{x}, \mathbf{x}) = \text{diag}(\text{vec}(\tilde{\Sigma}(\mathbf{x}))^2)$ for the variance of the dynamics. To obtain worst-case error bounds $e_F(\mathbf{x})$, we set $e_F(\mathbf{x}) = \|3.89\tilde{\Sigma}(\mathbf{x})\|$ (99.99% *confidence*).

TABLE I: Empirical SDF estimation error $\mathcal{E}$ in (34) and dropout-network SDF estimation error averaged across 8 object instances under different LiDAR measurement noise standard deviation $\sigma$.

| LiDAR Noise $\sigma$ | SDF Empirical Error | SDF Dropout Error |
|---|---|---|
| 0.01 | 0.0173 | 0.0132 |
| 0.02 | 0.0288 | 0.0184 |
| 0.05 | 0.0463 | 0.0242 |



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

Fig. 2: 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 $\mathbb{P}(\tilde{\varphi} \le 0) = 0.95$.

### B. 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. 2b) 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

$$\varphi_i(\mathbf{y}) := \begin{cases} -d(\mathbf{y}, \partial\mathcal{O}_i), & \mathbf{y} \in \mathcal{O}_i, \\ d(\mathbf{y}, \partial\mathcal{O}_i), & \mathbf{y} \notin \mathcal{O}_i, \end{cases} \tag{33}$$

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) [28, Sec. IV] to estimate an SDF $\varphi_i$ 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. To verify the accuracy of our online SDF estimation method, the TurtleBot is tasked to move along a circle of radius 2 while the object is placed at the center. In Table I, we measure the empirical SDF approximation error:

$$\mathcal{E}_i = \frac{1}{m}\sum_{j=1}^{m} |\hat{\varphi}_i(\mathbf{y}_j)|, \tag{34}$$

TABLE II: Success rate of the navigation tasks in 100 realizations (10 realizations for each of the 10 different environments) using the Probabilistic CLF-CBF-SOCP, Robust CLF-CBF-SOCP, and the original CLF-CBF-QP frameworks for different LiDAR measurement noise levels $\sigma$.

| LiDAR Noise $\sigma$ | QP Success Rate | Probabilistic Success Rate | | | Robust Success Rate |
|---|---|---|---|---|---|
| | | $p = 0.2$ | $p = 0.4$ | $p = 0.8$ | |
| 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 |

TABLE III: 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, 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 |

where $\{\mathbf{y}_j\}_{j=1}^m$ are $m = 500$ points uniformly sampled on the surface of the object instances. In Fig. 2, we show the results of SDF estimation with measurement noise standard deviation $\sigma = 0.01$.

Since we deal with system dynamics with relative degree one, one can verify [34] that the SDF is a valid CBF. 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. IV, 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. V 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 (6), (22), (28) for safe control synthesis.

### C. Safe Navigation

Our main experiments demonstrate safe trajectory tracking using the proposed probabilistic (22) and robust (28) CLF-CBF-SOCP formulations, utilizing the dynamics estimates from Sec. VI.A and the online CBF estimates from Sec. VI.B. We employ the approach in [28, Sec. VI] to construct a valid CLF $V(\mathbf{x})$ for path following. To avoid low velocity, we set $L(\mathbf{x}) = \text{diag}([0, 10, 3])$ and $\underline{\tilde{\mathbf{k}}}(\mathbf{x}) = [1, v_{\max}, 0]^\top$ where $v_{\max} = 0.65$ is the maximum linear velocity for the TurtleBot. The remaining parameters were $\lambda = 1000$, $\alpha_V(V(\mathbf{x})) = 2V(\mathbf{x})$, and $\alpha_h(h_i(\mathbf{x})) = h_i(\mathbf{x})$.

To emphasize the importance of accounting for estimation errors, we also implement the original CLF-CBF-QP controller (6), which assumes the estimated barrier functions and system dynamics are accurate (i.e., uses the mean values from the dropout-network estimation as the true values).

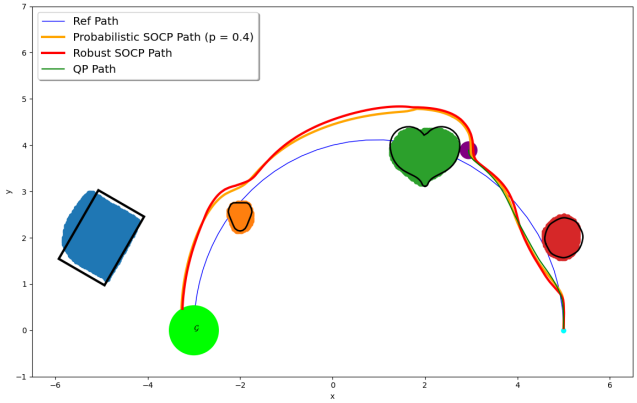



Fig. 3: Simulation results in environment 8 where the probabilistic ($p = 0.4$) and robust CLF-CBF-SOCP controllers succeeds but the CLF-CBF-QP one fails. The ground-truth obstacle surfaces are shown as black curves. The estimated obstacles, obtained after the whole path is traversed by the robust CLF-CBF-SOCP controller are shown in different colors (red, green, orange, blue). The reference path is shown in blue. The trajectories generated by the probabilistic and robust CLF-CBF-SOCP controllers are shown in orange and red, respectively, while the CLF-CBF-QP trajectory is shown in green. The starting point is cyan and the goal region is a light-green circle. The robot is shown as a purple circle when crashing into an obstacle using the CLF-CBF-QP controller.

In Table II, we report the success rate of the navigation task using the proposed formulations and the original QP framework under different measurement noises. As the noise increases, the success rate of the CLF-CBF-QP controller decreases rapidly, while the success rate of the probabilistic framework with high $p$ and the robust framework stays high.

In Fig. 3, we show one realization in environment 8 (with $\sigma = 0.01$), where the CLF-CBF-QP controller fails to avoid obstacles because it does not consider the errors in $CBC(\mathbf{x}, \underline{\mathbf{u}})$, while the proposed frameworks guarantee safety. When there is an obstacle near or on the reference path, the robot controlled by the robust SOCP controller stays furthest away, while the probabilistic SOCP controller also guarantees the robot stays further away from the obstacles than the robot controlled by the CLF-CBF-QP controller.

In Table III, we show quantitative results using the Fréchet distance [28, Sec. VI] as the metric to measure the trajectory similarities. The distance value is computed by averaging the successful realizations in each environment, and the LiDAR noise is set to be $\sigma = 0.02$ in this set of experiments. We see that the robust CLF-CBF-SOCP controller is most conservative as it has the largest Fréchet distance values while the probabilistic CLF-CBF-SOCP controller is less conservative if we set the user-specified risk tolerance $p = 0.8$. By lowering the risk tolerance value ($p = 0.2/0.4$), the robot with the probabilistic controller follows the reference path better while facing a higher risk of collision. The trajectory generated by the CLF-CBF-QP controller has the smallest Fréchet distance values, but fails in several environments.

In Fig. 1b, we report one realization of a region in an environment using the probabilistic controllers with different $p$ values. Observe that larger $p$ values correspond to higher probability of being safe for the robot. Finally, to demonstrate the efficiency of the proposed SOCP framework, we compare

the average time needed for solving the QP, probabilistic SOCP, and robust SOCP formulations per control synthesis along the navigation task. All optimization problems (QP and SOCP) are solved using the Embedded Conic Solver in CVXPY [35] with an Intel i7 9700K CPU. The time needed for solving one QP instance is $0.00863s$ while the times needed for solving the proposed probabilistic and robust SOCPs are $0.0109s$ and $0.0122s$. As expected, our SOCP formulations require slightly more time than the original QP but are still suitable for online robot navigation.

## VII. Conclusion

We considered the problem of enforcing safety and stability of unknown robot systems operating in unknown environments. We showed that accounting for either Gaussian or worst-case error bounds in the system dynamics and safety constraints leads to a novel CLF-CBF-SOCP formulation for control synthesis. We validated our approach in autonomous navigation tasks, simulating a ground robot in several unknown environments. Future work will consider deploying our formulations on real robots and formulating the problem with other probabilistic descriptions of uncertainty.

## References


[1] L. Lamport, "Proving the correctness of multiprocess programs," *IEEE Transactions on Software Engineering*, vol. SE-3, no. 2, pp. 125–143, 1977.

[2] B. Alpern and F. B. Schneider, "Defining liveness," *Information Processing Letters.*, vol. 21, pp. 181–185, 1985.

[3] Z. Artstein, "Stabilization with relaxed controls," *Nonlinear Analysis-theory Methods & Applications*, vol. 7, pp. 1163–1173, 1983.

[4] E. D. Sontag, "A 'universal' construction of Artstein's theorem on nonlinear stabilization," *Systems & Control Letters*, vol. 13, no. 2, pp. 117–123, 1989.

[5] S. Prajna, "Barrier certificates for nonlinear model validation," in *Conference on Decision and Control*, pp. 2884–2889, 2003.

[6] S. Prajna and A. Jadbabaie, "Safety verification of hybrid systems using barrier certificates," in *Hybrid Systems: Computation and Control*, pp. 477–492, Springer Berlin Heidelberg, 2004.

[7] P. Wieland and F. Allgöwer, "Constructive safety using control barrier functions," in *IFAC Proceedings Volumes*, pp. 462–467, 2007.

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

[9] A. Ames, X. Xu, J. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2016.

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

[11] A. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *European Control Conference*, pp. 3420–3431, 2019.

[12] L. Wang, A. D. Ames, and M. Egerstedt, "Safe certificate-based maneuvers for teams of quadrotors using differential flatness," *IEEE International Conference on Robotics and Automation*, 2017.

[13] Q. Nguyen, A. Hereid, J. W. Grizzle, A. D. Ames, and K. Sreenath, "3d dynamic walking on stepping stones with control barrier functions," in *IEEE Conference on Decision and Control*, pp. 827–834, 2016.

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

[15] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[16] F. P. Cantelli, "Sui confini della probabilità," *Atti del Congresso Internazionale dei Matematici*, vol. 6, pp. 47–60, 1929.

[17] M. Jankovic, "Robust control barrier functions for constrained stabilization of nonlinear systems," *Automatica*, vol. 96, pp. 359–367, 2018.

[18] Y. Emam, P. Glotfelter, and M. Egerstedt, "Robust barrier functions for a fully autonomous, remotely accessible swarm-robotics testbed," in *IEEE Conference on Decision and Control*, pp. 3984–3990, 2019.

[19] A. Clark, "Control barrier functions for complete and incomplete information stochastic systems," in *American Control Conference*, pp. 2928–2935, 2019.

[20] Q. Nguyen and K. Sreenath, "Robust safety-critical control for dynamic robotics," *IEEE Transactions on Automatic Control*, 2021.

[21] M. Ahmadi, X. Xiong, and A. D. Ames, "Risk-averse control via CVaR barrier functions: Application to bipedal robot locomotion," *IEEE Control Systems Letters*, vol. 6, pp. 878–883, 2022.

[22] V. Dhiman*, M. J. Khojasteh*, M. Franceschetti, and N. Atanasov, "Control barriers in bayesian learning of system dynamics," *IEEE Transactions on Automatic Control*, 2021.

[23] M. Z. Romdlony and B. Jayawardhana, "On the new notion of input-to-state safety," in *IEEE Conference on Decision and Control*, pp. 6403–6409, 2016.

[24] S. Kolathaya and A. D. Ames, "Input-to-state safety with control barrier functions," *IEEE Control Systems Letters*, vol. 3, no. 1, pp. 108–113, 2019.

[25] A. Alan, A. J. Taylor, C. R. He, G. Orosz, and A. D. Ames, "Safe controller synthesis with tunable input-to-state safe control barrier functions," *IEEE Control Systems Letters*, vol. 6, pp. 908–913, 2022.

[26] R. K. Cosner, A. W. Singletary, A. J. Taylor, T. G. Molnar, K. L. Bouman, and A. D. Ames, "Measurement-robust control barrier functions: Certainty in safety with uncertainty in state," *arXiv*, vol. abs/2104.14030, 2021.

[27] M. Srinivasan, A. Dabholkar, S. Coogan, and P. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 7139–7145, 2020.

[28] K. Long, C. Qian, J. Cortés, and N. Atanasov, "Learning barrier functions with memory for robust safe navigation," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4931–4938, 2021.

[29] T. T. Zhang, S. Tu, N. M. Boffi, J.-J. E. Slotine, and N. Matni, "Adversarially robust stability certificates can be sample-efficient," *arXiv preprint arXiv:2112.10690*, 2021.

[30] F. Alizadeh and D. Goldfarb, "Second-order cone programming," *Mathematical programming*, vol. 95, no. 1, pp. 3–51, 2003.

[31] E. Coumans and Y. Bai, "PyBullet, a Python module for physics simulation for games, robotics and machine learning." http://pybullet.org, 2016.

[32] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," *Advances in Neural Information Processing Systems*, 2018.

[33] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *International Conference on Machine Learning*, vol. 48, pp. 1050–1059, 2016.

[34] Y. Chen, M. Jankovic, M. A. Santillo, and A. Ames, "Backup control barrier functions: Formulation and comparative study," *arXiv*, vol. abs/2104.11332, 2021.

[35] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, 2016.


## Appendix A. Norm Sum as SOC Constraints

**Lemma A.1.** *The convex problem:*

$$\begin{aligned} \min_{\mathbf{z}\in\mathbb{R}^n} \quad & \boldsymbol{v}^\top \mathbf{z} \\ \text{s.t.} \quad & \|\mathbf{A}\mathbf{z}-\boldsymbol{a}\| + \|\mathbf{B}\mathbf{z}-\boldsymbol{b}\| \leq \boldsymbol{c}^\top \mathbf{z} \end{aligned}$$

*is equivalent to the second-order cone program:*

$$\begin{aligned} \min_{\mathbf{z}\in\mathbb{R}^n, p\in\mathbb{R}, q\in\mathbb{R}} \quad & \boldsymbol{v}^\top \mathbf{z} \\ \text{s.t.} \quad & \|\mathbf{A}\mathbf{z}-\boldsymbol{a}\| \leq p \\ & \|\mathbf{B}\mathbf{z}-\boldsymbol{b}\| \leq q \\ & p+q \leq \boldsymbol{c}^\top \mathbf{z}. \end{aligned}$$