Gaussian Control Barrier Functions: Safe Learning and Control

Mouhyemen Khan and Abhijit Chatterjee

***Abstract*— Safety is a critical component in today's autonomous and robotic systems. Many modern controllers endowed with notions of guaranteed safety properties rely on accurate mathematical models of these nonlinear dynamical systems. However, model uncertainty is always a persistent challenge weakening theoretical guarantees and compromising safety. For safety-critical systems, this is an even bigger challenge. Typically, safety is ensured by constraining the system states within a safe constraint set defined a priori by relying on the model of the system. A popular approach is to use Control Barrier Functions (CBFs) that encode safety using a smooth function. However, CBFs fail in the presence of model uncertainties. Moreover, an inaccurate model can either lead to incorrect notions of safety or worse, incur system critical failures. Addressing these drawbacks, we present a novel safety formulation that leverages properties of CBFs and positive definite kernels to design *Gaussian CBFs*. The underlying kernels are updated online by learning the unmodeled dynamics using Gaussian Processes (GPs). While CBFs guarantee forward invariance, the hyperparameters estimated using GPs update the kernel online and thereby adjust the relative notion of safety. We demonstrate our proposed technique on a safety-critical quadrotor on $SO(3)$ in the presence of model uncertainty in simulation. With the kernel update performed online, safety is preserved for the system.**

I. INTRODUCTION

Ensuring safety of today's autonomous systems such as self-driving vehicles and aerial delivery units is of prime importance. To this effect, Control Barrier Functions (CBFs) extended with quadratic programs (QP) have demonstrably proven safe control of safety-critical systems such as bipedal robots and quadrotors [1], [2]. A key limitation with CBFs is their reliance on having an accurate mathematical model of the dynamical system. This is a challenging problem since model uncertainties are pervasive in dynamical systems and can be very difficult to accurately model. It is clear that estimating and incorporating the unmodeled dynamics of systems in CBF design is paramount to ensuring safety. In this paper, we propose a novel approach for CBF construction that uses symmetric positive definite kernels called *Gaussian Control Barrier Functions*. This allows learning from the data and easily adapting the relative notion of safety without affecting the safety constrained set determined a priori.

The notion of barrier certificates is key to using CBFs for achieving safety control [3] [4]. By encoding the safety of the system as barrier certificates (or safe sets) with the aid of a smooth function, CBFs can be combined with QP to achieve safety constrained control [5]. CBF-QP based control has been demonstrated for a single quadrotor case [6], [7] and swarm settings [2], [8]. All these approaches make the assumption that the model is not subject to any disturbances or uncertainties.

Learning based approaches for ensuring safety has been investigated before. Learning the region of attraction for an uncertain nonlinear system is demonstrated in [9]. They use non-parametric bayesian formulations such as Gaussian processes (GPs) and Bayesian optimization (BO) to estimate and expand the safe set. GPs are flexible since they learn the model structure and estimate any hyperparameters from the data itself [10]. Thus, GPs are very powerful in capturing higher order nonlinearities with high prediction accuracy, e.g., in robot tracking and control [11]. BO was also used in systems such as quadruped, snake, and quadrotor for improving system performance while ensuring safety [12], [13], [14]. A key bottlebeck with BO is its high run-time complexity. The studies conducted using BO focused on estimating parameters over multiple iterations for the same trajectory until the satisfactory optima was found. This may not be feasible in practice especially when the system is deployed in the field. Using reachability analysis, GPs were used along with reinforcement learning (RL) to ensure safety [15]. An RL algorithm combined with CBF guarantees safety while learning the set of explorable policies as shown in [16]. A barrier certified approach along with adaptive RL was used for learning and extending the set of safe policies [17]. RL approaches can be limiting since they have to learn their policies before executing them on the system. Gaussian processes (GPs) were used along with CBFs to learn unmodeled quadrotor dynamics and expand the safe set for the system online [18]. However, the optimization framework used was expensive to address safety while confining the study only to the altitude setting.

Our *key contributions* are the following. Firstly, we design a novel CBF that uses symmetric positive definite kernels for encoding the safety of the system called Gaussian CBF. Kernel functions are flexible in that they can encode any assumptions made about the underlying hypothesis of functions that one wants to estimate. We use GPs to learn from the data and determine the hyperparameters of the kernel which then directly adjust the relative notion of safety. To the best of our knowledge, this is the first work exploiting properties of kernel functions for constructing CBFs. Secondly, we perform safety constrained attitude control while learning the model uncertainties for a quadrotor using Gaussian CBF on $SO(3)$. Due to the highly nonlinear attitude dynamics on the tangent bundle to $SO(3)$, learning in this group has not been demonstrated before especially while addressing

Mouhyemen Khan and Abhijit Chatterjee are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA: {mouhyemen.khan, abhijit.chatterjee}@gatech.edu

safety. By performing online kernel update and modifying the Gaussian CBF's kernel hyperparameters, we achieve safe attitude control in the presence of model uncertainties.

The outline of the paper is as follows. Section II defines the problem statement and assumptions made. Background preliminaries are covered in Section III. The design and online kernel update for Gaussian CBFs is discussed in Section IV. An application test case for a quadrotor on $SO(3)$ using our proposed approach is investigated in Section V. Simulation results are provided in Section VI, followed by conclusions in Section VII.

II. Problem Statement

We consider a general control affine dynamical system,

$$\dot{x} = \underbrace{f(x(t)) + g(x(t))u(t)}_{\text{parametric}} + \underbrace{\mathcal{E}(x(t))}_{\text{non-parametric}}, \tag{1}$$

where $x(t) \in \mathcal{X} \subseteq \mathbb{R}^n$ and $u(t) \in \mathcal{U} \subseteq \mathbb{R}^m$ are the state and control input of (1) at time t. The system dynamics is divided into a known parametric model, $[f(x(t)) + g(x(t))u(t)] \in \mathbb{R}^n$, and an unknown non-parametric component, $\mathcal{E}(x(t)) \in \mathbb{R}^n$. The latter is the unmodeled dynamics arising from system uncertainties or disturbances. Now, consider a safe set for the state space of system (1), encoded as the superlevel set $\mathcal{S}$ for a smooth function $h(x(t)) : \mathcal{X} \to R$ as follows,

$$\mathcal{S} = \{x(t) \in \mathcal{X} \mid h(x(t)) \geq 0\}. \tag{2}$$

We are interested in designing control input $u(t)$ that renders (1) safe while encoding the non-parametric component of (1) in the design of $h(x(t))$. We achieve this by ensuring forward invariance for $\mathcal{S}$ with the help of control barrier functions (CBF). Encoding the unmodeled component is done by using kernels to construct the smooth function, $h(x(t); \Theta)$, which is not only characterized by the state $x(t)$, but also by additional hyperparameters, $\Theta \in \mathbb{R}^d$. Henceforth, we omit denoting the implicit dependence on t unless otherwise required.

Estimating the unmodeled dynamics without making any prior assumptions is an ill-posed problem. To this effect, we assume that we can measure $\hat{y} = \dot{x} - (f(x) + g(x)u) + \mathcal{N}(0, \omega_n^2)$, which are corrupted by zero-mean noise and variance ω_n^2. The non-parametric component, $\mathcal{E}$ is assumed to be locally Lipschitz continuous. Finally, we assume a nominal controller, $u_{nom}(t)$, exists that drives the parametric model, $f(x) + g(x)u_{nom}$, to the zero equilibrium point.

III. Background Preliminaries

In order to ensure forward invariance of safe set $\mathcal{S}$ and construct the smooth function h, we leverage important theoretical properties from CBFs and positive definite kernels.

A. Control Barrier Function

Consider only the parametric model of system (1),

$$\dot{x} = f(x) + g(x)u, \tag{3}$$

The drift vector and control matrix fields, $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ respectively, are assumed to be locally Lipschitz continuous. Let the safe state space for (3) be encoded as the zero superlevel set $\mathcal{S}$ defined similar to (2) of a smooth function $h : \mathcal{X} \to R$.

Definition 1 (Control Barrier Function [5])**.** *The function $h(x) : \mathcal{X} \to \mathbb{R}$ is defined as a control barrier function (CBF), if there exists an extended class-κ function ($\kappa(0) = 0$ and strictly increasing) such that for any $x \in \mathcal{S}$,*

$$\sup_{u\in\mathcal{U}} \Big\{ L_f h(x) + L_g h(x)u + \kappa(h(x)) \Big\} \geq 0, \tag{4}$$

where $L_f h(x) = \frac{\partial h}{\partial x} f(x)$ and $L_g h(x) = \frac{\partial h}{\partial x} g(x)$ are the Lie derivatives of $h(x)$ along $f(x)$ and $g(x)$ respectively.

Theorem 1 ([5])**.** *Given a system (3), with safe set $\mathcal{S} \subset \mathbb{R}^n$ defined by (2), and smooth CBF $h(x) : \mathcal{S} \to \mathbb{R}$ defined by (4), any Lipschitz continuous $u \in \mathcal{U}$, that satisfies $\bar{\mathcal{U}} = \{u \in \mathcal{U} \mid L_f h(x) + L_g h(x)u + \kappa(h(x)) \geq 0\}$ for any $x \in \mathcal{X}$, renders the safe set $\mathcal{S}$ forward invariant for (3).*

As seen from Theorem 1, CBFs are limited to systems with relative degree one, $\rho = 1$. For systems with $\rho > 1$, we look at an extension of CBFs called Exponential CBFs [19].

Definition 2 (Exponential Control Barrier Function [19])**.** *The smooth function $h(x) : \mathcal{X} \to \mathbb{R}$, with relative degree ρ, is defined as an exponential control barrier function (ECBF), if there exists $\mathcal{K} \in \mathbb{R}^\rho$ such that for any $x \in \mathcal{X}$,*

$$\sup_{u\in\mathcal{U}} \Big\{ L_f^\rho h(x) + L_g L_f^{\rho-1} h(x)u + \mathcal{K}^\top \mathcal{H} \Big\} \geq 0, \tag{5}$$

where $\mathcal{H} = [h(x), L_f h(x), ..., L_f^{(\rho-1)} h(x)]^\top$ is the Lie derivative vector for $h(x)$, and $\mathcal{K} = [k_0, k_1, ..., k_{\rho-1}]^\top$ is the coefficient gain vector for $\mathcal{H}$. $\mathcal{K}$ can be easily determined using linear control methods such as pole placement. We refer the reader to [19] for proofs of ECBF forward invariance.

B. Positive Definite Kernels

Kernels furnish a notion of similarity between input points, x and x'. For example, in supervised learning problems, the basic assumption made is that input points that are close to eachother are likely to have their target values, y also close to one another. Hence, the notion of nearness or similarity measure plays a key role. However, any arbitrary function of input pairs, x and x', will not constitute a valid kernel [10]. We refer the reader to [10] for a review of different kernels.

Definition 3 (Positive Definite Kernel)**.** *Let $\mathcal{X}$ be a nonempty set. A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive definite kernel [10], if the matrix $K \in \mathbb{R}^{n\times n}$, with entries, $[K]_{(i,j)} = k(x_i, x_j)$ is positive semidefinite ($v^\top K v \geq 0,\ \forall v \in \mathbb{R}^n$), for a finite set $X := (x_1, \cdots, x_n) \in \mathcal{X} \subseteq \mathbb{R}^n$.*

A stationary kernel is a function of $(x - x')$ and is invariant to translations in the input space. A popular choice of the kernel function is the squared exponential kernel, also called the Gaussian kernel or Radial Basis Function (RBF). The Gaussian kernel is defined as follows,

$$k(x, x') = \exp\bigg(- \frac{||x - x'||^2}{2l^2} \bigg), \tag{6}$$

where $l \in \mathbb{R}$ denotes the characteristic length scale of the associated hypothesis space of functions. A small value of l implies the underlying function changes rapidly, while an increasing l denotes a slowly-varying function. Intuitively, as x and x' get closer, they constitute a very high similarity measure, i.e., $k \to 1$. Conversely, as the points get further apart, $k \to 0$. For stationary kernels, checking its mean square continuity implies checking for continuity at $k(0,0)$. The mean-square differentiability (or smoothness property) of a stationary kernel is determined around $x = 0$ [10].

The Gaussian kernel is infinitely differentiable, hence, it is infinitely mean-square differentiable. This property of the Gaussian kernel being infinitely smooth while encoding the notion of nearness makes it ideal for designing the smooth function h, which constructs the superlevel safe set $\mathcal{S}$.

IV. Gaussian Control Barrier Function

Given a CBF h, the system is guaranteed to remain inside $\mathcal{S}$ and be safe due to the forward invariance property [5]. However, h does not inherently evolve or encode additional properties to account for unmodeled effects. To that end, we define a new form of CBF using the Gaussian kernel that is flexible to account for free parameters, Θ, while preserving all the properties of CBF. We will later look at how these free parameters are defined. First, we formally define the Gaussian CBF using the Gaussian kernel and CBF.

A. Definition of Gaussian Control Barrier Function

Definition 4 (Gaussian Control Barrier Function)**.** *A smooth function $h(x;\Theta) : \mathcal{X} \to \mathbb{R}$ is defined as a Gaussian control barrier function for (3) with an infinitely mean-square differentiable positive definite kernel, $k(x,x') : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that $k(x,x') \to 0$ as $\|x - x'\| \to \infty$, if there exists an extended class-κ function such that for any $x \in \mathcal{X}$,*

$$\sup_{u \in \mathcal{U}} \Big\{ L_f h(x;\Theta) + L_g h(x;\Theta) u + \kappa(h(x;\Theta)) \Big\} \geq 0, \quad (7)$$

We then have the following corollary from Theorem 1.

Corollary 1. *Given a system defined by (3) with a safe set $\mathcal{S} \subset \mathbb{R}^n$ as defined by (2), and a smooth Gaussian CBF $h(x;\Theta) : \mathcal{S} \to \mathbb{R}$ be defined by (7), any Lipschitz continuous $u \in \mathcal{U}$, that satisfies $\bar{\mathcal{U}} = \{u \in \mathcal{U} \mid L_f h(x;\Theta) + L_g h(x;\Theta) u + \kappa(h(x;\Theta)) \geq 0\}$ for any $x \in \mathcal{X}$, renders the safe set $\mathcal{S}$ forward invariant for the system (3).*

We first develop the following Gaussian constraint:

$$\exp\left(\frac{-\|x - x'\|^2}{2l^2}\right) - \exp\left(\frac{-\|x_c - x'\|^2}{2l^2}\right) \geq 0, \quad (8)$$

where $x \in \mathcal{X}$ is the state, $x' \in \mathcal{X}$ is the safest state, $x_c \in \mathbb{R}^n$ is the safety limit of the Gaussian region. Intuitively, the constraint ensures that all values of $|x| \leq x_c$ are inside the safe set. Now, we develop the Gaussian CBF:

$$h := \sigma^2 \exp\left(\frac{-\|x - x'\|^2}{2l^2}\right) - \sigma^2 \exp\left(\frac{-\|x_c - x'\|^2}{2l^2}\right), \quad (9)$$

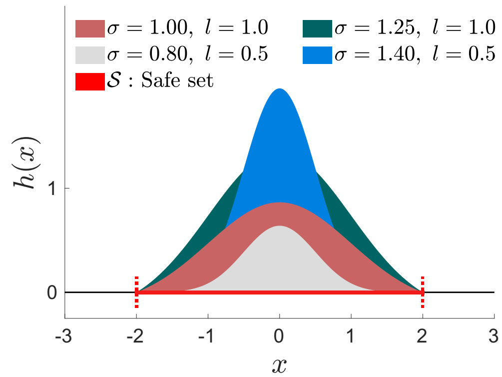

Fig. 1. Gaussian CBF in 1-dimension with $x' = 0$, $x_c = 2$, and different hyperparameter settings. Notice, the safe set $\mathcal{S}$ does not change with varying hyperparameters, although, the notion of safety given by $h(x)$ changes.

where $\Theta = [\sigma, l] \in \mathbb{R}^2$ constitute the hyperparameters of Gaussian CBF. Instead of using the normalized Gaussian kernel as expressed in (6), we pre-scale the kernel with a free parameter $\sigma \in \mathbb{R}^2$ representing the signal variance for the associated hypothesis of functions. The effect of the Gaussian CBF along with different hyperparameter settings are illustrated in Figure 1. As the hyperparameters are altered, the relative notion of safety encapsulated by h changes, but the safe set $\mathcal{S}$ does not change.

B. Online Kernel Update using Gaussian Process

Now, we look at how the hyperparameters are determined for the Gaussian CBF. In the case of normalized Gaussian CBF, we can simply set the hyperparameters to be unity i.e. $\sigma = 1, l = 1$ (see Figure 1). However, in order to account for unmodeled effects in (1), these hyperparameters need to be learned from observed data and accordingly adjusted depending on the unmodeled effects.

Gaussian processes (GP) are an ideal candidate for learning due to its non-parametric nature. GPs also offer flexible priors since they learn the underlying unknown function and estimate any hyperparameters from the data itself. The covariance (or kernel) function in a GP encodes any assumptions made about the unknown function. Hence, learning the function given an observed dataset also determines the hyperparameters for its covariance function (or kernel).

We are interested in determining hyperparameters Θ based on an underlying latent function $\mathcal{E}(x)$ in (1). Given N input vectors, $x \in \mathcal{X}$, and scalar noisy observations, $\hat{y} \in \mathbb{R}$, where $\hat{y} = \dot{x} - f(x) - g(x)u + \mathcal{N}(0, \omega_n^2) = \mathcal{E}(x) + \mathcal{N}(0, \omega_n^2)$, we compose the dataset: $\mathcal{D}_N = \{\mathbf{X}, \mathbf{y}\}$, where $\mathbf{X} = \{x_i\}_{i=1}^N$ and $\mathbf{y} = \{\hat{y}_i\}_{i=1}^N$. This allows us to compute the marginal likelihood for the observed dataset $\mathcal{D}_N = \{\mathbf{X}, \mathbf{y}\}$:

$$\left.\begin{aligned} p(\mathbf{y}|\mathbf{X}) &= \textstyle\int p(\mathbf{y}|\mathcal{E}, \mathbf{X}) p(\mathcal{E}|\mathbf{X}) d\mathcal{E} \\ &= \mathcal{N}(\mathbf{y}\ ;\ 0, K_y), \end{aligned}\right\} \quad (10)$$

where $K_y = K + \sigma_n^2 \mathbb{I}$, $K \in \mathbb{R}^{N \times N}$ has entries $[K]_{(i,j)} = k(x_i, x_j),\ i,j \in \{1, \ldots, N\}$, is the covariance matrix whose elements are given by the Gaussian kernel measure between pairs of input points using (6) , and $I \in \mathbb{R}^{N \times N}$ is the identity

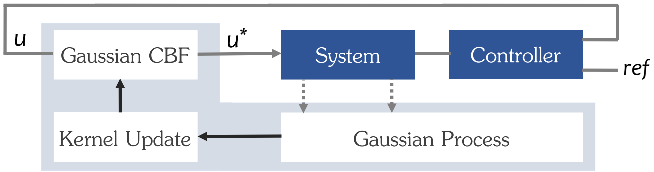

Fig. 2. The architecture with online kernel update and nominal controller modification using Gaussian CBF. By optimizing for the log marginal likelihood of a Gaussian distribution given a dataset, the hyperparameters for the kernel are updated online. The optimized hyperparameters further tune the relative safety notion for the Gaussian CBF.

matrix. The log marginal likelihood of (10) is then given by,

$$\log p(\mathbf{y}|\mathbf{X},\Theta) = -\frac{1}{2}\big(N\log(2\pi) + \log|K_y| + \mathbf{y}(K_y)^{-1}\mathbf{y}\big).$$

By taking the partial derivatives of the log marginal likelihood with respect to the hyperparameters [10], the hyperparameters are then optimized by maximizing:

$$\frac{\partial}{\partial\theta_j}\log p(\cdot) = \frac{1}{2}\mathrm{tr}\Big(\big(K_y^{-1}\mathbf{y}\mathbf{y}^\top K_y^{-\top} - K_y^{-1}\big)\frac{\partial K_y}{\partial\theta_j}\Big), \quad (11)$$

where θ_j is the j-th element in Θ. The optimization step can then be solved by using gradient ascent methods [10]. Solving (11) returns the desired hyperparameters, $\Theta^* = [\sigma^*, l^*]$ which are then used to update Gaussian CBF in (9). Notice that an inverse operation is performed in (11) which gives a complexity of $\mathcal{O}(N^3)$. Therefore, as the dataset grows larger, the online kernel update will not be tractable. To alleviate this issue, the points in the dataset are added and removed periodically to ensure the overall computation remains upper bounded to achieve online performance.

C. Safe Control with Gaussian Control Barrier Function

Consider a nominal control input $u_{nom}(t) \in \mathcal{U} \subset \mathbb{R}^m$ is designed as the feedback control policy for system (1). The safety of the system cannot be guaranteed with such a given nominal control policy. Corollary 1 allows rectifying the nominal control policy using the Gaussian CBF framed as an online quadratic program (QP) optimization problem [5]. By rectifying the control policy, the system is guaranteed to remain forward invariant for the safe set $\mathcal{S}$.

Gaussian CBF-QP: *Input modification*

$$u^* = \arg\min_{u\in\mathcal{U}} \frac{1}{2}\|u - u_{nom}\|^2 \quad \text{s.t.} \quad (12)$$
$$L_f h(x;\Theta^*) + L_g h(x;\Theta^*)u + \kappa(h(x;\Theta^*)) \geq 0,$$

where Θ^* is the solution to (11). The resultant architecture for performing safe control using Gaussian CBF with online kernel update is shown in Figure 2.

D. Illustrative Example: Inverted Pendulum

For verification and illustration purposes, we verify the simple case of an actuated planar pendulum using the Gaussian CBF in (9) with normalized hyperparameters for the kernel. Later, we will be particularly interested in applying online kernel update for a quadrotor scenario and juxtaposing its behavior without the kernel update for comparison. Here, we briefly demonstrate the safe control scenario using Gaussian CBF. The dynamics is given by,

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = -\frac{g\sin(x_1)}{l_p} - \frac{bx_2}{m} + \frac{u}{ml_p},$$

where m is the pendulum mass, l_p is its length, b is damping coefficient, and g is gravity. The system state is given by $x = [\phi,\ \dot{\phi}]$. We rewrite the Gaussian CBF for 1-D as a function of the pendulum's angular position,

$$h := \sigma^2\exp\left(\frac{-\|r_\phi\|^2}{2l^2}\right) - \sigma^2\exp\left(\frac{-\|r_{\phi_c}\|^2}{2l^2}\right), \quad (13)$$

where $r_\phi = \phi - \phi'$ and $r_{\phi_c} = \phi_c - \phi'$. Computing the Lie derivatives of (13) gives,

$$L_f h = -\frac{\sigma^2\dot{\phi}\,\gamma_\phi\, r_\phi}{l_p^2},$$
$$L_f^2 h = \frac{\sigma^2\ \gamma_\phi\ r_\phi\big(\frac{g\sin(\phi)}{l_p} + \frac{b\dot{\phi}}{m}\big)}{l_p^2} - \frac{\sigma^2\dot{\phi}^2\gamma_\phi}{l_p^2}\left(1 - \frac{r_\phi^2}{l_p^2}\right),$$
$$L_g L_f h = -\frac{\sigma^2\gamma_\phi r_\phi}{l_p^2},$$

where $\gamma_\phi = \exp\big(-r_\phi^2/(2l_p^2)\big)$. The Lie derivatives are used as constraints to the QP formulation. The nominal control is designed using feedback linearization and then rectified using the Gaussian CBF-QP formulation for relative degree 2. The reference is set to $\phi_d = 40°$, Gaussian barrier limit to $\phi_c = 20°$, and $\phi' = 0°$. The safe control behavior using Gaussian CBF is shown in Figure 3. Since the reference trajectory is outside the Gaussian barrier limit imposed by ϕ_c, the modified control input ensures the system does not violate the safety constraints designed using the Lie derivatives.

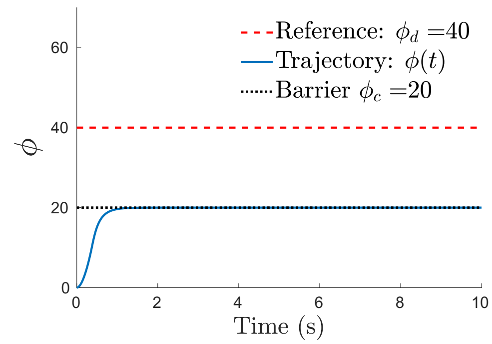

Fig. 3. Safe control for actuated planar pendulum using Gaussian CBF-QP. The tracking behavior is relaxed to uphold safety limit imposed by ϕ_c.

V. Application Test Case: Quadrotor on SO(3)

In this section, we demonstrate the versatility of our proposed approach for safe attitude control of a quadrotor whose attitude is represented in the *Special Orthogonal* group $SO(3)$. This is an interesting problem due to the highly nonlinear attitude dynamics of the quadrotor evolving in the tangent bundle to $SO(3)$. Moreover, constraining its motion in this group will require the Gaussian CBF to be generalizable to these manifolds. Due to the nature of Gaussian CBFs, the kernel can map the input points to another feature space and learn the necessary hyperparameters. An interesting application of constraining the quadrotor attitude is avoiding large tilt angles in order to maintain flight safety or payload delivery. Hence, the goal is to execute safe attitude constrained control in presence of disturbance for the quadrotor.

We first discuss the quadrotor's geometric attitude dynamics and controller on $SO(3)$. We then discuss the design of Gaussian CBF represented on $SO(3)$ along with the safe controller modification and kernel update.

A. Geometric Quadrotor Control on SO(3) Revisited

We consider a fully actuated quadrotor whose configuration is specified on the Lie group $SO(3)$. Due to full actuation, the number of internal forces equal to the manifold's dimension. The system dynamics evolving in the tangent bundle to $SO(3)$ is given by,

$$\left.\begin{aligned} \dot{R} &= R\Omega^{\times} \\ J\dot{\Omega} &= M - (\Omega \times J\Omega) \end{aligned}\right\}, \tag{14}$$

where $(\cdot)^{\times} : \mathbb{R}^3 \to so(3)$ is the cross-map defined by, $\forall x, y \in \mathbb{R}^3, x^{\times}y = x \times y$, $J \in \mathbb{R}^{3\times3}$ is the inertia matrix, $\Omega \in \mathbb{R}^3$ is the body-angular velocity, and $M \in \mathbb{R}^3$ is the control input. The quadrotor's dynamics evolves in a coordinate-free framework which uses a geometric representation for its attitude given by a rotation matrix $R \in SO(3)$, where $SO(3) := \{R \in \mathbb{R}^{3\times3} | R^{\top}R = \mathbb{I},\ \det(R) = 1\}$. R represents the rotation from the quadrotor's body-frame to the inertial-frame.

For a given smooth attitude command $R_d \in SO(3)$ and associated kinematics $\dot{R}_d = R_d\Omega_d^{\times}$, where $\Omega_d \in \mathbb{R}^3$ is the desired angular velocity, the tracking errors are given by [20],

$$\left.\begin{aligned} e_R &= \tfrac{1}{2}(R_d^{\top}R - R^{\top}R_d)^{\vee} \\ e_{\Omega} &= \Omega - R^{\top}R_d\Omega_d \end{aligned}\right\}, \tag{15}$$

where $(\cdot)^{\vee} : so(3) \to \mathbb{R}^3$ is the vee-map, also the inverse of the cross-map, satisfying $(y^{\times})^{\vee} = y$. The control input $M \in \mathbb{R}^3$ for attitude tracking is designed as follows [20]:

$$M = -k_R e_R - k_{\Omega} e_{\Omega} + \Omega \times J\Omega - J\beta, \tag{16}$$

where $\beta := (\Omega^{\times}R^{\top}R_d\Omega_d - R^{\top}R_d\dot{\Omega}_d) \in \mathbb{R}^3$, k_R, k_{Ω} are any positive constants, and $\dot{\Omega}_d$ is the desired angular acceleration. It is shown in [20] that for system (14), controller (16) exponentially stabilizes to the zero equilibrium for (15).

B. Gaussian CBF with Kernel Update on $SO(3)$

We first construct the Gaussian CBF on $SO(3)$ using the following smooth function,

$$\begin{aligned} h_{SO(3)} &:= \sigma^2 \exp\left(\frac{\text{tr}(R - R')^2}{l^2}\right) - \sigma^2 \exp\left(\frac{\text{tr}(R_c - R')^2}{l^2}\right) \\ &= \sigma^2 \exp\left(\frac{-\text{tr}(R - \mathbb{I})^2}{l^2}\right) - \sigma^2 \exp\left(\frac{-\text{tr}(R_c - \mathbb{I})^2}{l^2}\right) \end{aligned} \tag{17}$$

where $\Theta = [\sigma, l] \in \mathbb{R}^2$ constitute the hyperparameters, $R \in SO(3)$ is the quadrotor attitude, $R' \in SO(3)$ is the safest attitude offset, and $R_c \in SO(3)$ is the attitude constraint. The attitude offset is set to identity, $R' := \mathbb{I}$, which implies that the quadrotor is at its safest when it is oriented without any tilt (see Figure 4). The attitude constraint is parameterized by using Euler ZYX convention, $R_c := R_z(\psi)R_y(\theta)R_x(\psi)$. Gaussian CBF on $SO(3)$ is shown in Figure 5 by parameterizing R using Euler ZYX convention for different values of roll and pitch angles, and setting the yaw angle to $0°$. The safe set is defined as,

$$\mathcal{S} = \{R \in SO(3) \mid h_{SO(3)} \geq 0\}. \tag{18}$$

In order to rectify the nominal control moment (16), we compute the Lie derivatives of (17) which are used as constraints for the QP optimization step. The Lie derivatives computed are given by the following:

$$\begin{aligned} L_f h_{SO(3)} &= -\frac{2\sigma^2\alpha}{l^2}\exp\left(\frac{-\alpha^2}{l^2}\right)\text{tr}(R\Omega^{\times}), \\ L_f^2 h_{SO(3)} &= \frac{4\sigma^2}{l^4}\left[\exp\left(\frac{-\alpha^2}{l^2}\right)\Big(\alpha\text{tr}(R\Omega^{\times})\Big)^2\right] \\ &\quad - \frac{2\sigma^2}{l^2}\left[\exp\left(\frac{-\alpha^2}{l^2}\right)\text{tr}(R\Omega^{\times})^2\right] \\ &\quad - \text{tr}\Big\{R\Omega^{\times}\Omega^{\times} - R\Big[J^{-1}(\Omega \times J\Omega)\Big]\Big\}, \\ L_g L_f h_{SO(3)} &= \frac{2\sigma^2\alpha}{l^2}\exp\left(\frac{-\alpha^2}{l^2}\right)J^{-1}(R - R^{\top})^{\vee}, \end{aligned}$$

where $\alpha := \text{tr}(R - \mathbb{I}) \in \mathbb{R}$. Given the nominal control moment M_{nom} developed in (16), we can now construct the QP to compute the safe control input M^* resulting utlimately in a Gaussian CBF-QP formulation (see Figure 2),

---

Gaussian CBF-QP: *Moment Modification*

$$\begin{aligned} M^* = \underset{M\in\mathbb{R}^3}{\arg\min}\ &\frac{1}{2}\|M - M_{nom}\|^2 \quad \text{s.t.} \\ &L_f^2 h_{SO(3)} + L_g L_f h_{SO(3)} M + \mathcal{K}^{\top}\mathcal{H} \geq 0, \end{aligned}$$

---

where $\mathcal{K} = [\kappa_1\ \kappa_2]^{\top} \in \mathbb{R}^2$, and $\mathcal{H} = [L_f h_{SO(3)}\ h_{SO(3)}]^{\top} \in \mathbb{R}^2$. The modified control input M^* ensures the quadrotor

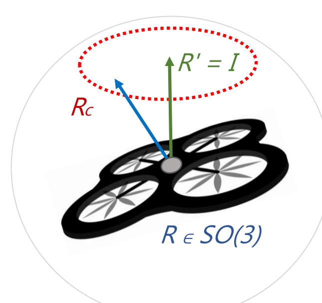

Fig. 4. Constrained quadrotor attitude inside unit sphere $\mathbb{S}^2$. $R' = \mathbb{I}$ gives the attitude offset relative to which the safe set $\mathcal{S}$ is determined.

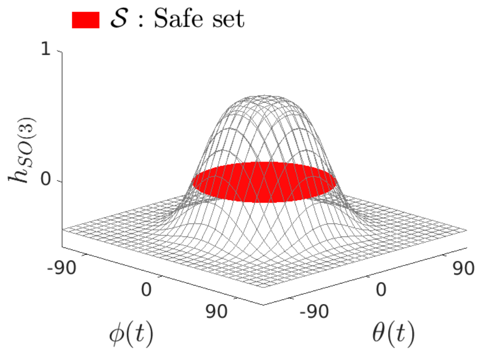

Fig. 5. Gaussian CBF on $SO(3)$ with the safe set $\mathcal{S}$ designating the safe attitude set. The roll angle is constrained to $\phi = 60°$.

remains in the safe set $\mathcal{S}$. If the reference attitude is inside the safe set, then no modification is done to the nominal control moment. However, if the reference attitude goes outside the constraint set, then tracking is relaxed to uphold safety.

Now, we look at the online kernel update for the Gaussian CBF on $SO(3)$. To perform the optimization step in (11), we first form the dataset $\mathcal{D}_N$ where points are added and removed at each sampling time to hold a maximum of upto N points. The input vector is formed by using the attitude and angular velocity states, $x = [\Omega\ R] \in \mathbb{R}^{12}$, where $R \in \mathbb{R}^9$ is recast into a column vector. The noisy target observations are computed using the attitude dynamics, $\hat{y} = \dot{\Omega} - J^{-1}M - J^{-1}(\Omega \times J\Omega) + \mathcal{N}(0, \omega_n^2) \in \mathbb{R}^3$. The optimization step is performed in every iteration generating the hyperparameters, $\Theta^* = [\sigma^*, l^*]$ using (11). These hyperparameters are then used to modify the Gaussian CBF and its Lie derivatives on $SO(3)$. The resulting safe control moment M^* along with the online kernel update is given by,

---

Gaussian CBF-QP: *Moment modification with Kernel update*

$$
\begin{aligned}
M^* = \underset{M \in \mathbb{R}^3}{\arg\min}\ & \frac{1}{2}\|M - M_{nom}\|^2 \quad \text{s.t.} \\
& L_f^2 h_{SO(3)}(R;\Theta^*) + L_g L_f h_{SO(3)}(R;\Theta^*)M + \\
& \mathcal{K}^\top \mathcal{H}(R;\Theta^*) \geq 0,
\end{aligned}
$$

$h_{SO(3)}(R;\Theta^*)$ is characterized by the state R and optimized hyperparameters Θ^* computed at every time step.

VI. Simulation Results

In this section, we present simulation results for safe attitude control of a quadrotor using the Gaussian CBF developed in Section V-B. We consider two cases:

1) Safe quadrotor control without model uncertainties
2) Safe quadrotor control with model uncertainties

The purpose is to demonstrate the efficacy of Gaussian CBF in both the absence and presence of model uncertainties. Note that, in both the cases, online kernel update is performed. The parameters used in the simulation are $k_\Omega = 0.3, k_R = 2.4, J = \text{diag}[\ 0.0156,\ 0.0156,\ 0.021], R(0) = \mathbb{I}, \Omega(0) = 0$. Simulation of the dynamics is performed using a 4^{th} order Range-Kutta solver with $t = [0, 20]$ sec. The QP is solved online using MATLAB's `quadprog` solver taking 0.8ms per time step on an Intel i7-7700HQ machine equipped with 16GB RAM. For covariance calculations and hyperparameter estimations, the `GPML` toolbox is used [21]. A smooth reference attitude command is described by using ZYX Euler angles, $R_d(t) = R_d(\phi(t), \theta(t), \psi(t))$, where $\phi_d(t) = \frac{\pi}{4}\sin(\frac{\pi}{2}t), \theta_d(t) = \frac{\pi}{4}\cos(\frac{\pi}{2}t)$, and $\psi_d(t) = 0$.

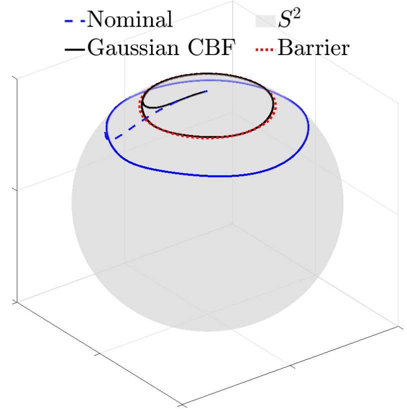

Fig. 6. Safe constrained control with the nominal controller and with Gaussian CBF where the constraint is defined by $R_c = R_z(0)R_y(0)R_x(30)$. The nominal control input violates the safe set. With Gaussian CBF, the control input is rectified to stay inside the constraint set, and thereby relax reference tracking.

A. *Safe Quadrotor Control on $SO(3)$ without uncertainty*

In the first scenario, we look at safe attitude control using the Gaussian CBF to constrain the attitude represented in $SO(3)$. We keep $R' = I$ and constrain the Euler angles to be $(\phi, \theta, \psi) = (30°, 0°, 0°)$. This results in a constrained attitude given as $R_c = R_z(0)R_y(0)R_x(30)$.

The quadrotor trajectory using only the nominal control moment and Gaussian CBF is shown in Figure 6. As seen from the figure, the quadrotor's nominal trajectory violates the barrier limit imposed by R_c. This is because the nominal control input is designed to follow the attitude reference R_d without upholding any safety constraints. By applying Gaussian CBF, the quadrotor's trajectory is constrained inside the safe set. The online QP modifies the control input to uphold the attitude barrier limit at the expense of relaxed reference tracking. Note that in the absence of uncertainties, the kernel update does not affect the safety performance.

B. *Safe Quadrotor Control on $SO(3)$ with uncertainty*

Next, we look at safe attitude control in the presence of uncertainties and the effect of online kernel update to adjust the Gaussian CBF on $SO(3)$. The attitude dynamics is disturbed using the following disturbance model,

$$\mathcal{E} = 0.2[\sin(2\pi t) \quad \cos(5\pi t) \quad R_{11}(t)], \tag{19}$$

where R_{11} is $(1,1)$ element of R. The dataset is collected upto $N = 21$ data points. The hyperparameter estimation is bounded within $l = [0.25, 2.0]$ and $\sigma = [0.25, 6.0]$. This prevents scaling the Gaussian CBF to arbitrarily small values in the presence of small or no disturbances (as discussed in VI-A). This is reasonable due to the assumpting that model uncertainties are locally Lipschitz continuous. Data

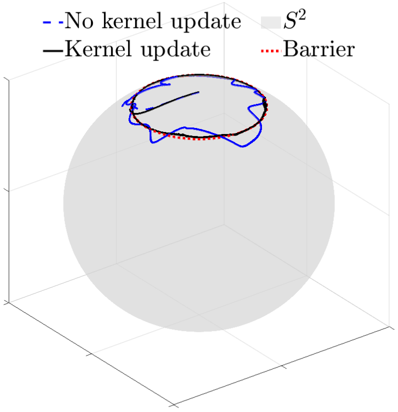

Fig. 7. Constrained control in the presence of disturbance for normalized hyperparameters and with online kernel update. For the normalized setting, safety is violated in the presence of unmodeled disturbances. By learning the unmodeled dynamics and performing the online kernel update, the relative notion of safety updates the smooth function $h_{SO(3)}$ and the Lie derivatives.

samples are iteratively updated at 100Hz. Performing the hyperparameter optimization step takes under 25ms.

A comparison of constrained attitude control without kernel update and with the kernel update is shown in Figure 7. When no kernel update is performed, a normalized Gaussian CBF is used i.e. $\Theta = [1, 1]$. In this case, we see that the system trajectory goes outside the constraint set. This is expected since the rectification step that uses QP does not take into account unmodeled dynamics such as disturbances. When the online kernel update is performed, the quadrotor remains inside the constraint set. By solving for the hyperparameters at each time step, the relative notion of safety quantified by the Gaussian CBF changes. Due to changing hyperparameters, the Lie derivatives acting as constraints to the Gaussian CBF-QP setup is also affected. This results in generating safe control policies while taking into account the unmodeled effects.

VII. Concluding Remarks

We constructed a new CBF called Gaussian CBF that uses positive definite kernels in its design. Gaussian CBFs incorporated state constraints along with free parameters called hyperparameters. These hyperparameters characterized the relative notion of safety along with the constraint set. A data-driven approach was taken using GPs to learn the unmodeled dynamics and perform an online kernel update to modify Gaussian CBF and its associated Lie derivatives. We demonstrated our proposed strategy for safe attitude control of a quadrotor platform, whose configuration is expressed on $SO(3)$. We successfully performed constrained attitude control ensuring forward invariance property of Gaussian CBF. In the presence of model uncertainties or disturbances, the online kernel update adjusted the Gaussian CBF and ensured safety.

VIII. Acknowledgements

This research was supported in part by National Science Foundation under grant CISE:S&AS:1723997.

References

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

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

[3] S. Prajna, A. Jadbabaie, and G. J. Pappas, "A framework for worst-case and stochastic safety verification using barrier certificates," *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1415–1428, 2007.

[4] P. Wieland and F. Allgöwer, "Constructive safety using control barrier functions," *IFAC Proceedings Volumes*, vol. 40, no. 12, pp. 462–467, 2007.

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

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

[7] M. Khan, M. Zafar, and A. Chatterjee, "Barrier functions in cascaded controller: Safe quadrotor control," *arXiv preprint arXiv:1903.09711*, 2019.

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

[9] F. Berkenkamp, R. Moriconi, A. Schoellig, and A. Krause, "Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes," pp. 4661–4666, 12 2016.

[10] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*, pp. 63–71, Springer, 2003.

[11] D. Nguyen-Tuong and J. Peters, "Using model knowledge for learning inverse dynamics," in *2010 IEEE International Conference on Robotics and Automation*, May 2010.

[12] D. J. Lizotte, T. Wang, M. H. Bowling, and D. Schuurmans, "Automatic gait optimization with gaussian process regression.," in *IJCAI*, vol. 7, pp. 944–949, 2007.

[13] M. Tesch, J. Schneider, and H. Choset, "Using response surfaces and expected improvement to optimize snake robot gait parameters," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1069–1074, IEEE, 2011.

[14] F. Berkenkamp, A. P. Schoellig, and A. Krause, "Safe controller optimization for quadrotors with gaussian processes," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016.

[15] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin, "Reachability-based safe learning with gaussian processes," in *53rd IEEE Conference on Decision and Control*, pp. 1424–1431, IEEE, 2014.

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

[17] M. Ohnishi, L. Wang, G. Notomista, and M. Egerstedt, "Barrier-certified adaptive reinforcement learning with applications to brushbot navigation," *IEEE Transactions on Robotics*, vol. 35, no. 5, pp. 1186–1205, 2019.

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

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

[20] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor uav on se (3)," in *49th IEEE conference on decision and control (CDC)*, pp. 5420–5425, IEEE, 2010.

[21] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (gpml) toolbox," *Journal of machine learning research*, vol. 11, no. Nov, pp. 3011–3015, 2010.