Safe Multi-Agent Interaction through Robust Control Barrier Functions with Learned Uncertainties

Richard Cheng, Mohammad Javad Khojasteh, Aaron D. Ames, and Joel W. Burdick

Abstract—Robots operating in real world settings must navigate and maintain safety while interacting with many heterogeneous agents and obstacles. Multi-Agent Control Barrier Functions (CBF) have emerged as a computationally efficient tool to guarantee safety in multi-agent environments, but they assume perfect knowledge of both the robot dynamics and other agents’ dynamics. While knowledge of the robot’s dynamics might be reasonably well known, the heterogeneity of agents in real-world environments means there will always be considerable uncertainty in our prediction of other agents’ dynamics. This work aims to learn high-confidence bounds for these dynamic uncertainties using Matrix-Variate Gaussian Process models, and incorporates them into a robust multi-agent CBF framework. We transform the resulting min-max robust CBF into a quadratic program, which can be efficiently solved in real time. We verify via simulation results that the nominal multi-agent CBF is often violated during agent interactions, whereas our robust formulation maintains safety with a much higher probability and adapts to learned uncertainties.

I. INTRODUCTION

Collision-free robot navigation in natural multi-agent environments is vital for a myriad of robotic applications, such as self-driving cars, navigation in crowds, etc. However, placing robots in rapidly evolving, uncertain environments introduces many challenges in guaranteeing safety [1]–[5]. Uncertainty in the prediction of other agents’ trajectories is inevitable, and robots should learn and account for this uncertainty to ensure safe operation.

In this work, we utilize the Multi-Agent Control Barrier Function (CBF) proposed in [6] to generate low-level controllers that guarantee collision-free behavior. The Multi-Agent CBF uses an optimization-based controller to prevent the robot from entering unsafe sets (i.e. states leading to inevitable collision) in a minimally invasive fashion. However, previous work either assumes that the robot dynamics and the other agents’ dynamics are perfectly known, or considers highly conservative worst-case bounds [7], [8]. It is clear though that all natural environments (e.g. settings with humans) are fraught with varying levels of uncertainty. For example, human trajectories remain notoriously difficult to predict and are highly stochastic [9]. Thus, both capturing and accounting for uncertainty is crucial for safe navigation.

Statement of contributions: The goal of this paper is to (1) learn uncertainty bounds online from agents’ observed trajectories, and (2) incorporate those uncertainty bounds into the Multi-Agent CBF while maintaining computational efficiency of the underlying controller (i.e. a quadratic program).

Our approach focuses first on learning high-confidence polytopic bounds on the, possibly coupled, uncertainties in both the robot dynamics and other agents’ dynamics. To achieve this, we utilize Matrix-Variate Gaussian Processes (MVG) and optimize their hyperparameters offline from interaction data; this allows us to predict ellipsoidal uncertainty in our dynamics online, which we convert to an uncertainty polytope given a desired confidence level. Using these polytopic bounds, we formulate a robust CBF as a min-max optimization problem over the robot controls and the potential uncertainties, respectively. We then transform this min-max problem into a quadratic program that can be efficiently solved to find a safe control action that is robust with respect to our estimated uncertainty. See Fig. 1 for an overview of our approach.

Organization: Sec. II goes over related work in the safe collision avoidance literature, Sec. III provides background information on CBFs used to guarantee safety, and Matrix-Variate Gaussian Processes used to estimate correlated uncertainties. Sect. IV introduces the robust multi-agent CBF and shows how it can be efficiently solved under given uncertainty bounds. Sec. V looks at how to learn these confidence bounds over the uncertainty. Finally, Sec. VI presents simulation results illustrating the benefits of our algorithm and verifying the safety of our controller.
II. RELATED WORK

Multi-agent collision avoidance has been a long-studied problem with different approaches proposed for enabling safe control in varying situations. Velocity obstacles is a popular approach that involves limiting control actions to a set of “safe” actions, though its constant velocity with linear dynamics assumption is limiting [10], [11]. Related works in this direction have loosened these assumptions, but require significant sampling of the action space and do not incorporate dynamic uncertainty [12], [13]. More recently, Buffered Voronoi Cells (BVC) have been proposed as a tool to provide safety guarantees with only positional information, though safety guarantees are provided only under linear dynamics and without uncertainty [14], [15]. Also, [16]–[18] provide safety guarantees, under worst case disturbances, by solving the Hamilton-Jacobi-Isaacs equation to obtain a minimally invasive control law. However, the heavy computational expense prohibits applicability to large-scale multi-agent systems.

Reinforcement learning methods have emerged recently, which directly learn actions in multi-agent settings in response to the observed environment [19], [20]. However, these methods provide no formal guarantees of safety, and as such are prone to collision in novel environments. Furthermore, they have not been shown to scale to settings with many agents. To ensure safety in the reinforcement learning settings, the work [21] combined a safe backup policy with the learned policies. [22] incorporated safety constraints into the reinforcement learning framework, although this work considered only decoupled uncertainties and was limited to polytopic safety constraints. Bayesian inference was utilized in [23]–[25] to learn system dynamics in an online manner while ensuring safety (with high probability) using Control Barrier Functions for a single agent.

Control Barrier Functions (CBF) are a tool for enforcing set invariance of dynamical systems [26]–[28], and they have been used to guarantee collision avoidance in multiagent settings by projecting desired actions, to the closest (in least-squares sense) safe actions according to a CBF condition [29]–[31]. However, defining a valid CBF for general systems remains a challenge, especially in cases with significant uncertainty. Recent works have looked at learning CBFs for general systems, either implicitly or explicitly [32], [33]. A multi-agent CBF was defined explicitly for multi-agent systems in the case of continuous-time linear dynamics [6], [7]. However, this multi-agent CBF does not incorporate uncertainty or nonlinearity in the dynamics. Other work on robust CBFs deals only with highly conservative worst-case bounds [8]. We seek to address this gap in this paper.

Our work builds upon the current literature by computing the minimally invasive action necessary to maintain safety in a computationally efficient manner for discrete-time systems while incorporating nonlinear dynamics and learned uncertainty.

III. BACKGROUND

Our robotic system is represented by nonlinear control-affine dynamics in discrete-time:

\[
\begin{bmatrix}
  \dot{p}_{t+1} \\
  \dot{v}_{t+1} \\
  \dot{z}_{t+1}
\end{bmatrix} =
\begin{bmatrix}
  f_p(x_t) \\
  f_v(x_t) \\
  f_z(x_t)
\end{bmatrix} +
\begin{bmatrix}
  g_p(x_t) \\
  g_v(x_t) \\
  g_z(x_t)
\end{bmatrix} u +
\begin{bmatrix}
  d_p(x_t) \\
  d_v(x_t) \\
  d_z(x_t)
\end{bmatrix},
\]

where

\[
\begin{align*}
  p_t & = \text{position (robot)} \\
  v_t & = \text{velocity (robot)} \\
  z_t & = \text{position (other agents)}
\end{align*}
\]
where \( p \in \mathbb{R}^2, v \in \mathbb{R}^2, \) and \( z \in \mathbb{R}^{n-4} \) denote position, velocity and other states, respectively. Here, \( f_j, g_j, \) and \( d_j \) are real-valued functions, for \( j \in \{ p, v, z \} \), and \( u \in \mathcal{U} \) where \( \mathcal{U} := \{ u \in \mathbb{R}^m : ||u||_2 \leq u_{\text{max}} \} \). The functions \( f(x) \) and \( g(x) \) are assumed to be known, whereas \( d(x) \) represents unknown uncertainty in the dynamics, which we model with a Gaussian process [cf. Sec. III-B]. We assume that this system has relative degree 2 with respect to the positional output \( p \); in discrete time, this directly implies that \( g_p(x) = 0 \times 2 \). Similarly, let us represent the other agents within our multi-agent system with dynamics,

\[
\begin{bmatrix}
    x_{i+1} \\
    z_{i+1}
\end{bmatrix} =
\begin{bmatrix}
    f_p^i(x_t) \\
    f_v^i(x_t) \\
    f_z^i(x_t)
\end{bmatrix} +
\begin{bmatrix}
    d_p^i(x_t) \\
    d_v^i(x_t) \\
    d_z^i(x_t)
\end{bmatrix},
\tag{2}
\]

where \( i \in \mathbb{N} \) indexes each of the other agents in our system. We assume the control input for other agents are a (uncertain) function of their state at the given time, so we do not show control inputs explicitly in (2).

As our robot is interacting with other unknown agents, it will be important for us to account for the uncertainties, \( d, d_i \), when considering safety via CBFs.

A. Control Barrier Functions

Consider a safe set, \( \mathcal{C} \), defined by the super-level set of a continuously differentiable function \( h : \mathbb{R}^n \to \mathbb{R} \):

\[
\mathcal{C} := \{ x \in \mathbb{R}^n : h(x) \geq 0 \},
\tag{3}
\]

To maintain safety during the learning process, the system state must always remain within the safe set \( \mathcal{C} \) (i.e. the set \( \mathcal{C} \) is forward invariant with respect to the system dynamics). A set \( \mathcal{C} \) is forward invariant if for every \( x_0 \in \mathcal{C}, x(t) \in \mathcal{C} \) for all \( t \geq 0 \). In our multi-agent setting, this set could include all states where collision can be avoided given the robot’s input bounds. Control barrier functions utilize a Lyapunov-like argument to provide a sufficient condition for ensuring forward invariance of the safe set \( \mathcal{C} \) under controlled dynamics.

Definition 1: Given a set \( \mathcal{C} \in \mathbb{R}^n \) defined by (3), the continuously differentiable function \( h : \mathbb{R}^n \to \mathbb{R} \) is a discrete-time control barrier function (CBF) for dynamical system (1) if there exists \( \eta \in [0,1] \) such that for all \( x_t \in \mathcal{C} \),

\[
\sup_{u \in \mathcal{U}} \text{CBC}(x_t, u_t) \geq 0,
\tag{4}
\]

for \( \text{CBC}(x_t, u_t) := h(x_{t+1}(u_t)) + (\eta - 1)h(x_t) \), where CBC stands for control barrier condition.

If a function \( h(x) \) is a CBF, then there exists a controller such that the set \( \mathcal{C} \) is forward invariant [27], [34]. In other words, system safety is guaranteed by ensuring satisfaction of condition (4). Our goal is to ensure safety, by computing a minimally invasive control action that satisfies (4), in an online fashion. In particular, we utilize the following multi-agent CBF inspired by [6]:

\[
h(x) := \frac{\Delta p^T\Delta v}{\|\Delta p\|} + \sqrt{a_{\text{max}}(\|\Delta p\| - D_s)} \tag{5},
\]

where \( a_{\text{max}} \) represents our robot’s max acceleration in the collision direction, \( D_s \) is the collision margin, \( \Delta p = p - p_i \) is the positional difference between the agents, and \( \Delta v = v - v_i \) is the velocity difference between the agents.

The work [6] introduced a CBF similar to (5) for continuous-time linear systems, such that \( a_{\text{max}} \) can be determined easily. In Sec. IV, under an appropriate assumption, we show that (5) is a valid CBF for the discrete-time nonlinear dynamics (1) and (2).

B. Matrix-Variate Gaussian Process Regression

Here we illustrate how Bayesian learning can be used to acquire a distribution over the uncertainty in the dynamics. Since we are estimating a multivariate uncertainty \( d(x) \), we must consider potential correlations in its components. Thus, we use the Matrix-Variate Gaussian Process (MVG) model to learn the system dynamics and uncertainty from data. By learning \( \mu_d(x) \) and \( \Sigma_d(x) \) in tandem with the controller, we can obtain high probability confidence intervals on the unknown dynamics, which adapt/shrink as we obtain more information (i.e. measurements) on the system. We first start by defining the MVG distribution [24], [35]–[37] as follows.

Definition 2: We say the random matrix \( X \in \mathbb{R}^{n \times m} \) is distributed according to a MVG distribution when its probability density function is defined as:

\[
p(X; M, \Sigma, \Omega) := \frac{\exp(-\frac{1}{2} \text{tr}[(\Sigma^{-1}(X-M)^T\Sigma^{-1}(X-M))])}{(2\pi)^{m^2/2} \det(\Sigma)^{m^2/2} \det(\Omega)^{n^2/2}},
\tag{6}
\]

where \( M \in \mathbb{R}^{n \times m} \) denotes the mean, and \( \Sigma \in \mathbb{R}^{n \times n} \) encodes the covariance matrix of the rows, and \( \Omega \in \mathbb{R}^{m \times m} \) encodes the covariance matrix of the columns. In this case, we write \( X \sim \mathcal{MN}(M, \Sigma, \Omega) \), and we have \( \text{vec}(X) \sim \mathcal{N} (\text{vec}(M), \Omega \Sigma) \), where \( \text{vec}(X) \in \mathbb{R}^{nm} \) is the vectorization of \( X \), obtained by stacking the columns of \( X \), and \( \otimes \) is the Kronecker product.

We continue by modeling \( d(x) \) as a MVG on \( \mathbb{R}^n \). Without loss of generality, we assume zero mean for the MVG with positive semi-definite parameter covariance matrix \( \Sigma \in \mathbb{R}^{m \times m} \), and kernel \( \kappa : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R} \), that is,

\[
\text{vec}(d(x_1), \ldots, d(x_N)) \sim \mathcal{N}(0, \Sigma (x) \otimes \Omega),
\]

where \( \Sigma \in \mathbb{R}^{n \times n} \) with \( \Sigma_{ij} = \kappa(x_i, x_j) \). There are many potential choices for the kernel function \( \kappa(x_i, x_j) \) (cf. [38]), though we utilize the simple squared-exponential kernel in this work,

\[
\kappa(x_i, x_j) = \sigma^2 \exp\left(-\frac{(x_i - x_j)^2}{2l^2}\right),
\tag{7}
\]

where \( \sigma \) and \( l \) are kernel hyperparameters. Since \( d(x) \) is an MVG, the training observations \( y_{\text{train}} := [d(x_1), \ldots, d(x_N)] \) at sampling points \( x_{\text{train}} := [x_1, \ldots, x_N] \), and the predictive target \( d(x_s) \) at query test point, \( x_s \), are jointly Gaussian as

\[
\begin{bmatrix}
    y_{\text{train}} \\
    d(x_s)
\end{bmatrix} \sim \mathcal{MN}\left(0, \begin{bmatrix}
    K(x_{\text{train}}, x_{\text{train}}) & K(x_{\text{train}}, x_s) \\
    K(x_s, x_{\text{train}}) & \kappa(x_s, x_s)
\end{bmatrix}, \Omega\right),
\]
where $K(x_{[i]}, x_{[j]}) \in \mathbb{R}^{n \times n}$ with $[K(x_{[i]}, x_{[j]})]_{i,j} = \kappa(x_{i}, x_{j})$, and $K(x_{s}, x_{[i]}) \in \mathbb{R}^{1 \times N}$ with $[K(x_{s}, x_{[i]})]_{i} = \kappa(x_{s}, x_{i})$. Thus, we can compute the posterior distribution as follows:

$$
vec(d(x_{i})) \sim \mathcal{N}(\hat{M}, \hat{\Sigma} \otimes \hat{\Omega})
$$

\begin{align}
\hat{M} &= K(X, X)^{T}K(X, X)^{-1}Y \\
\hat{\Sigma} &= \kappa(X, X) - K(X, X)^{T}K(X, X)^{-1}K(X, X) \\
\hat{\Omega} &= \Omega
\end{align}

(8)

This allows us to estimate our unknown dynamics and their possibly correlated uncertainties.

IV. ROBUST MULTI-AGENT CBF

In this section, we first show that under certain assumptions, (5) is a multi-agent CBF for our discrete-time nonlinear dynamics. Then, given bounds on the uncertainty in each agents’ dynamics, we incorporate robustness to these uncertainties into the CBF while maintaining the computational efficiency of a quadratic program.

Extending Multi-Agent CBF to Discrete-Time, Nonlinear Systems: The multi-agent CBF introduced in [6] was originally designed for continuous-time linear systems. However, we prove that under proper assumption, $h(x)$ defined in (5) is a discrete-time CBF for the discrete-time nonlinear system (1)/(2). The tradeoff is the additional conservativeness in $a_{\text{max}}$ introduced by the following assumption. Intuitively, this assumption ensures that the robot can accelerate in any direction relative to the other agents, as proved in Lemma 1.

Assumption 1: Assume that for all $x \in \mathcal{C}$, $g_{i}(x)$ is invertible and

$$
\sigma_{\min}(g_{i}(x))u_{\text{max}} < 1,
$$

where $f_{i}(x) + d_{i}(x) - f_{i}^{(i)}(x) - d_{i}^{(i)}(x) - \Delta v_{i}$ and $\sigma_{\min}(g_{i}(x))$ is the minimum singular value of $g_{i}(x)$.

Lemma 1: Under Assumption 1, the expression (5), defining set $\mathcal{C}$, represents a discrete-time CBF for system (1), with

$$
a_{\text{max}} = \min_{x} \left[ \sigma_{\min}(g_{i}(x))u_{\text{max}} - \|\beta_{i}(x)\| \right] > 0.
$$

(9)

Proof: First, we must show that set $\mathcal{C}$ defined by expression (5) is control invariant for the dynamics (1), given that the robot has acceleration authority in any direction of at least $a_{\text{max}}$ for all $x \in \mathcal{C}$. For this, we rely on the same proof structure in [6], with the main difference being that we have discrete-time (rather than continuous-time) dynamics. Let $\Delta v_{(x_{i})}$ denote the component of velocity $v(x_{i})$ in the direction of collision.

$$
\Delta v_{(x_{i})} = \frac{\Delta p^{T} \Delta v}{\|\Delta p\|}
$$

(10)

We know that collision can be avoided if we can match the other agent’s velocity (i.e. $\Delta v = 0$) by the time we reach them. If we assume that we can accelerate at least $a_{\text{max}}$ in any direction, we are guaranteed that we can achieve $\Delta v = 0$ within time $T_{c} = \frac{\Delta v(x_{i})}{a_{\text{max}}}$. In our discrete-time formulation, the following condition implies collision avoidance:

$$
\Delta v_{(x_{i})}T_{c} + \|\Delta p\| \geq D_{s},
$$

\begin{align}
\left( \frac{\Delta p^{T} \Delta v}{\|\Delta p\|} \right)^{2} &\leq a_{\text{max}}(\|\Delta p\| - D_{s}).
\end{align}

Note that this constraint is only active when two agents are moving closer to each other ($\Delta v < 0$), and no constraint is needed when two agents are moving away from each other ($\Delta v \geq 0$). Therefore, collision can always be avoided under the following condition,

$$
- \frac{\Delta p^{T} \Delta v}{\|\Delta p\|} \leq \sqrt{a_{\text{max}}(\|\Delta p\| - D_{s})}.
$$

Based on our geometric argument, we know that the set $\mathcal{C} := \{ x \in \mathbb{R}^{n} : h(x) \geq 0 \}$ defined by $h := \frac{\Delta p^{T} \Delta v}{\|\Delta p\|} + \sqrt{a_{\text{max}}(\|\Delta p\| - D_{s})}$ is control invariant. This implies that $h(x)$ is a discrete-time CBF [27], given that the robot can accelerate by at least $a_{\text{max}}$ in any direction.

Therefore, our second step is to show that for all $x \in \mathcal{C}$ and any unit vector $\hat{e}$, it holds that sup$_{u \in \mathcal{U}} \| (v_{t+1}(x, u) - v_{t}(i)(x)) - (v_{t} - v_{t}(i))^{T} \hat{e} \| \geq a_{\text{max}} > 0$.

$$
\sup_{u \in \mathcal{U}} \| (v_{t+1}(x, u) - v_{t}(i)(x)) - (v_{t} - v_{t}(i))^{T} \hat{e} \|
$$

$$
= \sup_{u \in \mathcal{U}} \| (\beta_{i}(x) + g_{i}(x)u)^{T} \hat{e} \|
$$

$$
= \sup_{u \in \mathcal{U}} \| \hat{e}^{T} \beta_{i}(x) + \hat{e}^{T} g_{i}(x)u \|
$$

$$
\geq \sup_{u \in \mathcal{U}} \| \hat{e}^{T} g_{i}(x)u \| - \| \hat{e}^{T} \beta_{i}(x) \|
$$

$$
\geq \sigma_{\min}(g_{i}(x))u_{\text{max}} - \| \beta_{i}(x) \|
$$

$$
\geq \min_{x} \left[ \sigma_{\min}(g_{i}(x))u_{\text{max}} - \| \beta_{i}(x) \| \right] = a_{\text{max}} > 0,
$$

where the last inequality follows directly from Assumption 1. Therefore, we are guaranteed that the robot can accelerate by at least $a_{\text{max}} > 0$ in any direction. Combined with the first part of the proof, this shows that the set $\mathcal{C}$ defined by (5) is a discrete-time CBF.

Incorporating Robustness into CBF: While uncertainty in robot/environmental dynamics can be directly incorporated into the Control Barrier Condition (CBC) for simple systems/constraints (e.g. linear CBFS) [22], this is not the case for the multi-agent CBF with discrete-time dynamics. Unfortunately, uncertainty cannot be directly incorporated into the CBC while maintaining a quadratic program.

Consider our CBF (5) and the dynamics defined in (1) and (2). Based on these, we can compute the following CBC:
\[
C{\mathcal{B}}(x_t, u_t) = \left\{ \frac{f_p(x_t) + g_p(x_t)u_t + d_p(x_t) - f_b(x_t) - d_b(x_t)}{\|f_p(x_t) + g_p(x_t)u_t + d_p(x_t) - f_b(x_t) - d_b(x_t)\|} \right\},
\]

\[f_b(x_t) + g_b(x_t)u_t + d_b(x_t) - f_b(x_t) - d_b(x_t) + \sqrt{a_{\max}(\|f_p(x_t) + g_p(x_t)u_t + d_p(x_t) - f_b(x_t) - d_b(x_t)\| - D_x)} + (\eta - 1)\sqrt{a_{\max}(\|\Delta p\| - D_y) + (\eta - 1)\frac{\|\Delta p\|}{\|\Delta p\|}}.\]

(14)

If we can (a) determine bounds on the dynamic uncertainties, \(d\), in (1) and (2), and (b) compute control actions that satisfy \(C{\mathcal{B}}(x, u) \geq 0\) in an online fashion, then we can obtain robust safety guarantees utilizing the multi-agent CBF. Ideally, we could incorporate (14) into an efficiently solvable program as follows.

\[
u = \arg\min_{u \in \mathcal{U}} \|u - u_{\text{des}}\|_2 \quad \text{s.t.} \quad \max_{d(x_t)} C{\mathcal{B}}(x_t, u, d_t) \geq 0 \quad \text{where} \quad d(x_t) \in \mathcal{D} \quad \|u\|_2 \leq u_{\max}.\]

(15)

However, the CBF constraint (14) in (15) is clearly not linear (nor convex). Therefore, the resulting program is non-convex and cannot be solved at high frequency for adequate safety assurances. However, recall that our system has relative degree 2, which allows us to derive the following bound.

\[
C{\mathcal{B}}(x_t, u_t, d_t) \geq k_c(x_t) - H_1(x_t) - H_2(x_t)u_t + \frac{1}{2}H_3(x_t)u_t + H_3(x_t)u_t, \]

(16)

where the definitions of the terms \((k_c, H_1, H_2, H_3)\) are given in (27) in the Appendix. We also move the derivation of the bound (16) to the Appendix due to space limitations.

The following lemma allows us to utilize CBF bound (16) to obtain safety guarantees under polytopic uncertainties.

**Lemma 2:** Suppose the uncertainty in our dynamics \(d\) is bounded in the polytope \(\{d \in \mathbb{R}^n \mid Gd \leq g\}\). Then, the action, \(u\), obtained from solving the following optimization problem (17) robustly satisfies the CBF condition (14) (i.e. renders the set \(\mathcal{C}\) forward invariant).

\[
\min_{u \in \mathcal{U}, \xi \in \mathbb{R}^n} \|u - u_{\text{des}}\|_2 \quad \text{s.t.} \quad H_1(x_t)u + \xi g \leq k_c(x_t) \quad H_1(x_t)u + u^T H_2(x_t) = \xi G \quad \xi \geq 0 \quad \|u\|_2 \leq u_{\max} \quad \text{actuation limits},\]

(17)

**Proof:** The robust optimization problem (17) can be equivalently represented by the following optimization problem (i.e. (17) is the dual to (18) with no duality gap [39].

\[
\min_{u \in \mathcal{U}, \xi \in \mathbb{R}^n} \|u - u_{\text{des}}\|_2 \quad \text{s.t.} \quad \forall d \in \{d \in \mathbb{R}^n \mid Gd \leq g\} \quad H_1(x_t)u + u^T H_2(x_t) + H_3(x_t)u \leq k_c(x_t) \quad \|u\|_2 \leq u_{\max} \quad \text{actuation limits},\]

(18)

where \(u\) is the decision vector, \(d\) is the uncertainty variable, and \(\{d \in \mathbb{R}^n \mid Gd \leq g\}\) is the uncertainty bound. If the inequality in (18) is satisfied such that \(H_1d + u^T H_2d + H_3u \leq k_c\), then it follows directly from (16) that the CBF condition is satisfied for all \(d\) in our polytopic uncertainty set. Therefore, the set \(\mathcal{C}\) is rendered forward invariant.

This lemma shows that if we bound \(d(x_t)\) in a polytope, we can transform our robust multi-agent CBF (15) into a quadratic program (17), which gives us a computationally efficient way to provide robust guarantees of safety under robot and environment uncertainties. Hence, in the following section, we examine the problem of learning accurate polytopic bounds on the uncertainty \(d\) in an online fashion.

**V. LEARNING UNCERTAINTY BOUNDS**

In this section, our goal is to learn accurate confidence supports for the uncertainties \(d\) and \(d^{(i)}\) (for all agents \(i\)) in an online manner, which will allow us to guarantee safety with high probability. To this end, we utilize Matrix-Variate Gaussian Processes which provide multivariate Gaussian distributions over the uncertainties, \(d\) and \(d^{(i)}\).

Using (1), we have \(d(x_t) = x_{t+1} - f(x_t) - g(x_t)u_t\). A similar relation holds based on (2) for other agents. Thus, given a sequence of measurements \((x_t, u_t, x_{t+1})\) over a horizon \(T\), we compute the uncertain variable, \(d_t = d_{t-1}, \ldots, d_{t-1}\) over that horizon. Then, we infer a distribution over the query point, \(d_t\) (i.e. next time point), as described in Equation (8).

**Learning Kernel Parameters:** Direct application of the MVG (8) to our multi-agent setup will be problematic without first training the MVG model hyperparameters. This is easy to see by noting that the covariance, \(\Sigma(X) \otimes \Omega\), does not depend on the observed values, \(Y\). Furthermore, the coupling between uncertainties, captured by \(\Omega\), is completely independent of our online measurements. Instead, much of the uncertainty prediction is baked into the kernel parameters, \(\kappa(l, \sigma^2)\), and matrix \(\Omega\). Thus, to obtain accurate estimates of \(d\), we must learn MVG model parameters offline from data. In other words, some agents might behave predictably and others might behave more erratically, and hyperparameter optimization is necessary to capture these uncertainty profiles in our Bayesian inference.

Based on the probability density function (6), we obtain the negative log-likelihood of a given set of training data \(X\),

\[
L(X, Y; K, \Omega) = -\ln p(X, Y; K, \Omega) = \frac{nm}{2} \ln(2\pi) + \frac{m}{2} \ln |K| + \frac{n}{2} \ln |\Omega| + \frac{1}{2} \text{tr}[(K)^{-1}Y\Omega^{-1}Y^T] \quad (19)
\]
which we optimize (over $\Sigma, \Omega$) using Stochastic Gradient Descent (see hyperparameter optimization in Fig. 1) [40]. We run the optimization several times with different initializations to avoid local optima. The gradient update expressions are shown in Equation (20) below. Note that we use projected gradient updates for $\Omega$, in order to enforce the condition that $\Omega$ must be positive definite.

$$
\frac{dL}{dl} = \frac{d}{2} \text{tr} \left( K^{-1} \frac{dK}{dl} \right) + \frac{1}{2} \text{tr} \left( -K^{-1} \frac{dK}{dl} K^{-1} Y \Omega^{-1} Y^T \right)
$$

$$
\frac{dL}{d\sigma} = \frac{d}{2} \text{tr} \left( K^{-1} \frac{dK}{d\sigma} \right) + \frac{1}{2} \text{tr} \left( -K^{-1} \frac{dK}{d\sigma} K^{-1} Y \Omega^{-1} Y^T \right)
$$

$$
\frac{dL}{d\Omega} = \frac{n}{2} \Omega^{-1} - \frac{1}{2} \Omega^{-1} Y K^{-1} Y \Omega^{-1}
$$

(20)

### Converting Gaussian Process Uncertainty to Polytopic Bound

After learning the kernel parameters, we can obtain the mean, $\mu_d = \bar{M}$, and variance, $\Sigma_d = \Sigma \otimes \Omega$, from data observed online based on the multivariate Gaussian Process (8). Then, the uncertainties should follow the distribution,

$$(d - \mu_d)^T \Sigma_d^{-1} (d - \mu_d) \sim \chi^2_N,$$

(21)

where $\chi^2_N$ represents the chi-squared distribution with $N$ degrees of freedom (equal to dimension of $d$). This allows us to obtain the confidence support,

$$(d - \mu_d)^T \Sigma_d^{-1} (d - \mu_d) \leq k_{\delta} \text{ with probability } 1 - \delta.$$  \hfill (22)

However, this set defines an ellipsoid over $d$ rather than a polytope, which we require for the robust optimization. To obtain a polytope, we compute the minimum bounding box surrounding this ellipsoid.

**Lemma 3:** Suppose our robot/environment uncertainty can be described by our MVG model (described by the distribution (21)). With probability $1 - \delta$, the following polytopic bound on the uncertainty $d$ holds:

$$-\sqrt{k_{\delta} \lambda_i} + v_i^T \mu_d \leq u_i^T d \leq \sqrt{k_{\delta} \lambda_i} + v_i^T \mu_d,$$

(23)

where $v_i$ and $\lambda_i$ represent the eigenvectors and eigenvalues of $\Sigma_d$, respectively.

**Proof:** Since $\Sigma_d$ is a positive symmetric covariance matrix, the eigendecomposition of $\Sigma_d$ exists, and we have $\Sigma_d = \Psi \Lambda \Psi^T$, where $\Lambda$ is a diagonal matrix containing positive eigenvalues of $\Sigma_d$ and $\Psi$ is the orthogonal eigenvector matrix. Thus, (22) can be rewritten as follows:

$$[\Psi^T (d - \mu_d)]^T \Lambda^{-1} [\Psi^T (d - \mu_d)] \leq k_{\delta} \text{ w.p. } 1 - \delta.$$  \hfill (24)

We can bound the left hand side:

$$[\Psi^T (d - \mu_d)]^T \Lambda^{-1} [\Psi^T (d - \mu_d)] = \sum_i [v_i^T (d - \mu_d)]^T \lambda_i^{-1} [v_i^T (d - \mu_d)] \geq [v_i^T (d - \mu_d)]^T \lambda_i^{-1} [v_i^T (d - \mu_d)] \forall i = 1, \ldots, N$$

(25)

where $v_i$ represent the eigenvectors of $\Sigma_d$ contained in $\Psi$, and $\lambda_i$ are the eigenvalues of $\Sigma_d$ contained in $\Lambda$. We can then conclude that with probability $1 - \delta$, the following relations hold giving us our polytopic bound,

$$-\sqrt{k_{\delta} \lambda_i} + v_i^T \mu_d \leq u_i^T d \leq \sqrt{k_{\delta} \lambda_i} + v_i^T \mu_d \text{ for } i = 1, \ldots, N$$

(26)

**High-Confidence Safety Guarantee:** Combining the uncertainty bound on $d$ with our result from Sec. IV leads us to the main result, summarized in the following Theorem.

**Theorem 4:** Using the polytopic bounds (23), the control action obtained from the quadratic program (17) guarantees robust safety (i.e. collision avoidance between agents) with probability at least $1 - \delta$.

**Proof:** We can represent (23) in the form $\{Gd \leq g\}$; therefore, with probability $1 - \delta$, the uncertainty $d$ is contained in the set $\{d \in \mathbb{R}^n | Gd \leq g\}$ (by Lemma 3). From Lemma 2 and Equation (16), if we solve the quadratic program (17), we are guaranteed that $CBC \{x, u, d\} \geq \kappa - H_1 d - u^T H_2 d - H_3 u \geq 0$ for all $d \in \{d \in \mathbb{R}^n | Gd \leq g\}$. Therefore, the CBF condition is satisfied with probability $1 - \delta$, so safety is guaranteed with probability at least $1 - \delta$ (by Definition 1 and the forward invariance property of CBFs [26]).

### VI. RESULTS

#### A. Navigation in Unstructured Environment

We test our algorithm in a simulated multi-agent environment in which our robot, with nonlinear dynamics satisfying Assumption 1, navigates from a start to goal position while avoiding collisions, in the presence of a random number of other agents (3-12 agents). Each of the other agents has a randomized (unknown) goal that they try to reach. Approximately half of them blindly travel from their start to goal position without accounting for others, while the other half exhibit some collision avoidance behavior through their own control barrier functions (with random CBF parameters). An example simulation instance is shown in Fig. 2.

We simulate several instances of the other agents moving and interacting, and use this data for hyperparameter optimization of an MVG model as described in Sec. V. We then equip our robot with the robust CBF described in Sec. IV, using the optimized MVG for uncertainty prediction.

By running 1000 simulated tests in randomized environments, we show that the robust CBF avoids collision in 98.5% of cases (when we set $\delta = 0.05$ [cf. (22)]), performing much better than the nominal multi-agent CBF, which avoids collisions in 85.0% of cases. The simulation results are summarized in Table I.

Robustness must always come at the cost of performance (e.g. we can reach the goal faster if we do not care about collisions). To investigate the conservativeness of our
TABLE I: Performance statistics for the robust vs nominal multi-agent CBF across 1000 randomized trials. For fair comparison, the robust and nominal CBFS were tested in the same randomized 1000 trials. **Collision Rate:** Percentage of trials that ended in collision. **Distance to Collision:** For trials without collision, the robot’s margin from collision. The closer the robust CBF is to the nominal CBF, the less conservativeness is introduced by the uncertainty prediction.

<table>
<thead>
<tr>
<th></th>
<th>Robust Multi-Agent CBF</th>
<th>Nominal Multi-Agent CBF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collision Rate</td>
<td>1.5%</td>
<td>15.0%</td>
</tr>
<tr>
<td>Distance to Collision</td>
<td>7.4 ± 2.3</td>
<td>7.3 ± 2.1</td>
</tr>
</tbody>
</table>

The code for implementing the robust multi-agent CBF in our simulated environment can be found at [https://github.com/rcheng805/robust_cbf](https://github.com/rcheng805/robust_cbf). A video of the simulations can be found at [https://youtu.be/hXg5kZO86Lw](https://youtu.be/hXg5kZO86Lw).

VII. CONCLUSION

Robot navigation in unstructured environments with humans must be safe, but such environments are fraught with uncertainty due to the unpredictability of agents. In this work, we have introduced a robust multi-agent control barrier formulation, which guarantees safety with high probability in the presence of multiple uncontrolled, uncertain agents. We learn uncertainties online for the agents in the environment using Matrix-Variate Gaussian Processes, and design our CBF to be robust to the learned uncertainties.

Future work will look at learning and designing safe controllers for a larger class of uncertainties, unencumbered by our Matrix Variate GP. Particularly, many agents in the real world exhibit multi-modal uncertainties, and it will be important for us to design safe, robust controllers for such uncertainties.

REFERENCES


Here we define the terms used in the lower bound of the Control Barrier Condition (CBC) in (16):

\[
H_1(1 \times P) = \left[ - \frac{f_v(x) - f_h^p(x)}{\| f_p(x) - f_h^p(x) \| - \xi_p(x) - \xi_p^h(x)} \right],
\]

\[
H_2(M \times P) = \begin{bmatrix} - \frac{g_v(x)}{\| f_p(x) - f_h^p(x) \| - \xi_p(x) - \xi_p^h(x)} & 0 \\ g_v(x) & \| f_p(x) - f_h^p(x) \| - \xi_p(x) - \xi_p^h(x) \end{bmatrix},
\]

\[
H_3(1 \times M) = \left[ - \frac{(f_p(x) - f_h^p(x))^T g_v(x)}{\| f_p(x) - f_h^p(x) \| + \xi_p(x) + \xi_p^h(x)} \right] + \sqrt{a_{\max}(\| f_p(x) - f_h^p(x) \| - \xi_p(x) - \xi_p^h(x) - D_s)} + \sqrt{(\eta - 1) a_{\max}(\| D_p \|) - \Delta_p^T \Delta_p} - \xi_p(x) \xi_p^h(x) + \xi_p^h(x) \xi_p^h(x) + \xi_p(x) \xi_p^h(x)
\]

(27)
In this section, we prove that the lower bound defined in (16) holds. We begin by expanding out the full CBC condition in (14), using our assumption of a relative degree 2 system, we reach the following expression:

\[
CBC(x, u, d) = \frac{(f_p(x) - f_p^h(x))^T (f_u(x) - f_u^h(x))}{\|f_p(x) + d_p(x) - f_p^h(x) - d_p^h(x)\|} + \sqrt{a_{\text{max}}(\|f_p(x) + d_p(x) - f_p^h(x) - d_p^h(x)\| - D_x)} + \\
(\gamma - 1) \sqrt{a_{\text{max}}(\|\Delta p\| - D_x)} + (\gamma - 1) \frac{\Delta p^T \Delta u}{\|\Delta p\|} + \left[ \frac{(f_p(x) - f_p^h(x))^T g_v(x)}{\|f_p(x) + d_p(x) - f_p^h(x) - d_p^h(x)\|}, \frac{f_v(x) - f_v^h(x)}{\|f_p(x) + d_p(x) - f_p^h(x) - d_p^h(x)\|}\right] + \\
\left[ u_R^T \left[ \frac{g_v(x)}{\|f_p(x) + d_p(x) - f_p^h(x) - d_p^h(x)\|}, 0, \frac{-g_v(x)}{\|f_p(x) + d_p(x) - f_p^h(x) - d_p^h(x)\|}\right], 0 \right] \times \\
\left[ d_p, d_v, d_p^h, d_v^h \right]^T + \left[ \frac{1}{\|f_p(x) + d_p(x) - f_p^h(x) - d_p^h(x)\|}, \frac{-1}{\|f_p(x) + d_p(x) - f_p^h(x) - d_p^h(x)\|}\right] \times \\
\left[ u_R, d_p, d_v, d_p^h, d_v^h \right]^T.
\]

By bounding the positional uncertainty terms \(\|d_p(x)\| \leq \xi_p(x)\) and \(\|d_p^h(x)\| \leq \xi_p^h(x)\), we obtain a lower bound on \(CBC(x, u, d)\):

\[
CBC(x) \geq \min \left( \frac{(f_p(x) - f_p^h(x))^T (f_u(x) - f_u^h(x))}{\|f_p(x) - f_p^h(x)\|} + \sqrt{a_{\text{max}}(\|f_p(x) - f_p^h(x)\| - \xi_p(x) - \xi_p^h(x) - D_x)} + \\
(\gamma - 1) \sqrt{a_{\text{max}}(\|\Delta p\| - D_x)} + (\gamma - 1) \frac{\Delta p^T \Delta u}{\|\Delta p\|} + \left[ \frac{(f_p(x) - f_p^h(x))^T g_v(x)}{\|f_p(x) - f_p^h(x)\| + \xi_p(x) + \xi_p^h(x)}, \frac{f_v(x) - f_v^h(x)}{\|f_p(x) - f_p^h(x)\| - \xi_p(x) - \xi_p^h(x)}\right] + \\
\left[ u_R^T \left[ \frac{g_v(x)}{\|f_p(x) - f_p^h(x)\| - \xi_p(x) - \xi_p^h(x)}, 0, \frac{-g_v(x)}{\|f_p(x) - f_p^h(x)\| - \xi_p(x) - \xi_p^h(x)}\right], 0 \right] \times \\
\left[ d_p, d_v, d_p^h, d_v^h \right]^T - \left[ \frac{\xi_p(x) \xi_p^h(x)}{\|f_p(x) - f_p^h(x)\| - \xi_p(x) - \xi_p^h(x)}, \frac{\xi_p(x) \xi_p^h(x)}{\|f_p(x) - f_p^h(x)\| - \xi_p(x) - \xi_p^h(x)}\right] - \\
\frac{\xi_v(x) \xi_p^h(x)}{\|f_p(x) - f_p^h(x)\| - \xi_p(x) - \xi_p^h(x)} - \frac{\xi_v(x) \xi_p^h(x)}{\|f_p(x) - f_p^h(x)\| - \xi_p(x) - \xi_p^h(x)}
\right)
\]

Grouping the terms, this can be written in simplified form using the parameters defined in Appendix I:

\[
CBC(x, u, d) \geq k_c(x) - H_1(x) d - u^T H_2(x) d - H_3(x) u
\]