Expanding Safe Sets with Learning-Based Barrier Functions and Reachability

Trevor Matthews, Mahesh Kumar A.R., Azra Begzadić, Sander Tonkens, Zhirui Dai,
Nikolay Atanasov, Jorge Cortés, and Sylvia Herbert

***Abstract*— This paper presents GP-CBF, a learning-based framework for synthesizing and continuously adapting a robust control barrier value function for safe robot navigation in unknown environments. The approach first learns a smooth, conservative estimate of the signed distance function from noisy LiDAR data using Gaussian process regression and a Bayesian Hilbert map in real time. This learned signed distance function serves as a candidate control barrier function, which is then formally refined using Hamilton-Jacobi reachability analysis to generate a valid control barrier value function. The resulting value function is updated online as new sensor measurements are added, enabling progressive expansion of the safe set while maintaining probabilistic safety guarantees. The efficacy of our approach is validated through both simulation and hardware experiments on a mobile robot traversing cluttered environments, ensuring safety and outperforming existing baselines.**

I. Introduction

As autonomous systems become increasingly integrated into daily life, from self-driving cars to robotic assistants, ensuring their safe operation is critical. Formal safety methods, such as control barrier functions (CBFs) and Hamilton-Jacobi reachability (HJR), have emerged as powerful tools for enforcing safety. However, these guarantees rely on accurate obstacle boundaries. In many robotic applications, obstacle boundaries are not known a priori and must instead be reconstructed from on-board sensor measurements. Since CBF and HJR formulations rely on continuous scalar values to represent safe sets, the inaccuracies inherent in real-time sensor reconstruction can introduce boundary errors that invalidate the system's safety certificates.

Signed distance functions. A common environment representation in robotic applications is the signed distance function (SDF), which encodes the distance to the nearest obstacle. Several methods have been proposed to construct SDFs from sensors such as LiDAR or depth cameras, including volumetric fusion methods [1]–[3] and neural network-based implicit representations [4]–[6]. However, these methods typically do not quantify uncertainty in the reconstructed obstacle boundary. To address sensing uncertainty, Gaussian Process (GP)-based methods [7]–[9] have been used to learn SDFs while providing uncertainty bounds, yet their computational cost grows rapidly with environment scale.

Control barrier functions. CBFs enforce safety by ensuring the system remains within a safe set defined by a constraint function. However, constructing CBFs is particularly challenging for systems with complex and high relative degree under control input bounds and disturbances. As a result, learning-based approaches have been proposed to approximate CBFs, most notably using neural networks [10]–[12]. Beyond NNs, efforts have been made to learn CBFs from SDFs using GPs, although these methods often impose restrictive assumptions on the system dynamics [13]. Uncertainty in sensor measurements have also been addressed through distributionally robust formulations that construct Wasserstein ambiguity sets to enforce probabilistic CBF constraints [14].

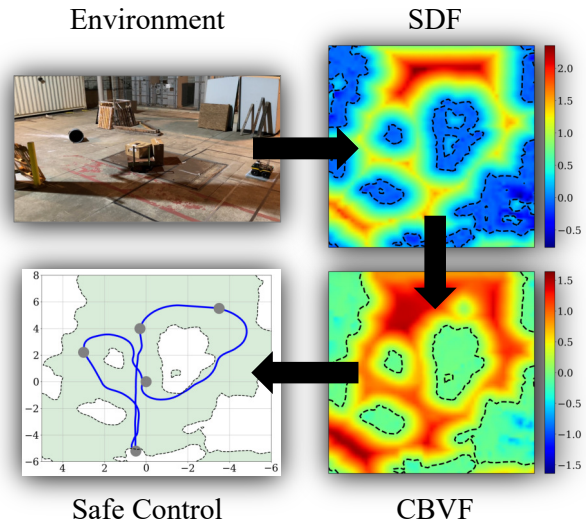

Fig. 1: Overview of the proposed learning-based safety framework. Noisy sensor measurements are used to learn a smooth signed distance function using a Bayesian Hilbert Map and Gaussian Process regression. The resulting learned signed distance function serves as a control barrier function candidate, which is refined via reachability analysis into a valid control barrier value function. The resulting value function is then used in an online safety filter to enforce safety constraints on the control input at each time step.

Hamilton-Jacobi Reachability (HJR) Analysis. HJR analysis offers a principled framework for ensuring the safety of the system under control input bounds and disturbances. Formulated as an optimal control problem, one can characterize the set of states from which failure can be avoided, yielding a value function that provides rigorous safety guarantees for general nonlinear systems. Despite its strengths, this approach encounters two main challenges. First, grid-based dynamic programming solvers suffer from the curse of dimensionality, restricting use to systems with fewer than six dimensions. Second, HJR is commonly implemented as a least-restrictive safety filter, meaning the optimal control is applied only near the safety boundary, which can induce chattering or overly cautious behavior.

Bridging the gap. An initial step to unify the HJR and CBF paradigms is the control barrier value functions (CBVFs) [15], which leverages HJR-like computations to construct a valid CBF. This combines the constructiveness of HJR with the efficient online enforcement of CBFs. However, this direct synthesis inherits the high computational cost

of HJR analysis, limiting real-time deployment. To address this, the refineCBF framework [16] uses a candidate value function as a "warm-start" for the formal refinement, converging rapidly to a provably safe value function. This was extended [17] to adapt to changes in the perceived obstacle boundary in-the-loop. However, this method assumes perfect sensor information and thus may result in safety violations under noisy or incomplete measurements.

Contributions. This paper presents a framework for computing and continuously adapting a measurement-robust CBF from online LiDAR observations. A measurement-robust smoothed SDF is first constructed using the approaches of [8], [13], which serves as a candidate CBF. This candidate is then refined using refineCBF [16] to account for system dynamics and provide probabilistic safety assurances under imperfect sensing. Once initialized, refineCBF runs in-the-loop as the robot navigates unknown environments, maintaining provable safety assurances during navigation.

Specifically, we make the following contributions:

- We utilize a smoothing technique to process on-board sensor measurements of the environment, using GP regression and Bayesian Hilbert Maps (BHM), to learn a smooth SDF with probabilistic error bounds. Moreover, we show that the safe set obtained from the learned SDF is contained in the true safe set with high probability.
- We introduce an in-the-loop refinement of the smooth SDF via HJ-reachability, resulting in a valid CBF that accounts for general control-affine robot dynamics and provides a technical analysis establishing probabilistic safety guarantees for the resulting filter, while allowing for an expanding safe set.
- We demonstrate this framework in both simulation and hardware using a limited-range LiDAR sensor and provide an extensive comparison to baselines and under different levels of sensor noise.

II. Background

Consider a control-affine system of the form

$$\dot{x} = f(x) + g(x)u, \tag{1}$$

where $x \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state and $u \in \mathcal{U} \subseteq \mathbb{R}^p$ is the control input. We assume a compact state space $\mathcal{X}$ and a convex and compact input set $\mathcal{U}$. We consider a time horizon $[t, 0]$. For each initial time $t \leq 0$, we denote the sets of admissible control signals by $\mathbb{U}(t) := \{\boldsymbol{u} : [t, 0] \to \mathcal{U} | \boldsymbol{u} \text{ is measurable}\}$. We assume the functions $f : \mathcal{X} \to \mathbb{R}^n$ and $g : \mathcal{X} \to \mathbb{R}^{n\times p}$ are Lipschitz and bounded. Under these assumptions, given an initial time t, state x, and $\boldsymbol{u} \in \mathbb{U}(t)$, there exists a unique solution $\boldsymbol{x}^{\boldsymbol{u}}_{x,t} : [t, 0] \to \mathbb{R}^n$ of (1) with initial condition $\boldsymbol{x}^{\boldsymbol{u}}_{x,t}(t) = x$. Throughout the paper, the argument t in $x(t)$ and $u(t)$ is omitted for brevity.

A. Hamilton-Jacobi Reachability

HJR is a framework that determines the set of initial states from which a system can reach a target set and/or avoid a failure set. In many robotic applications, we consider a constraint set $\mathcal{L}$, which represents the allowable states of the system (1). This set is described as a zero-superlevel set of a safety target function $\ell : \mathbb{R}^n \to \mathbb{R}$, i.e., $\mathcal{L} := \{x \in \mathcal{X} \mid \ell(x) \geq 0\}$. The objective of safe control is to ensure the trajectory stays within constraint set $\mathcal{L}$ for $\tau \in [t, 0]$.

For a given set $\mathcal{L}$, we are interested in computing the viability kernel $\mathcal{S}(t)$ defined as

$$\mathcal{S}(t) := \left\{x \in \mathcal{X} : \exists \boldsymbol{u} \in \mathbb{U}(t) \text{ s.t. } \forall \tau \in [t, 0],\ \boldsymbol{x}^{\boldsymbol{u}}_{x,t}(\tau) \in \mathcal{L}\right\}, \tag{2}$$

which represents the set of all initial states at time t in the set $\mathcal{L}$ from which an admissible control signal exists that ensures the system will remain in $\mathcal{L}$ over the entire time horizon.

The objective function for this problem is given as $J(x, t, \boldsymbol{u}) = \min_{\tau\in[t,0]} l(\boldsymbol{x}^{\boldsymbol{u}}_{x,t}(\tau))$, which represents the smallest value of l along the system trajectory over the time horizon. If this value falls below zero, the trajectory has entered the failure set. The associated value function $V(x, t) = \max_{\boldsymbol{u}\in\mathbb{U}} J(x, t, \boldsymbol{u})$ is the viscosity solution of the following Hamilton–Jacobi–Isaacs variational inequality

$$\begin{aligned} 0 &= \min\left\{\frac{\partial}{\partial t}V(x, t) + H(x, t),\ \ell(x) - V(x, t)\right\}, \\ V(x, 0) &= \ell(x), \end{aligned} \tag{3}$$

where the Hamiltonian $H : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is given by $H(x, t) = \max_{u\in\mathcal{U}} \nabla_x V(x, t)^\top \big(f(x) + g(x)u\big)$.

Given V, the viability kernel $\mathcal{S}(t)$ for the constraint set $\mathcal{L}$ is $\mathcal{S}(t) := \{x \in \mathbb{R}^n \mid V(x, t) \geq 0\}$. When the viability kernel $\mathcal{S}(t)$ is non-empty, the optimal safety controller $u^*(x, t)$ is given by

$$u^*(x, t) = \arg\max_u \nabla_x V(x, t)^\top \big(f(x) + g(x)u\big). \tag{4}$$

B. Control Barrier Functions

We start by formally defining the concept of CBF.

Definition 1 (CBF [18])**.** *Consider a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$ and define its superlevel set $\mathcal{C} := \{x \in \mathbb{R}^n \mid h(x) \geq 0\}$. The function h is a CBF of $\mathcal{C}$ for* (1) *if there exists an extended class $\mathcal{K}$ function α such that, for all $x \in \mathcal{C}$,*

$$\max_{u\in\mathcal{U}} \nabla_x h(x)^\top \big(f(x) + g(x)u\big) \geq -\alpha(h(x)). \tag{5}$$

Given a valid CBF h, the safety filter can minimally modify a nominal control law $u_{\text{nom}} : \mathcal{X} \to \mathbb{R}^p$ to maintain safety using the following quadratic program (QP)

$$\begin{aligned} u^*(x) = \arg\min_{u\in\mathbb{R}^p} &\|u_{\text{nom}}(x) - u\|_2^2 \\ \text{s.t. } &\nabla_x h(x)^\top \big(f(x) + g(x)u\big) + \alpha(h(x)) \geq 0. \end{aligned} \tag{6}$$

In practice, it is common to choose $\alpha(x) = \gamma x$, with $\gamma \geq 0$. The synthesis of a CBF for a nonlinear system is challenging, particularly when the system is subjected to bounded control inputs and disturbances. However, recent work has shown HJR can be used to constructively generate CBFs, resulting in control barrier-value functions (CBVFs) defined below.

Definition 2 (CBVF [19])**.** *A control barrier-value function $h_\gamma : \mathbb{R}^n \times (-\infty, 0] \to \mathbb{R}$ is given by*

$$h_\gamma(x, t) = \max_{u\in\mathcal{U}_{[t,0]}} \min_{s\in[t,0]} e^{\gamma(s-t)}\ell(\boldsymbol{x}^{\boldsymbol{u}}_{x,t}(s)), \tag{7}$$

for some $\gamma \in \mathbb{R}_{>0}$ *and* $\forall t \leq 0$*, with initial condition* $h_\gamma(x,0) = \ell(x)$.

The CBVF described in (7) can be constructed as in (3). As in [17], we set $\gamma = 0$, while using any $\gamma > 0$ online, such that (7) simplifies to the standard HJR problem. Once the value function h is defined, it can be integrated into a safety filter similar to (6) such that

$$u^*(x) = \arg\min_{u \in \mathbb{R}^p} \|u_{\text{nom}}(x) - u\|_2^2 \tag{8}$$

$$\text{s.t. } \frac{\partial}{\partial t} h(x,t) + \nabla_x h(x,t)^\top \big(f(x) + g(x)u\big) + \gamma h(x,t) \geq 0.$$

C. Gaussian Process Regression

GP regression is a non-parametric approach based on the assumption that any finite number of measurements $\{y(x^{(1)}), \ldots, y(x^{(N)})\}$, $N \in \mathbb{N}$, of an unknown function $f : \mathbb{R}^n \to \mathbb{R}$ at inputs $x \in \mathcal{X}$ follows a joint Gaussian distribution [20]. The GP $f(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot,\cdot))$ is fully defined by a prior mean $m : \mathcal{X} \to \mathbb{R}$ and a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$. By conditioning the prior on training points $\big(x^{(i)}, y^{(i)}\big)$ with $i = 1, \ldots, N$, where $y^{(i)} = f\left(x^{(i)}\right) + \epsilon_i$, $\epsilon_i \sim \mathcal{N}(0, \sigma_{on}^2)$, we obtain a posterior distribution that is again Gaussian, with mean and variance given by

$$\mu(x) = k(x)^\top (K + \sigma_{\text{on}}^2 I_N)^{-1} y, \tag{9a}$$

$$\sigma^2(x) = k(x,x) - k(x)^\top (K + \sigma_{\text{on}}^2 I_N)^{-1} k(x), \tag{9b}$$

with kernel vector $k(x) = k(x, x^{(i)})$, kernel matrix K through $K_{i,j} = k(x^{(i)}, x^{(j)})$, and $y = [y^{(1)} \ \cdots \ y^{(N)}]^\top$.

III. Problem Statement

Consider a robot with control-affine dynamics (1). We address the problem of controlling the robot to perform tasks while satisfying safety constraints, with a primary focus on obstacle avoidance. For this purpose, we consider an open obstacle set $\mathcal{O} \subset \mathcal{X}$, and its complement constraint set $\mathcal{C} = \mathcal{X} \setminus \mathcal{O}$. We consider the ground-truth SDF $d : \mathcal{X} \to \mathbb{R}$ of the set $\mathcal{O}$, which measures the signed Euclidean distance from a query point x to the boundary $\partial\mathcal{O}$, with sign indicating whether $x \in \mathcal{O}$, such that

$$d(x) = \begin{cases} \min_{y \in \partial\mathcal{O}} \|x - y\|_2 & \text{if } x \notin \mathcal{O}, \\ -\min_{y \in \partial\mathcal{O}} \|x - y\|_2 & \text{if } x \in \mathcal{O}. \end{cases} \tag{10}$$

We consider a robot equipped with a range sensor, such as LiDAR. The sensor provides discrete noisy boundary measurements for a subset of the obstacles $\mathcal{O}$ in the environment. We assume we obtain these samples during deployment at discrete sampling times. We want to ensure while obtaining new measurements of the environment, we retain safety as the robot navigates through the environment. Critically, this relies on growing the safe set from its initial estimate, thus expanding the perceived obstacle-free region. Based on the system description in (1), we address the following.

Problem 1. *Given robot with dynamics defined in* (1) *and a range sensor that provides boundary measurements, develop an adaptive safety-filtering framework that ensures safety in environments with unknown static obstacles. The framework should minimally modify nominal control inputs and iteratively reduce conservatism as the environment is mapped, while maintaining formal safety guarantees.*

IV. Methodology

We develop a method for synthesizing and updating a safety filter online, using noisy distance measurements collected continuously as the system operates over time. Since the true SDF d of the environment is unknown, our approach first learns a smooth and robust SDF representation from sensor data using a Bayesian Hilbert Map and GP regression. The resulting SDF, which serves as our initial CBF candidate, is then refined into a valid CBVF by accounting for the system dynamics and control input bounds. This refinement is achieved using HJR analysis, which treats the learned SDF as an efficient "warm-start" for a dynamic programming-based computation. The final result is a continually updated, data-driven CBVF that provides probabilistic safety guarantees and can be used in a real-time safety filter to ensure safety of the system. Crucially, this continuous learning process enables the progressive expansion of the learned safe set, allowing the system to operate less conservatively by refining its understanding of the true safety boundaries.

A. Learning a smooth SDF via BHMs & GP regression

SDFs represent distance to the boundary of obtacle space, and, hence, are a good choice for capturing geometric safety constraints. However, two key elements are required for their downstream use in a safety filter: 1) continuous differentiability (so that a safe control can be computed), and 2) probabilistic outputs (to measure uncertainty about the unknown environment). To achieve these properties, we learn a smooth surrogate function $\tilde{d}$ of the true SDF d using GP regression. In our online setting, the robot gathers and updates surface boundary measurements during operation rather than relying on a pre-collected dataset. To do this, we employ BHMs [21] to obtain a dataset of surface points by aggregating boundary measurements over time, inferring occupancy across the environment. Then, we run marching cubes [22] on the BHM to extract surface points p_i.

To train a GP SDF model on this data, we follow the Log-Gaussian Process Implicit Surface approach in [8], which leverages Varadhan's distance formula [23], to obtain an unsigned distance function. We consider a GP $s(\cdot) \sim \mathcal{GP}(0, k_s(\cdot,\cdot))$ with zero prior mean and a Mateŕn 3/2 kernel with kernel scale $\sqrt{3}/\lambda$. This kernel scale and shape are chosen to model the heat diffusion process underlying Varadhan's formula. Then, the unsigned distance $\tilde{s}(x)$ at a query point x is obtained by taking the logarithm of the GP inference, resulting in $\tilde{s}(x) = r(s(x)) = -\log(s(x))/\lambda$, which inverts the kernel's exponential "diffusion" to recover a linear metric distance. We denote $r(z) = -\log(z)/\lambda$ as the nonlinear transformation function. Based on this transformation, we construct our training dataset $\mathcal{D} = \{(p_i, 1)\}_{i=1}^{N_k}$ for $s(\cdot)$ by associating each surface point p_i with a target measurement of $y_i = 1$. This ensures that $r(1) = 0$, correctly mapping surface points to zero distance.

To obtain guarantees on the behavior of this function, we make the following assumption on the function s.

Assumption 1. *The function s lies in a Reproducing Kernel Hilbert Space (RKHS) associated with a kernel $k_s(\cdot,\cdot)$ such that s has a bounded RKHS norm, i.e., $\|s(x)\|_{\mathcal{H}_{k_s}} \leq B$.*

This assumption allows us to establish the following prediction error bound for GP regression.

Lemma 1 ([24])**.** *Under Assumption 1, for a given $\delta \in (0,1)$, there exists a constant $\beta > 0$ such that the following holds on a compact set $\mathcal{X}$*

$$\Pr\{\, |\mu_s(x) - s(x)| \leq \beta\sigma_s(x),\ \forall x \in \mathcal{X} \,\} \geq 1-\delta. \quad (11)$$

Determining the constant β in Lemma 1 typically relies on prior knowledge of the underlying function and the choice of kernel parameters [24]. While this aspect is important for practical implementation, a detailed treatment of selecting β lies beyond the scope of this paper. Since the nonlinear transformation $r(z) = -\log(z)/\lambda$ is monotonically decreasing, applying it to both sides of the inequality in (11) flips the signs. Hence, we have that

$$r(\mu_s(x) + \beta\sigma_s(x)) \leq \tilde{s}(x) \leq r(\mu_s(x) - \beta\sigma_s(x)) \quad (12)$$

holds with a probability of at least $1-\delta$. With the bounds on the unsigned distance established, we now address the sign estimation to obtain the SDF.

Note that so far, our distance estimate $\tilde{s}(x)$ is unsigned. To obtain the sign, we employ the previously mentioned BHM. Beyond using it to obtain the dataset of surface points $\mathcal{D}$, the BHM provides a probabilistic estimate $\tilde{b}(x)$ of the occupancy of each state $b(x)$. We denote $b(x) = 1$ if x is free, i.e., $x \in \mathcal{O}^C$, and $b(x) = -1$ if $x \in \mathcal{O}$. The estimate $\tilde{b}(x)$ has probability $\Pr(b(x) = 1) = q$ and $\Pr(b(x) = -1) = 1 - q$. Hence, our SDF estimation becomes:

$$\tilde{d}(x) = \tilde{b}(x)\tilde{s}(x). \quad (13)$$

Then, $\tilde{d}(x)$ has a mixture distribution with density:

$$p(\tilde{d}(x)) = \begin{cases} q p_{\tilde{s}}(\tilde{s}(x)) & \tilde{d}(x) \geq 0, \\ (1-q) p_{\tilde{s}}(\tilde{s}(x)) & \tilde{d}(x) < 0, \end{cases} \quad (14)$$

where $p_{\tilde{s}}(\tilde{s}(x))$ is the density of $\tilde{s}(x)$. This gives $\mathbb{E}[\tilde{d}(x)] = q\mathbb{E}[\tilde{s}(x)] + (1-q)\mathbb{E}[-\tilde{s}(x)] = (2q-1)\mathbb{E}[\tilde{s}(x)]$ and $\mathbb{V}[\tilde{d}(x)] = 4q(1-q)\mathbb{E}^2[\tilde{s}(x)] + \mathbb{V}[\tilde{s}(x)]$.

When x is in the free space, $q \to 1$; when x is in occupied space, $q \to 0$. Both cases give $\mathbb{E}[\tilde{d}] = \tilde{b}(x)\mathbb{E}[\tilde{s}(x)]$ and $\mathbb{V}[\tilde{d}(x)] = \mathbb{V}[\tilde{s}(x)]$. Assuming a well-trained BHM, q quickly transitions between 0 and 1 across the surface, hence, $\mathbb{V}[\tilde{d}(x)] \approx \mathbb{V}[\tilde{s}(x)]$, which means we can simply use a deterministic sign prediction with BHM, i.e., $b(x) = 1$ if $q \geq \tau$, or $b(x) = -1$ if $q < \tau$, where $\tau \in (0,1)$ is the decision boundary. While set to $\tau = 0.5$ by default, we can select a larger τ to ensure the resulting $\tilde{d}(x)$ is conservative, by requiring high probability τ for a state x to have $\tilde{d}(x) > 0$.

Since $\tilde{d}(x)$ has a mixture distribution, we utilize a deterministic sign prediction from the BHM to obtain a probabilistic pessimistic estimate of the SDF. Therefore, given the decision boundary $\tau \in (0,1)$, we define our conservative estimate $\underline{d}(x)$ as

$$\underline{d}(x) = \begin{cases} r(\mu_s(x) + \beta\sigma_s(x)), & \text{if } q \geq \tau, \\ -r(\max(\epsilon, \mu_s(x) - \beta\sigma_s(x))), & \text{if } q < \tau, \end{cases} \quad (15)$$

where $\epsilon > 0$ is a small constant to ensure the argument of the logarithm remains strictly positive in regions of high GP uncertainty. This piece-wise formulation guarantees that $\underline{d}(x) \leq d(x)$ for all x with a probability of at least $1-\delta$.

However, $r(z)$ distorts the variance. Therefore, in practice, we approximate $\mathbb{V}[\tilde{s}(x)]$ by a softmin fusion process as in [8]. With the aforementioned dataset of surface points, and associated variances, $\mathcal{D}$, we can write

$$\tilde{s}(x) \approx m(x, \{p_i\}_{i=1}^N) = \sum_i w_i z_i, \quad (16)$$

with $z_i = \|p_i - x\|_2$, $w_i = \frac{\exp(-\alpha z_i)}{\sum_i^N \exp(-\alpha z_i)}$, $\alpha > 0$, and m the softmin value. Then, we can calculate the approximation of $\mathbb{V}[\tilde{s}(x)]$ with the first-order approximation of (16):

$$\mathbb{V}[\tilde{s}(x)] \approx \sum_{i=1}^N \|\nabla_{p_i} m(x)\|_2^2 \, \sigma_i^2, \quad (17)$$

with $\mathbb{V}[\tilde{d}(x)] = \mathbb{V}[\tilde{s}(x)]$ as before.

B. CBVF construction

Since directly using the posterior mean of the SDF surrogate $\mu_{\hat{d}}$ may overestimate safety in uncertain regions, potentially leading to labeling unsafe states as safe, we consider the pessimistic estimate of the SDF $\underline{d}$ from Eq. (15). If the kernel parameters are suitably chosen, the resulting superlevel set of $\underline{d}$ is an inner approximation of the real safe set $\mathcal{C}$ as formalized in the following result.

Lemma 2. *Consider a prior $\mathcal{GP}(0, k_s(\cdot,\cdot))$ with kernel k_s. Given Assumption 1 and training dataset $\mathcal{D}$, the zero superlevel set of the GP-SDF $\underline{d}$, $\mathcal{C}_s = \{x \in \mathcal{X} : \underline{d}(x) \geq 0\}$, is included in $\mathcal{C}$ with a probability of at least $1-\delta$, i.e., $\mathbb{P}(\mathcal{C}_s \subseteq \mathcal{C}) \geq 1-\delta$.*

The proof of this result follows an analogous argument to [13, Lemma 5.5] and we omit it for space reasons. The resulting GP-SDF $\underline{d}$ can now be used as the constraint function in an HJR formulation to construct and refine a CBF that is valid with respect to the general control-affine dynamics (1) with bounded control inputs. refineCBF uses a discretized version of (3) for its update:

$$h^{(k+1)}(x) \leftarrow \min\left\{\ell_j(x), h^{(k)}(x) + \Delta^{(k)} L_f^* h^{(k)}(x)\right\}, \quad (18)$$

where $\ell_j(x) = \underline{d}_j$ and $\Delta^{(k)}$ the timestep.

At initialization, both the constraint function and the initial approximate CBF are set as the initial pessimistic SDF, i.e., $\ell_0(x) = h^{(0)}(x) = \underline{d}_0(x)$. At the initial iteration, our method runs (3) until convergence, i.e., $|h_0^{(k)}(x) - h_0^{(k-1)}(x)| < \varepsilon$ for $\varepsilon \in \mathbb{R}_{>0}$, resulting in the converged CBF $h_0^*(x)$. After initial convergence of h_0^* we update the obstacle function $\ell(x)$ through updates to the GP node and the CBF by applying (18) in parallel, as seen in Algorithm 1. The CBF update is based on the latest available $\ell(x)$.

Algorithm 1 Asynchronous GP-SDF and CBVF Update

Initialize d_0, set $\ell_0(x) \leftarrow \underline{d}_0(x)$, and $h_0(x) \leftarrow \underline{d}_0(x)$
$h_0^*(x) \leftarrow$ Solve (18) until convergence
$k \leftarrow 1$, $j \leftarrow 1$, $h^{(0)}(x) \leftarrow h_0^*(x)$
GP-SDF (runs continuously at rate $f_{\mathrm{GP}} = \frac{1}{\Delta_{\mathrm{GP}}}$)
loop
    Update measurements $\mathcal{D}_j = \mathcal{D}_{t_k - \Delta_{\mathrm{GP}}} \cup \cdots \cup \mathcal{D}_{t_k}$
    Update GP-SDF $d_j(x)$ given d_{j-1} and $\mathcal{D}_j$
    $j \leftarrow j + 1$
refineCBF (runs continuously at rate $f_{\mathrm{CBF}} = \frac{1}{\Delta_{\mathrm{CBF}}}$)
loop
    Query current constraint function $\ell_k(x) = \underline{d}_j(x)$
    $h^{(k+1)}(x) \leftarrow \min\{\ell_j(x),\ h^{(k)}(x) + \Delta L_f^* h^{(k)}(x)\}$
    $k \leftarrow k + 1$
Controller (runs continuously at rate f_{cntrl})
loop
    Query CBF $h = h^{(k)}(x)$
    Apply $u \leftarrow u^*(x)$ via the safety-filter QP (20)

The theoretical guarantee for this refinement is provided by the following result, which ensures convergence to a provably safe CBVF.

Theorem 1 (Convergence to a safe CBVF [17], Thm 1)**.** *If the value function* (18) *converges to $h^*(x)$ as $t \to -\infty$, then $h^*(x)$ is a safe CBVF for all $x \in \mathcal{H}^*$, its 0-superlevel set.*

However, the guarantees provided by Theorem 1 are difficult to apply directly in an online setting for two main reasons. First, Theorem 1 assumes a deterministic representation of the safe set $\mathcal{C}$. Second, its guarantees are only valid upon convergence, which would require running HJR until convergence upon each SDF estimate update.

In contrast to Theorem 1, we instead have an initial candidate CBF which is based on GP error bounds, providing only a probabilistic guarantee of set inclusion. Next, we provide the following result, which extends Theorem 1, and the deterministic refinement framework of [17], to include formal probabilistic guarantees under a learned, uncertain environment model. With this result, we can use a GP-SDF-based candidate CBF as an initial guess for a dynamic programming refinement framework, and obtain a probabilistically safe value function for our system as a result.

Theorem 2. *Let h_s be a learned candidate CBF function and let its zero-superlevel be $\mathcal{C}_s = \{x \in \mathcal{X} : h_s(x) \geq 0\}$ for system* (1)*. Assume that C_s provides a conservative approximation of the true, unknown constraint set $\mathcal{C}$, meaning it is a subset of the true set with high probability, i.e., $\mathbb{P}(\mathcal{C}_s \subseteq \mathcal{C}) \geq 1 - \delta$. If the value function initialized with the initial guess $h_0(x) = h_s(x)$ in* (18) *converges to a function h^* with zero-superlevel set $\mathcal{H}^*$, then*

1) *$\mathcal{H}^*$ is a control invariant set,*
2) *$\mathcal{H}^*$ is a subset of the true set $\mathcal{C}$ with high probability, i.e., $\mathbb{P}(\mathcal{H}^* \subseteq \mathcal{C}) \geq 1 - \delta$.*

Proof. Let h^* be the CBVF and $\mathcal{H}^* := \{x \in \mathcal{X} \mid h^*(x) \geq 0\}$ due to Theorem 1. By construction, h^* is the viscosity solution of the HJI (3) given the system description in (1) with convex and compact $\mathcal{U}$. For each $x \in \mathcal{H}^*$, the HJI implies the existence of an optimal safety controller u^*, such that $\nabla h^* \cdot f(x, u^*) \geq 0$. This control law guarantees that on the boundary, i.e., $h^*(x) = 0$, the vector field points inside. Hence trajectories starting in $\mathcal{H}^*$ do not leave the set under u^*. By Nagumo's theorem, the set $\mathcal{H}^* = \{x \mid h^*(x) \geq 0\}$ is forward invariant under the closed-loop dynamics (1), which proves the first claim. In order to prove the second claim, we recall the assumption on the inclusion of the sets, i.e., $\mathbb{P}(\mathcal{C}_s \subseteq \mathcal{C}) \geq 1 - \delta$. For $h(x,t)$ defined through (18) and any finite horizon $t < 0$, the refined set is defined as $\mathcal{H}_t = \{x \in \mathcal{X} : h(x,t) \geq 0\}$, which consists of all states from which there exists a control policy whose trajectory remains in $\mathcal{C}_s$ and terminates in $\mathcal{C}$. Since $\mathcal{C}_s \subseteq \mathcal{C}$ with probability of at least $1 - \delta$, the terminal state is in $\mathcal{C}$ with the same probability. Hence, $H_t \subseteq \mathcal{C}$ with probability of at least $1 - \delta$ for all $t < 0$. Hence, we have

$$\mathbb{P}(\mathcal{H}_t \subseteq \mathcal{C}) \geq \mathbb{P}(\mathcal{C}_s \subseteq \mathcal{C}) \geq 1 - \delta \tag{19}$$

for all $t < 0$. If h converges to h^* as $t \to -\infty$, the property extends to the infinite-horizon safe set $\mathcal{H}^*$, i.e., $\mathbb{P}(\mathcal{H}^* \subseteq \mathcal{C}) \geq 1 - \delta$, concluding the proof. □

Secondly, to address the issue of requiring convergence of (18), we introduce the following:

Lemma 3. *(Refining a safe CBVF will preserve safety [17, Lemma 4]) If there exists $t_1 \leq 0$ such that $h(x, t_1)$ from* (18) *is control invariant, $h(x,t)$ is control invariant for all $t < t_1 \leq 0$.*

Critically, this provides a guarantee of preserving safety while converging. By inspection of (18), Lemma 3 holds when we obtain further observations of the environment under the following assumption:

Assumption 2. *(Growing conservative safe set) We assume with each update of the GP $d(x)$ the safe set grows, in particular that we have $\ell_{j+1}(x) \geq \ell_j(x)$ for all $x \in \{x \mid \ell_{j+1}(x) \geq 0\}$, with $\ell_j(x) = \underline{d}_j(x)$, while remaining a subset of the true obstacle free region with high probability, i.e., $\mathbb{P}(\{x \mid \ell_j(x) \geq 0\} \subseteq \mathcal{C}) \geq 1 - \delta$ at every iteration j.*

Lemma 4 (Refining a safe probabilistic CBVF preserves probabilistic safety)**.** *Consider Alg. 1. Given Assp. 2, Alg. 1's value function is a probabilistically valid CBVF at every iteration as its 0-superlevel set $\mathcal{C}_s^{(k)}$ is a subset of the true set with high probability, i.e., $\mathbb{P}\left(\mathcal{C}_s^{(k)} \subseteq \mathcal{C}\right) \geq 1 - \delta$.*

Proof. Theorem 2 guarantees $h^{(0)}$(x) is control invariant and its 0-superlevel set is a subset of $\mathcal{C}$. Then, Lemma 3 assures $h^{(k)}$ is control invariant and satisfies the CBF constraint for all $x \in \mathcal{C}_s^{(k)} = \{x \mid h^{(k)} \geq 0\}$. Lastly, Eq. 3 guarantees $h^{(k+1)}(x) \leq \ell_j(x)$, thus $\mathcal{C}_s^{(k+1)} \subseteq \{x \mid \ell_j(x) \geq 0\}$, which by Asp. 2 is a subset of $\mathcal{C}$ with high probability. □

This is crucial for navigation in a priori unknown static environments, where we continuously improve our guess of the true SDF in the environment. This assumption can typically be satisfied easily by characterizing the GP-SDF such that unobserved areas of the state space have a negative

value, i.e., are considered unsafe. To improve notational clarity, we omit the iteration argument j.

This result extends the deterministic refinement framework to handle uncertainty, guaranteeing if an initial safe set lies within the true non-obstacle set with high probability, then the safe set characterized by the converged value function is included in the non-obstacle set with the same probability. Moreover, the safe set can be expanded upon new observations, enabling exploration of unknown environments.

C. Safe control synthesis

Assuming we remain stationary at the initial point until (18) converges, as in Alg. 1, any future CBVF $h^{(k)}$ for $k > 0$ characterizes a control invariant set (see Lemma 3). Therefore, the value function $h^{(k)}$ serves as a valid CBVF for the system (1). This function can be integrated into a real-time safety filter, which minimally modifies a desired nominal control law as given in (6) such that

$$u^*(x) = \arg\min_{u \in \mathbb{R}^p} \|u_{\text{nom}}(x) - u\|_2^2 \tag{20}$$
$$\text{s.t. } \nabla_x h^{(k)}(x)^\top \big(f(x) + g(x)u\big) + \alpha(h^{(k)}(x)) \geq 0,$$

The existence of a solution to this QP is crucial for guaranteeing safety as formalised in the following result.

Proposition 1. *Given the value function h_l for the control-affine system* (1)*, the optimization problem* (20) *is feasible for $x \in \{x \mid h^{(k)}(x) \geq 0\}$ whenever the gradient $\nabla_x h^{(k)}(x)$ exists. Furthermore, any solution u^* from* (20) *renders its 0-superlevel set forward invariant.*

Proof. The proof of this result follows an analogous argument to [15, Prop. 3] and we omit it for space reasons. □

V. Evaluation

Robot Dynamics. We evaluate the proposed method both in simulation and hardware[1] on a Clearpath Jackal robot with nonlinear dynamics and input bounds. The robot is modeled as a non-holonomic wheeled system with state $x = [p_x, p_y, \theta, v]^\top \in \mathbb{R}^2 \times [-\pi, \pi) \times [0, 1]$ and control input $u = [a, \omega]^\top \in \mathcal{U}$, governed by

$$\dot{p}_x = v\cos(\theta), \quad \dot{p}_y = v\sin(\theta), \quad \dot{\theta} = \omega, \quad \dot{v} = a. \tag{21}$$

In the state space, p_x and p_y denote the robot position in a 2D cartesian coordinate system, θ the orientation, and v the linear velocity. To reflect actuator limits, we constrain the inputs to $a \in [-0.5, 0.5]\,\text{m/s}^2$ and $\omega \in [-1.5, 1.5]\,\text{rad/s}$.

Objective and Environments. In each experiment, a nominal, safety-agnostic PD controller drives the robot toward a sequence of goals in an a priori unknown environment, while the respective safety filter modifies the control input, transitioning to the next goal once the current goal is reached. We consider two environments: `SimpleNav` (simple constrained environment with 4-5 obstacles), and `Warehouse` (a simulated factory environment with objects of varying shapes and sizes). The safety constraints enforce both obstacle avoidance and a minimum forward velocity $v_{\min} = 0.1$ m/s for deadlock mitigation.

GP Parameters. The surrogate function $s(x)$ is modeled using a GP with a Matérn kernel of order $\nu = 3/2$, trained online using a maximum of 256 measurements at each iteration. The kernel's length scale is $l = 0.087$, with a characteristic length scale $\lambda = 20$. To ensure real-time performance, we use a sparse approximation with 100 basis functions arranged on a 10×10 grid. The GP operates over a bounded local spatial domain of half-width 2.5 m in each coordinate, enabling efficient localized updates of the SDF.

Baselines. To evaluate the performance of our approach, we consider the following comparisons:

1) Oracle: CBVF computed online via HJR leveraging an offline mapped SDF (known environment, upper bound on achievable performance).
2) GP-SDF: Safety filter using the GP-learned SDF directly, without value-function refinement.
3) SDF-CBF (No GP): Our method using an SDF computed from a distance transform of a discretized occupancy map rather than a GP.
4) D* Lite [25]: Path planning method updated with the SDF-GP; if the goal is unobserved, a flood-fill algorithm selects the nearest reachable free-space target.
5) iSDF [6]: Our method using a continually learned neural SDF rather than a GP-SDF.
6) GP-CBF (Ours): Our proposed method, combining the GP-SDF with online CBVF refinement via refineCBF.

A. Simulation experiments

For simulation comparison, the `SimpleNav` consists of a 10 m × 10 m room with four cylindrical obstacles, and the `Warehouse` is a 30 m × 50 m warehouse with narrow corridors formed by shelving and support pillars. The simulated LiDAR has a 30 m range and additive zero-mean Gaussian noise $\mathcal{N}(0, \sigma^2)$ with standard deviation $\sigma = 0.05$.

Table I summarizes quantitative performance, and Fig. 2 shows representative trajectories and learned safe sets. In both environments, the Oracle and GP-CBF reach all goals safely, whereas all other baselines fail. Because SDF-CBF (No GP) constructs the SDF directly from noisy measurements without confidence bounds, it overestimates clearance near obstacles and misclassifies unsafe states as safe, resulting in collision. In contrast, our method synthesizes a conservative boundary estimate, ensuring the learned safe set remains contained within the true safe set with high probability. GP-SDF learns a smooth SDF but applies it directly as a control barrier function without incorporating system dynamics or control limits, so the filter cannot guarantee control invariance. D* Lite fails when the goal lies beyond the observed obstacle boundary, as the planner is unable to infer a feasible route through unobserved areas. In contrast, iSDF attempts to infer a global map from local observations, but sensor noise reduces the SDF's accuracy too early in the learning process to maintain safety.

Our method requires about 75% more time than the Oracle in `SimpleNav`, and about 100% more time in the `Warehouse`. This arises from narrow passages and partial

[1]The implementations of the GP-SDF mapping algorithm and HJR refinement are available on GitHub: GP-SDF Mapping and RefineCBF.

TABLE I: Performance comparison of control policies in the simulated `SimpleNav` and `Warehouse` environments. Performance is evaluated by the number of goals reached, total completion time, and total distance traveled.

`SimpleNav` Environment

Control Policy	Goals Reached	Completion Time (s)	Distance Traveled (m)
Oracle	5/5	125	44
GP-SDF	4/5	56	30
SDF-CBF	5/5	142	64
D* Lite	5/5	96	28
iSDF	5/5	134	44
GP-CBF (Ours)	**5/5**	**165**	**63**

`Warehouse` Environment

Control Policy	Goals Reached	Completion Time (s)	Distance Traveled (m)
Oracle	9/9	896	262
GP-SDF	0/9	7	2
SDF-CBF	2/9	178	55
D* Lite	2/9	98	57
iSDF	0/9	43	12.5
GP-CBF (Ours)	**9/9**	**1780**	**422**

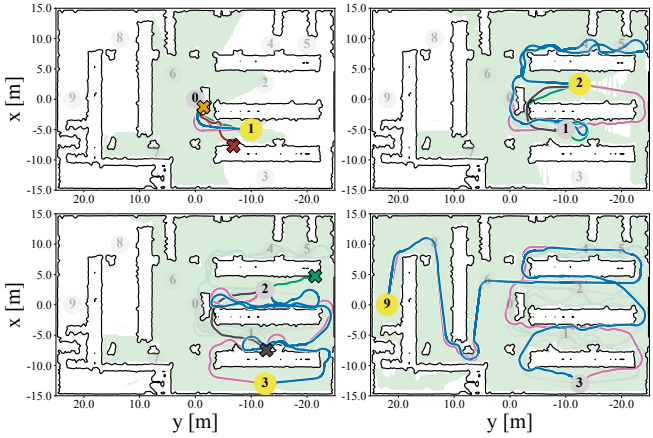

Fig. 2: Trajectories under different control policies in the `Warehouse` simulation. Each subplot shows the robot trajectory after reaching goals 1, 2, 3, and 9 (top left to bottom right), along with the corresponding safe regions estimated by GP-CBF.

observability: with conservative initialization and treating unobserved areas as unsafe, the robot must detour and revisit regions before certifying safe passage. Despite this increased travel time, Fig. 3 shows the safe set learned by our method progressively expands towards the ground-truth safe set available to the Oracle. As LiDAR measurements accumulate, the GP-SDF and associated CBVF expand to encompass the previously unobserved safe regions.

In our implementation, the HJR refinement runs at about 3Hz, while the GP-SDF update runs at 10Hz in the `SimpleNav` and 2Hz in the `Warehouse` environments. Because the 40Hz controller enforces safety asynchronously using the most recent CBVF, closed-loop safety is preserved; consequently, lower update frequencies primarily result in increased conservatism during exploration.

To assess robustness, we perform an ablation study in the `Warehouse` environment by varying the LiDAR sensing range and the standard deviation of the measurement noise. The results are reported in Table II. We observe that increasing measurement noise increases completion time, as the uncertainty increases the conservativism, but does not affect the completion rate. However, severely limiting the sensor range leaves significant portions of the environment outside the verifiable safe set, effectively confining the robot to its initial vicinity and halting progress.

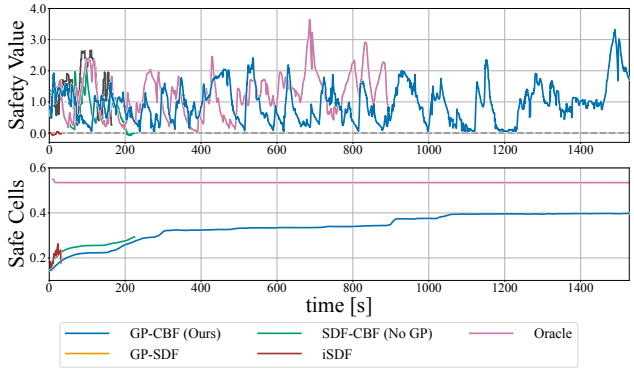

Fig. 3: Evolution of the safety value and the fraction of safe states in the `Warehouse` simulation. **Top:** Safety value along the executed trajectory. The proposed method and the Oracle remain strictly positive, whereas all remaining baselines violate the safety threshold and collides. **Bottom:** Fraction of the state space certified as safe over time. As additional regions are explored, the certified safe set expands and converges toward the ground-truth safe region.

TABLE II: Performance of GP-CBF under varying sensing range and measurement noise in the `Warehouse` simulation.

Configuration (range, noise)	Goals Reached	Completion Time (s)	Distance Traveled (m)
30 m, $\sigma = 0.01$	7/7	1120	284
30 m, $\sigma = 0.05$	7/7	1780	422
15 m, $\sigma = 0.03$	7/7	1701	580
8 m, $\sigma = 0.01$	0/7	48	10
8 m, $\sigma = 0.05$	0/7	47	7

B. Hardware experiments

Complementing the simulation results, we carry out real-world experiments using a Clearpath Jackal. On board, the robot is equipped with an Intel Core i7-9700TE CPU with 16 GB RAM, an NVIDIA GeForce GTX 1650 GPU, and an Ouster OS1-32 LiDAR sensor. The `SimpleNav` environment is a confined area with five rectangular obstacles, and the `Warehouse` is a mock factory floor, shown in Fig. 1.

Results are summarized in Table III. As in simulation, GP-SDF performs poorly, failing to reach goals that require safety filtering beyond the nominal controller. In contrast, our method achieves performance close to the Oracle in both completion time and distance traveled. The stronger relative performance compared to the `Warehouse` simulation is due to the smaller scale of the real environments, allowing the GP-CBF to capture relevant geometry with fewer exploration steps, as reflected in Fig. 4. Although the safe set initially expands, it eventually plateaus. This is due to increased model uncertainty and measurement noise in hardware, yielding more conservative predictions in sparsely observed regions.

VI. Conclusions

We present a learning-based framework for synthesizing a valid CBF through the generation of a continuously differentiable GP-based SDF, refined to ensure safety. We

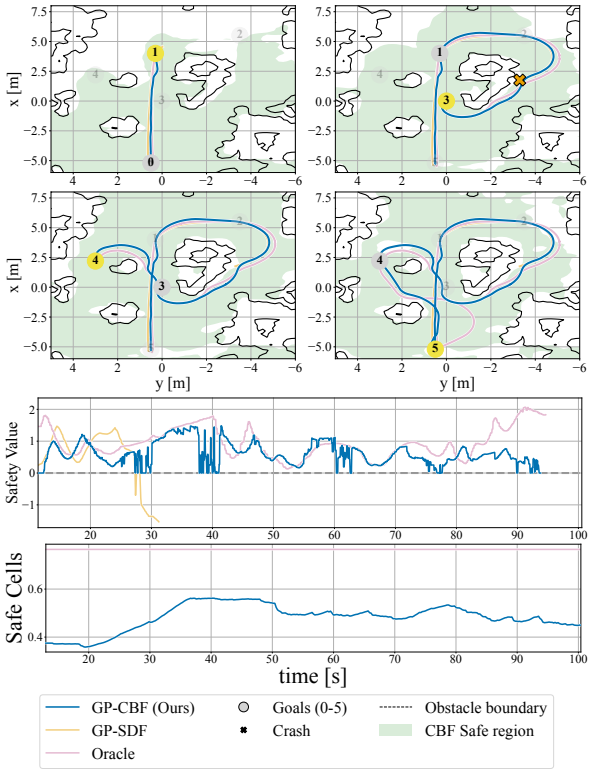

Fig. 4: Performance comparison in the `Warehouse` environment. **Top**: Robot trajectories for reaching five selected goals under different control methods. **Middle**: Evolution of the value function along the robot's trajectory for each method. **Bottom**: Fraction of the discretized state space certified as safe over time.

TABLE III: Performance comparison of the considered safety filters in hardware for the simple and factory environments.

`SimpleNav` Environment

Control Policy	Goals Reached	Completion Time (s)	Distance Traveled (m)
Oracle	6/6	114	23
SDF-GP	3/6	58	13
GP-CBF (Ours)	**6/6**	**93**	**23**

`Warehouse` Environment

Control Policy	Goals Reached	Completion Time (s)	Distance Traveled (m)
Oracle	5/5	94	31
SDF-GP	2/5	47	15
GP-CBF (Ours)	**5/5**	**104**	**34**

guarantee robustness against sensor noise by establishing formal probabilistic safety guarantees. The effectiveness of the framework is demonstrated through simulation and hardware experiments over a wide array of environments and noise conditions. Future work will explore the adaptation of this framework to dynamically evolving environments and different sensing modalities.

References

[1] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-board MAV planning," in *IROS*, 2017.

[2] L. Han, F. Gao, B. Zhou, and S. Shen, "FIESTA: Fast Incremental Euclidean Distance Fields for Online Motion Planning of Aerial Robots," in *IROS*, 2019.

[3] Y. Pan, Y. Kompis, L. Bartolomei, R. Mascaro, C. Stachniss, and M. Chli, "Voxfield: Non-Projective Signed Distance Fields for Online Planning and 3D Reconstruction," in *IROS*, 2022.

[4] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation," in *CVPR*, 2019.

[5] T. Takikawa, J. Litalien, K. Yin, K. Kreis, C. Loop, D. Nowrouzezahrai, A. Jacobson, M. McGuire, and S. Fidler, "Neural geometric level of detail: Real-time rendering with implicit 3d shapes," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11353–11362, 2021.

[6] J. Ortiz, A. Clegg, J. Dong, E. Sucar, D. Novotny, M. Zollhoefer, and M. Mukadam, "isdf: Real-time neural signed distance fields for robot perception," in *Robotics: Science and Systems*, 2022.

[7] B. Lee, C. Zhang, Z. Huang, and D. D. Lee, "Online continuous mapping using gaussian process implicit surfaces," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6884–6890, 2019.

[8] L. Wu, K. M. B. Lee, L. Liu, and T. Vidal-Calleja, "Faithful Euclidean distance field from log-Gaussian process implicit surfaces," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2461–2468, 2021.

[9] L. Wu, C. Le Gentil, and T. Vidal-Calleja, "VDB-GPDF: Online Gaussian Process Distance Field with VDB Structure," *IEEE Robotics and Automation Letters*, 2025.

[10] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, "Learning control barrier functions from expert demonstrations," in *Proc. IEEE Conf. on Decision and Control*, 2020.

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

[12] A. Peruffo, D. Ahmed, and A. Abate, "Automated and formal synthesis of neural barrier certificates for dynamical models," *Tools and Algorithms for the Construction and Analysis of Systems*, vol. 12651, pp. 370 – 388, 2021.

[13] A. Begzadić, A. Lederer, J. Cortés, and S. Herbert, "Learning high-order CBFs using Gaussian processes for systems in Brunovský canonical form," in *Proc. IEEE Conf. Decision and Control*, pp. 2945–2950, 2025.

[14] K. Long, Y. Yi, Z. Dai, S. Herbert, J. Cortés, and N. Atanasov, "Sensor-based distributionally robust control for safe robot navigation in dynamic environments," *International Journal of Robotics Research*, 2025.

[15] J. J. Choi, D. Lee, K. Sreenath, C. J. Tomlin, and S. L. Herbert, "Robust control barrier-value functions for safety-critical control," in *Proc. IEEE Conf. Decision and Control*, pp. 6814–6821, 2021.

[16] S. Tonkens and S. Herbert, "Refining control barrier functions through Hamilton-Jacobi reachability," in *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2022.

[17] S. Tonkens, S. Kojima, C. Liu, J. Masri, and S. Herbert, "Refining almost-safe value functions on the fly," *arXiv preprint arXiv:2602.23478*, 2026.

[18] 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, 2017.

[19] J. J. Choi, D. Lee, K. Sreenath, C. J. Tomlin, and S. L. Herbert, "Robust control barrier-value functions for safety-critical control," in *Proc. IEEE Conf. on Decision and Control*, 2021.

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

[21] R. Senanayake and F. Ramos, "Bayesian hilbert maps for dynamic continuous occupancy mapping," in *Conf. on Robot Learning*, 2017.

[22] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *Journal of the Association for Computing Machinery*, vol. 21, no. 4, p. 163–169, 1987.

[23] S. R. S. Varadhan, "On the Behavior of the Fundamental Solution of the Heat Equation with Variable Coefficients," *Communications on Pure and Applied Mathematics*, 1967.

[24] N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger, "Information-theoretic regret bounds for Gaussian process optimization in the bandit setting," *IEEE Transactions on Information Theory*, vol. 58, p. 3250–3265, May 2012.

[25] S. Koenig and M. Likhachev, "D* lite," in *Eighteenth national conference on Artificial intelligence*, pp. 476–483, 2002.