LiDAR-Based Online Control Barrier Function Synthesis for Safe Navigation in Unknown Environments

Shaghayegh Keyumarsi, *Graduate Student Member, IEEE*, Made Widhi Surya Atman, *Member, IEEE*, and Azwirman Gusrialdi, *Member, IEEE*

***Abstract*—This letter presents a novel extension of the control barrier function (CBF) as the low-level safety controller for autonomous mobile robots navigating in unknown environments. The main challenges of implementing CBF in real-world situations arise from the absence of a model or the lack of an exact one for the environment. Additionally, online learning is needed for the robot to maneuver in an unknown environment which leads to dealing with the sampled data set size, memory, and computational complexity. We address these challenges by designing an online non-parametric LiDAR-based safety function using the Gaussian process (GP). It is both efficient in data size and eliminates the requirement to store previous data. Then, a CBF is synthesized using the proposed safety function to rectify the safe control input. The effectiveness of the LiDAR-based CBF synthesis for navigation in unknown environments was validated by conducting experiments on unicycle-type robots.**

***Index Terms*—Autonomous agents, collision avoidance, machine learning for robot control.**

I. Introduction

DEPLOYING autonomous systems in uncertain or unknown environments poses a fundamental challenge. These systems ranging from autonomous mobile robots and drones to self-driving cars must navigate through unfamiliar terrains and scenarios while ensuring safety constraints. The main concern is enabling autonomous systems to perform their designated task without compromising on safety. Consequently, a robust safety framework is necessary for autonomous systems. Expanding this framework to sustain the well-being of individuals, communities, and the environment paves the way for the acceptance, integration, and widespread adoption of autonomous systems.

A promising approach to principally encode safety by utilizing/ensuring invariance of a set is via control barrier function (CBF) [1]. CBF is a powerful mathematical concept that achieves safety by ensuring that the system remains within the safe set. CBF gained popularity in recent years due to its potential to provide safety guarantees. It can be applied in various fields, especially in robotics and autonomous systems [2], [3], [4]. The core objective of CBF is defining a barrier by a safety function characterizing the safe set of the system. CBFs are implemented using optimization techniques. One of their advantages is handling both input-dependant and state-dependant constraints which provides the capability to include additional safety constraints as required.

The traditional CBF necessitates an accurate model of the system dynamics to guarantee safety in a known environment. Specifically, CBF requires a candidate safety function of the required relative degree and the nominal model of the system to render safety. The safe set of the system is modeled as the super-level set of a safety function. Obtaining the safety function is challenging in many practical settings. It requires knowledge and great effort to design the safety function in advance restricting its application even for known environments [5]. Due to the advancements in technology, computing power, and availability of data, recent works have extended the application of CBF using data-driven approaches. They enable autonomous systems to learn and adapt. There are existing works that incorporate online learning to CBF to compensate for unmodeled dynamics, model uncertainty, and parametric errors [6], [7]. A relevant line of research can be found in the class of safe reinforcement learning (RL) that learns the safety certificate using Neural Networks (Neural barrier methods) [8], [9]. In [10], the CBF is learned using demonstration data. The certificate can also be learned alongside the control policy [11]. However, certificate learning requires certificate verification and is limited to offline training. In [12], a neural network CBF is trained with replay memory for obstacle avoidance in static and unknown environments that takes the estimated distance error into account.

Recently, studies have proposed methods to extend CBF to unknown environments. Srinivasan et al. [13] sample the environment and classify the samples into safe and unsafe sets by a support vector machine classifier to generate the desired barrier. Bayesian learning [14] has been employed to update CBF as data is observed. Bayesian inference predicts the outcome of unobserved data by fitting a probability model to observed data if there are any. Bayesian learning is a powerful framework particularly when dealing with limited data and uncertainty. However, it is

Manuscript received 18 August 2023; accepted 25 November 2023. Date of publication 4 December 2023; date of current version 18 December 2023. This letter was recommended for publication by Associate Editor A. M. Metelli and Editor J. Kober upon evaluation of the reviewers' comments. The work of A. Gusrialdi was supported by the Research Council of Finland under Academy Project Decision, under Grant 330073. *(Corresponding author: Shaghayegh Keyumarsi.)*

The authors are with the Automation and Mechanical Engineering Unit, Tampere University, 33720 Tampere, Finland (e-mail: Shaghayegh.keyumarsi@tuni.fi; widhi.atman@tuni.fi; azwirman.gusrialdi@tuni.fi).

Digital Object Identifier 10.1109/LRA.2023.3339059

computationally demanding, especially as the complexity of the model and data increases. Also, it requires careful consideration of prior. GP [15] is a popular non-parametric Bayesian learning method. To be more specific, it is a generalization of infinite dimension Gaussian distributions. The GP-based functions offer smoothness, flexibility, interpretability, stable and reliable behavior, and mathematical convenience. In [16], online samples of the environment are collected to train a sparse Bayesian classifier estimating the probability of collision and subsequently utilized in CBF for developing a safe coverage control. Sparsification is employed within this methodology as it involves consistently sampling the environment to train the classifier. In [17], safety samples are collected online to model safety as a non-parametric Gaussian process (GP). This model requires some degree of familiarity with the environment as it builds upon the suggested safety function. Both of the approaches in [16], [17] are only applicable to static environments, as they are constantly collecting data from the environment and suffer from the computational complexity associated with large data sizes. While studies have investigated probabilistic machine learning in CBF, there has not been any GP-CBF formulation that uses LiDAR-based sensor data directly being efficient in data size and applicable to completely unknown dynamic environments. In this letter, a novel online CBF synthesis is proposed for obstacle avoidance in static and dynamic unknown environments using LiDAR measurements. In particular, our main contributions are the following.

- A novel GP-LiDAR-based safety function specifically designed for obstacle avoidance in completely unknown dynamic environments is proposed.
- Contrary to previous works such as [12], [13], [16], [17], the proposed GP-based safety function is efficient in the number of samples and it does not require previously sampled data to be stored.
- The proposed approach enables the robot to synthesize CBF online irrespective of the shape and number of obstacles. That is, the resultant safe set is not restricted to convex unsafe sets.
- The validity of the proposed safety function and the synthesis of CBF were confirmed through experimental verification on mobile robots equipped with an onboard LiDAR sensor.

The outline of the letter is as follows. In Section II, we review the mathematical preliminaries used in this letter. In the next section, the problem statement is described. A novel LiDAR-based CBF is proposed in Section IV. Finally, the effectiveness of the proposed algorithm is investigated through experiments in Section V. The conclusions and future work are provided in Section VI.

II. Preliminaries

This section outlines the CBF and Gaussian process regression (GPR) to encode safety through Bayesian learning.

A. Control Barrier Function

Consider a nonlinear control-affine system $\mathcal{P}$,

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

where $x \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state of the system and $u \in \mathbb{R}^m$ is the control input. The vector fields f and g are assumed to be continuous, locally Lipschitz, and the origin is an equilibrium point of $\mathcal{P}$. Assume that the state space can be decomposed into a *safe set* $\mathcal{X}_\mathcal{S}$ and an *unsafe set* $\mathcal{X}_\mathcal{U}$ such that $\mathcal{X}_\mathcal{S} \cup \mathcal{X}_\mathcal{U} = \mathcal{X}$. It is additionally assumed that $\mathcal{X}_\mathcal{S}$ can be modeled as *super-level set* of a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$:

$$\begin{aligned} \mathcal{X}_\mathcal{S} &= \{x \in \mathbb{R}^n : h(x) \geq 0\}, \\ \partial\mathcal{X}_\mathcal{S} &= \{x \in \mathbb{R}^n : h(x) = 0\}, \\ Int(\mathcal{X}_\mathcal{S}) &= \{x \in \mathbb{R}^n : h(x) > 0\}, \end{aligned} \tag{2}$$

where the boundary and interior of the set are defined by $\partial\mathcal{X}_\mathcal{S}$ and $Int(\mathcal{X}_\mathcal{S})$, respectively.

Definition 1: The set $\mathcal{D}$ is forward invariant if, $x(0) \in \mathcal{D}$ and for all $t \geq 0$, $x(t) \in \mathcal{D}$.

The system remains safe if $\mathcal{X}_\mathcal{S}$ is forward invariant. The CBF, defined in the following, gives the necessary and sufficient conditions for forward invariance (safety) as described.

Definition 2 (CBF [1]): Given the dynamic system $\mathcal{P}$ as in (1), the safe set $\mathcal{S}$ and the function *h(x)* as in (2), the function *h(x)* is a CBF, if there exists a locally Lipschitz, extended $class - \kappa_\infty$ function γ (i.e., $\gamma(0) = 0$ and it is strictly increasing) such that for any $x \in \mathcal{X}_\mathcal{S}$,

$$\sup_{u \in \mathbb{R}^m} L_f h(x) + L_g h(x) u + \gamma(h(x)) \geq 0. \tag{3}$$

B. Bayesian Learning: Gaussian Process Regression

GP is a flexible Bayesian regression method on an underlying kernel that does not need to specify the basis function. In addition, it has straightforward calculations computable by linear algebra. In the following, a brief review of GP regression is presented.

GP is a collection of random variables with joint Gaussian distribution over any finite subset of them. The GP is fully defined by its mean function $m(x) : \mathcal{X} \to \mathbb{R}$ and covariance function $k(x, x') : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ written as

$$\mathcal{GP}(m(x), k(x, x')), \tag{4}$$

where $\mathcal{X} \in \mathbb{R}^r$ is the input domain. Usually, the mean function is considered zero for simplicity. Note that the covariance matrix of samples joint Gaussian distributions, determined by the covariance function, must be positive definite. A popular choice of kernel function is the squared exponential (SE) kernel,

$$k(x_i, x_j) = \sigma_{SE}^2 e^{-\frac{1}{2}(x_i - x_j)^T L^{-2}(x_i - x_j)}, \tag{5}$$

with two parameters: $L \in \mathbb{R}^{r \times r}$ the length scale matrix and σ_{SE}^2 the signal scale.

Given a set of N data points with input x_i and corresponding output $y_i \in \mathbb{R}$, the training dataset consisting of $X =$

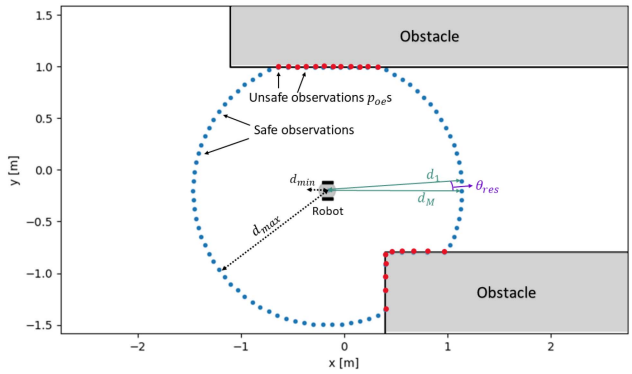

Fig. 1. Problem setting: navigation of a mobile robot equipped with a LiDAR sensor in an unknown environment with obstacles.

$[x_1, \ldots, x_N]^T \in \mathbb{R}^{N\times r}$ and $Y = [y_1, \ldots, y_N]^T \in \mathbb{R}^N$ are constructed. The GPR prediction at query point x_* is given by

$$y_*|X, Y, x_* \sim \mathcal{N}(\mu(x_*), \sigma^2(y_*)) \tag{6}$$

$$\mu(x_*) = k_*^T K^{-1} Y \tag{7}$$

$$\sigma^2(y_*) = k(x_*, x_*) - k_*^T K^{-1} k_*, \tag{8}$$

where $\mathcal{N}$ is the normal distribution, $k_* = [k(x_*, x_1), \ldots, k(x_*, x_N)]^T \in \mathbb{R}^N$ is the covariance vector between query point x_* and the training data set X, and K denotes the $N \times N$ matrix of covariance function evaluated at each pair of training point data set X with one another, i.e., its element $[K]_{ij} = k(x_i, x_j)$, $i, j \in \{1, \ldots, N\}$. The $\mu(x_*)$ and $\sigma^2(x_*)$ are respectively defined as the mean and variance of the predicted Gaussian distribution.

III. Problem Statement

Consider a mobile robot given by the control-affine system $\mathcal{P}$ in (1) where it is assumed that there is no constraints on control input u. The state $x = \begin{bmatrix} p & \theta \end{bmatrix}^T = \begin{bmatrix} p_x & p_y & \theta \end{bmatrix}^T \in \mathcal{X} \subseteq \mathbb{R}^3$ consists of the robot's position p in x-y plane and its rotational angle θ. For the sake of simplicity and clarity, the time dependence of variables x and p has been omitted from the formulas presented in this letter. It is assumed that the robot has access to its state at time t. The mobile robot navigates in an unknown environment consisting of both static and dynamic obstacles. To this end, the robot is equipped with a LiDAR to detect obstacles. Specifically, the noisy LiDAR's depth measurements are denoted by $d(t) = [d_1(t), \ldots, d_M(t)]^T \in \mathbb{R}^M_{>0}$, where M is the total number of samples determined by the LiDAR's angular resolution θ_{res}. Assume that θ_{res} is sufficiently small to detect objects in the surroundings from a distance. Each LiDAR's scalar observation d_i is bounded by $d_{\min}$ and $d_{\max}$, the LiDAR's minimum and maximum detection range, respectively. The position of the detected obstacle edge by the distance d_i is denoted by $p_{oe,i}$. An example of the setting is shown in Fig. 1.

Next, a mapping function $\pi : x \in \mathcal{X} \to p$ is defined to map the robot's state x to the robot's position p. Given a goal position p_g provided by a high-level planner, we aim to design a safe motion control u for ensuring that the robot reaches the goal position p_g without colliding with unknown obstacles. To this end, assume that there exists a continuously differentiable but unknown function $h(\pi(x))$ as defined in (2) whose super level set encodes the safe regions $\mathcal{X}_\mathcal{S}$ of the environment. It is assumed that there is no a priori knowledge about the unsafe regions $\mathcal{X}_\mathcal{U}$ except that the robot starts from a safe position $\pi(x(0)) \in \mathcal{X}_\mathcal{S}$. Furthermore, let u_{nom} be the nominal control input computed by the operator-defined controller. Examples of such nominal controllers include MPC, proportional (go-to-goal) control, and others. The problem is then boiled down into the following.

Problem 1: Consider the affine control mobile robot in (1) navigating in an unknown environment with obstacles. Train a safety function synthesizing the CBF $h(\pi(x))$ with online data $d(t)$ from the LiDAR sensor to design a controller $u(t)$ that minimally deviates from $u_{nom}(t)$ and ensures no collision between the robot and the environment.

IV. LiDAR-Based CBF Synthesis

In this section, a new online CBF synthesis specifically designed for guaranteeing safety in unknown environments is proposed. The main challenge of designing CBF for unknown environments is defining the safe set as the super-level set of the function $h(x)$. Due to the lack of prior knowledge about the unsafe regions, the safety function is learned based on LiDAR-sensor observations. Bayesian learning is used to extend CBFs to unknown environments. We use GPR for learning the safety function online using small data acquired by LiDAR at each sampling instance. The proposed GP provides a smooth, flexible, and adaptive model with analytical tractability that captures all of the detected edges and is capable of modeling all obstacles altogether as one safety function. In the following, a LiDAR-based GP-CBF synthesis is presented to render the system safe.

A. GP-Based Safety Function Synthesis

In this work, $h(x)$ is directly modeled as a continuously differentiable function of the minimum distance to the obstacle, which is efficient and bounded in terms of the number of sampled data points. This is contrary to the majority of common approaches [13], [16], which learn a safety classifier based on LiDAR sensor readings and/or continue collecting samples that burden the computational complexity of GP [17]. This approach is more closely aligned with the definition of a safety function being differentiable and only includes local essential information which elevates the robot's perception and performance. To clarify, the desired behavior as in (2) is achieved by only collecting the obstacle edges with a sampling distance. The data obtained at each time instance is exclusively utilized for safe control computation at that particular moment. Consequently, in our data collection approach, we specifically focus on gathering only the essential information at each sample time, without saving it for future reference which makes our approach efficient.

The first step is to model safety as a function of position. We formulate the safety function by the mean prediction of the $\mathcal{GP}$,

$$h(x) = \mu(\pi(x)). \tag{9}$$

To be efficient in the number of data points, the $\mathcal{GP}$ must have a safe assumption unless contradicting data has been sensed. This is delivered by assigning the fixed positive mean function, $m(\pi(x)) = 1$. This method is novel since previous works heavily depend on training rather than modifying the GP prior. Consequently, the robot explores freely and heads toward unknown areas as long as it is deemed safe. This safety function design is simple, and most efficient in the number of data points. It should be noted that the formulation does not incorporate the variance since including it would require more frequent sampling, contradicting the goal of efficient sampling. Alternatively, a small tube of uncertainty is incorporated with the same function as the variance of the $\mathcal{GP}$ without the high computational cost. Additionally, note that considering a deterministic constant mean function is trivial because the usual zero mean formulas (7) and (8) can be applied to the difference between the observations and the mean value [15]. Accordingly, the predictive mean in (7) becomes,

$$\mu(\pi(x_*)) = m(\pi(x_*)) + k_*^T K^{-1}(Y - m(\pi(x_*))). \tag{10}$$

and the predictive variance (8) remains unchanged. The squared exponential kernel function (5) is chosen as it is smooth (infinitely differentiable), and a universal approximator [15]. Since the kernel input data is the position p and no specific direction is prioritized, the SE kernel is simplified by setting $L = \ell I$, where ℓ is a scalar length scale and I is the Identity matrix. In addition, σ_{SE} is set to 1. This gives the kernel function

$$k_{SE}(\pi(x_i), \pi(x_j)) = e^{-\frac{1}{2\ell^2}\|p_i - p_j\|^2}. \tag{11}$$

Additionally, a noise term is added to account for uncertainty and measurement errors as,

$$k_N(\pi(x_i), \pi(x_j)) = \sigma_N^2 I, \tag{12}$$

where σ_N^2 represents the noise variance. As a result, the following kernel function is obtained,

$$\begin{aligned} &k_{SE,N}(\pi(x_i), \pi(x_j)) \\ &= k_{SE}(\pi(x_i), \pi(x_j)) + k_N(\pi(x_i), \pi(x_j)) \end{aligned} \tag{13}$$

resulting in the prior $\mathcal{GP}(1, k_{SE,N}(\pi(x), \pi(x')))$.

The last step to reach the desired behavior for the safety function would be to sample the detected unsafe positions i.e., locations of points on the boundary of the unsafe sets. To this end, only the positions of obstacle edges are collected for defining the unsafe set to train the $\mathcal{GP}$ model. Specifically, after detecting an obstacle edge by the kth light of the LiDAR sensor at time t, the corresponding unsafe position (i.e., the position of the point at the obstacle's edge) $p_{oe,k}(t)$ is used in the GP if it satisfies the following condition,

$$\min_{j=1,\dots,N} \|p_{oe,k}(t) - x_j(t)\| \; > d_{sample}, \tag{14}$$

where d_{sample} denotes the minimum sampling distance. Once the sampling condition is satisfied, this new observation data $x_i(t) = p_{oe,k}(t)$ is assigned with $y_i(t) = -1$. Simply put condition (14) avoids dense sampling. Note that d_{sample} should be chosen sufficiently small based on the GP hyperparameters to avoid collision.

Remark 1: For the differentiability of the GP [18], the data set must be non-empty. To address this if no obstacle is detected, the nominal control input can be applied directly.

The forthcoming proposition aims to encapsulate the core idea of GP-LiDAR-based safety function.

Proposition 1: Consider a GP-based safety function as in (10) with a smooth differentiable kernel function proportional to the reciprocal of Euclidean distance, i.e., $k(x_i, x_j) \propto \|x_i - x_j\|^{-\frac{1}{2}} = d_i^{-1}$, the mean function $m(\pi(x)) = 1$, and a data set consisting of unsafe positions $x_i = p_{oe}$ assigned with $y_i = -1$. The safety function formulated by the mean prediction of the $\mathcal{GP}$, characterizes safety based on $d^*(t)$ as follows:

$$h(x) = \begin{cases} 1 & \text{if } d^*(t) \geq d_{safe} \\ 1 - 2k_y(d_i(t)) & \text{if } 0 < d^*(t) < d_{safe}, \\ -1 & \text{if } d^*(t) = 0, \end{cases} \tag{15}$$

where $d^*(t)$ denotes the closest distance to obstacles, that is

$$d^*(t) = \min_{i=1,\dots,M} d_i(t),$$

d_{safe} is a user-defined safe distance threshold determined by the $\mathcal{GP}$ kernel function and hyperparameters, and $k_y(d_i(t)) = k_*^T K^{-1} 1_N : (0, \infty) \to (0, 1)$.

Proof: As defined by the proposition $h(x) = 1 + k_*^T K^{-1}(-1 - 1) = 1 - 2k_y(d_i(t))$. Near the boundary of the obstacles, i.e., $d^*(t) \sim 0$, $\mathcal{GP}$ returns $\mu(\pi(x_*)) \sim 1 - 2 = -1$ as the edges had been sampled. At positions that are far from sampled points, i.e., $d^*(t) \geq d_{safe}$, the kernel k_* is close to zero since no correlation is captured. Consequently, the $\mathcal{GP}$ yields $\mu(\pi(x_*)) \approx m(\pi(x_*)) = 1$. Finally, for the interval $0 < d^* < d_{safe}$, $k_y(d_i(t))$ varies proportionally to the reciprocal of Euclidean distance, exhibiting the same dependency as the kernel function. □

As established in Proposition 1, the proposed safety function maps the position of the robot to the range $[-1, 1]$ based on its distance from obstacles. An example of a color map of the local safety function using LiDAR based on *Proposition 1* is shown in Fig. 2. In the following, a $\mathcal{GP}$-based safety control according to *Proposition 1* is designed.

B. GP-CBF-Based Safety Control

Once the safety function has been characterized, a minimally invasive safe controller can be calculated based on the admissible control space of the CBF. The rectified control input u can be realized as the following Quadratic program (QP):

$$u(t) = \underset{u \in \mathbb{R}^m}{\operatorname{argmin}} \; \|u - u_{\text{nom}}(t)\|^2 \tag{16a}$$

$$\text{Subject to:} \quad L_f h(x) + L_g h(x) u + \gamma(h(x)) \geq 0, \tag{16b}$$

where the Lie derivatives of $h(x)$ along the vector fields f and g are denoted by $L_f h(x) = \frac{\partial h(x)}{\partial x} f(x)$ and $L_g h(x) = \frac{\partial h(x)}{\partial x} g(x)$

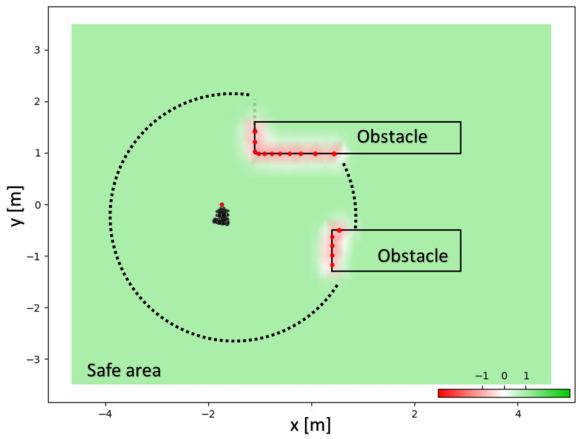

Fig. 2. 2D space is divided into a grid, and the safety function is computed using a GP model, as described in *Proposition 1*, at each grid point. Subsequently, the local safety function is visualized as a color map, where different colors represent various safety levels. Specifically, green indicates $p \in \mathcal{X}_S$, white represents the borderline safe areas, and red signifies $p \in \mathcal{X}_U$.

respectively. To summarize, real-time safety is ensured by solving (16) which ensures the forward invariance of the safe set.

The partial derivative of $h(x)$ in (16) with respect to x at query point $x_* = \begin{bmatrix} p_* \\ \theta_* \end{bmatrix}$ is given by,

$$\left.\frac{\partial h(x)}{\partial x}\right|_{x=x_*} = \left.\frac{\partial \mu(\pi(x))}{\partial x}\right|_{x=x_*} = \left[Y^T K^{-1} \left.\frac{\partial k}{\partial p}\right|_{p=p_*} \quad 0 \right]^T, \tag{17}$$

where vector Y and matrix K are the dataset and kernel matrix, respectively, as defined in Section II-B. The differentiability of the kernel is required by (17). Note that the kernel k_{SE} is infinitely differentiable. Consequently, the kernel $k_{SE,N}$ is infinite mean square differentiable if the data set is nonempty according to sample path differentiability theorem [18]. The kernels derivative in (17) can be calculated as

$$\left.\frac{\partial k}{\partial p}\right|_{p=p_*} = \begin{bmatrix} dk_1(p_*) \\ \vdots \\ dk_N(p_*) \end{bmatrix}, \tag{18}$$

where $dk_i(p_*)$ is given by

$$dk_i(p_*) = -\frac{1}{l^2} k_{SE}(p_*, x_i)(p_* - x_i)^T. \tag{19}$$

The validity of the proposed CBF (16) is discussed in the following proposition.

Proposition 2: Given a safety function $h(x)$ constructed as in *Proposition 1* for system (1) and assuming *Remark 1*, the safety function $h(x)$ is a valid CBF, the main QP (16) is feasible and the inequality constraint (16b) is nontrivial.

Proof: Firstly, we show that the partial derivative of the safety function exists and is non-zero. Following that, we guarantee the existence of at least one acceptable input by verifying the non-emptiness of the QP problem feasible set. As derived from Sample Path Differentiability [18], the safety function is mean square differentiable if the data set is nonempty, and as stated in *Remark 1* we only compute the derivative for detected edges. To be more specific, in (17), Y is a vector of -1, matrix K is the kernel matrix and the kernel evaluated between pairs that are in the safe set $\mathcal{X}_S$ lies within the range (0,1]. As a result, the partial derivative of h with respect to the state vector can only approach zero or reach values close to zero at the edges which are in the unsafe set $\mathcal{X}_U$. This shows that the CBF constraint is non-trivial. To show that the QP problem is feasible, we check the feasibility set of the inequality constraints. As one Gaussian process is trained to model detected obstacles altogether, there is one inequality constraint. Consequently, for control affine systems (1), the constraint $L_f h(x) + L_g h(x)u + \gamma(h(x)) \geq 0$ is a single linear inequality constraint that divides the space into two and is always non-empty. Hence, since there is no constraint on control input u, there always exists a control input that satisfies constraints in (16b). □

Note that for general control affine system (1) adding control input constraints to the QP (16) may potentially lead to infeasibility. As mentioned at the beginning of Section IV-A, the data obtained at each time instance is exclusively utilized for safe control computation at that particular moment. As a result, the zero-super-level set changes as the data is updated. To this end, we define a strictly increasing, unbounded switching sequence $\{\tau_k\}_{k=1}^{\infty}$ starting at t_0 ($\tau_1 = t_0$), where τ_k is the time of kth sequence of the measurements. Furthermore, we impose the following assumption.

Assumption 1: During the data update at τ_{k+1} the states stay within the intersection of the zero-super-level sets corresponding to times τ_k and τ_{k+1}.

The above assumption can be ensured in practice by having a smaller sampling time (for example using a LiDAR with a fast update rate). The following proposition, which provides the solution to Problem 1, shows that the forward invariance property is ensured for the switching zero-super-level set. This class of invariance is also referred to as hybrid forward invariance in [19].

Proposition 3: (Hybrid forward invariance) Given a safety function constructed as in *Proposition 1* for system (1) and assuming *Remark 1* and *Assumption 1*, any locally Lipschitz continuous controller satifying (16) renders the switching zero-super-level set forward invariant.

Proof: First, observe that updating the data set at each sampling instance τ_k ensures that the safety function is locally Lipschitz continuous and smooth, and the forward invariance in ensured within each sampling interval $[\tau_k, \tau_{k+1})$, as shown in *Proposition 2*. Furthermore, *Assumption 1* ensures that the system will be in the safe set after the switching. Hence, the forward invariance of the system is guaranteed for the switching zero-super-level set, which completes the proof. □

Algorithm 1 outlines the steps to obtain online GP-LiDAR-based CBF steps. Note that due to the dynamic nature of the environment and discretization, there is a speed upper limit computable for a given sampling time and d_{safe} to ensure safety.

It is worth noting that our choice of observation y_i allows us to deal with an unknown dynamic environment without requiring a candidate safety function for sampling. Also, the data size is bounded by the maximum number of LiDAR samples M. Solely utilizing the edges detected at that specific time with the

TABLE I
COMPARISON TO PRIOR WORKS

Online Learning method for LiDAR-based CBF	data acquisition	data collection	Prior knowledge	time-varying environment
Our work	Sparse p_{oe}	Single-pass usage	Not needed	Applicable
Sparse Bayesian [16]	Sparse p_{oe}+p_{safe} accompanied by their labels	Accumulative	Not needed	Not applicable
supervised ML [13]	Sparse p_{oe}+p_{safe} accompanied by their labels	Accumulative	Not needed	Not applicable
Gaussian [17]	safety metric to obstacle	Accumulative	Prior knowledge about obstacles needed	Not applicable

Algorithm 1: LiDAR-Based Safety Control Using GP-CBF.

Initialization: System (1) is at a safe position: $x(0) \in \mathcal{X}_{\mathcal{S}}$.

Procedure:

Calculate $u_{nom}(t)$ at query point $x_* = x(t)$.
$X(t) = \emptyset$ i.e., $N = 0$.
for $i = 1$ to M **do**
  if ($d_i(t) < d_{\max}$ and (14)) **then**
    $N = N + 1$
    Collect $x_N(t) = p_{oe,i}(t)$, $y_N(t) = -1$.
  end if
end for
Determine $h(x(t))$ and $\frac{\partial h(x(t))}{\partial x}$ using (10) and (17). Then, solve (16) to compute the rectified control input, $u(t)$.

sampling interval d_{sample} makes our approach efficient in the number of data points. A comparison with prior work on sample efficiency is provided in the following subsection.

C. Comparison of Data/Sample Size

The size of the data set directly affects the efficiency of the proposed CBF. To this end, we compare the data size of the proposed LiDAR-based-GP CBF to other CBFs that use LiDAR data to learn the safety function online. Table I presents a detailed analysis and comparison. As presented in the table, the proposed LiDAR-based CBF only processes a sparse subset of detected edges of obstacles (p_{oe}) by enforcing the sampling distance. This process is done in a single pass usage in comparison to accumulative data gathering. In [13], a sparse set of safe (p_{safe}) and unsafe (p_{oe}) grid points over the domain with their labels are collected accumulatively. Also, a sparse Bayesian classifier is used in [16] accumulatively. The GP-based CBF proposed in [17] collects safety metrics of the environment accumulatively and also requires prior knowledge about the environment and obstacles. In conclusion, our proposed LiDAR-based CBF is more efficient in the long run and is applicable to unknown environments.

V. EXPERIMENT

In this section, we confirm the effectiveness of the GP-CBF-based safety control according to Algorithm 1 in a real environment. The objective of the experiments is to verify the safe rectification of the proposed LiDAR-based CBF by TurtleBot3 burger robot in an unknown environment with dynamic and static obstacles.

A. Experiment Setup

The experiment environment consists of mobile robots, static obstacles, and dynamic obstacles in a 5.3 m $\times$ 3 m field. Three TurtleBot3 burger robots performed the experiment. The robot's speed limit is 0.15 m/s. The positions of the robots are obtained by Vicon localization. All computations are done remotely on a PC with an Intel i7 processor. The computed control input is sent to each individual robot via Wi-Fi through ROS messages. The update rate of the LiDAR data is 5 Hz. The angular resolution of the LiDAR, θ_{res}, is set as 1° ($M = 360$).

1) Robot's Dynamic and Nominal Controller: The dynamic of the TurtleBot3 burger robot is given by

$$\begin{bmatrix} \dot{p_x}(t) \\ \dot{p_y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} v(t)\cos(\theta(t)) \\ v(t)\sin(\theta(t)) \\ \omega(t) \end{bmatrix} \tag{20}$$

where the control inputs are the longitudinal velocity input $v(t)$ and the angular velocity input $w(t)$. The high-level planner, i.e., u_{nom} is a go-to-goal controller. To apply the control input, a virtual leading point in front of the robot is controlled. Specifically, near identity diffeomorphism transformation [20] given by,

$$\begin{bmatrix} v(t) \\ w(t) \end{bmatrix} = \begin{bmatrix} \cos(\theta(t)) & \sin(\theta(t)) \\ -\frac{1}{r}\sin(\theta(t)) & \frac{1}{r}\cos(\theta(t)) \end{bmatrix} \begin{bmatrix} u_{nom,x}(t) \\ u_{nom,y}(t) \end{bmatrix} \tag{21}$$

is applied, where $r \in \mathbb{R}^{>0}$ is the distance of the controlled point from the center of the robot, $u_{nom,x}$ and $u_{nom,y}$ are computed by the go-to-goal controller and later rectified by CBF.

2) Scenario. Online Obstacle Avoidance in an Unknown Environment: The experiment comprises three concurrent implementations, during which each TurtleBot aimed to independently and safely reach its goal position in an unknown environment with obstacles. Consequently, to each TurtleBot the other two are perceived as dynamic obstacles. The environment except for the safe starting positions is unknown to the robots. Each TurtleBot is equipped with a LiDAR sensor. The unsafe position p_{oe} detected by the ith emitted laser light can be computed as,

$$p_{oe} = \begin{bmatrix} p_{oe,x} \\ p_{oe,y} \end{bmatrix} = \begin{bmatrix} p_x + d_i.\cos(\theta + i.\theta_{res}) \\ p_y + d_i.\sin(\theta + i.\theta_{res}) \end{bmatrix}. \tag{22}$$

The GP hyperparameters are set as $\ell = 0.17$ ($d_{safe} = 0.558$ m), and $\sigma_N = 10^{-2}$. The LiDAR maximum detection range is $d_{\max} = 1$. The minimum sampling distance is set as $d_{sample} =$

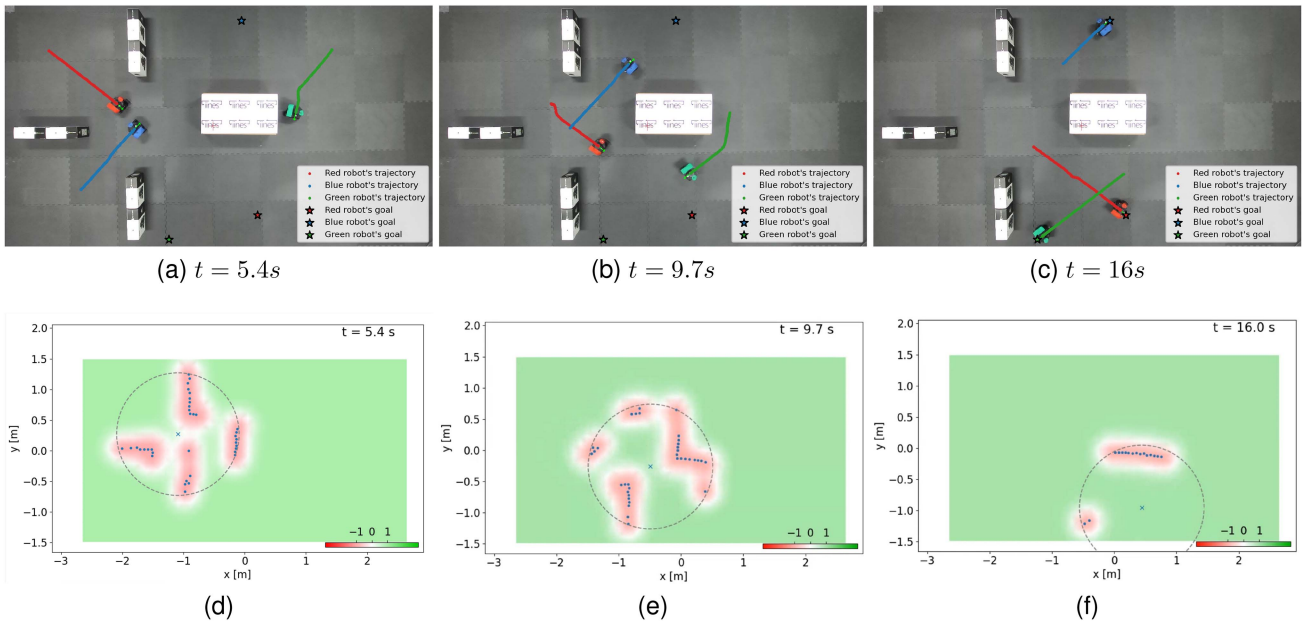

Fig. 3. Three frames of the implementation instances; (d)–(f) the plot of the safety function color map for the TurtleBot3 burger with the red tag. The colored lines in (a)–(c) shows the trajectories of the robots between each snapshot. The video recording of the experiment is accessible at https://youtu.be/T_68wJbRJgY.

TABLE II
DATA SIZE AND COMPUTATION TIME

	Minimum	Average	Maximum
Number of detected edges	0	119	153
Size of the data set	0	30	43
GP computation time (ms)	0	7.66	21.11
GP+QP computation time (ms)	0	7.96	21.11

0.05. Furthermore, function γ in (16) is assigned as $\gamma(h) = h^3$. The proportional gain of the nominal controller is set as $P_{gain} = 0.8$.

B. Results and Discussions

The computational analysis of the proposed LiDAR-based GP-CBF (the time it takes to acquire the data set, train the GP, and estimate the safety function and its derivative) is shown in Table II. Additionally, The computational analysis of LiDAR-based QP-GP-CBF (the time it takes to acquire the data set, train the GP, estimate the safety function and its derivative, and solve the QP) is shown as well. The minimum computation time is effectively 0.00 ms when no obstacle is detected because the system immediately applies the nominal control input. The number of data points detected and acquired in the GP is shown as well. The number of data used in the GPR is smaller than the data points detected due to the minimum sampling distance (14) which promotes data sparsity. The TurtleBots start heading toward their corresponding p_g and successfully avoid collision with static and dynamic obstacles. The GP is relatively fast because of having a small dataset and no sparsification method is required. Using online LiDAR observations, the LiDAR-based CBF rectifies the nominal control to navigate safely in the unknown environment. The red and blue robots detect each other between 2.9 seconds and 9.8 seconds, whereas the red and green robots detect one another starting from 9.7 seconds and continuing until the conclusion of the experiment. Fig. 3 shows three time frames of the experiment in one plot. The plot is accompanied by another plot that shows the sampled data, the local safety color map, and the radius of the local color map at the corresponding times of the red TurtleBot. Fig. 4 shows the minimum distance of TurtleBots to the obstacles over time. Note that the radius of the TurtleBot3 burger is 0.089 m. There is a jump in the minimum distance plot of the blue TurtleBot shown in Fig. 4 at $t = 4$ s when the red and blue TurtleBots meet. After a thorough analysis of the video recordings and the minimum distance plot of the blue and red TurtleBots, it became evident that the observed phenomenon is simply a result of the noise present in the LiDAR measurements. Specifically, we have observed that LiDAR sensors tend to get noisy measurements and are sensitive to reflective surfaces. Reflective surfaces result in larger inconsistencies in the LiDAR readings. Additionally, the noise usually intensifies when the robot has a relatively larger rotational speed causing larger errors in sensed range and accordingly on trained safety function. It is worth noting the QP in (16) remains feasible unless the LiDAR senses zero distance to a sampled obstacle edge which is a rare occurrence. Additionally, the QP CBF is still effective but more cautious since it tries to bring the state back to the safe set once it leaves the set. This property is a direct result of the asymptotic stability of the invariant set according to [21]. It is notable that the safety controller ensures safe navigation even in light of the significant noise present in the LiDAR measurements. In addition, since the dataset is completely updated in the subsequent timestep, the effect of noise is temporary and this can be seen in Fig. 4. One optional way to deal with this issue is to have the anomaly

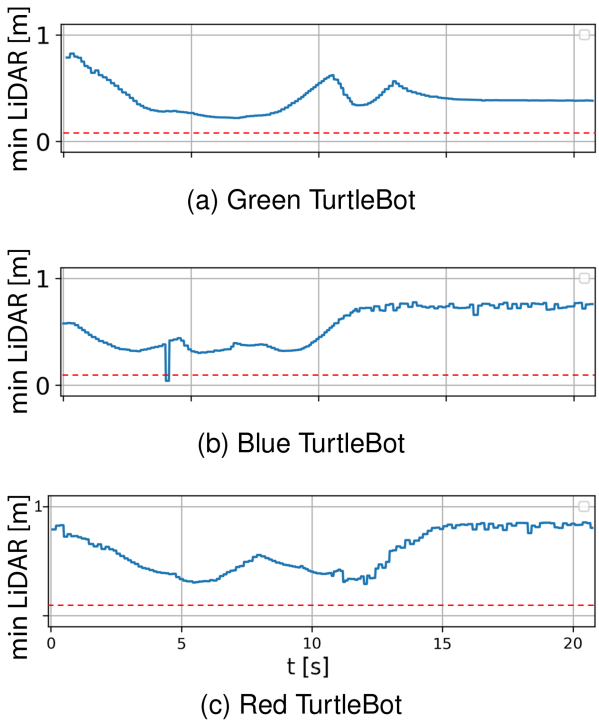

Fig. 4. Minimum distance to the obstacle sensed by LiDAR through experiment. The minimum safe distance is indicated by a red dashed line. (a) Green TurtleBot. (b) Blue TurtleBot. (c) Red TurtleBot.

detection on the LiDAR reading but we leave this as it is beyond the scope of this letter.

Remark 2: To ensure safety with a slow update rate, a larger d_{safe} is recommended by setting a larger ℓ value in (11) while navigating through narrow entrances requires opting for a smaller ℓ.

VI. Conclusion

In this work, first, we propose an online LiDAR-based local safety function for unknown dynamic environments with an optimum dataset without saving the data set for future reference. Then, a LiDAR-based GP CBF is constructed upon the trained safety function and the associated control synthesis is provided. The effectiveness of the proposed method is confirmed through implementations. In the future, we aim to add high-level planner features like resolving dead-lock situations of CBFs to our algorithm.

References

[1] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *Proc. 18th Eur. Control Conf.*, 2019, pp. 3420–3431.

[2] M. Rauscher, M. Kimmel, and S. Hirche, "Constrained robot control using control barrier functions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 279–285.

[3] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, Aug. 2017.

[4] W. Xiao and C. Belta, "High-order control barrier functions," *IEEE Trans. Autom. Control*, vol. 67, no. 7, pp. 3655–3662, Jul. 2022.

[5] G. Notomista and M. Saveriano, "Safety of dynamical systems with multiple non-convex unsafe sets using control barrier functions," *IEEE Control Syst. Lett.*, vol. 6, pp. 1136–1141, 2021.

[6] A. Taylor, A. Singletary, Y. Yue, and A. Ames, "Learning for safety-critical control with control barrier functions," in *Proc. 2nd Conf. Learn. Dyn. Control*, 2020, pp. 708–717.

[7] F. Castañeda, J. J. Choi, B. Zhang, C. J. Tomlin, and K. Sreenath, "Pointwise feasibility of Gaussian process-based safety-critical control under model uncertainty," in *Proc. IEEE 60th Conf. Decis. Control*, 2021, pp. 6762–6769.

[8] C. Dawson, S. Gao, and C. Fan, "Safe control with learned certificates: A survey of neural Lyapunov, barrier, and contraction methods for robotics and control," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 1749–1767, Jun. 2023.

[9] C. Dawson, B. Lowenkamp, D. Goff, and C. Fan, "Learning safe, generalizable perception-based hybrid control with certificates," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 1904–1911, Apr. 2022.

[10] A. Robey et al., "Learning control barrier functions from expert demonstrations," in *Proc. IEEE 59th Conf. Decis. Control*, 2020, pp. 3717–3724.

[11] T. Westenbroek, A. Agrawal, F. Castañeda, S. S. Sastry, and K. Sreenath, "Combining model-based design and model-free policy optimization to learn safe, stabilizing controllers," *IFAC-PapersOnLine*, vol. 54, no. 5, pp. 19–24, 2021.

[12] K. Long, C. Qian, J. Cortés, and N. Atanasov, "Learning barrier functions with memory for robust safe navigation," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4931–4938, Jul. 2021.

[13] M. Srinivasan, A. Dabholkar, S. Coogan, and P. A. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 7139–7145.

[14] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning*. Berlin, Germany: Springer, 2006.

[15] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT press, 2006.

[16] K. Mizuta, Y. Hirohata, J. Yamauchi, and M. Fujita, "Safe persistent coverage control with control barrier functions based on sparse Bayesian learning," in *Proc. IEEE Conf. Control Technol. Appl.*, 2022, pp. 311–318.

[17] M. A. Khan, T. Ibuki, and A. Chatterjee, "Gaussian control barrier functions: Non-parametric paradigm to safety," *IEEE Access*, vol. 10, pp. 99823–99836, 2022.

[18] D. Eriksson, K. Dong, E. Lee, D. Bindel, and A. G. Wilson, "Scaling Gaussian process regression with derivatives," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 6868–6878.

[19] P. Glotfelter, I. Buckley, and M. Egerstedt, "Hybrid nonsmooth barrier functions with applications to provably safe and composable collision avoidance for robotic systems," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 1303–1310, Apr. 2019.

[20] S. Wilson et al., "The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems," *IEEE Control Syst. Mag.*, vol. 40, no. 1, pp. 26–44, Feb. 2020.

[21] M. Egerstedt, *Robot Ecology: Constraint-Based Design for Long-Duration Autonomy*. Princeton, NJ, USA: Princeton Univ. Press, 2021.