SHIELD: Safety on Humanoids via CBFs In Expectation on Learned Dynamics

Lizhi Yang*[1], Blake Werner*[1], Ryan K. Cosner[1],
David Fridovich-Keil[2], Preston Culbertson[3], and Aaron D. Ames[1]

***Abstract*— Robot learning has produced remarkably effective "black-box" controllers for complex tasks such as dynamic locomotion on humanoids. Yet ensuring dynamic safety, i.e., constraint satisfaction, remains challenging for such policies. Reinforcement learning (RL) embeds constraints heuristically through reward engineering, and adding or modifying constraints requires retraining. Model-based approaches, like control barrier functions (CBFs), enable runtime constraint specification with formal guarantees but require accurate dynamics models. This paper presents SHIELD, a layered safety framework that bridges this gap by: (1) training a generative, stochastic dynamics residual model using real-world data from hardware rollouts of the nominal controller, capturing system behavior and uncertainties; and (2) adding a safety layer on top of the nominal (learned locomotion) controller that leverages this model via a stochastic discrete-time CBF formulation enforcing safety constraints in probability. The result is a minimally-invasive safety layer that can be added to the existing autonomy stack to give probabilistic guarantees of safety that balance risk and performance. In hardware experiments on an Unitree G1 humanoid, SHIELD enables safe navigation (obstacle avoidance) through varied indoor and outdoor environments using a nominal (unknown) RL controller and onboard perception.**

I. Introduction

As learning-based controllers achieve remarkable success in complex robotic tasks such as legged locomotion [1]–[8], they bring with them a fundamental tension: the black-box, data-driven nature, which enables their robust performance, simultaneously obscures our ability to provide formal safety guarantees or modify their constraints without expensive retraining. As more roboticists begin to field robust controllers trained using strategies like reinforcement learning (RL), developing ways to flexibly and adaptively constrain their behavior online to ensure safety (e.g., to avoid colliding with humans in their workspace) remains an open problem. Solving this problem is especially critical for humanoid robots, which by their very nature are designed to interact with humans in everyday environments.

Background. Several methods have emerged in recent years to enable the safe deployment of learning-based controllers. For example, conformal prediction provides a powerful framework for developing risk-aware controllers with quantile-based robustification [9], [10]. However, this approach often becomes computationally intractable as the number of samples increases. Alternatively, "backup"-style approaches employ a dual-controller strategy: a "performant" controller during normal operation and a separate safe controller that engages in high-risk scenarios, e.g., [11] proposed using a learned safe controller, which inherits the same unpredictability as the "performant" controller when operating outside its training distribution. The safe controller can alternatively be designed using optimal control techniques like backward reachability via the Hamilton-Jacobi-Bellman (HJB) equations [12], but these methods rely heavily on accurate dynamics models and often prove computationally prohibitive [13] for complex systems such as bipedal robots. The above methods can be broadly framed under the notion of data-driven *safety filters* [14], i.e., methods that modulate nominal signals to ensure safety in a data-driven context.

The concept of a safety filter originated with control

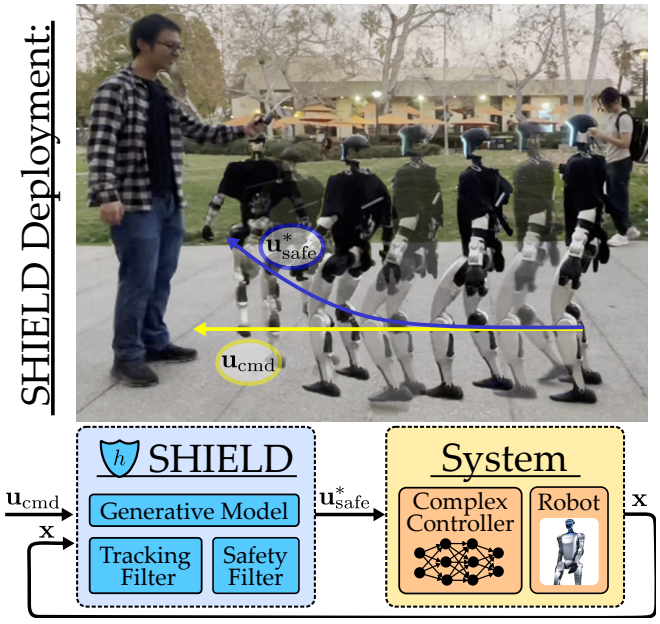

Fig. 1. A humanoid robot implementing the SHIELD architecture autonomously avoids collision with a human using onboard sensing. SHIELD combines a performant underlying controller (e.g., an RL-trained locomotion policy) with a safety layer, which modulates high-level reference signals through a generative model of tracking error trained using real-world trajectory data. This architecture allows safety constraints (like collision avoidance) to be specified and enforced at runtime, with rigorous probabilistic guarantees, even on high-dimensional systems like humanoid robots with complex or "black-box" control policies.

[1]Mechanical and Civil Engineering, California Institute of Technology
[2]Aerospace Engineering and Engineering Mechanics, UT Austin
[3]Computer Science, Cornell University
* represents equal contribution
This research is supported in part by the Technology Innovation Institute (TII), and in part by Dow via project #227027AW.

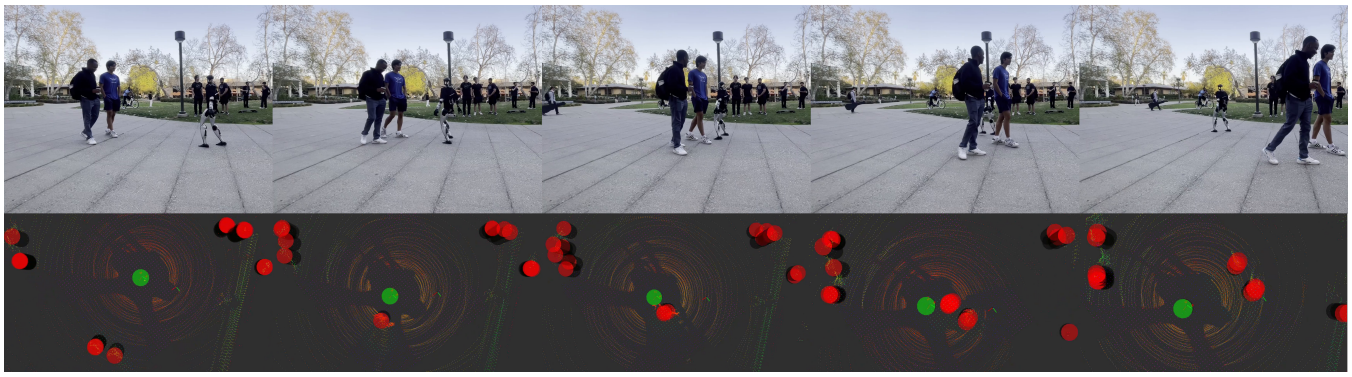

Fig. 2. SHIELD enables real-world pedestrian avoidance with a humanoid robot, using a "general-purpose" RL policy. *Top:* Our robot safely walks among pedestrians using SHIELD's stochastic safety framework. *Bottom:* The robot relies solely on onboard perception to detect and avoid obstacles. Experimental video of this experiment can be found at: https://vimeo.com/1061676063.

barrier functions (CBFs) [15], [16]. This method takes a nominal controller (potentially learning-based) and filters it via the CBF condition to ensure safety framed as forward set invariance. This approach has proven effective on a wide-variety of robotic systems, including quadrupedal and bipedal robots [17], [18]. Yet this approach assumes an accurate model of the system dynamics and environment—this is not available for complex humanoids operating in unstructured environments. To address this, recent work has leveraged reduced-order models in the synthesis of CBF-based safety filters [19], [20], but this requires the underlying assumption of accurate tracking of reference signals.

Contributions. This paper introduces SHIELD, a novel paradigm for guaranteeing safety in robotic systems that bridges the gap between data-driven and model-based safety methods. SHIELD is specifically designed for systems with complex, robust, but ultimately stochastic low-level controllers, such as RL policies used by humanoid robots for locomotion. Unlike traditional safety filters, SHIELD functions as a safety layer that sits "above" the nominal learning-based controller in the autonomy stack (cf. Fig. 1), modulating the reference signal rather than directly filtering control outputs. SHIELD is constructed through a three-step process:

Step 1: Constraint specification. The user specifies a safety constraint on a subset of the robot states (e.g. the pose of the robot torso) mathematically, with positive values corresponding to constraint satisfaction. The low-level policy does not need to be trained to satisfy this constraint but can instead be designed to track general reference commands provided to the reduced-order model (as is typical for RL [2], [5]).

Step 2: Dynamics residual learning. The user collects real-world data of the low-level policy being excuted and trains a conditional variational autoencoder (CVAE) to model the difference between the desired motion of the reduced-order model, and closed-loop system's real-world tracking of these commands. The result is improved reference signal tracking performance.

Step 3: Safety-aware reference generation. The learned residual distribution from the CVAE is used to compute "minimally-invasive" modifications to the reference command that closely track the desired motion of the reduced-order model while satisfying a stochastic discrete-time control barrier function (S-DTCBF) [21], [22] constraint. The result is a formal guarantee of safety in probability: the probability that the system state leaves a specified safe set in a finite specified horizon.

Crucially, in contrast to prior work [23], SHIELD uses the CVAE to both improve the qualitative performance (by achieving better tracking) and thereby, through the layered implementation, enforce user-specified safety constraints. The result is guarantees of safety in probability. We also provide a computationally tractable formulation of the S-DTCBF constraint for obstacle avoidance that is amenable to online computation on embedded robot hardware.

We validate our theoretical framework by implementing our approach on an Unitree G1 Humanoid robot and conducting comprehensive experiments. We first model the robot as a planar single integrator system, train an RL policy to track reference linear and yaw rates, and define obstacle avoidance constraints for these states using onboard perception (Step 1). We then train a CVAE to model these disturbances using obstacle-free locomotion data (Step 2). Online, we deploy the controller using a stochastic DTCBF with the generative dynamics residual (Step 3), which modulates the inputs to the RL walking policy. In controlled experiments, SHIELD consistently outperforms traditional DTCBF methods, as the robot tracks velocity commands while avoiding obstacles using only onboard perception. Finally, to demonstrate real-world applicability, we successfully deploy our system in unstructured outdoor environments (see Fig. 2) where the robot navigates safely around humans.

II. BACKGROUND

In this work we consider robots that can be modeled as discrete time dynamical systems of the form:

$$\mathbf{s}_{k+1} = \mathbf{\Phi}(\mathbf{s}_k, \mathbf{a}_k). \tag{1}$$

where $\mathbf{s}_k \in \mathbb{R}^{n_s}$ is the state of the system and $\mathbf{a} \in \mathbb{R}^{n_a}$ is the system input. This may be the high-dimensional representation of the system where $\mathbf{s}$ includes global pose, joint angles, joint angular velocities, etc. and $\mathbf{a}$ may be joint torques, voltages, etc. For this complex system, we assume

that we have some controller $\boldsymbol{\pi} : \mathbb{R}^{n_s} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_a}$ that takes the current system state $\mathbf{s}$ and user commands $\mathbf{u}$ to produce full-order system inputs $\mathbf{a}$. Using this controller yields:

$$\mathbf{s}_{k+1} = \boldsymbol{\Phi}(\mathbf{s}_k, \boldsymbol{\pi}(\mathbf{s}_k, \mathbf{u}_k)). \tag{2}$$

For navigation purposes, we consider a reduced-order representation of the system $\mathbf{x} \in \mathbb{R}^{n_x}$ where $n_x < n_s$ and $\mathbf{x} = \mathbf{p}(\mathbf{s})$ for some projection $\mathbf{p} : \mathbb{R}^{n_s} \to \mathbb{R}^{n_x}$ that projects the full-order state $\mathbf{s}$ onto the reduced-order state $\mathbf{x}$. Here $\mathbf{x}$ may be the outputs of the system that are considered in safety and navigation, such as its center of mass position. Similarly, we consider the reduced-order inputs to the system.

We can then represent the discrete-time dynamics of this reduced-order model of the system as:

$$\mathbf{x}_{k+1} = \mathbf{p}(\boldsymbol{\Phi}(\mathbf{s}_k, \boldsymbol{\pi}(\mathbf{s}_k, \mathbf{u}_k))) \tag{3}$$
$$\approx \mathbf{F}(\mathbf{x}_k) + \mathbf{G}(\mathbf{x}_k)\mathbf{u}_k + \mathbf{d}_k \tag{4}$$

where $\mathbf{F}(\mathbf{x}_k)+\mathbf{G}(\mathbf{x}_k)\mathbf{u}_k$ represents a simplified model of the system and $\mathbf{d}$ is the difference between the full-order model and this reduced order model, also called the dynamics residual. To capture the complexities of the full-order dynamics $\boldsymbol{\Phi}$ and the controller $\boldsymbol{\pi}$, we consider $\mathbf{d}_k$ to be a random disturbance sampled from a distribution $\mathcal{D}(\mathbf{s}_{k:0}, \mathbf{a}_{k:0})$ that is dependent on the history of full states and from time 0 through k, denoted as $\mathbf{s}_{k:0}$ and $\mathbf{a}_{k:0}$ respectively.

Safety in Probability. To consider the safety of this stochastic RL-guided system, we consider a system to be safe as long as its state is in a user defined *safe set* $\mathcal{C} \subset \mathbb{R}^{n_x}$. Due to the stochasticity of our system, it may not be possible to guarantee with complete certainty that our system will remain in $\mathcal{C}$ for all time [24, Sect. IV]. Instead, we look to bound finite-time safety probability as a metric for system safety, as is common in the stochastic safety literature [22], [25], [26].[1]

Definition 1 (K-step Exit Probability)**.** *For any $K \in \mathbb{N}_1$ and initial condition $\mathbf{x}_0 \in \mathbb{R}^n$, the K-step exit probability of the set $\mathcal{C}$ for a feedback controller $\mathbf{u}_k = \mathbf{K}(\mathbf{x}_k)$ applied to the system* (3) *is:*

$$P_{\mathbf{u}}(K, \mathbf{x}_0) = \mathbb{P}\left\{\mathbf{x}_k \notin \mathcal{C} \text{ for some } k \leq K\right\} \tag{5}$$

Stochastic Safety Filters. To enforce a bound on this K-step exit probability, we first define the safe set $\mathcal{C}$ as the 0-superlevel set of some function $h : \mathbb{R}^{n_x} \to \mathbb{R}$:

$$\mathcal{C} = \{\mathbf{x} \in \mathbb{R}^{n_x} \mid h(\mathbf{x}) \geq 0\}. \tag{6}$$

Using this definition of safety, the field nominally considers the classical discrete-time control barrier function (DTCBF) inequality to enforce safety: for $\alpha \in (0, 1)$ [28]:

$$h(\mathbf{x}_{k+1}) \geq \alpha h(\mathbf{x}_k). \tag{DTCBF}$$

[1]Given the discrete nature of our problem formulation, we focus exclusively on safety at sample times as in [22]. We refer to [27] for an analysis of inter-sample safety.

However, given the stochastic nature of our system, we instead consider the following stochastic discrete-time control barrier function (S-DTCBF) inequality to enforce safety guarantees on our system:

$$\mathbb{E}[\ h(\mathbf{x}_{k+1}) \mid \mathscr{F}_k\ ] \geq \alpha h(\mathbf{x}_k) \tag{S-DTCBF}$$

where $\mathscr{F}_k \triangleq \{\mathbf{s_k}, \mathbf{s_{k-1}}, ..., \mathbf{s_0}, \mathbf{a_k}, \mathbf{a_{k-1}}, ..., \mathbf{a_0}\}$. That is, S-DTCBF is the traditional DTCBF condition in expectation.

This constraint has been shown to provide bounds on the K-step exit probability (5) in a variety of contexts [22], [29] and is often enforced on the system in the form of the following safety filter:

$$\begin{aligned} \mathbf{u}_k^* = \underset{\mathbf{u} \in \mathcal{U}}{\operatorname{argmin}} \quad & \|\mathbf{u} - \mathbf{k}_{\text{nom}}(t)\| \\ \text{s.t.} \quad & \mathbb{E}[\ h(\mathbf{x}_{k+1}) \mid \mathscr{F}_k] \geq \alpha h(\mathbf{x}_k). \end{aligned} \tag{7}$$

Unlike standard applications of CBFs, this optimization problem may be computationally complex and non-convex. Thus, modifications involving Jensen's inequality and generative modeling can be made to improve computational efficiency for hardware applications [22], [23].

In this work, we specifically consider the probability bounds as generated by:

Theorem 1 (Freedman's Inequality for Stochastic Safety [29, Thm. 3])**.** *If, for some $K \in \mathbb{N}_1$, $\sigma > 0$ and $\delta > 0$, the following bounds on the difference between the true and predictable update and the conditional variance hold for all $k \leq K$:*

$$\text{Var}(h(\mathbf{x}_{k+1}) \mid \mathscr{F}_k) \leq \sigma^2 \tag{8}$$
$$\mathbb{E}[h(\mathbf{x}_k) \mid \mathscr{F}_{k-1}] - h(\mathbf{x}_k) \leq \delta \tag{9}$$

and the dynamics are constrained as in (S-DTCBF) *for some $\alpha \in (0, 1)$, then the K-step exit probability is bounded as:*

$$P_u(K, \mathbf{x}_0) \leq e^{\frac{\alpha^K h(\mathbf{x}_0)}{\delta}} \left(\frac{\sigma^2 K}{\lambda}\right)^{\frac{\lambda}{\delta^2}} \tag{10}$$

where $\lambda = \alpha^K h(\mathbf{x}_0)\delta + \sigma^2 K$.

To apply this theorem, we require two assumptions: first, a bound on the safety variance as in (8); second, a bound on the difference between the true safety value $h(\mathbf{x}_k)$ and the expected value as in (9). The first assumption is not very restrictive and allows for a large class of potential functions h, dynamics, and disturbance distributions. The second assumption is more restrictive, but generally applies in our setting, as the worst-case falling behavior would lead to a bounded difference between the commanded and true reduced-order-model behavior.

III. Disturbance Learning

While theoretical frameworks such as Freedman's inequality (Thm. 1) provide powerful methods for analyzing and synthesizing risk-aware controllers, their guarantees fundamentally depend on accurate characterization of the disturbance distribution $\mathcal{D}$. Rather than assuming that this distribution is known a priori or constrained to a simplified

parametric form (e.g., additive Gaussian noise), we propose a data-driven approach, based on [23], that leverages deep generative modeling to learn these distributions directly from empirical trajectories of the system. This approach enables us to capture complex, non-Gaussian, and state-dependent uncertainty patterns that more faithfully represent the actual disturbances encountered during hardware operation.

Conditional Variational Inference. To account for the dynamics residual, we seek to train a generative model to approximate the dynamics residual distribution. To do this, we first collect a dataset of state, command, and disturbance tuples $\mathfrak{D} = \{(\mathbf{x}_i, \mathbf{u}_i, \mathbf{d}_i)\}_{i=1}^{N}$. We then train a Conditional Variation Autoencoder (CVAE) [30] on this dataset, which yields a generative disturbance model $p_\theta(\mathbf{d}_k|\mathbf{x}_{k:k-N}, \mathbf{u}_{k:k-N})$. In contrast to previous work [23], the model is conditioned on a context window of length $N \in \mathbb{N}$, to allow the model to better capture temporal effects such as higher state derivatives or time delays. We find that providing this context greatly boosts modeling accuracy for a complex system like a humanoid robot (Sec. V). Note that the input $\mathbf{u}_i$ here is the unfiltered command, meaning we do not need to solve the algebraic loop of the filtered input (11) being a function of itself or its own history.

We note that any class of generative disturbance model (e.g., diffusion [31], flow matching [32]) can be used with our proposed safety framework (Sec. IV) - for SHIELD we choose to use CVAEs due to their expressivity and fast inference time, as shown empirically in [23].

Stochastic Tracking with Learned Disturbance. SHIELD distinguishes itself from conventional safety layers through how it modulates control signals. While traditional approaches [16], [23], [33] operate by modifying low-level signals (such as joint torques or raw actuation commands) to maintain safety, SHIELD instead modulates higher-level signals, i.e., the reference commands provided to the reduced-order model. This architecture is similar to that of a *reference governor* [34], which modulates reference or command signals into the controller/plant; the key difference is we modulate these signals with a CBF and without knowledge of the actual controller and plant dynamics. This modification enables the definition of safety constraints on simpler, more semantically meaningful states, making the system both more interpretable and manageable.

SHIELD recognizes that the ultimate objective is to achieve the intended system behavior, meaning the system accurately tracks the reduced-order model's trajectory. To derive this "best-tracking" control, we define the optimal control as minimizing the expected difference between the next state of the reduced-order model under the desired command, and the next state of the actual system:

$$\mathbf{u}_k^* = \underset{\mathbf{u}_k \in \mathcal{U}}{\operatorname{argmin}} \ \mathbb{E}[||\overline{\mathbf{x}}_{k+1} - (\mathbf{F}(\mathbf{x}_k) + \mathbf{G}(\mathbf{x}_k)\mathbf{u}_k + \mathbf{d}_k)||^2 | \mathscr{F}_k]$$

where $\overline{\mathbf{x}}_{k+1}$ is the desired next position. Assuming pseudo-

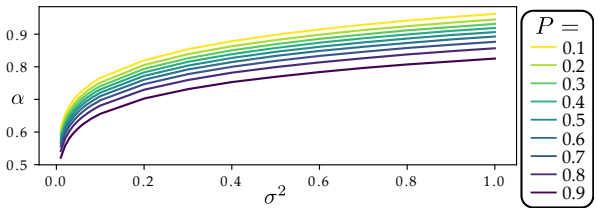

Fig. 3. Higher $\alpha = L(P, K = 10, h(x_0) = 10, \delta = 0.01, \sigma)$ values correspond to more conservative behavior, this increased conservatism a consequence of a lower K-step exit probability or a higher variance.

invertibility of $\mathbf{G}(\mathbf{x}_k)$, the optimal $\mathbf{u}$ is[2]:

$$\mathbf{u}_k^* = \mathbf{G}^\dagger(\mathbf{x}_k)(-\mathbf{F}(\mathbf{x}_k) + \overline{\mathbf{x}}_{k+1} - \mathbb{E}[\mathbf{d}_k|\mathscr{F}_k]). \tag{11}$$

However, since we do not have access to the true expectation $\mathbb{E}[\mathbf{d}_k|\mathscr{F}_k]$, we approximate this with the learned expectation computed from samples generated by the CVAE:

$$\mathbf{u}_k^* = \mathbf{G}^\dagger(\mathbf{x}_k)(-\mathbf{F}(\mathbf{x}_k) + \overline{\mathbf{x}}_{k+1} - \mathbb{E}_{p_\theta}[\mathbf{d}_k|\mathbf{x}_{k:k-N}, \mathbf{u}_{k:k-N}]).$$

This $\mathbf{u}_k^*$ uses the learned disturbance distribution to select the command which reduces the mean squared error to the desired next state $\overline{\mathbf{x}}_k$.

Safety with Learned Disturbance. In addition to using the learned dynamics residual to improve tracking, we can also use it to improve safety. To do this, we select a maximum allowable risk level $P \in (0, 1)$. Given the horizon length $K \in \mathbb{N}$, the initial safety value $h(\mathbf{x}_0)$, the step-wise bound δ from assumption (9), and the variance bound σ from assumption (8) we can solve for the α that will result in the desired risk level bound P:

$$\alpha = L(P, K, h(x_0), \delta, \sigma) \tag{12}$$

In practice we approximate $L : (0, 1) \times \mathbb{N} \times \mathbb{R}_{>0} \times \mathbb{R}_{>0} \times \mathbb{R}_{>0} \to (0, 1)$ numerically due to the complexity of the analytic solution. Evaluations of the function for different α values for a range of P and σ can be found in Fig. 3.

To apply Theorem 1, we address assumption (8) and (9) in turn, and how they apply to our application

1) For assumption (8), the variance bound σ^2 is approximated from the sampled dataset $\mathfrak{D}$
2) For assumption (9), we derive a bound from our application to bipedal robots. In this case, we can bound our difference between the true and predicted update for $h(\mathbf{x}_k)$ based upon the maximum step distance which can be measured in practice:

$$\delta \triangleq 2(h(\mathbf{x}_{\text{footstep } k}) - h(\mathbf{x}_{\text{footstep } k+1})) \tag{13}$$

In addition to meeting assumptions (8) and (9), we must also enforce the S-DTCBF inequality, which we incorporate

[2] The derivation of this follows from the equality $\mathbb{E}[||\overline{\mathbf{x}}_{k+1} - (\mathbf{F}(\mathbf{x}_k) + \mathbf{G}(\mathbf{x}_k)\mathbf{u}_k + \mathbf{d}_k)||^2|\mathscr{F}_k] = ||\overline{\mathbf{x}}_{k+1} - (\mathbf{F}(\mathbf{x}_k) + \mathbf{G}(\mathbf{x}_k)\mathbf{u}_k + \mathbb{E}[\mathbf{d}_k|\mathscr{F}_k])||^2 + \mathbb{E}[||\mathbf{d}||^2|\mathscr{F}_k] - ||\mathbb{E}[\mathbf{d}|\mathscr{F}_k]||^2$. Since this is true, it suffices to find the optimal $\mathbf{u}$ for $||\overline{\mathbf{x}}_{k+1} - (\mathbf{F}(\mathbf{x}_k) + \mathbf{G}(\mathbf{x}_k)\mathbf{u}_k + \mathbb{E}[\mathbf{d}_k|\mathscr{F}_k])||^2$ which is (11).

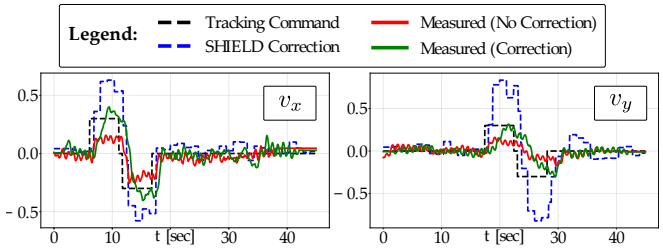

Fig. 4. SHIELD improves tracking performance by correcting learned disturbances. After applying the SHIELD correction as shown by the blue dashed lines, the robot's tracking of the user's intended velocities (shown as a black dashed lines) improves.

as a constraint in a safety filter of the form:

$$\mathbf{u}^*_{\text{safe}} = \underset{\mathbf{u}\in\mathcal{U}}{\text{argmin}} \quad \|\mathbf{u}-\mathbf{u}^*\| \tag{14}$$
$$\text{s.t.} \quad \mathbb{E}[h(\mathbf{F}(\mathbf{x}_k)+\mathbf{G}(\mathbf{x}_k)\mathbf{u}_k+\mathbf{d}_k)|\mathscr{F}_k] \geq \alpha h(\mathbf{x}_k)$$

To enforce this constraint, we need to be able to quickly evaluate or lower bound the expectation $\mathbb{E}[h(\mathbf{F}(\mathbf{x}_k) + \mathbf{G}(\mathbf{x}_k)\mathbf{u}_k + \mathbf{d}_k)|\mathscr{F}_k]$. To do this for concave h functions, we employ Jensen's inequality as in [22] to arrive at the following inequality:

Proposition 1 (Probabilistic Invariance with Concave Safety Functions [22, Lem. 1])**.** *Consider a twice-continuously differentiable, concave function* $h : \mathbb{R}^{n_x} \to \mathbb{R}$ *with* $\sup_{\mathbf{x}\in\mathbb{R}^{n_x}} \|\nabla^2 h(\mathbf{x})\|_2 \leq \lambda_{\max}$ *for some* $\lambda_{\max} \in \mathbb{R}_{\geq 0}$*, and a random variable* $\mathbf{x}$ *that takes values in* $\mathbb{R}^{n_x}$ *with* $\mathbb{E}[\|\mathbf{x}\|_2] < \infty$ *and* $\|\text{cov}(\mathbf{x})\| < \infty$*. This function* h *and random variable* $\mathbf{x}$ *satisfy:*

$$\mathbb{E}[h(\mathbf{x})] \geq h(\mathbb{E}[\mathbf{x}]) - \frac{\lambda_{\max}}{2}\text{tr}(\text{cov}(\mathbf{x})). \tag{15}$$

This allows us to enforce the (S-DTCBF) for concave, continuously differentiable h indirectly by instead enforcing the tightened constraint:

$$h(\mathbf{F}(\mathbf{x}_k)+\mathbf{G}(\mathbf{x}_k)\mathbf{u}_k+\mathbb{E}_{p_\theta}[\mathbf{d}_k|\mathbf{x}_{k:0},\mathbf{u}_{k:0}]) - \frac{\lambda_{\max}}{2}\text{tr}(\text{cov}_{p_\theta}(\mathbf{d}_k|\mathbf{x}_{k:0},\mathbf{u}_{k:0}) \geq \alpha h(\mathbf{x}_k) \tag{16}$$

where we can approximate $\mathbb{E}[\mathbf{d}_k|\mathscr{F}_k]$ and $\text{cov}(\mathbf{d}_k|\mathscr{F}_k)$ using the learned dynamics residual distribution $p_\theta(\mathbf{d}_k|\mathbf{x}_{k:0},\mathbf{u}_{k:0})$.

In summary, SHIELD uses the learned dynamics residual distribution from the CVAE to compute two distinct quantities: (1) an optimal input that minimizes the expected tracking error between the true system and desired next state, and (2) a minimal adjustment to this input that enforces probabilistic safety constraints. We emphasize that these components are fully modular. The tracking-optimized input can be used independently to reduce the sim-to-real gap, while the safety adjustment can be applied separately to enhance real-world safety guarantees. Alternatively, both components can be combined sequentially to simultaneously improve tracking performance and safety assurance, providing flexibility for different application requirements.

IV. Dynamic Obstacle Avoidance on Stochastic Reduced-Order Models

In this section, we detail our approach to improve tracking and safety on a bipedal robot operating under random disturbances with a stochastic reinforcement learning-based controller $\boldsymbol{\pi}$. In particular, we use a PPO Actor-Critic learned controller $\boldsymbol{\pi}_{\text{PPO}}$. This takes into account histories of proprioceptive and extereoceptive states $\mathbf{s}$ and a commanded velocity vector $\mathbf{u} = (v_x, v_y, \omega)$ using an LSTM and uses those to generate joint positions, $\mathbf{a}$.

To characterize the stochasticity of this controller, we use a CVAE to learn the distribution of the dynamics residual $\mathbf{d}$ conditioned on the last four[3] system states and commands, i.e. $(\mathbf{x}_{k:k-3}, \mathbf{u}_{k:k-3})$. Specifically, we use a single integrator system with an additive disturbance as our simplified model:

$$\underbrace{\begin{bmatrix}p_x\\p_y\\\theta\end{bmatrix}_{k+1}}_{\mathbf{x}_{k+1}} = \underbrace{\begin{bmatrix}p_x\\p_y\\\theta\end{bmatrix}_{k}}_{\mathbf{F}(\mathbf{x}_k)} + \underbrace{\Delta_t \mathbf{I}_3}_{\mathbf{G}(\mathbf{x}_k)} \underbrace{\begin{bmatrix}v_x\\v_y\\\omega\end{bmatrix}_{k}}_{\mathbf{u}_k} + \underbrace{\Delta_t \begin{bmatrix}d_x\\d_y\\d_\theta\end{bmatrix}}_{\mathbf{d}_k} \tag{17}$$

where $p_x, p_y \in \mathbb{R}$, $\theta \in [0, 2\pi)$, and $\Delta_t > 0$ represent the x and y position, the yaw angle, and the state-update period and where $\mathbf{d}_k$ is a random disturbance that models the difference between the simplified model and the true dynamics.

Using the Stochastic Tracking method detailed in Section III leads us to the optimal tracking command:

$$\mathbf{u}_{\text{adjusted}} = \frac{\overline{\mathbf{x}}_{k+1} - \mathbf{x}_k}{\Delta_t} - \mathbb{E}_{p_\theta}[\mathbf{d}|\mathbf{x}_{k:k-3},\mathbf{u}_{k:k-3}] \tag{18}$$

where $\mathbb{E}_{p_\theta}[\mathbf{d}|\mathbf{x}_{k:k-3},\mathbf{u}_{k:k-3}]$ is the mean disturbance learned by the CVAE. After modifying the command velocity with the predicted dynamics residual to improve tracking, we apply our safety filter which minimally modifies that command to enforce our safety constraint. For application, we consider obstacle avoidance with respect to $N \in \mathbb{N}$ obstacles as characterized by the signed distance function (sdf):

$$\text{sdf}(\mathbf{x}) = \min_{i\in\{1,\dots N\}} \left\| \begin{bmatrix}p_x\\p_y\end{bmatrix} - \boldsymbol{\rho}_i \right\| - R_i \tag{19}$$

where $\boldsymbol{\rho}_i \in \mathbb{R}^2$ is the planar position of obstacle i and $R_i > 0$ is the robot radius plus the obstacle radius.

To incorporate additional obstacles and reduce chattering oscillation that can occur when the closest obstacle switches, we smooth the SDF collision constraint to be:

$$h_{\text{smooth}}(\mathbf{x}_k) = \lambda(1 - e^{-\gamma \text{sdf}(\mathbf{x}_k)}) \tag{20}$$

where $\lambda > 0$, $\gamma > 0$ are positive constants controlling the maximum magnitude and smoothness of safety.

Since we are only considering the closest obstacle, we make the following concave approximation:

$$\widehat{h}(\mathbf{x}) = \lambda\left(1 - e^{-\gamma((\mathbf{p}-\boldsymbol{\rho}_i)^T\mathbf{e}_i - R_i)}\right) \tag{21}$$

In practice, we condition on the last $N = \min(k, 4)$ states and commands for the algorithm to run at start time

Algorithm 1 SHIELD: Deployment Phase

Initialize $k \leftarrow 0, \mathbf{x} \leftarrow \mathbf{x}_0$
Initialize P, δ, α
while true **do**
  obstacles $\leftarrow \{\boldsymbol{\rho}_1, ..., \boldsymbol{\rho}_M\}$
  $h_k \leftarrow \max_i \tilde{h}(\mathbf{x}, \boldsymbol{\rho}_i)$, $i^* \leftarrow \arg\max_i \tilde{h}(\mathbf{x}, \boldsymbol{\rho}_i)$
  if k modulo $K = 0$ **then**
    $\Sigma \leftarrow \text{cov}_{p_\theta}(\mathbf{d}|\mathbf{x}_{k:k-N}, \mathbf{u}_{k:k-N})$
    $\alpha \leftarrow L(K, h_k, P, \delta, \Sigma)$
  end if
  Get $\mathbf{u}_{\text{cmd}}$ as input
  $\mathbf{u}_{\text{adjusted}} \leftarrow \mathbf{u}_{\text{cmd}} - \mathbb{E}_{p_\theta}[\mathbf{d}|\mathbf{x}_{k:k-3}, \mathbf{u}_{k:k-3}]$
  $\mathbf{e} \leftarrow \frac{\mathbf{p}_k - \boldsymbol{\rho}_{\text{obs},i^*}}{||\mathbf{p}_k - \boldsymbol{\rho}_{\text{obs},i^*}||}$, $\lambda \leftarrow \lambda_{\max}(\boldsymbol{\rho}, \mathbf{e})$
  $\mathbf{u}^*_{\text{safe}} \leftarrow \min_{\mathbf{u}} \|\mathbf{u} - \mathbf{u}_{\text{adjusted}}\|^2$
    s.t. $h(\mathbf{F}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u}) - \frac{\lambda}{2}\mathbf{e}^T\Sigma\mathbf{e} \geq \alpha h_k$
  Apply command $\mathbf{u}^*_{\text{safe}}$, $\mathbf{x_k} \leftarrow \mathbf{x_{k+1}}$, $k \leftarrow k+1$
end while

$$\tilde{h}(\mathbf{x}) = \begin{cases} \widehat{h}(\mathbf{x}), & \text{if } (\mathbf{p} - \boldsymbol{\rho}_i)^\top \mathbf{e}_i \geq 0 \\ \nabla_{\mathbf{x}}\widehat{h}(\mathbf{x}) + \lambda(1 - e^{\gamma R_i}), & \text{else} \end{cases} \tag{22}$$

where $\mathbf{p} \triangleq [p_x, p_y]^T$ and $\mathbf{e}_i \in \mathbb{R}^2$ with $||\mathbf{e}_i||_2 = 1$ is the unit direction towards the closest obstacle from the previous timestep, i.e. $(\mathbf{p}_k - \boldsymbol{\rho}_i)/\|\mathbf{p}_k - \boldsymbol{\rho}_i\|$.

In the case of a single obstacle, we provide the following bound which will allow us to build conditions that enforce a bound on the $K-$step failure probability in practice:

Theorem 2 (Single-Obstacle Avoidance with Concave Barrier Functions)**.** *Consider the function $\tilde{h}$ as in* (22) *with $N = 1$ and a random variable $\mathbf{x}$ that takes values in $\mathbb{R}^{n_x}$ with $\mathbb{E}[\|\mathbf{x}\|_2] < \infty$ and $\|\text{cov}(\mathbf{x})\| < \infty$. This function $\tilde{h}$ and random variable $\mathbf{x}$ satisfy:*

$$\mathbb{E}\left[\tilde{h}(\mathbf{x})\right] \geq \tilde{h}(\mathbb{E}[\mathbf{x}]) - \frac{\lambda_{\max}}{2}\mathbf{e}_1^T \text{cov}(\mathbf{x})\mathbf{e}_1. \tag{23}$$

Please see the appendix of the extended version of this paper for the proof [35].

This allows us to enforce the (S-DTCBF) for concave, continuously differentiable h indirectly by instead enforcing the tightened constraint:

$$h(\mathbf{F}(\mathbf{x}_k) + \mathbf{G}(\mathbf{x}_k)\mathbf{u}_k + \mathbb{E}_{p_\theta}[\mathbf{d}_k|\mathbf{x}_{k:k-3}, \mathbf{u}_{k:k-3}]) \tag{24}$$
$$- \frac{\lambda_{\max}}{2}\mathbf{e}_i^T \text{cov}_{p_\theta}(\mathbf{d}_k|\mathbf{x}_{k:k-3}, \mathbf{u}_{k:k-3})\mathbf{e}_i \geq \alpha h(\mathbf{x}_k)$$

where we can approximate $\mathbb{E}[\mathbf{d}_k|\mathscr{F}_k]$ and $\text{cov}(\mathbf{d}_k|\mathscr{F}_k)$ using the learned dynamics residual distribution $p_\theta(\mathbf{d}_k|\mathbf{x}_{k:k-3}, \mathbf{u}_{k:k-3})$. In practice, we find that the utility of SHIELD generalizes to multiple obstacles; however, we leave a rigorous theoretical analysis of the nonconcave $\tilde{h}$ with multiple obstacles for future work.

To determine the appropriate α to get a desired risk level across K steps, we use the L function in (12). To calculate α, a desired risk level P is chosen, the current safety value is noted as $h(\mathbf{x}_k)$, the worst case δ is approximated as in (13), and the covariance σ is set to the maximum value experienced in the experimental data. Furthermore, to extend the guarantee beyond K steps, we recalculate α every K steps. Thus, each successive K steps satisfies the bound in Prop. 1 and they can be connected using the union bound:

$$\mathbb{P}\left\{\min_{k\in[0,K\times F]} h(\mathbf{x}_k) < 0\right\} \leq \sum_{i=0}^{F} \mathbb{P}\left\{\min_{k\in[Ki,K(i+1)]} h(\mathbf{x}_k) < 0\right\}$$

where F is the number of K-step intervals in the experiment. We show the SHIELD deployment stage in Alg. 1.

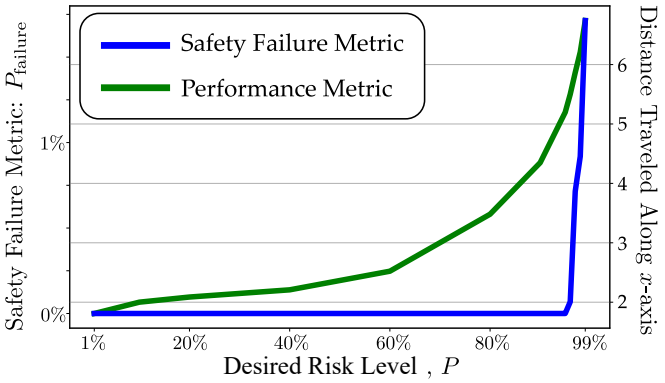

Fig. 5. The trade-off between performance and safety. As the probability of K-step exit increases, we achieve better performance at cost of an increasing amount of safety violation under the proposed metric.

V. Experiments

We demonstrate the validity of SHIELD on a simple simulated system and then on a Unitree G1 humanoid robot, aiming to show the method's adaptable conservativeness, performance, and robustness.

Simulation. We first present a simplified simulation problem consisting of a single integrator disturbed by a 0-mean, multivariate student's t-distribution with clipped tails. We randomize the placement of obstacles and the radii of the obstacles and the robot. The robot moves along the x-direction with a constant commanded velocity of 0.5m/sec, which we filter using SHIELD (1).

For all experiments, we set the discrete time difference $\Delta t = 0.01$ and the scaling factors in the safety function to be $\lambda = 10$, $\gamma = 0.5$, $K = 10$, and the upper bound on the assumption of bounded difference of (9) to be 1.

We then simulate the system over $N_{\text{trials}} = 100$ trials of $N_{\text{steps}} = 2000$ steps each and calculate the percentage of violations P_{failure} as:

$$P_{\text{failure}} = \frac{\sum_{i=1}^{N_{\text{trials}}} \sum_{j=1}^{N_{\text{steps}}} \mathbb{1}_{\{h(\mathbf{x}_{i,j})<0\}}}{N_{\text{trials}} N_{\text{steps}}} \tag{25}$$

and quantify performance as total distance traveled in the commanded direction. We observe that the probability of K-step failure is a tuning knob to encourage more risky behavior at the cost of higher chance of collisions (though still well below the target percentage); see Fig. 5.

Hardware Setup. The Unitree G1 humanoid robot has a height of 1.32 meters and weighs approximately 40kg, with 23 actuated degrees of freedom. We employ an onboard

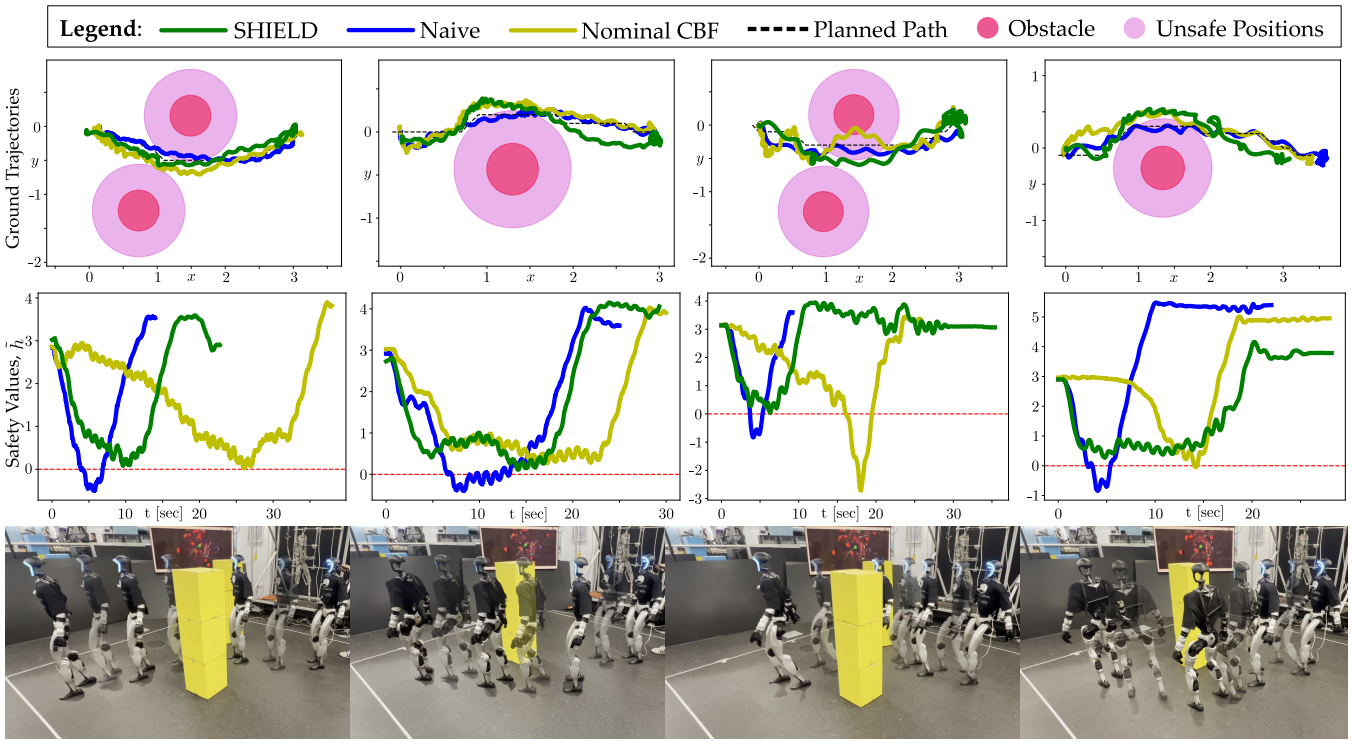

Fig. 6. SHIELD enforces safety in collision avoidance with adaptive conservatism. The A* planner path is not necessarily safe even it does not cross the obstacle, thus naively following the path would result in collisions or scrapes. Nominal CBF, due to not accounting for the inaccurate reduced order model, would also result in collisions or be extremely conservative.

Jetson Orin NX for computation, a Livox Mid-360 LiDAR for sensing the environment, and an Intel T263 to localize the robot. Euclidean clustering [36] is applied to the LiDAR pointcloud to locate obstacles of interest in the scene.

To test the generalization of SHIELD in deployment, we conduct experiments with two different walking controllers:

1) **built-in**: the Unitree built-in controller [37]
2) **custom**: We train a custom RL locomotion controller in IsaacLab [38] using standard rewards from [39].

Approximately 6 minutes of training data are collected for each controller to train the CVAE for both the *built-in* and *custom* controllers. We query the CVAE to update the mean and covariance of the disturbance distribution at 0.83Hz, and we filter the command velocity at 100Hz.

Learned Tracking. We first test the velocity tracking capabilities of the SHIELD framework. In these experiments, we send a pre-set sequence of velocity commands through the framework to the controller and compare our resulting velocities to the command sequence. We achieve noticeable improvements in tracking as shown in Fig. 4.

Obstacle Avoidance. First, we conduct controlled experiments with fixed obstacles. We define success as the robot walking past obstacles without making contact. We model the detected obstacles as cylinders of radius 0.3m and the robot to have a safety margin of 0.38m from the center of mass. To navigate, we first use A* [40] to first plan a path through free space, we then generate nominal velocities by directing the robot from its current position to the next node on the path and filter the commanded velocities with SHIELD. We present both single-obstacle and multi-obstacle cases. In single-obstacle experiments, naively following the A* path alone does not completely avoid obstacles due to state tracking errors. The nominal DTCBF filter, being unaware of the dynamics residual, either collides into the obstacle or exhibits extreme conservative behavior with $\alpha = 0.99$. However, SHIELD enables the robot to completely bypass the obstacle. We observe similar behavior in multi-obstacle scenarios, where SHIELD is able to adjust conservativeness online to only enforce maximum safety conditions when needed, resulting in more dynamic behavior. The results of these experiments can be seen in Fig. 6.

Unstructured Outdoor Environment. We also perform experiments in unstructured outdoor environments for further validation. In these tests, a user provides joystick inputs to the robot for safety reasons and would either control the robot to walk directly towards people or provide no input and let the robot stay in place unless people encroach on its safety boundary. These experiments can be seen in Fig. 1 and Fig. 2 and the experimental video [41].

VI. Conclusion

This paper presented SHIELD: a safety layer that leverages stochastic discrete-time control barrier functions (S-DTCBF) to guarantee safety in probability. Importantly, this can be added to an existing autonomy stack, wherein the dynamics of a nominal controller can be learned as the residual on a simplified model. SHIELD then filters the nominal commands to produce safe inputs as they are sent to the system via an S-DTCBF. This framework is instantiated on a humanoid robot in the context of collision avoidance, where it is shown to outperform a nominal safety filter in hardware

experiments on the Unitree G1 humanoid. This paper, therefore, demonstrates that by combining a general-purpose RL locomotion controller with a robot-specific stochastic safety layer, SHIELD achieves both high-performance walking and robust safety constraint satisfaction under uncertainty on humanoid robots.

APPENDIX

A. Proof of Theorem 2

Proof. Consider the convex, twice differentiable function $\eta : \mathbb{R}^{n_x} \to \mathbb{R}$ defined as $\eta = -\tilde{h}$. By second-order Taylor's theorem, for $\mathbf{x}, \boldsymbol{\mu} \in \mathbb{R}^n$ there exists an $\omega \in (0,1)$ such that:

$$\eta(\mathbf{x}) = \eta(\boldsymbol{\mu}) + \nabla\eta(\boldsymbol{\mu})^T\mathbf{d} + \frac{1}{2}\mathbf{d}^T\nabla^2\eta(\mathbf{c})\mathbf{d} \tag{26}$$

where $\mathbf{d} = \mathbf{x} - \boldsymbol{\mu}$ and $\mathbf{c} = \omega\mathbf{x} + (1-\omega)\boldsymbol{\mu}$. From the construction of $\tilde{h}$, the Hessian of η is :

$$\nabla^2\tilde{h}(\mathbf{x}) = \begin{cases} \varphi(\mathbf{x})\mathbf{e}_1\mathbf{e}_1^\top, & \text{if } (\mathbf{p}-\boldsymbol{\rho}_1)^\top\mathbf{e}_1 \geq 0 \\ 0, & \text{else.} \end{cases} \tag{27}$$

where $\varphi(\mathbf{x}) \triangleq \gamma^2\lambda e^{-\gamma((\mathbf{p}-\boldsymbol{\rho}_1)^\top\mathbf{e}_1 - R_1)}$ which is bounded over the if case $\{\ \mathbf{x} \in \mathbb{R}^{n_x} \mid (\mathbf{p}-\boldsymbol{\rho}_1)^\top\mathbf{e}_1 \geq 0\ \}$, we call this bound $\lambda_{\max} > 0$.

Here $\nabla^2\tilde{h}(\mathbf{x})$ is a diagonalizable, positive semi-definite matrix and when it has a non-zero eigenvalue, the associated eigenvector is equal to $\mathbf{e}_1$. Therefore:

$$\eta(\mathbf{x}) \leq \eta(\boldsymbol{\mu}) + \nabla\eta(\boldsymbol{\mu})^\top\mathbf{d} + \frac{\lambda_{\max}}{2}\mathbf{d}^\top\mathbf{e}_1\mathbf{e}_1^\top\mathbf{d} \tag{28}$$

Next, we follow the proof of [22, Lem. 1] with $\boldsymbol{\mu} = \mathbb{E}[\mathbf{x}]$:

$$\mathbb{E}[\eta(\mathbf{x})] - \eta(\mathbb{E}[\mathbf{x}]) = \int_{\mathbb{R}^{n_x}} (\eta(\mathbf{x}) - \eta(\boldsymbol{\mu}))p(\mathbf{x})d\mathbf{x} \tag{29}$$

$$\leq \int_{\mathbb{R}^{n_x}} \nabla(\boldsymbol{\mu})^\top\mathbf{d} + \frac{\lambda_{\max}}{2}\mathrm{tr}(\mathbf{e}_1^\top\mathbf{d}\mathbf{d}^\top\mathbf{e}_1)p(\mathbf{x})d\mathbf{x} \tag{30}$$

$$= \frac{\lambda_{\max}}{2}\mathrm{tr}(\mathbf{e}_1^\top\mathrm{cov}(\mathbf{x})\mathbf{e}_1) = \frac{\lambda_{\max}}{2}\mathbf{e}_1^\top\mathrm{cov}(\mathbf{x})\mathbf{e}_1. \tag{31}$$

□

REFERENCES

[1] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, Jan. 2019.

[2] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.

[3] G. Feng, H. Zhang, Z. Li, X. B. Peng, B. Basireddy, L. Yue, Z. Song, L. Yang, Y. Liu, K. Sreenath, *et al.*, "Genloco: Generalized locomotion controllers for quadrupedal robots," in *Conference on Robot Learning*. PMLR, 2023, pp. 1893–1903.

[4] Q. Liao, B. Zhang, X. Huang, X. Huang, Z. Li, and K. Sreenath, "Berkeley humanoid: A research platform for learning-based control," *arXiv preprint arXiv:2407.21781*, 2024.

[5] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science robotics*, vol. 7, no. 62, p. eabk2822, 2022.

[6] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath, "Real-world humanoid locomotion with reinforcement learning," *Science Robotics*, vol. 9, no. 89, p. eadi9579, 2024.

[7] K. Zakka, B. Tabanpour, Q. Liao, M. Haiderbhai, S. Holt, J. Y. Luo, A. Allshire, E. Frey, K. Sreenath, L. A. Kahrs, *et al.*, "Mujoco playground," *arXiv preprint arXiv:2502.08844*, 2025.

[8] Z. Zhuang, S. Yao, and H. Zhao, "Humanoid parkour learning," *arXiv preprint arXiv:2406.10759*, 2024.

[9] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe planning in dynamic environments using conformal prediction," *IEEE Robotics and Automation Letters*, 2023.

[10] P. Akella, A. Dixit, M. Ahmadi, J. W. Burdick, and A. D. Ames, "Sample-based bounds for coherent risk measures: Applications to policy synthesis and verification," *Artificial Intelligence*, vol. 336, p. 104195, 2024.

[11] T. He, C. Zhang, W. Xiao, G. He, C. Liu, and G. Shi, "Agile but safe: Learning collision-free high-speed legged locomotion," *arXiv preprint arXiv:2401.17583*, 2024.

[12] "A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on automatic control*, vol. 50, no. 7, pp. 947–957, 2005.

[13] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-jacobi reachability: A brief overview and recent advances," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 2242–2253.

[14] K. P. Wabersich, A. J. Taylor, J. J. Choi, K. Sreenath, C. J. Tomlin, A. D. Ames, and M. N. Zeilinger, "Data-driven safety filters: Hamilton-jacobi reachability, control barrier functions, and predictive methods for uncertain systems," *IEEE Control Systems Magazine*, vol. 43, no. 5, pp. 137–177, 2023.

[15] 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.

[16] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference (ECC)*, 2019, pp. 3420–3431.

[17] R. Grandia, A. J. Taylor, A. D. Ames, and M. Hutter, "Multi-layered safety for legged robots via control barrier functions and model predictive control," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8352–8358.

[18] N. Csomay-Shanklin, R. K. Cosner, M. Dai, A. J. Taylor, and A. D. Ames, "Episodic learning for safe bipedal locomotion with control barrier functions and projection-to-state safety," in *Learning for dynamics and control*. PMLR, 2021, pp. 1041–1053.

[19] T. G. Molnar, R. K. Cosner, A. W. Singletary, W. Ubellacker, and A. D. Ames, "Model-free safety-critical control for robotic systems," *IEEE robotics and automation letters*, vol. 7, no. 2, pp. 944–951, 2021.

[20] M. H. Cohen, T. G. Molnar, and A. D. Ames, "Safety-critical control for autonomous systems: Control barrier functions via reduced-order models," *Annual Reviews in Control*, vol. 57, p. 100947, 2024.

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

[22] R. K. Cosner, P. Culbertson, A. J. Taylor, and A. D. Ames, "Robust safety under stochastic uncertainty with discrete-time control barrier functions," *Robotics: Science and Systems*, 2023.

[23] R. K. Cosner, I. Sadalski, J. K. Woo, P. Culbertson, and A. D. Ames, "Generative modeling of residuals for real-time risk-sensitive safety with discrete-time control barrier functions," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. i–viii.

[24] P. Culbertson, R. K. Cosner, M. Tucker, and A. D. Ames, "Input-to-state stability in probability," in *2023 62nd IEEE Conference on Decision and Control (CDC)*. IEEE, 2023, pp. 5796–5803.

[25] C. Santoyo, M. Dutreix, and S. Coogan, "A barrier function approach to finite-time stochastic system verification and control," *Automatica*, vol. 125, p. 109439, 2021.

[26] J. Steinhardt and R. Tedrake, "Finite-time regional verification of stochastic non-linear systems," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 901–923, 2012.

[27] J. Breeden, K. Garg, and D. Panagou, "Control barrier functions in sampled-data systems," *IEEE Control Systems Letters*, vol. 6, pp. 367–372, 2022.

[28] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation," in *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017.

[29] R. K. Cosner, P. Culbertson, and A. D. Ames, "Bounding stochastic safety: Leveraging freedman's inequality with discrete-time control barrier functions," *IEEE Control Systems Letters*, 2024.

[30] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015.

[31] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851.

[32] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," 2023.

[33] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 6271–6278.

[34] E. Garone, S. Di Cairano, and I. Kolmanovsky, "Reference and command governors for systems with constraints: A survey on theory and applications," *Automatica*, vol. 75, pp. 306–328, 2017.

[35] L. Yang, B. Werner, R. Cosner, D. Fridovich-Keil, P. Culbertson, and A. Ames, "Extended version of this manuscript." [Online]. Available: http://www.rkcosner.com/assets/files/shield.pdf

[36] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China: IEEE, May 9-13 2011.

[37] U. Robotics, "Unitree sdk2," 2024, accessed: 2025-03-01. [Online]. Available: https://github.com/unitreerobotics/unitree_sdk2

[38] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023.

[39] X. Gu, Y.-J. Wang, X. Zhu, C. Shi, Y. Guo, Y. Liu, and J. Chen, "Advancing humanoid locomotion: Mastering challenging terrains with denoising world model learning," in *Robotics: Science and Systems*, 2024.

[40] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[41] "Experimental video." [Online]. Available: https://vimeo.com/1061676063