Towards Dynamical Safety on Humanoids with Stochastic Controllers with SHIELD

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 learned residual model via a stochastic discrete-time CBF formulation enforcing trajectory-long safety constraints in probability. The result is a minimally-invasive safety layer that can be added to the existing autonomy stack to improve user command tracking and provide probabilistic safety guarantees. In hardware experiments on a 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 their interpretability and our ability to provide formal safety guarantees without expensive retraining.

Several methods have emerged in recent years to enable the safe deployment of learning-based controllers. For example, conformal prediction [9], [10], "backup"-style approaches [11], and backward reachability via the Hamilton-Jacobi-Bellman (HJB) equations [12]. However, these approaches either do not resolve the inherent unpredictability of the controller or become computationally intractable.

To address these issues, we introduce the use of safety filters in the form of control barrier functions (CBFs) [13], [14] for these complicated systems. This method takes a nominal controller (potentially learning-based) and filters it via the CBF condition to ensure safety framed as forward set invariance, and has been proven effective on a wide range of robotic systems, including quadrupedal and bipedal robots [15], [16]. As the conventional formulation of CBFs assumes an accurate model of the dynamics and environment of the system, recent work has taken advantage of reduced-order models in the synthesis of CBF-based safety filters [17], [18], but this requires the underlying assumption of accurate tracking of reference signals. In this work we remove this assumption by capturing this error using a generative model and accounting for it through a stochastic safety constraint with trajectory-long guarantees.

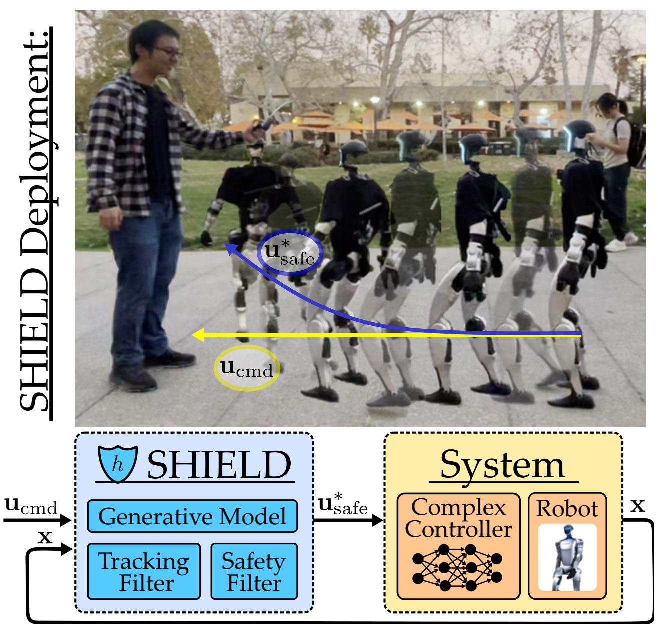

Fig. 1. A humanoid robot implementing the SHIELD architecture autonomously avoids collision with a human using onboard sensing. SHIELD combines a performant underlying controller (e.g., an RL-trained locomotion policy) with a safety layer, which modulates high-level reference signals through a generative 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.

Contributions. This paper introduces SHIELD, a novel paradigm to guarantee 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,

[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), BP p.l.c., and by Dow via project #227027AW.

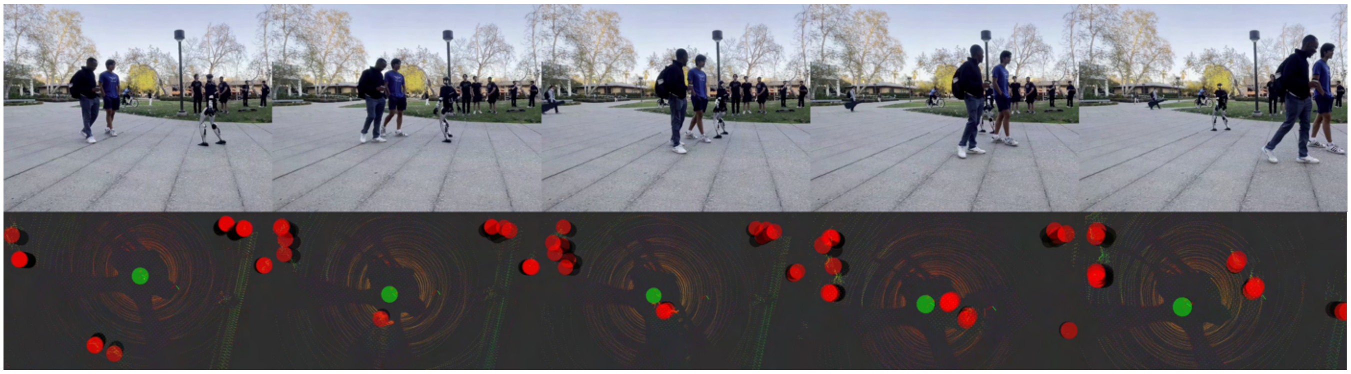

Fig. 2. SHIELD enables real-world pedestrian avoidance with a humanoid robot, using a "general-purpose" RL policy. *Top:* Our robot safely walks among pedestrians using SHIELD's stochastic safety framework. *Bottom:* The robot relies solely on onboard perception to detect and avoid obstacles. Experimental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

such as RL policies used by humanoid robots for locomotion. Unlike traditional safety filters, SHIELD functions as a safety layer that sits "above" / "before" the nominal learning-based controller in the autonomy stack (cf. Fig. 1), modulating the commanded reference signal rather than directly filtering control outputs. SHIELD is constructed over three steps:

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.

Step 3: Safety-aware reference generation. The learned residual distribution is then used to compute modifications to the reference command that improve the tracking of the desired motion of the reduced-order model while satisfying a stochastic discrete-time control barrier function (S-DTCBF) [19] constraint.The result is a formal, trajectory-long guarantee of safety in probability.

Crucially, in contrast to previous work [20], SHIELD uses the learned residual to improve tracking performance *in addition* to providing probabilistic safety guarantees.

II. Problem Statement

Consider robots (e.g., humanoids) 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 $\mathbf{s} \in \mathbb{R}^{n_s}$ is the state of the system and $\mathbf{a} \in \mathbb{R}^{n_a}$ is the input of the system. 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 state of the system $\mathbf{s}$ and user commands $\mathbf{u}$ to produce full-order system inputs $\mathbf{a}$. Using this controller yields the closed-loop system:

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

Due to the complexity of this system, it may be difficult to design safety specifications and an actionable control formulation. To mitigate this, 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 the safety specification. Thus, we can represent the discrete-time dynamics of this reduced-order model of the system as:

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

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 $\mathbf{\Phi}$ 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. Thus, our tasks are to characterize $\mathcal{D}$, use that characterization to construct a safety specification using stochastic discrete-time CBFs as in [19], [21], [22], and deploy that constraint to make safety guarantees for the robotic system.

III. SHIELD Framework

To address the tasks above, we construct the SHIELD framework in Fig. 1 that learns the disturbance, corrects the tracking and enforces discrete-time stochastic safety.

A. Disturbance learning and tracking correction

While theoretical frameworks for probabilistic safety (e.g., [23, Thm. 3]) provide powerful methods for analyzing and synthesizing risk-aware controllers, their guarantees depend on accurate characterization of the disturbance distribution $\mathcal{D}$. We propose a data-driven approach, based on [20], that

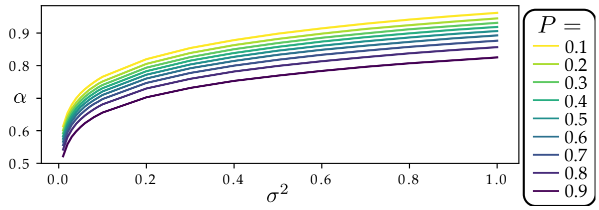

Fig. 3. Higher α values correspond to more conservative behavior, this increased conservatism a consequence of a lower K-step exit probability or a higher variance.

leverages generative modeling to learn these distributions directly from empirical trajectories of the system.

Conditional Variational Inference. We collect a dataset of state, command, and disturbance tuples $\mathfrak{D} = \{(\mathbf{x}_i, \mathbf{u}_i, \mathbf{d}_i)\}_{i=1}^N$ and train a Conditional Variation Autoencoder (CVAE) [24] on this dataset, which yields a generative disturbance model $p_\theta(\mathbf{d}_k | \mathbf{x}_{k:k-N}, \mathbf{u}_{k:k-N})$. We extend the method presented in [20] by conditioning the model on a context window of length $N \in \mathbb{N}$ to better capture temporal effects such as higher state derivatives or time delays, which greatly boosts modeling accuracy for a complex system like a humanoid robot (Sec. IV). Note that the input $\mathbf{u}_i$ here is the unfiltered user command, meaning we do not need to retrain the CVAE episodically.

Stochastic Tracking with Learned Disturbance. To improve the tracking of the intended system behavior, we define the following optimization problem that minimizes the expected difference between the next state of the reduced-order model under the desired command and the next state of the actual system:

$$\mathbf{u}_k^* = \mathrm{argmin}_{\mathbf{u}_k \in \mathcal{U}} \;\; \mathbb{E}[||\overline{\mathbf{x}}_{k+1} - (\mathbf{F}(\mathbf{x}_k) + \mathbf{G}(\mathbf{x}_k)\mathbf{u}_k + \mathbf{d}_k)||^2 | \mathbf{x}_{k:0}, \mathbf{u}_{k:0}]$$

where $\overline{\mathbf{x}}_{k+1}$ is the desired next position. Assuming pseudo-invertibility of $\mathbf{G}(\mathbf{x}_k)$, the optimal $\mathbf{u}_k$ is:

$$\mathbf{u}_k^* = \mathbf{G}^\dagger(\mathbf{x}_k)(-\mathbf{F}(\mathbf{x}_k) + \overline{\mathbf{x}}_{k+1} - \mathbb{E}[\mathbf{d}_k | \mathbf{x}_{k:0}, \mathbf{u}_{k:0}]). \quad (5)$$

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 $\overline{\mathbf{x}}_k$.

B. Safety with Learned Disturbance

With the learned dynamics residual, we use it to improve safety. To do this, we select a maximum allowable risk level $P \in (0, 1)$, and solve for the α that will result in the desired risk level bound P [23, Thm. 3] as shown in Fig. 3 given the horizon length $K \in \mathbb{N}$, the initial safety value $h(\mathbf{x}_0)$, the step-wise bound δ, and the variance bound σ:

$$\alpha = L(P, K, h(x_0), \delta, \sigma) \quad (6)$$

Algorithm 1 SHIELD: Deployment Phase

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

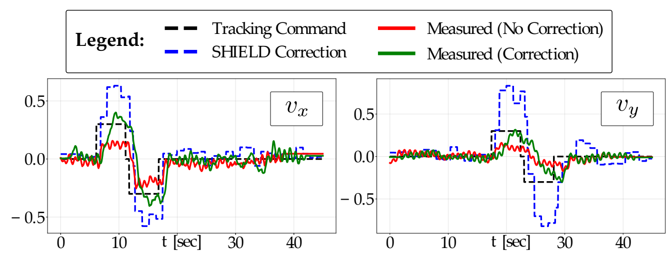

Fig. 4. SHIELD improves tracking performance by correcting learned disturbances. After applying the SHIELD correction as shown by the blue dashed lines, the robot's tracking of the user's intended velocities (shown as a black dashed lines) improves.

Thus we enforce the S-DTCBF condition by incorporating it as a constraint in a safety filter of the form:

$$\mathbf{u}^*_{\mathrm{safe}} = \underset{\mathbf{u} \in \mathcal{U}}{\mathrm{argmin}} \quad \|\mathbf{u} - \mathbf{u}^*\| \quad (7)$$
$$\text{s.t.} \quad \mathbb{E}[h(\mathbf{F}(\mathbf{x}_k) + \mathbf{G}(\mathbf{x}_k)\mathbf{u}_k + \mathbf{d}_k) | \mathbf{x}_{k:0}, \mathbf{u}_{k:0}] \geq \alpha h(\mathbf{x}_k)$$

To apply the framework to our task of obstacle avoidance, we characterize $N \in \mathbb{N}$ obstacles perceived by the robot with the signed distance function (sdf):

$$\mathrm{sdf}(\mathbf{x}) = \min_{i \in \{1, \ldots N\}} \left\| \begin{bmatrix} p_x \\ p_y \end{bmatrix} - \boldsymbol{\rho}_i \right\| - R_i \quad (8)$$

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

$$h(\mathbf{x}_k) = \lambda(1 - e^{-\gamma \mathrm{sdf}(\mathbf{x}_k)}) \quad (9)$$

where $\lambda > 0$, $\gamma > 0$ are positive constants controlling the maximum magnitude and smoothness of safety. The entire procedure can be seen in Algorithm 1, where $\mathbf{e}$ is the unit direction towards the closest obstacle.

IV. Experiments

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

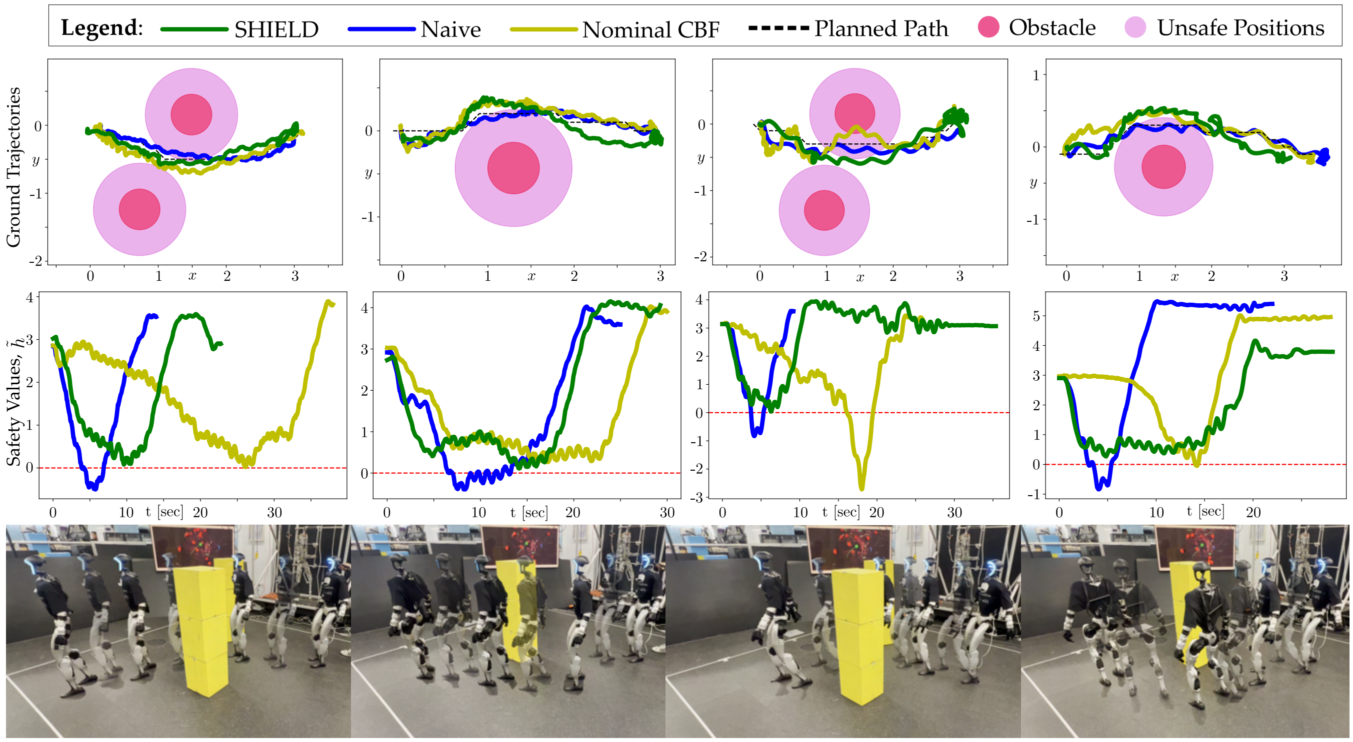

Fig. 5. SHIELD enforces safety in collision avoidance with adaptive conservatism. The A* planner path is not necessarily safe even it does not cross the obstacle, thus naively following the path would result in collisions or scrapes. Nominal CBF, due to not accounting for the inaccurate reduced order model, would also result in collisions or be extremely conservative.

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 Jetson Orin NX for computation, a Livox Mid-360 LiDAR for sensing the environment, and an Intel T263 to localize the robot. Euclidean clustering [25] 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 [26]
2) **custom**: We train a custom RL locomotion controller in IsaacLab [27] using standard rewards from [28].

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* [29] 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. 5.

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 [30].

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] 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.
[14] 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.
[15] 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.
[16] 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.
[17] 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.
[18] 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.
[19] 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.
[20] 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.
[21] 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.
[22] 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.
[23] 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.
[24] 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.
[25] 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.
[26] U. Robotics, "Unitree sdk2," 2024, accessed: 2025-03-01. [Online]. Available: https://github.com/unitreerobotics/unitree_sdk2
[27] 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.
[28] 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.
[29] 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.
[30] "Experimental video." [Online]. Available: https://vimeo.com/1061676063