

SHOES: Sim-to-real Hardware-adaptation via On-policy Expansion of Skills for Legged Robots

Kota Tsuruhara^{1,2} Tsung-Yuan Chen² Kento Kawaharazuka²

¹Tokyo University of Agriculture and Technology ²The University of Tokyo
{trhrkota, chentsungyuan}@g.ecc.u-tokyo.ac.jp
kawaharazuka@jsk.imi.i.u-tokyo.ac.jp

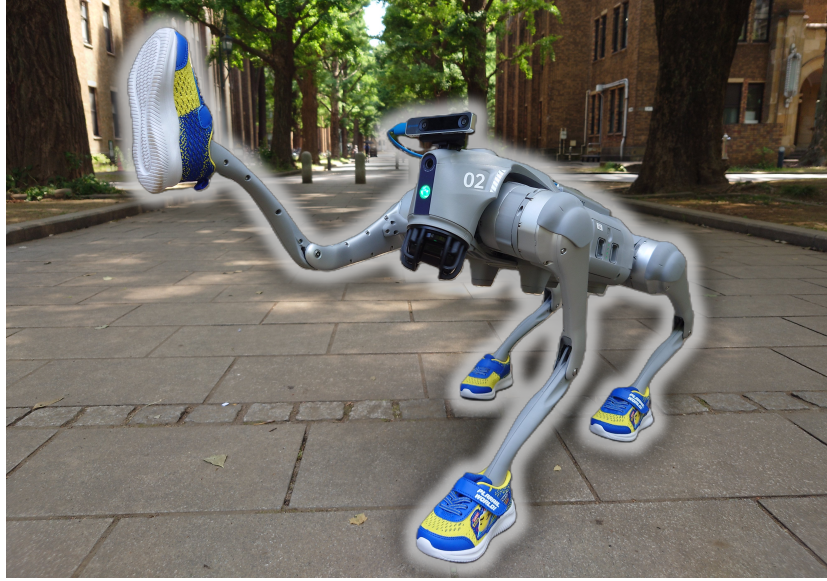


Figure 1: A Unitree Go2 wearing shoes.

Abstract: Humans expand their locomotion capability by wearing appropriate footwear, such as hiking boots for mountains or skates for ice. Footwear changes the ground-contact interface and enables traversal of otherwise difficult terrain. In contrast, locomotion controllers are typically trained for a fixed morphology, and their performance can degrade when shoes alter foot geometry, mass, and friction. We propose **SHOES**, a sample-efficient framework for adapting locomotion policies to footwear-induced morphology changes. SHOES freezes a simulation-trained nominal policy and fine-tunes only lightweight, identity-initialized modulation modules on the robot, preserving the initial behavior while enabling efficient hardware adaptation. We also introduce **SafeBox**, a workspace-aware waypoint sampler that constrains online adaptation within a user-defined workspace. The simulation results with a shoe-equipped Unitree Go2 show that SHOES substantially recovers locomotion performance after footwear-induced morphology changes using limited additional adaptation. Under the same adaptation budget, SHOES outperforms FiLM-only, Linear Adapter-only, and residual-RL baselines. Real-world experiments on a Unitree Go2 fitted with human shoes further evaluate the effectiveness of SHOES for hardware deployment.

Keywords: Legged Locomotion, Online Adaptation, Sim-to-Real, FiLM, Linear Adapter, Safe Online RL

1 Introduction

Humans expand their locomotion capabilities by changing footwear, from hiking boots to skates. Legged robots could similarly benefit from shoes that improve traction or protect floors and hardware (Figure 1). However, footwear changes the robot’s morphology and its interaction with the ground. It alters the foot-contact geometry, frictional behavior, compliance, mass distribution, and effective leg geometry. A policy trained for the bare-foot robot can therefore degrade when the same controller is executed with shoes attached. Figure 2 shows this failure mode on a Unitree Go2: attaching shoes changes the robot’s locomotion characteristics, and the bare-foot nominal policy becomes particularly poor at tracking backward commands.

Domain randomization (DR) improves robustness to deployment-time mismatch by varying simulation parameters during training [1, 2, 3]. Footwear-aware DR requires specifying variations in shoe geometry, contact properties, mass, and inertia in advance. Although this can improve zero-shot robustness, broader randomization increases modeling and training costs and can induce conservative behavior.

Another option is to fine-tune the full locomotion policy directly on hardware [4]. Here, we instead preserve the simulation-trained controller and learn only the correction required by the attached shoes.

This work develops such a post-training adaptation approach for footwear-induced morphology changes. Our main contributions are:

- We propose **SHOES** (Sim-to-real Hardware-adaptation via On-policy Expansion of Skills), a lightweight framework for footwear-induced morphology adaptation that preserves a simulation-trained nominal actor and fine-tunes only identity-initialized FiLM and Linear Adapter modules.
- We introduce SafeBox, a workspace-aware waypoint sampler that supports on-robot adaptation by collecting data under a consistent command interface in simulation and on hardware.
- We evaluate SHOES in simulation and real-robot experiments, showing that it substantially recovers bare-foot tracking performance on shoe-equipped Unitree Go2 robots and outperforms FiLM-only, Linear Adapter-only, and residual-RL baselines.

2 Related Work

Domain randomization for legged sim-to-real. DR is a standard recipe for improving sim-to-real robustness in legged locomotion by training policies over randomized dynamics, contact, terrain, and sensing parameters. It has been widely deployed in quadrupedal sim-to-real RL under diverse deployment conditions [5, 6, 7, 8, 9, 10] and for more advanced quadruped capabilities [11, 12, 13, 14, 15], and similarly underpins bipedal and humanoid sim-to-real RL across diverse platforms and skills [16, 17, 18, 19]. A long line of Cassie work has further developed DR-trained skill-specific controllers [20, 21, 22, 23], following an earlier non-DR iterative-design baseline [24]. Rudin et al. [25] made this recipe practical through massively parallel training, while Liu et al. [26] extends it to procedurally generated robots with long-context in-context adaptation. SHOES is complementary

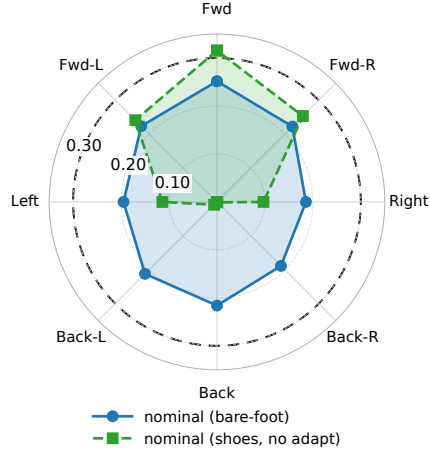


Figure 2: **Bare-foot nominal degrades with shoes attached.** Commanded-direction velocity at the 0.3 m/s setpoint for the same bare-foot nominal policy, executed on bare feet (blue) and after attaching shoes (green); the dashed ring marks the target. The morphology shift induced by shoes particularly degrades backward-command tracking.

to DR: footwear-aware DR requires a predefined distribution of contact properties, whereas SHOES adapts to the encountered condition after deployment and can also be applied to a DR-trained policy.

Real-world RL and sim-to-online fine-tuning. Smith et al. [27] demonstrate SAC-based reinforcement learning [28] from scratch on a real quadruped in 20 minutes, and Wu et al. [29] learn world models on hardware. Without a competent sim-trained initial policy, learning locomotion directly on hardware can be sample-inefficient and require careful safety supervision during early exploration [30]. Smith et al. [4] fine-tune the full policy using off-policy RL and a learned recovery policy. SHOES instead starts from a nominal policy trained with on-policy PPO, a standard choice in quadruped locomotion. Full-policy fine-tuning also requires a separate full-sized checkpoint per footwear, whereas SHOES reuses the same nominal policy and switches only small footwear-specific modules. As et al. [31] provide a systematic study of sim-to-online RL design choices on real robots, including data retention and delayed actor updates. Related lines of work align simulation and real-world physics for agile humanoid whole-body skills [32] and develop rapid online adaptive control for robust legged locomotion [33]. Their focus, however, is on algorithmic or physics-alignment design choices rather than identity-initialized adaptation modules around a frozen nominal policy.

FiLM-based fine-tuning on hardware. FiLM [34] provides a general-purpose feature-wise affine conditioning layer. The closest hardware-side fine-tuning method to ours is RTR [35], which fine-tunes a FiLM-conditioned latent on a humanoid robot while using a teacher robot arm for safety. SHOES differs in three key aspects: (i) modulation is conditioned directly on the policy observation rather than on a learned dynamics latent; (ii) the FiLM modulation is bounded and is attached to a nominal policy trained without FiLM, with identity initialization preserving the nominal actor exactly at the start of fine-tuning; and (iii) SHOES separates the correction into state-dependent hidden-layer modulation and an action-space Linear Adapter, a design choice that is validated by the ablation study in Section 5.

3 Preliminaries

We model footwear adaptation as a two-stage reinforcement learning problem on a fully observable MDP $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$. A policy $\pi_\theta(a \mid o)$ outputs a Gaussian over actions a conditioned on the proprioceptive observation o . Concrete observation, action, and control-rate specifications are deferred to Appendix A.

Stage 1 (pre-deploy, sim). We first train a nominal policy π_{nom} in simulation, solving

$$\pi_{\text{nom}} = \arg \max_{\pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_t r_{\text{bare}}(s_t, a_t) \right], \quad (1)$$

until convergence with a large simulation budget. The Stage-1 reward r_{bare} is the standard quadruped velocity recipe (Appendix B).

Stage 2 (post-deploy, online). The robot is then equipped with shoes that alter the foot-ground contact dynamics; under this morphology shift the zero-shot performance of π_{nom} may degrade. Stage 2 freezes π_{nom} and trains a small adapter parameterized by θ on top, subject to a budget N_{adapt} :

$$\theta^* = \arg \max_{\theta} \mathbb{E}_{\tau \sim \pi_{\theta}} \left[\sum_t r_{\text{shoe}}(s_t, a_t) \right], \quad \text{s.t. } \pi_{\theta}(\cdot) \big|_{\theta=0} = \pi_{\text{nom}}(\cdot), \quad N_{\text{steps}} \leq N_{\text{adapt}}. \quad (2)$$

The identity-at-initialization constraint $\pi_{\theta} \big|_{\theta=0} = \pi_{\text{nom}}$ ensures that Stage 2 starts regression-free with respect to the sim-trained base. The Stage-2 reward r_{shoe} retains only the velocity-tracking terms of r_{bare} with amplified weights (Appendix B): stable locomotion is already learned in Stage 1, so Stage 2 optimizes only velocity tracking.

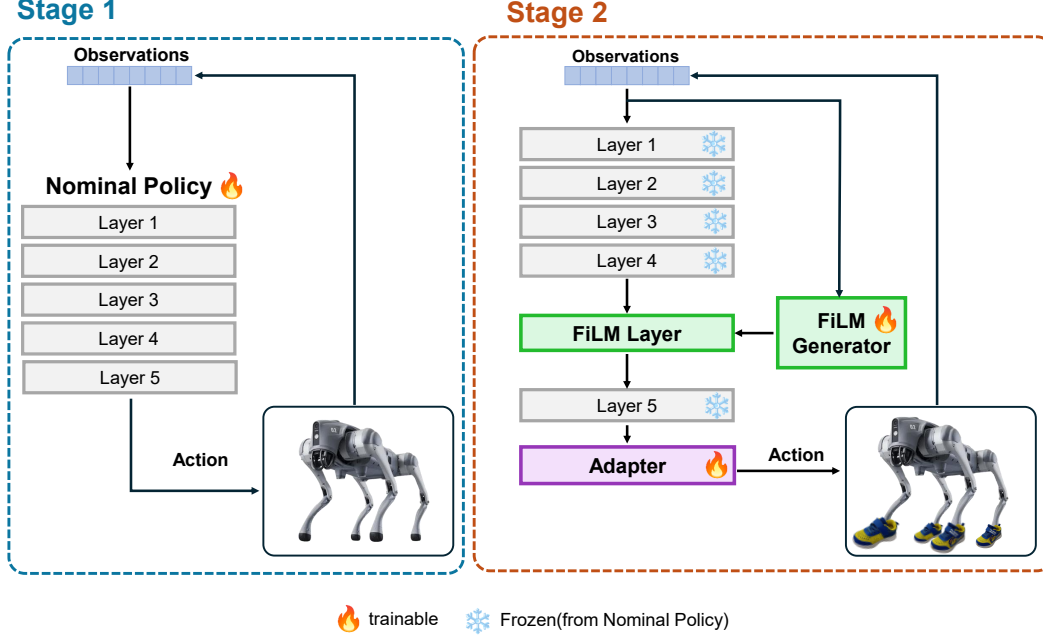


Figure 3: **SHOES two-stage training pipeline.** Stage 1 trains a nominal PPO actor-critic on the bare-foot Go2 in simulation with 4096 parallel envs; the resulting π_{nom} becomes the frozen base for Stage 2. Stage 2 wraps the frozen nominal MLP (gray, $\sim 190\text{k}$ params) with a trainable FiLM generator and a Linear Adapter (coral, $\sim 5.6\text{k}$ params); PPO online-fine-tunes the trainable modules on the shoe-equipped Go2 with the SafeBox sampler at $1 \text{ env} \times 30\text{k env-steps}$.

4 Method

Overview. SHOES adapts a simulation-trained locomotion policy to new footwear conditions while keeping the nominal policy interface unchanged. Starting from a Stage-1 nominal policy, SHOES freezes the nominal actor and observation normalizer, and fine-tunes only lightweight, identity-initialized adaptation modules in Stage 2. The framework has two architectural components: a bounded FiLM generator paired with a Linear Adapter for behavior-preserving policy adaptation (Sec. 4.1), and SafeBox for workspace-aware waypoint sampling during on-robot adaptation (Sec. 4.2). Stage 2 fine-tunes these modules with on-policy PPO [36]; full pseudocode and hyper-parameters are deferred to Appendices C and A. An overview of SHOES is shown in Fig. 3.

4.1 Behavior-Preserving Policy Adaptation

SHOES adapts the nominal policy by wrapping the frozen actor with lightweight trainable modules, while preserving the original policy interface. These modules are initialized so that, at the beginning of Stage 2, the wrapped actor is exactly equivalent to the Stage-1 nominal policy. Figure 3 shows this wrapped actor. Let the frozen nominal actor be decomposed into a feature extractor $f_{\text{body}} : \mathbb{R}^{d_o} \rightarrow \mathbb{R}^{d_h}$ and an output head $f_{\text{head}} : \mathbb{R}^{d_h} \rightarrow \mathbb{R}^{d_a}$, where d_o , d_h , and d_a denote the observation, hidden, and action dimensions, respectively:

$$h = f_{\text{body}}(o), \quad \mu_{\text{nom}} = f_{\text{head}}(h). \quad (3)$$

Around this frozen computation, we insert two trainable modules:

$$h' = \text{FiLM}_{\phi}(h, o), \quad \mu = A_{\psi}(f_{\text{head}}(h')). \quad (4)$$

Here, f_{body} and f_{head} are frozen, while ϕ (the FiLM generator) and ψ (the Linear Adapter) are trained online with PPO. The two trainable modules are instantiated as follows.

Bounded FiLM modulation. A small FiLM generator $g_\phi : \mathbb{R}^{d_o} \rightarrow \mathbb{R}^{2d_h}$ consumes the (normalized) observation o and outputs scale and shift parameters $(\gamma_{\text{raw}}, \beta_{\text{raw}}) \in \mathbb{R}^{d_h} \times \mathbb{R}^{d_h}$. The hidden representation is modulated as

$$h' = \underbrace{(1 + \alpha \tanh(\gamma_{\text{raw}}))}_{\text{multiplicative scale}} \odot h + \underbrace{\alpha \tanh(\beta_{\text{raw}})}_{\text{additive shift}}, \quad \alpha = 0.5. \quad (5)$$

We initialize the final layer of g_ϕ to zero so that $\gamma_{\text{raw}} = \beta_{\text{raw}} = 0$ at $\theta = 0$, making the FiLM modulation the identity map $h' = h$ at initialization. The $(1 + \alpha \tanh)$ form bounds the multiplicative gain to $[1 - \alpha, 1 + \alpha]$ and the additive shift to $[-\alpha, \alpha]$; with $\alpha = 0.5$ this restricts modulation to a nominal-anchored neighborhood under the limited on-robot adaptation budget.

Linear Adapter. An affine map $A_\psi : \mathbb{R}^{d_a} \rightarrow \mathbb{R}^{d_a}$ post-processes the action mean:

$$\mu = W_A [f_{\text{head}}(h')] + b_A. \quad (6)$$

We initialize $W_A = I_{d_a}$ and $b_A = 0$ so that A_ψ is the identity map $\mu = f_{\text{head}}(h')$ at initialization.

The Linear Adapter does not see o ; it captures static action-space offsets, such as systematic leg-height biases induced by footwear, while FiLM provides observation-conditioned modulation of the hidden representation. Together, they separate static output-space corrections from dynamic hidden-state modulation.

Combining (5) and (6), the actor performs a single forward pass per environment step. At initialization, $h' = h$ and $W_A = I$, so $\mu = \mu_{\text{nom}}$ exactly.

4.2 Workspace-Aware Waypoint Sampling (SafeBox)

Random velocity commands can drive a robot outside the available workspace during online adaptation. SafeBox instead generates commands toward waypoints sampled within a user-defined workspace, without changing the policy architecture, action space, or training objective.

Workspace-aware waypoint sampling. SafeBox is parameterized by a user-defined rectangular workspace $\mathcal{B} = [-a, a] \times [-b, b]$ in a local world frame. At each resampling event, it samples a waypoint $\mathbf{t} = (t_x, t_y)$ uniformly from $\mathcal{B}$, a cruise speed v_{cap} , and a yaw-rate command ω_z :

$$\mathbf{t} \sim \mathcal{U}(\mathcal{B}), \quad v_{\text{cap}} \sim \mathcal{U}(v_{\text{cap}}^{\min}, v_{\text{cap}}^{\max}), \quad \omega_z \sim \mathcal{U}(-\omega_{\max}, \omega_{\max}). \quad (7)$$

Here v_{cap} is the (scalar) magnitude of the world-frame velocity command used in Eq. (8), and the per-axis limits $v_x^{\max}, v_y^{\max}$ in Eq. (9) act on the body-frame components after projection. We reuse the forward limit as the cruise-speed upper bound, $v_{\text{cap}}^{\max} := v_x^{\max}$, since forward motion dominates the training distribution; this couples the two scalars by convention rather than by definition. The sampled waypoint and yaw-rate command are held until the robot reaches a small neighborhood of the waypoint or a resampling timer expires.

Closed-loop velocity command generation. Given the current planar position $\mathbf{p} = (x_w, y_w)$ and yaw ψ , SafeBox first computes a world-frame velocity that points toward the active waypoint:

$$d = \|\mathbf{t} - \mathbf{p}\|, \quad \mathbf{v}^w = \min(v_{\text{cap}}, k_p d) \frac{\mathbf{t} - \mathbf{p}}{\max(d, \delta)}. \quad (8)$$

Here, k_p is a proportional gain and δ is a small constant for numerical stability. The command magnitude saturates far from the waypoint and decreases smoothly near the target. The world-frame velocity is rotated into the body frame, clipped per axis, and combined with the sampled yaw-rate command:

$$\begin{pmatrix} v_x^{\text{body}} \\ v_y^{\text{body}} \end{pmatrix} = R(-\psi) \mathbf{v}^w, \quad \mathbf{v}^{\text{cmd}} = \begin{pmatrix} \text{clip}(v_x^{\text{body}}, -v_x^{\max}, v_x^{\max}) \\ \text{clip}(v_y^{\text{body}}, -v_y^{\max}, v_y^{\max}) \\ \omega_z \end{pmatrix}. \quad (9)$$

The policy receives $\mathbf{v}^{\text{cmd}}$ through the same command channels as in Stage 1 and does not observe the robot position, sampled waypoint, or workspace bounds. Since each waypoint is sampled inside $\mathcal{B}$ and each planar command points toward the active waypoint, SafeBox discourages long open-loop drifts while preserving the policy interface. Implementation details and parameter values are provided in Appendix D.

5 Experiments

We evaluate SHOES on a Unitree Go2 in simulation using the MuJoCo-Warp engine through the mjlab environment framework [37]. The bare-foot robot defines the reference morphology and the shoe-equipped robot defines the post-deployment morphology shift to be adapted. Unless noted otherwise, all Stage-2 methods start from the same frozen nominal checkpoint, use the SafeBox command sampler, and receive the same single-environment adaptation budget of $N_{\text{adapt}} = 30,000$ environment steps.

Performance is measured on a 24-cell velocity-tracking benchmark formed by crossing 3 speed setpoints with 8 directions. The primary figure of merit, Bare-Foot Parity (BFP), aggregates the per-cell ratio of the policy’s commanded-direction velocity to the bare-foot nominal’s: $\text{BFP} = 100\%$ indicates parity with the bare-foot reference in the commanded direction, and lower scores mean the policy tracks more slowly than bare-foot. Per-cell measurement details, the exact formula, and the multi-seed protocol are in Appendix E.

Q1. Can SHOES recover from footwear-induced degradation? We evaluate whether SHOES can recover the tracking performance lost when the frozen bare-foot nominal is executed with shoes attached. Without adaptation, the nominal drops to $\text{BFP} = 67.0\%$; after a 30k-step Stage-2 fine-tune, SHOES reaches $93.5 \pm 3.9\%$ over six seeds, recovering most of the loss. Figure 4 provides the per-direction breakdown: across all three speed setpoints, the no-adapt nominal sits well inside the bare-foot nominal, whereas SHOES closely matches it, indicating broad recovery across the 24-cell tracking benchmark.

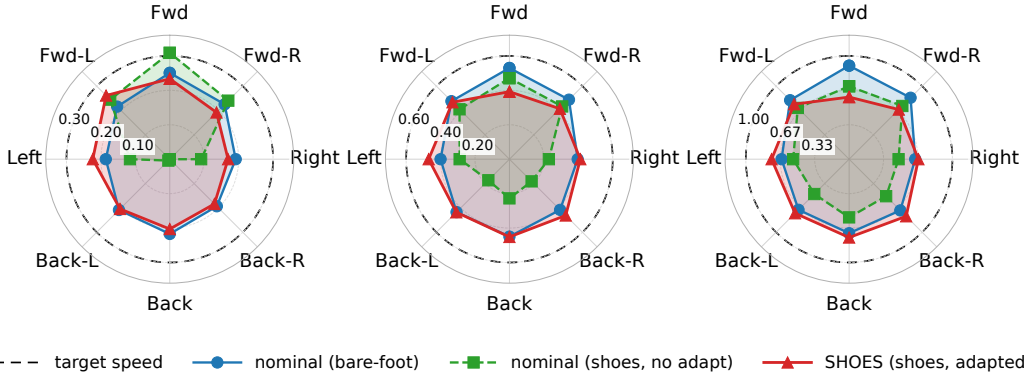


Figure 4: SHOES recovers bare-foot tracking across speeds and directions. Per-direction tracked velocity at the three command speeds (0.3, 0.6, 1.0 m/s). SHOES (red, best seed from Appendix F) closely matches the bare-foot nominal reference (blue), while the no-adapt nominal on shoes (green) tracks well inside it across all directions and speeds.

Q2. Is the decomposition of hidden-feature and action-space corrections necessary?

SHOES separates footwear adaptation into state-dependent hidden-layer modulation and an action-space Linear Adapter. We test whether this separation is necessary by ablating each component under the same 30k-step adaptation budget. Table 1 shows that the

Linear Adapter alone fails to improve over the no-adapt nominal, indicating that a static action-space correction cannot by itself compensate for the shoe-induced shift. FiLM-only recovers a meaningful portion of the gap but still falls short of the bare-foot reference. Combining the two components closes most of the gap, recovering +26.5 pp of Bare-Foot Parity over the no-adapt nominal. This result supports the core SHOES design: under a limited on-robot-equivalent budget, footwear adaptation is better captured by combining state-dependent feature modulation with a lightweight correction in the action space around the frozen nominal actor.

Table 1: **Component ablation** (30k-step budget). Δ BFP is the gain over the no-adapt nominal (first row).

FiLM	Adapter	Trainable	BFP $\uparrow$	Δ BFP $\uparrow$
$\times$	$\times$	0	67.0	0.0
$\times$	$\checkmark$	0.2k	65.1 ± 8.9	-1.9
$\checkmark$	$\times$	5.4k	78.7 ± 10.8	+11.7
$\checkmark$	$\checkmark$	5.6k	93.5 ± 3.9	+26.5

Q3. Which post-training adaptation method is most effective?

We compared SHOES with alternative post-training adaptation baselines under the same 30k-step budget. Figure 5 shows that the Linear Adapter, Residual SAC, and Residual PPO baselines all remain near the no-adapt nominal, failing to make meaningful progress within the on-robot-equivalent budget. FiLM-only is the strongest baseline but still falls short of the bare-foot reference. Only the full SHOES model substantially closes the gap, giving the largest recovery among all methods. The comparison indicates that effective post-training adaptation is not achieved by adding residual or single-module capacity alone; under the limited adaptation budget, the identity-initialized combination of hidden-feature modulation and action-space correction is what separates SHOES from the baselines. We additionally compared SHOES with full-policy PPO fine-tuning under the same 30k-step adaptation budget. Averaged over the four additional footwear variants evaluated in Q4, SHOES achieves a BFP of $107.3 \pm 3.3\%$, compared with $97.8 \pm 2.3\%$ for full-policy PPO fine-tuning.

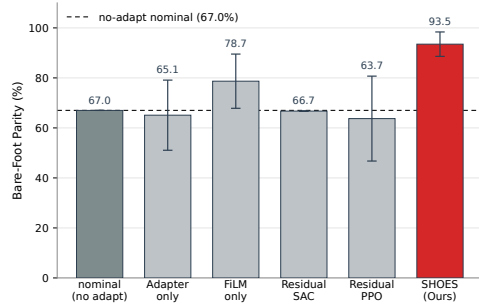


Figure 5: **Method comparison (flat shoes).** Bare-Foot Parity (%) mean $\pm$ std over six seeds. Dashed line: no-adapt nominal.

Q4. Does SHOES adapt beyond flat shoes?

We added simulated footwear variants that alter contact geometry and added mass beyond the flat shoe (Figure 6). For each footwear variant, SHOES starts from the same nominal checkpoint and adapts for 30k environment steps. SHOES recovers performance across all of them, as summarized in Table 2. Since BFP measures only velocity along the commanded direction, we additionally report fall rate, lateral velocity error, yaw-rate tracking error, and action-rate cost in Table 3, including evaluations with nonzero yaw-rate commands. SHOES improves commanded-velocity tracking and stability, while the mean action-rate cost increases modestly from 0.725 to 0.790. Indeed, Table 3 shows that optimizing only velocity tracking in Stage 2 also reduces the fall rate.

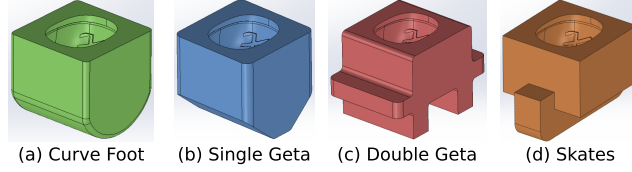


Figure 6: Simulated footwear variants: (a) Curve Foot, (b) Single Geta, (c) Double Geta, and (d) Skates.

Table 2: Velocity-tracking recovery across simulated footwear variants.

Footwear variant	Nominal (w/o adapt.)	SHOES (ours)
Curve Foot	36.4%	$115.0 \pm 5.8\%$
Single Geta	36.8%	$113.5 \pm 6.5\%$
Double Geta	17.7%	$103.6 \pm 12.7\%$
Skates	38.6%	$97.2 \pm 5.1\%$

Table 3: Stability and motion-quality metrics averaged over the four additional footwear variants.

Metric	Nominal (w/o adapt.)	SHOES (ours)
Fall rate	55.4%	$6.3 \pm 4.6\%$
Lateral error (m/s)	0.148	0.080 ± 0.005
Yaw-rate error (rad/s)	0.106	0.070 ± 0.005
Action-rate cost	0.725	0.790 ± 0.043

Q5. Can the same adaptation interface be deployed on hardware? We finally examined whether the Stage-2 adaptation transfers to the physical Unitree Go2 with shoes attached (Figure 7). The policy runs at 50 Hz on top of the 500 Hz onboard PD loop. We evaluated the backward setpoint $v_x^{\text{cmd}} = -0.30$ m/s, which exhibited the largest no-adapt gap in the simulation radar chart (Figure 4). Figure 7a reports the body-frame v_x over a constant-command window.

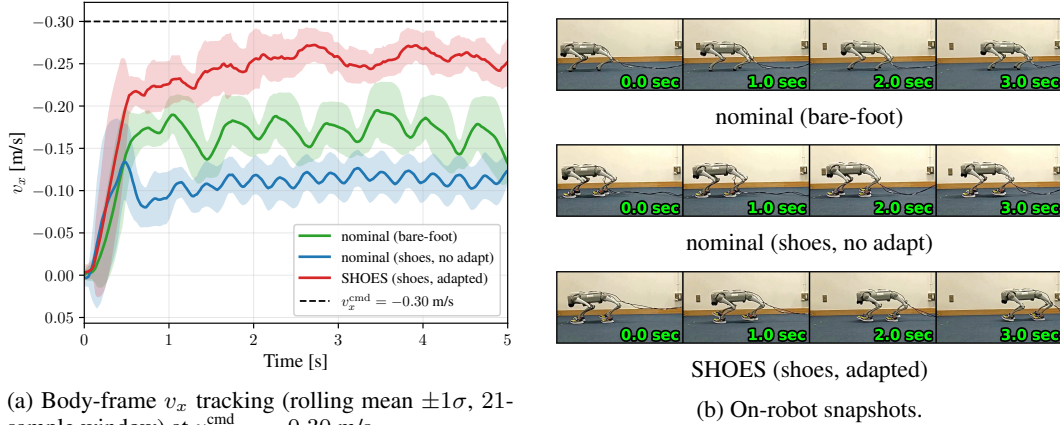


Figure 7: **SHOES recovers backward-velocity tracking on hardware.** Backward setpoint $v_x^{\text{cmd}} = -0.30$ m/s on the physical Unitree Go2 with flat shoes. (a) SHOES (red) substantially improves backward tracking over the no-adapt nominal on shoes (blue); the bare-foot nominal (green) is shown for reference (black dashed line: command). (b) Snapshots of the robot in the three settings overlaid in the left panel.

The no-adapt nominal policy with shoes attached (blue) tracked at $v_x \approx -0.10 \pm 0.03$ m/s, corresponding to only 33 % of the commanded velocity. In contrast, SHOES (red) reached $v_x \approx -0.25 \pm 0.03$ m/s, recovering 83 % of the command and reducing the steady-state tracking error from 0.20 to 0.05 m/s. The bare-foot nominal policy (green), evaluated on the same hardware as the reference morphology, tracked at $v_x \approx -0.15 \pm 0.04$ m/s, or 50 % of the command. This remaining gap reflects the sim-to-real discrepancy already present without the footwear-induced morphology shift.

Notably, SHOES exceeded the bare-foot nominal reference on the same hardware in this backward-velocity test. This result suggests that on-robot Stage-2 fine-tuning not only compensates for the

footwear-induced morphology change, but can also reduce part of the pre-existing sim-to-real gap of the nominal policy. Figure 7b qualitatively illustrates this difference on the physical robot: attaching shoes to the no-adapt nominal visibly slows locomotion, whereas Stage-2 fine-tuning with SHOES restores backward speed on the same hardware.

Throughout the 30k-step Stage-2 fine-tuning process, the SafeBox sampler constrained exploration safely: no episode required emergency intervention, no boundary violation was observed, and data collection proceeded without hardware damage. These results confirm improved tracking over the unadapted shoe-equipped policy, with SHOES additionally surpassing the bare-foot nominal reference in this backward-velocity test.

Q6. Does footwear enable traversal beyond basic protection? We added a real-robot experiment on a slotted surface (Figure 8). Without shoes, a foot drops into an opening and the robot becomes trapped, whereas the shoe-equipped robot adapted with SHOES traverses the surface. Task-specific footwear enables traversal of terrain that is inaccessible to the bare-foot nominal robot, and SHOES adapts the locomotion policy to the new footwear condition.

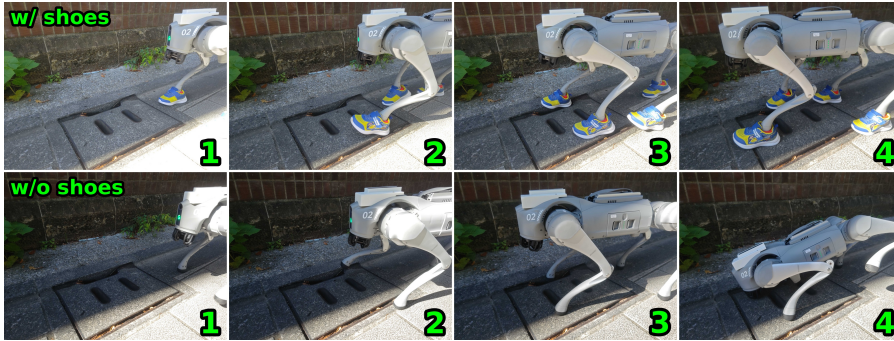


Figure 8: Real-robot traversal over a slotted surface. With shoes, the robot crosses the openings (top); without shoes, it becomes trapped (bottom).

6 Limitations

SafeBox enables on-robot Stage-2 fine-tuning by generating exploratory commands within a user-defined workspace. This reduces the need for continuous operator intervention to keep rollouts inside the workspace. However, the current system does not integrate physical resets into the online adaptation loop. Consequently, SHOES is not yet practical for footwear conditions that frequently induce falls or other reset-triggering failures. In such cases, training must be suspended until an operator manually rights the robot and restarts the episode. For footwear causing frequent failures, SHOES can be combined with a learned self-righting policy or a safety-tether reset triggered upon fall detection. Incorporating fall detection, autonomous stand-up recovery, and reset-aware episode management could extend online adaptation to more severe footwear-induced morphology shifts.

7 Conclusion

We presented SHOES, a sample-efficient framework for adapting locomotion policies to footwear-induced morphology changes. SHOES freezes a simulation-trained nominal policy and fine-tunes only lightweight, identity-initialized FiLM and Linear Adapter modules. This lets fine-tuning start from the nominal policy behavior and gradually adapt it to the shoe-equipped robot. SafeBox supports online adaptation by keeping exploratory rollouts within a user-defined workspace. On a shoe-equipped Unitree Go2, SHOES reached a bare-foot parity of $93.5 \pm 3.9\%$ using only 30k environment steps. These results suggest a practical route for adapting deployed controllers to post-deployment hardware changes.

Acknowledgments

We thank the anonymous reviewers and the area chair for their constructive feedback, which led to the additional footwear variants, complementary metrics, and the slotted-terrain experiment in the camera-ready version. This work was supported by JST Moonshot R&D Program Grant Number JPMJMS263D.

References

- [1] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.
- [2] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
- [3] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.
- [4] L. Smith, J. C. Kew, X. B. Peng, S. Ha, J. Tan, and S. Levine. Legged robots that keep on learning: Fine-tuning locomotion policies in the real world. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.
- [5] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. In *Robotics: Science and Systems (RSS)*, 2018.
- [6] 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*, 4(26):eaau5872, 2019.
- [7] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47):eabc5986, 2020.
- [8] A. Kumar, Z. Fu, D. Pathak, and J. Malik. RMA: Rapid motor adaptation for legged robots. In *Robotics: Science and Systems (RSS)*, 2021.
- [9] 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*, 7(62):eabk2822, 2022.
- [10] G. Ji, J. Mun, H. Kim, and J. Hwangbo. Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion. *IEEE Robotics and Automation Letters*, 7(2):4630–4637, 2022.
- [11] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal. Rapid locomotion via reinforcement learning. In *Robotics: Science and Systems (RSS)*, 2022.
- [12] G. B. Margolis and P. Agrawal. Walk these ways: Tuning robot control for generalization with multiplicity of behavior. In *Conference on Robot Learning (CoRL)*, 2022.
- [13] Y. Shao, Y. Jin, X. Liu, W. He, H. Wang, and W. Yang. Learning free gait transition for quadruped robots via phase-guided controller. *IEEE Robotics and Automation Letters*, 7(2):1230–1237, 2022.
- [14] G. Feng, H. Zhang, Z. Li, X. B. Peng, B. Basireddy, L. Yue, Z. Song, L. Yang, Y. Liu, K. Sreenath, and S. Levine. GenLoco: Generalized locomotion controllers for quadrupedal robots. In *Conference on Robot Learning (CoRL)*, 2022.

- [15] T. Wu, H. Guo, Y. Wang, J. Yang, X. Sui, J. Xie, X. Chen, Z. Liu, and X. Lan. Toward reliable sim-to-real predictability for MoE-based robust quadrupedal locomotion. In *Robotics: Science and Systems (RSS)*, 2026.
- [16] W. Yu, V. C. Kumar, G. Turk, and C. K. Liu. Sim-to-real transfer for biped locomotion. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.
- [17] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath. Reinforcement learning for robust parameterized locomotion control of bipedal robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.
- [18] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath. Real-world humanoid locomotion with reinforcement learning. *Science Robotics*, 9(89):eadi9579, 2024.
- [19] Z. Zhuang, S. Yao, and H. Zhao. Humanoid parkour learning. In *Conference on Robot Learning (CoRL)*, 2024.
- [20] J. Siekmann, S. Valluri, J. Dao, L. Bermillo, H. Duan, A. Fern, and J. Hurst. Learning memory-based control for human-scale bipedal locomotion. In *Robotics: Science and Systems (RSS)*, 2020.
- [21] J. Siekmann, Y. Godse, A. Fern, and J. Hurst. Sim-to-real learning of all common bipedal gaits via periodic reward composition. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.
- [22] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst. Blind bipedal stair traversal via sim-to-real reinforcement learning. In *Robotics: Science and Systems (RSS)*, 2021.
- [23] H. Duan, A. Malik, J. Dao, A. Saxena, K. Green, J. Siekmann, A. Fern, and J. Hurst. Sim-to-real learning of footstep-constrained bipedal dynamic walking. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.
- [24] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. van de Panne. Learning locomotion skills for Cassie: Iterative design and sim-to-real. In *Conference on Robot Learning (CoRL)*, 2019.
- [25] N. Rudin, D. Hoeller, P. Reist, and M. Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2021.
- [26] M. Liu, D. Pathak, and A. Agarwal. LocoFormer: Generalist locomotion via long-context adaptation. In *Conference on Robot Learning (CoRL)*, 2025.
- [27] L. Smith, I. Kostrikov, and S. Levine. Demonstrating a walk in the park: Learning to walk in 20 minutes with model-free reinforcement learning. In *Robotics: Science and Systems (RSS)*, 2023.
- [28] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, 2018.
- [29] P. Wu, A. Escontrela, D. Hafner, K. Goldberg, and P. Abbeel. DayDreamer: World models for physical robot learning. In *Conference on Robot Learning (CoRL)*, 2022.
- [30] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan. Learning to walk in the real world with minimal human effort. In *Conference on Robot Learning (CoRL)*, 2020.
- [31] Y. As, D. Tirumala, R. Zurbrugg, C. Li, S. Coros, A. Krause, and M. Wulfmeier. What matters for simulation to online reinforcement learning on real robots. *arXiv preprint arXiv:2602.20220*, 2026.

- [32] T. He, J. Gao, W. Xiao, Y. Zhang, Z. Wang, J. Wang, Z. Luo, G. He, N. Sobanbabu, C. Pan, Z. Yi, G. Qu, K. Kitani, J. Hodgins, L. Fan, Y. Zhu, C. Liu, and G. Shi. ASAP: Aligning simulation and real-world physics for learning agile humanoid whole-body skills. In *Robotics: Science and Systems (RSS)*, 2025.
- [33] S. Lyu, X. Lang, H. Zhao, H. Zhang, P. Ding, and D. Wang. RL2AC: Reinforcement learning-based rapid online adaptive control for legged robot robust locomotion. In *Robotics: Science and Systems (RSS)*, 2024.
- [34] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI Conference on Artificial Intelligence*, 2018.
- [35] K. Hu, H. Shi, Y. He, W. Wang, C. K. Liu, and S. Song. Robot trains robot: Automatic real-world policy adaptation and learning for humanoids. In *Conference on Robot Learning (CoRL)*, 2025.
- [36] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [37] K. Zakka, Q. Liao, B. Yi, L. Le Lay, K. Sreenath, and P. Abbeel. mjlab: A lightweight framework for GPU-accelerated robot learning. *arXiv preprint arXiv:2601.22074*, 2026.

Appendix

A Architecture and Hyper-parameters

Architecture. We instantiate Section 4.1 with $d_o = 47$, $d_h = 128$, and $d_a = 12$. The 47-d observation o stacks base angular velocity (3), projected gravity (3), velocity command (3), per-joint positions and velocities ($12 + 12$), the previous action (12), and a $\sin/\cos$ phase signal (2). The 12-d action specifies joint-position offsets executed by a 500 Hz onboard PD loop. The frozen nominal MLP has hidden widths (512, 256, 128) with ELU activations. The FiLM generator g_ϕ is $\text{Linear}(47 \rightarrow 16) \rightarrow \text{ELU} \rightarrow \text{Linear}(16 \rightarrow 16) \rightarrow \text{ELU} \rightarrow \text{Linear}(16 \rightarrow 256)$ with the final layer zero-initialized. The Linear Adapter is a single $d_a \times d_a$ affine layer initialized to identity. The trainable actor totals 5,560 parameters (5,392 FiLM + 156 Linear Adapter + 12 action log-std); the warm-started critic has $\sim 190\text{k}$ parameters.

Training. Stage 2 uses PPO (Table 4) with $\text{num_steps_per_env} = 256$, $K = 3$ epochs per update, and an adaptive KL learning rate ($\eta_0 = 3 \times 10^{-5}$, target KL 0.005). With $\text{num_envs} = 1$ and $N_{\text{adapt}} = 30,000$ we run 117 PPO iterations; the SafeBox sampler supplies velocity commands throughout. At identity initialization, a unit test verifies that the first actor forward pass is bit-identical to π_{nom} to 10^{-5} absolute action error. The FiLM bound $\alpha = 0.5$ and generator hidden dimensions (16, 16) are selected from the sweep in Appendix H. Stage 2 fine-tuning completes in ~ 10 min on a single RTX 4070; Stage 1 nominal training uses 4096 parallel envs for 30M env-steps on an RTX 5080.

B Reward Specification

Both stages optimize $R(\tau) = \sum_t \sum_i w_i r_i(s_t, a_t)$.

Stage 1 (bare-foot nominal training). Combines velocity tracking, posture, and gait-shaping terms following the standard quadruped recipe (Table 5).

The trot stance schedule is $s_i(t) = \mathcal{W}\{((t/T_{\text{phase}}) + \phi_i) \bmod 1 < 0.56\}$ with diagonal offsets $\phi = (0, 0.5, 0.5, 0)$. The posture standard deviation $\sigma_{\text{pose},j}$ depends on the joint and command magnitude: standing ($\sigma_{\text{hip}}, \sigma_{\text{thigh}}, \sigma_{\text{calf}} = (0.05, 0.10, 0.15)$) when $u^{\text{cmd}} := \|\mathbf{v}_{xy}^{\text{cmd}}\| + |\omega_z^{\text{cmd}}| < 0.1$, walking (0.15, 0.35, 0.50) otherwise. The foot-related terms (gait, clearance, slip) and stand_still are gated by the same u^{cmd} threshold $\varepsilon = 0.1$; soft_landing uses 0.05.

Table 4: Stage-2 PPO hyper-parameters.

Parameter	Value
Parallel environments	1
Steps per environment per rollout	256
PPO epochs per update	3
Mini-batches per epoch	4
Total PPO iterations	117
Total environment steps	30,000
Clip parameter	0.1
Initial learning rate	3×10^{-5}
Adaptive KL target	0.005
Discount γ	0.99
GAE λ	0.95
FiLM bound α	0.5
Generator hidden dims	(16, 16)
Optimizer	Adam (default β)

Table 5: Stage-1 rewards (bare-foot nominal training). $\mathbf{v}_{xy}^{\text{cmd}}, \omega_z^{\text{cmd}}$: commanded linear/yaw velocity (body frame); $\mathbf{v}_{xy}, v_z, \omega_{xy}$: base linear and angular velocities; $\mathbf{g}_{xy}^{\text{proj}}$: projected gravity (body horizontal); $\theta, \ddot{\theta}$: joint positions and accelerations; θ^{default} : default joint pose; $\mathbf{L}_{\text{root}}$: root angular momentum; $p_z^i, \mathbf{v}_{xy}^i, \mathbf{F}^i$: foot i height, xy velocity (world), contact force; $c_i \in \{0, 1\}$: stance indicator; $\eta_i \in \{0, 1\}$: first-contact (landing) indicator; $\theta_j^{\text{low}}, \theta_j^{\text{up}}$: soft joint position limits; “terminated” covers fall (base tilt $> 70^\circ$) and non-foot contact (> 10 N); $T_{\text{phase}} = 0.6$ s, $p_z^{\text{tgt}} = 0.10$ m, $\sigma_{\text{lin}}^2 = 0.25$, $\sigma_{\text{ang}}^2 = 0.5$.

Reward	Equation r_i	Weight w_i
Lin. velocity tracking	$\exp(-[\ \mathbf{v}_{xy}^{\text{cmd}} - \mathbf{v}_{xy}\ ^2 + 2v_z^2]/\sigma_{\text{lin}}^2)$	+1.0
Ang. velocity tracking	$\exp(-[(\omega_z^{\text{cmd}} - \omega_z)^2 + 0.05\ \omega_{xy}\ ^2]/\sigma_{\text{ang}}^2)$	+1.0
Body orientation	$\ \mathbf{g}_{xy}^{\text{proj}}\ ^2$	-1.0
Posture	$\exp(-\frac{1}{2J} \sum_j (\theta_j - \theta_j^{\text{default}})^2 / \sigma_{\text{pose},j}(\mathbf{v}^{\text{cmd}})^2)$	+1.0
Body angular velocity	$\ \omega_{xy}\ ^2$	-0.05
Angular momentum	$\ \mathbf{L}_{\text{root}}\ ^2$	-0.025
Termination	$\mathbb{I}_{\{\text{terminated}\}}$	-200
Joint accelerations	$\ \ddot{\theta}\ ^2$	-2.5×10^{-7}
Joint position limits	$\sum_j [\text{ReLU}(\theta_j^{\text{low}} - \theta_j) + \text{ReLU}(\theta_j - \theta_j^{\text{up}})]$	-10
Action rate	$\ \mathbf{a}_t - \mathbf{a}_{t-1}\ ^2$	-0.05
Foot gait	$\frac{1}{4} \sum_i \mathbb{I}_{\{c_i = s_i(t)\}}$	+0.5
Foot clearance	$\sum_i p_z^i - p_z^{\text{tgt}} \cdot \ \mathbf{v}_{xy}^i\ $	-1.0
Foot slip	$\sum_i \ \mathbf{v}_{xy}^i\ ^2 \cdot c_i$	-0.25
Soft landing	$\sum_i \ \mathbf{F}^i\ \cdot \eta_i$	-10^{-3}
Stand still	$\ \theta - \theta^{\text{default}}\ ^2 \cdot \mathbb{I}_{\{u^{\text{cmd}} \leq \varepsilon\}}$	-1.0

Stage 2 (SHOES online adaptation). Every Stage-1 shaping term is set to zero in Stage 2 (Table 6): the shaping rewards are already baked into the frozen nominal MLP, and keeping non-zero weights empirically collapses the policy into a “stand-still / walk conservatively” local optimum. The linear-velocity term uses an adaptive standard deviation,

$$\sigma_{\text{lin}}(s) = \max(\sigma_{\text{min}}, k \cdot \|\mathbf{v}_{xy}^{\text{cmd}}\|), \quad \sigma_{\text{min}} = 0.05, k = 0.3. \quad (10)$$

Table 6: Stage-2 rewards. $\sigma_{\text{lin}}(s)$ from Eq. (10), $\sigma_{\text{ang}} = 0.25$.

Reward	Equation r_i	Weight w_i
Lin. velocity tracking (adaptive)	$\exp(-[\ \mathbf{v}_{xy}^{\text{cmd}} - \mathbf{v}_{xy}\ ^2 + 2v_z^2]/\sigma_{\text{lin}}(s)^2)$	+10.0
Ang. velocity tracking	$\exp(-[(\omega_z^{\text{cmd}} - \omega_z)^2 + 0.05\ \omega_{xy}\ ^2]/\sigma_{\text{ang}}^2)$	+2.0

Algorithm 1 SHOES on-robot adaptation.

```

1: Train nominal policy  $\pi_{\text{nom}}$  in simulation on the bare-foot robot (Stage 1).
2: Freeze observation normalizer and nominal actor parameters.
3: Initialize FiLM generator final layer to zero and Linear Adapter to identity ( $W_A = I_{d_a}, b_A = 0$ ).
4: Warm-start critic from the nominal critic.
5: for each PPO iteration do
6:   for each environment step do
7:     Sample velocity command  $(v_x, v_y, \omega_z)_t$  with SafeBox.
8:     Obtain robot pose and observation  $o_t$ .
9:     Compute  $h_t = f_{\text{body}}(o_t)$  (frozen).
10:    Apply bounded FiLM:  $h'_t = \text{FiLM}_{\phi}(h_t, o_t)$ .
11:    Compute action mean  $\mu_t = A_{\psi}(f_{\text{head}}(h'_t))$ .
12:    Sample  $a_t \sim \mathcal{N}(\mu_t, \exp(\log \sigma)^2)$  and execute the PD target.
13:    Compute reward  $r_t$  and store transition.
14:   end for
15:   Update FiLM ( $\phi$ ), Linear Adapter ( $\psi$ ), log-standard deviation ( $\log \sigma$ ), and critic with PPO.
16: end for

```

C SHOES Stage-2 Procedure

Algorithm 1 summarizes the SHOES Stage 2 adaptation loop used in both simulation and on-robot runs.

D SafeBox Default Parameters

Defaults (Table 7) are identical in simulation and on hardware. The waypoint is drawn uniformly from the box and held until the robot enters the target-reached radius ϵ , or on hardware until the hold-time timer expires. Eq. (8) is projected to the body frame and clipped per Eq. (9) at every 50 Hz tick using the latest yaw, so yaw drift bends the command smoothly rather than producing a step change.

Table 7: SafeBox sampler defaults.

Parameter	Value
Box half-width a (x-axis)	1.5 m
Box half-width b (y-axis)	1.0 m
Cruise-speed lower bound $v_{\text{cap}}^{\min}$	0.1 m/s
Cruise-speed upper bound $v_{\text{cap}}^{\max} (= v_x^{\max})$ / body-frame $ v_x $ clip $v_x^{\max}$	1.0 m/s
Body-frame $ v_y $ clip $v_y^{\max}$	0.5 m/s
Max $ \omega_z $ ($\omega_{\max}$)	0.5 rad/s
Position gain k_p	2.0 s^{-1}
Target-reached threshold ϵ	0.1 m
Hold-time window (hardware)	[3, 8] s

E Evaluation Protocol

Performance is measured on a 24-cell velocity-tracking benchmark formed by crossing 3 speed set-points (0.3, 0.6, 1.0 m/s) with 8 directions (45° increments). Each cell is evaluated with $\text{num_envs} = 16$ parallel environments, 100 warmup steps, and $T = 200$ measurement steps. For each cell c with commanded direction $\hat{\mathbf{c}}_c \in \mathbb{R}^2$, we record the commanded-direction body-frame velocity, averaged over time and environments and clipped at zero:

$$v^c = \max\left(0, \overline{\mathbf{v}_{xy}} \cdot \hat{\mathbf{c}}_c\right), \quad \overline{\mathbf{v}_{xy}} = \frac{1}{T} \sum_{t=1}^T \mathbf{v}_{xy}(t). \quad (11)$$

The primary figure of merit is the Bare-Foot Parity (BFP), the mean per-cell ratio against the bare-foot nominal,

$$\text{BFP} = \frac{1}{24} \sum_{c=1}^{24} \frac{v_{\text{policy}}^c}{v_{\text{bare}}^c} \times 100 \%. \quad (12)$$

A score of $\text{BFP} = 100 \%$ indicates parity with the bare-foot reference in the commanded direction; lower scores mean the policy tracks more slowly. Note that BFP measures only the commanded-direction component: lateral drift and yaw error contribute only insofar as they reduce it. The main-table and component-ablation rows aggregate six seeds (mean $\pm$ std and best); the sweeps in Appendix H use a single seed per setting.

F Per-Method Results, Full Numbers

Table 8 expands Figure 5 with the best-seed column. Each row is six seeds at 30k env-steps with the SafeBox sampler from a common nominal checkpoint. The first row is the frozen nominal evaluated on the shoe-equipped robot.

Table 8: Method comparison on flat shoes. Mean $\pm$ std and best Bare-Foot Parity (BFP) over six seeds.

Method	Trainable	BFP $\uparrow$	Best $\uparrow$
nominal (no adapt)	0	67.0	—
Adapter-only	0.2k	65.1 ± 8.9	76.1
FiLM-only	5.4k	78.7 ± 10.8	90.3
Residual SAC	$\sim 38\text{k}$	66.7 ± 0.2	66.8
Residual PPO	$\sim 38\text{k}$	63.7 ± 8.3	81.8
SHOES (Ours)	5.6k	93.5 ± 3.9	96.9

G Real-Robot Protocol

A 50 Hz policy thread reads the latest IMU and T265 pose, builds the same 47-d observation as in simulation, and publishes the next PD target; a 500 Hz onboard PD thread executes it at the joint level. Reward is computed online from T265 linear velocity (rotated into the body frame via the IMU quaternion) and IMU yaw rate, using the Stage-2 reward (Appendix B). SafeBox supplies the velocity command, with the T265 providing world-frame pose. An emergency-stop button on the wireless controller triggers a damping-mode hand-off independent of the policy pipeline.

H FiLM Bound Width and Generator Capacity Sweeps

Single-seed trends justifying the FiLM hyper-parameter choices.

Bound width α . The multiplicative scale is $1 + \alpha \tanh(\gamma_{\text{raw}}) \in [1 - \alpha, 1 + \alpha]$. Unbounded FiLM ($\alpha \rightarrow \infty$, no $\tanh$ saturation) collapses to a fall-avoidance shortcut around 67 %; $\alpha = 0.3$ is too tight and trails $\alpha = 0.5$; $\alpha = 1.0$ trends back toward the unbounded shortcut. We use $\alpha = 0.5$ throughout.

Generator hidden dimension. For the generator $\text{Linear}(47 \rightarrow h) \rightarrow \text{ELU} \rightarrow \text{Linear}(h \rightarrow h) \rightarrow \text{ELU} \rightarrow \text{Linear}(h \rightarrow 256)$ we use $h = 16$. $h = 8$ underfits the shoe-induced shift; $h \in \{32, 64\}$ overfits and trends toward the unbounded shortcut.