COMPLIANT ROBOTIC CONTROL WITH GENERATIVE PRETRAINED CONTROLLERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Humanoid robots operating in human environments must interact safely and robustly with both humans and their surroundings. They must remain stable under disturbances, respond compliantly to external forces, and recover naturally from falls. We introduce Robotic Generative Pretrained Controller (Robo-GPC), a discrete generative controller that learns motor skills from large-scale human motion data and adapts to downstream whole-body control tasks through parameter-efficient fine-tuning. Robo-GPC represents motor skills in a compact Finite Scalar Quantization (FSQ) latent space and models their distribution with an autoregressive transformer decoder. By reusing the learned skill during task adaptation, Robo-GPC produces natural behaviors while supporting a wide range of downstream control tasks, providing a general framework for humanoid deployment. Deployed on a Unitree G1 humanoid, Robo-GPC enables robust whole-body control while responding naturally to pushes and pulls and recovering from large disturbances and falls. The supplementary videos are present in `https://anonymous.4open.science/r/robogpc/index.html`.

1 INTRODUCTION

Humanoid robots are increasingly expected to operate in environments shared with people, from homes to workplaces. In such settings, physical interaction is inevitable: a robot may be bumped into obstacles, leaned on while assisting a user, or forced to yield when a person enters its workspace. Rigidly resisting such interactions can generate large contact forces, potentially damaging the robot or harming nearby people. Humanoids must therefore respond in a stable and natural manner to external forces rather than maintaining their nominal motion (Margolis et al., 2025; Liu et al., 2026). Prior work focus on compliant motion tracking. Compliance typically refers to allowing body parts subject to external forces to deviate from the reference motion, while the rest of the body continues to track the motion.

In this work, we use *compliance* in a behavioral sense to describe a controller's ability to adapt its behavior naturally in response to external forces and changes in its physical state. This encompasses a spectrum of responses, including adapt its behavior under perturbations while following user-designated command, and transitioning safely into a recovering behavior from fallen states. Compliance therefore extends beyond simply reducing joint stiffness when tracking reference motion: it also concerns whether the controller can select an appropriate and natural whole-body response as the robot deviates from its nominal behavior.

We present Robotic Generative Pretrained Controller (Robo-GPC), a discrete generative controller that models a distribution of natural motor behaviors rather than prescribing specific responses to perturbations. Building on GPC (Shi et al., 2026), Robo-GPC quantizes motion into discrete latent skills with Finite Scalar Quantization (FSQ) (Mentzer et al., 2023) and models their distribution with a transformer decoder. We pretrain on diverse human motions spanning locomotion, gestures, falls, and recoveries, allowing contrusting a latent space that contain transitions toward human-like behaviors. We then adapt the frozen prior to downstream tasks with Conditional Low-rank Adaptation (CoLA), while preserving the behaviors acquired during pretraining.

We introduce several key design choices to enable effective sim-to-real transfer. First we use a compact FSQ vocabulary to reduce inference cost and support real-time control on hardware. Deployed

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

on a Unitree G1 humanoid, Robo-GPC demonstrates robust responses to a range of external interactions, including human-applied pushes and pulls. Compliant yielding and fall recovery emerges during these interactions without using rewards or curricula explicitly targeting these behaviors. More broadly, our results suggest that combining a generative motor prior with parameter-efficient task adaptation can produce robust and compliant controllers across downstream tasks without explicitly engineering compliance or recovery into each task objective. This points toward a more data-driven path to safe and robust humanoid control.

2 Related Work

2.1 Humanoid Whole-Body Control from Motion Data

Learning from human motion data has become the standard route to acquiring naturalistic skills for humanoid robots. Tracking-based controllers receive dense rewards for matching the full target state of a reference motion (Peng et al., 2018; Zhang et al., 2025), and prior studies in physics-based character animation scales this formulation to large-scale motion datasets, training a single controller across hundreds of hours of motion clips (Shi et al., 2026; Luo et al., 2024). Recent advances in humanoid whole-body control have enabled successful deployment of tracking-based controllers on physical humanoid robots (Liao et al., 2025; Zeng et al., 2026; Chen et al., 2025b; Luo et al., 2025; Qi et al., 2026). However, tracking alone does not directly solve downstream tasks, as the controller still requires reference trajectories to follow. For complex tasks, this leads to a multi-stage pipeline in which a separate kinematic motion planner generates reference motions for the tracking controller (Zhang et al.; Xu et al., 2025). Such planner-tracker systems can accumulate errors between the separately trained planner and tracker, requiring additional mechanisms to maintain robustness and consistency (Tevet et al., 2025; Chen et al., 2026).

2.2 Generative Controllers

Beyond tracking, a growing body of work learns latent representations that capture the underlying distribution of motor skills. Early approaches such as ASE (Peng et al., 2022) learn reusable skill latents through adversarial imitation. However, adversarial objectives can be difficult to scale to large and diverse motion datasets, motivating approaches that distill motion representations from pretrained tracking controllers. Prior work has explored VAEs (Won et al., 2022; Luo et al., 2024; Tessler et al., 2024), diffusion models (Truong et al., 2024; Huang et al., 2025b; Wu et al., 2025), and discrete latent representations (Yao et al., 2023; Zhu et al., 2023; Shi et al., 2026).

For downstream control, a key challenge is preventing the task policy from selecting poorly supported latents under model error or out-of-distribution states. Prior work addresses this by constraining continuous latents (Peng et al., 2022; Zhang et al., 2026) or restricting control to discrete codes grounded in motion data (Zhu et al., 2023; Yao et al., 2023). GPC (Shi et al., 2026) scales the discrete formulation to larger datasets with Finite Scalar Quantization and next-token prediction, while also exhibiting compliant responses and natural fall recovery in simulation.

2.3 Compliant Robotic Control

Compliance is critical for hardware deployment, where a humanoid must respond naturally to physical contact and recover safely from large perturbations. We use *compliance* in a behavioral sense, encompassing yielding, balance recovery, and transitions from fallen states back to nominal motion.

Prior work on compliance is largely built on motion-tracking controllers. Because deviations from the reference appear as tracking error, compliant behavior must be introduced explicitly. SoftMimic (Margolis et al., 2025) trains on compliant reference motions generated under external wrenches, LAC (Liu et al., 2026) extends this formulation to angular compliance, and CHIP (Chen et al., 2025a) instead introduces compliance through hindsight relabeling within the tracking framework.

Fall recovery is, on the other hand, often handled separately with dedicated objectives. Prior methods train get-up policies using discovered trajectories, reward shaping, and curricula (He et al., 2025;

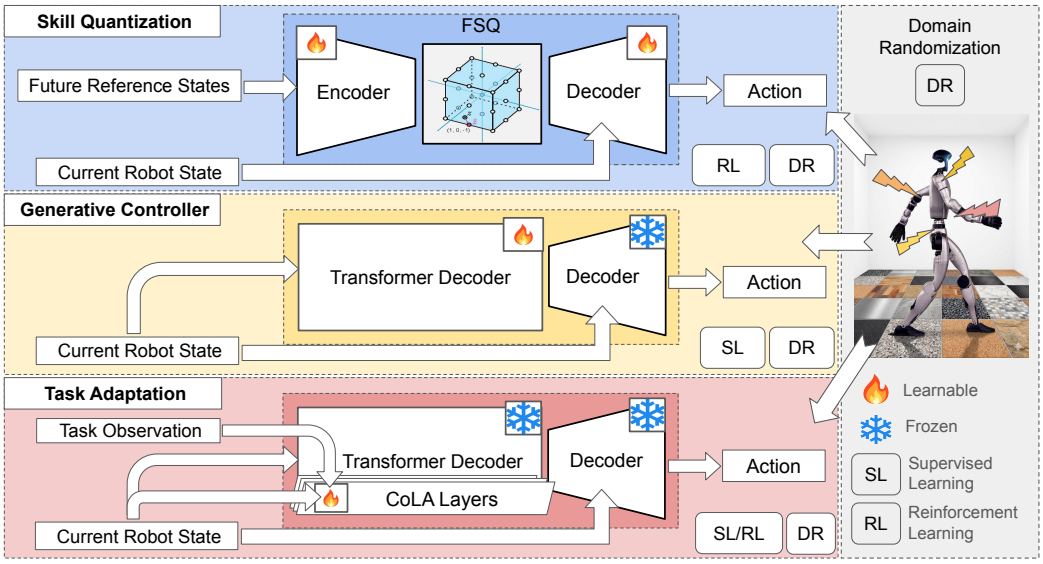


Figure 1: Overview of the three-stage Robo-GPC training pipeline. **Skill quantization**: An FSQ-based motion-tracking policy is trained jointly with reinforcement learning using a motion-tracking objective (top). **Generative controller training**: A causal Transformer decoder learns the distribution of discrete skills conditioned on the character state (middle). **Task adaptation**: Lightweight CoLA layers are introduced to adapt the frozen pretrained generative controller to downstream tasks. CoLA supports Supervised Finetuning to learn based on supervised task signal (optional) and Reinforcement Funetuning for downstream task adaptation. Domain randomization is applied during all three training stages to expose the policies to variations in simulated dynamics and improve robustness.

Huang et al., 2025a), while BFM-Zero (Li et al., 2026), building on (Tirinzoni et al.), discovers transitions between fallen and standing states through unsupervised skill learning.

Robo-GPC takes a different approach: compliant responses and fall recovery emerge from a generative motor prior pretrained on human motion. By modeling a discrete distribution of natural skills, the controller can transition to plausible alternative behaviors as perturbations move the robot away from its nominal motion, without explicitly engineering compliance or recovery into the downstream task objective.

3 Method

In this section, we provide an overview of Robo-GPC and discuss key design choices for effectively transfer to real robots. We first introduce our three-stage training pipeline in Section 3.1. Next, we provide an overview of the dataset composition used to train our controllers (Sec. A.1). Then, we provide implementation details about skill quantization (Sec. 3.2), generative controller (Sec. 3.3) and finally task adaptation (Sec. 3.4).

3.1 Preliminaries: Generative Pretrained Controller

Robo-GPC builds on the Generative Pretrained Controller (GPC) (Shi et al., 2026), and extends it toward real-world humanoid control by adapting the pretrained generative controller for pratical downstream control tasks for humanoids. At a high level, as illustrated in Figure 1, GPC is a generative controller that masters a wide range of human skills. It first encodes reference motion into discrete latent codes and decodes these codes into targets for low-level PD control. A GPT-style causal transformer is then trained to model the distribution of the resulting sequence of discrete skill tokens from tracking. This enables the controller to generate whole-body motion unconditionally by sampling from the learned distribution over skills, without requiring an explicit reference trajectory.

During *skill quantization*, a tracking policy is trained with a Finite Scalar Quantization bottleneck on the embedding of the reference motion. An encoder compresses a window of target states into a latent $\mathbf{z}_t \in \mathbb{R}^d$, where d is the latent dimension, and each dimension is quantized independently to one of L levels:

$$\hat{\mathbf{z}}_t = \left\lfloor \left\lfloor \tfrac{L}{2} \right\rfloor \tanh(\mathbf{z}_t) \right\rceil . \tag{1}$$

This quantization approach defines an implicit codebook of L^d entries without requiring learned embeddings (Mentzer et al., 2023). A decoder $\mathcal{D}(\mathbf{a}_t \mid \mathbf{o}_t, \hat{\mathbf{z}}_t)$ maps the discrete code $\hat{\mathbf{z}}_t$ and proprioceptive observation $\mathbf{o}_t$ to target joint positions $\mathbf{a}_t$, which are tracked by low-level PD controllers. The encoder and decoder are trained jointly with PPO (Schulman et al., 2017) under a motion-tracking reward, with gradients passing through the rounding operation via a straight-through estimator (Bengio et al., 2013). Once the discrete latent skill representation has been learned, the d scalar codes are packed into groups of G dimensions for more efficient sequence modeling, yielding $d' = d/G$ grouped tokens $\tilde{\mathbf{z}}_t = (\tilde{z}_t^1, \ldots, \tilde{z}_t^{d'})$ at each control step, where $\tilde{z}_t^j$ denotes the j-th grouped token.

A GPT-style causal transformer is then trained via next-token prediction on tokens produced by the frozen encoder:

$$p_\theta(\tilde{\mathbf{z}}_t \mid \mathbf{o}_t) = \prod_j p_\theta(\tilde{z}_t^j \mid \mathbf{o}_t, \tilde{\mathbf{z}}_t^{<j}). \tag{2}$$

where $\tilde{\mathbf{z}}_t^{<j} = (\tilde{z}_t^1, \ldots, \tilde{z}_t^{j-1})$ denotes the preceding grouped tokens within the same control step.

From any proprioceptive state, it places mass on the tokens that follow that state in the data. At deployment, the transformer runs autoregressively at each step with nucleus sampling (Holtzman et al., 2020), and obtains the action $\mathbf{a}_t$ from the frozen decoder $\mathcal{D}$.

Once the base generative controller has been trained, it can be adapted to new downstream tasks by inserting Conditional Low-rank Adaptation (CoLA) layers into the transformer for parameter-efficient fine-tuning, while keeping the base model weights frozen. CoLA combines the magnitude-direction decomposition of DoRA (Liu et al., 2024) with FiLM conditioning (Perez et al., 2018), allowing a single set of adapter parameters to produce task-conditioned adaptations based on the provided task input.

For a frozen pretrained weight $\mathbf{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ from generative controller, a task observation $\mathbf{c}$, and an input $\mathbf{x}$, CoLA produces an adapted weight $\mathbf{W}$, which is used during inference:

$$\mathbf{W}\mathbf{x} = \mathbf{W}_0\mathbf{x} + \mathbf{m}\,\frac{\mathbf{B}\mathbf{h}(\mathbf{x}, \mathbf{c})}{\|\mathbf{B}\mathbf{h}(\mathbf{x}, \mathbf{c})\|_2}, \mathbf{h}(\mathbf{x}, \mathbf{c}) = \operatorname{diag}(\boldsymbol{\gamma}(\mathbf{c}))\,\mathbf{A}\mathbf{x} + \boldsymbol{\beta}(\mathbf{c}), \tag{3}$$

where $\mathbf{A} \in \mathbb{R}^{r \times d_{\text{in}}}$ and $\mathbf{B} \in \mathbb{R}^{d_{\text{out}} \times r}$ are low-rank matrices with rank $r \ll \min(d_{\text{in}}, d_{\text{out}})$, $\mathbf{m} \in \mathbb{R}^{d_{\text{out}}}$ is a learned magnitude, and $\boldsymbol{\gamma}, \boldsymbol{\beta}$ are FiLM MLPs that map the task observation $\mathbf{c}$ to per-channel scale and shift. Normalizing the branch by its L_2 norm decouples the magnitude of the update from its direction, so adaptation re-weights the pretrained skill distribution rather than overwriting it. Adapters add under 1% of the base parameters, and the prior is never modified, so one pretrained controller backs every downstream task.

3.2 Humanoid Skill Quantization

Skill quantization is performed by an FSQ tracking controller trained end-to-end with reinforcement learning under a motion-tracking objective. At each step, the future reference motion is re-anchored to the robot's current torso position and aligned with the reference heading. The tracking reward is computed using the global reference trajectory, while the decoder receives only the local robot state, since it must operate without access to global trajectory information when used for downstream tasks. We find that the character's root position can still track the reference trajectory accurately under these partial observations.

For sim-to-real transfer, we follow the BeyondMimic recipe (Liao et al., 2025), and use the same settings throughout all subsequent training stages. The actor excludes the base linear velocity and anchor position, which are unavailable on hardware, while the critic has access to the privileged ground-truth state during training.

3.3 Generative Controller for Compliant Control

Once the FSQ tracking policy is trained, we use it to collect paired observations and codes in simulation. Notably, compliant responses and fall-recovery behaviors emerge at this stage, before any task-specific finetuning. This differs from conventional tracking-based controllers (Chen et al., 2025a; Liu et al., 2026), whose behavior is primarily driven by a reference motion and whose response to perturbations is therefore constrained by the tracking objective.

Robo-GPC instead operates without a reference trajectory: conditioned only on proprioception, the prior predicts codes corresponding to plausible behaviors from the current state. Under external perturbations, this naturally favors yielding responses rather than explicit correction toward a reference. Nucleus sampling (Holtzman et al., 2020) further filters out implausible codes to encourage on-manifold behavior.

3.4 Task Adaptation

After the GPC prior is trained as described in Sec. 3.3, we can perform supervised finetuning (SFT) on the CoLA adapter using the paired task observations and ground-truth latent codes to shift the distribution toward the task-conditioned behavior:

$$\begin{aligned}\mathcal{L}_{\mathrm{SFT}} = &- \mathbb{E}_t \left[\log q_{\mathrm{CoLA}}(z_t^* \mid s_t)\right] \\ &+ \beta_{\mathrm{KL}} \mathbb{E}_t \left[D_{\mathrm{KL}}(q_{\mathrm{CoLA}}(z_t \mid s_t) \,\|\, q_{\mathrm{GPC}}(z_t \mid s_t))\right].\end{aligned} \tag{4}$$

where z_t^* denotes the ground-truth FSQ code produced by the tracking policy, q_{CoLA} is the latent distribution induced by the adapted controller, and q_{GPC} denotes the latent distribution of the pretrained generative controller.

To preserve the compliant and responsive behaviors acquired during pretraining, we regularize the adapted policy with a KL penalty toward the pretrained controller. Without this penalty, fine-tuning on task data can collapse the latent distribution onto a narrow set of behaviors that reproduce the target motion but lose the prior's ability to yield to external forces and respond to perturbations.

With a pretrained unconditional or an SFT-pretrained generative controller, we can also further adapt the pretrained generative controller to downstream tasks through reinforcement learning fine-tuning (RLFT) with CoLA. The CoLA adapter is optimized using PPO while being regularized toward the latent distribution of the pretrained GPC prior. Specifically, we define the per-step RLFT reward as

$$\tilde{r}_t = r_t^{\mathrm{task}} + r_t^{\mathrm{reg}} - \beta_{\mathrm{KL}} D_{\mathrm{KL}}(q_{\mathrm{CoLA}}(z_t \mid s_t) \,\|\, q_{\mathrm{GPC}}(z_t \mid s_t)), \tag{5}$$

Here, $q_{\mathrm{CoLA}}(z_t \mid s_t)$ and $q_{\mathrm{GPC}}(z_t \mid s_t)$ denote the latent distributions of the adapted controller and the pretrained GPC controller, respectively. r_t^{task} denotes the task-specific reward and r_t^{reg} contains the shared regularization terms. We then optimize the expected discounted return using PPO. KL regularization can be applied whether RLFT is initialized from the unconditional pretrained controller or from an SFT-adapted controller.

Unlike GPC (Shi et al., 2026), which employs an unconditional-prior thresholding mechanism, we impose this penalty directly during fine-tuning through the KL penalty. As a result, the unconditional generative controller does not need to be evaluated at inference time, reducing inference cost and facilitating onboard deployment. We further replace top-p nucleus sampling with the Gumbel-Max trick, avoiding the ranking operation required by nucleus sampling and further reducing sampling overhead.

3.5 Downstream Tasks

We evaluate our controller across a range of downstream tasks that demonstrate its natural and compliant behaviors, as well as the robustness required for safe physical interaction between the robot and users. We introduce these tasks in this section, and for details, refer to Appendix A.4.

Robust-Mimic We fine-tune the CoLA adapter with RLFT. To leverage the recovery skills acquired by the pretrained generative controller, we randomly initialize the robot in grounded states

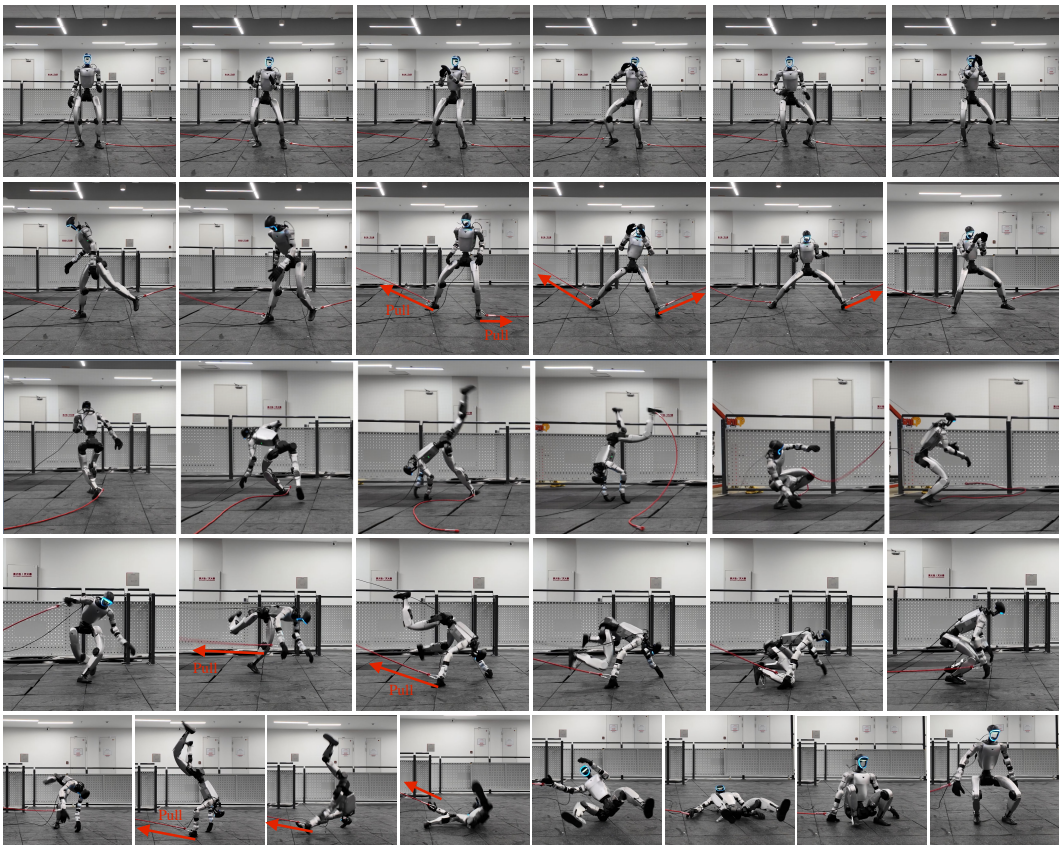


Figure 2: **Robust-Mimic under external perturbations.** The first two rows show dance motions, while the next two rows show cartwheel motions. In the first and third rows, the robot tracks these dynamic motions without external perturbations. In the second and fourth rows, external forces are applied to the legs and the right hand, respectively, yet the robot remains stable and continues tracking the reference motion while preserving the intended behavior. In the final row, the robot recovers from a perturbation-induced fall during a highly dynamic cartwheel motion.

and independently sample reference starting points. This allows the tracking policy to incorporate natural behaviors from the pretrained generative controller to enable successful tracking. This two-stage adaptation enables stable online reference switching and preserves robust recovery behaviors.

Joystick Steering We develop a joystick steering controller by commanding the robot with a desired facing direction and a target velocity. The controller must follow these high-level commands while autonomously generating coordinated full-body locomotion. We train the task policy with RLFT. This task evaluates whether Robo-GPC can achieve responsive user control while retaining natural recovery behaviors when the robot is pushed or otherwise perturbed during locomotion.

Stylized Steering Robo-GPC can also be adapted to stylized steering tasks, where the robot must follow locomotion commands while exhibiting a specified motion style. This setting demonstrates that the generative controller can acquire and preserve expressive motion styles through SFT, then reuse these stylized skills for downstream control through RLFT, while remaining robust to external disturbances and still having natural fall behavior skill.

VR Two-Point Tracking Robo-GPC supports sparse VR-based motion control via RLFT, where the controller tracks only the positions of a small set of user-controlled targets rather than a complete reference motion. In particular, the robot is conditioned on two tracked end-effectors derived from the VR interface and must generate a plausible full-body response that satisfies these sparse constraints. Since the task leaves much of the body motion unspecified, it provides a natural setting for

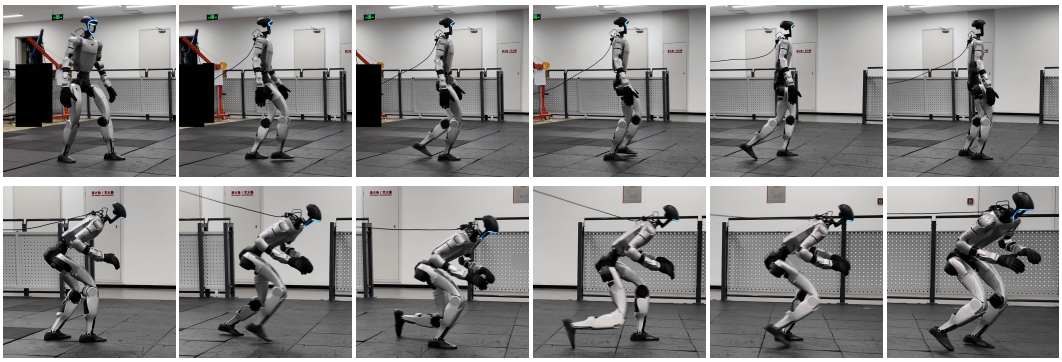


Figure 3: We can directly adapt generative controller to a steering task using RLFT, allowing the controller to automatically select the skills that maximize the task reward (upper). Alternatively, we can first adapt the controller on stylized motions with SFT before RLFT, enabling the controller to accomplish the downstream task using behaviors of the desired style (lower).

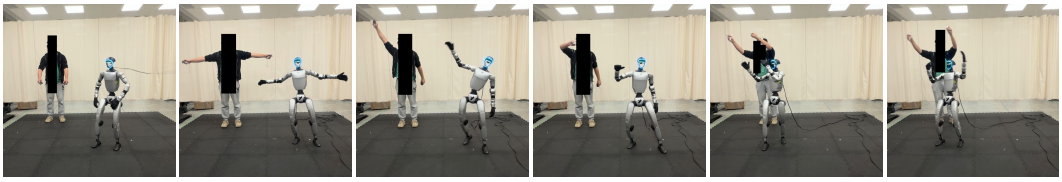


Figure 4: Our controller can be adapted to a two-point VR tracking task using VR controllers through RLFT. The resulting policy robustly tracks the sparse VR targets corresponding to the user's hand positions, while retaining the ability to recover from large disturbances and falls.

evaluating whether the pretrained generative controller can fill in unconstrained motion while maintaining balance, robustness, and compliant behavior during physical interaction. We formulate the target by extracting it from an arbitrary frame in the data and randomly generating target positions.

Safe-Fall As shown in Figure 5, switching a standing robot directly to damping mode can cause an uncontrolled collapse under gravity, leading to large impact forces and potentially damaging contacts with vulnerable components. Prior work on humanoid fall mitigation similarly emphasizes controlling the landing poses to reduce impact and protect the robot (Meng et al., 2025; Shi et al., 2025).We introduce a *safe fall* task, where the robot learns to coordinate its descent from arbitrary state and reach a stable configuration on the ground while avoiding high-impact contacts. Our safe-fall formulation specifies the safety objective through reinforcement learning, while the pretrained generative controller provides a prior over plausible human-like motor skills.

4 Experiments

We first describe the experimental setup in Sec. 4.1. We then compare Robo-GPC with planner-tracker methods in Sec. 4.2. For more detailed design choices and dataset settings, refer to A.4.

4.1 Experiment Settings

All policies reported in this section are trained with Isaac Lab (Mittal et al., 2025) on eight NVIDIA L20 GPUs. Simulation runs at 120 steps per second with a 50 Hz control rate. Prior to hardware deployment, each policy is validated in MuJoCo (Todorov et al., 2012) as a sim-to-sim evaluation, which runs at 500 Hz and is then deployed on a Unitree G1 29-DoF humanoid robot. Performance evaluations are primarily conducted in simulation. During deployment, we perform inference on a

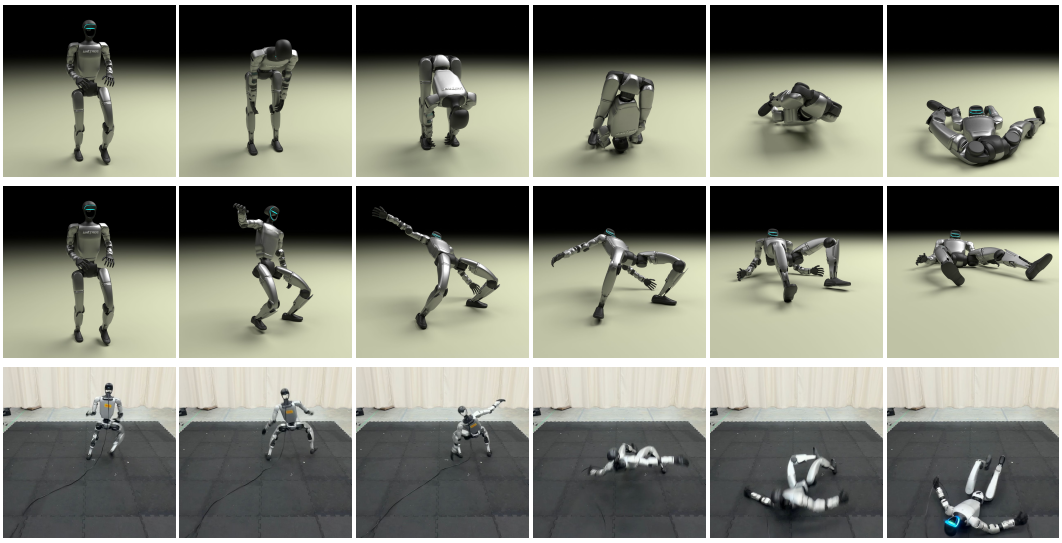


Figure 5: When the robot is standing and the controller switches to damping mode, it naturally collapses to the ground under gravity (upper). In contrast, for the safe-fall task, the robot learns human-like protective behaviors through reinforcement learning, reducing impact forces across the body and avoiding damaging contacts with vulnerable components such as motors and mechanical joints (middle). We deploy the controller on the physical robot, where it serves as a learned emergency-stop mechanism when we (bottom).

workstation with NVIDIA RTX 4090 GPU connected to the robot via an Ethernet cable. We export the controller to ONNX and use TensorRT for accelerated inference.

4.2 Comparison with Steering Planner-Tracker Controllers

Planner-tracker approaches combine a kinematic generator with a tracking controller. We evaluate two planner designs: ARDY (Zhao et al., 2026), and a GRU-based motion generator (Cho et al., 2014), both paired with SONIC as the tracking controller. We measure the success rate on a joystick-steering task, where external forces with a scale ranging from 0-1000N are applied to the torso from eight uniformly spaced directions; success is defined as steering commands achieved while remaining upright after 8s. As shown in Fig. 6, both planner-tracker approaches drop to a 0% success rate at 800N, whereas Robo-GPC remains active and capable of completing the task under strong perturbations. This highlights Robo-GPC's greater robustness to external disturbances and its ability to recover from falls compared with planner-tracker methods.

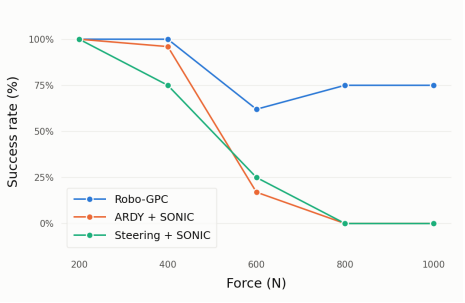


Figure 6: Push robustness in the Joystick Steering task. Success rate measures how often the controller successfully completes the commanded task under external forces of varying magnitudes.

4.3 Comparison with other Generative Controller

After training, the generative controller can sample diverse motor skills unconditionally from the learned skill distribution. Despite receiving no explicit training for robustness or fall recovery at this stage, it exhibits strong disturbance tolerance and can recover naturally from falls, as shown in Fig. 7. It also generates a diverse range of behaviors and outperforms prior generative approaches such as PULSE, as shown in Table 1.

Table 1: Unconditional sampling results for Robo-GPC and PULSE. Each condition uses 32 initial states and 8 samples per state. APD is computed from 896 sample pairs per method.

	Free			Pushed		
Model	APD Pose ↑	APD Root ↑	Fall (%) ↓	APD Pose ↑	APD Root ↑	Fall (%) ↓
Robo-GPC	0.103	1.47	2.0	0.217	1.49	24.2
PULSE	0.092	0.98	4.7	0.283	0.88	65.2

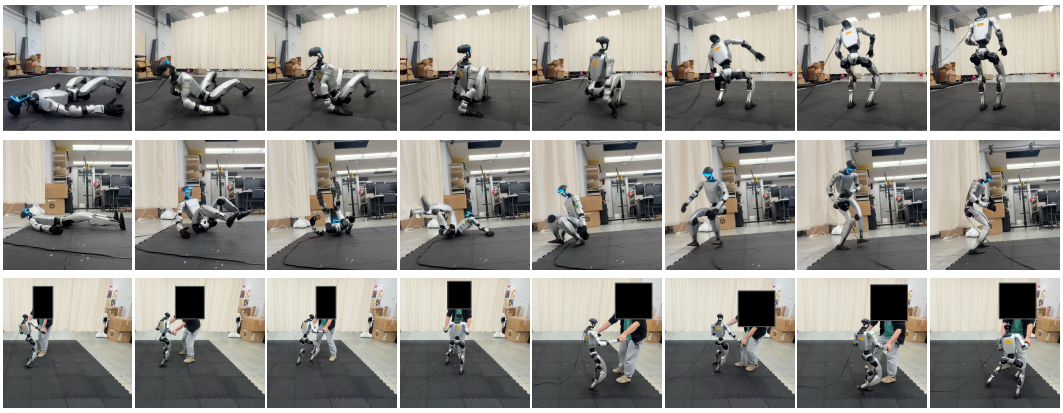


Figure 7: The robot can recover with unconditional sampling from fallen states(top). When adapted to downstream tasks such as steering, the controller retains this recovery behavior without requiring explicit recovery objectives or task-specific design (middle). The robot responds to external forces by transitioning to appropriate skills (bottom).

4.4 Safe-Fall

We compare our safe-fall policy against the standard G1 damping mode in MuJoCo. For each trial, the robot is initialized in a randomly sampled pre-fall state and subjected to an external push with different directions and magnitudes, after which either the damping controller or our safe-fall controller is activated. We conduct eight trials for each perturbation condition, resulting in 72 paired trials in total. Our policy reduces the peak floor contact force across all conditions, with an average reduction of 45%. The largest improvements occur during forward and backward falls, where the damping controller often causes the robot to land flat on its chest or back.

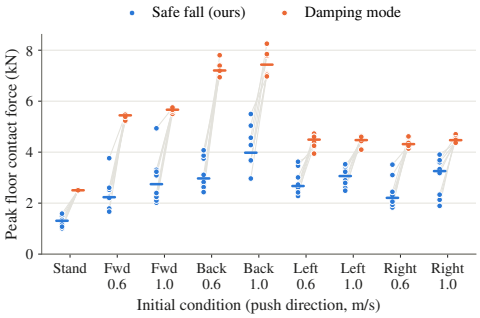


Figure 8: Evaluation of contact force with the floor using damping and Safe-Fall

5 Limitation and Discussion

Robo-GPC still has several limitations. First, our current downstream tasks rely only on proprioceptive inputs and do not incorporate visual perception. Integrating vision could enable more practical behaviors in everyday environments. For example, a fall controller could account for surrounding obstacles and terrain geometry, allowing the robot to select safer landing configurations and more precisely control the descent to reduce potential damage. Second, Robo-GPC can respond to external force with reasonable skills, but the stiffness of the joints cannot be controlled by the user. In the future, it might be good to have such functionality. Despite these limitations, Robo-GPC demonstrates that a single generative controller can capture diverse motor skills, adapt efficiently to downstream tasks, and naturally exhibit safety-relevant interaction behaviors, including compliant responses and controlled falling.

AI USE STATEMENT

In this work, we used generative AI tools to assist with editing manuscripts, including improvements to grammar, clarity, and phrasing. We also used generative AI tools to assist with code review and debugging during the development of our system. We did not use generative AI tools to generate research ideas, synthesize experiment results, conduct experiments, analyze data, or draw scientific conclusions. Other disclosure categories are not applicable to this work.

All AI-assisted text and code suggestions were reviewed and verified by the authors for technical correctness, consistency with the experimental results, and appropriate attribution of prior work. The authors take full responsibility for the final content of this work, including any text, code, claims, or other materials produced with the assistance of generative AI tools.

REPRODUCIBILITY STATEMENT

We have taken several steps to support the reproducibility of our results. Detailed descriptions of the model architecture, training procedure, objective functions, and implementation choices are provided in Sec. 3.1 and Sec. 3.4. The experimental setup, datasets, evaluation protocols, and baseline configurations are described in Sec. 4.1. We also include supplementary materials containing additional implementation details, configuration files, and qualitative results. Source code and pretrained model checkpoints will be released to facilitate reproduction of the reported results.

REFERENCES

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Bones Studio. BONES-SEED: Skeletal Everyday Embodied Dataset. `https://bones.studio/datasets`, 2026.

Sirui Chen, Zi-Ang Cao, Zhengyi Luo, Fernando Castañeda, Chenran Li, Tingwu Wang, Ye Yuan, Linxi Fan, C. Karen Liu, and Yuke Zhu. CHIP: Adaptive compliance for humanoid control through hindsight perturbation. *arXiv preprint arXiv:2512.14689*, 2025a.

Xiao Chen, Weishuai Zeng, Xiaojie Niu, Zirui Wang, Jianan Li, Huayi Wang, Furui Xu, Jiahe Chen, Weixiang Zhong, Lihe Ding, Kailin Li, Jiangmiao Pang, Tai Wang, Tianfan Xue, and Jingbo Wang. Reactivebfm: Reactive closed-loop motion planning towards universal humanoid whole-body control, 2026. URL `https://arxiv.org/abs/2606.30362`.

Zixuan Chen, Mazeyu Ji, Xuxin Cheng, Xuanbin Peng, Xue Bin Peng, and Xiaolong Wang. Gmt: General motion tracking for humanoid whole-body control. *arXiv preprint arXiv:2506.14770*, 2025b.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, Doha, Qatar, 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179.

Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. *ACM Transactions on Graphics (TOG)*, 39(4), 2020.

Xialin He, Runpei Dong, Zixuan Chen, and Saurabh Gupta. Learning getting-up policies for real-world humanoid robots. In *Robotics: Science and Systems (RSS)*, 2025. arXiv:2502.12152.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*, 2020.

Tao Huang, Junli Ren, Huayi Wang, Zirui Wang, Qingwei Ben, Muning Wen, Xiao Chen, Jianan Li, and Jiangmiao Pang. Learning humanoid standing-up control across diverse postures. In *Robotics: Science and Systems (RSS)*, 2025a. arXiv:2502.08378.

Xiaoyu Huang, Takara Truong, Yunbo Zhang, Fangzhou Yu, Jean Pierre Sleiman, Jessica Hodgins, Koushil Sreenath, and Farbod Farshidian. Diffuse-cloc: Guided diffusion for physics-based character look-ahead control. *ACM Transactions on Graphics (TOG)*, 44(4):1–12, 2025b.

Yitang Li, Zhengyi Luo, Tonghe Zhang, Cunxi Dai, Anssi Kanervisto, Andrea Tirinzoni, Haoyang Weng, Kris Kitani, Mateusz Guzek, Ahmed Touati, et al. Bfm-zero: A promptable behavioral foundation model for humanoid control using unsupervised reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 79697–79725, 2026.

Qiayuan Liao, Takara E Truong, Xiaoyu Huang, Yuman Gao, Guy Tevet, Koushil Sreenath, and C Karen Liu. Beyondmimic: From motion tracking to versatile humanoid control via guided diffusion. *arXiv preprint arXiv:2508.08241*, 2025.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In *International Conference on Machine Learning (ICML)*, 2024.

Yang Liu, Zhongkai Gu, Wei Zhu, and Mitsuhiro Hayashibe. Lac: Linear and angular compliance for humanoid whole-body control. *arXiv preprint arXiv:2608.25405*, 2026.

Zhengyi Luo, Jinkun Cao, Josh Merel, Alexander Winkler, Jing Huang, Kris Kitani, and Weipeng Xu. Universal humanoid motion representations for physics-based control. In *International Conference on Learning Representations (ICLR)*, 2024.

Zhengyi Luo, Ye Yuan, Tingwu Wang, Chenran Li, Fernando Castañeda, Sirui Chen, Zi-Ang Cao, Jiefeng Li, David Minor, Qingwei Ben, et al. Sonic: Supersizing motion tracking for natural humanoid whole-body control. *arXiv preprint arXiv:2511.07820*, 2025.

Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5442–5451, 2019.

Gabriel B. Margolis, Michelle Wang, Nolan Fey, and Pulkit Agrawal. Softmimic: Learning compliant whole-body control from examples. *arXiv preprint arXiv:2510.17792*, 2025.

Ziyu Meng, Tengyu Liu, Le Ma, Yingying Wu, Ran Song, Wei Zhang, and Siyuan Huang. Safefall: Learning protective control for humanoid robots, 2025.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple, 2023. URL `https://arxiv.org/abs/2309.15505`.

Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Ori Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, et al. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel Van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)*, 37(4):1–14, 2018.

Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Trans. Graph.*, 41(4), July 2022.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI Conference on Artificial Intelligence*, pp. 3942–3951, 2018.

Zekun Qi, Xuchuan Chen, Dairu Liu, Chenghuai Lin, Yunrui Lian, Sikai Liang, Zhikai Zhang, Yu Guan, Jilong Wang, Wenyao Zhang, et al. Humanoid-gpt: Scaling data and structure for zero-shot motion tracking. *arXiv preprint arXiv:2606.03985*, 2026.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Diyuan Shi, Shangke Lyu, and Donglin Wang. Discovering self-protective falling policy for humanoid robot via deep reinforcement learning, 2025.

Yi Shi, Yifeng Jiang, Chen Tessler, and Xue Bin Peng. Gpc: Large-scale generative pretraining for transferable motor control. *arXiv preprint arXiv:2606.29148*, 2026.

Chen Tessler, Yunrong Guo, Ofir Nabati, Gal Chechik, and Xue Bin Peng. Maskedmimic: Unified physics-based character control through masked motion inpainting. *ACM Transactions On Graphics (TOG)*, 43(6):1–21, 2024.

Chen Tessler, Yifeng Jiang, Xue Bin Peng, Erwin Coumans, Yi Shi, Haotian Zhang, Davis Rempe, Gal Chechik, and Sanja Fidler. Protomotions3: An open-source framework for humanoid simulation and control. `https://github.com/NVLabs/ProtoMotions/`, 2025.

Guy Tevet, Sigal Raab, Setareh Cohan, Daniele Reda, Zhengyi Luo, Xue Bin Peng, Amit H. Bermano, and Michiel van de Panne. Closd: Closing the loop between simulation and diffusion for multi-task character control. In *International Conference on Learning Representations (ICLR)*, 2025.

Andrea Tirinzoni, Ahmed Touati, Jesse Farebrother, Mateusz Guzek, Anssi Kanervisto, Yingchen Xu, Alessandro Lazaric, and Matteo Pirotta. Zero-shot whole-body humanoid control via behavioral foundation models.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5026–5033, 2012.

Takara E. Truong, Michael Piseno, Zhaoming Xie, and C. Karen Liu. Pdp: Physics-based character animation via diffusion policy. In *SIGGRAPH Asia 2024 Conference Papers*. ACM, 2024.

Jungdam Won, Deepak Gopinath, and Jessica Hodgins. Physics-based character controllers using conditional vaes. *ACM Transactions on Graphics (TOG)*, 41(4), 2022. doi: 10.1145/3528223.3530067.

Yan Wu, Korrawe Karunratanakul, Zhengyi Luo, and Siyu Tang. Uniphys: Unified planner and controller with diffusion for flexible physics-based character control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

Michael Xu, Yi Shi, KangKang Yin, and Xue Bin Peng. Parc: Physics-based augmentation with reinforcement learning for character controllers. In *SIGGRAPH 2025 Conference Papers (SIGGRAPH '25 Conference Papers)*, 2025.

Heyuan Yao, Zhenhua Song, Yuyang Zhou, Tenglong Ao, Baoquan Chen, and Libin Liu. Moconvq: Unified physics-based motion control via scalable discrete representations, 2023. URL `https://arxiv.org/abs/2310.10198`.

Weishuai Zeng, Kangning Yin, Xiaojie Niu, Shunlin Lu, Weixiang Zhong, Jiahe Chen, Feiyu Jia, Xiao Chen, Zirui Wang, Furui Xu, Ming Zhou, Kailin Li, Weinan Zhang, He Wang, Li Yi, Dahua Lin, Jiangmiao Pang, and Jingbo Wang. Scaling behavior foundation model for humanoid robots. *arXiv preprint arXiv:2607.15163*, 2026.

Haotian Zhang, Ye Yuan, Viktor Makoviychuk, Yunrong Guo, Sanja Fidler, Xue Bin Peng, and Kayvon Fatahalian. Learning physically simulated tennis skills from broadcast videos. *ACM Trans. Graph.* doi: 10.1145/3592408.

Zhikai Zhang, Haofei Lu, Yunrui Lian, Ziqing Chen, Yun Liu, Chenghuai Lin, Han Xue, Zicheng Zeng, Zekun Qi, Shaolin Zheng, Qing Luan, Jingbo Wang, Junliang Xing, He Wang, and Li Yi. Learning athletic humanoid tennis skills from imperfect human motion data. *arXiv preprint arXiv:2603.12686*, 2026.

Ziyu Zhang, Sergey Bashkirov, Dun Yang, Yi Shi, Michael Taylor, and Xue Bin Peng. Physics-based motion imitation with adversarial differential discriminators. In *SIGGRAPH Asia 2025 Conference Papers (SIGGRAPH Asia '25 Conference Papers)*, 2025.

Kaifeng Zhao, Mathis Petrovich, Haotian Zhang, Tingwu Wang, Siyu Tang, and Davis Rempe. ARDY: Autoregressive diffusion with hybrid representation for interactive human motion generation. *arXiv preprint arXiv:2607.08741*, 2026.

Qingxu Zhu, He Zhang, Mengting Lan, and Lei Han. Neural categorical priors for physics-based character control. *ACM Transactions on Graphics (TOG)*, 42(6):1–16, 2023.

A Appendix

A.1 Dataset composition

We train the generative controllers on a mixture of three motion capture dataset chosen to provide broad behavioral coverage. AMASS contains multiple capture datasets spanning a wide range of human behaviors (Mahmood et al., 2019); we adopt the 40-hour segmentation and humanoid configuration used in prior work (Luo et al., 2024). LaFAN1 features long, high-quality motion-capture sequences covering standard locomotion, stylized maneuvers, and fall-recovery motions, totaling approximately 5 hours of data (Harvey et al., 2020). To further increase the amount of recovery behaviors, we additionally include 5 minutes of fall and get-up sequences from BONES (Bones Studio, 2026). Together, these datasets provide a diverse collection of skills for training and evaluating generative controllers. We retarget all motion data to the G1 humanoid using the retargeting pipeline provided by ProtoMotions (Tessler et al., 2025).

A.2 Reward Design for FSQ tracking controllers

Table 2: Reward terms used to train the FSQ tracking policy. Tracking terms encourage accurate reproduction of the reference motion, while regularization terms promote smooth, energy-efficient, and physically feasible behaviors.

Category	Term	Weight	Std./Scale
Tracking	Anchor position	0.5	0.3
	Anchor orientation	0.5	0.4
	Body position	1.0	0.3
	Body orientation	1.0	0.4
	Body linear velocity	1.0	1.0
	Body angular velocity	1.0	3.14
Regularization	Action rate	-0.1	–
	Energy	-2.0×10^{-5}	–
	Joint limit	-10.0	–

A.3 Hyperparameters for Pretraining Generative Controller

GPC (Shi et al., 2026) uses a large codebook with 40 scalar tokens and 9 quantization levels per token, together with token grouping to keep the resulting sequences tractable. We find that a substantially smaller representation is sufficient and adopt 16 tokens with 7 levels each while retaining comparable tracking performance across the dataset. To enable real-time onboard sampling for the generative controller, we group the codes into blocks of four, resulting in a vocabulary size of 7^4. This grouping reduces the number of autoregressive prediction steps, accelerating inference while also reducing the accumulation of errors over long autoregressive sequences. For generative controller, we used a transformer model with 3 layers and 8 heads.

A.4 Implementation Details of Downstream Task

Robust-Mimic We divide the parallel training environments equally between two initialization and termination regimes. In the first half, we use the same reference-state initialization (RSI) and strict early-termination criteria as in FSQ tracker training. This regime is intended to preserve precise

Table 3: RLFT reward terms for the VR hand-tracking task.

Category	Symbol	Term	Weight	Scale
Task	r_{task}	Hand-position tracking	1.0	0.5
		Hand-position precision	0.5	0.15
Regularization	r_{reg}	Action rate	-0.1	–
		Joint limit	-10.0	–
		Undesired contacts	-0.1	–

motion-tracking performance during RL fine-tuning. In the remaining half, we modify RSI-based initialization so that the robot starts either from a fallen state or from a randomly sampled frame of the full training motion dataset. For fallen-state initialization, we either sample from a hand-selected fallen-state subset of the training dataset or disable actuation for 1 s in simulation to generate a wider variety of fallen states. Random-frame initialization is intended to promote natural transitions when the target reference motion is switched online. To give the policy time to recover or transition onto the reference, we disable tracking-error-based early termination for the first 5 s of each episode in this second group, preventing large initial tracking errors from prematurely ending these episodes.

Joystick Steering We train a joystick steering downstream task policy to enable user control. The task requires the robot to orient toward a commanded facing direction while tracking a commanded velocity. Both commands are specified in a global frame anchored to the IMU yaw reference. This global frame is reset whenever a new facing command is issued; if the facing command remains unchanged, the reference is not updated. The velocity command is expressed in the coordinate frame defined by the commanded facing direction. Consequently, yaw drift accumulates only over the duration of a single facing command. At each control step, we transform both commands into the robot's yaw-local frame using the heading estimated from the IMU orientation. The policy therefore observes a 4D yaw-local command

$$\mathbf{o}^{steer} = [v_x, v_y, \hat{f}_x, \hat{f}_y],$$

where (v_x, v_y) denotes the target velocity and $(\hat{f}_x, \hat{f}_y)$ denotes the target facing direction under current active coordinate system.

During training, we set the target speed to zero with probability p_{stop}. The reward consists of exponential terms for facing direction and velocity direction, with the velocity-direction term disabled at near-zero speeds, along with a Gaussian term for speed tracking. It is worth noting that our controller has inherited the fall recovery skill and compliance from generative controller training.

VR Two-Point tracking For the VR tracking task, the user control the humanoid with the hand controllers. At each control step, the target hand positions, target velocities, and target-to-hand errors are transformed into the robot's yaw-local frame using the IMU yaw estimate. The policy receives a five-frame history of these features,

$$\mathbf{o}_t^{\text{VR}} = \left[\mathbf{p}_{t-4:t}^{\text{rel}}, \mathbf{e}_{t-4:t}, \mathbf{v}_{t-4:t}, \mathbf{m}_{t-4:t}, \mathbf{a}_{t-4:t}\right],$$

where the feature groups denote hand positions relative to the robot root, target-to-hand errors, target velocities, validity masks, and target ages. This representation removes global translation and yaw while preserving the relative geometry and motion of the two controller targets. During training, targets are drawn from valid motion frames, with the sampling probability of paired synthetic targets gradually increased to 50%. Synthetic targets preserve plausible two-hand geometry and, in half of the cases, follow smooth sinusoidal trajectories with consistent target velocities.

The reward consists of dense hand-position tracking and a sharper precision bonus, together with the regularization terms used by the joystick controller. The full-body imitation rewards are removed so that the policy learns to steer the pretrained generative controller toward the VR targets while retaining its recovery and compliant behavior.

Table 4: Reward components for safe-fall training. Impact weights are their final values; during the initial stage they increase from -0.0002, -0.0004, and -0.0004, respectively.

Term	Weight	Description
Horizontal orientation	0.5	Rewards either a face-up or face-down horizontal torso.
Low torso	1.5	Rewards lowering the torso to $0.12\,\mathrm{m}$ or less.
Pose precision	0.5	Sharp bonus for being both low and horizontal.
Stillness	0.5	Rewards low pelvis linear and torso angular velocities after reaching the lying pose.
Arms down	0.3	Rewards keeping the hands near the torso after reaching the lying pose.
Whole-body impact	-0.007	Penalizes contact forces above $327\,\mathrm{N}$ on any body.
Torso impact	-0.014	Penalizes torso contact forces above $150\,\mathrm{N}$.
Wrist impact	-0.014	Penalizes wrist contact forces above $100\,\mathrm{N}$.
Action rate	-0.1	Penalizes changes between consecutive actions.
Joint limits	-10.0	Penalizes joint positions beyond the soft limits.

Safe-Fall We train the safe-fall policy in simulation using RLFT while keeping the pretrained generative controller frozen. We utilize a KL penalty of 0.05 relative to the unconditioned prior. At each reset, an initial robot state is sampled from a random time in the motion archive, with perturbations to the root pose, velocity, and joint positions. For 50% of resets, this state is replaced by a state recorded during joystick-steering controller rollouts; these states include standing, moving, and disturbed configurations. The motion archive supplies initial states only and is not used as a reference trajectory. We train with 4096 parallel environments on the full motion dataset. The reward encourages the character to reach to a low, horizontal, settled pose and penalizes hard contacts, rapid action changes, and joint-limit violations (Table 4). The safe-fall actor receives two observation groups. The frozen pretrained policy takes an input comprising a 93 dimensions of proprioception. The CoLA adapter separately receives a task input containing five past frames of torso orientation, torso, pelvis, and head heights, pelvis linear velocity, torso angular velocity. The adapter uses these features to steer the frozen policy toward a controlled lying pose.

A.5 Trajectory Visualization of Unconditional Sampling

Figure 9 visualizes the pelvis trajectories projected onto the 2D plane for the unconditional sampling comparison with pretrained generative controller in Table 1. We show three representative initial states. For each state, Robo-GPC and PULSE each generate eight rollouts without perturbation and eight rollouts with an external push. In the unperturbed setting, Robo-GPC produces a broader range of trajectories, whereas PULSE tends to remain closer to the initial position. Under perturbations, the trajectories illustrate how individual rollouts respond to the push, including cases where the robot falls and remains stuck. These qualitative examples complement the aggregate diversity and fall-rate metrics reported in Table 1.

Unconditional sampling metrics. For each model and condition, we sample eight 10-s rollouts from each of 32 initial states, recording 500 post-control frames at 50 Hz.

APD Pose measures pose diversity from a shared initial state. At each frame, we express 13 tracked body positions (excluding the pelvis) relative to the pelvis position and rotate them by the inverse pelvis yaw. For each of the $\binom{8}{2} = 28$ rollout pairs, we average the Euclidean distances between corresponding body positions over all 13 bodies and 500 frames. We then average over pairs and initial states. This removes global translation and heading while retaining differences in body configuration. Frames after a fall are also included in the APD Pose calculation.

APD Root measures trajectory diversity. For each rollout pair, we compare their 3D pelvis displacements from the shared starting position, averaging the framewise Euclidean distance only over the common prefix before either rollout becomes fallen and stuck. Pairs with fewer than 50 common frames are excluded. We aggregate the remaining pairwise scores across initial states, weighted by the number of valid pairs; all 896 pairs were valid in each reported model–condition group.

Fall Rate is the fraction of 256 rollouts classified as fallen and stuck. A frame is considered fallen when the pelvis height is below $0.45\ \mathrm{m}$ or its uprightness is below 0.6. A rollout is classified as

stuck if it contains 50 consecutive fallen frames during which the pelvis moves less than $0.20, \mathrm{m}$ horizontally, and the robot remains fallen from the start of that window through the end of the rollout. A robot that falls but subsequently recovers is therefore not counted as stuck.

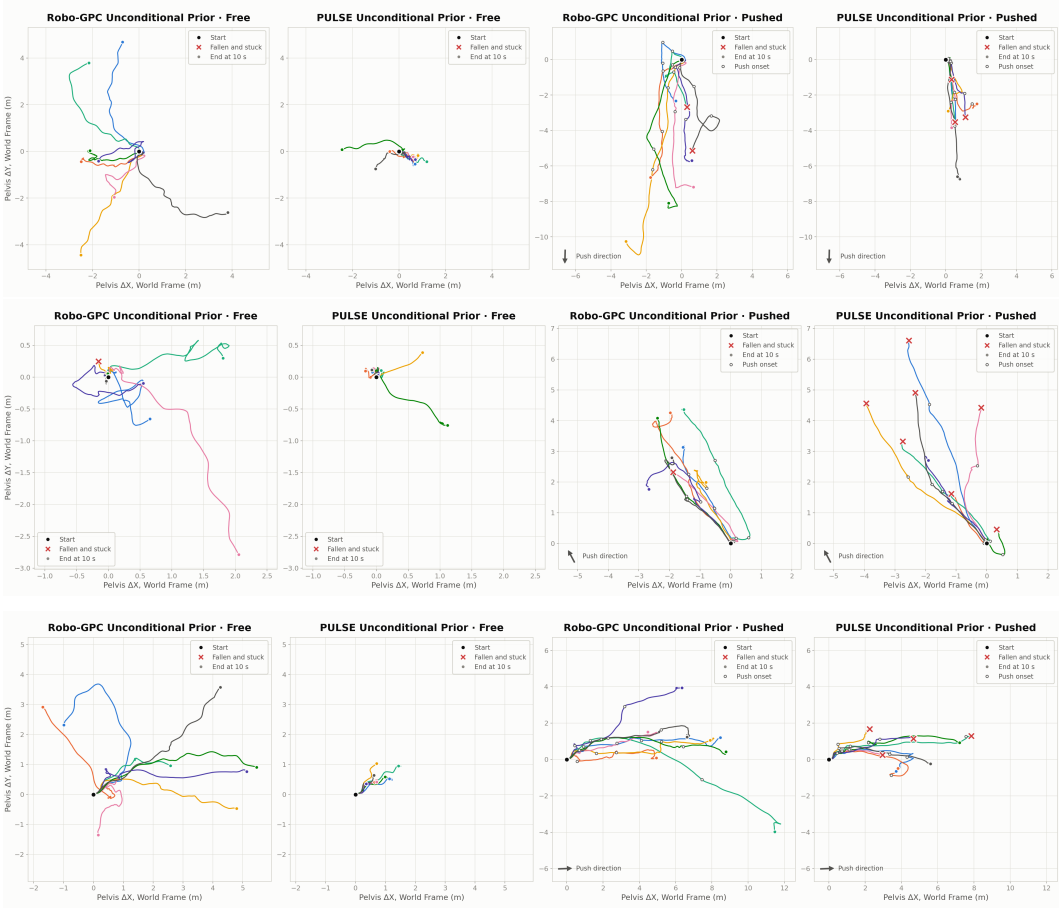


Figure 9: Unconditional pelvis trajectories from three initial states. Within each row, the four panels show, from left to right, Robo-GPC without a push, PULSE without a push, Robo-GPC with a push, and PULSE with a push. Each colored curve corresponds to one of eight sampled rollouts over a 10-s horizon. The filled black marker indicates the initial position, gray markers denote the final positions at 10 s, open markers indicate the onset of the push, and red crosses mark rollouts in which the robot falls and remains stuck. Arrows indicate the applied push direction. The axes show world-frame pelvis displacement relative to the initial position..