SMP: Reusable Score-Matching Motion Priors for Physics-Based Character Control

YUXUAN MU*, Simon Fraser University, Canada
ZIYU ZHANG*, Simon Fraser University, Canada
YI SHI*, Simon Fraser University, Canada
DUN YANG, Simon Fraser University, Canada
MINAMI MATSUMOTO, Sony Interactive Entertainment, Japan
KOTARO IMAMURA, Sony Interactive Entertainment, Japan
GUY TEVET, Stanford University, USA
CHUAN GUO, Snap Inc., USA
MICHAEL TAYLOR, Sony Interactive Entertainment, USA
CHANG SHU, National Research Council Canada, Canada
PENGCHENG XI, National Research Council Canada, Canada
XUE BIN PENG, Simon Fraser University, Canada and NVIDIA, Canada

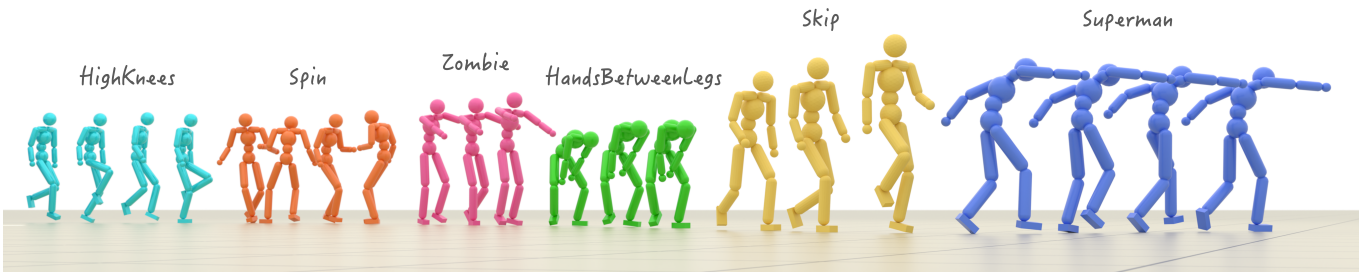


Fig. 1. Our framework constructs reusable and modular motion priors for training motion controllers. A general motion prior can be trained on a large dataset spanning 100 distinct styles. Once trained, the prior can be repurposed into 100 style-specific priors without requiring access to the original dataset or further model updates. These style priors serve as stationary reward models and can be reused to train control policies for diverse tasks with natural stylistic behaviors.

Data-driven motion priors that can guide agents toward producing naturalistic behaviors play a pivotal role in creating life-like virtual characters. Adversarial imitation learning has been a highly effective method for learning motion priors from reference motion data. However, adversarial priors, with few exceptions, need to be retrained for each new controller, thereby limiting their reusability and necessitating the retention of the reference motion data when applied to downstream tasks. In this work, we present Score-Matching Motion Priors (SMP), which leverages pre-trained motion diffusion models and score distillation sampling (SDS) to create reusable task-agnostic motion priors. SMPs can be pre-trained on a motion dataset, independent of any control policy or task. Once trained, SMPs can be kept frozen and reused as general-purpose reward functions to train new policies to produce naturalistic behaviors for downstream tasks. We show that a general motion prior trained on large-scale datasets can be repurposed into a variety of style-specific priors. Furthermore, SMP can compose different styles to synthesize new styles not present in the original dataset. Our method can create reusable and modular motion priors that produce high-quality motions comparable to state-of-the-art adversarial imitation learning methods. In our experiments, we demonstrate the effectiveness of SMP across a diverse suite of control tasks with physically simulated humanoid characters. Video available at youtu.be/jBA2tWk6vzU.

Additional Key Words and Phrases: character animation, score distillation sampling, diffusion model, reinforcement learning

1 INTRODUCTION

Creating virtual characters that move with natural and life-like behaviors is fundamental to immersive digital experiences in animation, films, games, and virtual reality (VR) applications. While motion capture and procedural animation techniques can produce high-quality movements, the challenge lies in developing controller that enable physically simulated characters to exhibit similarly natural behaviors dynamically, in response to diverse tasks and environmental contexts. Traditional physics-based controllers trained

*Joint First Authors.

Authors' addresses: Yuxuan Mu, yma101@sfu.ca, Simon Fraser University, Canada; Ziyu Zhang, zza333@sfu.ca, Simon Fraser University, Canada; Yi Shi, ysa273@sfu.ca, Simon Fraser University, Canada; Dun Yang, dya66@sfu.ca, Simon Fraser University, Canada; Minami Matsumoto, Minami.X.Matsumoto@sony.com, Sony Interactive Entertainment, Japan; Kotaro Imamura, Kotaro.Imamura@sony.com, Sony Interactive Entertainment, Japan; Guy Tevet, guy.tvt@gmail.com, Stanford University, USA; Chuan Guo, guochuan5513@gmail.com, Snap Inc., USA; Michael Taylor, Mike.Taylor@sony.com, Sony Interactive Entertainment, USA; Chang Shu, Chang.Shu@nrc-cnrc.gc.ca, National Research Council Canada, Canada; Pengcheng Xi, Pengcheng.Xi@nrc-cnrc.gc.ca, National Research Council Canada, Canada; Xue Bin Peng, xbpeng@sfu.ca, Simon Fraser University, Canada and NVIDIA, Canada.

without reference motion data often result in visually unnatural movements, lacking the subtle qualities of life-like motions [Coros et al. 2010; Hodgins et al. 1995; Yin et al. 2007]. Tracking-based methods improve realism by following reference motion clips, but they typically require controllers to rigidly mimic target motions frame-by-frame [Liu and Hodgins 2017; Peng et al. 2018a; Won et al. 2017], limiting their flexibility to deviate from the reference and adapt behaviors to perform new tasks. Alternatively, distribution-matching methods, such as adversarial imitation learning [Ho and Ermon 2016; Peng et al. 2021], provide a more versatile approach for imitating motion data. Adversarial methods can learn flexible motion priors from motion dataset, which can act as task-agnostic measures of motion naturalness, allowing policies to produce behaviors that resemble natural motions across different tasks. However, adversarial methods typically require a prior (*i.e.* discriminator) to be trained jointly with a given policy, which limits the reusability and modularity of the learned priors. For each new policy, the prior must be continuously trained with data from the policy and data from the original reference dataset. As a result, the original dataset must be retained for perpetuity.

We contend that for many practical control applications, an ideal motion prior should be **modular** and **reusable**:

- **Modular**: The prior should function as an independent motion-quality objective that can guide policy training without requiring access to the original reference dataset.
- **Reusable**: Once constructed, the same prior should be applicable across diverse tasks and multiple policies without requiring further training.

To develop motion priors that satisfy these criteria, we propose Score-Matching Motion Priors (SMP): a method for constructing reusable, modular motion priors that can be utilized to train control policies to produce naturalistic behaviors across diverse tasks. Given an unstructured motion dataset, we first train a motion diffusion model to capture the underlying data distribution independent of any task or control policy. Once trained, the diffusion model is kept frozen and repurposed as a prior via score distillation sampling (SDS) [Poole et al. 2022], providing a robust similarity measure between simulated motions and the behaviors in the reference dataset. By reusing the learned SMP as a general style reward, our system enables a prior to be used to train new policies to perform a diverse suite of tasks with natural life-like behaviors.

The central contribution of this work is a method for constructing *modular* and *reusable* motion priors for physics-based character animation by leveraging diffusion models as reward models in reinforcement learning. Our framework produces high-quality motions comparable to state-of-the-art adversarial imitation learning methods, while substantially improving modularity and reusability. SMP enables our system to completely discard the reference dataset during policy training. We show that a score-matching motion prior can be constructed from a task-agnostic diffusion model pretrained on large-scale motion datasets. This general-purpose motion prior can then be repurposed into style-specific priors through prompting and guidance. Furthermore, we show that priors for different styles can be composed to create new behavioral styles that are not present in the original dataset. Code is available in MimicKit.

2 RELATED WORK

Developing embodied agents that can act and react with life-like motions is essential for creating immersive experiences in games and virtual reality applications. This capability is also of vital importance in robotics, where naturalistic behaviors can improve safety, energy efficiency [Escontrela et al. 2022], and the learning of general-purposes skills from human data [Grauman et al. 2022]. Given the expressive and nuanced nature of human motion, data-driven approaches have emerged as an effective paradigm for producing realistic, life-like behaviors by learning from reference motion data. *Kinematics-based methods* typically train models, such as autoregressive models [Holden et al. 2017, 2016; Zhang et al. 2018], variational autoencoders (VAEs) [Ling et al. 2020; Rempe et al. 2021; Starke et al. 2024], or diffusion models [Shi et al. 2024; Tevet et al. 2023; Zhang et al. 2022], on large motion datasets to synthesize plausible character animations. However, most of these methods do not explicitly enforce physical constraints, often leading to physically implausible behaviors, especially for new scenarios and tasks.

Physics-Based Methods. In contrast, *physics-based methods* aim to create control policies that generate motion within simulated environments, governed by dynamic equations and realistic physical laws [Raibert and Hodgins 1991; Wampler et al. 2014]. These controllers are often constructed via trajectory optimization techniques or reinforcement learning (RL) [Peng et al. 2017; Wang et al. 2009]. Previous data-free approaches based on heuristics, such as SIMBICON [Yin et al. 2007], can achieve mechanically functional behaviors, but often produce unnatural motions. Manually designing heuristics that capture the expressiveness of real human movement remains a significant challenge [Coros et al. 2010; Faloutsos et al. 2001; Hodgins et al. 1995; Witkin and Kass 1988]. Instead of relying on hand-crafted heuristics, data-driven approaches based on model predictive control (MPC) or trajectory optimization have shown promise in emulating naturalistic human motions [Lee et al. 2010; Liu et al. 2010; Muico et al. 2009; Sharon and van de Panne 2005]. More recently, reinforcement learning-based motion tracking methods have enabled controllers to effectively imitate a wide range of reference motion clips, resulting in more life-like and agile behaviors [Liu and Hodgins 2017; Peng et al. 2018a; Won et al. 2017]. However, tracking-based methods limit a controller to closely following a given motion clip, which can impede the controller's ability to generalize to new tasks, particularly when task objectives are not closely aligned with the reference motions. To address this limitation, many systems incorporate task-specific motion planners [Bergamin et al. 2019; Liu et al. 2012; Park et al. 2019], or high-level policies [Yao et al. 2022, 2024; Zhu et al. 2023], which select appropriate reference motions or latent-space skills for the controller to perform in order to complete a given task.

Distribution matching offers an alternative to motion tracking by training controllers to imitate the broader behavioral distribution of a motion dataset rather than tracking reference motions frame-by-frame. Generative Adversarial Imitation Learning (GAIL) approximates this objective using a learned discriminator to distinguish agent's motions from dataset motions [Ho and Ermon 2016]. Adversarial Motion Priors (AMP) extend this idea to model flexible style objectives, enabling controllers to produce life-like behaviors

for novel tasks that are not observed in the original dataset [Peng et al. 2021]. However, adversarial priors must be trained jointly with a specific policy for each new task, often requiring retraining and persistent access to the dataset. In contrast, our method achieves comparable performance to state-of-the-art adversarial imitation learning approaches by introducing a reusable and modular score-matching motion prior, which can even be shaped into stylistic priors beyond those contained in the original dataset.

Diffusion Models for Control. Diffusion models' state-of-the-art performance across a wide range of domain has spurred growing interest in leveraging diffusion models for control. Given their ability to generate high-fidelity motion, a straightforward application is to use task-oriented diffusion models as motion planners [Ren et al. 2023; Serifi et al. 2024; Tevet et al. 2024; Xu et al. 2025]. These methods leverage auto-regressive motion diffusion models to predict target future trajectories, which are then executed by a low-level tracking controller. In addition to their use as motion planners, diffuse models have also been used to directly model controllers, enabling controllers to produce flexible multi-modal behaviors for tasks such as manipulation [Chi et al. 2023; Janner et al. 2022], and character control [Huang et al. 2024a, 2025; Truong et al. 2024; Wu et al. 2025b]. Unlike these prior methods that focus on incorporating diffusion models as components of a controller, our work aims to repurpose *pretrained* diffusion models as task-agnostic behavioral priors within an optimization objective. Our method utilizes diffusion models as a reward function that can be integrated into general reinforcement learning frameworks to guide policy learning via score distillation sampling.

Score Distillation Sampling. Pretrained diffusion models are not only capable of generating samples via a denoising process, but can also serve as optimization objectives through score distillation sampling (SDS) [Poole et al. 2022]. SDS has enabled pretrained image and video diffusion models to be adapted for a wide range of downstream generative modeling tasks [Jiang et al. 2024; Wang et al. 2023; Yin et al. 2024]. Inspired by the success of SDS in the vision domain, recent works have also explored incorporating diffusion-based priors into reinforcement learning frameworks for control. A series of methods replace the discriminator in a GAIL-style setup with a diffusion model [Huang et al. 2024b; Lai et al. 2024; Pang et al. 2025; Wang et al. 2024], leveraging its expressive capabilities to differentiate between real and fake trajectories. However, these methods still rely on adversarial training and typically require continual updates to the diffusion model during policy optimization. Beyond adversarial frameworks, other systems have attempted to apply SDS-like techniques directly to policy learning. For example, Luo et al. [2024] guide policy training using pretrained image or video diffusion models conditioned on text prompts. While these methods can produce behaviors that align with textual instructions, the resulting motions often appear unnatural. The work that is most reminiscent to ours is SMILING [Wu et al. 2025a], which train controllers using a score-matching objective similar to variational score distillation (VSD) [Wang et al. 2023]. However, unlike SMILING, which requires training task-specific diffusion models, our method trains a task-agnostic motion prior from unstructured motion data. This prior can then be reused to train diverse control policies by combining it with separate task objectives in a goal-conditioned reinforcement learning framework. We further introduce several key design decisions that enable high-quality naturalistic motions across a diverse repertoire of tasks.

3 BACKGROUND

Our system leverages reinforcement learning to train physics-based control policies and diffusion models to construct score-matching motion priors used as training objectives. In this section, we review the core concepts of reinforcement learning and diffusion models that underpin our method.

3.1 Reinforcement Learning

Our physics-based controllers are trained through a reinforcement learning (RL) framework, where an agent interacts with an environment according to a policy π in order to optimize a given objective $J(\pi)$ [Sutton et al. 1998]. At each time step t, the agent observes the current state $\mathbf{s}_t$, then samples and executes an action according to the policy $\mathbf{a}_t \sim \pi(\mathbf{a}_t|\mathbf{s}_t)$. The environment then transitions to a new state according to the dynamics $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, and the agent receives a scalar reward $r_t = r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$. The objective is to learn a policy that maximizes the expected discounted return:

$$J(\pi) = \mathbb{E}_{\tau\sim p(\tau|\pi)}\left[\sum_{t=0}^{T-1}\gamma^t r_t\right], \tag{1}$$

where $\tau = \{\mathbf{s}_0, \mathbf{a}_0, r_0, \mathbf{s}_1, \ldots, \mathbf{s}_{T-1}, \mathbf{a}_{T-1}, r_{T-1}, \mathbf{s}_T\}$ is a trajectory produced by the policy π, and $\gamma \in [0, 1]$ is a discount factor. The reward function serves as a flexible interface for specifying task objectives. However, crafting effective rewards that consistently induce naturalistic behaviors can be challenging and often requires extensive manual tuning.

3.2 Diffusion Models and Score Distillation Sampling

In score-based generative modeling methods [Song and Ermon 2019; Song et al. 2021], a neural network $f : \mathbb{R}^D \to \mathbb{R}^D$ is trained to estimate the *score* of a point $\mathbf{x}$ under a given distribution $p(\mathbf{x})$, which is defined as the gradient of the log-density $\nabla_{\mathbf{x}} \log p(\mathbf{x})$. However, the predicted scores may be unreliable, as the available training data typically do not cover the entire space $\mathbb{R}^D$. In regions with low data density, score estimates tend to be inaccurate. To address this issue, Vincent [2011] and Song et al. [2021] propose adding noise to the data. When the noise level is sufficiently large, the perturbed data distribution covers the entire space, enabling the training of more robust and scalable score estimators. This principle forms the foundation of our approach. Score-based generative modeling with multi-level noise perturbation allows the construction of reusable motion priors from relatively limited data that lies in a low-dimensional manifold.

Diffusion Models. Diffusion models approximate a data distribution by progressively corrupting and subsequently denoising samples through a sequence of transformations [Ho et al. 2020]. Given a clean sample from the data distribution, the forward process iteratively adds Gaussian noise over N steps to form a Markov chain

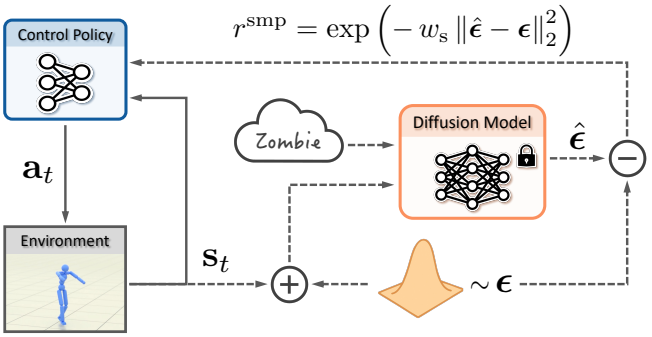


Fig. 2. Schematic overview of the system. The dashed arrows indicate components used only during policy training. A pretrained motion diffusion model serves as a reusable reward model for motion naturalness via score distillation sampling. The model can be style-conditioned, enabling the policy to learn specific skills or styles without retraining or continuous access to the original motion data.

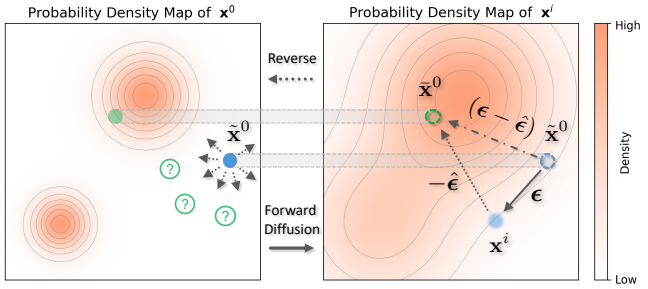


Fig. 3. A qualitative illustration of the probability density maps of $\mathbf{x}^0$ and $\mathbf{x}^i$. When the agent's motion $\tilde{\mathbf{x}}$ deviates significantly from the reference distribution, the gradient of $p(\mathbf{x}^0)$ cannot be reliably approximated in low-density regions. The forward diffusion process in Equation (3) maps $\tilde{\mathbf{x}}$ to a diffused sample $\mathbf{x}^i$, where the density $p(\mathbf{x}^i)$ is higher and the score estimate is more reliable. This estimated score can then be used to obtain a pseudo target $\hat{\mathbf{x}}^0$ from the reference distribution via a reverse process. The residual between the predicted noise $\hat{\epsilon}$ and the added noise ϵ provides the correction that aligns the agent's motion $\tilde{\mathbf{x}}$ with the reference distribution.

$\{\mathbf{x}^i\}_{i=0}^N$, defined as:

$$q\left(\mathbf{x}^i \mid \mathbf{x}^{i-1}\right) = \mathcal{N}\left(\mathbf{x}^i; \sqrt{1-\beta_i}\mathbf{x}^{i-1}, \beta_i \mathbf{I}\right), \tag{2}$$

where i indicates the noise level, $\{\beta_i\}_{i=1}^N$ is a predefined noise schedule. Since additive *i.i.d.* Gaussian noise is used in the diffusion process, $\mathbf{x}^i$ can be conveniently sampled from $\mathbf{x}^0$ directly via:

$$\mathbf{x}^i = \sqrt{\bar{\alpha}_i}\mathbf{x}^0 + \sqrt{1-\bar{\alpha}_i}\epsilon, \tag{3}$$

without performing the full iterative diffusion process. Here, $\bar{\alpha}_i = \prod_{j=1}^{i}(1-\beta_j)$ and $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. This forward diffusion process resembles the multi-level noise perturbation used in score-based generative models [Song et al. 2021]. To sample from the learned distribution $p(\mathbf{x})$, a denoising network f is typically trained using the *simple* DDPM objective [Ho et al. 2020]:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{i,\mathbf{x}^0,\epsilon}\left[\left\|\epsilon - f\left(\mathbf{x}^i\right)\right\|_2^2\right], \tag{4}$$

where $\mathbf{x}^i$ is generated through the forward diffusion process detailed in Equation (3).

Score Distillation Sampling (SDS). Once a diffusion model f is trained, samples can be generated by applying the reverse diffusion process or through score distillation sampling (SDS) [Poole et al. 2022]. SDS minimizes the KL divergence between the distribution of diffused samples derived from the forward diffusion process $q(\mathbf{x}^i \mid \mathbf{x}^0)$ and the reference data distribution learned by the pretrained diffusion model $f(\mathbf{x}^i)$. The gradient of the KL divergence can be estimated using the difference between the score from the forward diffusion process and the prediction from the diffusion model:

$$\nabla\mathcal{L}_{\text{SDS}} = \mathbb{E}_{i,\epsilon}\left[w(i)\left(f\left(\mathbf{x}^i\right) - \epsilon\right)\nabla\mathbf{x}\right], \tag{5}$$

where $\mathbf{x}$ is the sample being optimized, and $w(i)$ are coefficients determined by the diffusion noise schedule. Previous work has shown that the weighting function $w(i)$ has limited impact and can be substituted with uniform weighting without negatively affecting performance [Guo et al. 2023]. The SDS gradient can also be derived from the diffusion loss in Equation (4), omitting the Jacobian with respect to the score estimator f. The SDS loss can be simplified as

$$\mathcal{L}_{\text{SDS}} = \|\hat{\epsilon} - \epsilon\|_2^2, \tag{6}$$

where $\hat{\epsilon}$ denotes the noise predicted by the denoising network f. The optimum of the SDS objective corresponds to samples that minimize the diffusion loss, thereby resembling the characteristics of the dataset on which the diffusion model was originally trained.

4 OVERVIEW

In this work, we introduce the *Score-Matching Motion Prior (SMP)*, a *reusable*, *modular* imitation objective derived from a pretrained diffusion model via score distillation sampling (SDS). The pretrained diffusion model is used to estimate the gradient of the log-likelihood (*i.e.*, the *score*) of the reference distribution, which evaluates the similarity between an agent's motions and motions in a reference dataset. The robustness of score-based generative modeling enables a pre-trained diffusion model to provide reliable guidance even for motions that are different from those in the original motion dataset. The general style imitation objective provided by SMP can be combined with task-specific objectives to train control policies that can accomplish diverse tasks using natural life-like behaviors.

Figure 2 illustrates an overview of our framework. Unlike prior approaches that use diffusion models as *planners* [Ren et al. 2023; Serifi et al. 2024; Tevet et al. 2024], SMP repurposes a pretrained diffusion model as a general *reward model* for evaluating motion naturalness to guide training of a control policy. The task-agnostic diffusion model f is trained solely on reference motion data, independently of any control policy or task. Once trained, it is frozen and utilized as a reward function via score distillation sampling (SDS). When training a policy, the SDS objective encourages the policy to minimize the discrepancy between the noise ϵ added to the simulated motion, and the noise $\hat{\epsilon}$ estimated by the pretrained diffusion model. The SDS error is minimized when the agent's motions closely aligns with the reference distribution. Similar to other distribution-matching

objectives, SMP does not require the simulated character to exactly replicate specific reference motions. Instead, it encourages behaviors that capture the general characteristics of the reference data, enabling smooth transitions and adaptation to tasks that may require skills not explicitly present in the dataset. Furthermore, SMP can be trained on large and diverse datasets by employing a conditional diffusion model. This enables policies to acquire different stylistic behaviors by conditioning the pretrained diffusion model on style labels, without the need to retain the original motion dataset.

5 SCORE-MATCHING MOTION PRIORS

A score-matching motion prior is modeled as a diffusion model, which is trained to predict the *score* of noisy input motions. The predicted score can be interpreted as a correction applied to a noisy sample to denoise back to a clean sample from the training data distribution. This enables SMP to construct motion imitation objectives directly from the pretrained motion diffusion model.

Given a motion dataset, we first train a motion diffusion model to generate motion clips consisting of H consecutive frames $\mathbf{x} := (\mathbf{s}_{t-H+2}, \ldots, \mathbf{s}_{t+1})$. During policy training, each simulated motion clip produced by the agent is diffused with Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ to a diffusion timestep i through the forward diffusion process in Equation (3), resulting in a noisy sample $\mathbf{x}^i$. The pretrained diffusion model $\hat{\boldsymbol{\epsilon}} = f(\mathbf{x}^i)$ then predicts the score $\hat{\boldsymbol{\epsilon}}$ at $\mathbf{x}^i$, which provides a denoising direction back toward the reference motion distribution. The correction that should be applied to align the agent's motion with the reference distribution is therefore given by the noise residual $(\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}})$, as illustrated in Figure 3. The SMP reward used for motion imitation is then defined as

$$r^{\text{smp}} = \exp\left(-w_s \left\|\hat{\boldsymbol{\epsilon}} - \boldsymbol{\epsilon}\right\|_2^2\right), \tag{7}$$

which is maximized when the simulated motion aligns with the reference distribution. Following standard RL reward design [Peng et al. 2018a], we apply an exponential transformation to the SDS loss to normalize the reward between $[0, 1]$. While this deviates from the original SDS formulation, we find that this normalization produces empirical improvements when training RL policies.

Previous SDS-based methods have primarily shown promise in the 3D generation, but often produce low-quality, "blurry" results [Liang et al. 2024]. Earlier efforts to use SDS as a reinforcement learning objective have also struggled to match the performance of adversarial methods for training control policies [Luo et al. 2024; Wu et al. 2025a]. In the following sections, we introduce several key design decisions that improve stability for policy training using SMP, enabling policies to learn high-quality naturalistic behaviors.

5.1 Ensemble Score-Matching

The SDS objective can be highly sensitive to the choice of diffusion timestep i, as different noise scales provide disparate forms of guidance [Lin et al. 2023]. At higher noise levels, diffused samples are heavily corrupted and dominated by Gaussian noise, which matches the training settings of the diffusion model and leads to more reliable score predictions due to less susceptibility to out-of-distribution samples. Therefore, evaluating the objective at higher diffusion timesteps i can be more instructive when the agent's motions deviate substantially from the reference data distribution. However, excessive noise also removes much of the information present in the agent behaviors, which may prevent the objective from correcting more minute errors in the agent's motions. In such cases, the guidance signal may encourage the policy to produce generic or averaged behaviors, as the model no longer retains sufficient information to steer toward more realistic and detailed motions. Therefore, when the difference between the agent's motions and motions in the dataset is relatively small, diffusing the sample to lower noise levels can better preserve information of the agent's motions and apply finer-grained corrections.

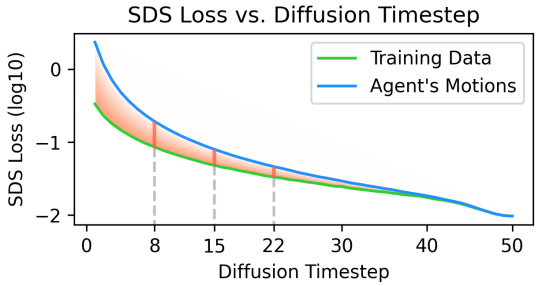


Fig. 4. An example of the SDS loss across diffusion noise levels. The y-axis shows $\log_{10}$ values for visualization. Larger diffusion timesteps correspond to higher noise levels applied to the sample. Statistics are averaged across 1024 samples for each diffusion timestep. The orange region highlights the difference between the SDS scores from the agent's motions and the motion data. The scale of the SDS loss can vary significantly across different diffusion noise levels.

In prior SDS-based methods in vision domains [Guo et al. 2023; Poole et al. 2022], it is common to sample the diffusion noise level uniformly $i \sim \mathcal{U}(1, N)$. However, in reinforcement learning, this stochasticity can introduce undesirable variance into the reward signal, leading to less reliable value and advantage estimation. For example, samples diffused with high noise levels are nearly pure Gaussian noise, allowing the diffusion model to easily predict the applied noise, leading to small SDS errors and limited information about differences between the agent's motions and the reference distribution, as illustrated in Figure 4. Whereas samples with lower noise levels generally produce larger SDS errors. Due to this noise-dependent variation in the SDS loss, computing the SMP reward with randomly sampled timesteps can result in a highly stochastic and unreliable measure of the similarity between the agent's motions and the reference data distribution.

To reduce the variance of the SMP reward function, instead of randomly sampling a single timestep per reward calculation, we propose *ensemble score-matching* (ESM), which constructs a lower variance reward function by aggregating multiple SDS evaluations over a fixed set of diffusion timesteps $i \in \mathbb{K}$. The resulting SMP reward is given by:

$$r^{\text{smp}} = \exp\left(-\frac{w_s}{|\mathbb{K}|} \sum_{i \in \mathbb{K}} \left\|\hat{\boldsymbol{\epsilon}}_i - \boldsymbol{\epsilon}_i\right\|_2^2\right), \tag{8}$$

where $\hat{\boldsymbol{\epsilon}}_i = f\left(\sqrt{\bar{\alpha}_i}\,\tilde{\mathbf{x}}^0 + \sqrt{1-\bar{\alpha}_i}\,\boldsymbol{\epsilon}_i\right)$, $\tilde{\mathbf{x}}$ denotes the simulated character's motion, and $\mathbb{K}$ is a predefined set of diffusion timesteps. When the diffusion model's predictions are sufficiently reliable, *i.e.*, for samples that are less out-of-distribution (OOD), the SDS loss computed at lower noise levels responds more sensitively to discrepancies between the agent's motion and the reference training distribution. In our experiments, we found that SDS error computed at low diffusion timesteps is highly sensitive to jittery artifacts, whereas SDS error computed at medium timesteps is more robust and provides better guidance for correcting behavioral artifacts. Moreover, evaluating a large number of diffusion steps incurs significant computational cost, and adjacent timesteps tend to provide redundant guidance at similar granularity levels. We therefore found that evaluating the SMP objective using a small set of representative diffusion timesteps is sufficient for effective performance. In all experiments, we compute the SDS loss at $\mathbb{K} = \{22, 15, 8\}$, which provides an effective balance across different noise scales.

In addition, we apply adaptive normalization using the running mean μ_i of the SDS error at each diffusion timestep i to mitigate the varying loss scales at different noise levels. This adaptive normalization also reduces SMP's sensitivity to variations across different pre-trained diffusion models and behavioral styles, thereby reducing the need for manual parameter tuning for the SMP reward function.

ALGORITHM 1: Policy Training with SMP

Input (optional): style label c
$f \leftarrow$ load pretrained diffusion model
$\pi \leftarrow$ initialize policy
$V \leftarrow$ initialize value function
$\mathcal{B} \leftarrow \emptyset$ initialize reply buffer

while not done **do**
  for trajectory $j = 1, ..., m$ **do**
    $\tau^j \leftarrow \{(\mathbf{s}_t, \tilde{\mathbf{x}}_t, \mathbf{a}_t, r_t^g)_{t=0}^{T-1}, \mathbf{s}_T^g, \tilde{\mathbf{x}}_T\}$ collect trajectory with π
    for $t = 0, ..., T-1$ **do**
      for diffusion timestep $i \in \mathbb{K}$ **do**
        $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, I)$
        $\hat{\boldsymbol{\epsilon}}_i \leftarrow f\left(\sqrt{\bar{\alpha}_i}\,\tilde{\mathbf{x}}_{t+1} + \sqrt{1-\bar{\alpha}_i}\,\boldsymbol{\epsilon}_i,\ c\right)$
      end for
      $r_t^{\text{smp}} \leftarrow$ compute prior reward via eq. (8) using $\{\boldsymbol{\epsilon}, \hat{\boldsymbol{\epsilon}}\}_i^{\mathbb{K}}$
      $r_t \leftarrow w^{\text{prior}} r_t^{\text{smp}} + w^g r_t^g$
      record r_t in τ^j
    end for
    store τ^j in $\mathcal{B}$
  end for

  update V and π using data from trajectories $\{\tau^j\}_{j=1}^m$
end while

5.2 Generative State Initialization

Reference state initialization (RSI), where the simulated character is initialized to states randomly sampled from a reference motion dataset, has been shown to be a vital technique for improving exploration in motion imitation methods [Peng et al. 2018a]. However, RSI requires access to the motion dataset to sample initial states from during policy training. To remove this reliance on the original dataset, SMP can instead leverage the generative prior to *generate* initial states when training the policy. When using SMP to train a policy, initial states are generated by sampling from the diffusion model used as the pretrained motion prior. SMP therefore serves dual purposes in our framework, as both a reward function and an initial state distribution when training new policies. This *generative state initialization* (GSI) method alleviates the need to retain the original motion dataset once the SMP has been trained by leveraging the generative capabilities of diffusion models to produce diverse, high-quality initial states.

6 MODEL REPRESENTATION

Our framework utilizes a pretrained diffusion model as a reusable, modular motion prior for imitation learning. The task-agnostic motion diffusion model serves as a reward function that evaluates the naturalness of the agent's motion within a reinforcement learning framework. This allows for the training of control policies that accomplish diverse tasks while exhibiting naturalistic behaviors that resemble those in the original motion dataset. In this section, we detail key design decisions of the learning framework.

6.1 Motion Representation

Designing an appropriate motion representation is critical both for training the diffusion model to capture the dataset distribution, and for repurposing it as a motion prior for policy training. The representation should contain sufficient information to reconstruct motion while remaining easy to extract from both kinematic motion clips and simulator state observations. Following the design of prior works in motion diffusion and AMP [Peng et al. 2021; Tevet et al. 2023; Zhang et al. 2022], our motion features include:

- Root linear and angular velocities, represented in the character's local coordinate frame at the last timestep in a motion segment.
- Local joint rotations.
- 3D positions of end-effectors (*e.g.*, hands and feet), represented in the character's local coordinate frame.

The character's local coordinate frame is defined with the origin at the root (*i.e.*, pelvis), the x-axis aligned with the root link's facing direction, and the y-axis aligned with the global up vector. Joint rotations are represented using a 6D representation for spherical joints [Zhou et al. 2019].

6.2 Diffusion Model Implementation

The motion diffusion model is implemented as a transformer encoder, using adaptive normalization to inject noise-level conditions (and style conditions, when available). Unless otherwise specified, we use a window of $H = 10$ frames. We find that a simple two-layer transformer encoder with only 3M parameters is sufficient to capture the distribution of large-scale motion datasets, such as 100STYLE [Mason et al. 2022] with over 20 hours of stylized motions. The number of diffusion timesteps is set to $N = 50$, and the model is trained to predict $\boldsymbol{\epsilon}$. Following standard practice for training diffusion models, we apply exponential moving average (EMA) on the model parameters during training [Dhariwal and Nichol 2021;

Karras et al. 2024]. The model is trained for 400k–800k iterations, depending on the dataset, with convergence typically achieved within five hours on a single RTX 4090 GPU.

6.3 Model Implementation

Following prior work, the control policy π is modeled using a multi-layer perceptron (MLP) that maps an input state $\mathbf{s}_t$ and goal $\mathbf{g}$ to a Gaussian distribution over the action space $\mathbf{a}_t \in \mathcal{A}$ [Peng et al. 2018a]. The value function $V(\mathbf{s}_t, \mathbf{g})$ is modeled by a separate MLP network with a similar architecture to the policy.

States and Actions. The state $\mathbf{s}_t$ is represented using features similar to those in Peng et al. [2021], such as body link positions, link rotations encoded in a 6D representation, and linear and angular velocities of each link. All features are calculated in the character's local coordinate frame. Since our policies are not trained to track specific reference motions, no phase variable or target reference state information is provided in the state. The action $\mathbf{a}_t$ specifies target joint positions for PD controllers at each joint. For spherical joints, each target rotation is parameterized using a 3D exponential map $q \in \mathbb{R}^3$ [Grassia 1998].

Training. The policy is trained using proximal policy optimization (PPO) [Schulman et al. 2017]. At each timestep t, the agent queries the motion prior (see Algorithm 1) to obtain a prior reward r_t^{smp}, and may also receive a task reward r_t^g from the environment. These are combined linearly to form the composite reward at that timestep

$$r_t = w^{\text{prior}} r_t^{\text{smp}} + w^g r_t^g, \tag{9}$$

following Peng et al. [2021]. After collecting a batch of trajectories, mini-batches are sampled from the buffer to update both the policy and the value function. The policy is updated with PPO using advantages estimated via GAE(λ) [Schulman et al. 2015], while the value function is trained with targets computed via TD(λ) [Sutton et al. 1998]. The pretrained diffusion model is kept fixed during the policy training process, and does not need to be updated.

Training a policy with SMP on a single motion clip (e.g., *spinkick*) takes approximately 30 minutes on a single RTX 4090 GPU. Training a HighKnees policy for the target location task takes 11.5 hours for 600M samples, compared to 6.2 hours for AMP. SMP uses a larger prior model that captures 100 styles, whereas AMP uses a smaller style-specific discriminator. Runtime performance is identical to AMP, since only the policy is executed.

7 TASKS

We evaluate the effectiveness of SMP across seven motion control tasks, showcasing its ability to train policies that can perform various control tasks using diverse motion styles. Below, we summarize each task and the corresponding goal observation $\mathbf{g}$, which provides the agent with task-relevant information from the environment. More comprehensive details, including task reward functions and specific hyperparameter settings are provided in Appendix D.

Target Speed. This task requires the character to move at a target speed. The goal for the policy is specified as $\mathbf{g}_t = v_t^*$. The target speed v_t^* is randomly sampled from $[1.2, 6.8]$m/s. To focus on speed tracking and gait transitions induced by different speeds, the target movement direction is kept fixed.

Steering. The task requires the character to face a specified 2D heading direction $\mathbf{h}_t^*$ while simultaneously traveling at a target speed v_t^* along a target horizontal direction $\mathbf{d}_t^*$. The steering policy receives the goal information as $\mathbf{g}_t = (\mathbf{d}_t^*, v_t^*, \mathbf{h}_t^*)$.

Target Location. In this task, the character is instructed to reach a 2D target location specified on the floor plane. The agent perceives the goal via $\mathbf{g}_t = \mathbf{p}_t^*$, where the target location $\mathbf{p}_t^*$ is represented in the character's local coordinate frame.

Dodgeball. In this task, a ball is launched toward the character from a random position up to 10m away, with a launch speed sampled from $[20, 25]$m/s. This gives the agent less than 0.5s to react and dodge. If the character is hit, the episode terminates early and the agent receives zero reward for all remaining timesteps as a penalty. The agent observes the ball state through goal observations $\mathbf{g}_t = \left(\mathbf{p}_t^{\text{ball}}, \dot{\mathbf{p}}_t^{\text{ball}}\right)$, where $\mathbf{p}_t^{\text{ball}}$ and $\dot{\mathbf{p}}_t^{\text{ball}}$ denote the ball's position and velocity represented in the character's local coordinate frame.

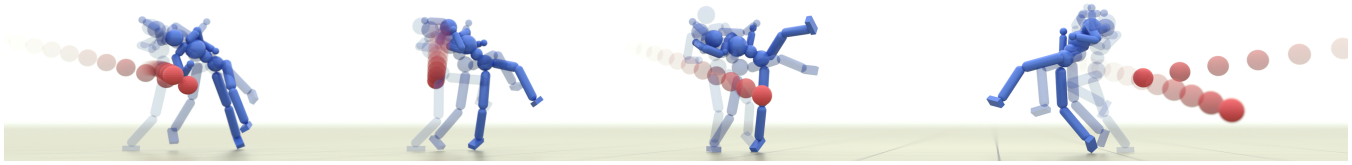
Object Carry. In addition to training priors for human motion, SMP can also be used to train priors that jointly model the interactions between the character and objects. First, we train an SMP on a dataset of human-object carrying motions. This prior is then used to train policies for an object carrying task, where the character is required to carry a box from the ground to a randomly placed target location. The goal $\mathbf{g}_t = \mathbf{p}_t^{\text{box}^*}$ records the target box position. The state $\mathbf{s}_t$ is augmented with additional features that describe the state of the box, including the position $\mathbf{p}_t^{\text{box}}$ and the orientation $\mathbf{q}_t^{\text{box}}$. All features are represented in the character's local coordinate frame.

Stair Traversal. To further evaluate SMP's ability for modeling character-object interactions, we apply SMP to train policies for a stair traversal task. The task requires the character to traverse a short staircase by walking up and then down a series of steps while following a target velocity $\mathbf{v}^{\text{tar}}$. At each timestep, the goal $\mathbf{g}_t = \hat{\mathbf{v}}_t^{\text{tar}}$ records the target velocity expressed in the character's local coordinate frame. The state $\mathbf{s}_t$ is augmented with additional features that encode the staircase's position and orientation.

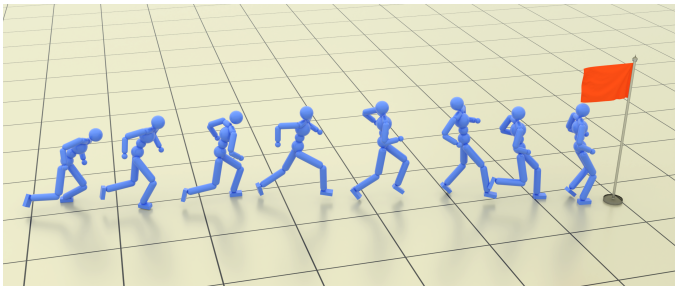
Getup. In this task, the policy is trained to recover from arbitrary fallen states and maintain a standing pose.

8 EXPERIMENTS

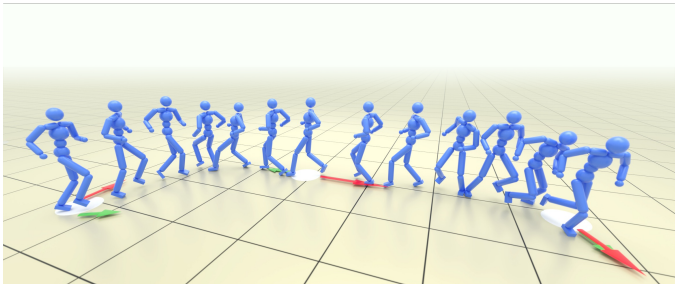
To evaluate the effectiveness of SMP, we apply our framework to train policies for a suite of challenging motion control tasks. In Section 8.1, we demonstrate the *modularity* of SMP by using a pre-trained prior to train new control policies without requiring access to the original motion dataset. By training a style-conditioned diffusion model on the 20-hour 100STYLE dataset, we show that a single pre-trained prior can then be used to train policies to perform tasks in a variety of different styles, where each style-specific prior reward is obtained simply by conditioning the pretrained SMP on the desired style label. New styles can also be synthesized by composing the pretrained diffusion model's predictions from different styles. In Section 8.2, we show that a single motion prior can be *reused* to



(a) Dodgeball

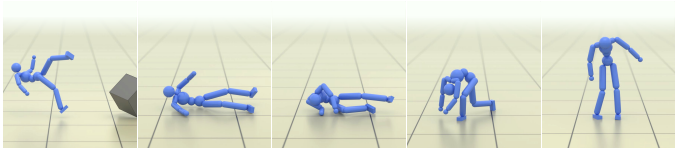


(b) Target Location

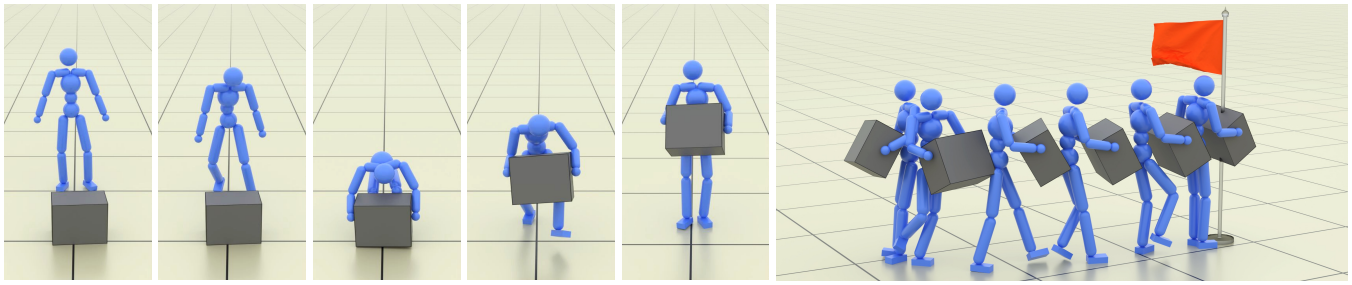


(c) Steering

(d) Target Speed



(e) Getup



(f) Object Carry

Fig. 5. Score-matching motion priors can be trained on a wide range of dataset sizes, independently of any task or control policy. Once trained, an SMP provides a motion imitation objective r^{smp} that can be composed with task rewards r^{g} to train policies that perform a diverse array of tasks, all the while exhibiting natural, life-like behaviors.

train policies across diverse tasks, such as steering, target location, and dodgeball. In addition to guiding the behavioral style of the character, in Section 8.3 we show that SMP can also model *human-object interactions*. By training the diffusion model on human-object interaction data, SMP learns to jointly model both object and character motions, allowing agents to use human-like behavior to perform object interaction tasks. Beyond task-directed behaviors, SMP also enables agents to learn robust and natural recovery behaviors, as demonstrated in Section 8.4. Similar to prior distribution-matching objectives, the policies trained with SMP can synthesize new skills that are not explicitly present in the reference dataset. In Section 8.5, we show that an SMP trained on a small-scale dataset, containing only 3 seconds of walking, jogging, and running motion clips, can automatically lead to the emergence of different locomotion gaits that enable a character to closely follow a wide range of different target speeds, as well as natural transitions between various gaits. Finally, to benchmark SMP's effectiveness for motion imitation, we evaluate SMP on single-clip imitation tasks (Section 8.6). SMP is able to closely reproduce a diverse set of dynamic and acrobatic skills, achieving comparable motion quality to state-of-the-art adversarial imitation learning methods.

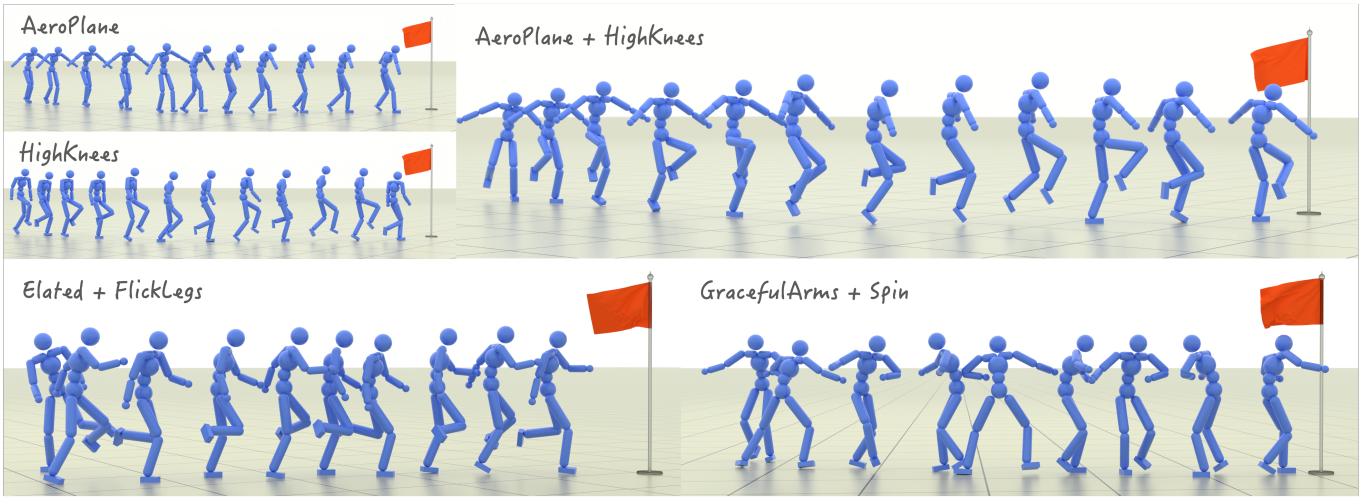


Fig. 6. A pretrained 100-style motion prior can be adapted to synthesize motion priors of new styles. For example, we create a novel "AeroPlane + HighKnees" prior by blending two existing styles in the ϵ-space. This crafted prior, which is used for both generative state initialization and the motion prior objective r^{smp}, enables the agent to perform the target location task with a new, agile style, all without requiring the corresponding reference data.

Baselines: In the following experiments we compare the performance of SMP with AMP [Peng et al. 2021], a widely-used adversarial imitation learning method. Following Peng et al. [2021], the reward weights for AMP are set to $w^{\text{prior}} = w^{g} = 0.5$. In addition to the standard AMP method, we also compare with a variant of AMP that uses a *frozen discriminator*. This configuration evaluates whether the learned adversarial motion prior can be effectively *reused* without further training. First, a control policy and discriminator are trained jointly using AMP. Then the trained discriminator is reused to train new control policies on the same task without further finetuning using data from the new policy. Furthermore, we include a simple *tabula-rasa* learning baseline (w/o Prior), where the policy is trained solely to maximize the task reward ($w^{g} = 1$), without any motion priors.

8.1 One Motion Prior for 100 + N Styles

SMP serves as a modular motion-style reward model that guides policy training without requiring access to the original motion dataset. This enables the application of various adaptation techniques to the diffusion-based reward model that can further shape the motion prior. To demonstrate this capability, we train a general 100style-conditioned motion diffusion model $f(\mathbf{x}^i, c)$ using the entire 20-hour 100STYLE dataset, where c is a style label. Classifier-free guidance (CFG) can then be applied to reshape this general prior into style-specific motion priors [Ho and Salimans 2022]:

$$f_{\text{style}} = f(\mathbf{x}^i, \emptyset) + w_{\text{cfg}} \left(f(\mathbf{x}^i, c_{\text{style}}) - f(\mathbf{x}^i, \emptyset) \right),$$

where the style-conditioned prediction $f(\mathbf{x}^i, c_{\text{style}})$ is applied to the unconditional prediction $f(\mathbf{x}^i, \emptyset)$ according to the guidance weight w_{cfg}. These adapted priors can be used to train policies for a given task using different behavioral styles. The performance statistics for policies trained with various styles are reported in Table 1. All policies are trained using the same underlying SMP model conditioned on different style labels. We find that simply setting the CFG scale to 1.0 is generally sufficient to specialize the 100-style prior into distinct style-specific priors, enabling agents to perform tasks with diverse stylistic behaviors, as shown in Figure 1. While AMP can produce comparable results, it requires training different style-specific discriminators using curated style-specific datasets. In contrast, using a fixed pretrained discriminator (AMP-Frozen) is ineffective in producing the desired stylistics behaviors. With AMP-Frozen, we observe that the discriminator's accuracy drops over the course of policy training, which indicates that the policy is exploiting errors in the discriminator. This exploitation often leads to unnatural behaviors that nonetheless elicit a high score from the discriminator. Qualitative comparisons of the various methods are available in the supplementary video.

Style Composition. SMP supports crafting novel motion priors by composing the diffusion model's outputs, enabling the creation of new styles that are not present in the original dataset without requiring additional training of the prior. As shown in Figure 6, two different styles can be composed by blending their style-conditioned predictions from the diffusion model to produce new styles, such as "*AeroPlane + HighKnees*". The composite prior is constructed via:

$$f_{\text{comp}} = M_{\text{upper}} \odot f(\mathbf{x}^i, c_{\text{aeroplane}}) + M_{\text{lower}} \odot f(\mathbf{x}^i, c_{\text{highknees}}),$$

where M_{upper} and M_{lower} are binary masks applied to the upper- and lower-body features, respectively. The resulting prior f_{comp} can be used directly as the SMP reward r^{smp}. Moreover, our proposed generative state initialization (GSI) can also adopt the composite prior to generate initial states of the new style, enabling effective exploration when training policies for the new style. Using SMP and GSI together with f_{comp}, our framework is able to train an agent that follows task commands while spreading its arms, according to the "AeroPlane" style, and lifting its knees high, according to the "HighKnees" style, all without relying on any reference motion data

Table 1. Performance of policies trained with different styles on the target location task. Task returns are normalized to $[0, 1]$. Style accuracy is evaluated using a style classifier trained on the 100STYLE dataset. AMP models are trained using style-specific motion datasets, whereas SMP is pretrained once on the full 100STYLE dataset and adapted to each style using CFG during policy training. The modularity of SMP enables effective training of style-specific policies without access to the original dataset. AMP with a frozen discriminator is ineffective for producing policies of the desired styles.

Dataset	Style	Task Return			Style Accuracy		
		AMP	AMP Frozen	SMP (Ours)	AMP	AMP Frozen	SMP (Ours)
100STYLE	AeroPlane	$0.867^{\pm 0.001}$	$0.871^{\pm 0.008}$	$0.882^{\pm 0.010}$	$0.981^{\pm 0.004}$	$0.000^{\pm 0.000}$	$0.995^{\pm 0.003}$
	Chicken	$0.876^{\pm 0.005}$	$0.874^{\pm 0.003}$	$0.877^{\pm 0.008}$	$0.973^{\pm 0.019}$	$0.306^{\pm 0.530}$	$0.989^{\pm 0.019}$
	CrossOver	$0.866^{\pm 0.001}$	$0.470^{\pm 0.363}$	$0.879^{\pm 0.002}$	$0.994^{\pm 0.005}$	$0.320^{\pm 0.554}$	$0.994^{\pm 0.005}$
	Dinosaur	$0.886^{\pm 0.001}$	$0.856^{\pm 0.020}$	$0.882^{\pm 0.015}$	$0.998^{\pm 0.004}$	$0.004^{\pm 0.006}$	$0.999^{\pm 0.001}$
	FlickLegs	$0.881^{\pm 0.001}$	$0.580^{\pm 0.269}$	$0.868^{\pm 0.012}$	$0.983^{\pm 0.004}$	$0.009^{\pm 0.011}$	$0.979^{\pm 0.024}$
	HandsBetweenLegs	$0.870^{\pm 0.003}$	$0.888^{\pm 0.007}$	$0.850^{\pm 0.004}$	$0.690^{\pm 0.198}$	$0.000^{\pm 0.000}$	$0.978^{\pm 0.007}$
	HighKnees	$0.882^{\pm 0.003}$	$0.872^{\pm 0.014}$	$0.897^{\pm 0.003}$	$0.988^{\pm 0.009}$	$0.012^{\pm 0.017}$	$0.992^{\pm 0.005}$
	Neutral	$0.873^{\pm 0.007}$	$0.755^{\pm 0.103}$	$0.891^{\pm 0.016}$	$0.991^{\pm 0.005}$	$0.426^{\pm 0.478}$	$0.999^{\pm 0.001}$
	Skip	$0.875^{\pm 0.003}$	$0.512^{\pm 0.321}$	$0.891^{\pm 0.008}$	$0.985^{\pm 0.003}$	$0.425^{\pm 0.479}$	$0.646^{\pm 0.350}$
	Spin (Clockwise)	$0.871^{\pm 0.001}$	$0.874^{\pm 0.012}$	$0.858^{\pm 0.004}$	$0.990^{\pm 0.003}$	$0.006^{\pm 0.010}$	$0.978^{\pm 0.015}$
	Superman	$0.872^{\pm 0.004}$	$0.870^{\pm 0.013}$	$0.893^{\pm 0.005}$	$0.998^{\pm 0.004}$	$0.304^{\pm 0.492}$	$0.999^{\pm 0.002}$
	Zombie	$0.872^{\pm 0.003}$	$0.823^{\pm 0.039}$	$0.875^{\pm 0.014}$	$0.973^{\pm 0.005}$	$0.649^{\pm 0.563}$	$0.996^{\pm 0.006}$
Average		0.874	0.771	0.879	0.962	0.205	0.962

of the composite style. We further demonstrate the flexibility of SMP by crafting more expressive and playful styles. For instance, we create an "Elated + FlickLegs" prior that combines the arm-swinging motion of the "Elated" style with exaggerated lateral leg flicks during locomotion, resulting in an energetic and humorous movement pattern. We also compose the "GracefulArms" and "Spin" styles to produce a ballroom-dance-like behavior, characterized by elegant arm motions and continuous rotational movements.

8.2 One Motion Prior for Multiple Tasks

To further demonstrate the *reusability* of SMP, we show that a pretrained score-matching motion prior can be reused to train multiple policies for different tasks, such as steering, target location and dodgeball, as show in Table 2. The score-matching motion prior is trained on a subset of the LaFAN1 dataset [Harvey et al. 2020], containing unstructured running behaviors. The resulting policies achieve high task returns while exhibiting naturalistic gaits. As shown in Figures 5(b) and 5(c), the character can dynamically transition between different gaits according to changing goals. The *Steering* policy automatically executes a backward jog when the heading direction is oriented in an opposite direction with respect to the target direction, then the policy transitions to a forward jog once the two directions are aligned. The policy also learns lateral gaits, such as side-stepping and cross-stepping, when the heading direction is approximately orthogonal to the target direction.

Policies trained with SMP also exhibit human-like strategies in the *Target Location* task. When the target is far away, the character executes a fast running behavior. When the target is near, the character automatically slows down to slower walking gaits. These nuanced behaviors arise purely from the score-matching motion prior combined with a simple target-distance task objective. No motion planner is required to explicitly specify which gait the character should perform in different scenarios.

SMP provides the policy with the flexibility to adapt behaviors in the dataset in order to create new skills for different tasks. In the challenging *Dodgeball* task, agents trained with the locomotion prior spontaneously develop agile jumping and dodging skills, as shown in Figure 5(a). These behaviors were not present in the original dataset, but they closely resemble real human strategies in dodgeball. In contrast, AMP is unable to learn effective policies for this task, potentially due to the instability of the adversarial objective when the required dodging skill deviates significantly from the reference dataset, which contains only locomotion motions. Furthermore, policies trained using AMP with a frozen discriminator are unable to produce natural behaviors or achieve strong task performance across all tasks. This is because freezing and reusing the AMP discriminator to train new policies introduces severe out-of-distribution inputs for the discriminator. These out-of-distribution inputs can lead to incorrect reward values that cause the policy to adopt behaviors that are ill-suited for the task. This suggests that adversarial priors cannot be effectively reused, even for identical tasks. In comparison, SMP allows a single pretrained motion prior to be reused across different policies and tasks. Moreover, as shown by the task return curves in Figure 7, SMP tends to exhibit more sample-efficient training. This efficiency improvement may in part be attributed to the fixed reward model, which provides more stationary and stable guidance throughout the policy training process compared to the nonstationary reward model from AMP.

8.3 Human-Object Interaction Priors

To demonstrate the effectiveness of our system beyond locomotion tasks, we apply SMP to create *interaction priors* that jointly capture both character and object motions. Given a human-object interaction (HOI) dataset, we train an unconditional HOI diffusion model $f(\mathbf{x}^i_{\text{char}}, \mathbf{x}^i_{\text{obj}})$, which jointly models the distribution of character

Table 2. Performance of combining different motion priors with task objectives, as well as optimizing task objectives alone. Task returns are normalized to $[0, 1]$. A single score-matching motion prior can be effectively reused to train policies across different tasks. Even with a small dataset of only three reference snippets, SMP still enables the agent to perform the target-speed task effectively and with natural motion.

Dataset	Task	Task Return			
		w/o Prior	AMP	AMP Frozen	SMP (Ours)
LaFAN1	Steering	$0.901^{\pm 0.006}$	$0.634^{\pm 0.019}$	$0.243^{\pm 0.008}$	$0.914^{\pm 0.006}$
	Target Location	$0.615^{\pm 0.268}$	$0.737^{\pm 0.014}$	$0.101^{\pm 0.012}$	$0.793^{\pm 0.006}$
	Dodgeball	$0.277^{\pm 0.046}$	$0.233^{\pm 0.003}$	$0.204^{\pm 0.004}$	$0.733^{\pm 0.035}$
Walk-Jog -Run	Target Speed	$0.905^{\pm 0.013}$	$0.904^{\pm 0.002}$	$0.158^{\pm 0.021}$	$0.918^{\pm 0.002}$

Table 3. Performance of SMP on object carry and getup tasks. SMP can be extended to model human–object interaction priors and train effective interaction policies. SMP can also be used to train robust getup policies that can recover from arbitrary fallen states.

Task	Task Return	Success Rate
Object Carry	$0.909^{\pm 0.022}$	$0.997^{\pm 0.004}$
Getup	$0.897^{\pm 0.029}$	$0.998^{\pm 0.003}$

motions $\mathbf{x}_{\text{char}}$ and object motions $\mathbf{x}_{\text{obj}}$. The object's motion is represented by its rotation and position, both expressed in the character's local coordinate frame. The prior reward r_t^{prior} is computed following the same procedure described in Algorithm 1 to evaluate the naturalness of character–object interactions. This object-interaction SMP is combined with task rewards to train policies to accomplish complex, multi-stage *Object Carry* tasks, where the agent walks to a box, picks it up, and then carries it to an arbitrary target location. The learned policy is able to successfully perform this task using natural, and coordinated whole-body manipulation skills.

We also demonstrate that a *Stair Traversal* policy can be trained using an interaction prior constructed from an unconditional human–scene interaction diffusion model. The position and orientation of the staircase are encoded in the character's local coordinate frame and jointly modeled by the diffusion model. As shown in Figure 8, the resulting policy enables the character to climb up and down the staircase with natural motions and stable foot contacts.

8.4 Learning to Get Up

Getting up from arbitrary fallen states is an essential skill for both humans and humanoid agents. Naturalistic and physically plausible recovery behaviors are particularly important for safety and feasibility in real-world robotic systems [He et al. 2025; Tao et al. 2022]. Producing such life-like get-up motions is challenging, as unconstrained policies often exploit highly dynamic or erratic movements that may succeed functionally in simulation but appear unnatural. As a result, many existing approaches rely on carefully designed auxiliary rewards or regularization terms to suppress these artifacts and encourage safe and smooth recovery behaviors [He et al. 2025].

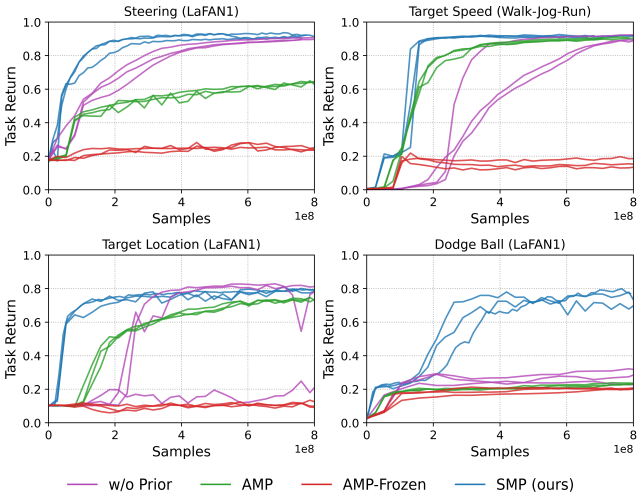


Fig. 7. Comparison of normalized task returns across motion control tasks. SMPs demonstrate better sample efficiency, potentially due to the more stable and consistent RL provided by the stationary SMP reward function.

Our results in Table 3 demonstrate that SMP can successfully train a robust get-up policy capable of recovering from random fallen states by simply combining a body-height objective with a motion-prior objective, without requiring complicated hand-crafted regularization terms. In Figure 5(e), after the character is knocked down, it naturally rolls over, pushes itself up with its hands, and returns to a standing posture. Additional qualitative results are provided in the supplementary video.

8.5 Skill Emergence under Data Scarcity

Reinforcement learning with distribution-matching objectives naturally allows agents to generalize and develop skill that not present in the dataset. In contrast to the typical application of SDS with image-based priors on large dataset, SMP can learn from as little as three seconds of motions, while still providing the agent with the flexibility to adapt and develop new behaviors beyond those in the original dataset. In the *Target Speed* task, the training dataset contains only three motion clips at distinct speeds. Despite the limited reference motion data, the learned policy is able to adapt these motions to follow a continuous range of target speeds, as shown in Figure 5(d). The character naturally adjusts its gait to match the target speed, walking at low target speeds, and transitioning into jogging and running as the speed increases. Although the reference dataset only contains motions at three discrete speeds, the policies learn to modulate the frequency and stride within each gait, producing continuous variations that preserve the style of the original motions while traveling at a broad range of speeds. Moreover, the policies exhibit smooth and naturalistic transitions between gaits that are not present in the dataset, including building up speed from a walk to a jog, decelerating from a run back to a walk, and bursting from a walk into a sprint.

As shown in Figure 7, the learning curves indicate that incorporating the motion prior significantly improves sample efficiency compared to the baseline trained without any prior reward (*w/o Prior*).

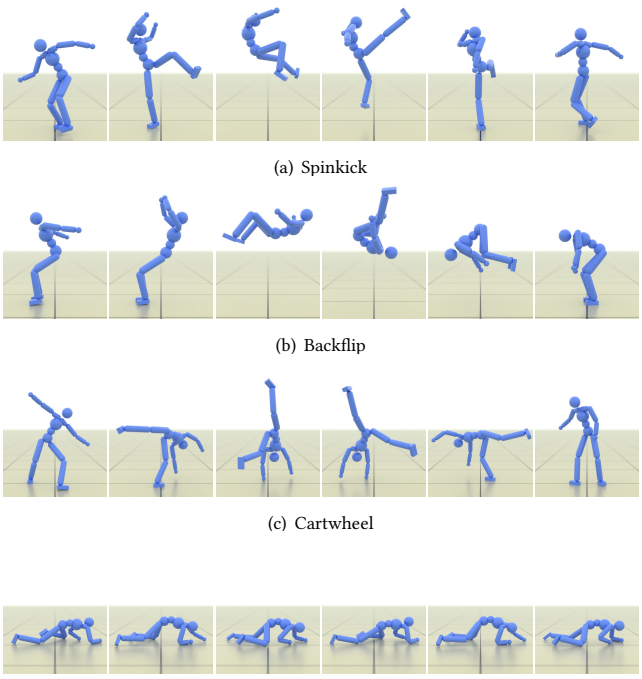
Fig. 8. Example of a stair traversal policy trained with an interaction prior. SMP enables the character to traverse multiple stairs with natural motions.

This improvement arises because the score-matching motion prior provides a dense and informative shaping signal that consistently biases exploration toward physically plausible and natural motions. By constraining the policy to remain within high-likelihood regions of the motion manifold during training, SMP reduces the need for trial-and-error exploration of unnatural behaviors, allowing the policy to focus its learning capacity on task-relevant control. As a result, task rewards and motion priors can reinforce each other, leading to faster convergence and more stable learning.

8.6 Benchmark: Single-Clip Imitation

To further evaluate the effectiveness of SMP at imitating behaviors from reference motion clips, we compare our method with AMP [Peng et al. 2021], AMP-Frozen, and SMILING [Wu et al. 2025a] on a series of single-clip imitation tasks. All policies are trained without the task reward r^g. In addition to comparisons to prior distribution matching methods, we also include comparisons with a motion tracking method, DeepMimic (DM) [Peng et al. 2018a]. Imitation performance is assessed using the position tracking error e_t^{POS}, as defined in Equation (10). Unlike motion-tracking methods such as DeepMimic, which are explicitly designed for precise replication of reference motions, SMP, AMP, and SMILING focus on imitating the general style of the motions. Therefore, these methods do not directly synchronize the policy with the reference trajectory. To ensure a more fair comparison across all methods, we follow the evaluation procedure of Peng et al. [2021] and apply dynamic time warping (DTW) to SMP, AMP, and SMILING to temporally align the motions from the simulated character before calculating the position tracking error [Sakoe and Chiba 1978]. Furthermore, pose-error termination is disabled for DeepMimic to ensure a consistent evaluation setup, as this early termination strategy is not applicable to non-tracking methods.

Quantitative results are summarized in Table 4, with learning curves provided in Figure 10. While AMP exhibits strong imitation performance, it requires retaining the reference motion data throughout policy training. In comparison, SMP accurately reproduces a diverse range of skills, including highly dynamic behaviors, such as backflip, and exhibits comparable imitation accuracy and sample efficiency to AMP, all without requiring access to the reference data during training. Across skills, SMP consistently outperforms SMILING and AMP-Frozen. AMP-Frozen, which attempts to remove AMP's data dependency by simply using a fixed pre-trained discriminator, is unable to accurately reproduce the desired motions.



(a) Spinkick

(b) Backflip

(c) Cartwheel

(d) Crawl

Fig. 9. Visual snapshots of humanoid characters trained via SMP imitating diverse skills, including highly dynamic and contact-rich motions.

The tracking-based DeepMimic benefits from explicit synchronization between the reference motion and the policy, and thereby attains lower tracking errors on certain motions. However, SMP still performs competitively to this tracking-based method. DeepMimic's performance degrades on some challenging motions such as spinkick and cartwheel, where disabling pose-error termination leads to some training runs converging to suboptimal behaviors.

9 REAL-WORLD ROBOTIC APPLICATIONS

In addition to experiments in simulated environments, SMP can be used to train controllers that enable real-world humanoid robots to perform natural, life-like behaviors. Figure 11 shows behaviors from a walking controller is trained entirely in simulation using SMP and then directly deployed on a G1 humanoid robot. Unlike the simulation experiments, the real-world controller has several modified in order to operate effectively in the real world. First, some state information available in simulation is difficult to reliably estimate on the physical robot. Therefore, the policy observes only a partial representation of the state consisting of proprioceptive information that can be recorded through onboard sensors:

$$\mathbf{o}_t \triangleq \left[\boldsymbol{\omega}_t^{\text{root}}, \mathbf{q}_t^{\text{root}}, \mathbf{q}_t, \dot{\mathbf{q}}_t\right] \in \mathbb{R}^{67},$$

which includes the root rotation $\mathbf{q}_t^{\text{root}} \in \mathbb{R}^6$, root angular velocity $\boldsymbol{\omega}_t^{\text{root}} \in \mathbb{R}^3$, local joint rotations $\mathbf{q}_t \in \mathbb{R}^{29}$, and local joint velocity $\dot{\mathbf{q}}_t \in \mathbb{R}^{29}$.

To ensure accurate reward and value estimation during training in simulation, we adopt an asymmetric actor-critic framework in

Table 4. Position tracking error for individual skills. SMP successfully imitates a variety of skills, with imitation accuracy comparable to AMP while not requiring access to reference motion data during policy training. AMP-Frozen, which attempts to eliminate AMP's data dependency by substituting a pre-trained discriminator, results in severely degraded behavior.

Skill	Position Tracking Error [m]				
	DM	AMP	AMP Frozen	SMILING	SMP (Ours)
Walk	$0.010^{\pm 0.001}$	$0.028^{\pm 0.004}$	$0.044^{\pm 0.010}$	$0.042^{\pm 0.007}$	$0.030^{\pm 0.004}$
Run	$0.013^{\pm 0.000}$	$0.088^{\pm 0.010}$	$0.129^{\pm 0.039}$	$0.115^{\pm 0.040}$	$0.067^{\pm 0.001}$
Spinkick	$0.073^{\pm 0.061}$	$0.049^{\pm 0.001}$	$0.324^{\pm 0.040}$	$0.088^{\pm 0.005}$	$0.059^{\pm 0.006}$
Cartwheel	$0.243^{\pm 0.157}$	$0.043^{\pm 0.002}$	$0.419^{\pm 0.013}$	$0.104^{\pm 0.005}$	$0.043^{\pm 0.005}$
Backflip	$0.073^{\pm 0.001}$	$0.058^{\pm 0.002}$	$0.272^{\pm 0.034}$	$0.144^{\pm 0.017}$	$0.069^{\pm 0.008}$
Crawl	$0.006^{\pm 0.000}$	$0.011^{\pm 0.000}$	$0.285^{\pm 0.008}$	$0.061^{\pm 0.057}$	$0.011^{\pm 0.001}$
Average	0.070	0.046	0.246	0.092	0.046

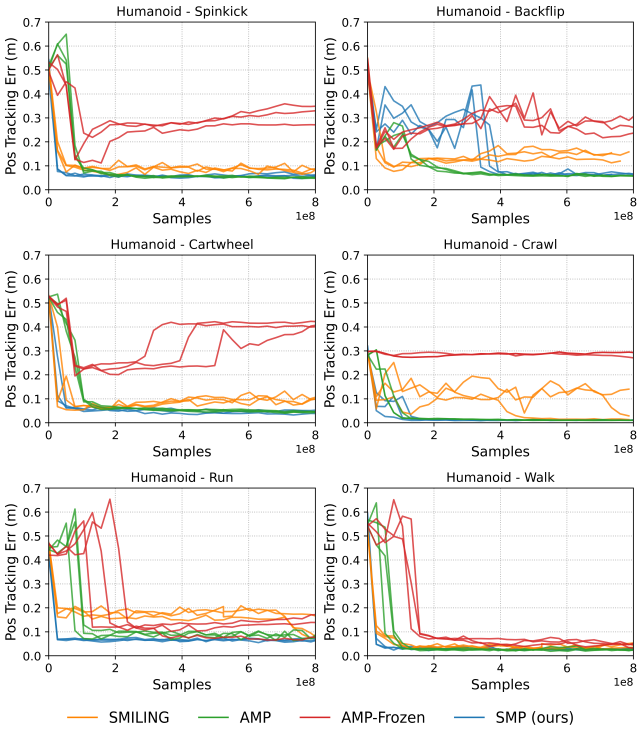
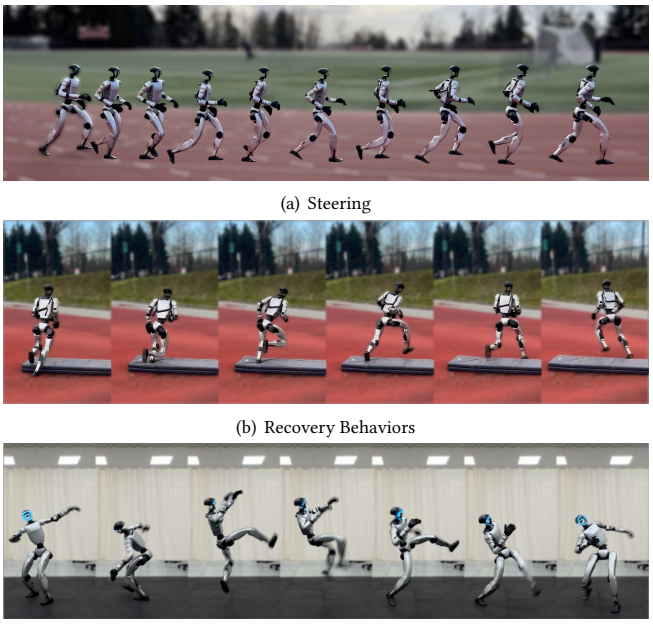


Fig. 10. Learning curves of different methods on the single-clip imitation tasks. Three models initialized with different random seeds are trained for each method. To evaluate the robustness of SMP, different diffusion models trained with different random seeds are used to train each SMP policy. SMP demonstrates consistent and effective performance across seeds.

which the SMP diffusion model and value function receive additional privileged information that are not provided to the policy [Pinto et al. 2018]. The motion features used by the diffusion model to compute the SMP reward r^{smp} are identical to those described in Section 6.1. The critic (*i.e.*, value function) observes additional privileged state information, such as the root height, root linear velocity, and end-effector positions relative to the root. All features



(a) Steering

(b) Recovery Behaviors

(c) Spinkick

Fig. 11. SMP policies deployed on a Unitree G1 humanoid robot. The learned motor controllers can reproduce natural locomotion gaits, robust recovery behaviors, and agile motion skills in the real world.

are recorded in the robot's local coordinate frame. To improve the robustness of the policy to variations in the system dynamics, domain randomization is applied during training in simulation [Peng et al. 2018b], which randomized physical parameters, such as mass, friction and restitution. External perturbations are also applied during training to further enhance the policy's robustness.

In real-world experiments, the robot demonstrates robust locomotion under external perturbations, automatically adjusting its stride to maintain balance and continue moving forward. These recovery behaviors emerge automatically without relying on a runtime motion planner or manually designed heuristics. Additional qualitative results are provided in the supplementary video.

10 ABLATION AND SENSITIVITY ANALYSIS

Our system is one of the first frameworks to construct a *reusable* motion prior based on score-matching, that can achieve high fidelity results comparable to state-of-the-art adversarial imitation learning methods. In this section, we identify the key design decisions that enable more stable training and higher-quality results.

Ensemble Score-Matching. We compare the proposed ensemble score-matching approach ("Random"), which averages multiple SDS evaluations over a fixed set of diffusion timesteps, against the typical strategy used in prior work, which evaluates SDS at a single randomly sampled timestep ("Random") [Luo et al. 2024; Poole et al. 2022; Wu et al. 2025a]. This comparison is conducted on the single-clip imitation tasks using reference motions of varying difficulties, including Run, Cartwheel, and Backflip. As shown in Table 5, the

Table 5. Comparison between computing the SDS objective using a single randomly sampled timestep versus ensemble over a fixed set of timesteps. Policies trained with random sampled timestep exhibit higher errors, particularly on challenging skills such as the backflip.

Skill	Position Tracking Error [m]	
	Random	Ensemble
Run	$0.062^{\pm 0.000}$	$0.067^{\pm 0.001}$
Cartwheel	$0.058^{\pm 0.015}$	$0.043^{\pm 0.005}$
Backflip	$0.195^{\pm 0.006}$	$0.069^{\pm 0.008}$
Average	0.105	0.060

Table 6. Effect of applying adaptive normalization (AdaNorm) to SDS errors. Position tracking errors are estimated with 3 runs initialized with different seeds for each motion, where each training run uses different independently trained diffusion models. AdaNorm improves robustness to variation across pretrained diffusion priors and motion skills.

Skill	Position Tracking Error [m]	
	w/o AdaNorm	w/ AdaNorm
SpinKick	$0.073^{\pm 0.002}$	$0.059^{\pm 0.006}$
Cartwheel	$0.177^{\pm 0.115}$	$0.043^{\pm 0.005}$
Backflip	$0.279^{\pm 0.110}$	$0.069^{\pm 0.008}$
Average	0.176	0.057

single-timestep SDS objective performs comparably on simple skills, but struggles on more challenging behaviors. For instance, when imitating a backflip, policies trained with a single SDS evaluation tend to collapse to standing still or falling backwards instead of executing a complete flip. In contrast, ensembling SDS evaluations across a fixed set of diffusion timesteps provides a more consistent and informative reward signal, which enables effective policy training even for challenging acrobatic motions.

To further analyze the statistical effect of ESM, we evaluate the SDS error of the same motion, a HighKnees clip, 1024 times. With ESM, the variance drops to 9.964×10^{-6}, compared to 1.140 without ESM, while the mean remains comparable at 1.339 versus 1.309. This confirms that ESM substantially reduces variance without introducing significant bias, thereby improving stability of RL training.

Adaptive Normalization of SDS Errors. We evaluate the effect of adaptive normalization of the SDS error across different diffusion model checkpoints and motion skills. As shown in Table 6, when using the same training configuration across multiple diffusion checkpoints, policies trained using SMP *without* adaptive normalization exhibit larger performance variations, indicating greater sensitivity to hyperparameter settings and a greater need for careful manual tuning. In contrast, policies trained *with* adaptive normalization achieve consistently strong performance and low position tracking errors, resulting in higher-quality motions while substantially reducing the burden of manual hyperparameter tuning.

Diffusion Timestep Choice. We evaluate the effect of different diffusion timestep sets $\mathbb{K}$ used in the SMP reward calculation (Equation (8)) on single-clip imitation tasks. As shown in Table 7, different

Table 7. Comparison of using different diffusion timesteps $\mathbb{K}$ for reward calculation. Ensemble score-matching (ESM) enables SMP to be robust to the choice of timesteps. Using diffusion timesteps [22, 15, 8] consistently leads to better performance across tasks and is the default configuration used in all experiments unless stated otherwise.

Skill	Position Tracking Error [m]		
	[15, 8, 1]	[43, 36, 29]	[22, 15, 8]
Run	$0.068^{\pm 0.006}$	$0.059^{\pm 0.006}$	$0.067^{\pm 0.001}$
Cartwheel	$0.055^{\pm 0.000}$	$0.172^{\pm 0.052}$	$0.043^{\pm 0.005}$
Backflip	$0.067^{\pm 0.009}$	$0.064^{\pm 0.005}$	$0.069^{\pm 0.008}$
Average	0.063	0.098	0.060

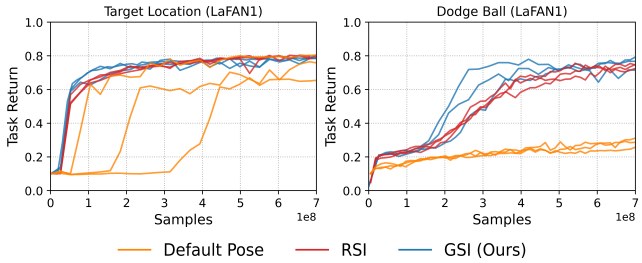


Fig. 12. Learning curves comparing the performance of policies trained with different state initialization strategies: default pose (T-pose) initialization, reference state initialization (RSI), and generative state initialization (GSI). GSI exhibits similar task performance and sample efficiency as RSI, without requiring initial states to be sampled from the original motion clips.

choices of diffusion timesteps yield similar performance, demonstrating that SMP is robust to timestep selection due to ESM. However, a trade-off remains between the reliability and precision of the SMP reward signal. Smaller diffusion timesteps provide more precise and fine-grained guidance, but are less reliable for out-of-distribution samples; this limitation is less critical for simpler tasks such as single-clip imitation, which do not require long-horizon exploration or generalization. In contrast, larger diffusion timesteps yield more reliable but less informative rewards, which can be insufficient for imitating challenging skills that require precise guidance. Across all evaluated tasks, we find that the timestep set [22, 15, 8] consistently achieves the best performance, and we adopt this setting for all experiments unless otherwise stated. However, careful selection of the diffusion timesteps could further improve performance for specific tasks.

Generative State Initialization. Effective state initialization strategies are crucial for improving exploration efficiency in imitation learning. We compare three initialization strategies for the target location task and the dodgeball task: default pose (T-pose) initialization, reference state initialization (RSI), and generative state initialization (GSI). Each control policy is trained under identical conditions, differing only in the initialization method. As shown in Figure 12, RSI achieves better sample efficiency than T-pose initialization by initializing the agent in states closer to the reference motion distribution. Our proposed GSI achieves performance and sample efficiency comparable to RSI, while eliminating the need for

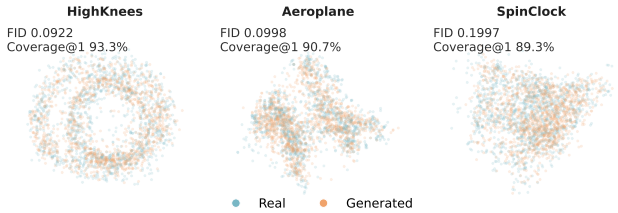


Fig. 13. Distributional similarity between samples from GSI and RSI on HighKnees, Aeroplane, and SpinClock. The samples largely overlap for each style, indicating closely aligned distributions.

access to the original motion dataset during policy training. This result highlights scenarios where reference motion data are unavailable or cannot be retained, and demonstrates that a fully *modular* motion-prior reward framework is feasible.

We further evaluate the quality of initial states produced by GSI compared with those from RSI. We train an autoencoder-based feature extractor and compute Frechet Inception Distance (FID) [Guo et al. 2020; Heusel et al. 2017] and Coverage at threshold 1 (Coverage@1) [Li et al. 2022] in the resulting latent space. Implementation details and metric definitions are provided in Appendix G. We also visualize the samples using PCA, as shown in Figure 13. We compute these statistics on 4096 samples of HighKnees, Aeroplane, and SpinClock, respectively, drawn from both the diffusion model and the reference dataset. The generated samples achieve FID values ranging from 0.0922 to 0.1997 and Coverage@1 above 89.3%, indicating that the generated and reference distributions are closely aligned. The distributional similarity indicated by FID varies across styles, especially for more challenging modes such as SpinClock. Nevertheless, the experiments in Section 8.1 show that the initial states generated by GSI is sufficient for training stylized policies, including SpinClock.

By leveraging a generative motion prior for both reward evaluation and state initialization, the original motion dataset can be completely discarded after prior construction. This enables efficient learning of natural and task-appropriate behaviors without reliance on original motion data. Such modularity facilitates flexible deployment of prior-based imitation learning frameworks and may also offer advantages in settings with data privacy constraints.

11 DISCUSSION AND LIMITATIONS

In this work, we presented *Score-Matching Motion Priors (SMP)*, a *reusable* and *modular* motion prior for physics-based character animation based on score-matching. Once trained, the learned motion prior can be reused to train control policies for diverse tasks, guiding the policies towards natural behaviors that match the reference distribution. Our priors can be effectively used without the need to retain the original motion dataset. Test-time diffusion techniques, such as classifier-free guidance, can be applied to shape the base motion prior and produce novel stylistic priors that enable agents to perform tasks in specific styles. We demonstrate the effectiveness of our method across a diverse variety of settings, ranging from single-character behaviors to human-object interactions. SMPs can be effectively constructed from a wide spectrum of different datasets, with as few as 3 seconds of motion clips to relatively large-scale (20-hour) motion datasets.

In our experiments, we demonstrate that reinforcement learning with score distillation sampling objectives can produce motions of comparable quality to adversarial imitation learning, without the need to continuously update the prior during policy training. SMP also demonstrates higher sample efficiency than adversarial priors in most scenarios, which may be in part due to the more stable stationary reward function from SMP compared to adversarial reward models. However, as with many mode-seeking objectives, policies trained with SMP are susceptible to mode-collapse, converging to a small subset of skills rather than modeling the full distribution of possible behaviors for a particular task. This issue becomes more pronounced when training with large-scale datasets. Possible directions for increasing behavioral diversity include using generative policies or incorporating fine-grained conditioning signals into the diffusion model to explicitly specify which skills should be used by the policy. For state initialization, vanilla GSI may generate invalid motions, such as those with self-collisions, which can introduce simulation instabilities during training. Combining GSI with diffusion sampling techniques, such as guidance, may provide an effective methods to mitigate invalid states, while also serving as a promising direction for prioritized sampling and mining of difficult motions.

While our work primarily focuses on humanoid motion control, we believe SMP can also be applied to a broader range of control problems. We hope this work opens new directions toward building general motion priors that enable more versatile controllers for physics-based character animation and robotics beyond motion tracking.

ACKNOWLEDGMENTS

This work was supported by Sony Interactive Entertainment, NSERC (RGPIN-2015-04843), and the National Research Council Canada (AI4D-166). We would like to thank Michiel van de Panne, Amit H. Bermano, and Alejandro Escontrela for their insights and discussions.

REFERENCES

Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. 2019. DReCon: data-driven responsive control of physics-based characters. *ACM Transactions On Graphics (TOG)* 38, 6 (2019), 1–11.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. 2023. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*.

Stelian Coros, Philippe Beaudoin, and Michiel van de Panne. 2010. Generalized Biped Walking Control. *ACM Transctions on Graphics* 29, 4 (2010), Article 130.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems* 34 (2021), 8780–8794.

Alejandro Escontrela, Xue Bin Peng, Wenhao Yu, Tingnan Zhang, Atil Iscen, Ken Goldberg, and Pieter Abbeel. 2022. Adversarial motion priors make good substitutes for complex reward functions. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 25–32.

Petros Faloutsos, Michiel Van de Panne, and Demetri Terzopoulos. 2001. Composable controllers for physics-based character animation. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 251–260.

F Sebastian Grassia. 1998. Practical parameterization of rotations using the exponential map. *Journal of graphics tools* 3, 3 (1998), 29–48.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie

Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina González, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jáchym Kolář, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbeláez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. 2022. Ego4D: Around the World in 3,000 Hours of Egocentric Video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 18995–19012.

Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. 2020. Action2motion: Conditioned generation of 3d human motions. In *Proceedings of the 28th ACM international conference on multimedia*. 2021–2029.

Yuan-Chen Guo, Ying-Tian Liu, Ruizhi Shao, Christian Laforte, Vikram Voleti, Guan Luo, Chia-Hao Chen, Zi-Xin Zou, Chen Wang, Yan-Pei Cao, and Song-Hai Zhang. 2023. threestudio: A unified framework for 3D content generation. https://github.com/threestudio-project/threestudio.

Félix G Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. 2020. Robust motion in-betweening. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 60–1.

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

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems* 30 (2017).

Jonathan Ho and Stefano Ermon. 2016. Generative adversarial imitation learning. *Advances in neural information processing systems* 29 (2016).

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

Jonathan Ho and Tim Salimans. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598* (2022).

Jessica K Hodgins, Wayne L Wooten, David C Brogan, and James F O'Brien. 1995. Animating human athletics. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. 71–78.

Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-functioned neural networks for character control. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–13.

Daniel Holden, Jun Saito, and Taku Komura. 2016. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–11.

Bo-Ruei Huang, Chun-Kai Yang, Chun-Mao Lai, Dai-Jie Wu, and Shao-Hua Sun. 2024b. Diffusion Imitation from Observation. In *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., 137190–137217. https://proceedings.neurips.cc/paper_files/paper/2024/file/f7faa46b563c2e5343a728c85bace833-Paper-Conference.pdf

Xiaoyu Huang, Yufeng Chi, Ruofeng Wang, Zhongyu Li, Xue Bin Peng, Sophia Shao, Borivoje Nikolic, and Koushil Sreenath. 2024a. DiffuseLoco: Real-Time Legged Locomotion Control with Diffusion from Offline Datasets. *ArXiv* abs/2404.19264 (2024). https://api.semanticscholar.org/CorpusID:269457018

Xiaoyu Huang, Takara Truong, Yunbo Zhang, Fangzhou Yu, Jean Pierre Sleiman, Jessica Hodgins, Koushil Sreenath, and Farbod Farshidian. 2025. Diffuse-CLoC: Guided Diffusion for Physics-based Character Look-ahead Control. *arXiv preprint arXiv:2503.11801* (2025).

Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. 2022. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991* (2022).

Yanqin Jiang, Chaohui Yu, Chenjie Cao, Fan Wang, Weiming Hu, and Jin Gao. 2024. Animate3D: Animating Any 3D Model with Multi-view Video Diffusion. In *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., 125879–125906. https://proceedings.neurips.cc/paper_files/paper/2024/file/e3b53f89136b1bc69a5714ea465f01b6-Paper-Conference.pdf

Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. 2024. Analyzing and improving the training dynamics of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 24174–24184.

Chun-Mao Lai, Hsiang-Chun Wang, Ping-Chun Hsieh, Yu-Chiang Frank Wang, Min-Hung Chen, and Shao-Hua Sun. 2024. Diffusion-Reward Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., 95456–95487. https://proceedings.neurips.cc/paper_files/paper/2024/file/ad47b1801557e4be37d30baf623de426-Paper-Conference.pdf

Yoonsang Lee, Sungeun Kim, and Jehee Lee. 2010. Data-driven biped control. In *ACM SIGGRAPH 2010 papers*. 1–8.

Peizhuo Li, Kfir Aberman, Zihan Zhang, Rana Hanocka, and Olga Sorkine-Hornung. 2022. Ganimator: Neural motion synthesis from a single sequence. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–12.

Yixun Liang, Xin Yang, Jiantao Lin, Haodong Li, Xiaogang Xu, and Yingcong Chen. 2024. Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 6517–6526.

Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. 2023. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 300–309.

Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel Van De Panne. 2020. Character controllers using motion vaes. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 40–1.

Libin Liu and Jessica Hodgins. 2017. Learning to schedule control fragments for physics-based characters using deep q-learning. *ACM Transactions on Graphics (TOG)* 36, 3 (2017), 1–14.

Libin Liu, KangKang Yin, Michiel van de Panne, and Baining Guo. 2012. Terrain runner: control, parameterization, composition, and planning for highly dynamic motions. *ACM Trans. Graph.* 31, 6 (2012), 154–1.

Libin Liu, KangKang Yin, Michiel Van de Panne, Tianjia Shao, and Weiwei Xu. 2010. Sampling-based contact-rich motion control. In *ACM SIGGRAPH 2010 papers*. ACM New York, NY, USA, 1–10.

Calvin Luo, Mandy He, Zilai Zeng, and Chen Sun. 2024. Text-Aware Diffusion for Policy Learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=nK6OnCpd3n

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. 2021. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470* (2021).

Ian Mason, Sebastian Starke, and Taku Komura. 2022. Real-time style modelling of human locomotion via feature-wise transformations and local motion phases. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 5, 1 (2022), 1–18.

Uldarico Muico, Yongjoon Lee, Jovan Popović, and Zoran Popović. 2009. Contact-aware nonlinear control of dynamic characters. In *ACM SIGGRAPH 2009 papers*. 1–9.

Liang Pan, Zeshi Yang, Zhiyang Dou, Wenjia Wang, Buzhen Huang, Bo Dai, Taku Komura, and Jingbo Wang. 2025. Tokenhsi: Unified synthesis of physical human-scene interactions through task tokenization. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 5379–5391.

Teng Pang, Bingzheng Wang, Guoqiang Wu, and Yilong Yin. 2025. DPR: Diffusion Preference-based Reward for Offline Reinforcement Learning. *arXiv e-prints*, Article arXiv:2503.01143 (March 2025), arXiv:2503.01143 pages. https://doi.org/10.48550/arXiv.2503.01143 arXiv:2503.01143 [stat.ML]

Soohwan Park, Hoseok Ryu, Seyoung Lee, Sunmin Lee, and Jehee Lee. 2019. Learning predict-and-simulate policies from unorganized human motion data. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–11.

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

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. 2018b. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 3803–3810. https://doi.org/10.1109/ICRA.2018.8460528

Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel Van De Panne. 2017. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *Acm transactions on graphics (tog)* 36, 4 (2017), 1–13.

Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. 2021. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics (ToG)* 40, 4 (2021), 1–20.

Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. 2018. Asymmetric Actor Critic for Image-Based Robot Learning. In *Proceedings of Robotics: Science and Systems (RSS)*.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. 2022. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988* (2022).

Sigal Raab, Inbal Leibovitch, Guy Tevet, Moab Arar, Amit Haim Bermano, and Daniel Cohen-Or. 2024. Single Motion Diffusion. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=DrhZneqz4n

Marc H Raibert and Jessica K Hodgins. 1991. Animation of dynamic legged locomotion. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*. 349–358.

Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J Guibas. 2021. Humor: 3d human motion model for robust pose estimation. In *Proceedings of the IEEE/CVF international conference on computer vision*.

11488–11499.

Jiawei Ren, Mingyuan Zhang, Cunjun Yu, Xiao Ma, Liang Pan, and Ziwei Liu. 2023. InsActor: Instruction-driven Physics-based Characters. *NeurIPS* (2023).

H. Sakoe and S. Chiba. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26, 1 (1978), 43–49. https://doi.org/10.1109/TASSP.1978.1163055

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* (2015).

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

Agon Serifi, Ruben Grandia, Espen Knoop, Markus Gross, and Moritz Bächer. 2024. Robot motion diffusion model: Motion generation for robotic characters. In *SIGGRAPH asia 2024 conference papers*. 1–9.

Dana Sharon and Michiel van de Panne. 2005. Synthesis of controllers for stylized planar bipedal walking. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation.* IEEE, 2387–2392.

Yi Shi, Jingbo Wang, Xuekun Jiang, Bingkun Lin, Bo Dai, and Xue Bin Peng. 2024. Interactive character control with auto-regressive motion diffusion models. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–14.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020).

Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems* 32 (2019).

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*. https://openreview.net/forum?id=PxTIG12RRHS

Sebastian Starke, Paul Starke, Nicky He, Taku Komura, and Yuting Ye. 2024. Categorical codebook matching for embodied character controllers. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–14.

Richard S Sutton, Andrew G Barto, et al. 1998. *Reinforcement learning: An introduction*. Vol. 1. MIT press Cambridge.

Tianxin Tao, Matthew Wilson, Ruiyu Gou, and Michiel Van De Panne. 2022. Learning to get up. In *ACM SIGGRAPH 2022 conference proceedings*. 1–10.

Guy Tevet, Sigal Raab, Setareh Cohan, Daniele Reda, Zhengyi Luo, Xue Bin Peng, Amit H Bermano, and Michiel van de Panne. 2024. Closd: Closing the loop between simulation and diffusion for multi-task character control. *arXiv preprint arXiv:2410.03441* (2024).

Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-or, and Amit Haim Bermano. 2023. Human Motion Diffusion Model. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=SJ1kSyO2jwu

Takara Everest Truong, Michael Piseno, Zhaoming Xie, and Karen Liu. 2024. Pdp: Physics-based character animation via diffusion policy. In *SIGGRAPH Asia 2024 Conference Papers*. 1–10.

Pascal Vincent. 2011. A connection between score matching and denoising autoencoders. *Neural computation* 23, 7 (2011), 1661–1674.

Kevin Wampler, Zoran Popović, and Jovan Popović. 2014. Generalizing locomotion style to new animals with inverse optimal regression. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–11.

Bingzheng Wang, Guoqiang Wu, Teng Pang, Yan Zhang, and Yilong Yin. 2024. DiffAIL: Diffusion Adversarial Imitation Learning. *Proceedings of the AAAI Conference on Artificial Intelligence* 38, 14 (Mar. 2024), 15447–15455. https://doi.org/10.1609/aaai.v38i14.29470

Jack M Wang, David J Fleet, and Aaron Hertzmann. 2009. Optimizing walking controllers. In *ACM SIGGRAPH Asia 2009 papers*. 1–8.

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. 2023. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems* 36 (2023), 8406–8441.

Andrew Witkin and Michael Kass. 1988. Spacetime constraints. *ACM Siggraph Computer Graphics* 22, 4 (1988), 159–168.

Jungdam Won, Jongho Park, Kwanyu Kim, and Jehee Lee. 2017. How to train your dragon: example-guided control of flapping flight. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 1–13.

Runzhe Wu, Yiding Chen, Gokul Swamy, Kianté Brantley, and Wen Sun. 2025a. Diffusing States and Matching Scores: A New Framework for Imitation Learning. In *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=kWRKNDU6uN

Yan Wu, Korrawe Karunratanakul, Zhengyi Luo, and Siyu Tang. 2025b. UniPhys: Unified Planner and Controller with Diffusion for Flexible Physics-Based Character Control. *arXiv preprint arXiv:2504.12540* (2025).

Michael Xu, Yi Shi, KangKang Yin, and Xue Bin Peng. 2025. Parc: Physics-based augmentation with reinforcement learning for character controllers. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 1–11.

Xinyu Xu, Yizheng Zhang, Yong-Lu Li, Lei Han, and Cewu Lu. 2024. HumanVLA: Towards Vision-Language Directed Object Rearrangement by Physical Humanoid. arXiv:2406.19972 [cs.RO] https://arxiv.org/abs/2406.19972

Heyuan Yao, Zhenhua Song, Baoquan Chen, and Libin Liu. 2022. Controlvae: Model-based learning of generative controllers for physics-based characters. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–16.

Heyuan Yao, Zhenhua Song, Yuyang Zhou, Tenglong Ao, Baoquan Chen, and Libin Liu. 2024. Moconvq: Unified physics-based motion control via scalable discrete representations. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–21.

KangKang Yin, Kevin Loken, and Michiel Van de Panne. 2007. Simbicon: Simple biped locomotion control. *ACM Transactions on Graphics (TOG)* 26, 3 (2007), 105–es.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. 2024. One-step Diffusion with Distribution Matching Distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 6613–6623.

He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. 2018. Mode-adaptive neural networks for quadruped motion control. *ACM Transactions on Graphics (ToG)* 37, 4 (2018), 1–11.

Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. 2022. Motiondiffuse: Text-driven human motion generation with diffusion model. *arXiv preprint arXiv:2208.15001* (2022).

Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. 2019. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5745–5753.

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

APPENDIX

A NETWORK ARCHITECTURE

A schematic illustration of the network architectures used for the different components of our system is shown in Figure 14. The policy is parameterized by a neural network that maps a state $\mathbf{s}$ and task goal g to a Gaussian action distribution $\pi(\mathbf{a}|\mathbf{s}, g) = \mathcal{N}(\mu_\pi(\mathbf{s}, g), \Sigma_\pi)$, where the mean $\mu_\pi(\mathbf{s}, g)$ is input-dependent and the covariance Σ_π is a fixed diagonal matrix. The policy mean is produced by a fully connected network with two hidden layers of sizes [1024, 512], followed by a linear output layer. The value function $V(\mathbf{s}, g)$ is modeled using a similar architecture, but with a single linear output unit. The diffusion model is implemented using two Transformer encoder blocks with multi-head self-attention, feed-forward layers, and adaptive normalization. Each self-attention block uses 4 attention heads with a feature dimension of 64 per head, resulting in an internal embedding dimension of 256. It takes as input a noised motion clip of 10 consecutive frames, $\mathbf{x} := (\mathbf{s}_{t-8}, \ldots, \mathbf{s}_{t+1})$, optionally conditioned on a style label c, and predicts the score $\boldsymbol{\epsilon}$.

B EXPERIMENTAL SETUP

All environments are simulated using Isaac Gym [Makoviychuk et al. 2021], and all neural networks are implemented in PyTorch. During training, 1024 environments are simulated in parallel on a single NVIDIA A100, H100, or RTX 4090 GPU. Most policies converge within 12 hours of training. The reinforcement learning hyperparameters are provided in Table 8. The datasets used for the object carrying and stair traversal tasks are internally collected.

C SINGLE-CLIP IMITATION

To evaluate the performance of the various methods in imitating individual motion clips, we record the position tracking error e_t^{pos}, which measures the differences in both the root position and relative joint positions between the simulated character and the reference motion:

$$e_t^{\text{pos}} = \frac{1}{N^{\text{joint}} + 1}\left(\sum_{j \in \text{joints}} \left\| (\hat{\mathbf{s}}_t^j - \hat{\mathbf{s}}_t^{\text{root}}) - (\mathbf{s}_t^j - \mathbf{s}_t^{\text{root}}) \right\|_2^2 + \left\| \hat{\mathbf{s}}_t^{\text{root}} - \mathbf{s}_t^{\text{root}} \right\|_2^2 \right), \tag{10}$$

where $\mathbf{s}_t^j$ and $\hat{\mathbf{s}}_t^j$ denote the 3D Cartesian position of joint j in the simulated and reference motions, respectively. N^{joint} is the number of joints in the character.

D TASK IMPLEMENTATION

In this section, we describe the implementation details of each task, including hyperparameters and reward design.

D.1 Target Location

In the *Target Location* task, the character is required to reach a user-specified target location $\mathbf{s}_t^{tar}$. The task reward function is formulated as,

$$r_t^{g\text{-}l} = \exp\left(-0.5 \left\| \mathbf{s}_t^{tar\_2d} - \mathbf{s}_t^{root\_2d} \right\|^2 \right). \tag{11}$$

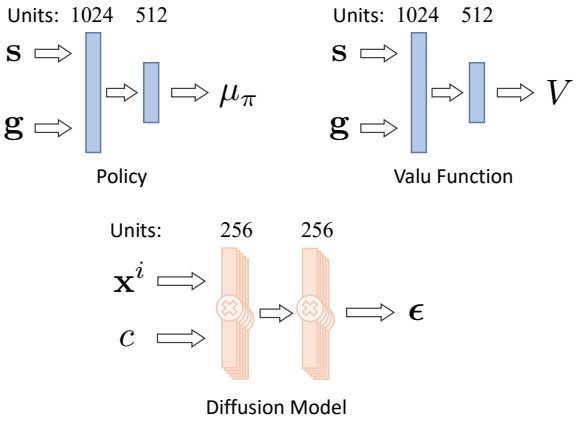


Fig. 14. Network architectures used to model each component of the system. The policy and value functions consist of fully connected layers. The diffusion model is implemented using Transformer blocks with self-attention and adaptive normalization.

The overall reward is then,

$$r_t^l = 0.7 r_t^{g\text{-}l} + 0.9 r^{\text{smp}}, \tag{12}$$

where r^{smp} is computed via Equation (8), and the SDS loss weight is set to $w^s = 4$.

D.2 Steering

The *Steering* task requires the character to track three commands: a desired horizontal moving direction $\mathbf{d}_t^{vel}$, a desired facing direction $\mathbf{d}_t^{face}$, and a target speed $v_t^\star$. To encourage the character to move in the correct direction at the desired speed while also orienting its body appropriately, we define the steering reward as a combination of a velocity tracking term and a facing alignment term:

$$r_t^{g\text{-}s} = 0.5 \exp\left(-2(e_t^{\parallel} + 0.1 e_t^{\perp}) \right) + 0.5 \max(\mathbf{d}_t^{face\top} \mathbf{h}_t, 0). \tag{13}$$

Here, $\mathbf{h}_t$ denotes the character's current horizontal heading direction. We decompose the velocity tracking error into an orthogonal component,

$$e_t^{\perp} = \left\| \dot{\mathbf{s}}_t^{root\_2d} - \mathbf{d}_t^{vel\top} \dot{\mathbf{s}}_t^{root\_2d} \mathbf{d}_t^{vel} \right\|^2, \tag{14}$$

and a parallel component,

$$e_t^{\parallel} = (v_t^\star - \mathbf{d}_t^{vel\top} \dot{\mathbf{s}}_t^{root\_2d})^2. \tag{15}$$

With that, the overall reward for training policies on the *Steering* task is

$$r_t^s = 0.5\, r_t^{g\text{-}s} + 0.5\, r^{\text{smp}}, \tag{16}$$

where the scaling parameter is $w^s = 6$ in r^{smp}.

D.3 Object Carry

The *Object Carry* task requires the character to walk to an object, lift it from the ground, and carry it to a user-specified target location $\mathbf{s}_t^{tar}$. Following Pan et al. [2025], we design the task reward as a composition of several stage-specific components that guide the character through this multi-stage task.

The $r_t^{c\text{-}walk}$ reward guides the character to walk in the direction of $\mathbf{d}_t^{root\_obj}$, a unit horizontal vector pointing from the character's root position $\mathbf{s}_t^{root\_2d}$ to the object position $\mathbf{s}_t^{obj\_2d}$.

$$r_t^{c\text{-}walk} = \begin{cases} 1.0, \left\|\mathbf{s}_t^{obj\_2d} - \mathbf{s}_t^{root\_2d}\right\|^2 < 0.5 \\ \exp\left(-10 \left\|0.2 - \mathbf{d}_t^{root\_obj} \cdot \dot{\mathbf{s}}_t^{root\_2d}\right\|\right), \text{otherwise} \end{cases} \tag{17}$$

Once the character is sufficiently close to the object, it is encouraged to pick the object up from the ground using its hands through $r_t^{c\text{-}hand}$ and $r_t^{c\text{-}pick}$. Here, $\mathbf{s}_t^{hand}$ denotes the midpoint between the character's hands, and $\mathbf{s}_t^{obj\_z}$ denotes the height of the object.

$$r_t^{c\text{-}hand} = \begin{cases} 0.0, \left\|\mathbf{s}_t^{obj\_2d} - \mathbf{s}_t^{root\_2d}\right\|^2 > 0.5 \\ \exp\left(-2 \left\|\mathbf{s}_t^{obj} - \mathbf{s}_t^{hand}\right\|^2\right), \text{otherwise} \end{cases} \tag{18}$$

$$r_t^{c\text{-}pick} = \begin{cases} 0.0, \left\|\mathbf{s}_t^{obj} - \mathbf{s}_t^{hand}\right\|^2 > 0.1 \\ \exp\left(-5 \left\|1.0 - \dot{\mathbf{s}}_t^{obj\_z}\right\|\right), \text{otherwise} \end{cases} \tag{19}$$

Additionally, $r_t^{c\text{-}height}$ encourages the character to maintain the object above a minimum height of 0.9m, while $r_t^{c\text{-}carry}$ incentivizes transporting the object toward the target location $\mathbf{s}_t^{tar}$.

$$r_t^{c\text{-}height} = \begin{cases} 0.0, \left\|\mathbf{s}_t^{obj} - \mathbf{s}_t^{hand}\right\|^2 > 0.1 \\ \exp\left(-7 \left\|0.9 - \mathbf{s}_t^{obj\_z}\right\|^2\right), \text{otherwise} \end{cases} \tag{20}$$

$$r_t^{c\text{-}carry} = \begin{cases} 0.2 r_t^{c\text{-}far} + 0.1 r_t^{c\text{-}near}, \left\|\mathbf{s}_t^{obj\_2d} - \mathbf{s}_t^{tar\_2d}\right\|^2 > 0.25 \\ 0.2 + 0.1 r_t^{c\text{-}near}, \text{otherwise} \end{cases} \tag{21}$$

$r_t^{c-carry}$ consists of two components, $r_t^{c\text{-}far}$ and $r_t^{c\text{-}near}$. $r_t^{c\text{-}far}$ incentives the object to move along the direction of $\mathbf{d}_t^{obj\_tar}$, which is a unit horizontal vector pointing from $\mathbf{s}_t^{obj}$ to $\mathbf{s}_t^{tar}$. It plays a more important role when the object is still far away from the target.

$$r_t^{c\text{-}far} = \exp\left(-5 \left\|1.0 - \mathbf{d}_t^{obj\_tar} \cdot \dot{\mathbf{s}}_t^{obj\_2d}\right\|\right), \tag{22}$$

$r_t^{c\text{-}near}$ rewards the policy based on the Euclidean distance between the object and its target.

$$r_t^{c\text{-}near} = \exp\left(-0.2 \left\|\mathbf{s}_t^{obj\_2d} - \mathbf{s}_t^{tar\_2d}\right\|^2\right), \tag{23}$$

The combined task reward for the *Object Carry* task is given by

$$r_t^{g\text{-}c} = 0.15 r_t^{c\text{-}walk} + 0.15 r_t^{c\text{-}hand} + 0.3 r_t^{c\text{-}pick} + 0.1 r_t^{c\text{-}height} + r_t^{c\text{-}carry}. \tag{24}$$

The reference motions may be imperfect in capturing precise physical interactions. These inaccuracies can be especially detrimental in contact-rich tasks like object carrying, where strict imitation of the reference motion may conflict with successful task execution. To address this, following Xu et al. [2024], we apply style reward clipping to prioritize task rewards. With style reward clipping, the overall reward for the *Object Carry* task is formulated as,

$$r_t^c = 0.5 r_t^{g\text{-}c} + 0.5 \min(r_t^{\text{smp}}, \lambda_t), \tag{25}$$

$$\lambda_t = \max(r_t^{g\text{-}c}, 0.3). \tag{26}$$

Here, r_t^{smp} is computed according to Eq. 8 with $w_s = 6$.

Table 8. Reinforcement learning hyperparameters used in SMP simulated character control experiments.

Parameter	Value
$\mathcal{B}$ Experience Buffer Size	1024×32
K Update Minibatch Size	1024×4
π Policy Stepsize	1×10^{-4}
V Value Stepsize	1×10^{-4}
γ Discount	0.99
SGD Momentum	0.9
GAE(λ)	0.95
TD(λ)	0.95
PPO Clip Threshold	0.2

D.4 Dodgeball

In the *Dodgeball* task, a ball is launched toward the character from a random position 8-10 m away, with a launch speed sampled uniformly from [20, 25] m/s. The ball is relaunched at a random time within a 3-second window. If the character is hit by the ball, the episode terminates early and the agent receives zero reward for all remaining timesteps as a penalty.

To encourage the character to avoid being hit, we define a dodge reward based on the distance between the character's root and the ball:

$$r_t^{g\text{-}d} = 1 - \exp\left(-0.3 \left(\left\|\mathbf{s}_t^{\text{root\_3d}} - \mathbf{s}_t^{\text{ball\_3d}}\right\|_2 - 1.5\right)\right), \tag{27}$$

where the exponential form emphasizes rapid penalty growth when the ball approaches the character, while yielding diminishing penalties once a safe distance is reached.

The final reward used for training dodgeball policies is a weighted combination of the dodge reward and the SMP prior reward:

$$r_t^d = 0.6\, r_t^{g\text{-}d} + 1.2\, r_t^{\text{smp}}, \tag{28}$$

where the SDS error scaling parameter is set to $w^s = 4$ in r^{smp}.

D.5 Getup

To learn the *Get-up* skill, the character is initialized from arbitrary fallen states and trained to recover to a standing pose. To improve exploration efficiency while still enabling recovery from diverse fallen conditions, we initialize the character from random frames of reference get-up motions with 90% probability, and from augmented random fallen states with the remaining 10% probability.

The motion prior reward r^{smp} guides the learning of complex recovery skills, such as rolling over, stepping out for support, and lifting the body, which are difficult to design manually using task-specific heuristics. On the contrary, the task reward $r_t^{g\text{-}up}$ encourages functional recovery by guiding the character to raise its body toward a standing configuration.

We design the task reward as a combination of height and vertical velocity terms:

$$r_t^{g\text{-}up} = 0.3\, r_t^{g\text{-}uph} + 0.7\, r_t^{g\text{-}upv}, \tag{29}$$

where the height reward is defined as

$$r_t^{g\text{-}uph} = \exp\left(-6\ \min(1.2 - \mathbf{s}_t^{\text{head\_h}}, 0)^2\right), \tag{30}$$

and the vertical velocity reward is given by

$$r_t^{g\text{-}upv} = \begin{cases} \exp\left(-100\ \min(0.25 - \dot{\mathbf{s}}_t^{\text{root\_h}}, 0)^2\right), & \mathbf{s}_t^{\text{head\_h}} < 0.6, \\ 1, & \text{otherwise.} \end{cases} \tag{31}$$

The final reward used to train the *Get-up* policy is

$$r_t^{\text{up}} = 0.2\, r_t^{g\text{-}up} + 0.8\, r_t^{\text{smp}}, \tag{32}$$

where the SDS weight is set to $w^s = 8$ in the prior reward r^{smp}.

D.6 Stair Traversal

The *Stair Traversal* task requires the character to walk up and then down a series of steps while tracking a target planar velocity. The staircase consists of 5 steps with a step height 0.175 m, step depth 0.3 m, and step width 0.51 m. The target velocity $\mathbf{v}^{\text{tar}}$ is set to = 1.0 m/s along the x-axis. The goal observation $\mathbf{g}_t = \hat{\mathbf{v}}_t^{\text{tar}}$ consists of the target planar velocity recorded in the character's local coordinate frame as a 2D vector.

The reward function encourages the character to travel at the target velocity while facing the target direction of travel. First, a velocity tracking reward encourages the character to move at the desired velocity:

$$r_t^{st\text{-}vel} = \begin{cases} 0, & \mathbf{v}_t^{\text{root}} \cdot \mathbf{v}^{\text{tar}} \leq 0, \\ \exp\left(-2\left\|\mathbf{v}_t^{\text{root}} - \mathbf{v}^{\text{tar}}\right\|^2\right), & \text{otherwise,} \end{cases} \tag{33}$$

where $\mathbf{v}_t^{\text{root}}$ is the character's 2D root velocity along the horizontal plane. An additional target facing reward is included to encourage the character to face the target direction of travel,

$$r_t^{st\text{-}face} = \max\left(0,\ \mathbf{d}^{\text{tar}} \cdot \mathbf{d}_t^{\text{root}}\right), \tag{34}$$

where $\mathbf{d}_t^{\text{root}}$ is the heading direction of the character's root, and $\mathbf{d}^{\text{tar}} = \mathbf{v}^{\text{tar}}/\|\mathbf{v}^{\text{tar}}\|$ denotes the target direction of travel. The task reward is then given by a weighted combination of the two reward terms:

$$r_t^{g\text{-}st} = 0.7\, r_t^{st\text{-}vel} + 0.3\, r_t^{st\text{-}face}. \tag{35}$$

The task reward is then combined with the SMP reward via:

$$r_t^{st} = 0.5\, r_t^{g\text{-}st} + 0.5\, r_t^{\text{smp}}, \tag{36}$$

where r_t^{smp} is computed from the SDS loss with a scale parameter $w^s = 8$.

E STYLE COMPOSITION

We demonstrate a practical approach for creating new motion styles from a base SMP by composing diffusion model predictions from different styles for the upper and lower body:

$$f_{\text{comp}} = M_{\text{upper}} \odot f(\mathbf{x}^i, c_{\text{style1}}) + M_{\text{lower}} \odot f(\mathbf{x}^i, c_{\text{style2}}),$$

where M_{upper} and M_{lower} are binary masks that select upper- and lower-body motion features, respectively. Although these predictions are computed from the same noised sample $\mathbf{x}^i$ and combined in the ϵ-space, composing styles that are highly dissimilar or contradictory can be challenging. In practice, directly blending upper- and lower-body priors may lead to diluted styles, as the resulting composite score may not correspond to a smoothly realizable motion.

To address this issue, we introduce *multi-step score matching* (MSM), where the score $\hat{\boldsymbol{\epsilon}}$ is estimated through a multi-step reverse diffusion process rather than a single-step prediction. We employ a DDIM sampler with a partial denoising schedule, using a stride of two over three reverse steps [Song et al. 2020]. MSM offers two key advantages. First, the iterative reverse diffusion process naturally blends the composed scores over multiple denoising steps, resulting in more coherent composite styles. Second, the resulting multi-step score estimate is more accurate than a single-step prediction [Liang et al. 2024].

This improved expressiveness comes at the cost of increased computation during reward evaluation. For example, training a policy to perform the composite "*AeroPlane + HighKnees*" style using MSM requires approximately 16 hours of training on a single RTX 4090 GPU. All such operations are applied exclusively during reward computation and do not modify the policy architecture or action space. Once training is complete, the policy alone can complete tasks while exhibiting the learned natural and expressive behavioral style.

Table 9. Domain randomization hyperparameters and regularization terms used in SMP real-world robot deployment experiments.

Term	Value
Observation Noise	
Joint Velocity Noise	$\mathcal{U}(-0.5, 0.5)$ rad/s
Joint Position Noise	$\mathcal{U}(-0.025, 0.025)$ rad
Root Angular Velocity Noise	$\mathcal{U}(-0.25, 0.25)$ rad/s
Root Rotation Noise	$\mathcal{U}(-5, 5)$ deg
Physical Properties	
Friction	$\mathcal{U}(0.1, 2)$
COM Offset	$\mathcal{U}(-0.1, 0.1)$ m
Kp, Kd scale	$\mathcal{U}(0.8, 1.25)$
Push Robot	$\mathcal{U}(-0.5, 0.5)$ m/s
Link Mass	$\mathcal{U}(0.9, 1.1)$
Regularization	
Action Rate	−0.04
Self Contact	−0.2

F REAL-WORLD ROBOTIC DEPLOYMENT

To facilitate real-world deployment, we apply domain randomization techniques during training in simulation. This includes applying external perturbations, injecting noise into the policy's observations, and randomizing physical parameters of the simulator. Table 9 provides a detailed summary of the domain randomization hyperparameters, as well as the regularization terms and their corresponding weights used in the experiments.

G QUANTITATIVE COMPARISON OF GSI AND RSI

To quantitatively evaluate the quality of motions used for state initialization under GSI, we compute the Frechet Inception Distance (FID) [Guo et al. 2020; Heusel et al. 2017] and Coverage at threshold 1 (Coverage@1) [Li et al. 2022], following standard protocols for evaluating kinematic motion quality.

Feature Extractor. For quantitative evaluation, we use the encoder of a trained VAE as the feature extractor. The encoder is implemented as a 4-layer DiT with 4 attention heads, which processes a normalized 10-step motion window and predicts the mean and log-variance of a 16-dimensional latent Gaussian. A lightweight MLP decoder is used to reconstruct the input motion. The VAE is trained with the standard reconstruction-plus-KL objective:

$$\mathcal{L}_{\text{VAE}} = \|\hat{\mathbf{x}} - \mathbf{x}\|_2^2 + \lambda_{\text{KL}} \,\text{KL}\Big(q_\phi(\mathbf{z} \mid \mathbf{x}) \,\|\, \mathcal{N}(\mathbf{0}, \mathbf{I})\Big), \tag{37}$$

where $\lambda_{\text{KL}} = 0.1$ in our implementation, and q_ϕ denotes the posterior distribution parameterized by the encoder with parameters ϕ.

FID and Coverage. We sample 4096 motions from both the diffusion model and the reference dataset. Real and generated motions are normalized using the same VAE normalizer and mapped into latent features using the posterior mean:

$$\mathbf{z} = \mu_\phi(\tilde{\mathbf{x}}) \in \mathbb{R}^{16}. \tag{38}$$

Given real features $\{\mathbf{z}_i^r\}_{i=1}^{N_r}$ and generated features $\{\mathbf{z}_j^g\}_{j=1}^{N_g}$, with empirical means and covariances (μ_r, Σ_r) and (μ_g, Σ_g), respectively, we compute FID as

$$\text{FID} = \|\mu_r - \mu_g\|_2^2 + \text{Tr}\Big(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\Big). \tag{39}$$

For coverage, we first compute the nearest-neighbor distance from each real feature to the generated set in the latent space:

$$d_i = \min_{1 \le j \le N_g} \|\mathbf{z}_i^r - \mathbf{z}_j^g\|_2, \tag{40}$$

and then define coverage at threshold τ as

$$\text{Coverage@}\tau = \frac{100}{N_r} \sum_{i=1}^{N_r} \mathbb{1}[d_i \le \tau]. \tag{41}$$

We report Coverage@1 with $\tau = 1.0$, following Li et al. [2022]; Raab et al. [2024].