VMP: Versatile Motion Priors for Robustly Tracking Motion on Physical Characters

Agon Serifi[1,2] Ruben Grandia[2] Espen Knoop[2] Markus Gross[1,2] Moritz Bächer[2]

[1]ETH Zurich, Switzerland [2]Disney Research, Switzerland

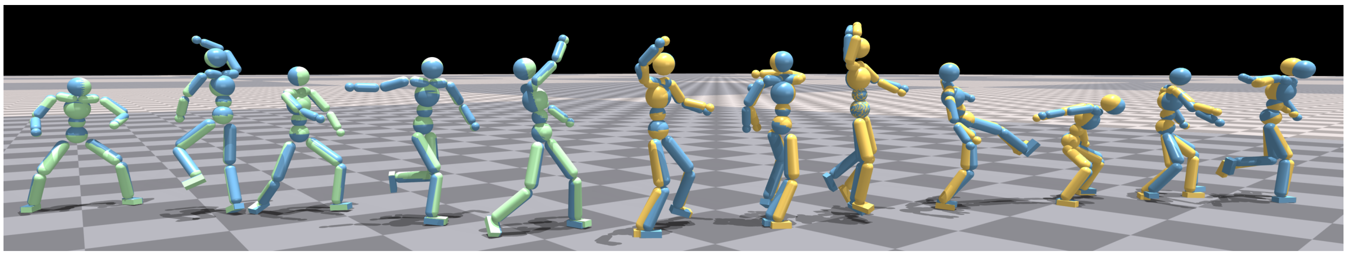

Figure 1: *Our framework enables the control of a physics-based character using user-specified kinematic reference motion (in blue), mapping it to a sequence of temporally and spatially consistent latent codes (decoded in green). In a second step, we condition a reinforcement learning policy on the latent code, generating actuator commands that execute the motion (yellow) while obeying the laws of physics.*

Abstract

Recent progress in physics-based character control has made it possible to learn policies from unstructured motion data. However, it remains challenging to train a single control policy that works with diverse and unseen motions, and can be deployed to real-world physical robots. In this paper, we propose a two-stage technique that enables the control of a character with a full-body kinematic motion reference, with a focus on imitation accuracy. In a first stage, we extract a latent space encoding by training a variational autoencoder, taking short windows of motion from unstructured data as input. We then use the embedding from the time-varying latent code to train a conditional policy in a second stage, providing a mapping from kinematic input to dynamics-aware output. By keeping the two stages separate, we benefit from self-supervised methods to get better latent codes and explicit imitation rewards to avoid mode collapse. We demonstrate the efficiency and robustness of our method in simulation, with unseen user-specified motions, and on a bipedal robot, where we bring dynamic motions to the real world.

CCS Concepts

• ***Computing methodologies*** $\rightarrow$ ***Learning from demonstrations; Learning latent representations; Reinforcement learning; Physical simulation;*** *Animation;* ***Control methods;***

1. Introduction

Imitation-driven reinforcement learning has led to an astonishing leap of progress towards a long-term goal of physics-based character animation, namely accurate and robust tracking across diverse skills. Despite this progress, we lack techniques that achieve this goal with a single policy, covering the diversity in an unfiltered dataset of universal and dynamic motions while providing full-body control of the interactive character.

In this paper, we propose a two-stage technique that takes kinematic reference motion as input and maps it to actuator commands of a character. In a first stage, we extract a latent representation of kinematic motion from an unstructured dataset of motion clips. To this end, we train a variational autoencoder to reconstruct kinematic motion, feeding it with short windows of motion data. In a second stage, we train a policy to imitate motions from the dataset, conditioned on both the current frame of the reference motion and the latent code that corresponds to a time-shifted window centered at the frame. After training, a user-provided kinematic reference is mapped to the latent space and then fed to the policy to control the character.

By training the latent space and control policy separately instead of end-to-end [MHG*18, HPH*20, WGH22], we take advantage of well-studied self-supervised techniques to obtain a structured latent code that captures the diverse distribution of motions within a dataset. Conditioning on latent codes, we can train a single control policy with an explicit imitation reward that preserves the diversity in the input data, without the need for adversarial strategies [PGH*22, DCF*23, TKG*23] that are prone to mode collapse, or a set of expert policies that are trained on clusters of similar motions [WGH20, LCW*23]. Taken together, the motion encoder and control policy enable the control of the full character for skills of high complexity, extracting motion features from short, overlapping subsequences of clips.

With a set of demonstrations on both virtual and a physical humanoid character, we show that a combination of known building blocks arranged in a novel way leads to a simple, yet effective technique that scales well when it comes to diversity and training complexity, tracks unseen dynamic motions closely and physically infeasible motions robustly, and interfaces with common animation techniques and character control modalities. Succinctly, we contribute

- A demonstration of enhanced downstream imitation learning through a pretrained versatile motion prior that provides a structured encoding of kinematic motion.
- A single control policy that is trained on an unfiltered, large-scale dataset of diverse human motions, providing accurate tracking and generalization to unseen input.
- A demonstration that our two-stage processing transfers to robotic characters in the real world, robustly executing expressive motions at the physical limits of hardware.

2. Related Work

Learning-based techniques for character control have received increasing attention in recent years, outperforming model-based techniques in the generation of life-like motions by interfacing with motion data. We focus our review on data-driven methods, in particular in the two broad categories of kinematic and physics-based motion synthesis.

Kinematic Motion Synthesis In a learning-based approach to kinematic motion generation, a common goal is to create a compact representation of motion that allows for smooth temporal and spatial composition [LWH*12, HYNP20, SZKZ20, SMK22], or to use auto-regressive models to learn the distribution of motion sequences [LZCVDP20, RBH*21, CZG*22]. Recently, advanced generative models were used to synthesize kinematic motions [TGH*22, TRG*23, STKB23, RLL*23, RLT*24, LKP*23]. To make the generated motions physically plausible, geometric losses are used to reduce visual artifacts like foot sliding or penetrations. To fulfill stricter constraints, physics engines can be incorporated into the generation process [WGH22, YSI*23].

These works use a latent space to generate output motions and are related to the first stage of our proposed processing. However, we focus on generating physically informed motion and use the embedding as an interface to a low-level controller of a physics-based character instead.

Physics-Based Methods The challenges of designing general objectives that produce natural motions have motivated the use of data-driven techniques that imitate target animations [SKL07, WPP14, LH17, GFK*23]. Simple imitation objectives, together with advances in deep reinforcement learning (RL) [SB18], yield high-quality physics-based character control [PALVdP18, BCHF19, PRL*19, LSCC20]. However, in these works, the policy is limited to the imitation of one or a handful of similar skills, requiring additional mechanisms to transition between more diverse skills.

To learn from large-scale motion datasets, sophisticated methods are proposed that balance and filter motions, or learn motion matching procedures to increase coverage [BCHF19, WGSF20]. Another way to make use of large heterogeneous datasets is to divide the data: Won et al. [WGH20] propose a motion clustering followed by learning a mixture-of-experts, and Luo et al. [LCW*23] propose to train a hierarchical policy where new policies are allocated for increasingly difficult motion sequences. While they can achieve impressive imitation quality on a diverse set of skills, we simplify the process by incorporating a self-supervised kinematic stage that enables the efficient training of a *single and simple* multilayer perceptron policy on a large corpus of data.

Another stream of work focuses on exploiting latent spaces that enable the reuse of policies in a more general setting, targeting a foundation model [BHA*21] for motion control. [ZZLH23, MHG*18, HPH*20, WGH22, GGW*23] jointly train a motion encoder with the policy to extract an embedding that can be reused in high-level RL tasks. Merel et al. [MTA*20] extract a latent space by post-processing multiple expert policies, distilling their knowledge into a unified policy. To increase sample efficiency, a world model can be incorporated [YSCL22, FBH21, FXL23], but does not scale to hours of data [YSCL22] or is sensitive to data quality [FBH21]. Additionally, adversarial learning has been used as an alternative to explicit imitation objectives [PMA*21, PGH*22]. However, this approach suffers from a long training time and is prone to mode collapse. A conditional discriminator mitigates mode collapse in these settings [TKG*23, DCF*23], but does not prevent it. Moreover, due to the long training time, these methods are trained on relatively small datasets, which limits generalization to diverse, unseen motions. We show that a pre-trained latent space in combination with explicit imitation rewards results in a universal motion controller that avoids mode collapse and scales well to large datasets.

Directability of Characters To direct the character, RL-based methods often use a combination of high-level policies, planners, or finite state machines that interface with low-level policies, bridging the gap between high-level commands or task specifications and actuator commands [PMA*21, WGSF20, PRL*19]. In this context, latent space embeddings were used as input to the RL policy [PGH*22, MTA*20]. However, in hierarchical approaches, explicit control of the resulting motion is lost. To give users more control, Juravsky et al. [JGFP22] couple latent control with the latent space of a pre-trained language model [RKH*21]. Xu et al. [XSZK23] present a method that enables spatial imitation of different motion clips while executing a high-level task. However, retraining is required for new tasks or styles.

Similar to previous work [WGSF20, BCHF19], we interface with

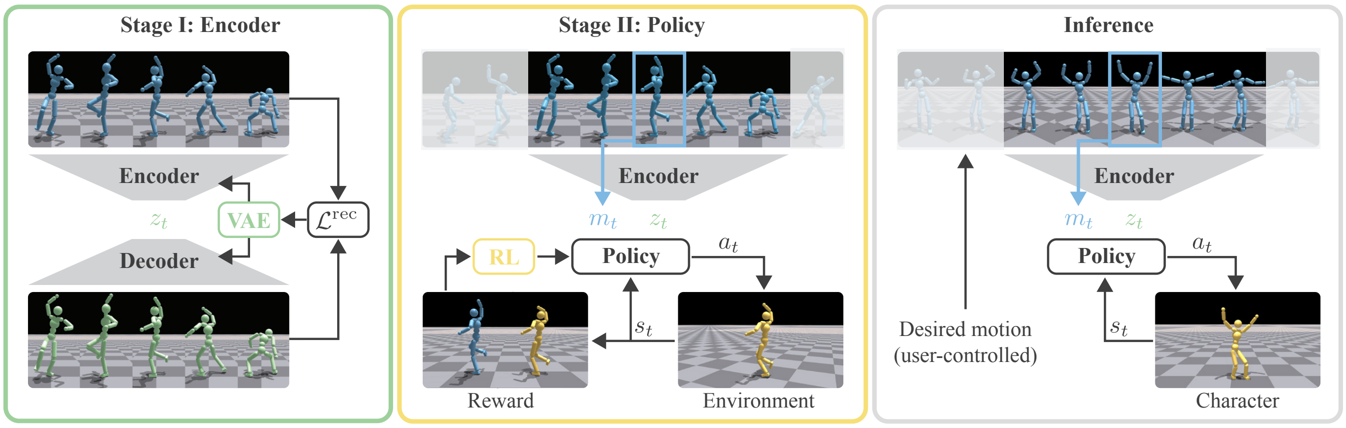

Figure 2: ***Overview.*** *Our two-stage processing starts by taking random samples of motion windows from a database of clips as input to train a variational autoencoder (VAE) to reconstruct them, extracting a latent kinematic motion space (Stage I). Feeding the encoder with time-shifted motion windows, we then train a policy, conditioned on the latent code of the window and the frame at its center, with rewards on the proximity of the simulated to the kinematic state at frame t (Stage II). The resulting encoder-policy pair provides full-body, motion control of the interactive physics-based or robotic character (Inference).*

common control modalities by providing fine-grained control of characters but generalize over a more diverse set of skills by utilizing a kinematic latent space. In summary, we are unaware of a technique that trains a single policy efficiently, provides the same level of coverage, robustness, and generalization properties on a large-scale dataset of complex motions, while also preserving high-fidelity full-body control of the character without the need for further training.

3. Two-Stage Processing

As outlined in Fig. 2, our processing separates the extraction of a kinematic latent space from the training of a dynamics-aware policy. In a first stage, we train an encoder-decoder pair to reconstruct short motion windows that we randomly sample from a large dataset of unfiltered motion clips. These clips are representative of the universal skills of a human, virtual character, or robot. In a second stage, we then train a control policy, conditioned on the kinematic state at the current frame and a latent code for a window around the frame, to maximize kinematic tracking and smoothness rewards. After training, we use the combination of the encoder and control policy to control the full-body motion of a character with unseen input motion. The user interface is therefore the specification of a kinematic motion sequence, which allows for stylized and precise control of the character.

In the next two subsections, we will discuss the first two stages of our processing in detail, followed by an evaluation of our approach with an ablation study, analysis of coverage and smoothness of our latent space, and comparisons to related approaches in Sec. 4. In Sec. 4, we will also demonstrate applications of our approach in the fine-grained motion control of a physics-based and the control of a robotic character.

3.1. Extracting Latent Motion Priors

To temporally and spatially resolve skills at a fine-grained level, we extract the kinematic state, consisting of positions and velocities, for a few past and a few future frames, in addition to the center frame. We then feed these randomly sampled motion windows to a VAE architecture [KW13], learning a latent space that captures the structure of the motion distribution and similarities between skills for a short time horizon around the current state. While velocities provide some information about what the future state of the character will be, velocities can suddenly change due to impact. Hence, it is important to be able to anticipate what the character will do in the near future, or know what has happened in the near past. The launch and landing for a jumping sequence are good examples of why context beyond the current state is important to capture in a latent space. However, to achieve generalization, it is important that the window is short enough, such that the latent space captures fundamental motion building blocks that will also appear in unseen motion and does not overfit to a particular training motion.

More formally, our first stage takes a dataset, $\mathcal{D}$, of distinct clips as input, consisting of a finite sequence of motion frames

$$m_t = \{h_t, \theta_t, v_t, q_t, \dot{q}_t, p_t\}. \tag{1}$$

h_t is the height of the character's root relative to the ground. While we ignore the absolute position of the root in the plane, the inclusion of the relative height is important to differentiate a jump where the height changes from walking on the ground where the height remains constant. θ represents the orientation of the root, expressed as a 6D vector [ZBL*19], and the 6D vector v_t its linear and angular velocity. q_t and $\dot{q}_t$ are the angular positions and angular velocities of all joints. In addition, we include the positions, p_t, of hands and feet, relative to the root, in the features for frame t.

To train our VAE, we extract motion windows of length $2W+1$

$$M_t = \{m_{t-W}, \dots, m_{t+W}\} \tag{2}$$

from individual clips. To normalize them, we first express the orientations, velocities, and end effector positions of individual frames in a local heading frame that we extract from the root pose of the center frame, making the normalized windows invariant to the heading direction. We then use the mean and standard deviation of quantities of all frames in $\mathcal{D}$ to normalize the quantities in the motion window frames, except for orientation, where we skip this second normalization.

The encoder of our VAE, $e_\psi(z_t|M_t)$, maps the motion window to a distribution of latents, $z_t \in \mathbb{R}^{d_z}$, and is modeled as a multivariate Gaussian distribution. The sampled latent representation is then mapped back to input space by a decoder, $M_t' = d_\phi(z_t)$. We train the β-VAE [HMP*17] with a reconstruction loss on the motion window

$$\mathcal{L}^{\text{rec}}(M_t, M_t') = \frac{1}{2W+1} \sum_{i=t-W}^{t+W} l^{\text{rec}}(m_i, m_i'), \tag{3}$$

and the weighted KL-divergence loss with a standard Gaussian distribution prior as the latent distribution. For individual frames, we compute the loss on standard normalized quantities, first computing rotation matrices for orientations using the Gram-Schmidt process

$$\begin{aligned} l^{\text{rec}}(m_i, m_i') = &\|h_i - h_i'\|_2^2 + \|R(\theta_i) - R(\theta_i')\|_F^2 + \|v_i - v_i'\|_2^2 \\ &+ \|q_i - q_i'\|_2^2 + \|\dot{q}_i - \dot{q}_i'\|_2^2 + \|p_i - p_i'\|_2^2. \end{aligned} \tag{4}$$

Because we work with normalized quantities, no relative weighting is needed here.

After training, we prepare for the next stage by encoding motion windows for all frames of all clips, resulting in a latent code z_t per frame m_t. At the beginnings and ends of clips, we repeat the start and end frames to initialize complete windows. Note that in contrast to previous work [PGH*22, JGFP22, TKG*23], we encode a clip with a sequence of latent codes instead of a single one, identifying similarities at a fine-grained level. Besides being advantageous for generalization, our embedding is crucial for precise control, because policies are less reactive otherwise and tend to finish the previous conditioned motion sequence before adapting to a new target.

3.2. Training Conditional Policy

In the second stage of our processing, we train a policy using a reinforcement learning framework [SB18], where the agent interacts with the environment and maximizes the expected discounted return. At each time step, the agent produces an action a_t according to the stochastic policy, $\pi(a_t|s_t, c_t)$, where c_t is the conditional input to the policy, and s_t is the observed state at time t. Provided with the action, the environment then produces the next state, s_{t+1}, and a scalar reward $r_t = r(s_t, a_t, s_{t+1}, c_t)$.

We control our character with a policy that is conditioned on the time-varying latent code, z_t, and the instantaneous motion reference, m_t, i.e., $c_t = (m_t, z_t)$. We normalize m_t with the same procedure as we use for motion windows, with $W = 0$. Our experiments show that adding both modalities is beneficial: The kinematic reference provides instantaneous feedback to the policy and improves tracking accuracy, while the latent code contains information about the intermediate past and future and helps the policy to bring the current target in alignment with similar motions.

During training, we initialize an episode of fixed length T by randomly choosing a frame from the dataset, retrieving the pair (m_t, z_t). We then shift by one frame within the same motion clip to retrieve the next pair. We continue this process until we reach the end of a clip, randomly sampling a new frame from a new clip if the episode has not terminated yet. The randomized initialization avoids the policy getting stuck within the starting sequence of motion clips and leads to an increased learning efficiency [PALVdP18].

Our reward contains a combination of motion tracking, staying alive, and regularization terms that mitigate high-frequency motion,

$$r_t = r_t^{\text{track}} + r_t^{\text{alive}} + r_t^{\text{smooth}}. \tag{5}$$

Following previous work on tracking-based imitation learning [LKL10, PALVdP18, PRL*19], we compute rewards between the reference m_t and simulated pose of the character,

$$\begin{aligned} r_t^{\text{track}} = &-c^h\|h_t - \hat{h}_t\|_2^2 - c^\theta\|R(\theta_t) - R(\hat{\theta}_t)\|_F^2 - c^v\|v_t - \hat{v}_t\|_2^2 \\ &- c^q\|q_t - \hat{q}_t\|_2^2 - c^{\dot{q}}\|\dot{q}_t - \hat{\dot{q}}_t\|_2^2 - c^p\|p_t - \hat{p}_t\|_2^2, \end{aligned} \tag{6}$$

where quantities with a hat denote observations from the simulated state, normalized with the same procedure as we use for motion frames.

To allow ground contact with every body part, we apply early termination on large deviations from the target state that persist for a longer time period, instead of the contact-based termination used in related work [PALVdP18]. Concretely, we terminate the episode if the maximum end-effector deviation, $\|p_t - \hat{p}_t\|_\infty$, exceeds a given threshold, t^p, for more than f frames.

A survival reward provides a simple objective that motivates the character to stay alive and prevent it from seeking early termination at the beginning of training,

$$r_t^{\text{alive}} = c^{\text{alive}}. \tag{7}$$

To mitigate vibrations and avoid unnecessary actions, we apply a first- and second-order action rate penalty, and penalize joint torques τ,

$$\begin{aligned} r_t^{\text{smooth}} = &-c^{\Delta a}\|a_t - a_{t-1}\|_2^2 - c^{\Delta^2 a}\|a_t - 2a_{t-1} + a_{t-2}\|_2^2 \\ &- c^\tau\|\tau\|_2^2, \end{aligned} \tag{8}$$

where the smoothness weights trade off tracking accuracy against the suppression of sliding or vibration artifacts.

Finally, we use domain randomization to increase the robustness of the policy and avoid overfitting to a single set of simulation parameters. The mass of each rigid body is randomized by a percentage error ϵ_m. We perform random pushes on the root, head, hands, and feet. Moreover, we randomize the frictional coefficient of the ground to prevent the policy from exploiting a particular coefficient through foot sliding or vibrations. To further reduce the sim-to-real gap, we added actuator models to the simulator. For robotic characters, we additionally perturb the joint positions by a maximum of ϵ_q to account for inaccuracy in joint calibration, and randomize the friction coefficient of the local ground plane to account for imperfections in the real world.

4. Evaluation and Results

Before we discuss evaluations and applications of our technique, we describe the large and diverse dataset (11 h) that we use for training, also detailing the network architectures and training procedures we use for the two stages.

4.1. Characters, Dataset, and Training Procedure

Characters We evaluate our technique on a standard humanoid with 36 degrees of freedom (DOF) and a bipedal robot (LIME) with 20 DOFs. The robot is 0.84 m tall and weighs 16.2 kg. Both characters are controlled with a set of torques, τ_t, which we compute from the actions with an actuator model [GKH*24]: Taking the actions as inputs, we compute motor torques with proportional-derivative (PD) controllers

$$\tau_t^{\text{motor}} = \kappa_P \cdot (a_t - q_t) - \kappa_D \cdot \dot{q}_t, \tag{9}$$

where $\cdot$ denotes component-wise multiplication. We set the proportional and derivative gains, κ_P and κ_D, according to entries in Tab. 1 top. In addition, we compute frictional torques using a Coulomb model with a viscous component

$$\tau_t^{\text{friction}} = \mu_s \cdot \tanh(\dot{q}_t/\dot{q}_s) + \mu_d \cdot \dot{q}_t, \tag{10}$$

dividing (component-wise) joint velocities by static activation velocities, $\dot{q}_s$, and multiplying the two terms with the static and dynamic friction coefficients, μ_s and μ_d. We finally form the joint torques by clamping the motor torques and subtracting the frictional torques

$$\tau_t = \text{clamp}(\tau_t^{\text{motor}}) - \tau_t^{\text{friction}}. \tag{11}$$

To clamp the torques, we define velocity-dependent minimum and maximum torques. These limits consist of constant limit torques for braking and low velocities, $\tau_{\max}$, and linear limits ramping down the available torques above limit velocities, $\dot{q}_{\tau_{\max}}$. The linear limits cross the maximum velocities, $\dot{q}_{\max}$, when the limit torques are zero. For the two 5 DOF legs of the robot, we use Unitree-A1 (U-A1), and for its neck and arms Dynamixel-XH540-V150-R (D-XH540) actuators. We also use actuator limits for the humanoid, scaled to the size and expected dynamic performance of the character.

The observable state for the humanoid is a 217-dimensional vector consisting of the measured root height, root velocities, joint states, and key body positions, normalized with respect to the heading direction. Additionally, the actions of the previous two time steps are added to the state, which allows the policy to perform well on our smoothness rewards. For our bipedal robot, we omit the root height and key body positions from the state vector. The former is hard to accurately estimate in the real world, while the latter removes the need to compute forward kinematics on the robot. On the physical system, we use encoder measurements from the actuators, together with measurements from an on-board IMU, to estimate the robot's state [HGEG20], incorporating motion capture data for increased accuracy.

Dataset We use three sources of motion data: Reallusion (214 clips, 0.5 h) [Rea24], the CMU mocap dataset (1870 clips, 8.5 h) [CMU01], and Mixamo (2150 clips, 2.0 h) [Mix24]. They span highly diverse motions, from simple walking cycles to acrobatic motions that only very skilled humans can perform. While the Reallusion and Mixamo datasets are artist-processed and hence clean, the CMU dataset contains infeasible motions and noisy data. We work with the full and unfiltered dataset. To retarget the kinematic motions onto our robotic character, we use an inverse kinematics formulation [SKB21].

Table 1: *Actuator, VAE, and RL Parameters.*

Actuator Parameters				
Param. entries	Units	Humanoid	U-A1	D-XH540
κ_P	$\mathrm{N\,m\,rad^{-1}}$	100	15	5
κ_D	$\mathrm{N\,m\,s\,rad^{-1}}$	1.0	0.6	0.2
$\dot{q}_s$	$\mathrm{rad\,s^{-1}}$	0.1	0.1	0.1
μ_s	$\mathrm{N\,m}$	0.45	0.45	0.05
μ_d	$\mathrm{N\,m\,s\,rad^{-1}}$	0.023	0.023	0.009
$\tau_{\max}$	$\mathrm{N\,m}$	500	34	4.8
$\dot{q}_{\tau_{\max}}$	$\mathrm{rad\,s^{-1}}$	20	7.4	0.2
$\dot{q}_{\max}$	$\mathrm{rad\,s^{-1}}$	100	20	7

VAE Parameters		VAE Training	
Param.	Humanoid & Robot	Param.	Value
β	0.002	Batch size	512
W	30	Number of epochs	50 000
d_z	64	Learning rate	0.003
Param.	0.8M	KL-scheduler cycles/ratio	7/0.5
		Warm restart T_0/T_{mult}	1000/5

RL Parameters			RL Training (PPO)	
Param.	Humanoid	Robot	Param.	Value
c^h	0.5	0.0	Batch size	8192×32
c^v	1.0	2.0	Mini-batch size	8192×8
c^θ	1.0	1.0	Clip range, e	0.2
c^q	1.0	7.0	Discount factor, γ	0.99
$c^{\dot{q}}$	0.1	0.1	GAE discount factor, λ	0.95
c^p	1.0	1.0	Number of epochs	5
c^{alive}	6.0	30.0	Desired KL-divergence	0.01
$c^{\Delta a}$	0.1	1.5	Max gradient norm	1.0
$c^{\Delta^2 a}$	0.01	0.45		
c^τ	$1 \cdot 10^{-5}$	$1 \cdot 10^{-4}$		
t^p	0.2 m	0.3 m		
f	3.0 s	3.0 s		
T	30.0 s	30.0 s		
ϵ_m	10.0 %	10.0 %		
ϵ_q	0.2 rad	0.2 rad		

VAE The variational autoencoder [KW13] for our motion latent space is built with 1D-convolutional layers, followed by 4 convResnet blocks without bias components [DJP*20], Layer Normalization [BKH16], ReLU activations [Fuk69], and a final linear layer. We use a latent code dimension of $d_z = 64$ and a window size of $W = 30$, amounting to 1 s. At the bottleneck layer, we double the encoder output dimension and sample from a multivariate Gaussian distribution. The decoder mirrors the encoder with deconv-layers.

For each training iteration, we extract a batch of 512 randomly sampled windows. We use the training objective of a β-VAE [HMP*17] with a KL weight of 0.002 and cyclical scheduling to mitigate KL vanishing [FLL*19]. We use the RAdam optimizer [LJH*20] with an initial learning rate of 0.003, adapted using a cosine annealing scheduler with warm restarts [LH16]. The VAE is trained on an RTX 4090 for 10 h. See Tab. 1 middle for hyperparameters.

RL We keep our policy network small and model it as a diagonal multivariate Gaussian distribution with the mean given by a neural network with ELU activations [CUH16] and 3 hidden multilayer perceptron (MLP) layers of 512 units. We simulate the character using the GPU-accelerated simulator in Isaac Gym [MWG*21], with 8192 environments simulated in parallel on an RTX 4090.

For training, we use Proximal Policy Optimization (PPO) [SWD*17] and train for 48 h. To represent the value function we use the same architecture as for the policy. Observations are normalized using a running mean. See Tab. 1 bottom for reward and PPO parameters.

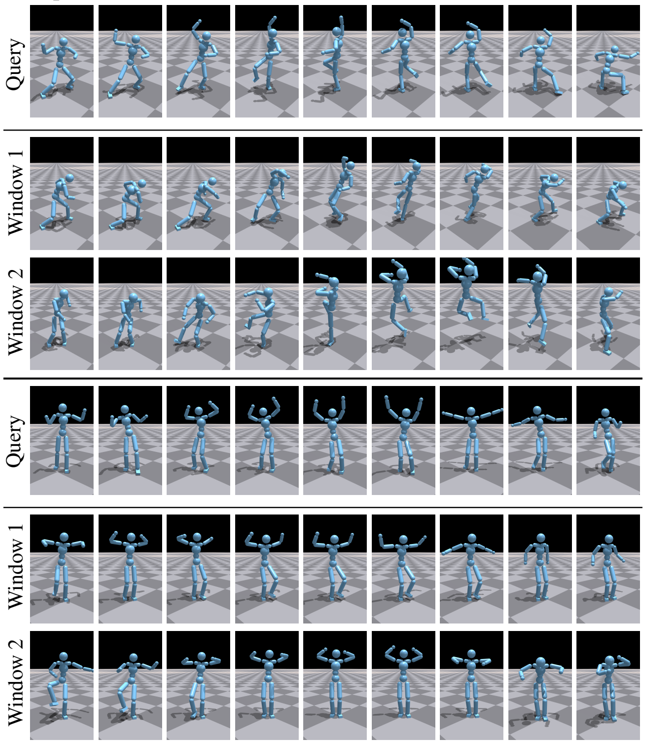

Figure 3: ***Latent Similarity.*** *Given a query motion window from the dataset (top) or unseen input (bottom), we compute the cosine similarity with all other windows using their latent representation. For each query, we show frames from the window with highest (1) and second highest (2) similarity score.*

4.2. Evaluation of Motion Embedding

The goal of our kinematic latent embedding is to capture similarities between short-horizon motion windows around individual frames. To enable generalization, motion windows from unseen kinematic input should map to latent codes that are in proximity to those of similar motion windows within the dataset. To validate that our latent space has desirable properties in this regard, we evaluate the latent space for the standard humanoid and show in Fig. 3 a randomly chosen window from our dataset (top query) and compare it to the two motion windows within the dataset that are closest in latent space. We repeat this experiment for an unseen user-specified input motion (bottom query).

Table 2: ***Ablation Study - Pose.*** *Motion Tracking MAE [joints | end-effectors] [deg | m].* ***Best*** *overall and best without latent codes. The RL policy inputs are based on motion input only (M), latent code (L) or both (LM).*

Input	Motions									
	Idle		Walk		Attack		Dance		Unseen	
M	7.52	0.037	8.63	0.044	13.11	0.065	12.79	0.057	13.29	0.075
M5	6.87	0.042	8.69	0.047	12.52	0.064	13.47	0.060	13.33	0.077
M10	7.96	0.038	9.06	0.044	12.93	0.062	13.76	0.058	13.60	0.079
L	6.10	0.040	7.53	0.054	10.70	0.069	10.45	0.064	10.92	0.072
LM	**4.31**	**0.031**	**4.15**	**0.037**	**7.08**	**0.060**	**5.80**	**0.046**	**7.83**	**0.069**

In the video, we also evaluate the smoothness of our latent space by linearly interpolating between the latent space trajectories corresponding to two different clips. After interpolation, we decode this sequence and visualize the middle frame of the decoded window as a new motion clip. This results in natural in-between skills and demonstrates that our latent space provides the expected smoothness. We also visualize the reconstruction of unseen motion windows, exhibiting the expected coverage and generalization.

Comparison with End-to-End Latent Codes CALM [TKG*23] trains a latent representation end-to-end with the control policy. We compare the quality of the resulting representation by assessing the separability between motion classes within the latent space using Linear Discriminant Analysis (LDA) [Bis95] and by quantifying the mutual information between input motion and latent codes. To facilitate comparison, we train the first stage of our method on their dataset and use the same 2 s window length. Upon encoding the entire dataset using both approaches, motion classes for LDA are constructed by labeling consecutive 2 s of latent codes as a single class, with the subsequent 2 s being omitted to ensure distinct classes. For mutual information analysis, the same dataset is utilized, and the latent codes are discretized into 100 bins per dimension. Mutual information scores are then computed for each pair of features. The results of LDA accuracy and mutual information in the inset table demonstrate that our latent space preserves more information and exhibits superior separability of motions compared to the end-to-end method.

Method	LDA acc. ↑	MI ↑
CALM	0.687	0.121
Ours	**0.854**	**0.341**

4.3. Evaluation of Two-Stage Processing

We first ablate our proposed choice of conditional policy inputs using a smaller dataset. Subsequently, we discuss the enhanced tracking performance and generalization potential achieved through training on a larger dataset. We then investigate the robustness of our policy in the face of infeasible inputs, disturbances, as well as its responsiveness to input discontinuities. All evaluations are performed with the standard humanoid.

Ablation For our ablation study, we train the latent space as well as variants of our policy on the Reallusion dataset (0.5 h) and evaluate tracking performance qualitatively (Fig. 4, accompanying video) and quantitatively (Tab. 2). For evaluation of a policy, we generate 1024 fixed-size episodes with the same procedure as we use for

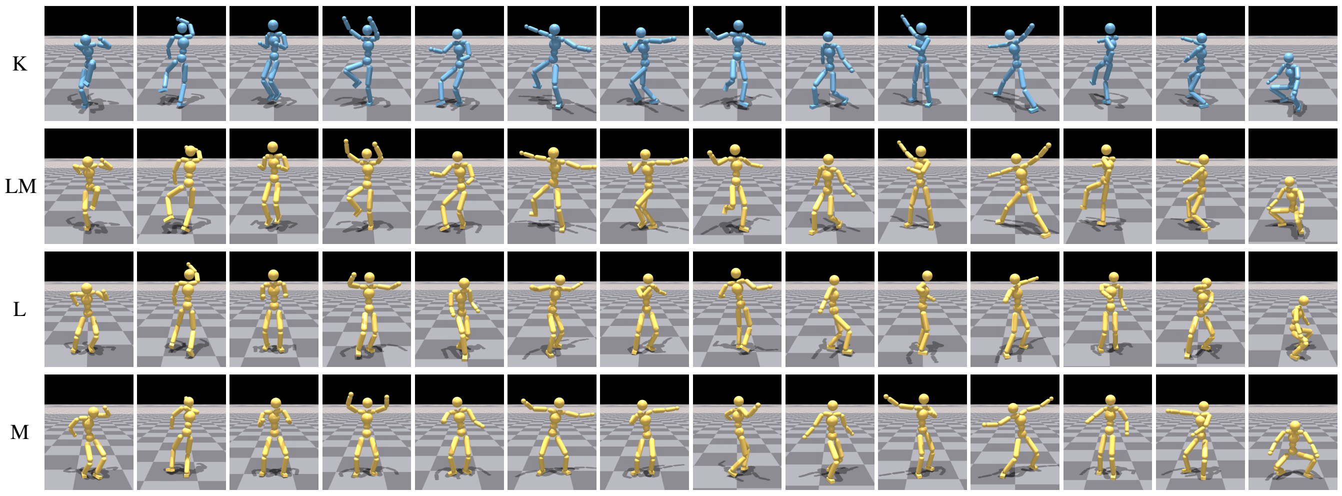

Figure 4: ***Tracking Performance.*** *Our proposed method (LM) tracks a kinematic input (K) significantly better than policy variants that are conditioned on the latent code (L) or the reference motion (M) only.*

training (Sec. 3.2), restricting the random sampling of clips to categories of increasing difficulty from Reallusion (Idle, Walk, Attack, Dance) and a small, randomly chosen subset of motions from the other two datasets (Unseen). To compare tracking performance, we report the mean absolute error (MAE) of joint positions over all frames, joints, and evaluation episodes (Tab. 2).

We first condition the policy on a single motion frame (M) and on motion windows with 5 (M5) and 10 (M10) additional future frames. These variants are similar to related approaches that target the fine-grained control of characters (e.g., DReCon [BCHF19], UniCon [WGSF20]), omitting the use of additional mechanisms or policies for a fair comparison. As expected, the tracking error is higher for more challenging motions or unseen data. While the performance is good for less complex motions, it degrades quickly for more challenging or unseen input, visually perceived as the result of a low-pass filter. Similar to UniCon [WGSF20], we did not observe significant benefits from additional future frames.

In the lower half of Tab. 2, we report numbers for a variant that is only conditioned on the latent code (L), together with our proposed processing (LM). While the L-variant already consistently improves tracking performance, our LM variant clearly outperforms all other variants in all categories, especially for highly dynamic or unseen motions (Fig. 4; $\sim$50 % reduction in tracking error for Dance).

Comparison with End-to-End Method To visually compare our method to CALM [TKG*23], we sample random motion windows from the dataset. For CALM, we compute the corresponding latent code and condition the policy with the same code for a 2 second window, aligning with their training strategy. We then track the same motion using our pipeline. The video presents the results side-by-side. Note that CALM requires two weeks of training on an industrial-grade A100 GPU, whereas our complete method trains in under three days on a consumer-grade RTX 4090. Retraining CALM on a consumer-grade GPU would take months. As demonstrated in the video, our method matches CALM's motion quality while exhibiting fewer artifacts. Our stronger coupling between latent code and target motion allows for easier control, whereas CALM's latent-only conditioning can result in repeated motion and is less reactive.

Generalization To achieve generalization, it is essential to train on a large dataset. In our two-stage processing, the additional conditioning on the latent code during RL has a negligible effect on iteration time compared to the simple M-variants. This enables us to scale our approach and train a policy on a significantly larger dataset compared to related end-to-end approaches (ASE [PGH*22], PADL [JGFP22], CALM [TKG*23], CASE [DCF*23]). To demonstrate the utility of our pre-trained latent space, we use the encoder and policy trained on the small Reallusion dataset as a baseline. Next, we use the small dataset during RL while using an encoder that is trained on the full set, with the exception of clips that we use for evaluation (Unseen). This already reduces the MAE tracking error on unseen motions by 15 % (see inset), indicating that the generalization performance of the policy is enhanced by improving the encoder. By training both the encoder and the policy on the large set (except Unseen), we reduce the error to about 5° on unseen data. In the accompanying video, we provide a visual comparison of these three variants on a sequence of unseen dancing and fighting motions. We observe excellent tracking performance with our single policy that is trained on the full set, with noticeable artifacts for the variant that uses only the Reallusion dataset for both stages.

	RL Small	RL Full
VAE Small	7.83	-
VAE Full	**6.62**	**5.03**

Robustness Our policy is robust to motions that are far from feasible, as shown in Fig. 5 and demonstrated in our video: The policy tracks the in-air stair climbing sequence as closely as possible while maintaining dynamic balance on the ground. In our video, we also demonstrate robustness under heavy disturbances by applying random forces and torques to the root, head, hands, and feet of the character. Additionally, on previously unseen uneven terrain, the character makes necessary adjustments to maintain balance and tracks the intended motion whenever possible. The method reaches its limits when the target motion is fast and physically unattainable. In such cases, the policy might either not track the motion or result in failure as demonstrated in our video.

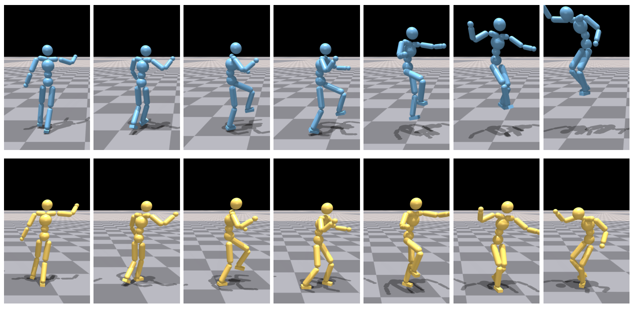
Figure 5: ***Robustness.*** *Conditioned on frames from an infeasible reference motion, the character remains stable and tries to track the performance as well as possible.*

Reactivity We evaluate the capability of the policy to abruptly transition between motions in the accompanying video. Since the policy is conditioned on a stream of latent codes rather than a single code for a longer sequence, it reacts instantly. This means that a user can arbitrarily sequence motions, without the need for smooth transitions. The policy is able to quickly adapt even if there are discontinuities in the input stream.

4.4. Directability

Our policy interfaces with kinematic reference motion, generalizes well to new motions, and is robust to irregular input. It, therefore, seamlessly integrates into the standard workflows of artists and allows them to directly control a physics-based character without additional training. Hereafter, we present examples of how an artist might use our technique.

Spatial Composition Our policy allows users to control the character by spatially composing motions of different body parts as illustrated in Fig. 6 and our video with three examples where arm motions are sourced from a different clip than the body motion. Even for random spatial compositions of motions that can result in implausible or unnatural input, we observe that our policy tracks the motion as closely as possible within the physical limits.

Motion Editing With our technique, users can sequence full or partial motion clips in an arbitrary order to quickly create an initial reference animation for a character, as illustrated in Fig. 7 for a fighting sequence (Start). Because our method provides full-body control, they can then edit the motion reference to precisely time key events at key locations to achieve a particular task (Precision, hitting a pillar), followed by a stylization pass (Stylization).

Artist-Created Motion We additionally asked an artist to create a dancing sequence with leg, arm, and full-body movements of increasing complexity with standard animation tools and workflows that are not physics-aware. As is common for artist input, there is visible foot sliding. Our method robustly tracks the performance and removes all foot sliding (see video). Because our inference is interactive, an artist could use our technique during animation to create physics-informed motions.

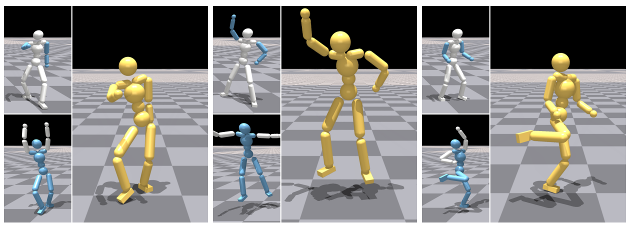
Figure 6: ***Spatial Composition.*** *Input motions can be spatially composed (selected bodies visualized in blue) without any additional training. Our method tracks the new motions as closely as possible within physical limits.*

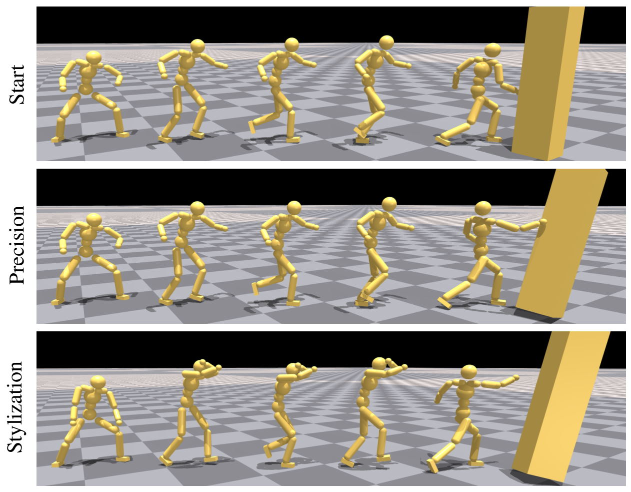

Figure 7: ***Motion editing.*** *Provided with an initial motion sequence (Start), a user can adapt the reference motion to precisely control the character (Precision) or change the style (Stylization).*

4.5. Robot Control

Recent progress in robotics in bridging the so-called sim-to-real gap [ZQW20] provides evidence that extended domain randomization [PAZA18] and the inclusion of actuator dynamics in the simulation [HLD*19] can facilitate real world deployment at the expense of presenting the RL algorithm with a significantly harsher training environment. We show that our presented pipeline, with actuator and learning parameters as summarized in Tab. 1, and domain randomization as discussed in Sec. 3.2, leads to a robust and effective policy that allows us to transfer versatile motions to a bipedal robot. Fig. 8 shows an example of a reference motion (top) and the resulting execution on hardware (bottom). Because our robotic character does not have an ankle-roll actuator, it cannot balance on a single leg for an extended amount of time. Interestingly, as shown in the last frame, the policy adapts by placing the tip of the right leg on the floor such that the reference pose can be closely matched without losing balance. The video shows additional demonstrations of highly dynamic motions. The quality of motion tracking remains high, even when constrained by the practical limits of physical actuators. Our results demonstrate that our method is effective on real robotic characters, showing that we do not rely on artificially strong simulated characters, external helper forces, or simulation artifacts.

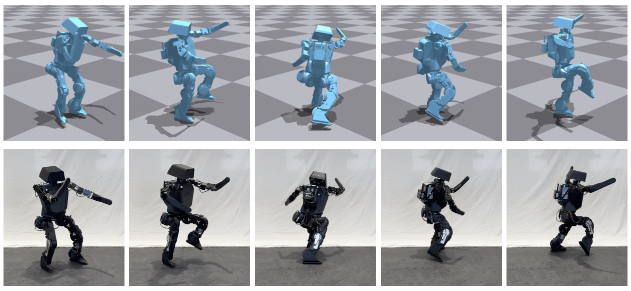
Figure 8: ***LIME Robot.*** *Transfer of dynamic motion skills onto a bipedal robot. The policy tracks the style as accurately as possible while maintaining balance.*

5. Conclusions

We have shown that a pre-trained latent representation of motion improves both the tracking and generalization performance of a downstream RL-based control policy. Our two-stage training allows us to robustly track a diverse set of skills. Moreover, our kinematic motion interface empowers users to craft character-specific animations and have full-body control over physics-based characters. While not demonstrated explicitly, our approach would also interface with other common control modalities. Recent kinematic motion generators [TRG*23, GZZ*22] could be used together with the policy to solve generative tasks in a physical environment.

However, while our method has proven its strength on a diverse set of motions, it has difficulties in imitating clips that require a longer planning horizon. Acrobatic motions like backflips require a certain commitment to the performance. We believe that simple MLP policies cannot achieve generalization to this class of motions, requiring more sophisticated architectures with hidden states to accomplish coverage over unseen acrobatic input with extended flight phases. Moreover, while our method can track a kinematic reference, we have yet to explore generative capabilities of our processing, requiring an additional mechanism to traverse our time-varying latent space.

By demonstrating expressive motions on robotic hardware, we have unified recent progress in both the computer graphics and robotics communities. We see great potential at the intersection of these two fields and are excited about the future avenue of bringing increased agility and expressivity to more physical characters. In particular, we believe that the combination of self-supervised and reinforcement learning on large datasets provides a path to universal control policies that serve as a foundation model for a wide range of downstream tasks.

Acknowledgments

We would like to thank Doriano van Essen and Violaine Fayolle for their artistic input and Reallusion [Rea24] for letting us use their datasets.

References

[BCHF19] Bergamin K., Clavet S., Holden D., Forbes J. R.: Drecon: Data-driven responsive control of physics-based characters. *ACM Trans. Graph. 38*, 6 (nov 2019). doi:10.1145/3355089.3356536. 2, 7

[BHA*21] Bommasani R., Hudson D. A., Adeli E., Altman R., Arora S., von Arx S., Bernstein M. S., Bohg J., Bosselut A., Brunskill E., et al.: On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258* (2021). 2

[Bis95] Bishop C. M.: *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., USA, 1995. 6

[BKH16] Ba J. L., Kiros J. R., Hinton G. E.: Layer normalization, 2016. arXiv:1607.06450. 5

[CMU01] CMU: Cmu graphics lab motion capture database, 2001. URL: http://mocap.cs.cmu.edu/. 5

[CUH16] Clevert D.-A., Unterthiner T., Hochreiter S.: Fast and accurate deep network learning by exponential linear units (elus), 2016. arXiv:1511.07289. 6

[CZG*22] Chandran P., Zoss G., Gross M., Gotardo P., Bradley D.: Facial animation with disentangled identity and motion using transformers. *Computer Graphics Forum 41*, 8 (2022), 267–277. doi:https://doi.org/10.1111/cgf.14641. 2

[DCF*23] Dou Z., Chen X., Fan Q., Komura T., Wang W.: C·ase: Learning conditional adversarial skill embeddings for physics-based characters. In *SIGGRAPH Asia 2023 Conference Papers* (New York, NY, USA, 2023), Association for Computing Machinery. doi:10.1145/3610548.3618205. 2, 7

[DJP*20] Dhariwal P., Jun H., Payne C., Kim J. W., Radford A., Sutskever I.: Jukebox: A generative model for music, 2020. arXiv:2005.00341. 5

[FBH21] Fussell L., Bergamin K., Holden D.: SuperTrack: motion tracking for physically simulated characters using supervised learning. *ACM Trans. Graph. 40*, 6 (Dec. 2021), 1–13. doi:10.1145/3478513.3480527. 2

[FLL*19] Fu H., Li C., Liu X., Gao J., Celikyilmaz A., Carin L.: Cyclical annealing schedule: A simple approach to mitigating KL vanishing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (June 2019), Burstein J., Doran C., Solorio T., (Eds.), Association for Computational Linguistics, pp. 240–250. doi:10.18653/v1/N19-1021. 5

[Fuk69] Fukushima K.: Visual feature extraction by a multilayered network of analog threshold elements. *IEEE Transactions on Systems Science and Cybernetics 5*, 4 (1969), 322–333. 5

[FXL23] Feng Y., Xu X., Liu L.: Musclevae: Model-based controllers of muscle-actuated characters. In *SIGGRAPH Asia 2023 Conference Papers* (New York, NY, USA, 2023), SA '23, Association for Computing Machinery. doi:10.1145/3610548.3618137. 2

[GFK*23] Grandia R., Farshidian F., Knoop E., Schumacher C., Hutter M., Bächer M.: DOC: Differentiable optimal control for retargeting motions onto legged robots. *ACM Trans. Graph. 42*, 4 (July 2023), 1–14. doi:10.1145/3592454. 2

[GGW*23] Gehring J., Gopinath D., Won J., Krause A., Synnaeve G., Usunier N.: Leveraging demonstrations with latent space priors. *Transactions on Machine Learning Research* (2023). URL: https://openreview.net/forum?id=OzGIu4T4Cz. 2

[GKH*24] GRANDIA R., KNOOP E., HOPKINS M. A., WIEDEBACH G., BISHOP J., PICKLES S., MÜLLER D., BÄCHER M.: Design and Control of a Bipedal Robotic Character. In *Proceedings of Robotics: Science and Systems* (Delft, the Netherlands, July 2024). 5

[GZZ*22] GUO C., ZOU S., ZUO X., WANG S., JI W., LI X., CHENG L.: Generating diverse and natural 3d human motions from text. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), pp. 5142–5151. doi:10.1109/CVPR52688.2022.00509. 9

[HGEG20] HARTLEY R., GHAFFARI M., EUSTICE R. M., GRIZZLE J. W.: Contact-aided invariant extended kalman filtering for robot state estimation. *The International Journal of Robotics Research 39*, 4 (2020), 402–430. 5

[HLD*19] HWANGBO J., LEE J., DOSOVITSKIY A., BELLICOSO D., TSOUNIS V., KOLTUN V., HUTTER M.: Learning agile and dynamic motor skills for legged robots. *Science Robotics 4*, 26 (2019), eaau5872. 8

[HMP*17] HIGGINS I., MATTHEY L., PAL A., BURGESS C., GLOROT X., BOTVINICK M., MOHAMED S., LERCHNER A.: beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations* (2017). 4, 5

[HPH*20] HASENCLEVER L., PARDO F., HADSELL R., HEESS N., MEREL J.: CoMic: Complementary task learning & mimicry for reusable skills. In *Proceedings of the 37th International Conference on Machine Learning* (13–18 Jul 2020), III H. D., Singh A., (Eds.), vol. 119 of *Proceedings of Machine Learning Research*, PMLR, pp. 4105–4115. URL: https://proceedings.mlr.press/v119/hasenclever20a.html. 2

[HYNP20] HARVEY F. G., YURICK M., NOWROUZEZAHRAI D., PAL C.: Robust motion in-betweening. *ACM Trans. Graph. 39*, 4 (aug 2020). doi:10.1145/3386569.3392480. 2

[JGFP22] JURAVSKY J., GUO Y., FIDLER S., PENG X. B.: Padl: Language-directed physics-based character control. In *SIGGRAPH Asia 2022 Conference Papers* (New York, NY, USA, 2022), Association for Computing Machinery. 2, 4, 7

[KW13] KINGMA D. P., WELLING M.: Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013). 3, 5

[LCW*23] LUO Z., CAO J., WINKLER A. W., KITANI K., XU W.: Perpetual humanoid control for real-time simulated avatars. In *International Conference on Computer Vision (ICCV)* (2023). 2

[LH16] LOSHCHILOV I., HUTTER F.: Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983* (2016). 5

[LH17] LIU L., HODGINS J.: Learning to schedule control fragments for Physics-Based characters using deep Q-Learning. *ACM Trans. Graph. 36*, 3 (June 2017), 1–14. doi:10.1145/3083723. 2

[LJH*20] LIU L., JIANG H., HE P., CHEN W., LIU X., GAO J., HAN J.: On the variance of the adaptive learning rate and beyond. In *Proceedings of the Eighth International Conference on Learning Representations (ICLR 2020)* (April 2020). 5

[LKL10] LEE Y., KIM S., LEE J.: Data-driven biped control. In *ACM SIGGRAPH 2010 Papers* (New York, NY, USA, 2010), SIGGRAPH '10, Association for Computing Machinery. doi:10.1145/1833349.1781155. 4

[LKP*23] LEE S., KANG T., PARK J., LEE J., WON J.: Same: Skeleton-agnostic motion embedding for character animation. In *SIGGRAPH Asia 2023 Conference Papers* (New York, NY, USA, 2023), Association for Computing Machinery. doi:10.1145/3610548.3618206. 2

[LSCC20] LUO Y.-S., SOESENO J. H., CHEN T. P.-C., CHEN W.-C.: CARL: controllable agent with reinforcement learning for quadruped locomotion. *ACM Trans. Graph. 39*, 4 (Aug. 2020), 38:1–38:10. doi:10.1145/3386569.3392433. 2

[LWH*12] LEVINE S., WANG J. M., HARAUX A., POPOVIĆ Z., KOLTUN V.: Continuous character control with low-dimensional embeddings. *ACM Trans. Graph. 31*, 4 (jul 2012). doi:10.1145/2185520.2185524. 2

[LZCVDP20] LING H. Y., ZINNO F., CHENG G., VAN DE PANNE M.: Character controllers using motion vaes. *ACM Trans. Graph. 39*, 4 (aug 2020). doi:10.1145/3386569.3392422. 2

[MHG*18] MEREL J., HASENCLEVER L., GALASHOV A., AHUJA A., PHAM V., WAYNE G., TEH Y. W., HEESS N.: Neural probabilistic motor primitives for humanoid control. *arXiv preprint arXiv:1811.11711* (2018). 2

[Mix24] MIXAMO: Mixamo. animated 3d characters for games, film, and more., 2024. URL: https://www.mixamo.com/#/. 5

[MTA*20] MEREL J., TUNYASUVUNAKOOL S., AHUJA A., TASSA Y., HASENCLEVER L., PHAM V., EREZ T., WAYNE G., HEESS N.: Catch & carry: reusable neural controllers for vision-guided whole-body tasks. *ACM Trans. Graph. 39*, 4 (Aug. 2020), 39:1–39:12. doi:10.1145/3386569.3392474. 2

[MWG*21] MAKOVIYCHUK V., WAWRZYNIAK L., GUO Y., LU M., STOREY K., MACKLIN M., HOELLER D., RUDIN N., ALLSHIRE A., HANDA A., ET AL.: Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470* (2021). 6

[PALVdP18] PENG X. B., ABBEEL P., LEVINE S., VAN DE PANNE M.: Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG) 37*, 4 (2018). 2, 4

[PAZA18] PENG X. B., ANDRYCHOWICZ M., ZAREMBA W., ABBEEL P.: Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)* (2018), IEEE, pp. 3803–3810. 8

[PGH*22] PENG X. B., GUO Y., HALPER L., LEVINE S., FIDLER S.: Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Trans. Graph. 41*, 4 (2022). 2, 4, 7

[PMA*21] PENG X. B., MA Z., ABBEEL P., LEVINE S., KANAZAWA A.: AMP: adversarial motion priors for stylized physics-based character control. *ACM Trans. Graph. 40*, 4 (2021). 2

[PRL*19] PARK S., RYU H., LEE S., LEE S., LEE J.: Learning predict-and-simulate policies from unorganized human motion data. *ACM Trans. Graph. 38*, 6 (Nov. 2019), 1–11. doi:10.1145/3355089.3356501. 2, 4

[RBH*21] REMPE D., BIRDAL T., HERTZMANN A., YANG J., SRIDHAR S., GUIBAS L. J.: Humor: 3d human motion model for robust pose estimation. In *International Conference on Computer Vision (ICCV)* (2021). 2

[Rea24] REALLUSION: 3d animation and 2d cartoons made simple., 2024. Motion Packs: studio-mocap-sword-and-shield-stunts, studio-mocap-sword-and-shield-moves, studio-mocap-hero-motion, studio-mocap-girl-dance, studio-mocap-evolution-of-dance-vol-1, studio-mocap-evolution-of-dance-vol-2, iclone-motion-pack—street-dance-locking. URL: https://actorcore.reallusion.com/. 5, 9

[RKH*21] RADFORD A., KIM J. W., HALLACY C., RAMESH A., GOH G., AGARWAL S., SASTRY G., ASKELL A., MISHKIN P., CLARK J., KRUEGER G., SUTSKEVER I.: Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning* (2021), Meila M., Zhang T., (Eds.), vol. 139 of *Proceedings of Machine Learning Research*, PMLR, pp. 8748–8763. 2

[RLL*23] RAAB S., LEIBOVITCH I., LI P., ABERMAN K., SORKINE-HORNUNG O., COHEN-OR D.: Modi: Unconditional motion synthesis from diverse data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2023), pp. 13873–13883. 2

[RLT*24] RAAB S., LEIBOVITCH I., TEVET G., ARAR M., BERMANO A. H., COHEN-OR D.: Single motion diffusion. In *The Twelfth International Conference on Learning Representations (ICLR)* (2024). URL: https://openreview.net/pdf?id=DrhZneqz4n. 2

[SB18] SUTTON R. S., BARTO A. G.: *Reinforcement learning: An introduction*. MIT press, 2018. 2, 4

[SKB21] SCHUMACHER C., KNOOP E., BÄCHER M.: A versatile inverse kinematics formulation for retargeting motions onto robots with kinematic loops. *IEEE Robotics and Automation Letters 6*, 2 (2021), 943–950. doi:10.1109/LRA.2021.3056030. 5

[SKL07] SOK K. W., KIM M., LEE J.: Simulating biped behaviors from human motion data. *ACM Trans. Graph. 26*, 3 (jul 2007), 107–es. URL: https://doi.org/10.1145/1276377.1276511, doi:10.1145/1276377.1276511. 2

[SMK22] STARKE S., MASON I., KOMURA T.: DeepPhase: periodic autoencoders for learning motion phase manifolds. *ACM Trans. Graph. 41*, 4 (July 2022), 1–13. doi:10.1145/3528223.3530178. 2

[STKB23] SHAFIR Y., TEVET G., KAPON R., BERMANO A. H.: Human motion diffusion as a generative prior. *arXiv preprint arXiv:2303.01418* (2023). 2

[SWD*17] SCHULMAN J., WOLSKI F., DHARIWAL P., RADFORD A., KLIMOV O.: Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017). 6

[SZKZ20] STARKE S., ZHAO Y., KOMURA T., ZAMAN K.: Local motion phases for learning multi-contact character movements. *ACM Trans. Graph. 39*, 4 (aug 2020). URL: https://doi.org/10.1145/3386569.3392450, doi:10.1145/3386569.3392450. 2

[TGH*22] TEVET G., GORDON B., HERTZ A., BERMANO A. H., COHEN-OR D.: Motionclip: Exposing human motion generation to clip space. In *ECCV 2022* (2022), Springer, pp. 358–374. 2

[TKG*23] TESSLER C., KASTEN Y., GUO Y., MANNOR S., CHECHIK G., PENG X. B.: Calm: Conditional adversarial latent models for directable virtual characters. In *ACM SIGGRAPH 2023 Conference Proceedings* (New York, NY, USA, 2023), SIGGRAPH '23, Association for Computing Machinery. doi:10.1145/3588432.3591541. 2, 4, 6, 7

[TRG*23] TEVET G., RAAB S., GORDON B., SHAFIR Y., COHEN-OR D., BERMANO A. H.: Human motion diffusion model. In *The Eleventh International Conference on Learning Representations* (2023). URL: https://openreview.net/forum?id=SJ1kSyO2jwu. 2, 9

[WGH20] WON J., GOPINATH D., HODGINS J.: A scalable approach to control diverse behaviors for physically simulated characters. *ACM Trans. Graph. 39*, 4 (Aug. 2020), 33:1–33:12. doi:10.1145/3386569.3392381. 2

[WGH22] WON J., GOPINATH D., HODGINS J.: Physics-based character controllers using conditional VAEs. *ACM Trans. Graph. 41*, 4 (July 2022), 1–12. doi:10.1145/3528223.3530067. 2

[WGSF20] WANG T., GUO Y., SHUGRINA M., FIDLER S.: Unicon: Universal neural controller for physics-based character motion, 2020. arXiv:2011.15119. 2, 7

[WPP14] WAMPLER K., POPOVIĆ Z., POPOVIĆ J.: Generalizing locomotion style to new animals with inverse optimal regression. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 1–11. 2

[XSZK23] XU P., SHANG X., ZORDAN V., KARAMOUZAS I.: Composite motion learning with task control. *ACM Transactions on Graphics 42*, 4 (2023). doi:10.1145/3592447. 2

[YSCL22] YAO H., SONG Z., CHEN B., LIU L.: ControlVAE: Model-Based learning of generative controllers for Physics-Based characters. *ACM Trans. Graph. 41*, 6 (Nov. 2022), 1–16. doi:10.1145/3550454.3555434. 2

[YSI*23] YUAN Y., SONG J., IQBAL U., VAHDAT A., KAUTZ J.: Physdiff: Physics-guided human motion diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2023). 2

[ZBL*19] ZHOU Y., BARNES C., LU J., YANG J., LI H.: On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 5745–5753. 3

[ZQW20] ZHAO W., QUERALTA J. P., WESTERLUND T.: Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)* (2020), IEEE, pp. 737–744. 8

[ZZLH23] ZHU Q., ZHANG H., LAN M., HAN L.: Neural categorical priors for physics-based character control. *ACM Trans. Graph. 42*, 6 (dec 2023). doi:10.1145/3618397. 2