Lightweight Policy-Space Control for Physics-Based Characters

ANONYMOUS AUTHOR(S)
SUBMISSION ID: PAPERS_1462

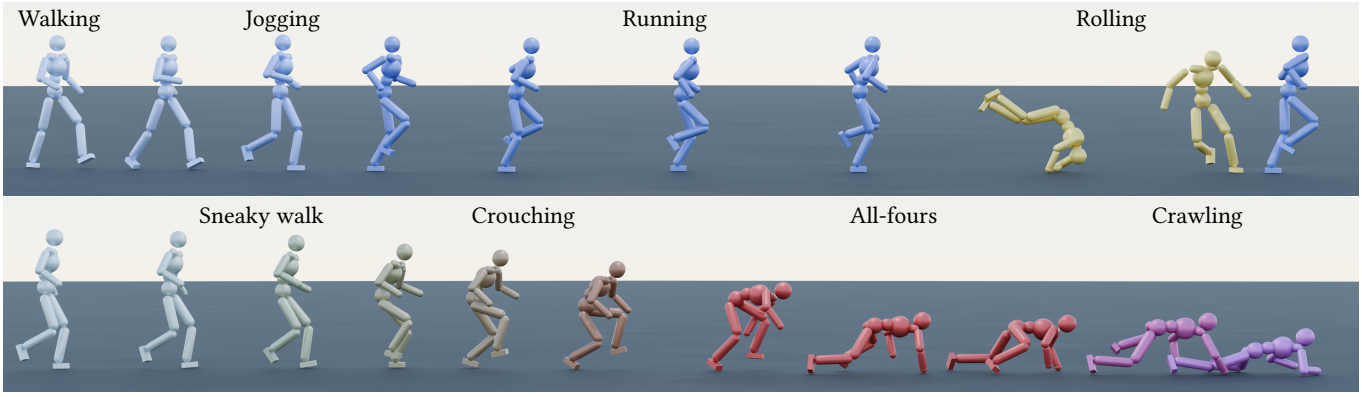

Fig. 1. Our lightweight control strategy can offer a full move-set of physics-based simulations with a continuum of real-time variations. Policies with similar state spaces can be linearly interpolated, while the systems still allow for animating episodic motions as Rolling. These motions are controlled by a gamepad with minimal logic

Physics-based character animation often relies on large policy networks or complex frameworks, which demand significant computational resources. These systems typically require partial or complete retraining to integrate new motions, limiting their flexibility during prototyping. We propose a novel approach that uses a lightweight network and dynamically changes weights, enabling new behaviors with minimal complexity. Our technique animates characters through simple operations such as policy-space linear interpolation, policy switch, and graph search. Central to our method is a weight-space regularization strategy, which organizes pre-trained policies to allow seamless transitions between motions and diverse animation styles. This approach supports real-time control of character animations while maintaining low computational overhead, preserving the benefits of physics-based controllers. Moreover, new policies can be incorporated into the system without retraining existing components. Results demonstrate that interpolated policies produce smoother transitions than direct policy switch while maintaining high motion quality comparable to state-of-the-art imitation-based frameworks at a significantly lower computational cost. An interesting finding is an equivalence between our interpolated policies and direct action interpolation, opening research paths to clarify how the policies encode actions. The possibility of interpolating single layers resulting in motion variations of the internal pattern of the motion, suggests that ours is a powerful representation that can open the path to improved control over animation variability. We showcase an application where gamepad input drives desired motions with minimal logic, highlighting the system's real-time responsiveness and simplicity.

ACM 0730-0301/2025/1-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

CCS Concepts: • **Computing methodologies** → **Physical simulation**; **Motion processing**.

Additional Key Words and Phrases: Control Policies, Deep Reinforcement Learning, Interactive Animation, Weight-space

ACM Reference Format:
Anonymous Author(s). 2025. Lightweight Policy-Space Control for Physics-Based Characters. *ACM Trans. Graph.* 1, 1 (January 2025), 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Physics-based character animation has become a cornerstone of modern computer graphics, enabling the creation of highly realistic and physically plausible motion. This type of animation allows for interaction with the environment calculating forces and contacts in a physics simulator. A deep reinforcement learning policy is usually capable of learning complex motor skill. However, learning new skills often leads to catastrophic forgetting of the previous ones. Researchers have been working on methods to learn articulated sequences of skills, as the simple interpolation of action produced by different policies does not always keep the realism of the trained motion. Many methods resort to a two-level structure where a high-level model composes the effects of the low-level policies to achieve more complex skills. This usually entails having to retrain, partially or totally, the model when a new motion is added to the set. These methods often rely on training latent encoding of the motion set or other generative modeling setups that require heavy computation resources.

We propose a novel control technique for efficiently animating characters using compact policy networks. Our approach emphasizes simplicity and computational efficiency while preserving the robustness and adaptability of physics-based systems. By leveraging a lightweight network our method generates diverse motion

styles with minimal computational overhead. A key innovation is our weight-space regularization strategy, which organizes a set of pre-trained, policies into a structured framework. This allows seamless transitions between motion types, reduces computational costs, and enables the addition of new policies without retraining existing components.

Our method retains the physical plausibility and robustness of physics-based animation, ensures smoother motion transitions compared to naive policy switching, and remains compatible with resource-limited systems, making it ideal for applications like video games and virtual reality. Experimental results demonstrate its effectiveness in animating characters with minimal logic, such as gamepad control, while achieving visually equivalent motions to those produced by action interpolation. Our representation, however, allows us to manipulate the motion generation by changing the weights, introducing variability to the internal motion patterns modifying how the actions are generated.

2 RELATED WORK

Our work has been inspired by the recent development in the field of physics-based animation. We identify 3 broad areas of overlap: The composition of different motor skills from policy-based controllers, smooth transition among different motions, and directable control of the animations.

2.1 Composing Motor Skills

Motion imitation policies for character animation have traditionally been limited to reproducing a single complex motor skill. Even with large neural networks, catastrophic forgetting has hindered the ability to learn a large-scale motion dataset with a single policy. The approaches proposed by [Peng et al. 2018] introduce a one-hot conditional policy for selecting which skill to imitate, or employ the value function to sample an appropriate policy from the current state among a set of pre-trained policies. Goal inputs allow to learn high-level task.

Ho and Ermon [2016] employs Generative Adversarial Imitation Learning (GAIL) to achieve high-quality, realistic motor skills by training a generator to produce actions that mimic expert demonstrations, refined through an adversarial training process.

In the same vein, Peng et al. [2021] explores Adversarial Motion Priors (AMP) for learning motion skills from unstructured datasets by dynamically selecting and interpolating motions without explicit clip selection or sequencing. AMP effectively mitigates catastrophic forgetting and enables the composition of various motor skills from diverse datasets. Here a discriminator loss is combined with a goal loss for laveraging imitation and high-level tasks.

Peng et al. [2022] developed ASE (Adversarial Skills Embedding) to separate the motion imitation task from the goal task. First, it learns a low-level imitation policy on the full dataset alongside a latent state representation. Novel goal-based tasks can be trained using a high-level controller that reuses the action set learned by the low-level policy through the latent encoding. This method confines the set of poses used for solving the task to a set of highly realistic poses, with the drawback of the pose set being discrete.

2.2 Smooth transitions in a Policy-Based Setting

Motion blending is a cornerstone of creating smooth and realistic character animations, whether in classical kinematic settings or modern physics-based frameworks. In kinematic-based animation, techniques such as barycentric interpolation, radial basis functions, k-nearest neighbors, and inverse blending optimization have been extensively analyzed by Feng et al. [2012]. Their study provides a detailed comparison of these methods, highlighting their respective strengths and limitations in achieving fluid transitions.

Modern advancements have leveraged machine learning and physics-based approaches to push the boundaries of motion blending. For instance, Juravsky et al. [2022] introduced a system that integrates natural language processing (NLP) with physics-based character control. It enables users to command character motions using natural language, bridging the gap between intuitive user input and complex motion generation.

Tessler et al. [2024] presented a unified system for physics-based control that employs masked motion in-painting. This approach generates diverse motions from partial descriptions, such as keyframes or text commands, offering greater flexibility and creative possibilities in animation design.

Further progress was demonstrated by Luo et al. [2024a], who developed a foundational model for physics-based control. Their framework learns a universal motion representation from large, unstructured datasets and employs hierarchical reinforcement learning to produce stable, diverse human motions adaptable to various tasks.

Despite these advancements, challenges persist in achieving seamless integration and realism, particularly in dynamic and interactive environments.

2.3 Directable Control of Physics-Based Animation

Achieving directable control in physics-based animation is a complex challenge that has seen remarkable advancements. Bergamin et al. [2019] developed a method that employs motion-matching [Clavet 2016] examples to train deep reinforcement learning policies for controllable locomotion of physically simulated characters in real-time, demonstrating its feasibility in dynamic environments. Wang et al. [2020] developed universal neural controller capable of mastering thousands of motions with different styles, supporting real-time interactive applications. Dou et al. [2023] expands the for from Peng et al. [2022] to control real-time animation by dividing skill motions into homogeneous subsets for training, offering direct control over character skills through low-level conditional models. This enables seamless and responsive character control, benefitting interactive animations. The research by [Tessler et al. 2023] enables user-controlled virtual characters by leveraging adversarial training and imitation learning. This model captures the complexity and diversity of human motion, allowing user-controlled characters to be both directable and intuitive. The approach jointly learns a control policy and a motion encoder, enabling real-time behavior control and motion conditioning for high-level task performance. The work by Luo et al. [2024b] presents a comprehensive motion representation that covers a wide range of motor skills, using an encoder-decoder structure to distill skills from a large dataset. This approach achieves high coverage and expressiveness, making it effective for

both generative tasks and motion tracking. Another significant advancement is the work by Ren et al. [2023] uses diffusion-based human motion models to generate instruction-driven animations of physics-based characters. This framework captures complex relationships between high-level human instructions and character motions, achieving state-of-the-art results in various tasks.

2.4 This work

Our approach differs from other works by viewing the policy as a dynamic object that can change its weights to change its behavior. We train our policies using a modified implementation of Peng et al. [2021] and regularize the weights to preserve similarity for leveraging simple operations in the weight-space. Linear piecewise interpolation allows us to navigate the regularization graph to obtain smooth behavior changes. The main contributions of this work are

- A regularization strategy to preserve similarity of the weights during policy training
- A novel policy network design allowing dynamic weight manipulation to adapt and control character behavior.
- The use of simple operations, such as interpolation, policy switch, and graph search, to control character behavior with minimal gamepad logic.

3 METHODOLOGY

In this section we break down the methodology of our apporach, by describing our weight regularization, training strategy, and gamepad control logic.

3.1 Policy Regularization Strategy

Our approach incorporates heuristics that enable linear interpolation within the weight space of the actor network. To achieve this, we adopt a regularization strategy that preserves policy similarity for similar motions. Specifically, we utilize a policy gradient algorithm, Proximal Policy Optimization (PPO), in combination with a Multi-Layer Perceptron (MLP) with ReLU activations. This architecture supports meaningful policies under linear interpolation, unlike other architectures, such as Gated Recurrent Units (GRUs) [Xu and Karamouzas 2021; Xu et al. 2023].

We build on the Adversarial Motion Prior (AMP) framework [Peng et al. 2021], which jointly trains a discriminator and actor network for motion imitation. To apply the same motion prior across different tasks, we incorporate a task-specific goal by defining appropriate rewards. Initially, we train a primary policy, referred to as Main policy, starting from random weights without regularization. Subsequent policies are trained with regularization by applying a Gaussian prior to the network weights, encouraging similarity to the weights of policies trained for related tasks. The overall loss function is given by:

$$\mathcal{L} \equiv c_{\text{Disc}}\mathcal{L}_{\text{Disc}} + c_{\text{Task}}\mathcal{L}_{\text{Task}} + \lambda||w - w_{\text{Reg}}||^2,$$

where w_{Reg} represents the regularization weights, c_{Disc} the importance given to the imitation task, c_{Task} the importance given to the high-level task, and λ the penalization for the weights moving away from the reference. Here $c_{\text{Disc}} + c_{\text{Task}} = 1$ and we set $\lambda = 0.02$.

The regularization strategy follows a hierarchical graph structure, starting from Main policy and branching into derived policies based on state space overlap, as illustrated in Figure 2.

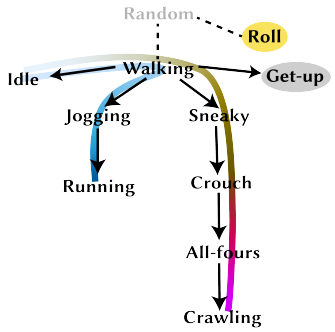

Fig. 2. Graph-based structure for the regularization strategy. Two main branches allow linear piecewise interpolation for Idle to Running, and Idle to Crawling. Random here is a different set of weights for the two deriving policies.

Policies are trained for motion imitation alongside a target localization task. In this task, a target is placed on the x-y plane, and the character is rewarded for moving toward the target and stopping upon reaching it. By maintaining a consistent input structure across policies, linear interpolation becomes feasible while preserving the character's controllability. The reward function can be specialized according to the learned skill, for example, the Idle policy is rewarded for maintaining a stationary stance, regardless of the target location.

3.2 Policy architecture and training

The policies are implemented as a multi-layer perceptron (MLP) with two fully connected hidden layers of sizes [128, 32], using ReLU activations. The input is a 225-dimensional state observation that includes the root-bone height, local body positions, rotations, velocities, angular velocities, and the relative x-y coordinates of the target location with respect to the character's root bone. The output is a 28-dimensional action vector for the Humanoid character, as described in [Peng et al. 2021]. This output represents the mean of a multivariate Gaussian distribution, with the standard deviation fixed across all dimensions. The discriminator network, also modeled as an MLP with hidden layers of sizes [1024, 512], is used during training to evaluate the imitation quality and during replay to automate the termination of episodic policies, such as Get-up or Roll motions.

To effectively learn motions it is found to be beneficial dividing the training into phases with different combinations of the parameters c_{Disc} and c_{Task}. At the beginning of the training, it is convenient to set $c_{\text{Disc}} = 1$ and consequently $c_{\text{Task}} = 0$, to allow the policy to learn the motion of quality faithful to the dataset. In this phase, is important to tune the gradient penalty to give the possibility for complex motions to be learned. Once the motion is learned, c_{Task} can be increased to allow the high-level task to be learned. In the final phase, the policy is trained on a wavy terrain to improve its robustness to falling.

3.3 Gamepad Control

Character control is achieved by placing the x-y target in the desired direction relative to the chatacter, where the motion speed is determined by the target's distance. Switching between policies leads to abrupt behavior changes, which can be leveraged to perform specific motion skills. For instance, rolling or getting up from the floor can be executed for a fixed number of frames upon receiving an input command. The discriminator trained for the Walk motion evaluates the character's current pose and determines when it aligns with the distribution of walking states, allowing seamless transitions back to locomotion. The threshold for the discriminator is chosen empirically to be triggered from sufficiently in-distribution states.

Our regularization approach enables linear interpolation to smoothly transition between policies. This ensures the character remains "in-distribution" during transitions by leveraging the robustness of physics-based policies. Linear piecewise interpolation along the regularization graph achieves smooth, controllable behavior, avoiding the abrupt changes caused by direct policy switch. Simple policy switch, by contrast, often results in frequent falls and unrealistic motion transitions.

To run our control experiments, we set up a gamepad logic to control the character's behavior using Isaac Sim. We map the controller input from the left stick to control the character's direction and we assign the left and right analog triggers to linear piecewise interpolation paths to control the motion style. Other buttons are programmed to perform occasional policy switch as for Roll and Get-up that are eventually terminated by a threshold on the Walk discriminator.

To ensure real-time performance, all policies are preloaded to eliminate file-loading delays. Policy decisions, linear interpolation, and policy evaluation occur seamlessly between frames without impacting the framerate.

4 RESULTS

The results demonstrate that our control strategy enables smooth transitions between policies, effectively bridging the gap between imitation policies.

Figure 1 illustrates how motion is smoothly interpolated across different styles and how episodic policies are seamlessly integrated into the control framework. The linear piecewise interpolation follows a path on the policy graph shown in Figure 2. During inference, the interpolation of 33K parameters and the evaluation of a single network [128,32] collectively result in approximately 100K multiply-add operations per simulation step.

4.1 Comparison with Action Interpolation

Interestingly, the visual quality of the motion generated by our method is equivalent to that produced by interpolating the actions from the same two networks. During inference, evaluating two networks [128,32] and interpolating 28 action values amount to approximately 64K multiply-add operations per simulation step.

4.2 Comparison with ASE

To compare our method with ASE, we trained a low-level imitation policy using the same dataset employed for our set of regularized policies, excluding motion for the "Roll" animation to prevent accidental triggering during high-level tasks. To ensure a fair comparison, we aimed to provide control over the character's speed and z-level of the root bone, aligning with our selection of motions in our policy graph.

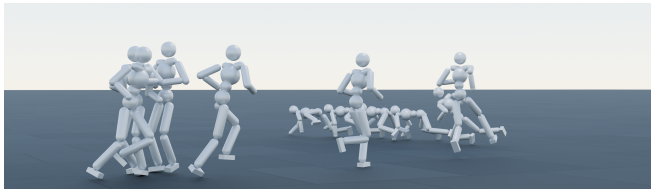

Fig. 3. With the ASE framework we obtain controllable motion of high quality, however, it can only use the actions learned from the low-level controller to perform the high-level task. In this example, the motion is controllable only when the character is standing, from the moment it falls, the character starts imitating crawling motions and becomes indifferent to the control input

We defined an augmented target location high-level task with four goal inputs: the x-coordinate and y-coordinate of the target location, the root-bone speed, and the z-level of the root bone. The character was rewarded for achieving the desired location, speed, and z-level using a reward function of the form:

$$r \equiv r_w e^{-r_s (v - v_{\text{desired}})^2},$$

where v represents the parameter being compared to the desired value v_{desired}, r_s controls the exponential decay scale as the parameter deviates from the desired value, and r_w weights the parameter's contribution to the total reward.

The resulting motion is shown to be controllable for reaching the target location; however, the inputs for speed and z-level were largely ignored by the high-level task. As demonstrated in figure 3 and the supplementary video, the character could walk and run toward the desired direction but eventually fell and started crawling, rendering the location task infeasible due to insufficient poses for crawling. The network largely disregarded the speed and z-level inputs. This experiment highlighted ASE's ability to reproduce high-quality motions using a discrete set of actions from the dataset. However, with a small dataset, our method offers greater motion variety. As a low-level imitation policy guided by high-level goals, our method generates novel actions, effectively overcoming dataset limitations.

During inference, ASE evaluates three networks: the high-level policy [1024,512], a style network [512,256] transforming the latent z, and the low-level policy [1024,1024,512]. Accounting for activations, this amounts to approximately 2.2M multiply-add operations per simulation step, 22 times more than our method and 34 times more than action interpolation. We have trained the low-level policy for 40K epochs and the high-level task for 32K epochs with 4096 parallel environments.

4.3 Interpolation of single layers

The results in the supplementary video show that interpolation of a single hidden layer or the output layer results in motion variations

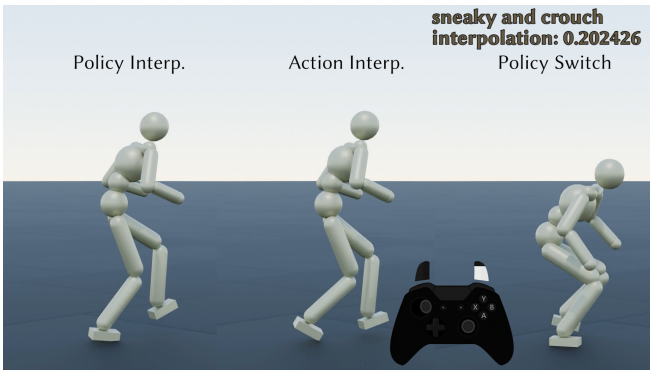

Fig. 4. An illustrative frame from the supplementary video. Our method produces poses equivalent to direct action interpolation, while direct policy switching fails to ensure smooth transitions.

that do not directly translate into an interpolation of the final action. Instead, these variations indicate that weight interpolation affects the internal representations in a way that alters the motion trajectory, producing behaviors that deviate from simple action blending. This suggests that the intermediate layers of the network encode complex, nonlinear transformations that capture the temporal and spatial dynamics of the motion.

The observed phenomenon highlights the role of the network's architecture in shaping the interpolation outcomes. Interpolating weights in hidden layers influences the latent structure of the policy, which can lead to transitions that are not easily predictable from the final action alone. These results underline the potential of weight manipulation as a tool for generating diverse and expressive motion patterns beyond what is achievable through action interpolation.

5 DISCUSSION

Our method demonstrates the effectiveness of weight regularization for preserving similarity during policy training, enabling interpolation in the weight space. As shown in Figures 5 and 6, regularized policies with similar state distributions cluster closely in PCA representation, while non-regularized policies or those with highly dissimilar state distributions are farther apart.

However, regularization can sometimes slow or limit the training of specific skills. Using the AMP algorithm, our regularization is effective when the state spaces of the motions are similar. For instance, training a chain of policies from "Walking" to "Crawling" requires approximately 2,000 epochs with 4,096 environments running in parallel for each regularized training session. However, regularizing a jump motion from idle or another locomotion policy can hinder training, as the RL algorithm focuses on matching the jump's starting and landing states. We attribute this limitation to jumps requiring significantly higher torque than locomotion. Encouraging jumps by rewarding states with higher Z-positions of the root bone could help, yet it may reduce realism, as the policy might adopt non-imitative strategies to achieve the goal.

The supplementary video demonstrates that our method achieves visual equivalence to direct action interpolation. Figure 4 highlights

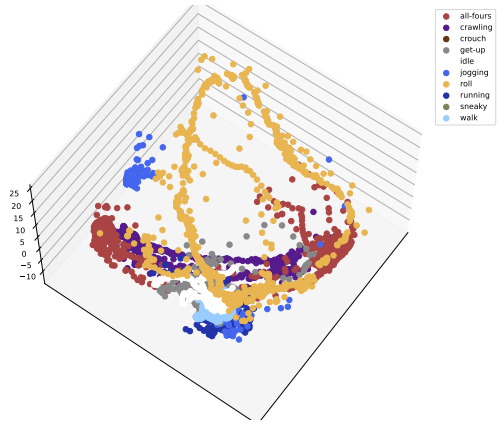

Fig. 5. 3D PCA representation of states visited by the policies. Regularized policies produce realistic transitions for motions close in state space.

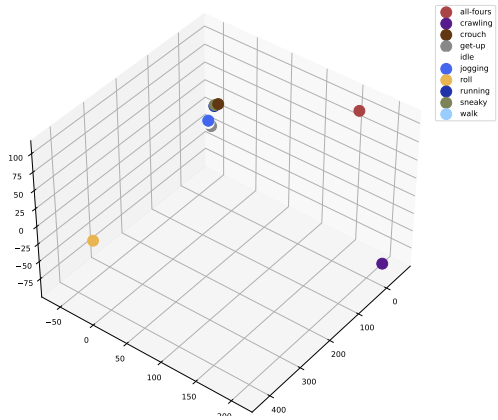

Fig. 6. 3D PCA representation of policy weights. Policies close in this reduced space yield realistic results during interpolation. Comparison with Figure 5 shows the correlation between weight distances and state distribution overlap.

how the same gamepad input produces consistent poses. This behavior may stem from the structure of the policy networks, which consist of linear layers with ReLU activations. While ReLU introduces localized nonlinearity by zeroing out negative values, the interpolation remains piecewise linear within regions where the same set of neurons is active.

This piecewise linearity suggests that interpolation operates in regions where linearity is preserved. Additionally, the learned policy manifold may align so that intermediate states during interpolation remain consistent with valid actions, influenced by PPO training and data constraints. While this highlights the robustness of the network design, further investigation is needed to confirm whether this behavior generalizes across architectures or tasks.

Figure 5 shows the state distribution in PCA-reduced space, where action interpolation creates realistic transitions when end-point policies share similar state distributions. This requirement aligns with

our regularization strategy. Although action interpolation is computationally cheaper and requires less training, our representation contains more information than its output actions.

The weights w of a control policy can be interpreted as a latent motion encoding, capturing information about the training dataset's visited states, gravity's influence, and robustness to perturbations. During evaluation, this information is "collapsed" into a single action, representing the instantaneous policy output. This behavior reflects the realization of a trajectory within a high-dimensional policy manifold. Working at the weight level modifies the internal patterns of the motion, as demonstrated by the interpolation of single layers.

The observed linear characteristics during weight interpolation suggest that the policy manifold has an underlying structure that could support dimensionality reduction. This would enable more efficient and interpretable policy representations. Further analysis is required to understand how this structure generalizes across tasks and architectures and how it can be exploited for practical applications.

6 CONCLUSION

We propose a novel approach to physics-based character animation that has the potential to establish a new paradigm for representing complex human motion in simulation. By representing policies as dynamic objects capable of altering their behavior through weight manipulation, we enable variations across the full motion representation rather than restricting them to contextual actions. Our results demonstrate that interpolating individual layers within the policies generates motion variations, showing that weight manipulation is more expressive than simply blending output motions. This reveals the ability to uncover novel motion trajectories that are inaccessible through traditional action interpolation techniques.

Thus far, this approach has been effectively applied to motions with similar state distributions, consistent with the conditions under which action interpolation achieves realistic results. However, we see promising opportunities to extend this method to a broader range of motions with more diverse state distributions. This presents a chance to explore new policy representations that enhance the effectiveness and interpretability of weight manipulation, transitioning from "black-box" systems to a more explainable and versatile representation of motion.

Future research should focus on developing techniques to generalize this approach to more diverse motion datasets while ensuring smooth transitions between motions with dissimilar dynamics. Incorporating domain knowledge could further refine weight manipulation strategies and improve guidance. Additionally, systematic analysis of how specific weights contribute to motion variations could lead to more precise control over motion style and behavior. Understanding the structure of the policy manifold could also support dimensionality reduction, optimizing storage, computation, and policy interpretation.

Another avenue for exploration is incorporating learning objectives that encourage interpretable latent encodings, fostering a deeper understanding of the relationship between weights, dynamics, and the resulting motion. Finally, integrating these techniques into interactive or real-time systems could unlock applications in animation, robotics, and virtual environments, where adaptability and explainability are critical.

REFERENCES

Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. 2019. DReCon: data-driven responsive control of physics-based characters. *ACM Trans. Graph.* 38, 6, Article 206 (Nov. 2019), 11 pages. https://doi.org/10.1145/3355089.3356536

Simon Clavet. 2016. Motion Matching and The Road to Next-Gen Animation. Identifier: GDC2016Clavet, Addeddate: 2016-10-17, Scanner: Internet Archive HTML5 Uploader 1.6.3.

Zhiyang Dou, Xuelin Chen, Qingnan Fan, Taku Komura, and Wenping Wang. 2023. C·ASE: Learning Conditional Adversarial Skill Embeddings for Physics-based Characters. In *SIGGRAPH Asia 2023 Conference Papers* (Sydney, NSW, Australia) *(SA '23)*. Association for Computing Machinery, New York, NY, USA, Article 2, 11 pages. https://doi.org/10.1145/3610548.3618205

Andrew Feng, Yazhou Huang, Marcelo Kallmann, and Ari Shapiro. 2012. An Analysis of Motion Blending Techniques. In *Motion in Games*. 232–243.

Jonathan Ho and Stefano Ermon. 2016. Generative Adversarial Imitation Learning. *CoRR* abs/1606.03476 (2016). arXiv:1606.03476 http://arxiv.org/abs/1606.03476

Jordan Juravsky, Yunrong Guo, Sanja Fidler, and Xue Bin Peng. 2022. PADL: Language-Directed Physics-Based Character Control. In *SIGGRAPH Asia 2022 Conference Papers (SA '22)*. ACM, 1–9. https://doi.org/10.1145/3550469.3555391

Zhengyi Luo, Jinkun Cao, Josh Merel, Alexander Winkler, Jing Huang, Kris M. Kitani, and Weipeng Xu. 2024a. Universal Humanoid Motion Representations for Physics-Based Control. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=OrOd8PxOO2

Zhengyi Luo, Jinkun Cao, Josh Merel, Alexander Winkler, Jing Huang, Kris M. Kitani, and Weipeng Xu. 2024b. Universal Humanoid Motion Representations for Physics-Based Control. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=OrOd8PxOO2

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. 2018. DeepMimic: example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graph.* 37, 4, Article 143 (July 2018), 14 pages. https://doi.org/10.1145/3197517.3201311

Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. 2022. ASE: large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Trans. Graph.* 41, 4, Article 94 (July 2022), 17 pages. https://doi.org/10.1145/3528223.3530110

Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. 2021. AMP: adversarial motion priors for stylized physics-based character control. *ACM Trans. Graph.* 40, 4, Article 144 (July 2021), 20 pages. https://doi.org/10.1145/3450626.3459670

Jiawei Ren, Mingyuan Zhang, Cunjun Yu, Xiao Ma, Liang Pan, and Ziwei Liu. 2023. InsActor: instruction-driven physics-based characters. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) *(NIPS '23)*. Curran Associates Inc., Red Hook, NY, USA, Article 2618, 13 pages.

Chen Tessler, Yunrong Guo, Ofir Nabati, Gal Chechik, and Xue Bin Peng. 2024. MaskedMimic: Unified Physics-Based Character Control Through Masked Motion Inpainting. *ACM Trans. Graph.* 43, 6, Article 209 (Nov. 2024), 21 pages. https://doi.org/10.1145/3687951

Chen Tessler, Yoni Kasten, Yunrong Guo, Shie Mannor, Gal Chechik, and Xue Bin Peng. 2023. CALM: Conditional Adversarial Latent Modelsnbsp; for Directable Virtual Characters. In *ACM SIGGRAPH 2023 Conference Proceedings* (Los Angeles, CA, USA) *(SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, Article 37, 9 pages. https://doi.org/10.1145/3588432.3591541

Tingwu Wang, Yunrong Guo, Maria Shugrina, and Sanja Fidler. 2020. UniCon: Universal Neural Controller For Physics-based Character Motion. arXiv:2011.15119 [cs.GR]

Pei Xu and Ioannis Karamouzas. 2021. A GAN-Like Approach for Physics-Based Imitation Learning and Interactive Character Control. *Proc. ACM Comput. Graph. Interact. Tech.* 4, 3, Article 44 (Sept. 2021), 22 pages. https://doi.org/10.1145/3480148

Pei Xu, Xiumin Shang, Victor Zordan, and Ioannis Karamouzas. 2023. Composite Motion Learning with Task Control. *ACM Trans. Graph.* 42, 4, Article 93 (July 2023), 16 pages. https://doi.org/10.1145/3592447