SUPPLEMENTARY MATERIAL FOR "UNIVERSAL HUMAN MOTION LATENT SPACE FOR PHYSICS-BASED CONTROL"

Zhengyi Luo[1,2] Jinkun Cao[2] Josh Merel[1] Alexander Winkler[1] Jing Huang[1]
Kris Kitani[1,2] * Weipeng Xu[1] *

[1]Reality Labs Research, Meta; [2]Carnegie Mellon University
https://zhengyiluo.github.io/PULSE/

A INTRODUCTION

In this document, we include additional details about our method that are omitted from the main paper due to the page limit. In Sec.B, we include additional details about PHC+ and our modifications made to imitate all motion from a large-scale dataset. In Sec.C, we include additional details about our method, PULSE, such as architecture, training details, and downstream task configurations *etc*.**All code and models will be released for research purposes.**

Extensive qualitative results are provided on the project page as well as in the supplementary zip (the zipped version is of lower video resolution to fit the upload size). As motion is best seen in videos, we highly encourage the readers to view them to better understand the capabilities of our method. Specifically, we evaluate motion imitation and fail-state recovery capabilities for PHC+ and PULSE after online distillation and show that PULSE can largely retain the abilities of PHC+. Then, we show long-formed motion generation result sampling from the PULSE' s prior and decoder. Sampling from PULSE, we can generate long-term, diverse and human-like motions, and we can vary the variance of the input noise to control the behaviors of random generation. We also compare with SOTA kinematics-based method, HuMoR, and show that while HuMoR can generate unnatural motions, ours are regulated by the laws of physics and remain plausible. Compared to SOTA physics-based latent space(ASE and CALM), our random generation appears more diverse. Finally, we show visualization for downstream tasks for both generative and estimation/tracking tasks and compare them with SOTA methods.

*Equal Advising.

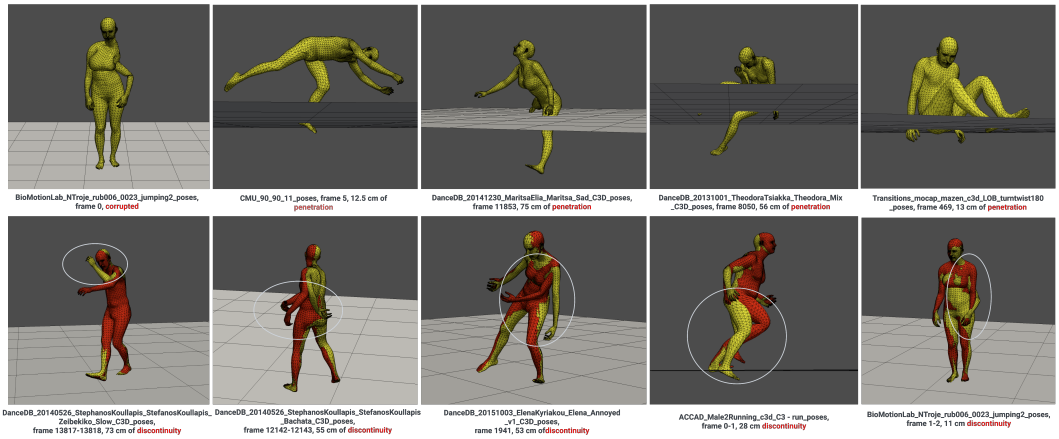

Figure 1: Visualization of issues in the AMASS dataset. Here we show sequences with corrupted poses, large penetration, and discontinuity. In the second row, the red and yellow mesh are 1 frame apart in 120Hz MoCap.

Table 1: Hyperparameters for PHC+ and Pulse. σ: fixed variance for policy. γ: discount factor. ϵ: clip range for PPO. α: coefficient for $\mathcal{L}_{\text{regu}}$. β: coefficient for $\mathcal{L}_{\text{KL}}$.

Method	Batch Size	Learning Rate	σ	γ	ϵ	w_{jp}	w_{jr}	w_{jv}	$w_{\text{j}\omega}$	# of samples
PHC+	3072	2×10^{-5}	0.05	0.99	0.2	0.5	0.3	0.1	0.1	$\sim 10^{10}$
	Batch Size	Learning Rate	α	β	Latent size	# of samples				
PULSE	3072	5×10^{-4}	0.005	$0.01 \rightarrow 0.001$	32	$\sim 10^{9}$				

B Details about PHC+

B.1 Data Cleaning

We perform a failure case analysis and identified two main sources of imitation failure. First, we have dynamic motion, such as cartwheeling and consecutive back flips. Another, often overlooked, is that MoCap sequences can still have a large discontinuity and penetration due to failures in the MoCap optimization procedure or the fitting process (Loper et al., 2014). After filtering out human-object interaction data following UHC (Luo et al., 2021), we found additional corrupted sequences in PHC's training data that have a large discontinuity or penetration. In Fig.1, we visualize some of the sequences we have identified and removed from the training data. Since we use the random state initialization proposed by DeepMimic (Peng et al., 2018), sampling frames that have large penetration could lead to the humanoid "flying off" from the ground as the physics simulation applies a large ground reactionary force. Naively adjusting the height of the sequence based on penetration could lead to floating sequences or discontinuity. Frames that have large discontinuities could lead to imitation failure or humanoid learning bad behavior to anticipate large jumps between frames. We remove these sequences and obtain 11313 training and 138 testing motion suitable for motion imitation training and testing, which we will release with the code and models.

B.2 Action and Rewards

Our action space, state, and rewards follow the specifications of the PHC paper. Specifically, the action $\boldsymbol{a}_t$ specifies the target for the proportional derivative (PD) controller on each of the 69 actuators. The target joint is set to $\boldsymbol{q}_t^d = \boldsymbol{a}_t$ and the torque applied at each joint is $\boldsymbol{\tau}^i = \boldsymbol{k}^p \circ (\boldsymbol{a}_t - \boldsymbol{q}_t) - \boldsymbol{k}^d \circ \dot{\boldsymbol{q}}_t$. Joint torques are capped at 500 N-m. For the motion tracking reward, we use:

$$\begin{aligned} r_t &= 0.5 r_t^{\text{g-imitation}} + 0.5 r_t^{\text{amp}} + r_t^{\text{energy}}, \\ r_t^{\text{g-imitation}} &= w_{\text{jp}} e^{-100 \|\hat{\boldsymbol{p}}_t - \boldsymbol{p}_t\|} + w_{\text{jr}} e^{-10 \|\hat{\boldsymbol{q}}_t \ominus \boldsymbol{q}_t\|} + w_{\text{jv}} e^{-0.1 \|\hat{\boldsymbol{v}}_t - \boldsymbol{v}_t\|} + w_{\text{j}\omega} e^{-0.1 \|\hat{\boldsymbol{\omega}}_t - \boldsymbol{\omega}_t\|}, \end{aligned} \tag{1}$$

Table 3: Ablations on training PULSE from scratch using RL (no distillation).

	AMASS-Train*					AMASS-Test*				
Distill	Succ ↑	$E_{\text{g-mpjpe}}$ ↓	E_{mpjpe} ↓	E_{acc} ↓	E_{vel} ↓	Succ ↑	$E_{\text{g-mpjpe}}$ ↓	E_{mpjpe} ↓	E_{acc} ↓	E_{vel} ↓
✗	72.0%	76.7	52.8	3.5	8.0	32.6%	98.4	79.4	9.9	16.2
✓	99.8 %	39.2	35.0	3.1	5.2	97.1%	54.1	43.5	7.0	10.3

where $r_t^{\text{g-imitation}}$ is the motion imitation reward, r_t^{amp} is the discriminator reward, and r_t^{energy} an energy penalty. $r_t^{\text{g-imitation}}$ measures the difference between the translation, rotation, linear velocity, and angular velocity of the 23 rigid bodies in the humanoid. r_t^{amp}. r_t^{amp} is the Adversarial Motion Prior (AMP) reward (Peng et al., 2021), provided by a discriminator trained on the AMASS dataset. The energy penalty r_t^{energy} is $-0.0005 \cdot \sum_{j \in \text{ joints}} |\boldsymbol{\mu_j} \boldsymbol{\omega_j}|^2$ where $\boldsymbol{\mu_j}$ and $\boldsymbol{\omega_j}$ correspond to the joint torque and the joint angular velocity, respectively. The energy penalty Fu et al. (2022) regulates the policy and prevents high-frequency jitter.

B.3 Model Architecture and Ablations

All primitives and composers in PHC + are a 6 layer MLP with units [2048, 1536, 1024, 1024, 512, 512] and SiLU activation. We find that changing the activation from ReLU (Fukushima, 1975; Nair & Hinton, 2010) to SiLU (Hendrycks & Gimpel, 2016) provides a non-trivial boost in tracking performance. Combining with larger networks (from 3 layer MLP to 6 layer), we use only three primitives to learn fail-state recovery and achieve a success rate of 100%. To study the effect of the new activation function and the progressive training procedure, we perform ablation studies on the training of a single primitive $\boldsymbol{\mathcal{P}}$ (not the full PHC + policy) using the proposed changes.

Each primitive is trained for 3×10^9 samples. From Table 2, we can see that comparing Row (R1) and R3, the new progressive training procedure improves the success rate by a large amount, showing that $\boldsymbol{\mathcal{P}}$'s capacity is not fully utilized if $\hat{\boldsymbol{Q}}_{\text{hard}}$ is not formed and updated during each $\boldsymbol{\mathcal{P}}$'s training. When comparing R2 and R3, we can see that changing the activation function from ReLU to SiLU improves the tracking performance and improves E_{mpjpe}. Table.1 reports the hyperparameters we used for training.

Table 2: Ablations on PHC+'s primitive $\boldsymbol{\mathcal{P}}$ training. Progressive: refers to whether $\hat{\boldsymbol{Q}}_{\text{hard}}$ is updated during the primitive training (rather than waiting until convergence and initialize a new primitive).

		AMASS-Test				
Activation	Progressive	Succ ↑	$E_{\text{g-mpjpe}}$ ↓	E_{mpjpe} ↓	E_{acc} ↓	E_{vel} ↓
SiLU	✗	92.0%	43.0	29.2	6.7	8.9
ReLU	✓	97.8%	44.4	32.8	6.9	9.1
SiLU	✓	**98.5%**	**39.0**	**28.1**	**6.7**	**8.5**

C Details about PULSE

C.1 Training Procedure

We train π_{PULSE} using the training procedure we used to train a primitive $\boldsymbol{\mathcal{P}}^{(0)}$ in PHC+, where we progressively form $\hat{\boldsymbol{Q}}_{\text{hard}}$ while training the policy. Since π_{PULSE} and $\pi_{\text{PHC+}}$ share the same state and action space, we query $\pi_{\text{PHC+}}$ at training time to perform online distillation. We anneal the coefficient β of $\mathcal{L}_{\text{KL}}$ from 0.01 to 0.001 starting from 2.5×10^9 to 5×10^9 samples. Afterward, β remains the same. We report our hyperparameters for training π_{PULSE} in Table. 1.

C.2 Comparison to Training Scratch without Distillation

One of our main contributions for PULSE is the using online distillation to learn π_{PULSE}, where the latent space uses knowledge distilled from a trained imitator, PHC+. While prior work like MCP (Peng et al., 2019) demonstrated the possibility of training such a policy from scratch (using RL without distillation), we find that using the variational information bottleneck together with the imitation objective creates instability during training. We hypothesize that random sampling for the

variational bottleneck together with random sampling for RL leads to noisy gradients. In Table 3, we report the result of motion imitation from training from scratch. We can see that the training using RL does not converge to a good imitation policy after training for more than 1×10^{10} samples.

C.3 Comparison to Other Latent Formulation (VQ-VAE, Spherical)

In our earlier experiments, we studied other forms of latent space such as a spherical latent space, similar to ASE (Peng et al., 2022), or a vector quantized latent space, similar to NCP (**?**). For spherical embedding, we use the same encoder-decoder structure as in PULSE and use a 32-dimensional latent space normalized to the unit sphere. For a vector-quantized motion representation, we follow Liu et al. (2021); Van Den Oord et al. (2017) and use a 64-dimensional latent space divided into 8 partitions, using a dictionary size of 64. Dividing the latent space into partitions increases the representation power combinatorially (Liu et al., 2021) and is more effective than a larger dictionary size. Although through distillation, each of these representations could reach a high imitation success rate and MPJPE (spherical: 100% Succ and 28.1 $E_{\text{g-mpjpe}}$, VQ: 99.8 % Succ and 36.5 $E_{\text{g-mpjpe}}$), both lose the ability to serve as a generative model: random samples from the latent space do not generate coherent motion. In NPC, an additional prior needs to be learned. The quantized latent space also introduces artifacts, such as high-frequency jitter, since the network is switching between discrete codes. We visualize this artifact in our `supplement site`'s last section.

C.4 Downstream Tasks

Each generative downstream task policy π_{task} is a three-layer MLP with units [2048, 1024, 512]. For VR controller tracking, we use a six-layer MLP of units [2048, 1536, 1024, 1024, 512, 512]. The value function has the same architecture as the policy. All tasks are optimized using PPO. For simpler tasks (speed, reach, strike), we train for $\sim 2 \times 10^9$ samples. For the complex terrain traversal task, the policy converges after $\sim 1 \times 10^{10}$ samples. The strike, speed, and reach tasks follow the definition in ASE (Peng et al., 2022), while the following trajectory task follows PACER Rempe et al. (2023). VR controller tracking task follows QuestSim Winkler et al. (2022).

Speed. For training the x-direction speed task, the random speed target is sampled between 0 m/s $\sim$ 5m/s (the maximum target speed for running in AMASS is around 5m/s). The goal state is defined as $\boldsymbol{s}_t^{\text{g-speed}} \triangleq (d_t, v_t))$ where d_t is the target direction and v_t is the linear velocity the policy should achieve at timestep t. The reward is defined as $r_{\text{speed}} = \text{abs}(v_t - \boldsymbol{v}_t{}^0)$ where $\boldsymbol{v}_t{}^0$ is the humanoid's root velocity.

Strike. For strike, since we do not have a sword, we substitute it with "strike with hands". The objective is to knock over the target object and is terminated if any body part other than the right hand makes contact with the target. The goal state $\boldsymbol{s}_t^{\text{g-strike}} \triangleq (\boldsymbol{x}_t, \dot{\boldsymbol{x}}_t))$ contains the position and orientation $\boldsymbol{x}_t$ as well as the linear and angular velocity $\dot{\boldsymbol{x}}_t)$ of the target object in the agent frame. The reward is $r_{\text{strike}} = 1 - \boldsymbol{u}^{\text{up}} \cdot \boldsymbol{u}_t$ where $\boldsymbol{u}^{\text{up}}$ is the global up vector and $\boldsymbol{u}_t$ is the target's up vector.

Reach. For the reach task, a 3D point $\boldsymbol{c}_t$ is sampled from a 2-meter box centered at (0, 0, 1), and the goal state is $\boldsymbol{s}_t^{\text{g-reach}} \triangleq (\boldsymbol{c}_t))$. The reward for reaching is the difference between the humanoid's right hand and the desired point position $r_{\text{reach}} = \exp(-5\|\boldsymbol{p}_t{}^{\text{right hand}} - \boldsymbol{c}_t\|_2^2)$.

Trajectory Following on Complex Terrains . The humanoid trajectory following on complex terrain task, used in PACER (Rempe et al., 2023), involves controlling a humanoid to follow random trajectories through stairs, slopes, uneven surfaces (Rudin et al., 2021), and to avoid obstacles. We follow the setup in P and train policy $\pi_{\text{task}}(\boldsymbol{z}_t|\boldsymbol{s}_t^{\text{p}}, \boldsymbol{s}_t^{\text{g-terrain}})$, $\boldsymbol{s}_t^{\text{g-terrain}} \triangleq (\boldsymbol{o}_t, \boldsymbol{\tau}_{1:T})$ where $\boldsymbol{o}_t$ represents the height map of the humanoid's surrounding, and $\boldsymbol{\tau}_{t+10}$ is the next 10 time-step's 2D trajectory to follow. The reward is computed as $r_t^{\text{terrain}} = \exp(-2\|\boldsymbol{p}_t{}^{(0)} - \boldsymbol{\tau}_t\|) - 0.0005 \cdot \sum_{j \in \text{ joints}} |\boldsymbol{\mu}_j \dot{\boldsymbol{q}}_j|^2$ where the first term is the trajectory following the reward and the second term an energy penalty. Random trajectories are generated procedurally, with a velocity between [0, 3]m/s and acceleration between [0, 2] m/s^2. The height map $\boldsymbol{o}_t$ is a rasterized local height map of size $\boldsymbol{o}_t \in \mathcal{R}^{32\times32\times3}$, which captures a 2m $\times$ 2m square centered at the humanoid. We do not consider any shape variation or human-to-human interaction as in PACER. Different from PACER, which relies on an additional adversarial reward to achieve realistic and human-like behavior, our framework policy does not rely

on any additional reward but can still solve this challenging task with human-like movements. We hypothesize that this is a result of sampling from the pre-learned prior, where human-like motor skills are easier to sample than unnatural ones.

VR Controller Tracking. Tracking VR controllers is the task of inferring full-body human motion from the three 6DOF poses provided by the VR controllers (headset and two hand controllers). Following QuestSim (Winkler et al., 2022), we train this tracking policy using synthetic data. Essentially, we treat the humanoid's head and hand positions as a proxy for headset and controller positions. One can view the VR controller tracking task as an imitation task, but with only three joints to track, with the goal state being: $\boldsymbol{s}_t^{\text{g-vr}} \triangleq (\hat{\boldsymbol{\theta}}_{t+1}^{\text{vr}} \ominus \boldsymbol{\theta}_t{}^{\text{vr}}, \hat{\boldsymbol{p}}_{t+1}^{\text{vr}} - \boldsymbol{p}_t{}^{\text{vr}}, \hat{\boldsymbol{v}}_{t+1}^{\text{vr}} - \boldsymbol{v}_t, \hat{\boldsymbol{\omega}}_t^{\text{vr}} - \boldsymbol{\omega}_t^{\text{vr}}, \hat{\boldsymbol{\theta}}_{t+1}^{\text{vr}}, \hat{\boldsymbol{p}}_{t+1}^{\text{vr}})$ where the superscript $^{\text{vr}}$ refers to selecting only the head and two hands joints. During training, we use the same (full-body) imitation reward to train the policy. We use the same progressive training procedure for training the tracking policy.

REFERENCES

Zipeng Fu, Xuxin Cheng, and Deepak Pathak. Deep whole-body control: Learning a unified policy for manipulation and locomotion. *arXiv preprint arXiv:2210.10044*, 2022.

K Fukushima. Cognitron: a self-organizing multilayered neural network. *Biol. Cybern.*, 20:121–136, 1975. ISSN 0340-1200,1432-0770.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Dianbo Liu, Alex Lamb, Kenji Kawaguchi, Anirudh Goyal, Chen Sun, Michael Curtis Mozer, and Yoshua Bengio. Discrete-valued neural communication. *arXiv preprint arXiv:2107.02367*, 2021.

Matthew Loper, Naureen Mahmoody, and Michael J Blackz. Mosh: Motion and shape capture from sparse markers. *ACM Trans. Graph.*, 33, 2014. ISSN 0730-0301,1557-7368.

Zhengyi Luo, Ryo Hachiuma, Ye Yuan, and Kris Kitani. Dynamics-regulated kinematic policy for egocentric pose estimation. *NeurIPS*, 34:25019–25032, 2021. ISSN 1049-5258.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines, 2010.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. Deepmimic. *ACM Trans. Graph.*, 37:1–14, 2018. ISSN 0730-0301.

Xue Bin Peng, Michael Chang, Grace Zhang, Pieter Abbeel, and Sergey Levine. Mcp: Learning composable hierarchical control with multiplicative compositional policies. *arXiv preprint arXiv:1905.09808*, 2019.

Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Trans. Graph.*, abs/2104.02180: 1–20, 2021.

Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *arXiv preprint arXiv:2205.01906*, 2022.

Davis Rempe, Zhengyi Luo, Xue Bin Peng, Ye Yuan, Kris Kitani, Karsten Kreis, Sanja Fidler, and Or Litany. Trace and pace: Controllable pedestrian animation via guided trajectory diffusion. *arXiv preprint arXiv:2304.01893*, 2023.

Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. *arXiv preprint arXiv:2109.11978*, 2021.

Aaron Van Den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *Adv. Neural Inf. Process. Syst.*, 2017-Decem:6307–6316, 2017. ISSN 1049-5258.

Alexander Winkler, Jungdam Won, and Yuting Ye. Questsim: Human motion tracking from sparse sensors with simulated avatars. *arXiv preprint arXiv:2209.09391*, 2022.