PhaseMP - Appendix

Mingyi Shi
The University of Hong Kong
myshi@cs.hku.hk

Sebastian Starke
Meta
sebastian.starke@mail.de

Yuting Ye
Meta
yuting.ye@gmail.com

Taku Komura
The University of Hong Kong
taku@cs.hku.hk

Jungdam Won*
Seoul National University
jungdam@imo.snu.ac.kr

A. Introduction

This document provides more explanation of significant issues that have not been addressed in the primary content. Additionally, we present a range of experiments and outcomes obtained from our system.

B. Deep Phase Feature

Expressive Power Phase P is a fundamental component in our system, which we use to express the temporal alignment among the motion frames. In our system, it's formed by multi-dimensional periodic parameters from the neural FFT layer. Despite its efficacy, some individuals may question whether non-periodic movements can be represented through the phase. To address this, we conducted an experiment using the Phase Network (PAE) on two distinct movements, one involving dancing and another involving walking.

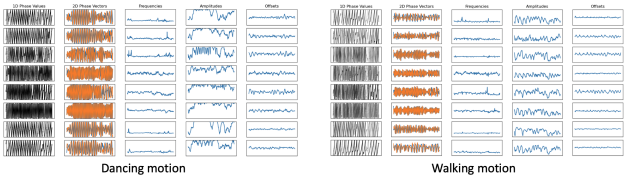

Figure 1. Here we show the periodic parameters extracted from two different motions. The left is from the dancing, and the right is from the walking.

Figure 2 depicts an intriguing observation from the phase feature. Despite dancing appearing more complex than walking at an intuitive level, our extracted feature highlights that the only difference between the two movements is the larger amplitude value associated with dancing due to its faster motion. This phenomenon can be attributed to the real-world motion of walking, which although appearing stable, contains a significant amount of high-frequency differences. Consequently, we utilize multi-dimensional periodic parameters to represent the entirety of the dataset, instead of relying on a simple pattern like foot contact or hand waving. This approach enables us to accurately express both periodic and non-periodic motions, leading to more versatile and practical applications.

In addition, we have employed the T-SNE dimensional reduction method to cluster the periodic parameters extracted from a motion stylization dataset [3]. The results demonstrate that distinct motion styles are identifiable based on their corresponding phase features. This observation serves as compelling evidence that similar motions share similar phase features. Thus, by extending the learned phase features from visible frames to invisible frames, our method can effectively produce consistent and plausible movements.

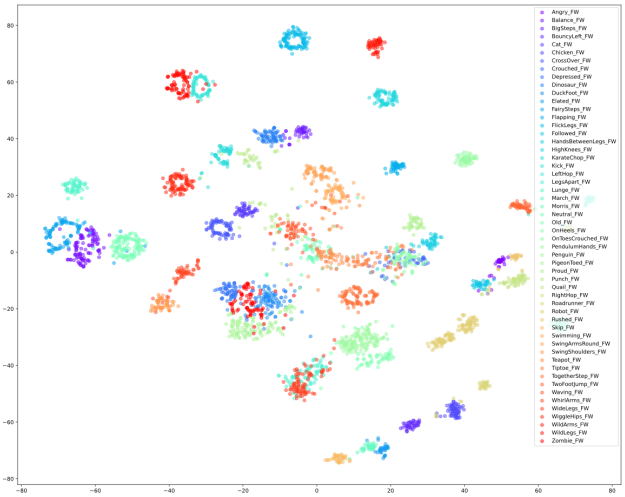

Figure 2. The 2D cluster for the T-SNE of phase parameters which are extracted from different motion clips. The same color represents the clips from the same style of motion.

*Corresponding Author

Unidirectionality The concept of unidirectionality is a main aspect of our system, as it allows us to ensure that movements remain in a consistent direction. This is accomplished through the clockwise rotation of the phase vector, as outlined in Equation 10, which serves as the primary strategy for updating the phase value. Additionally, the frequency parameter is maintained as positive and accumulative, creating a cyclic manner that preserves the one-directional property of the motion.

To provide further insight, we can visualize the 2D PCA embedding of the learned phase vector, as illustrated in Figure 3. In this representation, each data point is located on a cycle wave, indicating that motion generated under such conditions will maintain the unidirectional property. By enforcing this feature through our system, we can generate realistic and natural-looking movements with consistent directionality, enabling a broad range of practical applications.

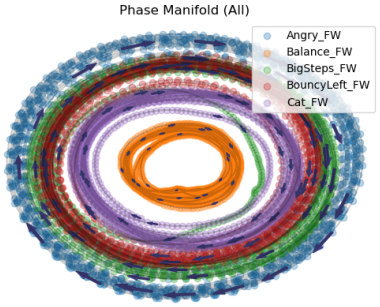

Figure 3. Here we show the 2D PCA embedding of phase vector from different motions. The colors identify the motion, and the arrows identify the direction of updating.

Generalization Training a model using the phase-amplitude encoding (PAE) approach on a large dataset can be computationally expensive due to the neural FFT operation. To overcome this issue, we train the PAE network on a subset (CMU) and then apply it to the entire dataset (AMASS [2]). Furthermore, we conduct an experiment to demonstrate the ability of the trained PAE model to interpolate and extrapolate across different motion categories, indicating its capacity to adapt easily to unseen motion data and produce periodic parameters.

To evaluate the performance of our approach, we partitioned the Lafan motion dataset [1] into subsets A and B based on different motion categories and conducted three training experiments, as presented in Table 1. The reconstruction error mean and maximum values remained within a standard range, indicating reliable performance. Moreover, we tested our approach by removing half of the data from each motion category and evaluating on the remaining data, which demonstrated the ability of our method to perform interpolation effectively. Furthermore, testing on unknown motion categories revealed no significant decrease in network performance, demonstrating the ability of our model to extrapolate to new, unseen data. This suggests that the PAE model trained on the subset can accurately capture the entire dataset.

Method	Train Dataset	Test Dataset	Mean Error	Max Error
PAE	$\frac{1}{2} \times A + \frac{1}{2} \times B$	rest of A, B	0.1286	0.2951
PAE	A	B	0.1546	0.3196
PAE	B	A	0.1481	0.3622

Table 1. Comparison of different training data splitting strategies. The first one use a half and A and B, then test the model on the rest of A and B; Others use one of them to do the training and test on another one.

C. Network Details

C.1. PAE network

The periodic autoencoder (PAE) comprises three primary components: the encoder, feature transformers, and decoder. The encoder comprises two convolutional layers with a kernel size of 31 and a padding size of 15, operating solely on the channel dimension to obtain the embedding with 8 channels. Subsequently, the periodic parameters are extracted using previously described methods, including FFT and 2-argument arctangent. In contrast, the decoder also comprises two convolutional layers, but with the output channel number changed to the original dimension. Batch normalization and tanh activation functions are applied to each convolutional layer except the last one.

To train the PAE model, we employed the CMU subset and subsequently applied it to the entire AMASS dataset [2]. Training the model for 200 epochs on an RTX3090 graphics card takes approximately 5 hours.

C.2. Prior network

The system architecture used in the prior network is similar to that of HuMoR [5]. The prior and encoder are fully connected using 5 MLPs with ReLU activations and group normalizations, which operate on the input data into 48-dimensional latent by a hidden size of 1024. To capture the periodic nature of our phase feature, we incorporate SirenMLP as an activation layer in the decoder network, inspired by previous work on periodic activation and NeRF [6, 4]. The decoder comprises of 4 SirenMLPs with sine activation of 60 sine factors, which predict the next frame data. We initialize all the weights of SirenMLP and the first layer of SirenMLP separately, as introduced in [6]. This choice improves reconstruction performance, as demonstrated in our experiments. Additionally, we introduce skip connections

from the phase variable to each layer of the network, enhancing the influence of the phase feature. By employing these techniques, we can more effectively model the periodicity inherent in the motion data, resulting in improved performance in our motion synthesis task.

Data Processing We use the following subset from AMASS dataset in the training: CMU, MPI Limits, TotalCapture, Eyes Japan, KIT, BMLrub, BMLmovi, EKUT, and ACCAD. The data processing phase involved the selection of motion clips from these subsets that were longer than two seconds, which were used to train the prior network. Subsequently, downsampling of the frames per second was performed for each subset to 30 Hz, and padding of 0.5 seconds was added to the beginning and ending frames. The PAE network was then utilized to extract the phase feature for the entire sequence. To eliminate low-dynamic movements, the first and last 10% frames were dropped, as well as all motion involving interaction with the terrain. A 4cm threshold was used to detect and compute the ground truth of the contact label, resulting in 10k motion sequences for final training

Training Strategy In line with the original MotionVAE approach, scheduled sampling was used during the training phase to increase the robustness of the noise input. For network forwarding, a 10 frames subsequence was taken, and there were three stages wherein the prediction of x_t was based on either the ground truth of x_{t-1} or predicted x'_{t-1} To enhance the robustness of the system, the ground truth was fed as input at the beginning, while predicted x'_{t-1} was fed as input later in the training process to enforce correlated results.

Training Time The model was trained for 200 epochs over approximately four days, using a single Tesla V100 16GB GPU.

D. Optimization Details

As introduced in the paper, there are three stages for test-time optimization. We describe them one by one as follows.

D.1. Stage 1: SMPL Alignment

The primary objective of the first stage is to obtain an initial prediction of the SMPL pose sequence with constraints of observation alignment. Similar to other approaches, the observation term E_{data} is employed as the primary energy term to ensure the prediction is aligned with the given observations. Additionally, a smooth term is utilized to enhance temporal coherence. More details, 30 iterations are dedicated to optimizing the root translation and orientation, followed by 70 iterations to optimize the full-body pose and shape. The optimization process in this stage adheres to the details outlined in the HuMoR paper [5].

D.2. Stage 2: Phase&Latent Initialization

In this stage, the initial pose sequence is used to generate joint velocity, which is subsequently fed into our PAE model to produce initial phase features. To compute the phase with the same frame number, the sequence is padded accordingly. Due to unstable global orientation, the initial production of the PAE model results in high-frequency noise. To mitigate this, the phase curve is smoothed and stabilized using Eq. 10, and the process is repeated five times.

Furthermore, the initial z latent sequence is also generated in this stage. This is achieved by providing a pair of frames (the previous and the next) as input to the Encoder.

D.3. Stage 3: Optimization with Prior

In contrast to the previous stages, stage 3 presents significant challenges in practice, such as encountering local minima and diverging optimization. Therefore, we address this issue by dividing the entire sequence into sub-clips with 90 frames (3 seconds) and optimizing each clip separately. Also, we will use the last frame from the previous clip as the initial pose in the next clip for optimization of a long sequence to ensure consistent.

Using the initial pose x_0, and sequences of z and phase P, we can generate a sequence of SMPL poses autoregressively, frame by frame. The optimization variable in this stage consists of x_0 and the sequence of z. The energy term comprises four modules: (1) Plausibility, which refers to the negative log-likelihood for the motion prior term; (2) Alignment, which measures the distance between the predicted and observed poses; (3) Phase, which quantifies the difference between the predicted and target phases; and (4) Regularization, which aims to optimize the motion to be smooth and consistent.

$$\begin{aligned}\underset{z_{1:T-1},\beta,g}{\operatorname{argmin}} \; & (\lambda_{obs}E_{obs} + \lambda_{prior}E_{prior} \\ & +\lambda_{reg}E_{reg} + \lambda_{phase}E_{phase})\end{aligned} \tag{1}$$

D.4. Term Weights in Optimization

In different tasks, we carefully design the weights for different optimization terms. The input data is more reliable for the pose estimation from space 3d joint input, so we emphasize the energy on the 3d joint alignment. Also, the ground won't be predicted in these tasks. In the video2motion task, because the detected 2d pose includes noises, we should emphasize other energy terms except the 2d joint alignment loss. The weight values are listed in Table 2.

Task	λ_{obs}^{3d}	λ_{obs}^{2d}	λ_{prior}	λ_{reg}	λ_{phase}
Sparse 3D Fitting	1.0	-	$5e^{-4}$	0.1	0.1
Motion in-between	1.0	-	$1e^{-3}$	0.1	0.2
3D Denoising	0.5	-	$1e^{-3}$	1	0.2
Video2Motion	-	$1e^{-3}$	0.05	2	0.4

Table 2. The weight of different energy terms in the test-time optimization.

D.5. Training tricks

In practical implementation, we employ a specific optimization strategy for each clip. In the first 30 iterations, we optimize the first 15 frames to enhance the autoregressive updating process based on the initial state x_0 and P_0. This initial prediction plays a critical role in the entire sequence prediction. Afterward, we fix x_0 and P_0 and optimize only $z_{1:T}$ for 25 iterations to align the full movements. Finally, we optimize all parameters for the remaining 15 iterations to obtain the final results. During each iteration, we refine the phase feature again. For optimization of a 90 frames sequence, it will take 20 minutes with RTX 3090 graphic card.

E: Extended Evaluation

E.1. Qualitative for Occlusions

Here we do more experiments to explore how robust our method is in different scenarios. With AMASS dataset, we show the results by using different occlusion heights in the upper-body experiment, and different time steps in the temporal occluded experiment.

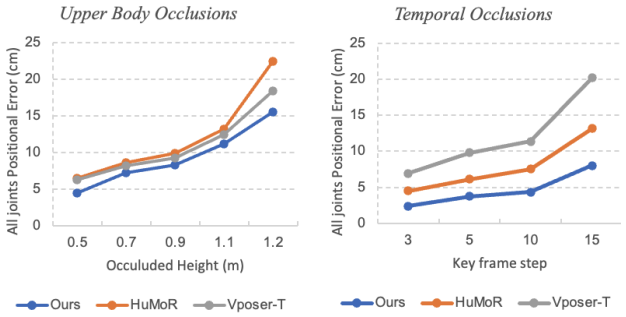

Figure 4. Here we show the different performances when we apply different occlusion parameters on the upper-body visible experiment and temporal occlusion experiments.

The positional reconstruction error is shown in Figure 4. By increasing the strength of occlusions, our system is able to keep the best performance and also show superiority and robustness under heavy occlusion environments.

F. More Results

In this section, we show more comparisons in two parts: 1) Estimation from partial 3D and 2) Estimation from RGB

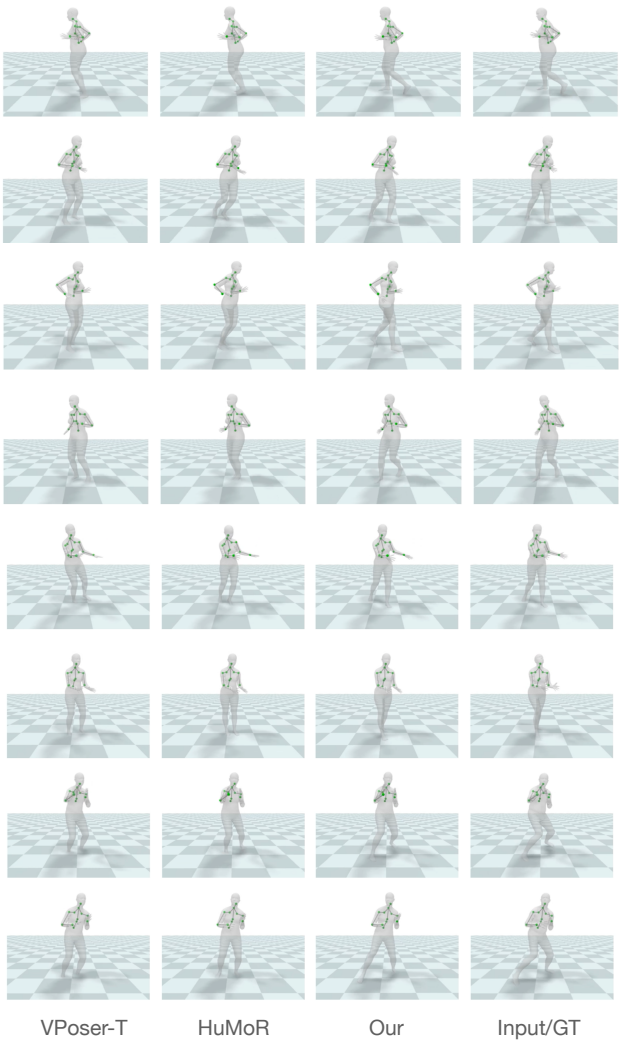

Figure 5. Here we show the results from upper body reconstruction experiments.

videos. Also, we encourage to review our supplementary video to appreciate the improvement of our method.

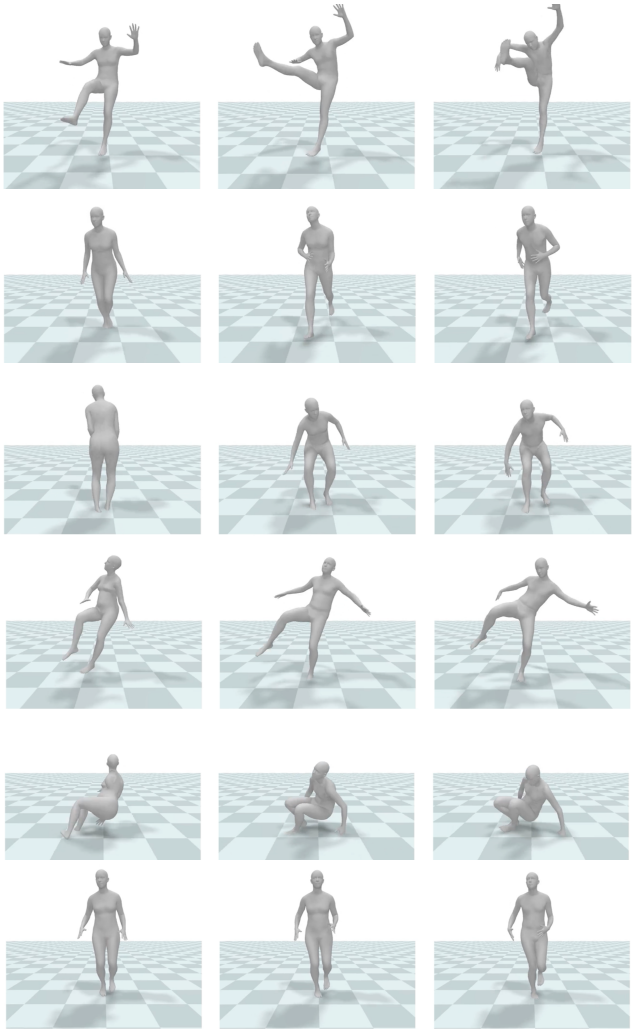

Figure 6. Here we show the results from temporal occlusion experiments.

Figure 7. Here we show the different performances from the video2motion experiment. Middle: HuMoR, right: Ours

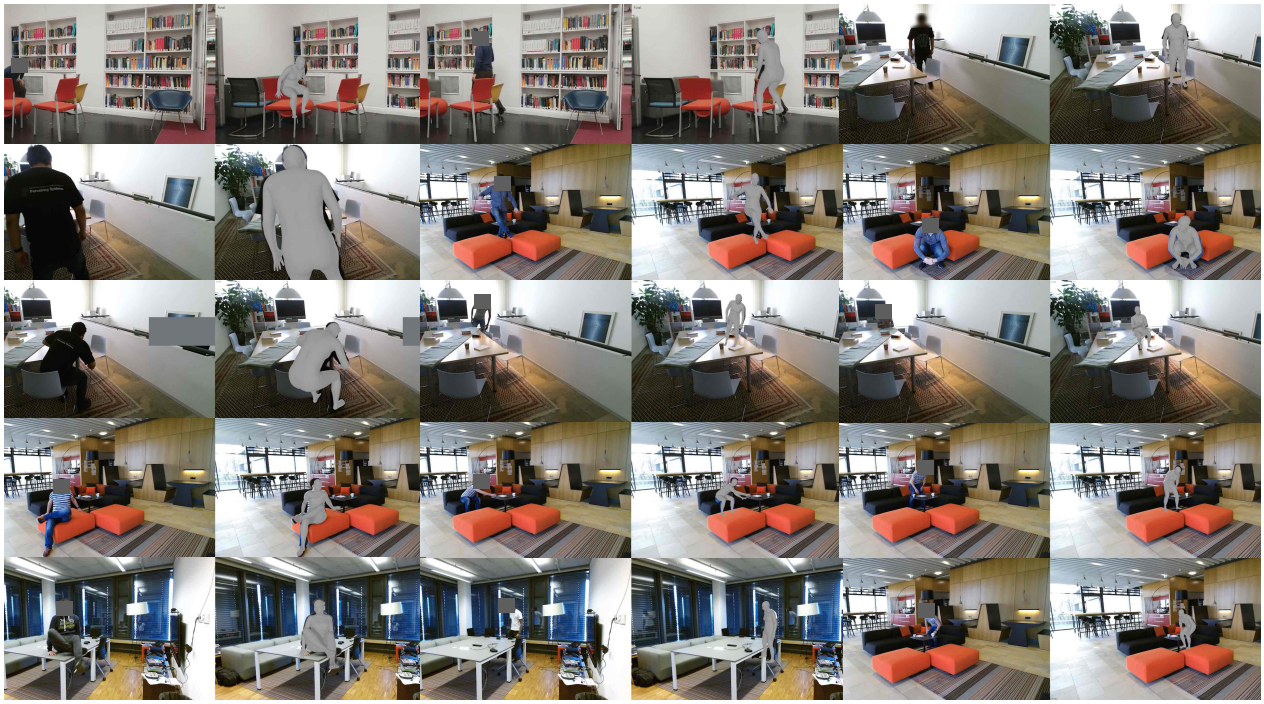

Figure 8. Here we show more video2motion results.

References

[1] Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. 39(4), 2020.

[2] Muhammed Kocabas, Nikos Athanasiou, and Michael J. Black. VIBE: Video inference for human body pose and shape estimation. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5252–5262. IEEE, June 2020.

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

[4] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[5] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J. Guibas. Humor: 3d human motion model for robust pose estimation. In *International Conference on Computer Vision (ICCV)*, 2021.

[6] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020.