NeMF: Neural Motion Fields for Kinematic Animation

Chengan He
Yale University
chengan.he@yale.edu

Jun Saito
Adobe Research
jsaito@adobe.com

James Zachary
Adobe Research
zachary@adobe.com

Holly Rushmeier
Yale University
holly.rushmeier@yale.edu

Yi Zhou
Adobe Research
yizho@adobe.com

Abstract

We present an implicit neural representation to learn the spatio-temporal space of kinematic motions. Unlike previous work that represents motion as discrete sequential samples, we propose to express the vast motion space as a continuous function over time, hence the name *Neural Motion Fields (NeMF)*. Specifically, we use a neural network to learn this function for miscellaneous sets of motions, which is designed to be a generative model conditioned on a temporal coordinate t and a random vector $\boldsymbol{z}$ for controlling the style. The model is then trained as a Variational Autoencoder (VAE) with motion encoders to sample the latent space. We train our model with a diverse human motion dataset and quadruped dataset to prove its versatility, and finally deploy it as a generic motion prior to solve task-agnostic problems and show its superiority in different motion generation and editing applications, such as motion interpolation, in-betweening, and re-navigating. More details can be found on our project page: https://cs.yale.edu/homes/che/projects/nemf/.

1 Introduction

Motion synthesis and editing is a core problem in animation and game production, as well as in emerging applications like artificial agents. Traditional algorithms have limited capability of automatically producing convincing motions with high diversity and complexity. With the recent availability of large-scale motion capture data, more interest has shifted to deep learning-based methods. People have adapted various deep learning technologies such as Recurrent Neural Networks [34], Reinforcement Learning [25, 35] and Normalizing Flows [10] for skeletal motion generation. However, those methods are designed as an auto-regressive process that depends on its own past values and some stochastic terms. With this constraint, those methods have to sequentially predict the motion at discrete time steps. Consequently, they cannot just directly infer the motion at certain frames since they must predict the past motion first. Moreover, with this temporally asymmetric design, it is always hard for those methods to incorporate the control or editing of future frames and, thus, it is difficult to apply them to tasks like motion in-betweening.

Inspired by the recent success in neural radiance fields (NeRF) for novel view synthesis [32], we introduce *Neural Motion Fields (NeMF)* to model the spatio-temporal space of kinematic motions. Given that a motion sequence consists of different poses at different time steps, we represent a motion sequence as a continuous function $f : t \mapsto f(t)$ which parameterizes the entire sequence by the temporal coordinate t. This function can be approximated by a multilayer perceptron network (MLP) whose parameters are optimized by minimizing the reconstruction loss between the generated and ground truth motion.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

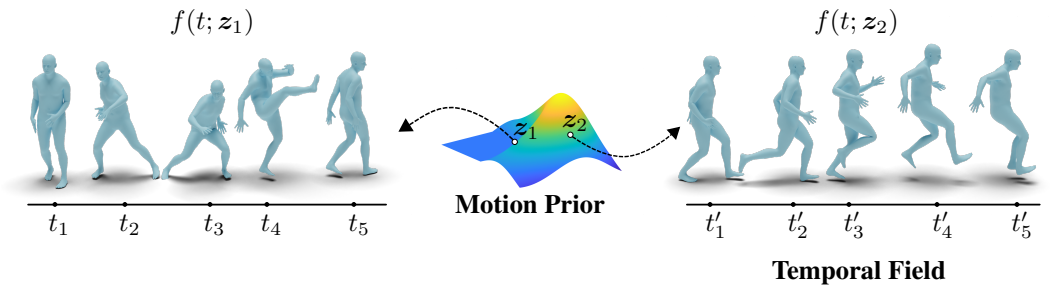


Figure 1: Overview of generative NeMF, which consists of a motion prior and a continuous temporal field. By sampling the motion prior, NeMF is able to infer a variety of motions with different styles. By sampling the temporal field, NeMF is able to generate motion at arbitrary frames.

However, the NeMF function defined above overfits one motion sequence. To extend it to general motion synthesis, as illustrated in Figure 1, we further introduce a random vector $\boldsymbol{z}$ as the conditioning variable to the function as $f(t; \boldsymbol{z})$, which now encodes the mapping from the normal distribution to the manifold spanned by all plausible motions. By varying $\boldsymbol{z}$, we expect f to output motion with different styles, and by varying t, we obtain the pose at arbitrary time. We train this generative NeMF model in the form of a Variational Autoencoder (VAE), where motion sequences are fed to convolutional encoders first to obtain the sampling of $\boldsymbol{z}$, and then $\boldsymbol{z}$ is passed through an MLP decoder for the reconstruction of the whole motion. To the best of our knowledge, our paper presents the first continuous and generative implicit motion representation that can sample different types of motion at an arbitrary time step.

This neural motion representation can serve as a generic motion prior to solve task-agnostic problems. Following Li et al. [21] who trained a hierarchical human motion prior and applied it in different tasks, we demonstrate utilizing a trained NeMF in various applications, including motion interpolation, motion in-betweening and motion re-navigating. We formulate those tasks as optimization problems to find the latent variable $\boldsymbol{z}$ that minimize the target energies and restore the complete plausible motion sequences. Although we did not train this NeMF model for any of these tasks specifically, it achieves performance equivalent or superior to task-specific alternatives in our experiments.

In summary, our contributions are as follows:

- We propose NeMF that represents the continuous motion field as a function over time and design a VAE architecture to train a generative NeMF.
- We validate that our model can reconstruct and synthesize motion with superior quality and diversity than state-of-the-art neural motion priors.
- We demonstrate NeMF in various offline motion editing and creation tasks.

2 Related Work

To clarify our design choices, we categorize the kinematic motion modeling into two types: time series models and space-time models. The former approach views motions as the kinematic pose evolving over time, which is often the choice for interactive applications where the future depends on unknown factors such as real-time user inputs. The latter is our approach, which is suitable for offline applications where we know what to expect for the overall motion within some time range.

2.1 Time Series Models

Time series models are often formulated as an auto-regressive model that predicts the future based on current and past observations. The predictions are fed into the model again to make further predictions recursively. GPDM [44] modeled such dynamics of human locomotion over time with Gaussian process. Recently people studied various deep learning-based auto-regressive architectures, including Recurrent Neural Networks [5, 29, 49], Reinforcement Learning Networks, Neural ODE [16], Transformers [1, 22, 36] and other attention models [28]. Many works demonstrated success in

real-time interactive kinematic character controls [13, 40–42, 47]. The auto-regressive approach is also suitable for embracing the uncertainty of future predictions with generative models, for example, in the form of VAE as in HuMoR [38] and Motion VAE [25], and in the form of Normalizing Flows as in MoGlow [10].

In applications, Harvey et al. [9] used LSTM [11] with positional encoding to achieve the state-of-the-art result for the motion in-betweening task. But it is limited to a fixed in-betweening setting where some consecutive past frames need to be given as the seed. One of the key applications of our model is also motion in-betweening, but our method has no constraint on the setting and can generate the missing motion both between motion clips and between sparse keyframes.

2.2 Space-Time Models

Different from time series models, another approach is to directly model the spatio-temporal kinematic state itself. Such models are often used in combination with a time series model, as in [20] where the dynamic control policy is learned over the GPLVM-based spatial kinematic prior. One of our inspirations, Motion Fields [19], can be seen as a variant of this approach where the time-series dynamics are learned over the spatial kinematic model represented directly by the data and a hand-crafted distance function. More recently, space-time neural network models overcame the scalability issue in previous methods due to the availability of large-scale datasets [12, 14, 15, 45, 48]. Zhou et al. [51] demonstrated long-term motion in-betweening by learning the space-time kinematic prior without supervision using Generative Adversarial Networks (GAN) [7]. Kaufmann et al. [18] also proposed a convolutional network for motion in-betweening. Our other inspiration, HM-VAE [21], learned a generic motion prior for multiple downstream applications. However, their reliance on a temporally convolutional decoder fixes the output frame rate, thus limiting its application to temporal sub-sampling. As a space-time model, our work NeMF has the advantage of being able to edit any frames in the motion sequence while maintaining fidelity and the source style compared to the aforementioned works.

2.3 Implicit Neural Representations

Implicit neural representations explosively gained popularity with the success of SIREN [39] and NeRF [32], which achieved state-of-the-art results in many tasks including novel view synthesis, geometry reconstruction, and solving differential equations. Among different tasks, the core idea of neural implicit representations is to build a *continuous function* that maps the spatial or temporal coordinates to any signal at that location while maintaining high-frequency details. Inspired by their success, our key insight is to interpret time as the parameter of a *motion function* and learn the landscape of the spatio-temporal kinematics manifold as an implicit motion representation parameterized by temporal coordinates. Similar ideas have been proposed in modeling time-varying 3D geometries [33] and dynamic scenes [24, 37], and we extend it to the animation domain.

3 Formulation

In this section, we will first give the definition and notation of our motion representation, then present the formulation of NeMF for a single motion sequence, followed by an extended version of generative NeMF for the entire motion space.

3.1 Motion Representation

Similar to Zhou et al. [51] and Li et al. [21], we divide the motion into two parts: *local motion*, which contains the pose of the skeleton relative to the root at time t, and *global motion*, which is the global translation of the root joint. Following Fussell et al. [6], we represent the local motion at time t as a matrix $\mathbf{X}_t$ composed of joint positions $\mathbf{x}_t^p \in \mathbb{R}^{J\times 3}$, velocities $\dot{\mathbf{x}}_t^p \in \mathbb{R}^{J\times 3}$, rotations $\mathbf{x}_t^r \in \mathbb{R}^{J\times 6}$ in 6D rotation form [50], and angular velocities $\dot{\mathbf{x}}_t^r \in \mathbb{R}^{J\times 3}$:

$$\mathbf{X}_t = (\mathbf{x}_t^p \quad \dot{\mathbf{x}}_t^p \quad \mathbf{x}_t^r \quad \dot{\mathbf{x}}_t^r) \in \mathbb{R}^{J\times 15}, \tag{1}$$

where J is the number of joints. Since all other quantities can be computed from joint rotations, we focus on predicting joint rotations $\mathbf{x}_t^r$ in the formulation. We then factor out the root orientation $\mathbf{r}_t^o \in \mathbb{R}^6$ from local motion by multiplying the inverse of the root transform to each quantity in $\mathbf{X}_t$.

In this way, all poses will be transformed to a local space with the same facing direction, thus making them easier to learn for our network. For the global motion, based on previous works [21, 51], we use a neural network to predict the velocity of the root joint $\dot{\mathbf{r}}_t \in \mathbb{R}^3$ as well as its height $\mathbf{r}_t^h \in \mathbb{R}$ from $\mathbf{X}_t$. We provide details of our global motion predictor in the supplemental.

3.2 Neural Motion Fields

Unlike most other works that represent motion as a discrete sequential process, we represent motion as a continuous vector field of kinematic poses in the *temporal* domain. Hence we define this motion field as a function that maps temporal coordinates t to joint rotations and root orientations:

$$f : t \mapsto (\mathbf{x}_t^r, \mathbf{r}_t^o). \tag{2}$$

The function f can be approximated by a neural network with the parameters θ that can be optimized by minimizing the reconstruction loss between the generated and ground truth motion, where the ground truth poses can be considered as discrete samples of f at integer time steps.

Similar to NeRF [32], we train an MLP with positional encoding of t to fit a given motion sequence. Considering a sequence with T frames, we first convert the 6D rotations $\mathbf{x}_t^r$ and $\mathbf{r}_t^o$ to rotation matrices $\mathbf{R}_t$ and $\mathbf{R}_t^o$ with the Gram-Schmidt-like process described in [50], and then compute the geodesic distance to measure the rotational difference:

$$\mathcal{L}_{\text{rot}} = \sum_{t=1}^{T} \arccos \frac{\text{Tr}\left(\mathbf{R}_t(\hat{\mathbf{R}}_t)^{-1}\right) - 1}{2}, \quad \mathcal{L}_{\text{ori}} = \sum_{t=1}^{T} \arccos \frac{\text{Tr}\left(\mathbf{R}_t^o(\hat{\mathbf{R}}_t^o)^{-1}\right) - 1}{2}. \tag{3}$$

We then evaluate the L_1 loss on local joint positions obtained from forward kinematics (FK) [43], which regularizes the skeletal topology to obtain better results [34]:

$$\mathcal{L}_{\text{pos}} = \sum_{t=1}^{T} \|\mathbf{x}_t^p - \hat{\mathbf{x}}_t^p\|_1. \tag{4}$$

The reconstruction loss is finally expressed as a weighted sum of the above terms with weighting factors λ_{rot}, λ_{ori}, and λ_{pos}:

$$\mathcal{L}_{\text{rec}} = \lambda_{\text{rot}}\mathcal{L}_{\text{rot}} + \lambda_{\text{ori}}\mathcal{L}_{\text{ori}} + \lambda_{\text{pos}}\mathcal{L}_{\text{pos}}, \tag{5}$$

and we optimize the network parameters θ to minimize this loss function.

3.3 Generative Neural Motion Fields

Although the neural motion field defined above can represent a motion sequence in a compact way, it cannot generate a variety of motions. To change to a different motion sequence, the entire network needs to be re-trained from scratch, which severely limits its application. Therefore, in this section, we extend NeMF to a generative model which can represent the entire motion space instead of a specific motion sequence.

First of all, we introduce a conditioning variable $\boldsymbol{z}$ to the input of f, thus parameterizing the entire *spatio-temporal* kinematics space as a function:

$$f : (t, \boldsymbol{z}) \mapsto (\mathbf{x}_t^r, \mathbf{r}_t^o), \tag{6}$$

where $\boldsymbol{z}$ defines the spatial location on the manifold and t controls the temporal evolution of the sequence.

We then propose a VAE to formulate Equation 6 as a latent variable model with Gaussian distribution. Based on our motion representation, the VAE contains two separate convolutional motion encoders to learn and parameterize the posterior distribution of latent variables $\boldsymbol{z}_l$ and $\boldsymbol{z}_g$, which control the local motion and root orientation respectively:

$$q_{\phi_1}(\boldsymbol{z}_l \mid \mathbf{X}) = \mathcal{N}(\boldsymbol{z}_l; \mu_{\phi_1}(\mathbf{X}), \sigma_{\phi_1}(\mathbf{X})), \quad q_{\phi_2}(\boldsymbol{z}_g \mid \mathbf{r}^o) = \mathcal{N}(\boldsymbol{z}_g; \mu_{\phi_2}(\mathbf{r}^o), \sigma_{\phi_2}(\mathbf{r}^o)), \tag{7}$$

where $\mathbf{X}$ and $\mathbf{r}^o$ are the concatenation of all $\mathbf{X}_t$ and $\mathbf{r}_t^o$ within the same sequence.

The combination of $\boldsymbol{z}_l$ and $\boldsymbol{z}_g$ forms the final representation of the latent variable $\boldsymbol{z}$, which are then passed through the MLP decoder to produce joint rotations $\mathbf{x}_t^r$ and root orientations $\mathbf{r}_t^o$ for each time step t, thus, defining the output probability distribution $p_\theta(\mathbf{x}^r, \mathbf{r}^o \mid \boldsymbol{z}_l, \boldsymbol{z}_g)$.

In addition to the probabilistic perspective, our model can also be interpreted as learning a *motion prior* from input sequences, where each latent variable on the prior corresponds to a natural motion sequence. The MLP decoder then approximates the mapping function between the motion prior and pose space, thus allowing the navigation in the pose space to have a consistent motion style with a fixed $\boldsymbol{z}$.

During training, we consider the modified variational lower bound:

$$\begin{aligned}\log p_\theta(\mathbf{x}^r, \mathbf{r}^o) \geq \mathbb{E}_{q_{\phi_1}, q_{\phi_2}}[\log p_\theta(\mathbf{x}^r, \mathbf{r}^o \mid \boldsymbol{z}_l, \boldsymbol{z}_g)] - \mathcal{D}_{\text{KL}}(q_{\phi_1}(\boldsymbol{z}_l \mid \mathbf{X}) \parallel p(\boldsymbol{z}_l)) \\ - \mathcal{D}_{\text{KL}}(q_{\phi_2}(\boldsymbol{z}_g \mid \mathbf{r}^o) \parallel p(\boldsymbol{z}_g)),\end{aligned} \tag{8}$$

where the expectation term measures the reconstruction error of the decoder, and the KL divergence $\mathcal{D}_{\text{KL}}$ regularizes the encoders' outputs to be near $p(\boldsymbol{z}_l)$ and $p(\boldsymbol{z}_g)$, which are $\mathcal{N}(\mathbf{0}, \mathbf{I})$ in our scenario. Thus, we formulate the loss function $\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}}$ with the weight λ_{KL} to approximate Equation 8, and the network parameters (ϕ_1, ϕ_2, θ) are optimized during training to minimize the loss.

3.4 Applications

After training the generative NeMF, we can deploy it as a generic motion prior to solve different tasks via latent space optimization.

Motion In-betweening is a long-standing animation creation problem of generating motion in the interval between two clips or sets of keyframes. Our insight is, given a sparse set of observations, we can search in the latent space of NeMF to approximate the entire underlying motion thanks to our continuous representation. Thus, we define the energy function as the reconstruction loss on given frames $\mathcal{T}$, which measures the differences on joint rotations, root orientations, joint positions and root translations:

$$\boldsymbol{z}_l^*, \boldsymbol{z}_g^* \coloneqq \arg\min_{\boldsymbol{z}_l, \boldsymbol{z}_g} \sum_{\mathcal{T}} \lambda_{\text{rot}}\mathcal{L}_{\text{rot}} + \lambda_{\text{ori}}\mathcal{L}_{\text{ori}} + \lambda_{\text{pos}}\mathcal{L}_{\text{pos}} + \lambda_{\text{trans}}\mathcal{L}_{\text{trans}}, \tag{9}$$

where $\mathcal{L}_{\text{trans}}$ weighted by λ_{trans} evaluates the L_1 loss on root joint positions $\mathbf{r}_t$ predicted from our standalone global motion predictor. To facilitate convergence, SLERP is first used to perform interpolation in the joint angle space, and then the interpolated inputs are passed through encoders to obtain the initialization of $\boldsymbol{z}_l$ and $\boldsymbol{z}_g$.

Motion Re-navigating is a task where we redirect a reference motion with a new trajectory while preserving its style. More formally, the goal is to generate a motion that 1) looks similar to the given exemplar; and 2) follows the given trajectory as close as possible. To formulate the energy function, we first borrow ideas from time-series analysis by introducing the soft dynamic time warping metric ($\mathcal{L}_{\text{sDTW}}$) [4] to measure the similarity between joint positions in the canonical frame [38]. Then, we project the predicted root joints onto the ground plane and compute the L_1 loss between them and the given 2D trajectory. Last but not least, an additional regularization term is introduced to ensure that the generated and reference motion have a similar angle between their forward direction $\mathbf{f}_t$ and the tangent direction $\mathbf{t}_t$ on the trajectory, where the similarity is determined by $\mathcal{L}_{\text{sDTW}}$ as well. In total, the energy function we try to minimize can be expressed as:

$$\boldsymbol{z}_l^*, \boldsymbol{z}_g^* \coloneqq \arg\min_{\boldsymbol{z}_l, \boldsymbol{z}_g} \sum_{t=1}^{T} \lambda_{\text{sim}}\mathcal{L}_{\text{sDTW}}(\mathbf{x}_t^p, \hat{\mathbf{x}}_t^p) + \lambda_{\text{traj}}\|\mathbf{r}_t^{\text{proj}} - \hat{\mathbf{r}}_t^{\text{proj}}\|_1 + \lambda_{\text{angle}}\mathcal{L}_{\text{sDTW}}(\mathbf{f}_t \cdot \mathbf{t}_t, \hat{\mathbf{f}}_t \cdot \hat{\mathbf{t}}_t), \tag{10}$$

where λ_{sim}, λ_{traj} and λ_{angle} are three weighting factors introduced to balance different terms. In our experiments, $\boldsymbol{z}_l$ are initialized from the encoder's output, while $\boldsymbol{z}_g$ are optimized from scratch.

4 Experiments

We train our model on the AMASS dataset [27] for most of the experiments. After processing, we have roughly 20 hours of human motion sequences at 30 fps for training and testing. We additionally train our model for the reconstruction experiments on a quadruped motion dataset [47], which contains 30 minutes of dog motion capture at 60 fps. Details are provided in the supplemental.

4.1 Sanity Test

4.1.1 Motion Reconstruction

We first perform a sanity test to validate our single-motion NeMF model described in Section 3.2, where we test the reconstruction capability of NeMF with different lengths of motions.

For AMASS data, we pick 16 different motion sequences for each length and report the reconstruction errors in Table 1. For evaluation metrics, we report mean rotation error (**MRE**, $^\circ$) and mean position error (**MPE**, cm) for each joint, which measure the geodesic distance between joint rotations and Euclidean distance between root-aligned joint positions. For the root joint, we further report its orientation error (**MOE**, $^\circ$), which measures the geodesic distance on the root orientation. From the results shown in Table 1, we can observe that a simple MLP with positional encoding is able to to achieve very low reconstruction error (lower than $4mm$ positional error and 0.7° rotational error) for sequence lengths varying from 32 to 512 frames.

We further train our NeMF model on a very long ($4,336$ frames and $73s$) quadruped motion sequence and visualize the result in the supplemental. Our predicted motion is visually almost identical to the ground truth.

Table 1: Mean reconstruction errors of single-motion NeMF for motion of different lengths. Mean rotation error ($^\circ$), mean position error (cm), and mean orientation error ($^\circ$) are reported.

	Sequence Lengths				
Metrics	32	64	128	256	512
MRE	0.610	0.482	0.381	0.369	0.379
MPE	0.314	0.249	0.213	0.192	0.170
MOE	0.465	0.340	0.344	0.321	0.300

4.1.2 Temporal Sampling

Unlike other motion models, NeMF is theoretically guaranteed to generate smooth motion in arbitrary frame rates. We find that the dimension of the Fourier features generated by positional encoding plays a critical role in the generalization here.

Since the maximal frequency of the Fourier features is determined by a hyperparameter L as illustrated in [32], we uniformly sample L from 1 to 21 and train NeMF with these different setups. For each trained model, we respectively generate 30 and 60 fps motions and report their mean per-joint velocity as the smoothness metric in Figure 2. Although a large value of L does not harm the results with the training frame rate, such a model struggles to generalize to different frame rates. However, since positional encoding is crucial to achieve satisfactory reconstruction results, we set $L = 7$ throughout our experiments to balance the trade-off. In the supplemental video, we show that the generated motion remains smooth even sampled at 240 fps.

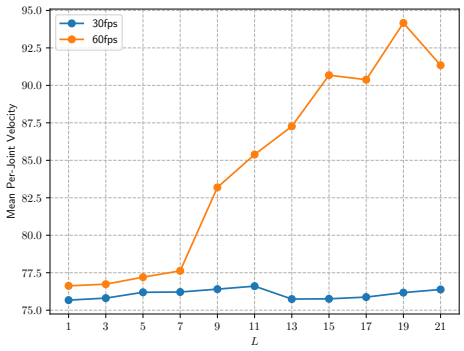


Figure 2: Mean per-joint velocity (cm/s) evaluated on the same motion with different L.

4.2 Generative NeMF

4.2.1 Evaluation Metrics

We evaluate the reconstruction capability of our generative NeMF model through direct network inference and motion synthesis capability through latent space sampling. To better describe motion reconstruction, we further measure the Euclidean distance on the root joint translation (**MTE**, cm) and global joint displacement on the final step (**FDE**, cm) [45]. For estimating the versatile motion quality, we introduce three metrics, namely Fréchet Inception Distance (**FID**), diversity (**Diversity**), and foot skating (**FS**), following related papers [8, 25, 36, 47]. Generally speaking, a lower FID suggests a more natural result, a higher diversity indicates a more various result, and foot skating shows the accumulated drift of foot joints during contact. Please see our supplemental for details.

4.2.2 Ablation Study

We first take a look at the effect of the positional encoding function. By removing positional encoding, the input temporal coordinate t will be passed directly to the MLP to infer pose parameters. Results in the first row of Table 2 show that the model without positional encoding produces higher errors in most metrics we evaluate, thus demonstrating the necessity of positional encoding in terms of the fidelity of the results.

We further experiment on using single motion encoder instead of two separated ones to process both local motion and root orientation. In this simpler design, the root orientation and local motion are entangled in the same latent space. From the results we reported in the second row of Table 2, though the simpler version produces a lower reconstruction error on the root orientation, it has worse performance on joint rotations, joint positions and root translations.

Table 2: Ablation study.

	Motion Reconstruction				Motion Synthesis		
	MRE ↓	**MPE** ↓	**MOE** ↓	**MTE** ↓	**FID** ↓	**Diversity** ↑	**FS** ↓
No Positional Encoding	6.413	2.952	6.027	33.524	2.146	2.783	0.612
Single Motion Encoder	8.661	4.503	**5.982**	35.391	2.185	**2.806**	0.818
Full Model	**5.988**	**2.870**	6.157	**29.692**	**2.073**	2.774	**0.573**

4.2.3 Latent Space Sampling

In our model, we can navigate in the latent space to control the motion styles and synthesize novel motion. To examine the smoothness of the latent space and see whether our model can blend different styles of motion at the sequence level, we linearly interpolate $\boldsymbol{z}$ from two existing motion sequences and infer the novel ones. Previous works like [46] demonstrate motion interpolation between similar motion patterns, such as from walking to zombie-style walking. However, we show in the supplemental video that our model can blend the high-level perceptual style between much harder cases like jumping and punching.

Since we disentangle the latent space for local motion and root orientation, we also experiment to combine $\boldsymbol{z_l}$ and $\boldsymbol{z_g}$ from different motions. Through this, we can create interesting editing results like canceling the spinning motion of a pirouette jump (see supplemental).

4.2.4 Comparison with Other Generative Models

We compare our method with other deep learning-based motion priors on motion reconstruction and synthesis in Table 3. We select HuMoR [38] and HM-VAE [21] to represent the "time series model" and "space-time model" respectively.

Table 3: Comparison of NeMF with other generative motion models.

	Motion Reconstruction					Motion Synthesis		
	MRE ↓	**MPE** ↓	**MOE** ↓	**MTE** ↓	**FDE** ↓	**FID** ↓	**Diversity** ↑	**FS** ↓
HuMoR [38]	13.008	9.071	17.097	**23.882**	62.756	8.687	1.741	0.904
HM-VAE [21]	10.258	7.686	13.054	90.924	137.218	7.998	2.002	0.690
Ours	**5.988**	**2.870**	**6.157**	29.692	**42.985**	**6.508**	**2.118**	**0.566**

Although HuMoR can generate convincing motions, we observe two shortcomings of HuMoR: 1) in motion reconstruction, HuMoR's results will gradually diverge due to its auto-regressive prediction. From the quantitative results in Table 3 and qualitative results in Figure 3, divergence can be observed in both the large FDE and global translation difference respectively. 2) In motion synthesis, HuMoR tends to favor common motions like walking when inferring long sequences, thus yielding a relatively high FID and low diversity. As for our method, we can get rid of these artifacts since we handle the entire sequence at once instead of predicting the motion frame by frame.

HM-VAE [21] HuMoR [38] Ours

Figure 3: Comparison of motion reconstruction with other generative motion models. Predicted motions are overlapped on top of the ground truth motion (yellow).

Compared to HuMoR and our model, HM-VAE models an over-smoothed latent space, thus filtering out high-frequency details in both the reconstructed and synthesized motions. In short, our model outperforms these state-of-the-art methods in most of the metrics reported in Table 3, as well as achieving the closest match of the ground truth motion visualized in Figure 3.

4.3 In-betweening Tasks

In this task, we compare our method with traditional and deep learning-based motion in-betweening methods. For quantitative evaluation, we use the FID and foot skating metrics only, since reconstruction errors cannot fully reflect the quality of motion for highly-varied plausible solutions.

Table 4: Motion clips in-betweening.

	FID		
Length (frames)	10	20	30
SLERP	0.027	0.170	0.455
Inertialization [2]	0.025	0.184	0.496
RMI [9]	0.264	0.362	0.609
HM-VAE [21]	0.137	0.387	0.675
Ours	**0.024**	**0.141**	**0.365**
	Foot Skating		
SLERP	1.193	1.200	1.023
Inertialization [2]	1.237	1.234	1.151
RMI [9]	1.629	1.833	1.643
HM-VAE [21]	0.862	0.835	0.832
Ours	**0.646**	**0.697**	**0.660**

Table 5: Sparse keyframe in-betweening.

	FID			
Length (frames)	5	10	15	20
SLERP	**0.032**	0.305	0.713	1.191
HM-VAE [21]	2.300	2.370	2.434	2.423
Ours	0.085	**0.302**	**0.612**	**0.879**
	Foot Skating			
SLERP	0.868	1.224	1.278	1.193
HM-VAE [21]	0.892	0.837	0.841	**0.751**
Ours	**0.719**	**0.826**	**0.837**	0.804

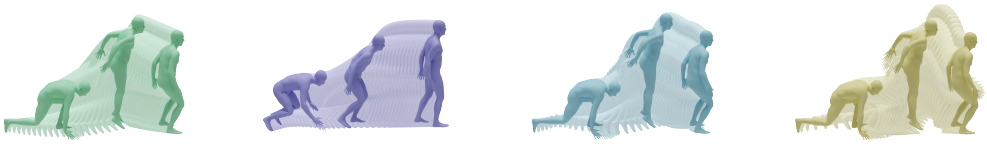
Motion Clips In-betweening. In this task, we aim at generating motion in the interval from 10 frames ($0.33s$) to 30 frames ($1s$) between two clips. We compare with traditional local interpolation methods including SLERP and Inertialization [2], and deep learning-based methods including Robust Motion In-betweening (RMI) [9] and HM-VAE [21]. In the results reported in Table 4, our method outperforms all other alternatives quantitatively. We further experiment on real dancing footage by randomly picking several pairs of videos from AIST++ [23] with their corresponding SMPL [26] parameters. We set the transition length to 30 frames and optimize for a latent variable to produce the gap-filling motion. As demonstrated in Figure 4, our model is capable of producing a natural one-second transition between these real dancing footage.

Sparse Keyframe In-betweening. In this task, a set of sparse keyframe skeleton poses are given every 5, 10, 15 or 20 frames. We compare our method with SLERP and HM-VAE, and report the quantitative results in Table 5. Neither RMI nor Inertialization is applicable here since they both require multiple frames at the beginning.

From Table 5, SLERP has the best FID score for interval of 5 frames, but becomes worse with increased interval length. Compared to SLERP and HM-VAE, ours has much better quantitative and qualitative results, especially for long intervals (Figure 5). In the onionskin images, though SLERP's

Figure 4: Generating the transition (cyan) between AIST++ motion clips (yellow).

result looks similar to ours, our result has more natural high-frequency details. HM-VAE fails to reconstruct the poses at keyframes and generates over-smoothed results. Our result, although different from the ground truth, is visually plausible as in the supplemental video.

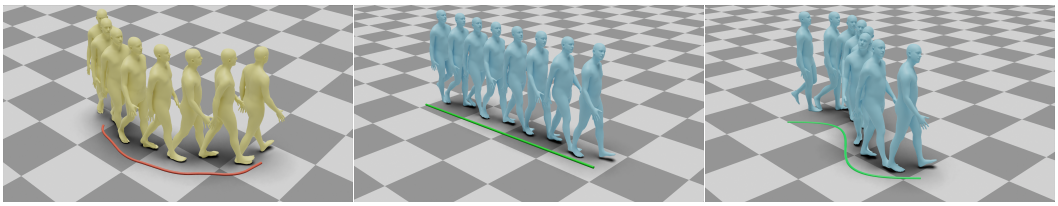


SLERP HM-VAE [21] Ours Ground Truth

Figure 5: Comparison of sparse keyframe in-betweening. The translucent in-between poses are generated from the opaque reconstructed keyframes given every 20 frames.

4.4 Re-navigating Tasks

In this task, we experiment on redirecting the reference motion to different synthetic trajectories. Qualitative results are reported in Figure 6. The original motion style is well preserved in the results, and the character has a natural orientation while walking along the trajectory.



Reference motion Straight line Sinusoidal curve

Figure 6: Motion re-navigating from a reference walking motion (left) to a straight line (middle) and sinusoidal curve (right).

5 Conclusion

We propose an implicit neural motion representation that defines a continuous motion field over style and time. We design it to be a generative model for the whole motion space with a learned prior. We design and test different network architectures and use the trained generative model as a motion prior for solving different tasks like motion interpolation, motion in-betweening and motion re-navigating.

5.1 Limitation and Future Work

In this paper we train the generative NeMF model in the form of a VAE, so it has the limitation of not always giving satisfactory results when sampling the latent space. A promising direction is to design other generative architectures such as GAN, Normalizing Flows, and Diffusion Models for NeMF. Besides, though several latent variables may exist that satisfy the optimization constraints, our deterministic optimization process can only find one possible z as the final result, thus limiting probabilistic motion synthesis like MoGlow [10].

In the future, since the optimization problems we formulated are analogous to the latent space optimization problems of StyleGAN [17], how to transfer the algorithm of those successful applications designed for StyleGAN to our NeMF setup is an interesting direction to explore.

So far our method focuses on motion modeling and cannot adapt to outputs with different body shapes. On the other hand, some works [3, 30, 31] leverage implicit representations to model skinned articulated objects with given poses. Therefore, it would be interesting to combine our work with them to enable the animation and rendering of a specific character with novel motions.

To keep our model task-agnostic, we choose to solve motion tasks by optimizing $\boldsymbol{z}$. But for practical applications, task-specific inference models may be preferred for performance consideration. To have an end-to-end inference model for real-time applications, a possible solution is to design and train the encoder to directly predict $\boldsymbol{z}$ with different input settings.

5.2 Broader Impact

Our model can be applied for kinematic motion creation in animation and game production. Due to the limitation of diversity in the training data, our model is not guaranteed to generate feasible but uncommon motions. We cannot guarantee that our model will avoid generating motions that could be perceived as offensive to some viewers.

Acknowledgments and Disclosure of Funding

This work was supported in part by NSF Grant No. IIS-2007283. Many thanks to Ruben Villegas for helpful discussions. Special thanks to Jiaman Li for sharing with us the pre-trained HM-VAE model.

References

[1] Emre Aksan, Manuel Kaufmann, Peng Cao, and Otmar Hilliges. A spatio-temporal transformer for 3d human motion prediction. In *2021 International Conference on 3D Vision (3DV)*, pages 565–574. IEEE, 2021. 2

[2] David Bollo. Inertialization: High-Performance animation transitions in 'gears of war'. *Proc. of GDC 2018*, 2018. 8

[3] Xu Chen, Yufeng Zheng, Michael J Black, Otmar Hilliges, and Andreas Geiger. Snarf: Differentiable forward skinning for animating non-rigid neural implicit shapes. In *International Conference on Computer Vision (ICCV)*, 2021. 10

[4] Marco Cuturi and Mathieu Blondel. Soft-dtw: A differentiable loss function for time-series. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 894–903. JMLR.org, 2017. 5

[5] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proceedings of the IEEE international conference on computer vision*, pages 4346–4354, 2015. 2

[6] Levi Fussell, Kevin Bergamin, and Daniel Holden. Supertrack: Motion tracking for physically simulated characters using supervised learning. *ACM Trans. Graph.*, 40(6), dec 2021. 3

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. 3

[8] Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2motion: Conditioned generation of 3d human motions. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*, 2020. 6

[9] Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. *ACM Trans. Graph.*, 39(4), jul 2020. 3, 8

[10] Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. Moglow: Probabilistic and controllable motion synthesis using normalising flows. *ACM Trans. Graph.*, 39(6), nov 2020. 1, 3, 9

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997. 3

[12] Daniel Holden, Oussama Kanoun, Maksym Perepichka, and Tiberiu Popa. Learned motion matching. *ACM Trans. Graph.*, 39(4):53:1–53:12, July 2020. 3

[13] Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neu-ral networks for character control. *ACM Trans. Graph*, 36, 2017. 3

[14] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35(4):1–11, July 2016. 3

[15] Daniel Holden, Jun Saito, Taku Komura, and Thomas Joyce. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs*, number Article 18 in SA '15, pages 1–4, New York, NY, USA, November 2015. Association for Computing Machinery. 3

[16] Boyan Jiang, Yinda Zhang, Xingkui Wei, Xiangyang Xue, and Yanwei Fu. Learning compositional representation for 4d captures with neural ode. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5340–5350, 2021. 2

[17] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 10

[18] Manuel Kaufmann, Emre Aksan, Jie Song, Fabrizio Pece, Remo Ziegler, and Otmar Hilliges. Convolutional autoencoders for human motion infilling. In *2020 International Conference on 3D Vision (3DV)*, pages 918–927. IEEE, 2020. 3

[19] Yongjoon Lee, Kevin Wampler, Gilbert Bernstein, Jovan Popović, and Zoran Popović. Motion fields for interactive character locomotion. In *ACM SIGGRAPH Asia 2010 papers*, number Article 138 in SIGGRAPH ASIA '10, pages 1–8, New York, NY, USA, December 2010. Association for Computing Machinery. 3

[20] Sergey Levine, Jack M. Wang, Alexis Haraux, Zoran Popović, and Vladlen Koltun. Continuous character control with low-dimensional embeddings. *ACM Trans. Graph.*, 31(4), July 2012. 3

[21] Jiaman Li, Ruben Villegas, Duygu Ceylan, Jimei Yang, Zhengfei Kuang, Hao Li, and Yajie Zhao. Task-generic hierarchical human motion prior using vaes. 2021. 2, 3, 4, 7, 8, 9

[22] Jiaman Li, Yihang Yin, Hang Chu, Yi Zhou, Tingwu Wang, Sanja Fidler, and Hao Li. Learning to generate diverse dance motions with transformer. *arXiv preprint arXiv:2008.08171*, 2020. 2

[23] Ruilong Li, Shan Yang, David A. Ross, and Angjoo Kanazawa. Ai choreographer: Music conditioned 3d dance generation with aist++, 2021. 8

[24] Tianye Li, Mira Slavcheva, M. Zollhöfer, S. Green, Christoph Lassner, Changil Kim, Tanner Schmidt, S. Lovegrove, M. Goesele, R. Newcombe, and Z. Lv. Neural 3d video synthesis from multi-view video. 2022. 3

[25] Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel Van De Panne. Character controllers using motion VAEs. *ACM Trans. Graph.*, 39(4):40:1–40:12, July 2020. 1, 3, 6

[26] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015. 8

[27] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pages 5442–5451, October 2019. 5

[28] Wei Mao, Miaomiao Liu, and Mathieu Salzmann. History repeats itself: Human motion prediction via motion attention. In *European Conference on Computer Vision*, pages 474–489. Springer, 2020. 2

[29] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2891–2900, 2017. 2

[30] Marko Mihajlovic, Shunsuke Saito, Aayush Bansal, Michael Zollhoefer, and Siyu Tang. COAP: Compositional articulated occupancy of people. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2022. 10

[31] Marko Mihajlovic, Yan Zhang, Michael J Black, and Siyu Tang. LEAP: Learning articulated occupancy of people. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 10

[32] 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. 1, 3, 4, 6

[33] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 3

[34] Dario Pavllo, David Grangier, and Michael Auli. Quaternet: A quaternion-based recurrent model for human motion. In *British Machine Vision Conference (BMVC)*, 2018. 1, 4

[35] Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. Sfv: Reinforcement learning of physical skills from videos. *ACM Transactions On Graphics (TOG)*, 37(6):1–14, 2018. 1

[36] Mathis Petrovich, Michael J. Black, and Gül Varol. Action-conditioned 3D human motion synthesis with transformer VAE. In *International Conference on Computer Vision (ICCV)*, pages 10985–10995, October 2021. 2, 6

[37] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 3

[38] 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. 3, 5, 7, 8

[39] 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. 3

[40] Sebastian Starke, He Zhang, Taku Komura, and Jun Saito. Neural state machine for character-scene interactions. *ACM Trans. Graph.*, 38(6):1–14, November 2019. 3

[41] Sebastian Starke, Yiwei Zhao, Taku Komura, and Kazi Zaman. Local motion phases for learning multi-contact character movements. *ACM Trans. Graph.*, 39(4):54:1–54:13, July 2020.

[42] Sebastian Starke, Yiwei Zhao, Fabio Zinno, and Taku Komura. Neural animation layering for synthesizing martial arts movements. *ACM Trans. Graph.*, 40(4):1–16, July 2021. 3

[43] Ruben Villegas, Jimei Yang, Duygu Ceylan, and Honglak Lee. Neural kinematic networks for unsupervised motion retargetting. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 4

[44] Jack M Wang, David J Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):283–298, February 2008. 2

[45] Ye Yuan and Kris Kitani. Dlow: Diversifying latent flows for diverse human motion prediction. In *European Conference on Computer Vision*, pages 346–364. Springer, 2020. 3, 6

[46] M Ersin Yumer and Niloy J Mitra. Spectral style transfer for human motion between independent actions. *ACM Transactions on Graphics (TOG)*, 35(4):1–8, 2016. 7

[47] He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. Mode-adaptive neural networks for quadruped motion control. *ACM Trans. Graph.*, 37(4), jul 2018. 3, 5, 6

[48] Siwei Zhang, Yan Zhang, Federica Bogo, Marc Pollefeys, and Siyu Tang. Learning motion priors for 4d human body capture in 3d scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11343–11353, 2021. 3

[49] Yan Zhang, Michael J Black, and Siyu Tang. We are more than our joints: Predicting how 3d bodies move. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3372–3382, 2021. 2

[50] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019. 3, 4

[51] Yi Zhou, Jingwan Lu, Connelly Barnes, Jimei Yang, Sitao Xiang, et al. Generative tweening: Long-term inbetweening of 3d human motions. *arXiv preprint arXiv:2005.08891*, 2020. 3, 4

NeMF: Neural Motion Fields for Kinematic Animation – Supplementary Material –

Chengan He
Yale University
chengan.he@yale.edu

Jun Saito
Adobe Research
jsaito@adobe.com

James Zachary
Adobe Research
zachary@adobe.com

Holly Rushmeier
Yale University
holly.rushmeier@yale.edu

Yi Zhou
Adobe Research
yizho@adobe.com

Contents

1 Method Details

1.1 Global Motion Predictor

Given the fact that the character's global translation is conditioned on its local poses, similar to [14, 33], we design a fully convolutional network to generate the global translation $\mathbf{r}$ of the root joint based on the local joint positions, velocities, rotations, and angular velocities as inputs. To eliminate ambiguities in the output, instead of generating the root position directly, we try to predict its velocity

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

$\dot{\mathbf{r}}$, which can be integrated using the forward Euler method to compute $\mathbf{r}$:

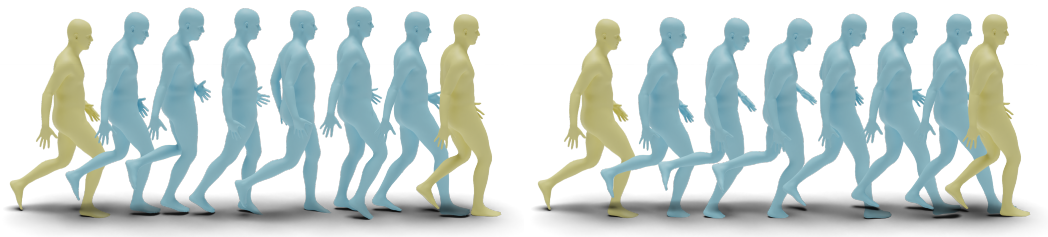
$$\mathbf{r}_{t+1} = \mathbf{r}_t + \dot{\mathbf{r}}_t \Delta t = \mathbf{r}_1 + \sum_{i=1}^{t} \dot{\mathbf{r}}_i \Delta t. \tag{1}$$

However, cumulative errors are inevitable during the integration process, and this becomes more pronounced in the upward direction, where the character gradually moves into the air or under the ground. To avoid this phenomenon, we directly predict the height $\mathbf{r}^h$ of the root joint, which is reasonable since it lies in a region bounded by the height of the character. Then, we measure the differences on the generated velocities and integrated positions as the loss function to minimize:

$$\mathcal{L} = \mathcal{L}_{\text{vel}} + \mathcal{L}_{\text{trans}} \tag{2}$$

$$\mathcal{L}_{\text{vel}} = \sum_{t=1}^{T} \|\dot{\mathbf{r}}_t - \hat{\dot{\mathbf{r}}}_t\|_1, \quad \mathcal{L}_{\text{trans}} = \sum_{t=1}^{T} \|\mathbf{r}_t - \hat{\mathbf{r}}_t\|_1. \tag{3}$$

A Note on An Alternative Integrated Model. A simpler design choice is to predict the global translation, orientation and local motion all at once, and we call it an *integrated model*. However, in our experiments, we observed that this integrated model cannot fully decouple local and global motion, which is more evident when applied in tasks like motion in-betweening. As shown in Figure 1 and our supplemental video, the integrated model tends to generate static local poses with global translation for motion in the interval, thus causing sliding artifacts. While in the separate model, it doesn't have this artifact since this model explicitly decouples local and global motion.



Separate Model Integrated Model

Figure 1: Comparison between separate model and integrated model on motion in-betweening. The two yellow poses are ground truth and the cyan poses are generated results in the interval.

A Note on Our Single-Motion NeMF Model. When training our single-motion NeMF model, we train the MLP to fit both the local and global motion. Therefore, its loss function contains terms both from Equation 5 of our main paper and Equation 2 above. While for our generative model, we observe some artifacts as illustrated in Figure 1 and choose to train a standalone global motion predictor to handle the global motion. Therefore, we drop the loss terms related to global motion when training our VAE.

1.2 Network Architecture

Motion Encoders. We introduce two separate motion encoders to parameterize the latent space of local motion and root orientation respectively. To encode local motion, we adopt the Skeleton Convolution and Skeleton Pooling layers proposed in [1] to build a residual block with PReLU activations [11] and group normalization [31]. The motion encoder contains 4 layers of these skeleton convolution residual blocks with kernel size 4 to extract latent features from local pose parameters, which are followed by 2 fully-connected layers to obtain the mean and variance of $\boldsymbol{z}_l$ in 1024 dimensions. As for root orientation, its residual block is built on 1D convolution and 1D average pooling layers with PReLU activations, and its encoder also contains 4 layers of residual blocks with kernel size 4 to gradually upscale the root orientation in $\mathbb{R}^6$ to latent features in 128, 256, 512, and 512 dimensions. The fully-connected layers then map the latent features to the mean and variance of $\boldsymbol{z}_g$ in 256 dimensions.

MLP Decoder. Similar to [22], we build an MLP to predict local pose parameters and root orientation based on latent variables and temporal coordinates with positional encoding. The MLP contains 11 fully-connected layers with ReLU activations and layer normalization [3]. Each hidden layer has the output size 1024, while a skip connection is introduced in each layer of the MLP to emphasize the importance of the input and to help prevent posterior collapse [16].

Global Motion Predictor. Our global motion predictor has a similar architecture as our motion encoder, where 3 layers of skeleton convolution residual blocks with kernel size 15 are applied to to extract latent features from the local pose parameters. These latent features are then mapped to root velocity and root height with 4 additional residual blocks composed of 1D convolution and 1D average pooling layers with kernel size 15.

2 Experiment Details

2.1 Datasets

We train our model on the AMASS dataset[1] [19] for human motion, which is a motion capture database that aggregates mocap data from multiple datasets and normalizes them into a uniform format. We then leverage the data processing scripts provided by HuMoR [25] to filter some outlier data and unify their frame rates to 30 fps. Here, we plot the duration distribution of the processed AMASS data in Figure 2 and collect some detailed statistics in Table 1.

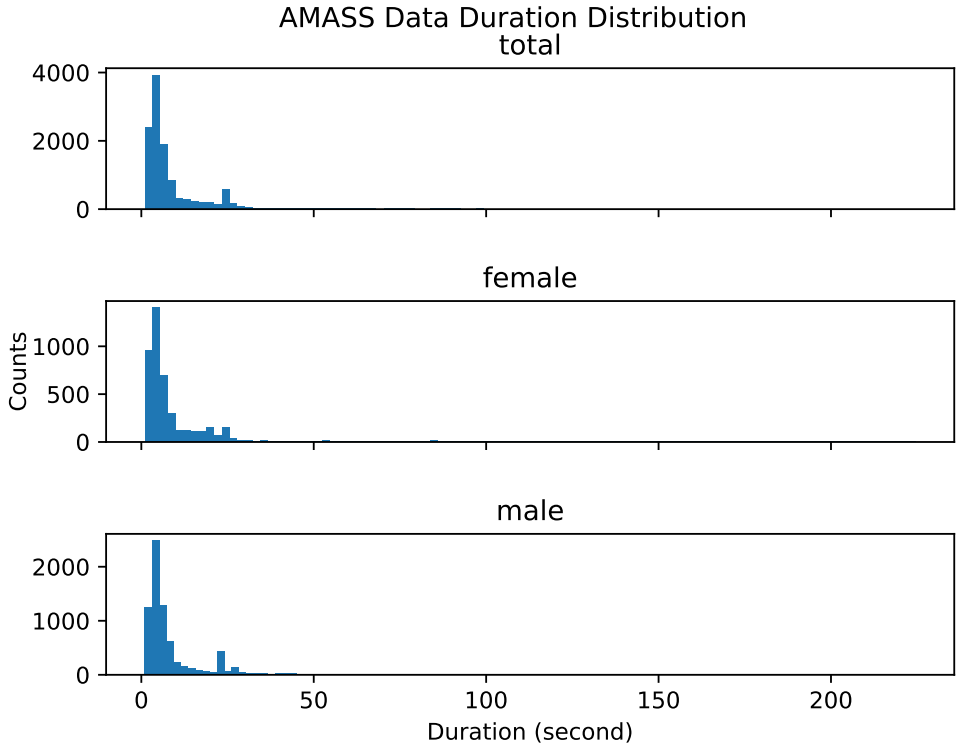


Figure 2: Duration distribution of AMASS data.

From the statistics we collect, the duration of AMASS data has a large variance while most of them are between 0 and $50s$. To be more specific, only more than 50.98% of the data are longer than $5s$. Therefore, to fully utilize the AMASS data, we choose to chop the sequences into clips with 128 frames ($4.3s$) and set the batch size to be 16 throughout experiments. Then we split these processed

[1]For the license of AMASS, please check: https://amass.is.tue.mpg.de/license.html.

data into training, validation and testing sets, where the training set contains data from CMU [29], MPI Limits [2], TotalCapture [28], Eyes Japan [18], KIT [20], BMLrub [27], BMLmovi [8], EKUT [20], ACCAD [6], BMLhandball [12], DanceDB [5], DFaust [4], and SSM [19], the validation set contains data from MPI HDM05 [23], SFU [30], and MPI Mosh [17], and the testing set contains data from HumanEva [26] and Transitions [19]. This split results in $11,642$ sequences in the training set, $1,668$ sequences in the validation set and 164 sequences in the testing set, roughly 20 hours in total for use.

We additionally train our model for the reconstruction experiments on a quadruped motion dataset [32], which contains 30 minutes of dog motion capture. Similar to [13], we manually filter those clips on uneven terrain and the remaining data are all in 60 fps with various lengths from 155 to $13,399$ frames.

Table 1: Detailed statistics collected from the AMASS data.

AMASS Data Statistics	
Total motion sequences	$11,831$
Total Duration	$119,661.40s$
Minimal Duration	$0.97s$
Maximal Duration	$224.57s$
Average Duration	$10.11s$
Sequences longer than $5s$	$6,031$ (50.98%)
Sequences longer than $10s$	$2,739$ (23.15%)
Male Data	$7,400$ (Duration: $73,989.10s$)
Female Data	$4,431$ (Duration: $45,672.30s$)

2.2 Training & Optimization

We train our model and conduct all the experiments on a cluster with 8 Intel® Xeon® Gold 6136 CPUs @ 3.00GHz, 64GB memory, and 2 NVIDIA Tesla V100 GPUs. Our code is implemented with Python 3.9.7 and PyTorch 1.9.0.

Training. We employ Adam optimizer throughout the training for all NeMF architectures with the learning rate of 0.0001. We train our single-motion NeMF for 500 iterations to fit a 32-frame sequence, and scale the number of iterations proportionally as the sequence length increases to make sure that our model is sufficiently trained for each length of sequences. As for our generative NeMF and global motion predictor, we train their architectures for $1,000$ epochs with weight decay 0.0001.

Test-Time Optimization. Our test-time optimization utilizes Adam with the initial learning rate of 0.1. In all experiments, our method converges within 600 iterations with proper initialization, and we decay the learning rate to 0.07 and 0.049 at iteration 200 and 400, respectively.

Hyperparameters. In all of our experiments, we set the weights λ_{rot} to 1.0, λ_{ori} to 1.0, and λ_{pos} to 10.0. In training our generative NeMF, we initially set λ_{KL} to $1e^{-5}$. To combat posterior collapse, we adopt the cyclical annealing schedule [7] to linearly anneal λ_{KL} from $1e^{-7}$ to its full value every 50 epochs. For the energy functions formulated during test-time optimization, we set the weights λ_{trans} to 1.0, λ_{sim} to 0.5, λ_{traj} to 1.0, and λ_{angle} to 1.0.

2.3 Motion Reconstruction & Synthesis

In the ablation study and comparison, we evaluate both the reconstruction and synthesis capability of our generative NeMF. For motion reconstruction, we use the trained network to directly infer the 164 samples in the testing set and compute some deterministic metrics to measure the reconstruction errors. For motion synthesis, we generate 400 samples through latent space sampling and introduce three additional metrics to measure the quality of motion.

2.4 Qualitative Metrics

In our experiments, we employ the following three metrics to measure the motion characteristics that reconstruction errors cannot capture, namely Fréchet Inception Distance (**FID**), diversity (**Diversity**) and foot skating (**FS**).

Fréchet Inception Distance (FID). FID is a statistical metric which has been widely used for measuring the image quality, while Guo et al. [9] and Petrovich et al. [24] have transferred it to the motion domain and employ it in tasks such as action recognition. To evaluate FID, we use a

pre-trained feature extractor to extract motion features from real and generated motions, then the FID is computed from the distribution of these feature vectors.

Diversity. Diversity was first introduced by Guo et al. [9] to measure the variance of generated motions. To evaluate diversity, we randomly split all generated data into two subsets with equal size. Feature vectors are then extracted from them respectively, and diversity is computed as the mean Euclidean distance between these feature vectors.

Foot Skating (FS). To measure the foot skating artifact, we use the metric proposed in [16, 32]. To be specific, this metric measures the accumulated drift on the ankle and toe joints when their height h is within a certain threshold H. Their velocity is first projected onto the horizontal plane to compute the magnitude v, which is further weighted with the formula $s = v(2 - 2^{\frac{h}{H}})$. In our experiments, we set H according to the values provided by HuMoR [25], which are $4cm$ for toe joints and $8cm$ for ankle joints. This parameter setting leads to an average foot skate of $0.512cm$ per frame in the ground truth data.

To build the feature extractor for FID and diversity, we train an auto-encoder that maps the input motion parameters to feature vectors. The auto-encoder has a similar architecture as our VAE, except that it takes both local and global motion as input and the fully-connected layer outputs latent vectors directly instead of their mean and variance. Similar to [9, 24], we randomly pick 100 samples from the generated data to evaluate these plausible metrics in each iteration, and perform 20 iterations with different random seeds. We then report the mean value of these metrics in our tables.

2.5 Baselines

HM-VAE [14]. We use the pre-trained HM-VAE model released by the authors. The model was trained on the AMASS dataset with a sequence length of 64. To accommodate longer sequences, we use the concatenation method in their open-source code[2] to connect each sub-sequence.

HuMoR [25]. We use the pre-trained HuMoR model released by the authors[3].

Robust Motion In-betweening (RMI) [10]. We choose an open-source implementation of RMI[4] since Ubisoft does not release their official code. We modified this unofficial implementation and trained it on the AMASS dataset to fit our experiment setup.

3 Additional Results

3.1 Motion Reconstruction

In Figure 3 we show our reconstruction result on the quadruped motion. This sequence contains $4,336$ frames at 60 fps ($73s$), which takes $8,000$ iterations to converge. From the visualization in Figure 3 and our supplemental video, our predicted motion is almost identical to the ground truth.

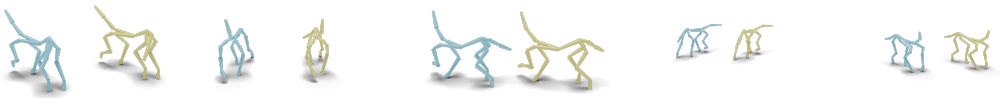
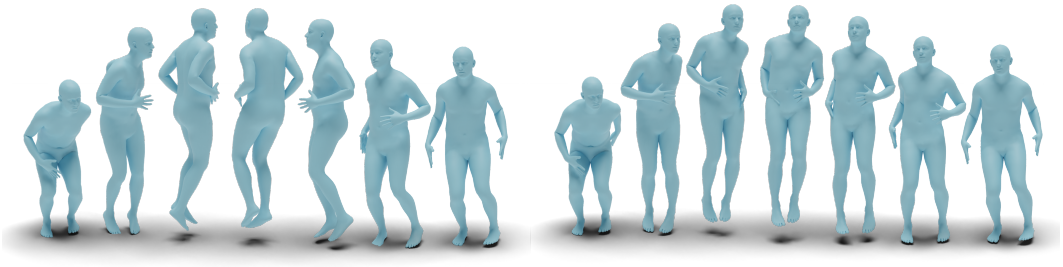


Figure 3: Quadruped motions generated from single-motion NeMF, where the cyan skeleton indicates our generated result and the yellow skeleton is the ground truth motion.

[2] https://github.com/lijiaman/hm-vae
[3] https://github.com/davrempe/humor
[4] https://github.com/xjwxjw/Pytorch-Robust-Motion-In-betweening

3.2 Motion Composition

Since we disentangle the latent space for local motion and root orientation, we can create interesting motion editing results as in Figure 4, where we cancel the spinning motion of a pirouette jump by assigning different $\boldsymbol{z}_g$ to the same $\boldsymbol{z}_l$.



$\boldsymbol{z}_l(\text{jumping}) + \boldsymbol{z}_g(\text{spinning})$ $\boldsymbol{z}_l(\text{jumping}) + \boldsymbol{z}_g(\text{no spinning})$

Figure 4: Orientation editing by assigning different $\boldsymbol{z}_g$ to the same $\boldsymbol{z}_l$.

3.3 Latent Space Interpolation

To examine the smoothness of the latent space and see whether our model can blend different styles of motion at the sequence level, we linearly interpolate $\boldsymbol{z}$ from two existing motion sequences and infer the novel ones as shown in Figure 5.

Figure 5: Latent space interpolation.

3.4 Time Translation in Latent Space

As a motion prior, different motion clips will be mapped to different positions in the latent space. Therefore, it would be interesting to examine the latent patterns formed by those clips which share a large portion of overlap while containing some temporal offsets.

We then set up an experiment by first picking 3 different sequences from AIST++ [15], each containing about 200 to 300 frames. For each sequence, we use a sliding window with an offset of 10 to obtain clips with 118-frame overlap, and then encode these clips with our encoder. The latent variables are projected to 3D with UMAP [21] and visualized in Figure 6. In the cases we test, the clips with overlapping frames are mapped to nearby positions and even form some interesting patterns, thus suggesting that they maintain certain connection in the latent space.

References

[1] Kfir Aberman, Peizhuo Li, Sorkine-Hornung Olga, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Skeleton-aware networks for deep motion retargeting. *ACM Transactions on Graphics (TOG)*, 39(4):62, 2020. 2

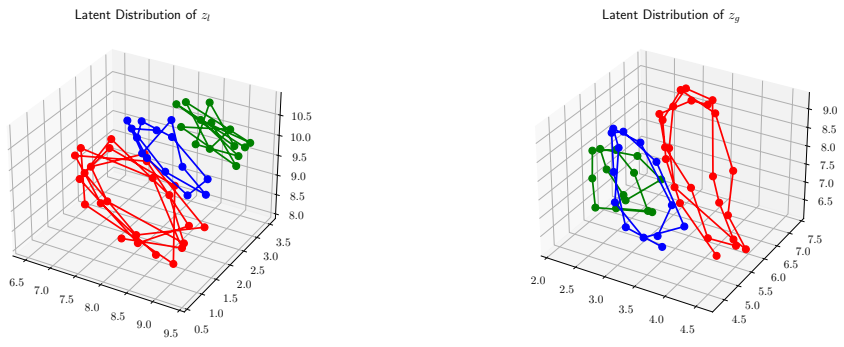


Figure 6: Latent distribution of clips with time translation. Different colors refer to different AIST++ sequences.

[2] Ijaz Akhter and Michael J. Black. Pose-conditioned joint angle limits for 3D human pose reconstruction. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2015*, June 2015. 4

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[4] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: Registering human bodies in motion. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 4

[5] DanceDB. Dance motion capture database. 4

[6] Advanced Computing Center for the Arts and Design. Accad mocap dataset. 4

[7] Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. *arXiv preprint arXiv:1903.10145*, 2019. 4

[8] Saeed Ghorbani, Kimia Mahdaviani, Anne Thaler, Konrad Kording, Douglas James Cook, Gunnar Blohm, and Nikolaus F Troje. Movi: A large multipurpose motion and video dataset. *arXiv preprint arXiv:2003.01888*, 2020. 4

[9] Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2motion: Conditioned generation of 3d human motions. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*, 2020. 4, 5

[10] Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. *ACM Trans. Graph.*, 39(4), jul 2020. 5

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015. 2

[12] Fabian Helm, Nikolaus F Troje, and Jörn Munzert. Motion database of disguised and non-disguised team handball penalty throws by novice and expert performers. *Data in brief*, 15:981–986, 2017. 4

[13] Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. Moglow: Probabilistic and controllable motion synthesis using normalising flows. *ACM Trans. Graph.*, 39(6), nov 2020. 4

[14] Jiaman Li, Ruben Villegas, Duygu Ceylan, Jimei Yang, Zhengfei Kuang, Hao Li, and Yajie Zhao. Task-generic hierarchical human motion prior using vaes. 2021. 1, 5

[15] Ruilong Li, Shan Yang, David A. Ross, and Angjoo Kanazawa. Ai choreographer: Music conditioned 3d dance generation with aist++, 2021. 6

[16] Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel Van De Panne. Character controllers using motion VAEs. *ACM Trans. Graph.*, 39(4):40:1–40:12, July 2020. 3, 5

[17] Matthew M. Loper, Naureen Mahmood, and Michael J. Black. MoSh: Motion and shape capture from sparse markers. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 33(6):220:1–220:13, November 2014. 4

[18] Eyes JAPAN Co. Ltd. Eyes japan mocap dataset. 4

[19] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pages 5442–5451, October 2019. 3, 4

[20] Christian Mandery, Ömer Terlemez, Martin Do, Nikolaus Vahrenkamp, and Tamim Asfour. The kit whole-body human motion database. In *International Conference on Advanced Robotics (ICAR)*, pages 329–336, 2015. 4

[21] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861, 2018. 6

[22] 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. 3

[23] Meinard Müller, Tido Röder, Michael Clausen, Bernhard Eberhardt, Björn Krüger, and Andreas Weber. Documentation mocap database hdm05. 2007. 4

[24] Mathis Petrovich, Michael J. Black, and Gül Varol. Action-conditioned 3D human motion synthesis with transformer VAE. In *International Conference on Computer Vision (ICCV)*, pages 10985–10995, October 2021. 4, 5

[25] 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. 3, 5

[26] L. Sigal, A. Balan, and M. J. Black. HumanEva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision*, 87(1):4–27, March 2010. 4

[27] Nikolaus F Troje. Decomposing biological motion: A framework for analysis and synthesis of human gait patterns. *Journal of vision*, 2(5):2–2, 2002. 4

[28] Matt Trumble, Andrew Gilbert, Charles Malleson, Adrian Hilton, and John Collomosse. Total capture: 3d human pose estimation fusing video and inertial sensors. In *2017 British Machine Vision Conference (BMVC)*, 2017. 4

[29] Carnegie Mellon University. Cmu graphics lab motion capture database. 4

[30] Simon Fraser University and National University of Singapore. Sfu motion capture database. 4

[31] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 2

[32] He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. Mode-adaptive neural networks for quadruped motion control. *ACM Trans. Graph.*, 37(4), jul 2018. 4, 5

[33] Yi Zhou, Jingwan Lu, Connelly Barnes, Jimei Yang, Sitao Xiang, et al. Generative tweening: Long-term inbetweening of 3d human motions. *arXiv preprint arXiv:2005.08891*, 2020. 1