RSMT: Real-time Stylized Motion Transition for Characters

Xiangjun Tang
Zhejiang University
Hangzhou, China
fcsx1tf@163.com

Linjun Wu
Zhejiang University
Hangzhou, China
3190104528@zju.edu.cn

He Wang
University of Leeds
Leeds, United Kingdom
H.E.Wang@leeds.ac.uk

Bo Hu
Tencent Technology Co., Ltd.
Shenzhen, China
corehu@tencent.com

Xu Gong
Tencent Technology Co., Ltd.
Shenzhen, China
xugong@tencent.com

Yuchen Liao
Tencent Technology Co., Ltd.
Shenzhen, China
bluecatliao@tencent.com

Songnan Li
Tencent Technology Co., Ltd.
Shenzhen, China
sunnysnli@tencent.com

Qilong Kou
Tencent Technology Co., Ltd.
Shenzhen, China
rambokou@tencent.com

Xiaogang Jin*
Zhejiang University
Hangzhou, China
jin@cad.zju.edu.cn

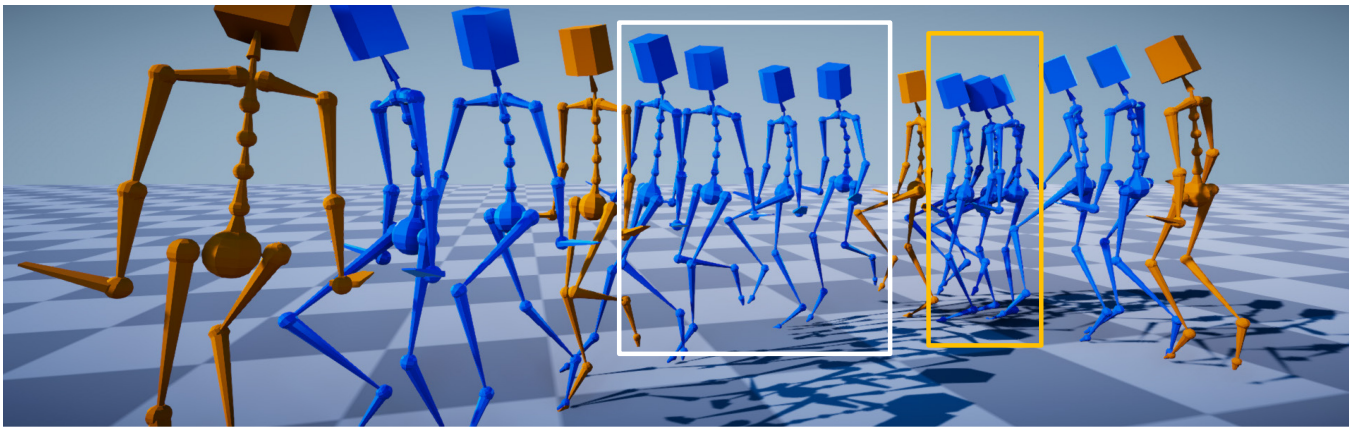

Figure 1: Our method generates an in-between motion sequence with a high leg lifting style (blue) between target frames (orange). Given a target frame, a desired transition duration and the required style, the character can dynamically adjust motions to reach the target with the desired style. Specifically, the character in the white box needs bigger steps to reach the target within the specified duration, while the character in the orange box needs smaller steps to reduce the speed for the same reason.

ABSTRACT

Styled online in-between motion generation has important application scenarios in computer animation and games. Its core challenge lies in the need to satisfy four critical requirements simultaneously: generation speed, motion quality, style diversity, and synthesis controllability. While the first two challenges demand a delicate balance between simple fast models and learning capacity for generation quality, the latter two are rarely investigated together in existing methods, which largely focus on either control without style or uncontrolled stylized motions. To this end, we propose a Real-time Stylized Motion Transition method (RSMT) to achieve all aforementioned goals. Our method consists of two critical, independent components: a general motion manifold model and a style motion sampler. The former acts as a high-quality motion source and the latter synthesizes styled motions on the fly under control signals. Since both components can be trained separately on different datasets, our method provides great flexibility, requires less data, and generalizes well when no/few samples are available for unseen styles. Through exhaustive evaluation, our method proves to be fast, high-quality, versatile, and controllable. The code and data are available at https://github.com/yuyujunjun/RSMT-Realtime-Stylized-Motion-Transition.

*Corresponding author

SIGGRAPH '23 Conference Proceedings, August 06–10, 2023, Los Angeles, CA, USA

ACM ISBN 979-8-4007-0159-7/23/08…$15.00
https://doi.org/10.1145/3588432.3591514

CCS CONCEPTS

• **Computing methodologies** → **Motion capture**; *Motion transition*; Neural networks; Motion manifold.

KEYWORDS

Animation, real-time, locomotion, motion manifold, conditional transitioning, in-betweening, deep learning

ACM Reference Format:
Xiangjun Tang, Linjun Wu, He Wang, Bo Hu, Xu Gong, Yuchen Liao, Songnan Li, Qilong Kou, and Xiaogang Jin. 2023. RSMT: Real-time Stylized Motion Transition for Characters. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings (SIGGRAPH '23 Conference Proceedings), August 06–10, 2023, Los Angeles, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3588432.3591514

1 INTRODUCTION

We investigate an under-explored motion generation problem: styled online in-between motion synthesis. Rather than controlled style variation, our research focuses on maintaining a specific style while filling motions between keyframes, avoiding collapse to simple neutral motion, and has important applications in computer animation and games. It consists of two key elements that can potentially contradict with one another. Online in-between motion synthesis aims to quickly compute natural high-quality motions that satisfy the given constraints e.g. specified by frames [Harvey and Pal 2018; Harvey et al. 2020], while stylized motion generation focuses on motions that are visually different but with the same semantics (e.g. normal and zombie walking) [Dong et al. 2020; Yin et al. 2023]. Overall, the former attempts to find the optimal (in a broad sense) motion while the latter explores the motion diversity.

Existing research mostly investigates the above two problems separately [Mason et al. 2022, 2018; Oreshkin et al. 2022; Qin et al. 2022]. Given the initial and end frame(s), in-between motions can be formulated into complex optimization problems [Wang et al. 2015]. If data is available, it can be solved by searching in structured data [Kovar et al. 2008; Min and Chai 2012; Shen et al. 2017], or maximizing the likelihood or a posteriori [Li et al. 2013]. In deep learning, it can also be formulated into a motion manifold learning or control problem [Chen et al. 2020; Holden et al. 2016; Li et al. 2021; Petrovich et al. 2021]. In parallel, motion style has also been separately investigated. Motion can be stylized via optimization [Hsu et al. 2005], or learning parametric or non-parametric models on limited data [Brand and Hertzmann 2000; Wang et al. 2007]. More recently, deep learning has been employed to extract and transfer the style features of motions [Du et al. 2019; Park et al. 2021; Wen et al. 2021].

Combining in-between and style motion needs to address several challenges simultaneously: generation speed, motion quality, style diversity, and synthesis controllability. The first two challenges dictate that simple models need to be employed and strictly rule out post-processing. Existing approaches are either too slow, rely on post-processing [Aberman et al. 2020; Duan et al. 2021; Jang et al. 2022] or can only handle limited amount of data [Chai and Hodgins 2007]. Further, existing research rarely considers the latter two challenges together, either focusing on control without style [Tang et al. 2022] or uncontrolled style motions [Mason et al. 2022, 2018]. Enforcing control can easily break the intended style and vice versa. Table 1 shows the high level differences between our method and the literature.

Table 1: High level comparison between our method and existing methods.

	Speed	Quality	Style	Controllability	large dataset
Optimization	×	×	✓	✓	n/a
Traditional in-between	✓	✓	×	✓	×
Traditional style	✓	✓	✓	×	×
Deep learning (in-between)	✓	✓	×	✓	✓
Deep learning (style)	✓	✓	✓	×	✓
Ours	✓	✓	✓	✓	✓

To this end, we propose a novel method, which can generate high-quality styled in-between motions in real time, given the starting frame, the end frame, the time duration, and a motion with the target style. Inspired by the work in motion manifold/representation learning [Du et al. 2019; Ling et al. 2020] and conditional generation in images/motion [Huang and Belongie 2017; Starke et al. 2021], our key insight is to decouple the source of generation from the control of synthesis. Similar to representation learning, a good motion source should provide high-quality motions with varying styles. Furthermore, the extracted motion and style features should be ready for controlled synthesis, as a downstream task. This observation naturally leads to two key components in our system: a general motion manifold representation and a style motion sampler incorporating control. Moreover, such a design also allows the manifold and the sampler to be trained under different settings, e.g. the manifold trained on large unstyled/not-strongly-styled datasets widely available, and the sampler trained on smaller/dedicated datasets with desired styles, with little or no overlapping with the manifold data. This is ideal because the manifold can be pre-trained and shared, while the sampler is easily adaptable.

Our manifold model is an autoencoder that encodes motion transition randomness [Ling et al. 2020; Tang et al. 2022] and motion phases [Starke et al. 2022]. The autoencoder encodes motions into a latent space conforming to a Gaussian distribution, which together with the extracted phase information jointly serve as a good source. A motion sampler based on recurrent neural networks is then employed to sample from the source, with specified target frames, time duration and style, avoiding generating only the most probable motions such as [Harvey et al. 2020; Tang et al. 2022]. We exhaustively evaluate our method on 100STYLE dataset [Mason et al. 2022] and compare it with the most recent baseline methods [Harvey et al. 2020; Tang et al. 2022] under multiple metrics regarding motion quality, controllability, style diversity, etc. Our method proves to be fast, high-quality, versatile, and controllable.

Our main contributions can be summarized as follows:

- We present a novel online framework for styled real-time in-between motion generation without post-processing.
- We propose a new method to combine styles and controllability in motion generation.
- We propose a new model that has strong generalization capacity in both motion control and stylization.

2 RELATED WORK

2.1 In-between motion generation.

In-between motion synthesis can be regarded as motion planning [Arikan and Forsyth 2002; Beaudoin et al. 2008; Levine et al. 2012; Safonova and Hodgins 2007; Wang and Komura 2011; Wang et al. 2013], which solves complex optimization problems with various constraints. Data-driven methods are faster and produce more natural motions by searching in structured data, such as motion graphs [Kovar et al. 2008; Min and Chai 2012; Shen et al. 2017], but there is a trade-off between the diversity of animation and the space complexity. Recently, deep neural networks [Holden et al. 2020] leverage compressed data representations to reduce the space complexity.

Learning-based online in-between motion synthesizing can be formulated as motion manifold learning problems [Chen et al. 2020; He et al. 2022; Holden et al. 2016; Li et al. 2021; Petrovich et al. 2021; Rempe et al. 2021; Wang et al. 2019] with time-space constraints [Harvey and Pal 2018; Harvey et al. 2020]. Besides, explicitly modeling the low-level motion transition as a motion manifold [Tang et al. 2022] improves the transition generalization for unseen control. However, without style control explicitly, the character reaches the target by the most probable motion, which might break the intended style.

Offline methods can be considered motion completion problems [Hernandez et al. 2019; Kaufmann et al. 2020], which can be solved by convolution neural networks or transformers [Duan et al. 2021; Oreshkin et al. 2022; Qin et al. 2022]. Recent research employs generative models, such as diffusion models [Tevet et al. 2022] or generative adversarial networks [Li et al. 2022] to fill the missing frame with the specific actions by user control. However, the models give little attention to time efficiency, so it's hard to employ them for real-time applications.

2.2 Motion style transfer

One solution to motion style transfer models the style variations between two motions [Hsu et al. 2005]. Some research employs data-driven methods, such as Bayesian Networks [Ma et al. 2010] or mixtures of autoregressive models [Xia et al. 2015]. Unuma et al. [1995] and Yumer and Mitra [2016] extracted style features between independent actions by transforming the motion into the frequency domain, but they largely handle relatively small amounts of data. Recent deep learning methods model the style directly. The style can be represented as one-hot embedding [Smith et al. 2019], where the style is stored as parameters of the neural network. However, one-hot embedding representation cannot be generalized to unseen styles. To solve this, Mason et al. [2018] proposed a residual block to model the style. Further, recent methods [Aberman et al. 2020; Jang et al. 2022] extract the style feature as the latent variable explicitly, making zero-shot learning available. When the content sequence is missing for in-between motion generation, the style transfer problem can be transformed into a style-guided generation problem. The most related work synthesizes motion sequences with the style feature introduced by the FiLM block [Mason et al. 2022]. However, generating the stylized in-between sequence that satisfies control signals is still an unsolved problem.

3 METHODOLOGY

A human motion with T frames $\mathbf{M} = \{\mathbf{s}^{0\sim T}\}$ can be represented as a series of skeletal poses $\mathbf{s}^t \in \mathbb{R}^{N\times D}$, where N is the joint number and D is the degrees of freedom per joint at time $t \in [0, T]$. An often employed assumption on human motion is a linear dynamical system [Li et al. 2002; Pavlovic et al. 2000]:

$$s^{t+1} = s^t + f_1(h^{t+1}, s^t) \text{ and } h^{t+1} = f_2(h^t, s^t), \tag{1}$$

where h is a latent variable and governs the dynamics. When f_1 and f_2 are stochastic, we can model the joint probability of M and $H = \{h^{0\sim T}\}$ as:

$$\begin{aligned} P(M,H) &= P(s^{1\sim T-1}, h^{1\sim T}|s^T, s^0, h^0, k)P(s^0)P(h^0)P(s^T)P(k) \\ &= P(s^0)P(h^0)P(s^T)P(k) \\ &\prod P(s^{t+1}|s^T, s^0, h^0, h^{t+1}, s^t)P(h_{t+1}|s^T, s^0, h^0, k, h^t, s^t), \end{aligned} \tag{2}$$

where s^0, h^0 and s^T can be seen as control variables for in-between motion generation and k is a summarizing style code. The motion dynamics is governed by the last line in Eq. (2). Here we employ $h^t = \{z^t, v_h^t, p^t\}$, where z^t is a latent variable to govern the stochasticity of the transition of h, v_h^t is the hip velocity and orientation, p^t is a phase-related feature. We omit s^0 and h^0 for simplicity in the rest of the paper. Learning can be conducted through maximizing Eq. (2) which is decomposed into two parts: the motion manifold $P(s^{t+1}|s^T, h^{t+1}, s^t)$ and the sampler $P(h_{t+1}|s^T, h^t, s^t, k)$. We model both by deep neural networks. In summary, the sampler samples the stylized transition motion from the manifold by auto-regressively predicting the phase, hip, and latent variable. Below, we introduce their high-level design and refer the readers to the supplementary material for network details.

3.1 The Motion Manifold

Learning the motion manifold requires specifying the model for $P(s^{t+1}|s^T, h^{t+1}, s^t)$. First, instead of learning the next pose s^{t+1} directly, we learn a pose change $\Delta s^{t+1} = s^{t+1} - s^t$, as given a fixed time interval, this is equivalent of learning a velocity profile which captures the dynamics better [Martinez et al. 2017]. Next, we drop k and s^T from h^{t+1}. This is because k is a summarizing style code and s^T is the target state, while we aim to make the manifold more focused on learning local natural motions with any style. So $P(s^{t+1}|s^T, h^{t+1}, s^t) = P(\Delta s^{t+1}|v_h^{t+1}, p^{t+1}, z^{t+1}, s^t)$. Directly learning it is not possible as only Δs^{t+1}, v_h^{t+1} and s^t are observable and p^{t+1} and z^{t+1} need to be learned as latent variables. Therefore, we propose to use an autoencoder to learn the mapping while regularizing p^{t+1} and z^{t+1}.

The architecture of the autoencoder is shown in Fig. 2. The input is two consecutive frames and the output is the pose change. The encoder encodes the input into z regularized by $z \sim N(0, 1)$ to capture the transition stochasticity. The decoder recovers motions based on z, the style related phase features p^{t+1}, the current frame s^t, and the future hip velocity and orientation v_h^{t+1}. v_h^{t+1} is a strong indicator of the next frame in terms of the general motion trend [Tang et al. 2022], especially on the lower body where the motion diversity is smaller compared with the upper body. However, v_h^{t+1} is insufficient to include style-related information, so we use another phase-related feature p^{t+1}.

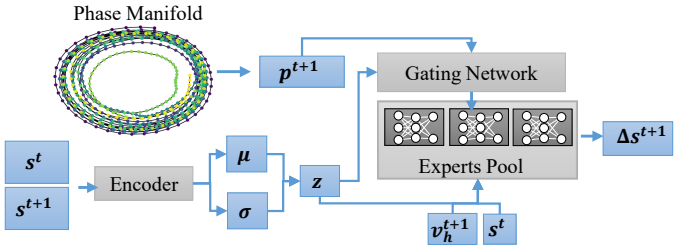

Figure 2: During training, the encoder takes s^t and s^{t+1} and generates the mean value μ and log variance σ of the Gaussian distribution. Then the latent variable z is calculated by reparameterization. Next, the gating network takes the phase sampled from the phase manifold and the latent variable z to generate the blending coefficients for experts. Finally, the blended expert takes the current frame s^t, the future root v_h^{t+1}, and the latent vector z to predict the pose change.

The phase feature in the latent space reveals well-behaved motion patterns [Starke et al. 2022]. For our problem, it is complementary to v_h^{t+1} in that it describes a continuous change of motion in patterns, which is highly related to styles [Yumer and Mitra 2016]. Following [Starke et al. 2022], we train a periodic autoencoder to extract a multi-dimensional phase manifold. The resulting phase vectors on this manifold are represented by a signed phase shift S, which when multiplied by 2π represents the angle of the phase vector, and an amplitude A, representing the phase vector magnitude:

$$p_{2i-1} = A_i \cdot sin(2\pi \cdot S_i), \quad p_{2i} = A_i \cdot cos(2\pi \cdot S_i), \tag{3}$$

where i is the dimension index. After encoding, the decoder aims to recover the velocity, which is designed to be generative and learns $P(\Delta s^{t+1}|v_h^{t+1}, p^{t+1}, z^{t+1}, s^t)$. z^{t+1} is drawn from $N(\mu, \sigma)$. p^{t+1} is sampled from the phase manifold. Both are fed into a gating network for a Conditional Mixtures of Experts (CMoEs), conditioned on the current frame s^t, the future hip feature v_h^{t+1} and z^{t+1}. The reason to employ a CMoEs here is that the transition distribution itself is multimodal [Holden et al. 2017; Wang et al. 2019] and this multimodality is further amplified with styles. A single network would average the data. In the CMoEs, each expert is expected to learn one mode and a combination of them can capture the multi-modality, with their weighting computed from z^{t+1} and p^{t+1}. Here z^{t+1} and p^{t+1} also play different roles. z^{t+1} governs the stochastic state transition of the linear dynamical system (Eq. (1)) while we find p^{t+1} is a good continuous representation that encodes motion styles. This also means our motion manifold implicitly learns style-related features which can be utilized later by the sampler.

3.1.1 Losses. We minimize $L = L_{rec} + \beta L_{kl} + L_{foot}$ where:

$$L_{rec} = \frac{1}{ITND} \sum ||M_i - M_i'||_2^2, \tag{4}$$
$$L_{kl} = -0.5 \cdot (1 + \sigma - \mu^2 - e^{\sigma}), L_{foot} = \frac{1}{ITN_f} \sum ||f_v' \delta(f_v)||_2^2,$$

where I, T, N, D, and N_f are the number of data samples, the motion length, the number of joints, the degrees of freedom per joint, and the number of foot joints that contact on the ground. M_i and M_i' are the ground-truth and predicted motion. Overall, L_{rec} is a reconstruction loss. L_{foot} penalizes foot sliding, where f_v' is the predicted foot velocity and $\delta(f_v) \in [0, 1]$ is the probability of ground-truth foot contact for each foot in each frame, which is defined by:

$$\delta(f_v) = \begin{cases} 1, & f_v \le 0.5, \\ 0, & f_v \ge 1.0, \\ 2t^3 - 3t^2 + 1, & t = 2(f_v - 0.5), f_v \in (0.5, 1.0). \end{cases} \tag{5}$$

L_{kl} is the KL-divergence between the z distribution parameterized by μ and σ and $N(0, 1)$, which we reparameterize following [Kingma and Welling 2013]. β is a hyper-parameter balancing between quality and generalization, which is set to 0.001 [Ling et al. 2020; Tang et al. 2022]. Note we intentionally employ a relatively small β to ensure the motion quality in the manifold. Later experiments will show that the generalization ability, which would be otherwise lost, will be compensated by the sampler.

3.2 The Motion Sampler

After training, the manifold is captured by the decoder and ready to be sampled. We learn the desired motions by learning the transition distribution $P(h_{t+1}|s^T, h^t, s^t)$. The sampler's design is shown in Fig. 3. The sampler takes as input the current frame s^t, the target frame s^T, the current phase vector p^t and the styled motion M_s with the target style. Then it outputs the latent vector $h^{t+1} = \{p^{t+1}, z^{t+1}, v_h^{t+1}\}$ which can then be fed into the manifold motion to sample the next frame.

Specifically, we first extract a summarizing style code k from M_s by the style encoder based on convolutional neural networks. k has a temporal dimension which we find is crucial in sampling, where the style information for each frame is lost in some earlier work [Aberman et al. 2020; Park et al. 2021]. Next, k is combined with the current frame s^t in a style embedding after s^t is pulled through a state encoder. This embedding co-embeds a single frame and a style code into a common space. In parallel, the target frame s^T and the offset $s^T - s^t$ are individually encoded and concatenated, then perturbed by z_{noise}. This term captures the motion randomness in terms of completing the motion at the current time. The randomness is time-varying, i.e. small when t is close to T as the remaining motion is short, and bigger otherwise:

$$z_{noise} = \lambda N(0, 0.5) \ , \lambda = clamp\left(\frac{T - t - t_{zero}}{t_{period} - t_{zero}}\right) \in (0, 1), \tag{6}$$

where $t_{zero} = 5$ and $t_{period} = 30$ are parameters. So far, combining encoded information, we have:

$$\begin{aligned} z_h^{t+1} &= \Phi_{LSTM}(c^t, z_h^t), \\ c^t &= (E_{sty}(k, E_{stat}(s^t)) + z_{dt}, h_{target}^t), \\ h_{target}^t &= (E_{tar}(s^T), E_{off}(s^T - s^t)) + z_{noise} + z_{dt}, \\ z_{dt,2i} &= \sin\left(\frac{dt}{10000^{2i/d}}\right), \quad z_{dt,2i+1} = \cos\left(\frac{dt}{10000^{2i/d}}\right), \end{aligned} \tag{7}$$

where E_{sty}, E_{stat}, E_{tar}, E_{off}, and Φ_{LSTM} are the style embedding, state encoder, target encoder, offset encoder, and long short-term memory network with cell state c^t, respectively. d represents the dimension of z_{dt} and $i \in [0, ..., d/2]$ represents the dimension index. The Φ_{LSTM} predicts the next state z_h^{t+1} which is used to

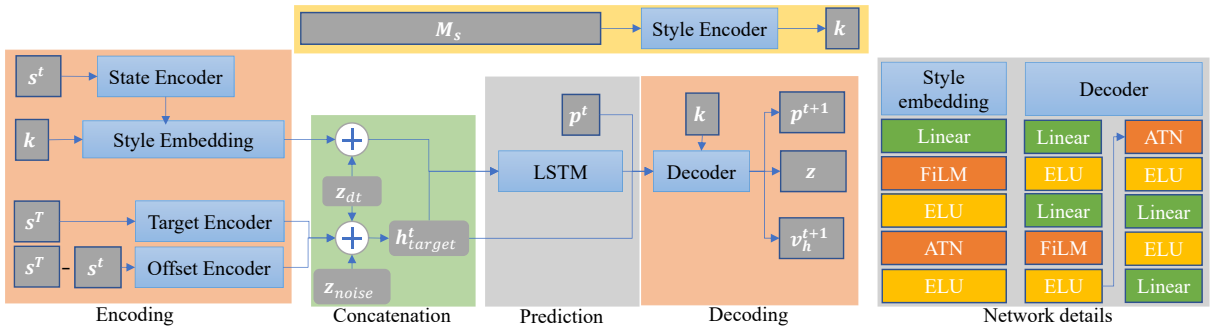

Figure 3: The motion sampler is divided into four stages: encoding, concatenation, prediction, and decoding. The gray boxes represent the data and the blue ones represent network blocks. The merge of arrows represents vector concatenation.

predict $h^{t+1} = \{z^{t+1}, p^{t+1}, v_h^{t+1}\} = D(k, (p^t, z_h^{t+1}, h^t_{target}))$, which is further fed into the manifold model to sample the next frame. Instead of predicting the next phase directly [Starke et al. 2022], we predict an intermediate next phase $\hat{p}^{t+1}$, and the amplitude $\hat{A}^{t+1}$ and the frequency $\hat{F}^{t+1}$ separately. Then another intermediate phase vector is computed as:

$$\tilde{p}^{t+1} = \hat{A}^{i+1} \cdot (R(\theta) \cdot p^t), \theta = \Delta t \cdot 2\pi \cdot \hat{F}^{t+1}, \tag{8}$$

where Δt is the time step, R is a 2D rotation matrix. Then we interpolate the angles of $\tilde{p}^{i+1}$ and $\hat{p}^{i+1}$ with spherical linear interpolation with weight 0.5, and linearly interpolate the amplitude with weight 0.5 for the final prediction p^{t+1}.

3.2.1 Losses. To train the sampler, we minimize $L = L_{rec} + L_{last} + +L_{foot} + L_{phase}$ where:

$$\begin{aligned} L_{rec} &= \frac{1}{ITND} \sum ||M_i - M_i'||_1, L_{last} = \frac{1}{IND} \sum ||s^T - (s^T)'||_1, \\ L_{phase} &= \frac{1}{ITN_p} \sum (||A^t - \hat{A}^t||_2^2 + ||F^t - \hat{F}^t||_2^2) \\ &+ \frac{1}{2ITN_p} \sum (||p^t - \hat{p}^t||_2^2 + ||p^t - \tilde{p}^t||_2^2), \end{aligned} \tag{9}$$

where I, T, N, D, and N_p are the number of data samples, the motion length, the number of joints, the degrees of freedom per joint, and the number of phase channels. L_{rec} is the reconstruction loss of motion. L_{last} emphasizes the predicted last frame should be the same as the target frame. L_{foot} is defined in Equation 4, penalizing the foot skating artifacts. L_{phase} is the reconstruction loss of the phase, where F^t is calculated as the difference between two consecutive signed shift:

$$F^t \begin{cases} S^t - S^{t-1}, & S^t - S^{t-1} \in [-0.5, 0.5] \\ S^t - S^{t-1} + 1, & S^t - S^{t-1} \in [-1, -0.5). \\ S^t - S^{t-1} - 1, & S^t - S^{t-1} \in (0.5, 1.0] \end{cases} \tag{10}$$

4 IMPLEMENTATION

Data Formatting. We use the 100STYLE motion dataset [Mason et al. 2022] and retarget the motions to a skeleton with 23 joints by removing all the wrist and thumb joints. Further, we subsample the motion sequences to 30 fps and augment them by mirroring and temporal random cropping [Jang et al. 2022]. We use 120-frame clips for M_s and 60-frame clips with 20-frame overlapping for the motion manifold. We orient the motions so that first frame faces toward the X-axis [Holden et al. 2016]. However, instead of using features in local space [Li et al. 2022] that is invariant to the global translation and rotation of the motion, our experiments show that representing the starting and target pose in the same coordinate system improves the performance. So we use the global joint orientation by rotations along a forward and up vector $\mathbf{r} \in \mathbb{R}^6$ [Zhang et al. 2018]. Overall, a motion $M \in \mathbb{R}^{23\times12\times60}$ includes the global joint position ($\mathbb{R}^3$), velocity ($\mathbb{R}^3$), and rotation ($\mathbb{R}^6$). The style motion $M_s \in \mathbb{R}^{23\times12\times120}$ is chosen at random from a set of data with the same style as M.

Training. We chose a 25-frame window at random from the 60-frame clip for one training step instead of using the whole sequence, which significantly speeds up the training convergence.

We use AmSgrad with parameters ($\beta_1 = 0.5, \beta_2 = 0.9$) and a learning rate $1e-3$. To train the sampler, we shuffle all the style sequences randomly. To ensure that our model is robust to different time durations, we randomly sample a sequence whose length is 20 in the beginning and increases linearly to 40 in each epoch, similar to [Harvey et al. 2020]. We employ AMSGrad with the same setting as before, with a weight decay $1e-4$ for training style encoder to avoid overfitting and 0 for other modules. More information is available in the supplementary material (SM).

5 EXPERIMENTS AND RESULTS

We conduct our experiments on a PC with an Nvidia RTX 3090 graphics card, with an Intel I7-11700K CPU and 32G memory. Our method takes on average 1.7 ms to synthesize one frame.

Data split. 100STYLE contains 100 styles. Since our manifold and sampler can be trained/tested on different datasets, we set up two testing protocols: style-overlapping and style-no-overlapping. We use the last 10 styles as the style-no-overlapping testing data, and use 10% of each style in the first 90 styles as the style-overlapping testing data. The remaining is the training data.

5.1 Metrics

We employ three sets of metrics to evaluate the motion quality, the synthesis controllability and the diversity, under various control signals. We first test our model in reconstructing the missing frames of variable lengths (10 frames, 20 frames, and 40 frames) and measure the reconstruction accuracy. Then we change the time duration and

the locations of the target frames (spatial/temporal control), which requires the character to reach the target via different motions. The locations of the target frame and the time duration are changed by: $(x^T)' = x^0 + d \cdot (x^T - x^0)$ and $t'_{durations} = dt \cdot t_{durations}$, where x^T is the location of the target frame projected on the ground, $t_{durations}$ is the time durations of the missing frames. d and dt are the ratio parameters. We set $d = 2$ and $d = -1$ to place the target frame before and after the starting frame. We set $dt = 2$ and $dt = 0.5$ to slow down and speed up the motion.

We measure controllability using three different metrics. When infilling motion, we use the averaged L2 distance of global joint positions and the Normalized Power Spectrum Similarity (NPSS) in the joint angle space to calculate reconstruction accuracy [Harvey et al. 2020]. When the ground truth for spatial/temporal control is unavailable, we measure controllability by determining whether the predicted last frame matches the target. In both cases, we measure foot skating [Zhang et al. 2018] using the averaged foot velocity v_f when the foot height h is within a threshold $H = 2.5$: $L_f = v_f \cdot clamp(2 - 2^{h/H}, 0, 1)$.

To evaluate the diversity, we generate ten samples for each combination of a motion sequence and a style sequence, then calculate the L2 distance between the global joint positions of any two sequences.

Quantitatively measuring style is hard. A possible solution is the Fréchet Motion Distance (FMD) [Jang et al. 2022; Yin et al. 2023], which compares the distributions of the generated sequences and the dataset. However, the distribution under the spatial/temporal control shifts the distribution significantly. So we only use FMD when the time duration and target location do not change, in which the style similarity is highly related to the distribution similarity. The results of spatial/temporal control are shown in the video.

To compute the FMD, we first train a style classifier by modifying the style encoder in [Jang et al. 2022] by adding a pooling layer and a softmax layer. The pooling layer removes the temporal and the joint axis of the output from the style encoder. Then the softmax layer transforms the latent to the one-hot embedding of the style. The FMD is then calculated between the generated sequences and the ground truth in the latent space after the pooling layer.

5.2 Manifold Generalization

One advantage of our model is that the manifold and sampler do not have to be trained on the same dataset, which makes it possible to train the manifold on a general dataset with more data samples and train the sampler on a dedicated style dataset. To validate this, we further split the training dataset into two subsets named A (the first 46 styles) and B (the following 44 styles) to ensure there is no style overlapping. We name the testing dataset C (the last 10 styles).

We show exhaustive experiments in Tab. 2. Both the reconstruction and the NPSS are better when the sampler is trained and tested on the same dataset, understandably. Similar observation can be made in foot skating except that AA on B is better than AB on B but by a small margin. Overall, these metrics are not severely affected regarding the dataset for manifold training, verifying that manifold, which being high-quality, can be trained as a somewhat independent motion source, which gives great flexibility in training and reduces the need to capture large amounts of dedicated style motions.

Table 2: Manifold and Sampler trained on different data: "BA on C" represents the manifold trained on B and the sampler trained on A and the full model tested on C. Note there is no style overlap among A, B, and C. Fuller results are in the SM.

	L2 norm of global position		
Frames	10	20	40
AA on A (BA on A)	**0.59** (0.64)	**0.76** (0.80)	**1.31** (1.38)
BB on A (AB on A)	0.78 (0.80)	1.15 (1.19)	1.85 (1.92)
AA on B (BA on B)	0.63 (0.62)	0.94 (0.89)	1.63 (1.53)
BB on B (AB on B)	**0.53** (0.58)	**0.68** (0.74)	**1.11** (1.23)
AA on C (BA on C)	**0.80** (0.82)	1.21 (**1.18**)	**1.95** (1.97)
BB on C (AB on C)	0.97 (0.98)	1.56 (1.51)	2.53 (2.44)
	100×**NPSS**		
AA on A (BA on A)	**0.435** (0.502)	**1.640** (1.828)	9.850 (**9.576**)
BB on A (AB on A)	0.518 (0.534)	2.195 (2.141)	13.408 (11.461)
AA on B (BA on B)	0.549 (0.612)	2.838 (2.395)	20.981 (18.043)
BB on B (AB on B)	**0.380** (0.504)	**1.442** (1.922)	**10.067** (12.387)
AA on C (BA on C)	**0.634** (0.743)	**3.798** (4.057)	**16.784** (18.372)
BB on C (AB on C)	0.872 (1.032)	5.046 (5.541)	26.806 (19.864)
	Foot skate		
Ground Truth on A	0.161	0.167	0.167
AA on A (BA on A)	**0.171** (0.202)	**0.197** (0.222)	**0.297** (0.321)
BB on A (AB on A)	0.217 (0.199)	0.250 (0.230)	0.353 (0.340)
Ground Truth on B	0.184	0.181	0.186
AA on B (BA on B)	**0.211** (0.256)	**0.218** (0.232)	0.324 (0.346)
BB on B (AB on B)	0.213 (0.221)	0.231 (0.249)	**0.309** (0.343)
Ground Truth on C	0.271	0.269	0.255
AA on C (BA on C)	**0.196** (0.210)	0.264 (**0.241**)	0.336 (**0.302**)
BB on C (AB on C)	0.235 (0.218)	0.276 (0.279)	0.345 (0.358)
	Diversity		
AA on A (BA on A)	0.869 (0.908)	2.172 (2.210)	7.194 (7.423)
BB on A (AB on A)	0.899 (**0.938**)	2.283 (**2.303**)	7.446 (**7.587**)
AA on B (BA on B)	0.856 (**0.892**)	2.189 (**2.191**)	7.471 (**7.612**)
BB on B (AB on B)	0.834 (0.875)	2.049 (2.096)	6.738 (6.989)

In terms of diversity, a higher diversity is achieved in general when the manifold and the sampler are trained on different data. It is challenging to reproduce the original motion when sampling from another manifold, which drives the sampler to approximate it by stochastic sampling, essentially increasing the diversity. Besides, compared with the manifold, the sampler has a stronger influence on diversity, especially when trained and tested on different data. The diversity comes from the fact that the character is forced to explore more poses to reach the target.

FMD (Tab. 3) measures the distributional similarity between our model and the ground truth. Again, the sampler has a bigger influence, e.g. both AA and BA are better than AB on A. Further, although BA performs slightly worse than AA on A, the style effects do not differ much in visual observations. It is reasonable because the motion sampled from manifold A has a higher probability of having a similar distribution as dataset A. Similar results (in SM) are also observed on dataset C.

In terms of the controllability, the metrics on the last frame is mainly affected by the manifold, as shown in Tab. 3. The sampling from manifold A achieves better synthesis controllability in dataset A than from manifold B. However, there is no significant visual

Table 3: Manifold and Sampler trained on different data. The table shows FMD of 40 frames and the L2 norm of global position between the last predicted frame and the target frame.

	FMD (40 frames)			
Dataset A	BA	AA	AB	BB
	16.303	**10.359**	50.448	58.154
Dataset B	BA	AA	AB	BB
	45.898	62.092	14.964	**11.431**
Dataset C	BA	AA	AB	BB
	70.655	74.404	105.360	102.639
	L2 norm of global position of last frame			
Conditions	dt=2, d=1	dt=0.5, d=1	dt=1, d=2	dt=1, d=-1
AA on A (AB on A)	0.38 (**0.37**)	**0.42** (0.42)	**0.39** (0.40)	0.39 (**0.37**)
BB on A (BA on A)	0.43 (0.45)	0.48 (0.48)	0.46 (0.43)	0.45 (0.44)
AA on B (AB on B)	0.40 (0.37)	0.43 (0.40)	0.39 (0.37)	0.39 (0.36)
BB on B (BA on B)	**0.33** (0.35)	**0.36** (0.38)	0.33 (**0.33**)	**0.32** (0.34)

difference between them in most cases. For some poses that differ from the neutral pose in dataset A, sampling from manifold B cannot reconstruct the pose accurately, resulting in a gap in the target frame. Training the manifold on a more diverse dataset alleviates this problem. Overall, the manifold affects the controllability and mildly influences the quality while the sampler mainly influence the style and the generalization.

5.3 Comparisons

We compare our model with two in-between models: CVAE [Tang et al. 2022] and RTN [Harvey et al. 2020]. Results are shown in Tab. 4. Besides, for a fair comparison, an additional experiment was carried out in which we replaced the CVAE's sampler with our sampler so that it could accept the style code as input. The manifold design distinguishes the two methods. The comparison demonstrates that the phase manifold is critical for preserving the styled motion, as discussed in the SM's manifold discussion. Furthermore, we compare our method with and without the attention block, which is also detailed in the SM. From reconstruction and NPSS, CVAE performs overall slightly better than ours and both are better than RTN. Since the motion sampling of CVAE is strictly constrained within a latent Gaussian distribution, its NPSS is slightly better than ours while we achieves better reconstruction in general. However, when it comes to foot skating and motion diversity, our method undoubtedly outperforms all baseline methods. We argue that in styled motion generation, the latter two metrics are more important as the foot skating is a key indicator for any motion while the diversity serves the generation task better.

Further, without imposing style explicitly, RTN and CVAE simply learn the most likely motion given the past context and the target frame from the datasets. Sometimes they can generate motion with styles but only when there is a style that is dinstinctive from other styles without ambiguity in the data. When there are several similar styles, they tend to mix them incorrectly, which can be noticed in the visual comparison in the video. In addition, when the motion differs from the distribution of the training set by changing the time duration or locations, RTN and CVAE cannot generate vivid styled motions or generate visible artifacts. Please refer to Figs. 4, 5 in the figures only pages, as well as the accompanying video.

Table 4: Comparison on reconstruction, foot skating, and diversity metrics.

	L2 norm of global position			100×NPSS		
Frames	10	20	40	10	20	40
RTN	0.663	0.847	1.365	0.387	1.539	8.794
CVAE	**0.506**	0.692	1.224	**0.321**	**1.353**	**8.108**
Our method	0.525	**0.680**	**1.148**	0.384	1.507	9.659
	Foot skate			**Diversity**		
RTN	0.343	0.339	0.474	0.678	1.956	6.172
CVAE	0.217	0.210	0.284	0.867	1.791	5.560
Our method	**0.174**	**0.194**	**0.272**	**0.910**	**2.017**	**6.683**

Table 5: Comparison on the L2 norm of global position of last predicted frame and the target, foot skating metrics, under the conditions that change the time duration and locations of target frame of 40 missing frames.

	L2 norm of global position of the last frame			
Conditions	dt=2, d=1	dt=0.5, d=1	dt=1, d=2	dt=1, d=-1
RTN	0.688	0.599	0.751	0.599
CVAE	0.908	0.781	0.746	0.811
Our method	**0.292**	**0.358**	**0.302**	**0.303**
	Foot skate			
RTN	0.498	1.294	1.181	0.795
CVAE	**0.208**	**0.995**	0.875	0.664
Our method	0.307	1.018	**0.592**	**0.557**

Alternatively, in the absence of style specification, our method achieves similar high-quality results compared to CVAE, and both methods outperform RTN by motion quality, especially for the conditions that the time duration and the locations of the target frames are changed.

Varying transition duration. We also significantly speed up and slow down the velocity. Speeding up dictates that the character needs to make faster steps or larger strides to reach the target. In this situation, CVAE and our method adopt the same phase as the original sequence but with higher velocity. In contrast, RTN drifts to the target. During slowing down, CVAE generates motions where the character stays in the middle and waits without performing the stylized motion, while our method also slows down but still performs the styled motion or keeps moving but turns slightly if it moves to a location that is slightly different from the target, e.g. overshooting. The reason for such motions is that a longer duration drives more cycles in the the phase manifold in our model, so that it avoids simply drifting/idling in the middle like the motions by RTN and CVAE. More visual comparisons can be found in the video.

Different target locations. To test drastic spatial control, we run two experiments in which the target location is set to be further away in front of and behind the character, respectively. When the target is further away, the CVAE fills the gap with more small footsteps or fewer bigger ones. Our method uses the bigger steps to fill the gap while preserving the same phase as the original sequence. However, RTN usually performs at the same pace as the ground truth but fills the distance gap by drifting. When the target is behind the character, which is drastically different from the data, the character must change the velocity direction during the motion, causing RTN to easily drift. But our method and CVAE keep foot contact with the ground.

5.4 Generation on Unseen styles

Compared to earlier motion in-between methods [Harvey et al. 2020; Tang et al. 2022], which cannot generalize to unseen styles, we have shown our method can generalize well in style-no-overlapping results (Tab. 2 and Tab. 3). In addition, the design of our model makes fine-tune on limited data easy, so we do not have to blindly extrapolate on new styles. This is important in application as existing data might not give enough samples in the style space [Ji et al. 2021] and it is highly desirable if only a few sequences of a new style are needed for models to generate diversified motions with that style.

To validate this, we fine-tune the style encoder and the linear layers of the FiLM and ATN blocks while keeping all other parameters fixed. We choose the "TwoFootJump" style in the testing dataset as it is significantly different from the training set. The "TwoFootJump" data consists of 8 types. We sample merely one sequence per type and augment the sequences by mirroring and temporal random cropping, leading to 24 sequences for fine-tuning. After fine-tuning, the character can reach the target by jumping rather than walking as before fine-tuning, which captures the new style perfectly. The accompanying video shows the visual results.

6 LIMITATIONS, CONCLUSIONS AND FUTURE WORK

One limitation is our model still tends to generate the sequences that have similar distribution as the training data, similar to other data-driven methods. When the specified style severely contradicts with the space-time constraints, our model favors the control rather than the style, especially if the remaining time is not long enough to show that style. For example, the character should kick strongly to deliver the style, but it might raise its leg slightly if the remaining time is insufficient and the target pose does not raise the leg. However, we argue in most application scenarios, control is more important.

In summary, we have proposed a novel learning framework for styled real-time in-between motion generation (RSMT), consisting of two critical, independent components: a general motion manifold acting as a high-quality motion source and a motion sampler generating motions with user-specified style and control signals. Exhaustive evaluation proves that our model has a strong generalization capacity in motion control and stylization. Our method generates high quality styled motions and is general to unseen control signals. It also outperforms state-of-the-art methods. One future direction is to enable style transitions before reaching the target frame, which requires tunable weighting on control and varying styles simultaneously on the fly.

ACKNOWLEDGMENTS

Xiaogang Jin was supported by Key R&D Program of Zhejiang (No. 2023C01047), and the National Natural Science Foundation of China (Grant Nos. 62036010, 61972344). He Wang has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 899739 CrowdDNA.

REFERENCES

Kfir Aberman, Yijia Weng, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. 2020. Unpaired motion style transfer from video to animation. *ACM Transactions on Graphics* 39, 4 (2020), 1–12.

Okan Arikan and D. A. Forsyth. 2002. Interactive motion generation from examples. *ACM Transactions on Graphics* 21, 3 (2002), 483–490.

Philippe Beaudoin, Stelian Coros, Michiel van de Panne, and Pierre Poulin. 2008. Motion-motif graphs. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 117–126.

Matthew Brand and Aaron Hertzmann. 2000. Style machines. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. 183–192.

Jinxiang Chai and Jessica K. Hodgins. 2007. Constraint-based motion optimization using a statistical dynamic model. *ACM Transactions on Graphics* 26, 3 (2007), 8–es.

Wenheng Chen, He Wang, Yi Yuan, Tianjia Shao, and Kun Zhou. 2020. Dynamic future net: diversified human motion generation. In *Proceedings of the 28th ACM International Conference on Multimedia*. 2131–2139.

Yuzhu Dong, Andreas Aristidou, Ariel Shamir, Moshe Mahler, and Eakta Jain. 2020. Adult2child: motion style transfer using cyclegans. In *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*. 1–11.

Han Du, Erik Herrmann, Janis Sprenger, Klaus Fischer, and Philipp Slusallek. 2019. Stylistic locomotion modeling and synthesis using variational generative models. In *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*. 1–10.

Yinglin Duan, Tianyang Shi, Zhengxia Zou, Yenan Lin, Zhehui Qian, Bohan Zhang, and Yi Yuan. 2021. Single-shot motion completion with transformer. *arXiv:2103.00776 [cs]* (2021).

Félix G. Harvey and Christopher Pal. 2018. Recurrent transition networks for character locomotion. In *SIGGRAPH Asia 2018 Technical Briefs (SA '18)*. Association for Computing Machinery, 1–4.

Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. 2020. Robust motion in-betweening. *ACM Transactions on Graphics* 39, 4 (2020), 1–12.

Chengan He, Jun Saito, James Zachary, Holly Rushmeier, and Yi Zhou. 2022. NeMF: Neural Motion Fields for Kinematic Animation. In *NeurIPS*.

Alejandro Hernandez, Jurgen Gall, and Francesc Moreno-Noguer. 2019. Human motion prediction via spatio-temporal inpainting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7134–7143.

Daniel Holden, Oussama Kanoun, Maksym Perepichka, and Tiberiu Popa. 2020. Learned motion matching. *ACM Transactions on Graphics* 39, 4 (2020), 1–12.

Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-functioned neural networks for character control. *ACM Transactions on Graphics* 36, 4 (2017), 1–13.

Daniel Holden, Jun Saito, and Taku Komura. 2016. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics* 35, 4 (2016), 1–11.

Eugene Hsu, Kari Pulli, and Jovan Popović. 2005. Style translation for human motion. *ACM Transactions on Graphics* 24, 3 (2005), 1082–1089.

Xun Huang and Serge Belongie. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*. 1501–1510.

Deok-Kyeong Jang, Soomin Park, and Sung-Hee Lee. 2022. Motion puzzle: arbitrary motion style transfer by body part. *ACM Transactions on Graphics* 41, 3 (2022), 1–16.

Xu Ji, Razvan Pascanu, Devon Hjelm, Balaji Lakshminarayanan, and Andrea Vedaldi. 2021. Test sample accuracy scales with training sample density in neural networks. In *Proceedings of the 1st Conference on Lifelong Learning Agents*. 629–646.

Manuel Kaufmann, Emre Aksan, Jie Song, Fabrizio Pece, Remo Ziegler, and Otmar Hilliges. 2020. Convolutional autoencoders for human motion infilling. In *Proceedings of the 2020 International Conference on 3D Vision*. 918–927.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

Lucas Kovar, Michael Gleicher, and Frédéric Pighin. 2008. Motion graphs. In *ACM SIGGRAPH 2008 Classes (SIGGRAPH '08)*. 1–10.

Sergey Levine, Jack M Wang, Alexis Haraux, Zoran Popović, and Vladlen Koltun. 2012. Continuous character control with low-dimensional embeddings. *ACM Transactions on Graphics* 31, 4 (2012), 1–10.

Jiaman Li, Ruben Villegas, Duygu Ceylan, Jimei Yang, Zhengfei Kuang, Hao Li, and Yajie Zhao. 2021. Task-generic hierarchical human motion prior using vaes. In *Proceedings of the 2021 International Conference on 3D Vision*. 771–781.

Peizhuo Li, Kfir Aberman, Zihan Zhang, Rana Hanocka, and Olga Sorkine-Hornung. 2022. GANimator: neural motion synthesis from a single sequence. *ACM Transactions on Graphics* 41, 4 (2022), 1–12.

Wanyi Li, Jifeng Sun, Xin Zhang, and Yuanchang Wu. 2013. Spatial constraints-based maximum likelihood estimation for human motions. In *Proceedings of the 2013 IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC 2013)*. 1–6.

Yan Li, Tianshu Wang, and Heung-Yeung Shum. 2002. Motion texture: a two-level statistical model for character motion synthesis. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*. 465–472.

Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel van de Panne. 2020. Character controllers using motion VAEs. *ACM Transactions on Graphics* 39, 4 (2020), 1–12.

Wanli Ma, Shihong Xia, Jessica K Hodgins, Xiao Yang, Chunpeng Li, and Zhaoqi Wang. 2010. Modeling style and variation in human motion. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 21–30.

Julieta Martinez, Michael J Black, and Javier Romero. 2017. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2891–2900.

Ian Mason, Sebastian Starke, and Taku Komura. 2022. 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 (2022), 1–18.

Ian Mason, Sebastian Starke, He Zhang, Hakan Bilen, and Taku Komura. 2018. Few-shot learning of homogeneous human Locomotion Styles. *Computer Graphics Forum* 37, 7 (2018), 143–153.

Jianyuan Min and Jinxiang Chai. 2012. Motion graphs++: a compact generative model for semantic motion analysis and synthesis. *ACM Transactions on Graphics* 31, 6 (2012), 1–12.

Boris N Oreshkin, Antonios Valkanas, Félix G Harvey, Louis-Simon Ménard, Florent Bocquelet, and Mark J Coates. 2022. Motion inbetweening via deep Δ-interpolator. *arXiv preprint arXiv:2201.06701* (2022).

Soomin Park, Deok-Kyeong Jang, and Sung-Hee Lee. 2021. Diverse motion stylization for multiple style domains via spatial-temporal graph-based generative model. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 4, 3 (2021), 1–17.

Vladimir Pavlovic, James M Rehg, and John MacCormick. 2000. Learning switching linear models of human motion. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*. 942–948.

Mathis Petrovich, Michael J Black, and Gül Varol. 2021. Action-conditioned 3d human motion synthesis with transformer vae. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10985–10995.

Jia Qin, Youyi Zheng, and Kun Zhou. 2022. Motion in-betweening via two-stage transformers. *ACM Transactions on Graphics* 41, 6 (2022), 1–16.

Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J Guibas. 2021. Humor: 3d human motion model for robust pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 11488–11499.

Alla Safonova and Jessica K. Hodgins. 2007. Construction and optimal search of interpolated motion graphs. *ACM Transactions on Graphics* 26, 3 (2007), 106–es.

Yijun Shen, He Wang, Edmond S. L. Ho, Longzhi Yang, and Hubert P. H. Shum. 2017. Posture-based and action-based graphs for boxing skill visualization. *Computers and Graphics* 69, Supplement C (2017), 104–115.

Harrison Jesse Smith, Chen Cao, Michael Neff, and Yingying Wang. 2019. Efficient neural networks for real-time motion style transfer. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 2, 2 (2019), 1–17.

Sebastian Starke, Ian Mason, and Taku Komura. 2022. DeepPhase: periodic autoencoders for learning motion phase manifolds. *ACM Transactions on Graphics* 41, 4 (2022), 1–13.

Sebastian Starke, Yiwei Zhao, Fabio Zinno, and Taku Komura. 2021. Neural animation layering for synthesizing martial arts movements. *ACM Transactions on Graphics* 40, 4 (2021), 1–16.

Xiangjun Tang, He Wang, Bo Hu, Xu Gong, Ruifan Yi, Qilong Kou, and Xiaogang Jin. 2022. Real-time controllable motion transition for characters. *ACM Transactions on Graphics* 41, 4 (2022), 1–10.

Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. 2022. Human motion diffusion model. *arXiv preprint arXiv:2209.14916* (2022).

Munetoshi Unuma, Ken Anjyo, and Ryozo Takeuchi. 1995. Fourier principles for emotion-based human figure animation. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*. 91–96.

He Wang, Edmond SL Ho, and Taku Komura. 2015. An energy-driven motion planning method for two distant postures. *IEEE Transactions on Visualization and Computer Graphics* 21, 1 (2015), 18–30.

He Wang, Edmond SL Ho, Hubert PH Shum, and Zhanxing Zhu. 2019. Spatio-temporal manifold learning for human motions via long-horizon modeling. *IEEE Transactions on Visualization and Computer Graphics* 27, 1 (2019), 216–227.

He Wang and Taku Komura. 2011. Energy-based pose unfolding and interpolation for 3D articulated characters. In *Proceedings of the 4th International Conference on Motion in Games*. 110–119.

He Wang, Kirill A Sidorov, Peter Sandilands, and Taku Komura. 2013. Harmonic parameterization by electrostatics. *ACM Transactions on Graphics* 32, 5 (2013), 1–12.

Jack M Wang, David J Fleet, and Aaron Hertzmann. 2007. Multifactor gaussian process models for style-content separation. In *Proceedings of the 24th International Conference on Machine Learning*. 975–982.

Yu-Hui Wen, Zhipeng Yang, Hongbo Fu, Lin Gao, Yanan Sun, and Yong-Jin Liu. 2021. Autoregressive stylized motion synthesis with generative flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13612–13621.

Shihong Xia, Congyi Wang, Jinxiang Chai, and Jessica Hodgins. 2015. Realtime style transfer for unlabeled heterogeneous human motion. *ACM Transactions on Graphics* 34, 4 (2015), 1–10.

Wenjie Yin, Hang Yin, Kim Baraka, Danica Kragic, and Mårten Björkman. 2023. Dance style transfer with cross-modal transformer. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 5058–5067.

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

He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. 2018. Mode-adaptive neural networks for quadruped motion control. *ACM Transactions on Graphics* 37, 4 (2018), 1–11.

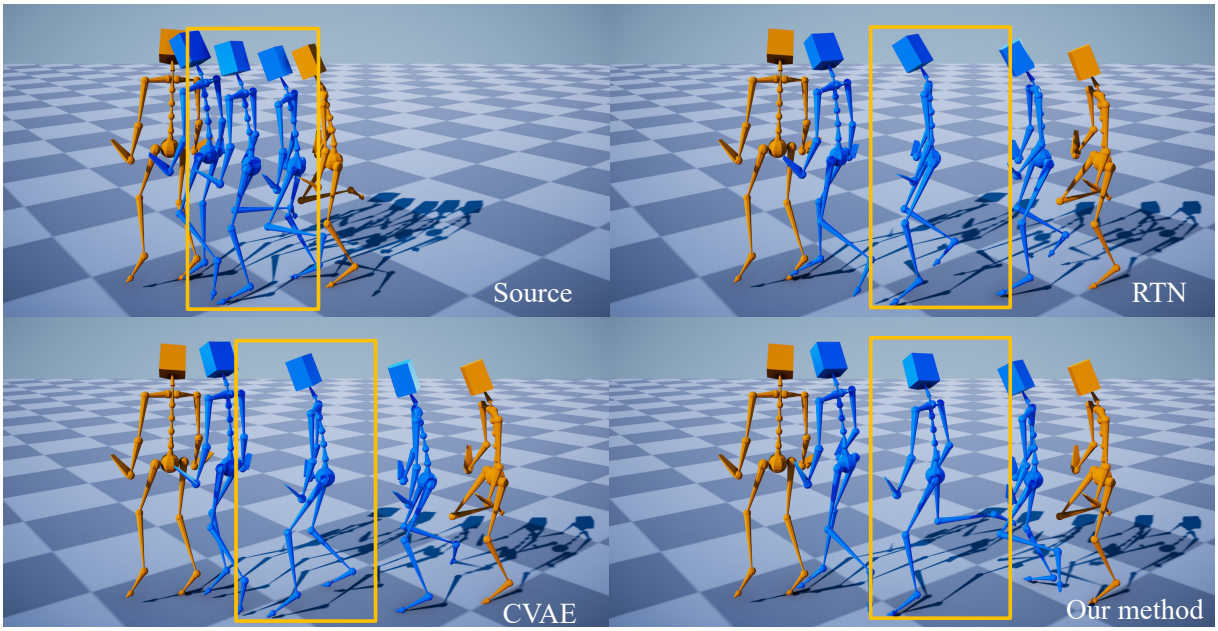

Figure 4: The target location is set to further away in front of the starting frame. Our method performs the desired stylized motion with bigger footsteps. Comparatively, the character synthesized by RTN and CVAE reaches the target via neutral walking. The yellow boxes highlight the most stylized pose.

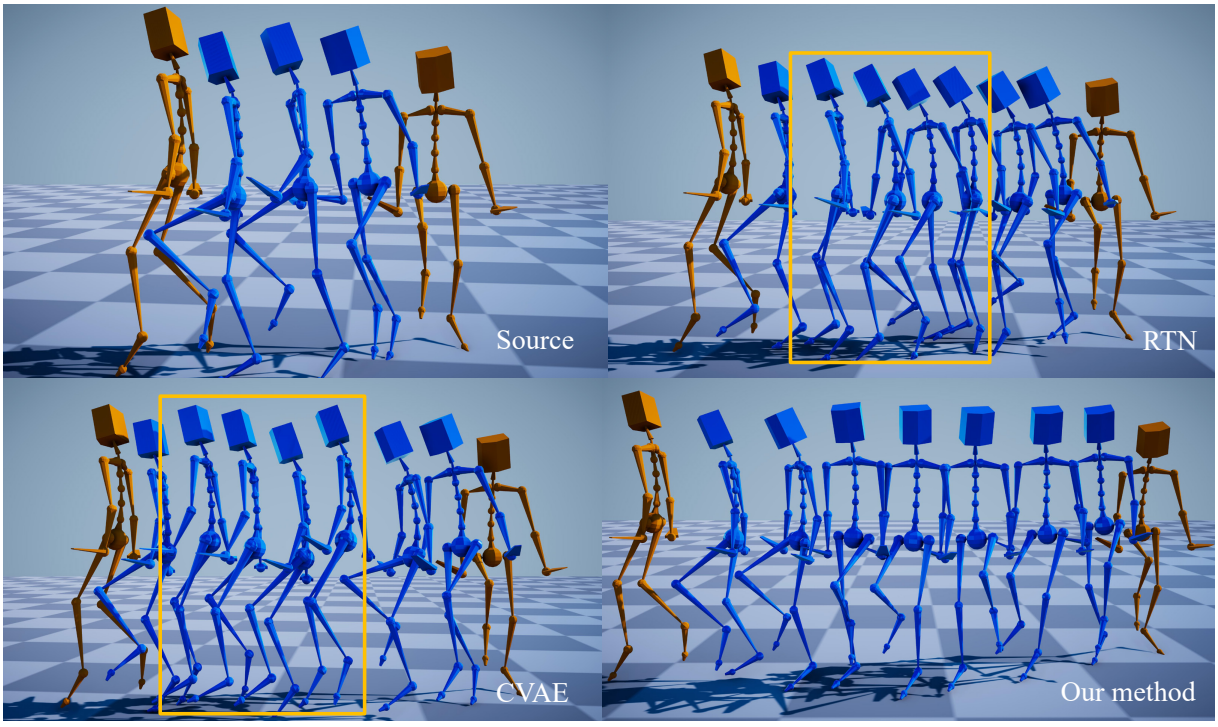

Figure 5: We set the time duration twice as long as the original (i.e. requiring a slowing down motion) and plot the key poses for easy viewing. RTN and CVAE remain in the middle and wait without performing the stylized motion (yellow box), whereas our method continues performing the stylized motion.