StyleVR: Stylizing Character Animations With Normalizing Flows

Bin Ji, Ye Pan, Yichao Yan, Ruizhao Chen, and Xiaokang Yang, *Fellow, IEEE*

***Abstract*—The significance of artistry in creating animated virtual characters is widely acknowledged, and motion style is a crucial element in this process. There has been a long-standing interest in stylizing character animations with style transfer methods. However, this kind of models can only deal with short-term motions and yield deterministic outputs. To address this issue, we propose a generative model based on normalizing flows for stylizing long and aperiodic animations in the VR scene. Our approach breaks down this task into two sub-problems: motion style transfer and stylized motion generation, both formulated as the instances of conditional normalizing flows with multi-class latent space. Specifically, we encode high-frequency style features into the latent space for varied results and control the generation process with style-content labels for disentangled edits of style and content. We have developed a prototype, StyleVR, in Unity, which allows casual users to apply our method in VR. Through qualitative and quantitative comparisons, we demonstrate that our system outperforms other methods in terms of style transfer as well as stochastic stylized motion generation.**

***Index Terms*—Character animation, motion generation, style transfer, normalizing flow, virtual reality.**

I. Introduction

GENERATING diverse and lifelike VR/AR character animations has been a topic of interest for a long time. Different approaches have been taken to tackle this challenge [1], [2], including the development of animation systems such as Motion Doodles [3], Spatial Motion Doodles [2] and ARAnimator [1], which aim to create characters in the virtual scene. However, these methods all need sophisticated procedures to control the interaction between characters and rough terrains of the virtual environment. Additionally, there is still a significant gap between characters with high-dimensional motions and 3-DoF or 6-DoF action control strategies, which remains under-explored. Consequently, creating compelling animations can be both time-consuming and costly.

Manuscript received 1 July 2022; revised 27 January 2023; accepted 6 March 2023. Date of publication 20 March 2023; date of current version 26 June 2024. This work was supported in part by Shanghai Sailing Program under Grant 20YF1421200, in part by the National Natural Science Foundation of China (NSFC) under Grant 62102255, in part by the National Natural Science Foundation of China (NSFC) under Grant 62201342 and in part by Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102. Recommended for acceptance by T. Komura. *(Corresponding author: Ye Pan.)*

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by Shanghai Jiao Tong University Research Ethics Committe.

The authors are with the Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: bin.ji@sjtu.edu.cn; whitneypanye@sjtu.edu.cn; yanyichao@sjtu.edu.cn; stelledge@sjtu.edu.cn; xkyang@sjtu.edu.cn).

Digital Object Identifier 10.1109/TVCG.2023.3259183

In order to improve the laborious manual processes, a promising and attractive option is to apply deep neural networks. Some success has been achieved by PFNN [4], NSM [5] and Local Motion Phases [6] in realizing lifelike human movements and precise scene interactions. However, these methods mainly focus on the motion qualities, leading to the synthesis of quasi-periodic motions.

Recently, the advancement of generative models, such as MoGlow [7] and MotionVAE [8], has opened up a new paradigm for synthesizing characters conditioned on high-level control signals like game-pad inputs or trajectories. Given the same path, unlike deterministic models that tend to regress to the mean pose and field a single predicted motion, generative models can produce diverse motions on each invocation due to their probabilistic nature. However, these methods do not take style into account and fail to meet users' demands for personalized motion generation.

Animation styles refer to the personal style, such as Mickey Mouse or the action's expression level (exaggerated or artistic). Recently, there has been an increasing interest among novice users to generate animated virtual characters of different styles. To add the style attribute to the process of motion synthesis, some ideas of image style transfer techniques have been implemented into the character animation domain: parameterizing and manipulating the style features of an existing motion clip rather than making a new one frame by frame from scratch.

Early methods of motion style transfer use handcrafted features [9], [10] or physics-based representation [11] to manipulate the style of a given animation. Since the style eludes a precise definition, inferring style features with data-driven models is more appropriate. Today, deep learning methods have been introduced to automatically extract style features and combine them with other motion contents [12], [13], [14]. However, most of these methods require paired training data, which means a tedious process of data collection where actors must perform the same motion cycle in different styles with identical steps.

Recently, Aberman et al. [15] have presented a framework that can learn styles from unpaired motion clips. This framework can effectively disentangle even previously unseen style and content features. However, for a given pair of input content and style motion clips, the outputs of the deterministic model are short of motion diversity. On the other hand, Wen et al. [16] make use of a flow-based generative network that can be trained via unsupervised learning over a collection of unlabeled motions. Theoretically, unlabeled motions of different styles can be easily

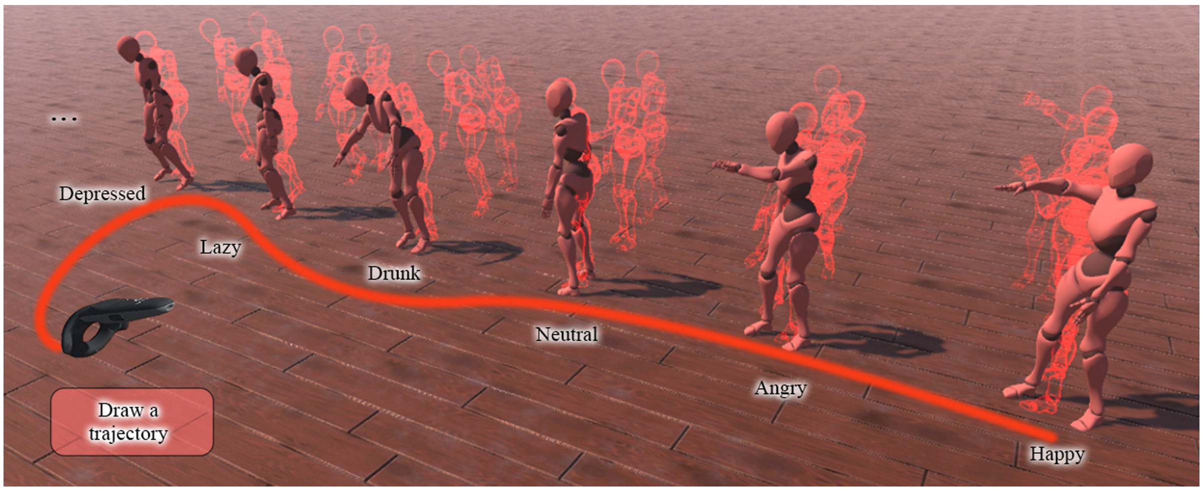


Fig. 1. Given a specific trajectory in VR scene, our proposed model can directly generate the animated characters with different styles. More details are shown in the supplementary video.

captured in reality, which is crucial for generating high quality motions.

However, unlike temporally-invariant image style features [17], the motion style contains not only low-frequency features (category information) but also time-varying high-frequency features within each category. When editing the style features of the animation clip, only the high-frequency component need to be modified, rather than the low-frequency component that remains constant over time. So the above two methods have a significant drawback in that they only disentangle style and content and do not further implement fine-grained disentangled edits of high and low frequency style features. This results in deterministic and short-term results.

In this article, we treat animation stylization as a conditional generative task. First, we decompose style features into high and low frequency parts and encode high-frequency features into the latent space for diverse motion sampling. Then we utilize style-content disentangled editing to ensure rationality during the long and aperiodic motion. Specially, we investigate animation stylization from two aspects: i) motion style transfer, transforming the style features while retaining the content details in a portion of the trajectory; ii) stylized motion generation, transforming the style labels while retaining the content labels in most of the trajectory.

Technically, with the help of normalizing flows, we propose a novel architecture named Flow Student's *t* Mixture Model (FlowSMM) to construct the invertible transformation between motion clips and style latent codes. FlowSMM models the probability density of the latent distribution as a Student's *t* Mixture Model (SMM). Each mixture component of SMM encodes style features of a certain class. The latent codes within a certain component have the same low-frequency features (category of style) and different high-frequency features. When the style-specified motion is generated by drawing a random sample from the corresponding component of SMM, we can keep the low-frequency features consistent and only change the high-frequency features, resulting in various plausible stylized motions. Considering the limited dataset, we choose SMM, which is robust to the small dataset, rather than Gaussian Mixture Model (GMM) as the latent distribution. Furthermore, to make the content-style attributes of the prolonged acyclic animations editable, FlowSMM treats content-style labels as external conditional priors. In terms of the module details, the actnorm layer [18] in the invertible nonlinear transformation is improved as a learnable multi-actnorm layer to overcome the problem of overfitting. The local self-attention mechanism is adopted to reduce the time complexity of the transformer module from $\mathcal{O}(T^2)$ to linear $\mathcal{O}(T)$.

To the best of our knowledge, FlowSMM is the first to realize disentangled edits of style and content while keeping the motion synthesis probabilistic, and solve the two problems of motion style transfer and stylized motion generation. In order to further demonstrate the great potential of our model in VR scenes, we designed a prototype called StyleVR in Unity. As shown in Fig. 1, after the process of trajectory drawing, StyleVR can directly generate stylized animations of different styles. Experiments show that our system outperforms state-of-the-art methods in motion style transfer and stylized motion generation.

In summary, our contributions are threefold:

- We develop a novel prototype called StyleVR in Unity to solve two style-related problems: motion style transfer and stylized motion generation.
- We introduce a probabilistic model that develops the disentanglements of style and content as well as high and low frequency style features.
- The extensive experiments demonstrate that FlowSMM surpasses state-of-the-art methods in motion style transfer and stylized motion generation.

II. Related Work

A. Character Animation

Character animation in VR/AR scenes is a booming research area. Considerable attention has been paid to this direction to create and edit animated characters in virtual environments. Recently, Vogel et al. [19] explore the field of VR animation with a mind-blowing tool: AnimationVR. Then, Pan et al. [20] present the PoseMMR, which allows multiple users to perform immersive animation editing in an AR environment. Besides, much care has been taken into the field of accessibility and portability. Motion Doodles [3] pioneer this field and use 2D curves with 2-DoF gestures to create animated character. Lockwood et al. [21] first control virtual characters on a virtual plane with the sensor data of a mobile phone. Recently, the 3D extension of Motion Doodles: Spatial Motion Doodles [2] is proposed to draw 3D trajectory doodles and guide animation generation with 3-DoF gestures.

In order to generate convincing character animation conditioned on simpler control signals like trajectories in VR/AR scenes or game-pad inputs, there has been a growing trend towards data-driven approaches. The PFNN [4], for example, uses a global phase variable to animate human locomotion. Then MANN [22] replaces the handcrafted phase function with a gating structure for quadruped motion control. Additionally, the NSM [5] uses the gating network from MANN to generate interaction motions. And local motion phase in [6] is utilized to achieve motion interactions with dynamic objects. In order to make the generation process more controllable, Holden et al. [23] improve the Motion Matching algorithm with the scalability of neural-network-based models. Starke et al. [24], propose the Periodic Autoencoder that transforms unstructured character motions into a periodic manifold in an unsupervised manner. Mason et al. [25] realize real-time style modelling based on local motion phases. However, none of these approaches take probabilistic animation synthesis into account.

Recently, generative models have emerged as a promising solution for increasing motion flexibility. Many GAN-related works and adversarial training strategies [26], [27], [28] have been implemented for motion generation. VAE-based approaches have also gained recent interest. Ling et al. [8] propose MVAE to avoid posterior collapse and balance motion quality against generalization. Petrovich et al. [29] utilize CVAE combined with a transformer for multimodal motion synthesis. Meanwhile, Henter et al. [7] propose a different type of generative model based on normalizing flow, which allows for invertible inference and tractable probability computation. Alexanderson et al. [30] extend this work by generating attribute-controllable gesture animation and edit some concrete attributes like hand height, speed, and gesticulation radius with the flow-based model. However, none of the above-mentioned methods consider the motion style. Therefore, we propose a novel solution based on a normalizing flow model to control both style and content.

B. Motion Style Transfer

The concept of transferring features from the source to the target has been applied in many fields, such as image style transfer [31], face swapping [32] and motion Reenactment [33]. In this section, we introduce how to manipulate the styles of motions.

Early studies of motion style transfer rely on intricately crafted features to represent styles [10], [34]. However, since the concept of "style" lacks a precise definition, it is logical to explore statistical learning methods and learn style features from data. For instance, Hsu et al. [35] model the stylistic variances with a linear time-invariant model. Taylor et al. [12] employ a Conditional Restricted Boltzmann Machine to capture diverse motion styles. Xia et al. [13] construct a series of local mixtures of autoregressive models to depict the interrelationships between different styles.

Recent works [15], [36], [37] have incorporated ideas from image style transfer and image recognition. For example, Holden et al. [36] propose a convolutional autoencoder framework that can transform the style in the space of the motion manifold. Du et al. [37] further improve the method by replacing the optimization procedure with a neural network. Recently, inspired by the concept of Adaptive Instance Normalization (AdaIN) [31], Aberman et al. [15] disentangle style features from motions and inject them into the generation process with a temporally invariant AdaIN. However, the model may not perform well when dealing with styles that are significantly different from those in the training dataset. Meanwhile, Wen et al. [16] firstly introduce the generative flow model for motion style transfer, which disentangles style and content more smoothly. While their unsupervised method can be trained on unlabelled motion data, it may not allow for frequency-based fine-grained disentangled edits of style. In this article, we model the latent space as an SMM to solve the problem.

III. Method

In this section, we present our solution for style-based editing using a flow-based model called FlowSMM. Fig. 2 shows an instance of our model. Further information about the latent distribution, flow architecture and training strategy is elaborated below.

Motion Representation: We represent a character animation clip $P \in \mathbb{R}^{T\times d}$ as a sequence of poses with T frames. Each pose x_t at timestamp t is a vector of 63 dimensions comprising the 3D positions of 21 skeleton joints. We use the 3D position representation for several reasons: 1). In the 3D Cartesian coordinate, the representation of pose is continuous and can be interpolated using simple linear operators. 2). We aim to keep the dimension of the motion representation as small as possible to avoid the issue of high data dimension and the small number of data samples. Specifically, we use a root-relative coordinate system, whose origin is the trajectory on the floor. The trajectory is represented by a displacement sequence $S \in \mathbb{R}^{T\times 3}$, where each displacement signal $d_t \in \mathbb{R}^3$ contains the translational velocity $(\Delta x_t, \Delta z_t)$ and y-axis rotational velocity Δy_t.

Style and Content Labels: To make the content-style attributes editable, we use semantic labels represented as one-hot encoding. There are two reasons for this choice. Firstly, one-hot encoding can remove the ambiguity of similar attributes such as walking/jogging and happy/proud. Secondly, one-hot vectors are easy to mark on the motions. In this way, the style and content label for each frame can be represented as $s_t \in \mathbb{R}^{N_s}$ and $c_t \in \mathbb{R}^{N_c}$, where N_s is the number of content classes, N_c is the number of style classes.

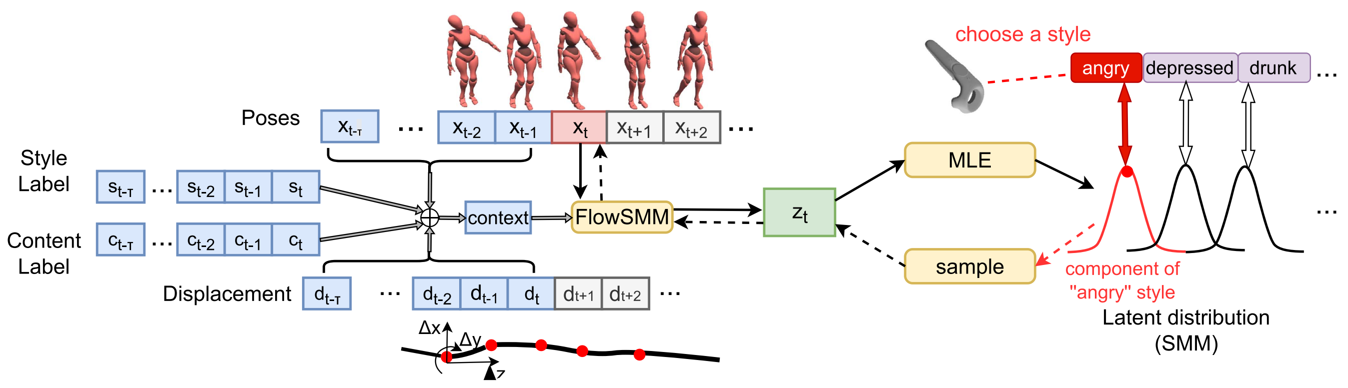


Fig. 2. The framework of stylized motion synthesis. Our proposed FlowSMM is bijective. In the training phase, following the solid arrow, the current pose x_t is inputted to FlowSMM to infer the latent vector z_t, which is supervised to a SMM based latent distribution by means of Maximum likelihood estimation (MLE). In the testing phase, following the dotted arrow, we first select our favorite style by sampling a z_t vector from the corresponding distribution component and then obtain the reconstructed pose x_t via the reversed FlowSMM process. Both the training and testing are conditioned on the motion context (the concatenation of previous poses $[x_{t-\tau:t-1}]$, style label $[s_{t-\tau:t}]$, content label $[c_{t-\tau:t}]$ and the displacement $[d_{t-\tau:t}]$).

A. Generative Flow Models

The normalizing flow [38] is a flexible generative model for density estimation, defined as an invertible and differentiable transformation $f : \mathcal{Z} \to \mathcal{X}$ from a latent distribution $\mathcal{Z}$ modeled as $p_\mathcal{Z}$ to the real and complex data distribution $\mathcal{X}$. In this way, a data sample $x \in \mathbb{R}^{63}$ (with subscript t omitted for brevity) can be generated as $x = f_\theta(z)$, where $z \in \mathbb{R}^{63}$ is a random variable from the latent distribution $z \sim p_\mathcal{Z}$. The function f_θ is a bijection, which enables both efficient sampling and inference. In particular, the inference from x to z is achieved by $z = f_\theta^{-1}(x) = g_\theta(x)$. For brevity, we omit subscript θ in the following paragraphs.

To construct the transformation f, flow-based models [18], [38], [39] combine K steps of nonlinear transformations $\{f_k\}_{k=1}^K$ together:$f = f_1 \circ f_2 \circ ... \circ f_K$ and parameterize them with $\theta = \{\theta_k\}_{k=1}^K$. Therefore, z can be transformed to x by f. The relationship between x and z can be defined as follows:

$$z = z_K \overset{f_K}{\to} z_{K-1} \overset{f_{K-1}}{\to} ... \overset{f_2}{\to} z_1 \overset{f_1}{\to} z_0 = x, \tag{1}$$

$$x = f(z) = f_1\left(f_2(...f_K(z))\right), \tag{2}$$

$$z = f^{-1}(x) = f_K^{-1}\left(f_{K-1}^{-1}(...f_1^{-1}(x))\right). \tag{3}$$

Such a sequence of equation (3) is called a normalizing flow [38]. Using the change of variables theorem [40], the probability density function of x can be written as

$$p_\mathcal{X}(x) = p_\mathcal{Z}(z) \cdot \prod_{k=1}^K \left| det\left(\frac{\partial z_k}{\partial z_{k-1}}\right)\right|. \tag{4}$$

In order to train the parametric model, it is typical to perform maximum likelihood. The log-likelihood of the probability density function of x is given by

$$\log p_\mathcal{X}(x) = \log p_\mathcal{Z}(z) + \sum_{k=1}^K \log \left| det\left(\frac{\partial z_k}{\partial z_{k-1}}\right)\right|, \tag{5}$$

where the value $\log|det(\frac{\partial z_k}{\partial z_{k-1}})|$ is the *log-determinant* of the Jacobian matrix $\frac{\partial z_k}{\partial z_{k-1}}$. The Jacobian-determinant computation takes a time cost of $\mathcal{O}(D^3)$, which is intractable for large input dimension D. In order to reduce the computation complexity, a highly efficient transformation f has been designed to facilitate the calculation of the Jacobian determinant. While much attention has been paid to the model design, the problem of sampling the style latent codes $\{z_t\}_{t=1}^T$ with the same style label but different high-frequency features from the latent space $\mathcal{Z}$ and synthesizing stylized animation x_t is still under-explored. Our solution will be described in the following subsection.

B. Flow Student's t Mixture Model

Here we propose the Flow Student's t Mixture Model (FlowSMM), a conditional probabilistic model for stylizing character motion. In FlowSMM, we denote the style label with a discrete variable $y \in \{1, \ldots, \mathcal{C}\}$. The latent space, conditioned on a label i, is modeled as a fat-tailed multivariate t-distribution with μ_i and Σ_i

$$p_\mathcal{Z}(z|y = i) = t_\nu(z|\mu_i, \Sigma_i), \tag{6}$$

where the scalar $\nu > 0$ is called the degrees of freedom.

The marginal distribution of z is a Student's t mixture model

$$p_\mathcal{Z}(z) = \sum_{i=1}^{\mathcal{C}} \pi_i t_\nu(z|\mu_i, \Sigma_i), \tag{7}$$

we hypothesise that all categories of styles have the same proportion and set each mixture coefficient $\pi_i = \frac{1}{\mathcal{C}}$.

Combing equations (4) and (6), the style-class-conditional likelihoods of x is

$$p_\mathcal{X}(x|y = i) = t_\nu(z|\mu_i, \Sigma_i) \cdot \prod_{k=1}^K \left| det\left(\frac{\partial z_k}{\partial z_{k-1}}\right)\right|. \tag{8}$$

Student's t Mixture Model: To encode different kinds of style features into the latent space, we formulate the probability density of the latent distribution as a mixture model. Previous work of Izmailov et al. [41] employs a GMM as the latent distribution of the flow model. Then Alexanderson et al. [42] explore the

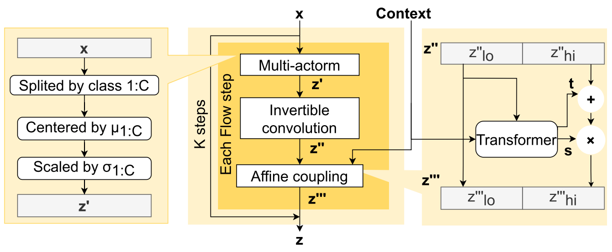


Fig. 3. Illustration of the FlowSMM architecture. FlowSMM consists of K flow steps. Each flow step contains three subsections: *actnorm*, *invertible 1 × 1 convolution and affine coupling layer*. We improve the actnorm module into a multi-actnorm module to mitigate overfitting. Moreover, the local self-attention-based transformer is imposed into the affine coupling layer to optimize the time complexity from $\mathcal{O}(T^2)$ to $\mathcal{O}(T)$.

statistical robustness of the flow model with the fat-tailed t-distribution. In this article, we aim to implement FlowSMM on the small dataset where limited motion data often fails to capture the full diversity of the real motion distribution. If the latent distribution cannot account for outliers in the motion distribution, these outliers will adversely affect the process of maximum likelihood. Therefore, we mitigate the impact of these outlying data points by selecting t-distributions as the components of the mixture model and using SMM as the latent distribution.

Mean and Covariance Choices of t-Distribution: Previously, flow models used the Gaussian-based latent distribution with a simple density $z \sim \mathcal{N}(0, \mathbf{I})$. While in the case of SMM, the means and covariances are relate to the proportions of different kinds of styles. In this article, we draw the mean vectors μ of each component randomly from a standard normal distribution $\mu_i \sim \mathcal{N}(0, \mathbf{I})$ and keep their covariance matrices as identity. The mean setup can be attributed to the fact that the dimension of the pose is $D = 63$. Due to the curse of dimensionality [41], mean vectors of different components are almost orthogonal to each other, and their distances are about $\sqrt{2D}$, which keep mean vectors apart from each other in the latent space regardless of the number of style categories. On the other hand, the covariance matrix of each mixture component measures the distribution magnitude of the style. A larger magnitude indicates a higher likelihood that this type of style feature will be selected during random sampling time. Therefore, we fix the covariance matrices as identity because of our above assumption that all style classes have the same proportion.

Flow Architecture: In this section, we elaborate more details about the architecture of our model. As illustrated in Fig. 3, each flow step of transformation $g_\theta(\cdot)$ consists of 3 parts: *actnorm*, *invertible 1 × 1 convolution* and the *affine coupling layer*.

Actnorm is a type of operation that works similarly to batch normalization, which functions as an affine transformation $z' = \frac{x-\mu}{\sigma}$ with the mean μ and standard deviation σ of an initial batch of data to alleviate the problems of deep model training. However, to better clarify the inter-class boundaries between different types of style features, multi-actnorm layer is proposed for multi-class data-dependent initialization [43]. The data from each style will share the same μ and σ. Moreover, we treat μ and σ as trainable parameters during the following training steps to realize a data-independent learning process. In the section V, experiments demonstrate the critical role of multi-actnorm in handling model overfitting.

Following the multi-actnorm lies an invertible 1×1 convolution layer for the permutation of the channel variables, which performs a linear transformation denoted as $z'' = W \cdot z'$, where $W \in \mathbb{R}^{D \times D}$. Similar to Glow [18], we factorize W in the form of LU-decomposition to compute the Jacobian determinant in linear time. Some elements of L and U are set trainable during the training.

As mentioned above, pose x_t has editable style and content attributes. Meanwhile, it should also follow the trajectory on the floor and maintain continuity with previous poses. As a result, x_t is conditional not only on its style label s_t and content label c_t but also on the previous poses $[x_{t-\tau}, x_{t-\tau+1}, \ldots, x_{t-1}]$ and displacements $[d_{t-\tau}, d_{t-\tau+1}, \ldots, d_t]$ like a Markov chain of order τ, which can be formulated mathematically as

$$p_{\mathcal{X}}(x_t|s_t, c_t, x_{t-1:0}, d_{t:0}) = p_{\mathcal{X}}(x_t|s_t, c_t, x_{t-1:t-\tau}, d_{t:t-\tau}). \tag{9}$$

Therefore, we take inspiration from [7], [30] and incorporate a transformer structure [44] in the affine coupling layers to enable conditional intervention during the generation process. The transformation $x_t = f(z_t)$ should be modified to

$$x_t = f(z_t, s_t, c_t, x_{t-\tau:t-1}, d_{t-\tau:t}). \tag{10}$$

As is shown in Fig. 3, the coupling layers affinely transform half of the input z''_{hi} based on the other half z''_{lo}. To remain tractability when performing reverse calculation, z''_{lo} is left unchanged. Specifically, we first split the input into two halves as $[z''_{lo}, z''_{hi}]$. Then, we perform the following transformations as

$$z'''_{hi} = (z''_{hi} + t) \odot s, \tag{11}$$

$$t, s = \mathcal{F}(z''_{lo}, s_t, c_t, x_{t-\tau:t-1}, d_{t-\tau:t}), \tag{12}$$

where the transformer $\mathcal{F}$ feeds the combined context of s_t, c_t, $[x_{t-\tau:t-1}]$ and $[d_{t-\tau:t}]$ as the additional inputs to extract transformation parameters t and s.

Wen et al. [16] first introduces the transformer structure into flow model, which outperforms the RNN-based MoGlow [7] by utilizing attention mechanism to construct long-range temporal receptive field and enable parallelizable training of multi-frame poses. However, its quadratic time complexity $\mathcal{O}(T^2)$ with respect to pose length T limits its application due to excessive memory consumption [45]. Inspired by the work of Huang et al. [46], we modify the global self-attention mechanism into a local self-attention mechanism by restricting the position connections within τ previous elements per layer to reduce the time complexity to $\mathcal{O}(T)$, where τ is much smaller than T. This improvement align with the above Markov assumption and the intuition that the nearest neighbors' context has a significant impact on the current pose. In this way, we optimize the training process in an efficient and time-saving way.

C. Solutions to Sub-Problems

In this section, we present our solutions for two problems: first, *motion style transfer*, which involves transferring user-specified style features to the given character animation while

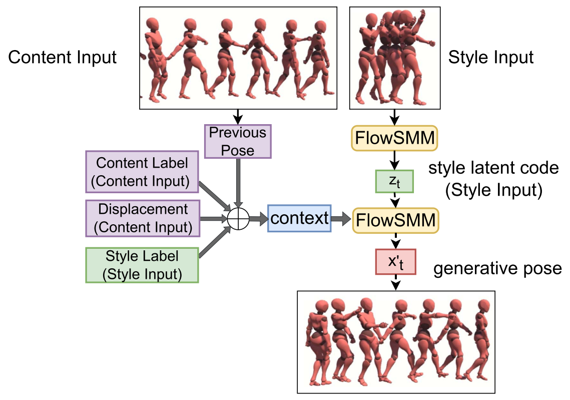

Content Input
Style Input
Previous Pose
FlowSMM
z_t
style latent code (Style Input)
Content Label (Content Input)
Displacement (Content Input)
Style Label (Style Input)
context
FlowSMM
x'_t
generative pose

Fig. 4. Illustration of the style transfer. Maintaining the motion content of the content input, we can extract style latent codes from style input to transfer style during the generation process.

preserving the content features of this animation; second, *stylized motion generation*, which aims to produce probabilistic stylized motions with more relaxed constraints. Hence we transfer only the user-specified style labels to the given character animation while preserving the content labels of this animation.

Motion Style Transfer: The objective of this task is to edit style features of content input to match those of style input while preserving the content features of content input. Given the content input $\{x_t^1\}_{t=1}^{T_1}$, $\{d_t^1\}_{t=1}^{T_1}$, s_t^1, c_t^1 and style input $\{x_t^2\}_{t=1}^{T_2}$, $\{d_t^2\}_{t=1}^{T_2}$, s_t^2, c_t^2, we first project $\{x_t^2\}_{t=1}^{T_2}$ into the latent space to obtain $\{z_t^2\}_{t=1}^{T_2}$, and then invoke the mapping function f with the form $f(z_t^2, x_{t-\tau:t-1}^1, d_{t-\tau:t}^1, s_t^2, c_t^1)$, which can be written as

$$z_t^2 = f^{-1}(x_t^2, x_{t-\tau:t-1}^2, d_{t-\tau:t}^2, s_t^2, c_t^2), \tag{13}$$

$$x_t = f(z_t^2, x_{t-\tau:t-1}^1, d_{t-\tau:t}^1, s_t^2, c_t^1). \tag{14}$$

The process is illustrated in Fig. 4, and the content input is modified by the style input and evolves to a stylized motion clip.

Stylized Motion Generation: There are some limitations when editing motion clips with style transfer: i) the content input is modified by the specific style code, resulting in a deterministic output; ii) ignoring the impact of the newly added style on the content and displacement results in short-term motion generation.

To overcome these limitations, we propose a scheme of stylized motion generation, as depicted in Fig. 2. The first step involves sampling the new style feature z_t from the mixture component $t_\nu(\mu_2, \mathbf{I})$ that corresponds to the style class of s_t^2. Next, we replace s_t^1 with s_t^2 and the original $d_{t-\tau:t}^1$ is substituted with a new displacement sequence $d_{t-\tau:t}^{12}$ that matches well with the new style-content pairs. Details of $d_{t-\tau:t}^{12}$ will be explained in Section IV-A. Finally, we obtain the pose x_t of each moment with a learned nonlinear transformation f, which can be formulated as:

$$x_t = f\left(z_t, x_{t-\tau:t-1}^1, d_{t-\tau:t}^{12}, s_t^2, c_t^1\right) \quad \text{and} \quad z_t \sim t_\nu(\mu_2, \mathbf{I}), \tag{15}$$

D. Training Objectives

Based on the (8), we have access to the likelihood for labeled data. Then we can train the FlowSMM by minimizing the negative log-likelihood loss

$$\mathcal{L}_{nll} = -\sum_{t=0}^{T-1} \log p(x_t|y_t), \tag{16}$$

While $\mathcal{L}_{nll}$ guarantees motion diversity, it does not provide sufficient guidance for motion continuity. To address this, we propose a new loss function $\mathcal{L}_{\text{con}}$ to supervise motion continuity. After obtaining the latent code z of x through the forward transformation, we sample a new z' with the same style label as z and generate the corresponding x' through the reverse transformation. The motion continuity loss is as follows:

$$\mathcal{L}_{\text{con}} = \sum_{t=1}^{T-1} ||x_t - x_{t-1} - x'_t + x'_{t-1}||_1 \tag{17}$$

The intuition behind $\mathcal{L}_{\text{con}}$ is that joint positions, rather than velocities, reflect motion diversity. And that enforcing velocity constraint can help maintain the same low-frequency features between x' and x, such as the position of the standing foot joint while keeping high-frequency features as different as possible, such as the positions of high-speed end-effectors. Finally, our loss function is as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{nll} + \lambda_2 \mathcal{L}_{\text{con}} \tag{18}$$

E. Implementation Details

For FlowSMM, the number of flow steps is $K = 16$, the order of the Markov chain is $\tau = 10$ and degrees of freedom is $\nu = 50$. The Transformer in each coupling layer consists of 2 identical layers. Each layer has 2 blocks: a local self-attention block with 8 heads followed by a position-wise fully connected block. During training, the weight of $\mathcal{L}_{nll}$ and $\mathcal{L}_{\text{con}}$ are $\lambda_1 = 1$ and $\lambda_2 = 24$. We use the Adam solver and train with a batch size of 128. The learning rate is initialized to $lr = 10^{-4}$ and decreases linearly to 5×10^{-5} at the 6000th step. Our implementation is in PyTorch.

IV. StyleVR System

Based on FlowSMM, we developed a prototype system called StyleVR to help users stylize their VR animations automatically. Additionally, StyleVR considers postprocessing to address issues such as motion jitters and foot skating. The processing pipeline for transforming a trajectory doodle into a stylized animation consists of three steps: editing, animating and postprocessing.

A. Editing

Trajectory Drawing: Existing VR/AR animation creation systems utilize an operable medium within the VR space to produce animations [1], [2]. Our system operates in a similar way, allowing users to draw their desired path in the Unity with the VR controllers. As illustrated in Fig. 5(a), we use the controller as an emitter for ray casting, and the trajectory is

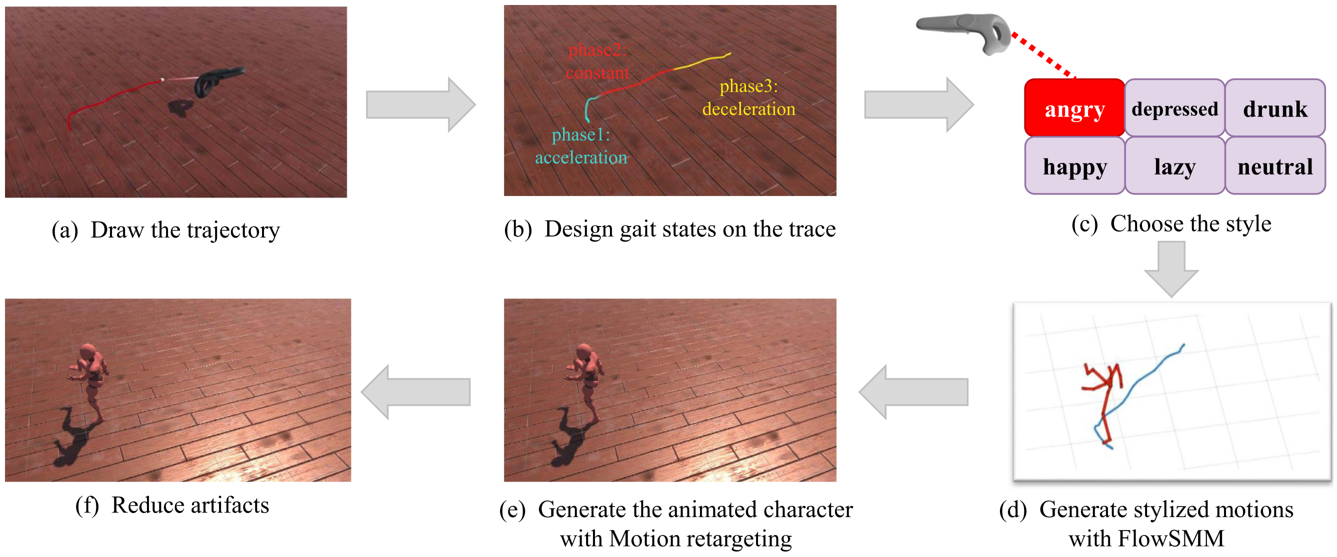


Fig. 5. Overview of the proposed StyleVR system. Novice users can generate stylized animations using StyleVR by following the steps of: (a) drawing a trajectory with VR controller; (b) designing gait states for the trace; (c) choosing their favorite styles; (d) generating stylized joint locations with FlowSMM; (e) transferring poses to the avatar with motion retargeting; (f) coping with artifacts like motion jitters and foot sliding.

formed by the points of intersection on the XZ-plane, which are sampled at 60 HZ.

Velocity Design: Once the track has been drawn, a displacement sequence $\{d_t\}_{t=1}^{T}$ must be designed along the path. As mentioned in Section III-C, $\{d_t\}_{t=1}^{T}$ need to be consistent with the style-content pair. Since many contents have similar speeds, $\{d_t\}_{t=1}^{T}$ is designed based on style-gait pairs. Initially, certain gait states are defined, including 3 "constant" states with different periodic speeds: "idleness", "walking" and "running", along with 3 "acceleration" and 3 "deceleration" states for transitions between different "constant" states. In this way, users only need to design the "constant" states, and the system will complete the "acceleration"/"deceleration" states automatically. In this article, we use the small dataset presented by Aberman et al. [15], which only contains the motion states of "idleness", "walking" and the corresponding "acceleration" and "deceleration". We make a dictionary with motion styles and states as keys and the typical speed samples from the dataset as values. As shown in Fig. 5(b) and (c), we design states and choose the style and content labels for the trace. Then we query the dictionary for the d_t according to the relation $d_t = \phi(style, state, t)$.

B. Animating

Generating: As shown in Fig. 5(d), FlowSMM allows users to synthesize stylized motions with inputs of velocities $\{d_t\}_{t=1}^{T}$, style features $\{z_t\}_{t=1}^{T}$, style label s_t and content label c_t. For demonstration, there are 6 styles included in the current implementation: "angry", "depressed", "drunk", "happy", "lazy" and "neutral".

Motion retargeting: Motion retargeting is necessary for transferring the stylized motion to an animated avatar. As the motion representation of FlowSMM is based on 3D positions, we convert positions into rotation matrices and apply these rotations to the virtual avatar. A joint rotation $R \in \mathbb{SO}(3)$ can be decomposed into a swing rotation and a twist rotation. While the twist rotation rotates around the bone vector and cannot be calculated analytically, it is set to a default value. For the swing rotation, once the start vector $\vec{s}$ and target vector $\vec{t}$ of a bone are determined, the swing angle θ between $\vec{s}$ and $\vec{t}$ will satisfy the conditions: $\sin\theta = \frac{\|\vec{s}\times\vec{t}\|}{\|\vec{s}\|\|\vec{t}\|}$ and $cos\theta = \frac{\vec{s}\cdot\vec{t}}{\|\vec{s}\|\|\vec{t}\|}$. And the closed-form solution of the swing rotation matrix can be obtained using the Rodrigues formula

$$R^{\text{swing}} = \mathcal{I} + \sin\theta[\vec{n}]_\times + (1-\cos\theta)[\vec{n}]_\times^2. \tag{19}$$

Where $\mathcal{I}$ is the 3×3 identity matrix, $\vec{n}$ is the unit normal vector of $\vec{s}$ and $\vec{t}$ and $[\vec{n}]_\times$ is the skew symmetric matrix of $\vec{n}$.

Finally, with the avatar rest pose template T and the relative rotations R, we are able to compute the reconstructed pose P in real time through the process of Forward kinematics (FK) in Unity.

C. Postprocessing

Since our model is trained on the limited dataset and tested with various trajectories, it is inevitable to generate some noticeable visual artifacts such as motion jitters and foot sliding. Subsequently, we will discuss the issues of foot sliding and motion jitters.

Motion jitters: In regards to motion jitters, we find that most outlying observations produced by FlowSMM are related to joints of left/right hands. To achieve temporally smooth hand motions, we first represent left/right hand joints relative to their parent nodes (left/right elbow) and then apply Gaussian filters to smooth out the values. The window size of Gaussian filters is set to 3.

Foot sliding: In order to address the issue, Aberman et al. [15] adjusted the foot contact positions based on the extra content motion, while Xia et al. [13] utilized the detected foot plant constraints to the artifacts.

Algorithm 1: Alleviate Foot Sliding.

```
Input: end-effector $E[0, \dots, T-1]$, pose $P[0, \dots, T-1]$
Output: $\hat{P}$
1  $\hat{P} \leftarrow P$; $i \leftarrow 0$;
2  while $i < T$ do
3      while $E[i]$ does not contact the floor do
4          $i \leftarrow i + 1$;
5      $t \leftarrow i$;
6      $avg \leftarrow E[t]$;
7      while $t + 1 < T$ and $E[t+1]$ contact the floor do
8          $t \leftarrow t + 1$;
9          $avg \leftarrow avg + E[t]$;
10     $avg \leftarrow avg/(t - i + 1)$;
11     for $j \leftarrow i$ to $t + 1$ do
12         $E[j] \leftarrow avg$;
13         $\hat{P}[j] \leftarrow \mathrm{IK}(E[j], P[j])$;
14     $i \leftarrow t + 1$;
15 return $\hat{P}$
```

In Section IV-A, we have avoided severe foot sliding through velocity design. To further eliminate the slight foot sliding artifacts, we identify the left/right feet as end-effectors and determine if they are stationary or in motion. Then we average the positions of "standing" end-effectors and use the modified end-effectors in the inverse kinematics (IK) process to fine-tune the positions of other joints. The pseudocode of Algorithm 1 is provided below, which is applied to the left/right feet respectively.

V. Experiment and Evaluation

A. Datasets

For training and quantitative evaluation, we utilize the dataset provided by Aberman et al. [15], which contains motion sequences of 16 styles. This dataset is considered 'tiny' due to the short duration of the motion sequence, which is only 5–6 minutes per style. By contrast, the mainstreaming dataset, such as CMU MoCap for action tasks, contains 204079 frames for 8 actions (35 mins per action). Additionally, some previous works on motion style transfer [47], [48] still take small datasets into account. Yumer et al. [47] use the smallest dataset, but their work only focuses on style transfer. The dataset of Mason et al. [48] is larger than ours since they first need a large dataset to train the motion generation module and then use an extra small dataset to train the style-related module.

We first downsample the dataset from 120 fps to 30 fps. This is done because the input of FlowSMM includes the previous poses of τ frames, and we want to expand the receptive field of the limited τ frame poses while avoiding the problem of high data dimension and the small number of data samples. Then for further batch processing, we split the dataset into fixed-length clips of $T = 64$ frames with an overlap of T/8. In this way, we have access to 12549 motion sequences. Each style's sequences are divided into two disjoint subsets, with the smaller one (consisting of 10% of the samples) will act as the test set. Moreover, in the training phase, the data is augmented by lateral mirroring and motion reversing [7]. To calculate the displacements, we map the hip motion to the floor and smooth the trajectory with the Gaussian-filter. To reduce the difficulty of regression, $d_t = (\Delta x_t, \Delta z_t, \Delta y_t)$ is expressed in the form of frame-wise delta-translation and delta-rotation relative to the previous frame.

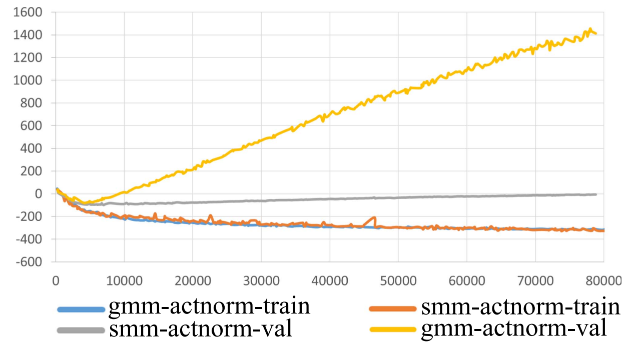


Fig. 6. Quantitative evaluation of latent distributions. The training losses of GMM and SMM are of similar trends. Nevertheless, the gap between validation losses keeps increasing over time, indicating the superiority of Student's *t* Mixture Model to mitigate overfitting.

B. Ablation Study

To put the performance of FlowSMM in perspective, we conduct thorough evaluations to assess the effectiveness of both SMM and multi-actnorm layers. Specifically, we held all other components constant and vary the latent distribution ranging in *GMM/SMM*, as well as the actnorm layer ranging in *actnorm/multi-actnorm*.

Student's t Mixture Model: To demonstrate the effectiveness of SMM, we compare the $\mathcal{L}_{nll}$ curves of *Flow-GMM-actnorm* and *Flow-SMM-actnorm*. In the training phase, both models show a similar decreasing trend in their $\mathcal{L}_{nll}$ curves, as illustrated in Fig. 6. However, in the testing phase, their performance gap becomes more pronounced over time. The $\mathcal{L}_{nll}$ of *Flow-GMM-actnorm* starts to rise steeply early on and diverges linearly, whereas *Flow-SMM-actnorm* performs much better but still exhibit slight overfitting. In addition, we show several representative results of the two models in Fig. 7. The results indicate that outputs of *Flow-GMM-actnorm* are not competitive, while the motion sequences produced by *Flow-SMM-actnorm* are more realistic and lifelike.

The main cause of the behavior can be attributed to insufficient training data. This limits the GMM to learn the complete motion diversity and leaves it vulnerable to outliers, ultimately resulting in low-performance generations. In contrast, the SMM is resistant to outlying data points because it models the latent probability distribution with a fatter-tail structure, leading to greater training stability and further improving generalization results of FlowSMM.

Multi-actnorm layer: Although the *Flow-SMM-actnorm* produces impressive visual results, it tends to overfit in terms of $\mathcal{L}_{nll}$ on testing data. One intuitive explanation is that outliers in

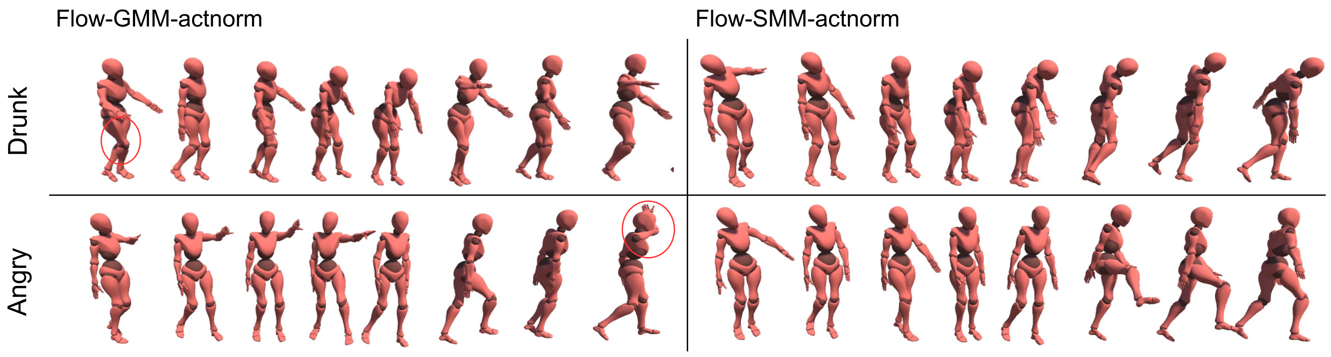


Fig. 7. Qualitative comparison of Flow-GMM-actnorm to Flow-SMM-actnorm. Snapshots of two representative stylized results show that FlowGMM is prone to field artifacts (marked with red circles), while SMM has a decisive effect on the FlowSMM to learn the motion diversity of natural behavior and the generations of FlowGMM are more lifelike.

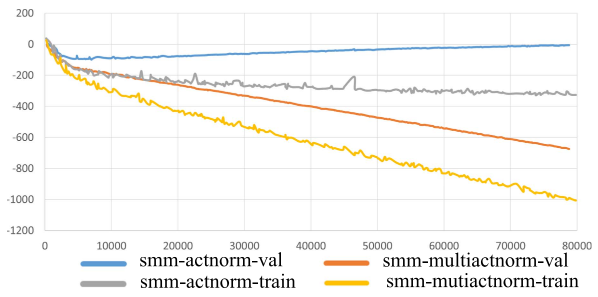


Fig. 8. Quantitative evaluation of actnorm layers. From the loss results of different actnorm layers, it can be seen that both training and validation losses decrease more monotonically by imposing our proposed multi-actnorm layer than the other set of common actnorm layers.

the dataset do not amount to errors but uncommon data points. When we want to reduce the impact of outliers on the parameter estimates, the model tends to "give up" some particular samples and instead focuses on fitting the majority, albeit at the expense of the diversity in synthesized motions. As a result, the learned model regards some testing examples as unnatural outputs, resulting in overfitting.

To enhance the style diversity of the model without compromising its statistically-robust fitting strategy, we instead take the layer of activation normalization into account. Our solution is to propose a multi-actnorm layer that allows each style of data to share the same scale and bias parameters. These parameters capture the typical style feature of the batch rather than that of individual data points, which are more stable and easier to learn. As shown in Fig. 8, testing with the *actnorm layer* fails to converge, while testing with the *multi-actnorm layer* is stable, thereby addressing the problem of overfitting. Moreover, it further improves the training process of FlowSMM.

C. Comparative Performance

In this section, we evaluate FlowSMM in two aspects: (i). the ability of style transfer; (ii). the quality of generated actions.

In order to give an intuitive visualization of how the model encodes style features, we compare FlowSMM with two state-of-art methods developed by Aberman et al. [15] and Wen et al. [16]. Specifically, we extract the style codes of 6 styles from motion clips and project these codes onto a 2D embedded space using t-distributed stochastic neighbor embedding (t-SNE) [49], with each style's samples marked with the same color. Fig. 9 demonstrates the comparison results of all three models. It can be seen that Wen's approach produces blurry boundaries between different styles, resulting in small, distinct clusters for the "happy" and "lazy" styles, while samples with "neutral" labels visibly mixed with other styles, which is consistent with common sense that unsupervised methods like wen et al. [16] are not suitable for style learning and need labels for better learning. On the other hand, Aberman et al.'s approach, which uses supervised learning with style labels, can cluster samples based on style, but still have some "asocial" anomalies that could interfere with the effectiveness of style transfer during the AdaIN transformation step. However, our proposed FlowSMM performs best in style clustering, with the highest intra-class similarity and lowest inter-class similarity. The reason for this is that FlowSMM not only makes use of style labels but also designs a latent distribution of SMM for style codes to cluster into. The settings of means and covariances discussed in Section III-B are also supported by results in Fig. 9(c).

TABLE I
COMPARISON RESULTS WITH OTHER METHODS

Methods	SCoeff	SC	Continuity Loss	Stylized Gen.?
Aberman et al. [15]	0.43	6.76±4.37	-	×
Wen et al. [16]	0.29	2.10±1.74	25.82	×
Ours	**0.88**	**1.82±1.40**	**14.17**	✓

To quantitatively analyze the impact of style transfer, we use silhouette coefficients (SCoeff) [50] to measure the separation distance of the style encoding clusters. The SCoeff scale ranges from -1 to $+1$, with higher value indicating that the sample is well matched to its own cluster and apart from other clusters. As shown in Table I, our analysis shows that the combination of style labels (Aberman et al. [15], FlowSMM) and a well-designed latent space (FlowSMM) account for high values of SCoeff. Additionally, Wen et al. [16] proposed a metric of style consistency (SC) to evaluate the effectiveness of style transfer, which is based on the observation that if content input and style

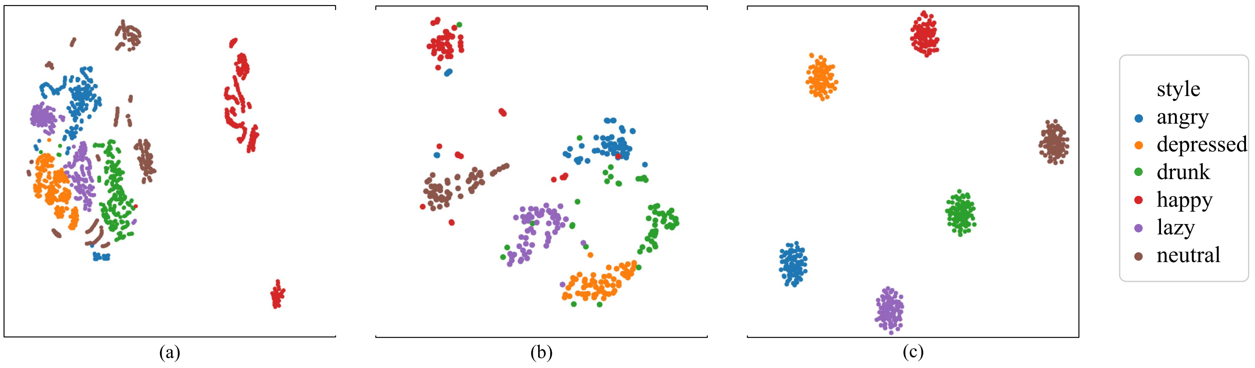


Fig. 9. The visualization of style latent space. Style codes extracted from the motion clips with the methods of Wen et al. [16] (a), Aberman et al. [15] (b) and FlowSMM (c) are evaluated on the 2D embedding space using t-SNE. It can be seen that style representations extracted by the unsupervised learning method (a) can not be clustered properly. While the use of style label (b) count for the learning of style codes, and well-designed latent distribution (c) can further improve the clustering results of latent codes.

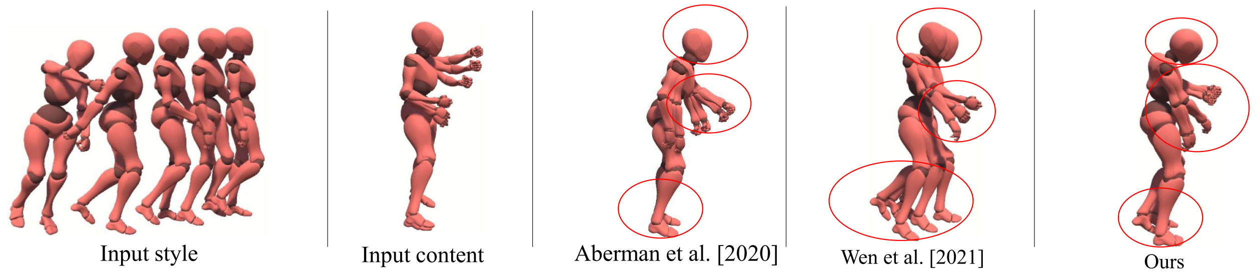


Fig. 10. The representative case. The style input is about "drunk walking", and the content input is about "angry punching". From the details of head pose, punching force and gait, it can be seen that our result is more like a "punching drunkard". The full video can be found in the supplemental video.

input share the same content label, the results after style transfer should be close to the input style. Therefore, SC is defined by computing the Euclidean distance between the stylized motion and the content input. To make the SC unaffected by the probabilistic nature of the generative model, we further keep the length of the test sequence as 160 frames. The results in Table I confirm that manipulating the motion features with temporally invariant AdaIN parameters [15] may not work well. In contrast, invertible transformations of flow-based models (Wen et al. [16] and FlowSMM) can ensure the effectiveness of per-frame style editing. Furthermore, FlowSMM can disentangle the high and low frequency style features in the latent space, resulting in the best performance.

We further present an illustrative example, in which we created a character engaging in "drunk punching" by utilizing the style input "drunk walking" and the content input "angry punching". The "drunk puncher" ought to exhibit behaviours such as head shaking, weakly waving his fist and a slightly unsteady footing. As is shown in Fig. 10, the result of FlowSMM is more realistic.

To evaluate the quality of generated animations, we calculate the motion continuity loss $\mathcal{L}_{\text{con}}$ over a long trajectory. Results presented in Table I demonstrate that FlowSMM outperforms Wen et al. [16] in terms of continuity loss. The superiority of FlowSMM can be attributed to its mixture-model-based latent distribution, where we can sample style-specific latent code for stylized motion generation. Although mixture model is difficult to encode motion continuity, the loss function utilized in the inverse transformation can effectively mitigate the problem.

Finally, we give some examples to visualize the style-content disentanglement. As illustrated in Fig. 11, the avatar performs the same content of putting something down while exhibiting different styles, all while following the same trajectory.

VI. User Study

20 students and staff (12 male) at Shanghai Jiao Tong University were recruited to take part in our experiment, three of them were professional animators. The average age was 23.5 (SD = 1.8).

A. *Accuracy of Stylization*

In this experiment, participants were instructed to watch animations generated by FlowSMM and classify them into style categories. Prior to the measured trial, each participant undertook a practice trial to learn about various styles by watching some ground-truth clips and could ask questions. Then they undertook the test of 36 generated clips, with 6 clips per style. Additionally, 12 ground-truth examples were included to verify the participants' attentiveness. Participants who performed poorly on these examples were asked to redo the test.

Classification accuracy is summarized in Fig. 12. Results show that FlowSMM is capable of producing well-stylized animations for most scenarios. Specially, the labels of "drunk",

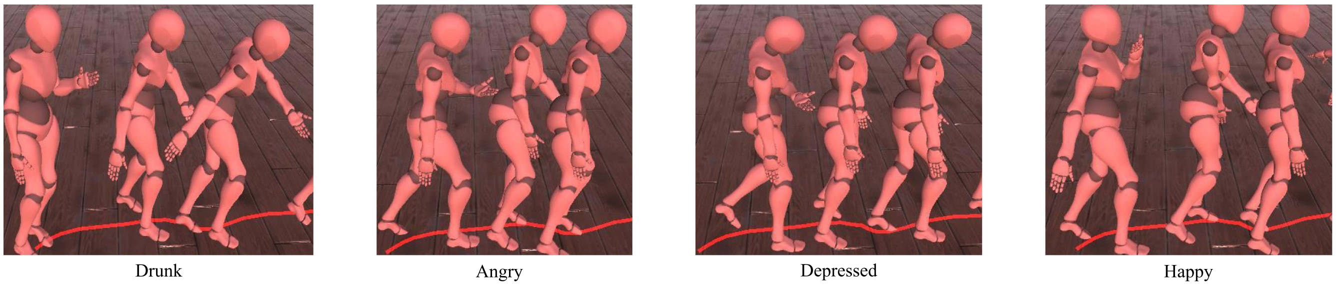
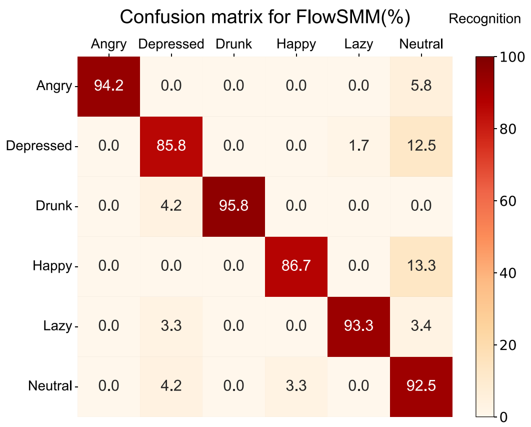


Fig. 11. The visualization of style-content disentanglement. Following the same red trajectory, the avatar can perform the motion content of putting something down at the similar place with different styles. More details can be found in the supplemental video.



	Angry	Depressed	Drunk	Happy	Lazy	Neutral
Angry	94.2	0.0	0.0	0.0	0.0	5.8
Depressed	0.0	85.8	0.0	0.0	1.7	12.5
Drunk	0.0	4.2	95.8	0.0	0.0	0.0
Happy	0.0	0.0	0.0	86.7	0.0	13.3
Lazy	0.0	3.3	0.0	0.0	93.3	3.4
Neutral	0.0	4.2	0.0	3.3	0.0	92.5

Fig. 12. The confusion matrix of classification accuracy. Participants were asked to recognize the style categories of the generated motions.

TABLE II
LIKERT SCALE MARKERS TO ASSESS THE RATIONALITY

Strongly Unsatisfied	Posing at an impossible angle
Slightly Unsatisfied	Interpenetration/foot-sliding/jitter
Mediocre	Reasonable but unappealing results
Satisfied	Natural and stylized results
Extremely Satisfied	Realistic and stylized results

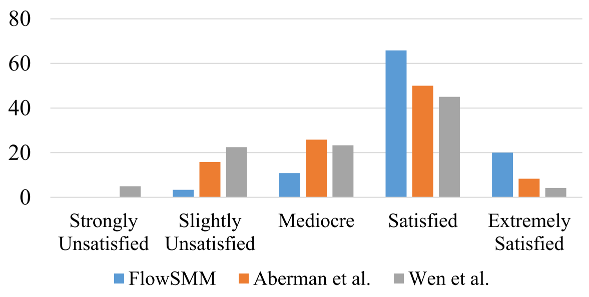


Fig. 13. Responses on the motion rationality of style transfer. The results showed the comparison among three methods based on the Likert scale.

"angry" and "lazy" were always recognized as intended (95.8% vs. 94.2% vs. 93.3%) due to their apparent characteristics (e.g., "lazy": walking idly, "angry": occasionally kicking or stomping, "drunkard": walking unsteadily). However, recognition failures were more frequent for styles of "depressed" and "happy" since their micro-expressions (e.g., "depressed": drooping head, "happy": exaggerated body movements) were challenging to learn with the limited data. Therefore, the "depressed" and "happy" styles were often mistaken for "neutral" style (12.5% vs. 13.3%). To address the issue, we will capture more stylized motion data in future work.

B. Rationality of Generated Animations

We conducted two trials to assess the rationality of style transfer and stylized motion generation using a within-subject design. Each trial required participants to rate three groups of six animations. For style transfer, animations were generated using the same six pairs of content/style input motions but with three different methods. For stylized motion generation, animations were generated using the same six trajectories but with three different methods. Participants who rated the MoCap clips low (scores$\leq$ 2) were asked to repeat the test. Content/style input motions and trajectories were generated through random sampling, and the presentation order of animations was randomized. Evaluations were conducted using a five-point Likert scale, as shown in Table II.

As shown in Fig. 13, the works of Aberman et al. and Wen et al. are not competitive. 15.8% and 22.5% of their responses were "slightly unsatisfied" with generated animations, and even 5% of participants found Wen et al.'s results to be "strongly unacceptable". However, FlowSMM received only 3.4% "slightly unsatisfied" responses, and 65.8% of participants were "satisfied" with its stylized results. This rating was higher than the other two methods, which only received 50% and 45% satisfaction ratings, respectively. The reason for this difference is that Aberman et al.'s model treats style features as temporally invariant, making it less effective for non-periodic motions, while Wen et al.'s approach modifies the per-frame style feature, resulting in poor performance on long-term motions. By contrast, FlowSMM not only considers local style of each frame but also ensure the rationality of long-term motions with style-content labels.

The evaluation of stylized motion generations is visualized in Fig. 14. Our result was deemed acceptable (scores $\geq$ 3) by 90.8% of the respondents, which was lower than those of MoGlow and MoCap (95.8% vs. 100%). However, FlowSMM received higher scores for satisfaction ("Satisfied": 44.2%, "Extremely Satisfied": 28.3%) compared to MoGlow ("Satisfied": 18.3%, "Extremely Satisfied": 11.7%). The reason is that FlowSMM

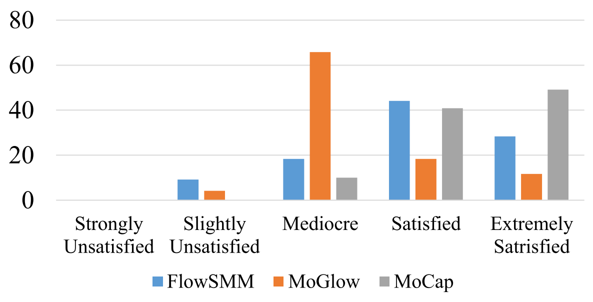


Fig. 14. Responses on the rationality of generated animations. The evaluation compared the rationality of stylized motions with those of MoGlow and MoCap.

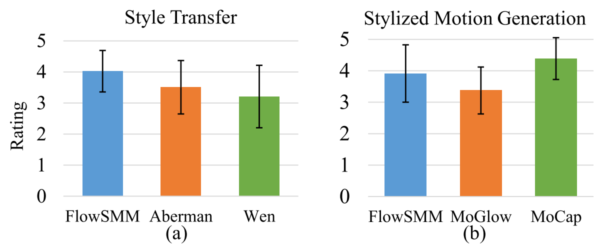


Fig. 15. Mean ratings of the user studies. Mean ratings with standard deviation bars are calculated to evaluate the performance of models.

learns both style and content features to create lifelike and stylized synthetic motions. Even if the latent space has been modeled as SMM to improve training stability, occasional artifacts like foot-sliding and jitter may occur due to limited motion data. By contrast, MoGlow models its latent space as a Gaussian distribution which can make its motions more plausible but less stylized. Therefore, the animations of MoGlow can be more plausible but barely stylized and 65.8% of the responses thought animations of MoGlow "mediocre". Furthermore, only 28.3% of the responses were "extremely satisfied" with animations of FlowSMM, compared to 49.2% of MoCap clips. This result indicates that even if our generated animations are primarily acceptable, they still lack micro-expressions compared to MoCap due to limited datasets.

Fig. 15 illustrates mean ratings of user studies. FlowSMM received the highest rating for style transfer. However, there was a gap between the results of FlowSMM and MoCap for stylized motion generation. One-way repeated measures ANOVA and post-hoc Tukey HSD test were performed to access statistical significance of these differences. Prior to these operations, Kruskal-Wallis test and Bartlett's test were conducted to verify normality and homoscedasticity assumptions of ANOVA, and the boxplot detected no data outliers. The results of ANOVA show significant main effects of different models for both style transfer ($F(2,238) = 129.24$, $p < 0.001$) and stylized motion generation ($F(2,238) = 191.66$, $p = <0.001$). Then Tukey's HSD test revealed that, for style transfer, all differences between models were significant ($p < 0.001$) except for the comparison between the models of Aberman et al. [15] and Wen et al. [16] ($p = 0.019 < 0.05$). For stylized motion generation, all differences between models are significant ($p < 0.001$).

VII. Conclusion

In this article, we propose FlowSMM, a flow-based model that enables the probabilistic and controllable synthesis of stylized animations. Given a specific trajectory in the VR scene, novice users can directly sample the style codes from the SMM-based latent space and generate the animated characters of the user-specified style using invertible flow transformations. Specially, FlowSMM achieves disentanglements of style and content as well as high and low frequency style features, making it applicable to style-related sub-problems: motion style transfer and stylized motion generation. Moreover, FlowSMM imposes the local self-attention mechanism into the transformer module to reduce time complexity and applies learnable multi-actnorm layers to relieve overfitting. We demonstrate FlowSMM's robustness to small datasets through experiments and showcase its application potential with a prototype called StyleVR in Unity, including details about the trajectory design, animation generation and post-processing.

Experiments show that although the "Velocity Design" step of FlowSMM could alleviate foot sliding artifacts, it may not be effective in complex situations. In the future, we plan to draw inspiration from the work of "Motion Matching" [23], and search the motion database for the velocity which is best suited for the given context. On the other hand, FlowSMM mainly deals with the small dataset to evaluate its effectiveness in comparison to the work of [15], [16]. We will further expand our research to larger datasets such as 100STYLE [25] and LaFAN1 [51].

References

[1] H. Ye, K. C. Kwan, W. Su, and H. Fu, "Aranimator: In-situ character animation in mobile AR with user-defined motion gestures," *ACM Trans. Graph.*, vol. 39, no. 4, pp. 83–1, 2020.
[2] M. Garcia, R. Ronfard, and M.-P. Cani, "Spatial motion doodles: Sketching animation in VR using hand gestures and laban motion analysis," in *Proc. 12th ACM SIGGRAPH Conf. Motion, Interact. Games*, 2019, pp. 1–10.
[3] M. Thorne, D. Burke, and M. van de Panne, "Motion doodles: An interface for sketching character motion," in *ACM Trans. Graph.*, vol. 23, no. 3, pp. 424–431, 2004.
[4] D. Holden, T. Komura, and J. Saito, "Phase-functioned neural networks for character control," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–13, 2017.
[5] S. Starke, H. Zhang, T. Komura, and J. Saito, "Neural state machine for character-scene interactions," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 1–14, 2019.
[6] S. Starke, Y. Zhao, T. Komura, and K. Zaman, "Local motion phases for learning multi-contact character movements," *ACM Trans. Graph.*, vol. 39, no. 4, pp. 54:1–54:13, 2020.
[7] G. E. Henter, S. Alexanderson, and J. Beskow, "Moglow: Probabilistic and controllable motion synthesis using normalising flows," *ACM Trans. Graph.*, vol. 39, no. 6, pp. 1–14, 2020.
[8] H. Y. Ling, F. Zinno, G. Cheng, and M. Van De Panne, "Character controllers using motion VAEs," *ACM Trans. Graph.*, vol. 39, no. 4, pp. 40:1–40:12, 2020.
[9] M. U. K. A. R. Takeuchi, "Fourier principles for emotion-based human figure animation," *Proc. 22nd Annu. Conf. Comput. Graph. Interactive Techn.*, 1995, pp. 91–96.
[10] K. Amaya, A. Bruderlin, and T. Calvert, "Emotion from motion," in *Proc. Conf. Graph. Interface*, Toronto, Canada, 1996, pp. 222–229.
[11] C. K. Liu, A. Hertzmann, and Z. Popović, "Learning physics-based motion style with nonlinear inverse optimization," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1071–1081, 2005.
[12] G. W. Taylor and G. E. Hinton, "Factored conditional restricted boltzmann machines for modeling motion style," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 1025–1032.

[13] S. Xia, C. Wang, J. Chai, and J. Hodgins, “Realtime style transfer for unlabeled heterogeneous human motion,” *ACM Trans. Graph.*, vol. 34, no. 4, pp. 1–10, 2015.
[14] H. J. Smith, C. Cao, M. Neff, and Y. Wang, “Efficient neural networks for real-time motion style transfer,” *Proc. ACM Comput. Graph. Interactive Techn.*, vol. 2, no. 2, pp. 1–17, 2019.
[15] K. Aberman, Y. Weng, D. Lischinski, D. Cohen-Or, and B. Chen, “Unpaired motion style transfer from video to animation,” *ACM Trans. Graph.*, vol. 39, no. 4, pp. 64:1–64:12, 2020.
[16] Y.-H. Wen, Z. Yang, H. Fu, L. Gao, Y. Sun, and Y.-J. Liu, “Autoregressive stylized motion synthesis with generative flow,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 13612–13621.
[17] L. A. Gatys, A. S. Ecker, and M. Bethge, “A neural algorithm of artistic style,” 2015, *arXiv:1508.06576*.
[18] D. P. Kingma and P. Dhariwal, “Glow: Generative flow with invertible 1x1 convolutions,” 2018, *arXiv: 1807.03039*.
[19] D. Vogel, P. Lubos, and F. Steinicke, “AnimationVR - interactive controller-based animating in virtual reality,” in *Proc. IEEE Conf. Virtual Reality 3D User Interfaces*, 2018, pp. 1–1.
[20] Y. Pan and K. Mitchell, “PoseMMR: A collaborative mixed reality authoring tool for character animation,” in *Proc. IEEE Conf. Virtual Reality 3D User Interfaces Abstr. Workshops*, 2020, pp. 758–759.
[21] N. Lockwood and K. Singh, “Finger walking: Motion editing with contact-based hand performance,” in *Proc. 11th ACM SIGGRAPH/Eurograph. Conf. Comput. Animation*, 2012, pp. 43–52.
[22] H. Zhang, S. Starke, T. Komura, and J. Saito, “Mode-adaptive neural networks for quadruped motion control,” *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–11, 2018.
[23] D. Holden, O. Kanoun, M. Perepichka, and T. Popa, “Learned motion matching,” *ACM Trans. Graph.*, vol. 39, no. 4, pp. 53:1–53:12, 2020.
[24] S. Starke, I. Mason, and T. Komura, “DeepPhase: Periodic autoencoders for learning motion phase manifolds,” *ACM Trans. Graph.*, vol. 41, no. 4, pp. 1–13, 2022.
[25] I. Mason, S. Starke, and T. Komura, “Real-time style modelling of human locomotion via feature-wise transformations and local motion phases,” *Proc. ACM Comput. Graph. Interactive Techn.*, vol. 5, no. 1, pp. 1–18, 2022.
[26] H.-Y. Lee et al., “Dancing to music,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 3586–3596.
[27] Z. Wang, J. Chai, and S. Xia, “Combining recurrent neural networks and adversarial training for human motion synthesis and control,” *IEEE Trans. Visual. Comput. Graph.*, vol. 27, no. 1, pp. 14–28, Jan. 2019.
[28] P. Li, K. Aberman, Z. Zhang, R. Hanocka, and O. Sorkine-Hornung, “Ganimator: Neural motion synthesis from a single sequence,” *ACM Trans. Graph.*, vol. 41, no. 4, 2022, Art. no. 138.
[29] M. Petrovich, M. J. Black, and G. Varol, “Action-conditioned 3D human motion synthesis with transformer VAE,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 10985–10995.
[30] S. Alexanderson, G. E. Henter, T. Kucherenko, and J. Beskow, “Style-controllable speech-driven gesture synthesis using normalising flows,” *Comput. Graph. Forum*, vol. 39, no. 2, pp. 487–496, 2020.
[31] X. Huang and S. Belongie, “Arbitrary style transfer in real-time with adaptive instance normalization,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1501–1510.
[32] L. Ma and Z. Deng, “Real-time face video swapping from a single portrait,” in *Proc. Symp. Interactive 3D Graph. Games*, 2020, pp. 1–10.
[33] Z. Xu et al., “APES: Articulated part extraction from sprite sheets,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11625–11634.
[34] M. Unuma, K. Anjyo, and R. Takeuchi, “Fourier principles for emotion-based human figure animation,” in *Proc. 22nd Annu. Conf. Comput. Graph. Interactive Techn.*, 1995, pp. 91–96.
[35] E. Hsu, K. Pulli, and J. Popović, “Style translation for human motion,” *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1082–1089, 2005.
[36] D. Holden, J. Saito, and T. Komura, “A deep learning framework for character motion synthesis and editing,” *ACM Trans. Graph.*, vol. 35, no. 4, pp. 1–11, 2016.
[37] H. Du et al., “Stylistic locomotion modeling with conditional variational autoencoder,” in *Proc. 40th Annu. Conf. Eur. Assoc. Comput. Graph.*, 2019, pp. 9–12.
[38] D. Rezende and S. Mohamed, “Variational inference with normalizing flows,” in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1530–1538.
[39] L. Dinh, J. Sohl-Dickstein, and S. Bengio, “Density estimation using real NVP,” 2016, *arXiv:1605.08803*.
[40] A. J. Bell and T. J. Sejnowski, “An information-maximization approach to blind separation and blind deconvolution,” *Neural Comput.*, vol. 7, no. 6, pp. 1129–1159, 1995.
[41] P. Izmailov, P. Kirichenko, M. Finzi, and A. G. Wilson, “Semi-supervised learning with normalizing flows,” in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 4615–4630.
[42] S. Alexanderson and G. E. Henter, “Robust model training and generalisation with studentising flows,” 2020, *arXiv: 2006.06599*.
[43] T. Salimans and D. P. Kingma, “Weight normalization: A simple reparameterization to accelerate training of deep neural networks,” *Adv. Neural Inf. Process. Syst.*, vol. 29, pp. 901–909, 2016.
[44] A. Vaswani et al., “Attention is all you need,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[45] R. Child, S. Gray, A. Radford, and I. Sutskever, “Generating long sequences with sparse transformers,” 2019, *arXiv: 1904.10509*.
[46] R. Huang, H. Hu, W. Wu, K. Sawada, M. Zhang, and D. Jiang, “Dance revolution: Long-term dance generation with music via curriculum learning,” 2020, *arXiv:2006.06119*.
[47] M. E. Yumer and N. J. Mitra, “Spectral style transfer for human motion between independent actions,” *ACM Trans. Graph.*, vol. 35, no. 4, pp. 1–8, 2016.
[48] I. Mason, S. Starke, H. Zhang, H. Bilen, and T. Komura, “Few-shot learning of homogeneous human locomotion styles,” *Comput. Graph. Forum*, vol. 37, no. 7, pp. 143–153, 2018.
[49] L. v. d. Maaten, “Visualizing datausing t-SNE,” *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.
[50] P. J. Rousseeuw, “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis,” *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, 1987.
[51] F. G. Harvey, M. Yurick, D. Nowrouzezahrai, and C. Pal, “Robust motion in-betweening,” *ACM Trans. Graph.*, vol. 39, no. 4, pp. 60:1–60:12, 2020.

Bin Ji received the BS degree in mechanical engineering from Xidian University, the MS degree in computer science and engineering from Shanghai Jiao Tong University. He is currently working toward the PhD degree in computer science and engineering with Shanghai Jiao Tong University. He is a member of Character Lab, Shanghai Jiao Tong University, Shanghai, China. His research interest includes computer graphics and computer vision.

Ye Pan received the BSc degree in communication and information engineering from Purdue/UESTC, in 2010 and the PhD degree in computer graphics from the University College London (UCL), in 2015. She is currently an associate professor with Shanghai Jiao Tong University. Her research interests include AR/VR, avatars/characters, 3D animations, HCI, and computer graphics. Previously, she was an associate research scientist with AR/VR, Disney Research Los Angeles. She has served as associate editor of the International Journal of Human Computer Studies, and a regular member of IEEE virtual reality program committees.

Yichao Yan received the BE and PhD degrees in electrical engineering from Shanghai Jiao Tong University, in 2013 and 2019, respectively. He is currently an assistant professor with the AI Institute, Shanghai Jiao Tong University. He has authored/coauthored more than 20 peer-reviewed papers, including those in highly regarded journals and conferences such as *IEEE Transactions on Pattern Analysis and Machine Intelligence*, CVPR, ECCV, ACMMM, IJCAI, TMM, etc. His research interests include 3D generation, video analysis, and deep learning.

Ruizhao Chen received the BE degree in computer science and engineering from Shanghai Jiao Tong University. He is a member of Digital Media & Computer Vision Laboratory (DMCV), Shanghai Jiao Tong University, Shanghai, China. His main areas of research interest are computer graphics and computer vision, especially the application of neural network algorithms in both areas.

Xiaokang Yang (Fellow, IEEE) received the BS degree from Xiamen University, Xiamen, China, in 1994, the MS degree from the Chinese Academy of Sciences, Beijing, China, in 1997, and the PhD degree from Shanghai Jiao Tong University, Shanghai, China, in 2000. He is currently a distinguished professor with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University. His research interests include visual signal processing and communication, media analysis and retrieval, and pattern recognition. He serves as an associate editor of the *IEEE Transactions on Multimedia* and the *IEEE Signal Processing Letters*.