# Autoregressive Stylized Motion Synthesis with Generative Flow

Yu-Hui Wen[1†], Zhipeng Yang[2†], Hongbo Fu[3], Lin Gao[2,4*], Yanan Sun[1], Yong-Jin Liu[1*]
[1]CS Dept, BNRist, Tsinghua University
[2]University of Chinese Academy of Sciences
[3]School of Creative Media, City University of Hong Kong
[4]Beijing Key Laboratory of Mobile Computing and Pervasive Device, ICT, CAS
{wenyh1616,liuyongjin}@tsinghua.edu.cn, {yangzhipeng19s,gaolin}@ict.ac.cn,
hongbofu@cityu.edu.hk, sunyn20@mails.tsinghua.edu.cn

## Abstract

*Motion style transfer is an important problem in many computer graphics and computer vision applications, including human animation, games, and robotics. Most existing deep learning methods for this problem are supervised and trained by registered motion pairs. In addition, these methods are often limited to yielding a deterministic output, given a pair of style and content motions. In this paper, we propose an unsupervised approach for motion style transfer by synthesizing stylized motions autoregressively using a generative flow model $\mathcal{M}$. $\mathcal{M}$ is trained to maximize the exact likelihood of a collection of unlabeled motions, based on an autoregressive context of poses in previous frames and a control signal representing the movement of a root joint. Thanks to invertible flow transformations, latent codes that encode deep properties of motion styles are efficiently inferred by $\mathcal{M}$. By combining the latent codes (from an input style motion $S$) with the autoregressive context and control signal (from an input content motion $C$), $\mathcal{M}$ outputs a stylized motion which transfers style from $S$ to $C$. Moreover, our model is probabilistic and is able to generate various plausible motions with a specific style. We evaluate the proposed model on motion capture datasets containing different human motion styles. Experiment results show that our model outperforms the state-of-the-art methods, despite not requiring manually labeled training data.*

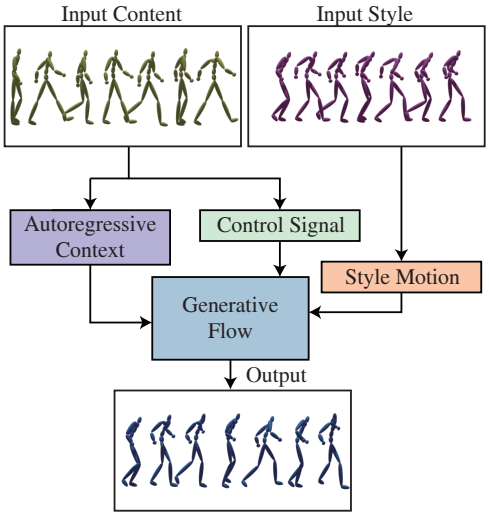



Figure 1. Our stylized motion synthesis that transfers the style from an input style motion to an input content motion. The generative flow model is trained via unsupervised learning on unlabeled motion data of different styles. The trained model extracts the style latent codes from the input style motion. Then, it outputs a high quality stylized motion with the style latent codes (from the input style motion), the autoregressive context and the control signal (from the input content motion).

## 1. Introduction

In computer graphics, there has been a long-standing interest in motion style transfer, since this task benefits various applications including human animation, games, and robotics, etc. Early methods rely on handcrafted features to design different motion styles [38, 2]. To release the burden of handcrafted feature design, data-driven motion style transfer methods using deep learning models have been proposed. They automatically learn useful features from the input motion samples. However, most of the existing deep learning methods [19, 36, 43, 35] are supervised and require paired and registered data to perform style transfer. Such methods also need a large number of motion samples to extract a specific style. Therefore, these methods are limited by a tedious preprocess to collect a large amount of motion

*Corresponding author
†Equal contribution

data for training. For example, actors have to perform several motion cycles in different styles with almost identical steps, followed by a motion registration step.

Recently, Aberman et al. [1] propose a style transfer method, which does not require paired and registered training data. However, the method still requires motion samples with manual labeling styles. It encodes input content and style motions into latent codes for content and style, which are then recombined and decoded to output a stylized motion. Since they use a deterministic model to extract style and content latent codes, the output motion is also deterministic. Moreover, their model is built upon 1D temporal convolutional layers. Thus, its raw outputs have some artifacts, which should be resolved with additional efforts. For example, the foot contact positions of the output motion are corrected in accordance with the input content motion to address the problem of foot skating during walking. Then, dynamic time warping is used to make the global velocity of the output motion to properly reflect the style of the input style motion (e.g., when transferring the style from "old" walking to "neutral" walking, the global velocity should decrease).

In this paper, we propose an unsupervised method for motion style transfer by synthesizing stylized motions based on a generative flow model $\mathcal{M}$. $\mathcal{M}$ is trained to maximize the exact log-likelihood (rather than a lower bound of it in other models such as variational autoencoders (VAEs)) over unlabeled motions of different styles captured in reality. Therefore, the synthesized motions from $\mathcal{M}$ are humanlike and have no artifacts such as foot skating. Compared to [1], the distinguishing feature of our method is to be trained via unsupervised learning and output a high quality stylized motion in a probabilistic way. The probabilistic nature of the generative flow model offers more flexibility to remove the artifacts appearing in the deterministic results in [1]. Through invertible flow transformations, the latent codes that encode deep properties of motion styles are effectively inferred from the input style motion. Then, the inferred latent codes are combined with the input content motion for synthesizing various plausible stylized motions directly. To the best of our knowledge, we are the first to introduce the generative flow model for motion style transfer. We improve the efficiency of $\mathcal{M}$ by performing the flow transformation on one half of the motion features and keeping the other half unchanged, based on the affine coupling layers [6, 7, 22]. Furthermore, a Transformer is imposed into the invertible flow transformation to extract autoregressive features. Experiment results show that the Transformer enables the generative flow model to learn a flexible latent distribution, in which the latent vectors encode deep properties of the motion style that is even unseen during training. Thus, $\mathcal{M}$ has a good scalability by transferring the unseen style from the style motion to the content motion.

In summary, we make three contributions in this paper: (1) We introduce a generative flow model for motion style transfer by synthesizing stylized motions with input style and content motions. The probabilistic nature of the model offers more flexibility to generate high quality stylized motions. (2) Thanks to the invertible flow transformations in the generative model, the latent codes are efficiently inferred from the input style motion to encode deep style features. (3) We impose a Transformer into each invertible flow transformation in our generative model to learn a flexible latent distribution for encoding deep properties of the motion style that is even unseen during training. Thus, our proposed model is able to transfer the unseen style to the content motion.

## 2. Related Work

### 2.1. Motion Synthesis

Motion synthesis methods can be broadly categorized into deterministic and probabilistic methods. The deterministic methods [5, 15, 24, 44] yield a single motion for a given condition (prior poses and/or control signals). They usually regress towards the mean pose, failing to produce distinct and lifelike motions. In contrast, the probabilistic motion synthesis methods are able to generate a range of possible output motions with the given information, by building motion models of all plausible pose sequences. Next, we focus on the probabilistic motion synthesis methods, which are closely related to our work.

Traditional methods [4, 30] assume a Gaussian or Gaussian mixture distribution for motion samples, and use local linear models for probabilistic motion synthesis. In recent years, VAEs have been applied to model human motions along a given path [14] and to generate head motions from speech [11, 12]. Generative adversarial networks (GANs) [34], and adversarial training [9, 42] were also applied to generate motions and similar tasks, such as generating speech-driven videos of talking faces [40, 41, 32, 31]. GANs avoid regression towards the mean pose. However, GANs still have some limitations, such as intractable or ill-defined likelihoods [13]. On the contrary, a less explored methodology normalizing flows (or flows), especially a variant called generative flows, permit tractable and efficient inference [23, 28, 16]. In particular, the generative flow model can be trained efficiently using exact maximum likelihood to describe highly complex motion distributions. Normalizing flows have been used for motion synthesis and motion reconstruction [16, 45]. The aforementioned researches mainly focus on motion synthesis given the movement condition (prior poses and/or control signals). In our work, we develop a generative flow model for stylized motion synthesis based on both the style from the input style motion and the movement condition from the

input content motion.

### 2.2. Motion Style Transfer

Early works for motion style transfer designed handcrafted features in the frequency domain [38] or in the time domain [2] to manipulate styles. Instead of using the handcrafted features, machine learning methods that infer style features from training data have been proposed [19, 36, 43, 35]. Hsu et al. [19] represented the relationships between different styles with a linear time-invariant model [25]. Taylor et al. [36] used restricted Boltzmann machines conditioned on a specific style label to model motion styles. Xia et al. [43] constructed a mixture of regression models by a KNN search over a database of motions to transfer style between motion pairs. Smith et al. [35] later improved the method [43] by using a neural network trained on registered motion pairs. All above methods required a complex collection procedure for pair-wise motion samples in different styles. As a comparison, our proposed method uses unpaired motion collections without style labels for training.

More recently, deep learning has become popular to tackle the problem of character animation controlled by styles [17, 18]. Specifically, the flourishing image style transfer techniques [10, 20] have been successfully adapted to the task of motion style transfer [18, 17, 8, 1]. Gatys et al. [10] showed that image styles can be described by the statistics of features extracted from a pre-trained image classification network. Inspired by the idea of Gatys et al. [10] for image style transfer, Holden et al. [18] proposed a deep learning framework, which enables motion style transfer. They extracted style features by an autoencoder, which was pre-trained based on paired motion clips. However, the method was not efficient, because a slow optimization procedure was required for motion style transfer between each pair of style and content motions. Holden et al. [17] and Du et al. [8] improved the efficiency of the method [10] by replacing the optimization with a feed-forward network for motion style transfer. Huang et al. [20] proposed an Adaptive Instance Normalization (AdaIN) layer to impose different style statistics into a deep learning network, thus enabling apply different styles to an image directly. Aberman et al. [1] proposed to use temporally invariant AdaIN parameters to learn motion styles [20]. They constructed a style transfer network to encode motions into two latent codes for the content and style [1]. Specifically, the style code was extracted by the AdaIN mechanism from the style motion to modify the second-order statistics of the features of the entire content motion.

The aforementioned methods either require a dataset of paired and registered motion data [18, 17, 8], or a dataset of labeled motion data [18, 17, 8, 1]. These methods are thus not well suited for practical applications. In this paper, we propose a generative flow model that can be trained via unsupervised learning over a collection of unlabeled motions.

## 3. Method

Motion style transfer aims to transfer the style from an input style motion $\mathcal{S}$ to an input content motion $\mathcal{C}$, while preserving the content of the latter. In this paper, we propose a new approach for motion style transfer by synthesizing stylized motions based on a generative flow model $\mathcal{M}$. $\mathcal{M}$ enables maximisation of the exact log-likelihood of data samples and efficient inference of latent codes in the latent distribution (Sec. 3.1). To synthesize stylized motions (Sec. 3.2), $\mathcal{M}$ is trained on a collection of motion samples in different styles. In more details, we train the model to parameterise the conditional probability distribution of a pose based on an autoregressive context of the previous poses and a control signal of the root joint movement. Then, a stylized human motion is synthesized autoregressively, by generating poses from the probability distribution with the latent codes from $\mathcal{S}$, the autoregressive context and the control signal from $\mathcal{C}$. Specifically, the latent codes are inferred from $\mathcal{S}$ efficiently (instead of random sampling), because the generative flows in $\mathcal{M}$ are invertible. In addition, we propose a novel flow architecture to make $\mathcal{M}$ more efficient and expressive for modeling motion samples in different styles, as demonstrated in our experiments (Sec. 4.2).

### 3.1. Generative Flow Model

In this subsection, we briefly review the notation of generative flows. Generative flows are conceptually attractive, due to tractability of the exact log-likelihood and efficient latent-code inference [22]. Specifically, a generative flow model has been proposed for image style transfer [22].

Given a dataset $X = \{x_1, ..., x_N\}$ with an unknown complex distribution, it is typical to perform maximum marginal likelihood to learn its parametric model [23]:

$$\log p(X) = \sum_{i=1}^{N} \log p(x_i). \tag{1}$$

To make the marginal likelihood easy to compute and differentiate directly, normalizing flow (or flow) has been introduced in previous works [33, 6, 7].

The key idea of the flow is to transform a simple, fixed distribution $Z$ to obtain a new, more complex distribution $X$. In flow-based generative models [33, 6, 7], a data sample $x$ from the complex distribution can be generated as: $x = g_\theta(z)$. Here, $z$ is a latent variable from the simple distribution $Z$ modeled as $p_\theta(z)$, which has tractable density with parameters $\theta$. The function $g_\theta$ is invertible to allow not only efficient sampling but also efficient infer-

ence. Then, the inference of the latent variable $z$ is done by $z = g_\theta^{-1}(x) = f_\theta(x)$.

To make the invertible function $f$ (with parameter $\theta$ omitted for brevity) flexible and expressive, numerous simpler nonlinear transformations $\{f_k\}_{k=1}^K$ are chained together: $f = f_1 \circ f_2 \circ ... \circ f_K$. Thus, the relationship between $x$ and $z$ can be defined as follows:

$$z \xrightarrow{f_K} h_{K-1} \xrightarrow{f_{K-1}} ... \xrightarrow{f_2} h_1 \xrightarrow{f_1} x, \tag{2}$$

$$x = f(z) = f_1(f_2(...f_K(z))), \tag{3}$$

$$z = f^{-1}(x) = f_K^{-1}(f_{K-1}^{-1}(...f_1^{-1}(x))), \tag{4}$$

where the transformation $f_k$ is named as *flow* and parameterized by $\theta_k$ (omitted in $f_k$ for brevity). Using the change-of-variables formula [27], the log-likelihood of a data sample $x$ can be computed as:

$$\log p_\theta(x) = \log p_\theta(z) + \sum_{k=1}^{K} \log |\det \frac{\partial h_k}{\partial h_{k-1}}|, \tag{5}$$

where $\log |\det \frac{\partial h_k}{\partial h_{k-1}}|$ (called *log-determinant*) is the logarithm of the absolute value of the determinant of the Jacobian matrix $\frac{\partial h_k}{\partial h_{k-1}}$. Computing the log-determinant incurs the complexity close to $\mathcal{O}(D^3)$, which is infeasible for data of high-dimension $D$ [27]. Many flow transformations [33, 23, 22] have been explored to reduce the computation complexity. Based on these works, we propose a novel generative flow architecture for learning motion samples, and it will be described in the following subsection.

### 3.2. Stylized Motion Synthesis

Here, we introduce how to design the architecture of our network based on the notation of generative flows to synthesize realistic stylized human motions.

**Motion Representation.** We represent the motion data by 3D Cartesian coordinates of all joint positions in a skeleton. Specifically, the coordinates are expressed in a root-relative coordinate system, whose origin is on the floor below the root joint. In our implementation, there are in total 21 joints in a human skeleton, resulting in 63 degrees of freedom for the motion. Thus, a motion clip $m \in \mathbb{R}^{T \times d}$ is represented as a sequence of $T$ frames, where each pose $m_t$ has $d = 63$ channels. Moreover, we use a control signal $c \in \mathbb{R}^{T \times 3}$ to represent the movement of the root joint, which is encoded by the forward, sideways, and angular velocity of the root [18].

**Motion Generative Model.** Similar to previous autoregressive generative models [23, 28, 16], our model for motion synthesis is developed as:

$$p(m|c) = p(m_{1:\tau}|c_{1:\tau}) \Pi_{t=\tau+1}^{T} p(m_t|m_{t-\tau:t-1}, c_{t-\tau:t}), \tag{6}$$

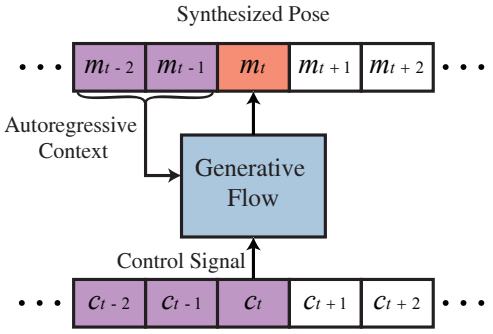



Figure 2. Illustration of our autoregressive motion generative model. The output motion (in orange) $m_t$ at time $t$ is synthesized by the generative flow model with the inputs (in purple) including the autoregressive context of previous poses and the control signal.

where the motion distribution $m_t$ at frame $t$ depends only on $\tau$ previous motions $m_{t-\tau:t-1}$ and the control signal values from the current and previous frames $c_{t-\tau:t}$, as illustrated in Figure 2. Next, we describe how to use flows to learn the generative model effectively.

**Flow Architecture.** Real-valued non-volume preserving (RealNVP) models [6, 7] adopt a type of bipartite flow, which performs nonlinear transformations to one part of the input. It is efficient to do the flow transformation by using one part of the input, because the remaining part is kept unchanged. Inspired by these models, we propose a new type of flow for motion generation, and show its architecture in Figure 3.

In our flow, we denote its input motion features and output latent codes as $a$ and $b$, respectively. Firstly, the input features are split into two equal parts $a = [a', a'']$. Then, we transform half of the input based on the scale and translation parameters extracted from the remaining half with an affine coupling layer, similar to the bipartite flow [6, 7]. Mathematically, the affine coupling operation is defined as:

$$[b', b''] = [a', (a'' + \mu) \odot \delta], \tag{7}$$

where $\delta > 0$ and $\mu$ denote the scaling and translation parameter terms, respectively. The transformation with the affine coupling is invertible as: $a'' = \delta^{-1} b'' - \mu$. In addition, the calculation of its log-determinant is efficient for training and evaluation [23, 28].

The flow described above is not expressive, because a stack of flows alone compute only an affine transformation on half of the input, while doing nothing to the other half [6, 7]. Dinh et al. [6] proposed to shuffle the order of variables between each flow to learn more flexible distributions. Since a linear transformation with equal number of input and output channels can be seen as a generalization of a permutation operation [22], we use a linear transformation $W$ between coupling layers. As illustrated in [22], $W$

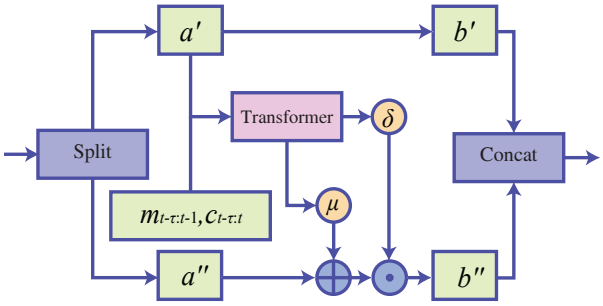



Figure 3. Architecture of our proposed flow. The input motion features of the flow are firstly split into two parts $a'$ and $a''$. A coupling layer applies a Transformer to $a'$, $m_{t-\tau:t-1}$ and $c_{t-\tau:t}$, to extract two transformation parameters, consisting of a translation $\mu$ and a scaling $\delta$. The output of the transformations is $b''$ from $a''$. Finally, $b''$ is concatenated with the other part $b'$, which keeps the same as the input part $a'$. The direction of each arrow shows the transformation of features in the forward propagation. $\odot$ denotes element-wise product and $\oplus$ denotes element-wise sum.

is defined by an LU-decomposition $W = LU$ to simplify the log-determinant. The non-fixed elements in $L$ and $U$ are trainable parameters.

To make our flow more expressive, we further improve the affine coupling by imposing a Transformer [39] to extract parameters for the invertible transformation. In more details, the invertible transformation is proposed based on autoregressive functions, such as recurrent neural networks (RNNs) and Transformers [39]. The autoregressive functions have been successfully applied to many domains for modeling sequential data [26, 3, 37]. Specifically, RNNs have been widely used for the task of modeling motion sequences [26, 3]. However, RNNs generally have a critical drawback of error accumulation. Transformers outperform RNNs by constructing long-range dependencies based on attention mechanisms, and enabling parallelizable training. In this way, we use the Transformer for encoding autoregressive information and extracting transformation parameters, i.e., $\mu, \delta = \mathcal{T}(a', m_{t-\tau:t-1}, c_{t-\tau:t})$. Specifically, the Transformer $\mathcal{T}$ (Figure 3) computes a location-scale transformation based on the input motion features $a'$, the autoregressive context $m_{t-\tau:t-1}$ and the control signal $c_{t-\tau:t}$.

In our model, we improve the flow by imposing the Transformer into the coupling layer. Moreover, the linear transformation is adopted between the coupling layers. Finally, we apply a normalization operation to the whole coupling output to facilitate a deeper architecture and avoid instability problems [6, 7]. Specifically, we adopt an affine transformation using a scale and bias parameter per channel [22] to normalize the output, instead of using batch normalization [6, 7], to save the memory consumption.

**Style Inference and Transfer.** The proposed flow-based generative model $\mathcal{M}$ is trained to learn the distribution of pose at frame $t$ with previous poses $m_{t-\tau:t-1}$ and control signal values in the current and previous frames $c_{t-\tau:t}$. Thus, we can use $\mathcal{M}$ to synthesize poses autoregressively. To control the style of the synthesized motion, we use the latent code $z$ inferred from the input style motion $\mathcal{S}$, instead of a random variable or sampling from a reduced-temperature model [29]. As discussed in Sec. 3.1, invertible flows enable efficient inference (Eq. 4). Given the input style motion $\mathcal{S}$, its latent code can be inferred by $\mathcal{M}$ efficiently. Then, the style is transferred to the input content motion $\mathcal{C}$ by using the latent code of $S$, the autoregressive context and the control signal of $C$ to synthesize output motions. Moreover, we can edit the style latent code inferred from $\mathcal{S}$ in the latent space to generate various plausible stylized motions.

# 4. Experiment and Evaluation

## 4.1. Implementation Details

We use two different datasets with various types of motions [1] to train our generative flow model. The first dataset, captured by Xia et al. [43], contains eight types of motion sequences with different contents (e.g., walking and jumping). The second dataset [1] contains more motion samples, which were performed by a single person in 16 distinct styles. In the following discussions, we refer to the first and second datasets as A and B, respectively. All the motion sequences in both the datasets are downsampled to 30 frames per-second, and sliced into short overlapping clips of 32 frames with overlap of 8. As a result, there are about 1500 motion clips in dataset A, and the clips are split into a train set and a test set with the split ratio of 9:1. In dataset B, the motion clips are split into a train set (18830 clips) and a test set (256 clips).

In the whole network, 16 steps of flow are used in the generative model. The Transformer in each coupling layer consists of two layers, followed by a linear transformation. We implement the model in PyTorch, and train the model to maximize the log-likelihood (minimize the negative log-likelihood) on motion samples of the train set by the Adam optimizer.

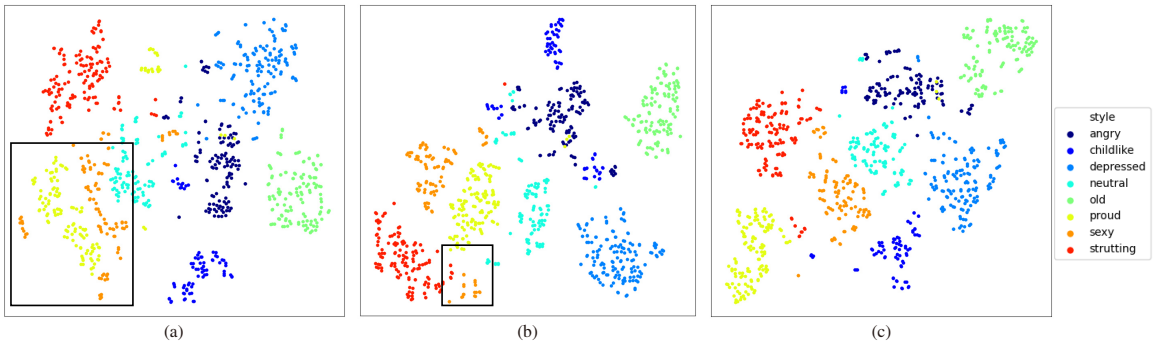
## 4.2. Latent Code Visualization

We show several representative results of stylized motion synthesis by our model in Figure 4. Please find the full motion clips in the supplementary video. The results show that our model can synthesize a motion sequence based on a given style from an input style motion, while retaining the content of an input content motion.

We infer the latent codes by our proposed generative flow model from walking motion samples of dataset A. Then, the latent codes are projected onto a 2D space by using t-distributed stochastic neighbor embedding (t-SNE) [1]. We plot the projected latent codes in 2D to get a better under-

(a) (b) (c)

Figure 4. Samples of two representative style transfer results. In each example, a style motion input (a) and a content motion input (b) are used to synthesize an output motion (c). The style of a kicking motion (in a "strutting" style) is transferred to a jumping motion (in a "sexy" style) in the first row, and the style of a running motion (in an "old" style) is transferred to a walking motion (in an "angry" style) in the second row.




(a) (b) (c)

Figure 5. Style latent codes extracted by our generative flow model are projected onto 2D space using t-SNE, and colored according to their style labels. In the generative flow model, different settings of the coupling layer (i.e., without imposing an autoregressive function (a), with an autoregressive LSTM (b) and with a Transformer (c)) are evaluated. The model is trained on dataset A, which has eight different styles of motions. The latent codes of motions in "sexy" style are clustered more properly by imposing the Transformer into the coupling layer, compared to the other settings.

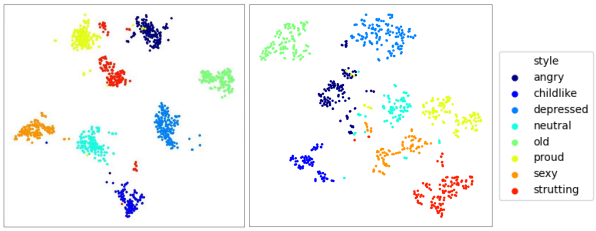
standing of how the generative flows learn to synthesize stylized human motions.

**Latent Style Code.** As previously discussed (Sec. 3.2), we use the latent codes inferred from the generative flow model to control motion styles. We project style latent codes onto 2D space, and color each action sample based on its style label. As shown in Figure 5, the latent codes are clustered into groups of different styles. It demonstrates that our proposed generative flows can learn distributions of motions in different styles, and the latent codes encode deep properties of motion styles. Thus, the motions in the same style may be manipulated in a similar way based on the corresponding latent codes. What's more, it is obvious that the latent codes of the "neutral" style are more centralized in the style latent space than the other styles (as shown in Figure 5). It demonstrates that generative flows tend to learn to branch from the neutral style into the other styles, and it is consistent with common sense.

Next, we evaluate different settings (i.e., without imposing an autoregressive function, with an autoregressive LSTM and with a Transformer) of the flow architecture, as illustrated in Figure 5. We use the flow architecture, in which the coupling layer has no autoregressive function as the baseline. As shown in Figure 5, the latent codes of motions with the "sexy" label are clustered more properly by the generative model with the Transformer, compared to the other two settings. Specifically, we calculate Silhouette Coefficient (SCoeff) and Calinski-Harabaz

| Setting | SCoeff | CHI |
|---|---|---|
| Baseline | 0.36 | 727.51 |
| LSTM | 0.42 | 1049.95 |
| Transformer | **0.48** | **1245.48** |

Table 1. Evaluation of clustering results of latent codes inferred by our generative flow model with different settings. Silhouette Coefficient (SCoeff) and Calinski-Harabaz Index (CHI) are used for the evaluation. Higher SCoeff and CHI values represent better clustering results.




(a) (b)

Figure 6. Illustration of latent codes from unseen styles. The latent codes are extracted by the network of Aberman et al. [1] (a) and our generative flow model (b). Both of the models are trained on dataset A excluding the action samples in the "strutting" style to evaluate their generalization abilities.

| Method | CC | SC |
|---|---|---|
| Holden et al. [18] | $21.6 \pm 6.6$ | $11.3 \pm 5.0$ |
| Aberman et al. [1] | $57.9 \pm 21.4$ | $69.5 \pm 18.1$ |
| Ours (Zero Initialize) | $20.0 \pm 5.2$ | $23.3 \pm 5.5$ |

Table 2. Quantitative comparison of our method to the approaches of Holden et al. [18] and Aberman et al. [1]. Content consistency (CC) and style consistency (SC) are used as evaluation metrics. Lower CC and SC values represent better results .

Index (CHI), to evaluate the clustering results of the latent style codes, which are extracted by our generative flow model with different settings. The clustering results are better with higher values of SCoeff and CHI. In Table 1, it shows that the clustering results of the latent style codes inferred from our generative flow model with the Transformer outperform the other settings. As illustrated in Sec. 3.1, flows are defined to transform a simple, fixed distribution to obtain a more complex distribution. The more flexible the latent distribution, the better the generative model for parameterising the exact distribution of real motion samples. Consequently, the more proper clustered latent codes (motions in the same styles are close to each other in the latent space) can be inferred from the generative flows. The comparison results of latent codes (shown in Table 1 and Figure 5) confirm that our proposed flow transformations with the Transformer is more expressive for encoding motion styles in latent space.

**Unseen Styles.** In practice, our model may be used to extract styles from arbitrary motions. However, it is uncertain whether the model can be successfully used for daily applications, when it is trained on motion samples in limited styles. To verify this, we train our model on dataset A without the motions that are labeled by the "strutting" label, and then test it using the motions including the "strutting" style. As seen in Figure 6, the network of Aberman et al. [1] successfully clusters the samples with styles that can be seen during training. However, the motions in style "strutting" that is never seen during training are adapted to a visually similar style "proud". For example, the "proud" style codes are close to those of "strutting" in Figure 6 (a). On the contrary, our generative flows can not only learn to cluster the action samples in the same style, but also generalize to the samples in an unseen style during training. The experiment demonstrates that the generative flows can infer latent style properties from even unseen style motion samples, implying that the proposed generative flow model has a good generalization ability. We also notice that the latent codes of the same style extracted by the method of Aberman et al. [1] are closer to each other, compared to those inferred from our model. The reason is that Aberman et al. [1] train their model on motion samples with style labels and introduce a triplet loss to make the latent code clusters tighter, while our model is trained in an unsupervised manner.

### 4.3. Comparison

We compare our method with the related works of Holden et al. [18] and Aberman et al. [1], both of which perform a similar task for motion style transfer. Similar to the seminal work of Gatys et al. [10] for image style transfer, Holden et al. [18] perform style transfer by optimizing a motion sequence to satisfy the constraints of both the content and the style. Aberman et al. [1] perform style transfer by encoding motions into two latent codes, one for content and the other for style. During the process of style transfer, the style code modifies the content features by adopting the temporally invariant AdaIN [21]. Then, a post processing is required to match the foot contact and global velocity of the output motion and those of the input motions.

**Quantitative Evaluation.** If the input content motion and the input style motion share the same style label, it is more reasonable that the output motion of style transfer is more close to the content motion. With this observation, we use all pair-wise motion sequences from the test collection (containing 56 motion sequences) in dataset A with the same style labels to generate output motions. Then, we compute the average Euclidean distance between the output motions and the corresponding content motions along the temporal dimension, to evaluate the content consistency (CC). Since the test motion clips in dataset A may have different lengths, we make the pair-wise test motions

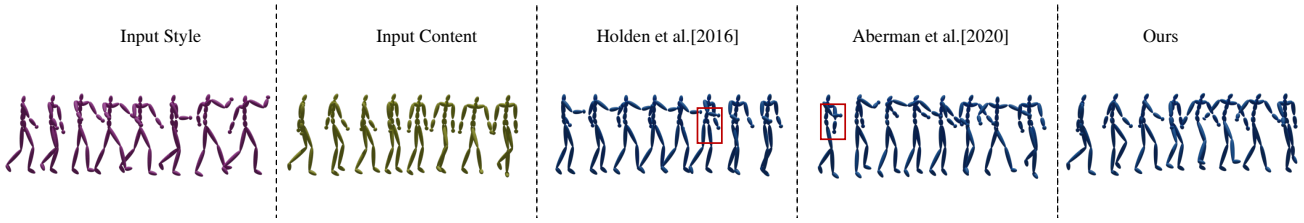



Figure 7. Qualitative comparison of our method to the approaches of Holden et al. [18] and Aberman et al. [1]. The input and content motions are from dataset B. The full video sequences and more results can be found in the supplemental video.

with the same style labels consistent in length for evaluating the SC.

We compare the CC of our method with the state-of-the-art methods [18, 1] to evaluate the superiority of our method. As shown in Table 2, the stylized motions synthesized by our model outperform those by the other methods in retaining the content input. Specifically, we can generate plausible results with different motions in the initialized frames, such as initialized to zero. As the generative flows learn to maximise the exact log-likelihood on motion data samples, the output motion is realistic and retains content consistency very well for the same label. Similarly, we also evaluate the style consistency (SC). In case the content input and the style input share the same content, it is expected that the output motion of style transfer is close to the style motion.

In Table 2, it can be seen that the approach of Holden et al. [18] can keep the style consistency well, but struggles to keep the content consistency. The main reason is that both content and style representations are manually defined and derived from the same deep features. Specifically, they represent the content as a set of deep features that are extracted by a motion autoencoder, and describe the style by the Gram matrix of those features. This may lead to a dependency of the content and style. Generally, the method of Holden et al. [18] tends to produce an output motion that is more consistent with the input style motion. Aberman et al. [1] impose a content consistency term into their loss function to extract deep properties that are shared among samples of the same style in a more proper way. And, their method performs better in keeping the content consistency than keeping the style consistency. However, the CC and SC values of the method are higher compared to the other two methods. The main reason is that the method performs a global velocity warping, which has great effect on the whole motions.

**Qualitative Evaluation.** Qualitatively, we perform the comparison with the related approaches [18, 1] for motion style transfer. We choose input style and content motions from dataset B, in which the motion samples have complex rotations in the root joint. We use the motions from dataset B to evaluate the superiority of our model to control the motion content. As shown in Figure 7, the content of the motion sequence generated by our method is retained better than the previous works [18, 1]. Specifically, the root rotation movements of our synthesized motion are more similar to those of the content motion compared to the results of the other two methods. What's more, the output motion is generated by our model directly, without any further steps for manipulating the global velocity. Moreover, the motions generate by the related method [1] may have some artifacts like crossing hands, while the motions generated by our model are more realistic.

## 5. Conclusion

In this paper, we present a novel method based on generative flows for motion style transfer by synthesizing stylized motions directly, with the latent code (inferred from the input style motion), the autoregressive context and the control signal (from the input content motion). Specifically, our model can be trained on a collection of motion samples in different styles in an unsupervised manner. As no style labels are required during training, the model can be easily applied in daily life. Moreover, our experiments show that the proposed generative flow model has a good generalization capability for encoding latent style codes, so it can successfully synthesize motions in styles that are unseen during training.

Our model has a limitation in synthesizing stylized motions of characters that have body proportions from those who are not seen during training. However, the problem can be solved by performing motion retargeting before motion synthesis. We notice that motion retargeting is another important topic in the scope of human motion study, and we will study it in the future work. Another future work is to solve the problem of synthesizing stylized human motions in different skeletal structures in an end-to-end manner.

## 6. Acknowledgments

This work was partially supported by NSFC (61725204, 62061136007), MOE-Key Laboratory of Pervasive Computing.

## References


[1] Kfir Aberman, Yijia Weng, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Unpaired motion style transfer from video to animation. *ACM Trans. Graph.*, 39(4):64, 2020. 2, 3, 5, 7, 8

[2] Kenji Amaya, Armin Bruderlin, and Tom Calvert. Emotion from motion. In *Graphics Interface*, pages 222–229. Canadian Human-Computer Communications Society, 1996. 1, 3

[3] Judith Bütepage, Michael J. Black, Danica Kragic, and Hedvig Kjellström. Deep representation learning for human motion prediction and classification. In *CVPR*, pages 1591–1599. IEEE Computer Society, 2017. 5

[4] Jinxiang Chai and Jessica K. Hodgins. Performance animation from low-dimensional control signals. *ACM Trans. Graph.*, 24(3):686–696, 2005. 2

[5] Chuang Ding, Pengcheng Zhu, and Lei Xie. Blstm neural networks for speech driven head motion synthesis. In *INTERSPEECH*, pages 3345–3349. ISCA, 2015. 2

[6] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. In *ICLR (Workshop)*, 2015. 2, 3, 4, 5

[7] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *ICLR*, 2017. 2, 3, 4, 5

[8] Han Du, Erik Herrmann, Janis Sprenger, Noshaba Cheema, Somayeh Hosseini, Klaus Fischer, and Philipp Slusallek. Stylistic locomotion modeling with conditional variational autoencoder. In *Eurographics (Short Papers)*, pages 9–12. Eurographics Association, 2019. 3

[9] Ylva Ferstl, Michael Neff, and Rachel McDonnell. Multi-objective adversarial gesture generation. In *MIG*, pages 3:1–3:10. ACM, 2019. 2

[10] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *CVPR*, pages 2414–2423. IEEE Computer Society, 2016. 3, 7

[11] David Greenwood, Stephen D. Laycock, and Iain Matthews. Predicting head pose from speech with a conditional variational autoencoder. In *INTERSPEECH*, pages 3991–3995. ISCA, 2017. 2

[12] David Greenwood, Stephen D. Laycock, and Iain Matthews. Predicting head pose in dyadic conversation. In *IVA*, volume 10498, pages 160–169. Springer, 2017. 2

[13] Aditya Grover, Manik Dhar, and Stefano Ermon. Flow-gan: Combining maximum likelihood and adversarial learning in generative models. In *AAAI*, pages 3069–3076. AAAI Press, 2018. 2

[14] Ikhsanul Habibie, Daniel Holden, Jonathan Schwarz, Joe Yearsley, and Taku Komura. A recurrent variational autoencoder for human motion synthesis. In *BMVC*. BMVA Press, 2017. 2

[15] Dai Hasegawa, Naoshi Kaneko, Shinichi Shirakawa, Hiroshi Sakuta, and Kazuhiko Sumi. Evaluation of speech-to-gesture generation using bi-directional lstm network. In *IVA*, pages 79–86. ACM, 2018. 2

[16] Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. Moglow: Probabilistic and controllable motion synthesis using normalising flows. *CoRR*, abs/1905.06598, 2019. 2, 4

[17] Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Trans. Graph.*, 36(4):42:1–42:13, 2017. 3

[18] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35(4):138:1–138:11, 2016. 3, 4, 7, 8

[19] Eugene Hsu, Kari Pulli, and Jovan Popovic. Style translation for human motion. *ACM Trans. Graph.*, 24(3):1082–1089, 2005. 1, 3

[20] Xun Huang and Serge J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, pages 1510–1519. IEEE Computer Society, 2017. 3

[21] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pages 4401–4410. Computer Vision Foundation / IEEE, 2019. 7

[22] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, pages 10236–10245, 2018. 2, 3, 4, 5

[23] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *NIPS*, pages 4743–4751, 2016. 2, 3, 4

[24] Taras Kucherenko, Dai Hasegawa, Gustav Eje Henter, Naoshi Kaneko, and Hedvig Kjellström. Analyzing input and output representations for speech-driven gesture generation. In *IVA*, pages 97–104. ACM, 2019. 2

[25] L. Ljung. *System identification – theory for the user*. Prentice Hall, 1999. 3

[26] Julieta Martinez, Michael J. Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *CVPR*, pages 4674–4683. IEEE Computer Society, 2017. 5

[27] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. In *ICLR Workshop*, 2017. 4

[28] George Papamakarios, Iain Murray, and Theo Pavlakou. Masked autoregressive flow for density estimation. In *NIPS*, pages 2338–2347, 2017. 2, 4

[29] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, volume 80, pages 4052–4061. PMLR, 2018. 5

[30] Vladimir Pavlovic, James M. Rehg, and John MacCormick. Learning switching linear models of human motion. In *NIPS*, pages 981–987. MIT Press, 2000. 2

[31] Hai Xuan Pham, Yuting Wang, and Vladimir Pavlovic. Generative adversarial talking head: Bringing portraits to life with a weakly supervised neural network. *CoRR*, abs/1803.07716, 2018. 2

[32] Albert Pumarola, Antonio Agudo, Aleix M. Martínez, Alberto Sanfeliu, and Francesc Moreno-Noguer. Ganimation: Anatomically-aware facial animation from a single image. In *ECCV (10)*, volume 11214, pages 835–851. Springer, 2018. 2

[33] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, volume 37, pages 1530–1538, 2015. 3, 4

[34] Najmeh Sadoughi and Carlos Busso. Novel realizations of speech-driven head movements with generative adversarial networks. In *ICASSP*, pages 6169–6173. IEEE, 2018. 2

[35] Harrison Jesse Smith, Chen Cao, Michael Neff, and Yingying Wang. Efficient neural networks for real-time motion style transfer. *Proc. ACM Comput. Graph. Interact. Tech.*, 2(2):13:1–13:17, 2019. 1, 3

[36] Graham W. Taylor and Geoffrey E. Hinton. Factored conditional restricted boltzmann machines for modeling motion style. In *ICML*, volume 382, pages 1025–1032. ACM, 2009. 1, 3

[37] Dustin Tran, Keyon Vafa, Kumar Krishna Agrawal, Laurent Dinh, and Ben Poole. Discrete flows: Invertible generative models of discrete data. In *NeurIPS*, pages 14692–14701, 2019. 5

[38] Munetoshi Unuma, Ken ichi Anjyo, and Ryozo Takeuchi. Fourier principles for emotion-based human figure animation. In *SIGGRAPH*, pages 91–96. ACM, 1995. 1, 3

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017. 5

[40] Konstantinos Vougioukas, Stavros Petridis, and Maja Pantic. End-to-end speech-driven facial animation with temporal gans. In *BMVC*, page 133. BMVA Press, 2018. 2

[41] Konstantinos Vougioukas, Stavros Petridis, and Maja Pantic. Realistic speech-driven facial animation with gans. *Int. J. Comput. Vis.*, 128(5):1398–1413, 2020. 2

[42] Zhiyong Wang, Jinxiang Chai, and Shihong Xia. Combining recurrent neural networks and adversarial training for human motion synthesis and control. *CoRR*, abs/1806.08666, 2018. 2

[43] Shihong Xia, Congyi Wang, Jinxiang Chai, and Jessica K. Hodgins. Realtime style transfer for unlabeled heterogeneous human motion. *ACM Trans. Graph.*, 34(4):119:1–119:10, 2015. 1, 3, 5

[44] Youngwoo Yoon, Woo-Ri Ko, Minsu Jang, Jaeyeon Lee, Jaehong Kim, and Geehyuk Lee. Robots learn social skills: End-to-end learning of co-speech gesture generation for humanoid robots. In *ICRA*, pages 4303–4309. IEEE, 2019. 2

[45] Andrei Zanfir, Eduard Gabriel Bazavan, Hongyi Xu, William T Freeman, Rahul Sukthankar, and Cristian Sminchisescu. Weakly supervised 3d human pose and shape reconstruction with normalizing flows. In *European Conference on Computer Vision*, pages 465–481, 2020. 2