Bailando: 3D Dance Generation by Actor-Critic GPT with Choreographic Memory

Li Siyao[1] Weijiang Yu[2] Tianpei Gu[3,4] Chunze Lin[4]
Quan Wang[4] Chen Qian[4] Chen Change Loy[1] Ziwei Liu[1] ✉
[1]S-Lab, Nanyang Technological University
[2]Sun Yat-Sen University [3]UCLA [4]SenseTime Research
siyao002@e.ntu.edu.sg weijiangyu8@gmail.com gutianpei@ucla.edu
{linchunze, wangquan, qianchen}@sensetime.com {ccloy, ziwei.liu}@ntu.edu.sg

Figure 1: **Dance examples generated by our proposed method** on various types of music. The character is from Mixamo [1]

Abstract

*Driving 3D characters to dance following a piece of music is highly challenging due to the **spatial** constraints applied to poses by choreography norms. In addition, the generated dance sequence also needs to maintain **temporal** coherency with different music genres. To tackle these challenges, we propose a novel music-to-dance framework, **Bailando**, with two powerful components: **1)** a choreographic memory that learns to summarize meaningful dancing units from 3D pose sequence to a quantized codebook, **2)** an actor-critic Generative Pre-trained Transformer (GPT) that composes these units to a fluent dance coherent to the music. With the learned choreographic memory, dance generation is realized on the quantized units that meet high choreography standards, such that the generated dancing sequences are confined within the spatial constraints. To achieve synchronized alignment between diverse motion tempos and music beats, we introduce an actor-critic-based reinforcement learning scheme to the GPT with a newly-designed beat-align reward function. Extensive experiments on the standard benchmark demonstrate that our proposed framework achieves state-of-the-art performance both qualitatively and quantitatively. Notably, the learned choreographic memory is shown to discover human-interpretable dancing-style poses in an unsupervised manner. Code and video demo are available at https://github.com/lisiyao21/Bailando/.*

✉ Corresponding author

1. Introduction

Music-conditioned 3D dance generation is an important task for its huge potential to facilitate a variety of real-world applications, *e.g.*, assisting human artists choreograph and driving virtual characters performance. However, to produce satisfactory dancing sequence on given music is still very difficult due to two main challenges: **1) Spatial constraint:** Not all the physically feasible 3D human poses are applicable for dance. The *subspace* of dancing-style poses has stricter positional standards on body, and is selective to be visually expressive and emotionally infectious based on the choreography norms. **2) Temporal coherency with music:** The generated dancing sequence should be consistent with the music rhythm on various genres of beats, while keeping the whole movements fluent.

Most existing dance generation studies intend to solve the two challenges both in a single ingeniously designed network that directly maps music to 3D joint sequence in high-dimensional continuous space [3, 19, 37, 11, 2, 30]. However, such methods are usually unstable in practice and are prone to regress to nonstandard poses beyond the dancing *subspace*, *e.g.*, freezing or meaningless swaying. Because there is no explicit constraints on target domain to restrict the synthesized dance to be spatially qualified. To deal with the spatial constraint, some works collect real dancing clips as *dance unit* and choreograph by splicing these units [43, 18]. While these methods guarantee the spatial quality of generated dance by directly manipulating on real data, the collection of dance units costs tremendous manual efforts, and they are not compatible with different rhythms. In addi-

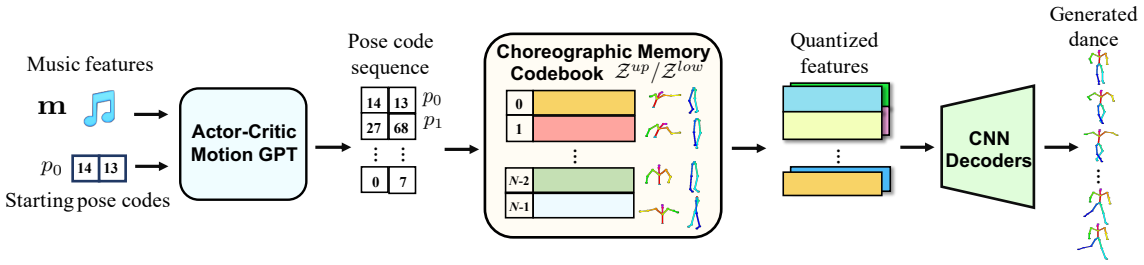


Figure 2: **Dance generation pipeline of *Bailando*.** Given a piece of music, an actor-critic motion GPT autoregressively predicts the future upper-lower pose code pairs according to the music features and starting pose codes. The pose code sequence is then embedded to quantized features via a learned choreographic memory and finally decoded into a dance sequence by a CNN-based decoder.

tion, the units cannot be reused for different kinds of music beats due to their fixed length and speed.

In view of the shortcomings of existing methods, we propose a novel dance generation framework, ***Bailando***, that possesses two main components aiming at the spatial and temporal challenges, respectively. First, to address the spatial challenge, a finite dictionary of quantized dancing units, namely *choreographic memory*, is made by summarizing fundamental and reusable constituents from movements in the dancing-style subspace. Instead of manually indicating the dance units, we leverage the recent advances of VQ-VAE [38] to encode and quantize 3D joint sequence to a codebook in an unsupervised manner, where each learned code is shown to represent a unique dancing pose. To further enlarge the range that choreography memory can represent, we divide 3D poses into compositional upper and lower half bodies and learn VQ-VAEs for the half bodies separately, such that any piece of dance can be represented into a sequence of paired pose codes.

Second, to generate temporally harmonic dance sequence, a GPT-like [34] network, named motion GPT, is introduced to translate music and source pose codes to targeted future pose codes. Since the 3D poses are divided into compositional half bodies in the choreographic memory, we enhance our motion GPT with proposed cross-conditional causal attention layer to keep the coherence of the generated body. Moreover, to achieve accurate temporal synchronization between diverse motion tempos and music beats, we apply an on-policy reinforcement learning scheme to further improve the motion GPT via actor-critic [22] finetuning with a newly-designed beat-align reward function.

The inference procedure of *Bailando* is shown in Figure 2. Given a piece of music and a starting pose code pair, the actor-critic GPT autoregressively predicts the future pose code sequence, which are then embedded to corresponding quantized features in choreographic memory, and are finally decoded and composed to 3D dance sequence by the dedicated CNN-based decoders of learned pose VQ-VAE.

The contributions of our work can be summarized in three folds: **1)** A choreographic memory is created to encode and quantize dancing-style 3D poses, which is achieved by VQ-VAE in an unsupervised manner. **2)** To align diverse motion tempos with different genres of music beats, an actor-critic GPT incorporated with the choreographic memory and cross-conditional causal attention is introduced. **3)** Extensive experiments show that our proposed *Bailando* significantly outperforms the existing state of the art on both automatic metrics and visualization judgements. Code and models will be released upon acceptance.

2. Related Work

Motion Synthesis and Music to Dance. Producing realistic human motions has been long studied. A typical class of approaches is *graph-based* methods. They are developed on the idea of "cropping and pasting", which cut motion clips from existing data as individual nodes and splice these nodes to synthesize new motions according to proper rules [24, 4, 23, 26]. For music to dance, further constraints on the music rhythms, including source-target music similarity [27], beat-wise motion connectivity [10], and deep rhythm signatures [18], are introduced into the linking rules of the graph-based methods to align the motion with music beats. However, since the tempos, length, and speed of the cropped dance units are fixed, the graph-based methods would encounter temporal conflicts on diverse rhythms. For example, the dance units cropped in music of 4/4 time signature cannot synthesize movement for 3/4, while the motion tempos of 60 beats per minute (BPM) is not adaptable for 80 BPM. As a result, this kind of works can perform well in restricted rhythm ranges but is not compatible with various genres of music beats in wild scenarios. In recent years, with the emergence of deep learning, many works design a dedicated network structure, including CNNs [14], RNNs [37, 3, 40, 15], GCNs [41, 35, 9], GANs [25, 36] and Transformers [29, 30, 28], to map the given music to a joint sequence of the continuous human pose space directly. Due to lacking explicit restrictions to keep the generated pose within the spatial constraint, such methods would regress to nonstandard poses that are beyond the dancing *subspace* during inference, resulting in instability in real uses. Besides various kinds of methods, different 3D dancing sequence data are made from mocap and reconstruction [37, 3, 44]. Recently, a large-scale 3D dancing dataset AIST++ [30] is built from multi-camera videos along with the music in different styles and speeds,

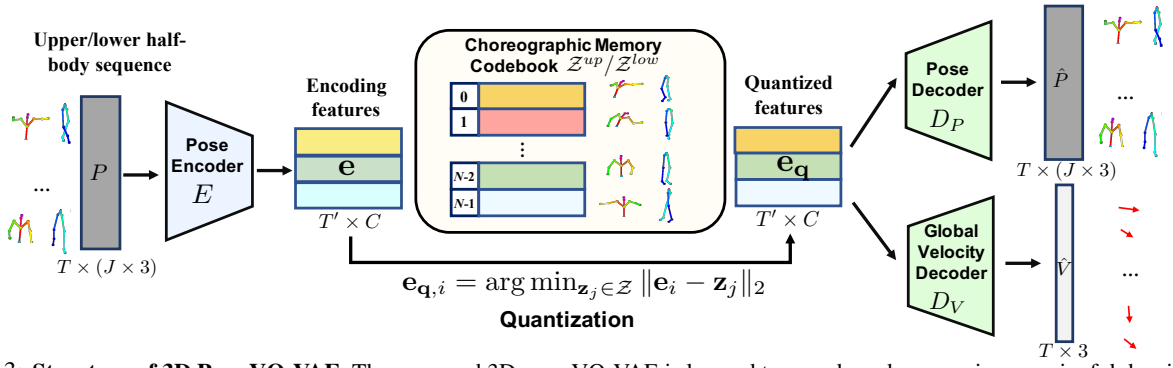


Figure 3: **Structure of 3D Pose VQ-VAE.** The proposed 3D pose VQ-VAE is learned to encode and summarize meaningful dancing units to choreographic memory, and to reconstruct the target pose sequence from quantized features. The parameters of encoder and decoders and the codebook are jointly learned during training.

facilitating both training and testing of this task.

Two-Stage Generation. The two-stage approaches, which first encode data and afterwards learn a probabilistic model to generate the encoding, have been applied in multiple generative areas [7, 42, 8]. For example, Dhariwal *et al*. [7] extracts audio features and generate songs according to the lyrics, while most recently Esser *et al*. [8] encode perceptually rich image constituents to quantized patches and tames the Transformer to generate contextually plausible images in large resolutions. In our work, we encode and quantize meaningful dancing constituents into a choreographic memory and generate visually satisfactory dance by jointly translating the music and existing movements to targeted future poses.

3. Our Approach

The overview of our dance generation framework, *Bailando*, is shown in Figure 2. Unlike other learning based methods, we do not learn a direct mapping from audio features to the continuous domain of 3D joint sequence. Instead, we first encode and quantize the spatially standard dance movements into a finite codebook $\mathcal{Z} = \{\mathbf{z}_i\}_{i=0}^{N-1}$ as choreographic memory in Section 3.1, where N is the codebook length and every code $\mathbf{z}_i$ is shown to represent a dancing-like pose with contextual semantic information. Specifically, we learn VQ-VAEs on the upper and lower half bodies separately, and represent the dance movement into a sequence of compositional upper-and-lower pose code pairs $\boldsymbol{p} = [p^u, p^l]$. Then, we introduce a motion GPT to translate the music feature and source pose codes to the future pose codes in Section 3.2. Furthermore, to achieve synchronized alignment between generated motion tempos and music beats, we propose actor critic learning on the motion GPT with our newly designed beat-align rewards in Section 3.3. The generated pose code sequences are finally decoded composed to fluent 3D dance by VQ-VAE decoders.

3.1. 3D Pose VQ-VAE with Choreographic Memory

Dance positions, *i.e*., the meaningful poses in dancing movements, are the basic constituents of a piece of dance and the process of choreography can be regarded as the combinations and connections of dance positions. Although dances can vary greatly in style or speed, they share common dance positions. Instead of indicating fixed units of dance with plenty of manual efforts, our goal is to summarize such dance positions into a rich and reusable codebook in an unsupervised manner, such that any piece of dance $P \in \mathbb{R}^{T\times(J\times 3)}$, where T is the time length and J is joint amount, can be represented by a sequence of codebook elements $\mathbf{e_q} \in \mathbb{R}^{T'\times C}$, where $T' = T/d$, d is the temporal downsampling rate, and C is the channel dimension of features.

To collect distinctive pose codes as well as to reconstruct them back to represented dancing sequence efficiently, we design a 3D pose VQ-VAE as shown in Figure 3. In this scheme, we first adopt a 1D temporal convolution network E to encode the 3D joint sequence P to context-aware features $\mathbf{e} \in \mathbb{R}^{T'\times C}$. Then, we quantize $\mathbf{e}$ by substituting each temporal feature $\mathbf{e}_i$ to its closest codebook element $\mathbf{z}_j$ as

$$\mathbf{e}_{\mathbf{q},i} = \arg\min_{\mathbf{z}_j \in \mathcal{Z}} \|\mathbf{e}_i - \mathbf{z}_j\|. \quad (1)$$

Finally, we decode the quantized features $\mathbf{e_q}$ via a CNN D_P and reconstruct the dance movement $\hat{P}$.

Compositional Human Pose Representation. In order to represent a larger range of motions by training on limited dance data, we train independent 3D pose VQ-VAEs and learn two separate codebooks $\mathcal{Z}^u$ and $\mathcal{Z}^l$ for the upper and lower half bodies, respectively, such that we can combine different upper-lower code pairs to enlarge the range of dance positions that the learned codebooks can cover. Meanwhile, to avoid encoding confusion caused by global shift of joints (*e.g*., the same motion may be encoded to different features when it is at different locations), we normalize the absolute locations of input P, *i.e*., setting the root joints (hips) to be 0. To realize the overall movement, we add a separate decoder branch D_V, which predicts the global movement velocity $\hat{V} \in \mathbb{R}^{T\times 3}$ according to pose codes of the lower half body, where $\hat{V}_t$ represents the shift of root joint between the $(t+1)$-th and the t-th frames.

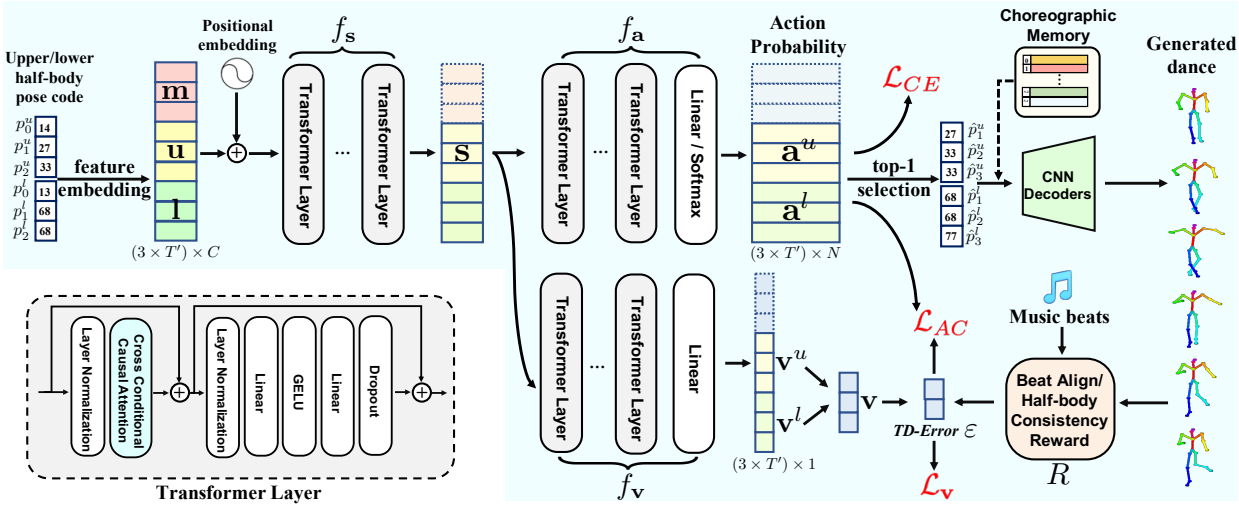


Figure 4: **Actor-Critic GPT.** The GPT is learned to sequentially translate the source pose codes $[p_t^u, p_t^l]$ of upper-and-lower half bodies along with music features $\mathbf{m}$ to the targeted future pose codes $[\hat{p}_{t+1}^u, \hat{p}_{t+1}^l]$. The parameters of the networks are learned via cross-entropy loss $\mathcal{L}_{CE}$ with ground truth and actor-critic loss $\mathcal{L}_{AC}$.

Learning Stable 3D Pose VQ-VAEs. The pose encoder E and decoder D_P are simultaneously learned with the codebook via the following loss function:

$$\mathcal{L}_{VQ} = \mathcal{L}_{rec}(\hat{P}, P) + \|\text{sg}[\mathbf{e}] - \mathbf{e_q}\| + \beta\|\mathbf{e} - \text{sg}[\mathbf{e_q}]\|. \quad (2)$$

The global velocity decoder branch is learned afterwards by fixing the parameters of other parts of VQ-VAE via loss function $\mathcal{L}_{rec}(\hat{V}, V)$, where V is the ground truth global velocity. $\mathcal{L}_{rec}$ is the reconstruction loss that constrains the predicted 3D joint sequence to ground truth. In this loss, we regress not only the original 3D points of joints, but also the velocities and accelerations of movements:

$$\mathcal{L}_{rec}(\hat{P}, P) = \|\hat{P} - P\|_1 + \alpha_1\|\hat{P}' - P'\|_1 + \alpha_2\|\hat{P}'' - P''\|_1, \quad (3)$$

where P' and P'' represent the 1^{st}-order (velocity) and 2^{nd}-order (acceleration) partial derivatives of 3D joint sequence P on time, while α_1 and α_2 are trade-off weights. Experimental results show the "velocity-and-acceleration" loss items play essential roles to prevent jitters in generated dance. (See Section 4.2.)

The second part of $\mathcal{L}_{VQ}$ is the "codebook loss" to learn codebook entries, where sg[·] denotes "stop gradient" [6], while the third part is the "commitment loss" with trade off β [8, 7]. Since the quantization operation of Equation 2 is not differentable, to train the whole networks end to end, the back-propagation of this operation is achieved by simply passing the gradient of $\mathbf{e_q}$ to $\mathbf{e}$.

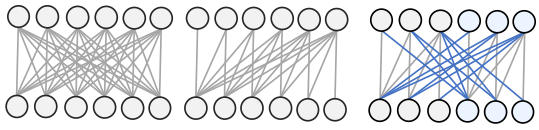
The learned choreographic memory codes are interpretable. After the training process of pose VQ-VAEs, each quantized feature in the codebook is decoded into a unique dance position. And any permutation and combination of codes can be decoded to a piece of fluent movement based on corresponding dance positions. (See Section 4.3.)

3.2. Cross-Conditional Motion GPT

Now that we can represent any piece of dance by a sequence of quantized position codes, the dance generation task is then reframed as to select proper codes from codebook $\mathcal{Z}$ for future actions according to given music and existing movements. For any target time t, we estimate the probability of every $\mathbf{z}_i \in \mathcal{Z}$ and select the one with the largest possibility as the predicted pose code $\hat{\boldsymbol{p}}_t$. Since we model the upper and lower half bodies separately, in order to keep the coherence of composed body and to avoid the asynchronous situation (*e.g.*, the direction of the upper half is opposite to that of the lower), the prediction of the future action should be cross-conditioned between existing upper and lower movements to make the most of mutual information:

$$\begin{cases} \hat{p}_t^u &= \arg\max_k \mathbb{P}(\mathbf{z}_k^u | \mathbf{m}_{1\ldots t}, p_{0\ldots t-1}^u, p_{0\ldots t-1}^l) \\ \hat{p}_t^l &= \arg\max_k \mathbb{P}(\mathbf{z}_k^l | \mathbf{m}_{1\ldots t}, p_{0\ldots t-1}^u, p_{0\ldots t-1}^l) \end{cases} \quad (4)$$

We introduce the powerful GPT model [34] to estimate the action probabilities as shown in Figure 4. Given a dance position code sequence with length of T', we first embed the upper and lower pose codes to learnable features $\mathbf{u} \in \mathbb{R}^{T' \times C}$ and $\mathbf{l} \in \mathbb{R}^{T' \times C}$, respectively, and concatenate them with music features $\mathbf{m}$ on the temporal dimension. Then, we add a learned positional embedding to this concatenated $(3 \times T') \times C$ tensor and feed it to 12 successive Transformer layers, the structure of which is shown in Figure 4. At last, we employ a linear transform and softmax layer to map the output of Transformer layers to normalized action probability $\mathbf{a} \in \mathbb{R}^{(3 \times T') \times N}$, where N is the size of learned codebook and $\mathbf{a}_{t,i}$ reveals the probability of pose code $\mathbf{z}_i \in \mathcal{Z}$ predicted for time $t+1$. The action probabilities for upper and lower half bodies are indexed as $\mathbf{a}_{0:T'-1}^u = \mathbf{a}_{T':2T'-1}$ and $\mathbf{a}_{0:T'-1}^l = \mathbf{a}_{2T':3T'-1}$, respectively.



(a) Full attention (b) Causal attention (c) Cross-conditional causal attention

Figure 5: **Different types of attention layers.** The proposed cross-conditional causal attention realizes causal inferences intra (gray lines) and inter (blue lines) different kinds of components (gray and blue circles). Two kinds of components are shown here for concision, but three (music, upper, lower bodies) are in reality.

In Transformers [39], the attention layer is the core component that determines the computational dependency among sequential elements of data, and is implemented as

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V},\mathbf{M}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T+\mathbf{M}}{\sqrt{C}}\right)\mathbf{V}, \tag{5}$$

where $\mathbf{Q},\mathbf{K},\mathbf{V}$ denote the query, key and value from input, and $\mathbf{M}$ is the mask, which determins the type of attention layers. The most two common types of attentions are "full attention" [39] and "causal attention" [39], where the former realizes the intercommunication of input data at all times while the latter only allows the current and previous data to compute the state for the time of interest. As our goal is to infer the future dance position codes, we adopt the causal attention. However, since the generation of upper and lower half bodies are dependent on each other, we cannot realize the inference by just reordering the sequence of input to fit the causality as previous works [8, 7]. Therefore, we propose an attention layer, namely cross-conditional attention, to comply the causality cross conditioned among features of the music, the upper half and the lower half bodies, where $\mathbf{M}$ is designed to be a 3×3 repeated block matrix with a lower triangular matrix of size T' as its element. As shown in Figure 5, the proposed attention can exchange information of different components, and guarantee that the future information will not be transmitted back to the past.

Learning Motion GPT. The motion GPT is optimized via supervised training with cross-entropy loss on action probability $\mathbf{a}$:

$$\mathcal{L}_{CE} = \frac{1}{T'}\sum_{t=0}^{T'-1}\sum_{h=u,l}\text{CrossEntropy}\left(\mathbf{a}_t^h, p_{t+1}^h\right). \tag{6}$$

Given a sequence of pose codes $\boldsymbol{p}_{0:T'-1}$ and relevant music features $\mathbf{m}_{1:T'}$ as input, the learned GPT outputs the sequence of actions $\mathbf{a}_{0:T'-1}$ all at once to predict $\boldsymbol{p}_{1:T'}$. This parallel characteristic makes Transformer an ideal model for reinforcement learning [16, 5]. In the following subsection, we adopt the learned motion GPT as a pretrained policy maker and propose a novel actor-critic based finetuning scheme to further improve its performance as complementary to the supervised training above.

3.3. Actor-Critic Learning

While the supervised learning scheme for the motion GPT is straightforward and easy to train, it is intractable to further involve a more flexible constraint of generated dance (*e.g.*, a regularization item that strengthens the consistency of dance beats) to Equation (6), since the supervision target is the code number, which is not differentiable to compute the quantitative constraints on the final dance sequence.

To address this issue and to achieve more accurate synchronized alignment between diverse motion tempos and music beats, we apply actor-critic learning to the motion GPT with a newly-designed reward function. In particular, we regard the first 6 Transformer layers of motion GPT as "state network" $f_{\mathbf{s}}$, and the outputs of $f_{\mathbf{s}}$ are states $\mathbf{s}$ for time 0 to $T'-1$, while the latter 6 Transformer layers along with the linear-softmax layer are regarded as "policy making network" $f_{\mathbf{a}}$, where the actions are computed according to state as $\mathbf{a} = f_{\mathbf{s}}(\mathbf{s})$. Besides, we add a separate three-layer Transformer branch as "critic value network" $f_{\mathbf{v}}$ to estimate the critic values $\mathbf{v}_{0:T'-1}\in\mathbb{R}^{T'\times 1}$ as

$$\mathbf{v} = \mathbf{v}^u + \mathbf{v}^l = f_{\mathbf{v}}(\mathbf{s})_{T':2T'-1} + f_{\mathbf{v}}(\mathbf{s})_{2T':3T'-1}. \tag{7}$$

With well defined reward function $R(t) = R(\mathbf{a}_t,\mathbf{s}_t)$, the objective of reinforcement learning is to maximize the expected accumulated rewards:

$$J = \mathbb{E}_\tau\left[\sum_{t=0}^{T'-1} R(t)\right], \tag{8}$$

where $\tau = \{\mathbf{a}_t\}_{t=0}^{T'-1}$ is the trajectories of actions predicted by the policy making network. This objective is then converted to optimize the parameters of policy making network using the following loss function:

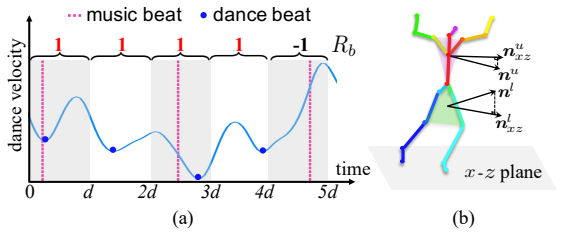
$$\mathcal{L}_{AC} = \frac{1}{T'-1}\sum_{t=0}^{T'-2}\left(\sum_{h=u,l}\text{CrossEntropy}\left(\mathbf{a}_t^h,\hat{p}_{t+1}^h\right)\right)\cdot\text{sg}[\varepsilon_t], \tag{9}$$

where $\hat{p}_{t+1}^h = \arg\max_i \mathbf{a}_{t,i}^h$ is the pose code number predicted by the policy making network. $\boldsymbol{\varepsilon}\in\mathbb{R}^{(T'-1)\times 1}$ denotes the so-called TD-error calculated as

$$\boldsymbol{\varepsilon}_{0:T'-2} = \mathbf{r}_{0:T'-2} + \text{sg}[\mathbf{v}_{1:T'-1}] - \mathbf{v}_{0:T'-2}, \tag{10}$$

where $\mathbf{r}_t = R(t)$. The detailed derivation of Equation (9) can be found in the supplementary file. Meanwhile, the critic value network is optimized by bootstrap training on difference between $\mathbf{v}_{0:T'-2}$ and $R(\mathbf{a}_t,\mathbf{s}_t)+\mathbf{v}_{1:T'-1}$:

$$\mathcal{L}_{\mathbf{v}} = \frac{1}{T'-1}\|\boldsymbol{\varepsilon}\|_2^2. \tag{11}$$



(a) (b)

Figure 6: **Designed rewards.** (a) Beat-align reward penalizes the absence of dance beat for the interval that has music beat. (b) Half-body consistency reward is computed on the angle between normal directions of half bodies to prevent asynchronizations.

The computation of actor-critic loss $\mathcal{L}_{AC}$ depends on real-time actions predicted by the motion GPT and the optimization direction is determined on the value of TD-error. When ε_t is positive, the optimization on $\mathcal{L}_{AC}$ will intensify the convergence to predicted code $\hat{p}_{t+1}$, while in the opposite situations, the probability estimated for $\hat{p}_{t+1}$ will be reduced.

The value of TD-error and the learning effectiveness are strongly influenced by the reward function R. In this work, we design a motion-music beat-align reward to generate dance more accurate to the rhythm of music. As shown in Figure 6 (a), the beat-align reward is defined as

$$R_b(t) = \begin{cases} -1, & \exists \textit{ music beat} \wedge \nexists \textit{ dance beats} \in \hat{P}_{td:(t+1)d} \\ 1, & \textit{otherwise,} \end{cases} \tag{12}$$

where $\hat{P}_{0:T-1} = D(\hat{\boldsymbol{p}}_{0:T'-1})$ is the dance motion sequence decoded from predicted dance position codes. Meanwhile, to avoid the compositional asynchronization between upper and lower half bodies during actor-critic learning, we introduce a compositional consistency reward to impose penalties for situations where the upper and lower body are in the opposite direction:

$$R_c(t) = \inf\left\{\hat{R}_c(t)\right\}, t \in [dt, d(t+1))\,, \tag{13}$$

where

$$\hat{R}_c(t) = \begin{cases} \left\langle \boldsymbol{n}^u_{xz}(t), \boldsymbol{n}^l_{xz}(t) \right\rangle, & \left\langle \boldsymbol{n}^u_{xz}(t), \boldsymbol{n}^l_{xz}(t) \right\rangle < 0 \\ 1, & \textit{otherwise.} \end{cases} \tag{14}$$

Here, $\boldsymbol{n}^u_{xz}(t), \boldsymbol{n}^l_{xz}(t)$ are the normal directions of upper and lower bodies of $\hat{P}_t$ projected to the x-z plane, which is illustrated in Figure 6 (b). The final reward is then a weighted combination of R_b and R_c as $R = \gamma_b R_b + \gamma_c R_c$.

In the finetuning process, we fix the parameters of state network $f_{\mathbf{s}}$, and alternately train the policy making network $f_{\mathbf{a}}$ and the critic value network $f_{\mathbf{v}}$ using the losses introduced above with a small learning rate. After such finetuning, the proposed framework will be further enhanced.

4. Experiments

Dataset. We perform the training and evaluation on the AIST++ dataset proposed in [30], which to our best knowledge is the largest public available dataset for paired music and motions. This dataset contains 992 pieces of high-quality 60-FPS 3D pose sequence in SMPL format [31], where 952 are kept for training and 40 are used for evaluation.

Implementation Details. In this work, the choreographic memory codebook size N for both upper and lower bodies is set to 512, while the channel dimension C of encoded features is 512 and the temporal downsampling rate d of encoders is 8. The structures of the convolutional encoder and decoders are provided in the supplementary file. While training the VQ-VAEs, dance data are cropped to length of $T = 240$ (4 seconds) and sampled in batch size of 32. The commit loss trade-off β in $\mathcal{L}_{VQ}$ is 0.1, while α_1 and α_2 in $\mathcal{L}_{rec}$ are both set to be 1. We adopt Adam optimizer [21] with $\beta_1 = 0.9$ and $\beta_2 = 0.99$ to train both pose VQ-VAEs for 400 epochs with learning rate 3×10^{-5}. As to the motion GPT, we comply a structure mirroring [20], where the channel dimension is 768, and the attention layer is implemented in 12 heads with dropoout probability 0.1. The music features are extracted by the public audio processing toolbox Librosa [17], including *mel frequency cepstral coefficients (MFCC)*, *MFCC delta*, *constant-Q chromagram*, *tempogram* and *onset strength*, which are 438-dim in total, and are mapped to the same dimension of GPT via a learned linear transform. The block size T' of GPT is set to be 29. While training, the dance sequences are first encoded to pose codes $\boldsymbol{p}$ and sampled to length of 30, where $\boldsymbol{p}_{0:28}$ are used as input and $\boldsymbol{p}_{1:29}$ are supervision labels. The motion GPT is optimized using Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.99$ for 400 epochs, where the learning rate is initialized as 3×10^{-4} and decayed after 200 epochs with factor 0.1. In the actor-critic finetuning process, we adopt a small learning rate of 1×10^{-5} to learn $f_{\mathbf{a}}$ and $f_{\mathbf{v}}$ for 10 epochs. The reward trade-offs γ_b and γ_c are 5 and 1, respectively. In our experiment, the pose VQ-VAEs and the motion GPT are trained sequentially, and the weights of VQ-VAEs are fixed during the learning process of GPT. The whole framework is learned in three days on one Tesla V100 GPU. During test, the motion GPT takes a pair of starting pose codes, which can be either manually indicated or randomly sampled, as input and autoregressively generates the motion sequence as long as the target music.

Evaluation Metrics. For quantitative evaluations, we measure the generated dance from three perspectives: the quality of generated dances, the diversity of motions and the alignment between the rhythms of music and generated movements. In concrete, for the dance quality, we calculate the Fréchet Inception Distances (FID) [13] between the generated dance and all motion sequences (including training and test data) of the AIST++ dataset on kinetic features [33] (denoted as 'k') and geometric features [32] (denoted as 'g'), which are both extracted using the toolbox of [12]. As to the diversity, we compute the average feature distance

Table 1: **Quantitative results on AIST++ test set.** The best and runner-up values are bold and underlined, respectively. Among compared methods, "Li *et al.*", DanceNet and FACT are multiplexing the same results of AIST++ benchmark [30], while DanceRevolution [15] is reproduced using officially released code with the optimal settings. † FID_k and DIV_k are fetched from [30] while FID_g and DIV_g are recomputed using the officially updated evaluation code. *The generated dances of "Li *et al.*" are highly jittery making its velocity variation extremely high, which is also reported in [30].

	Motion Quality		Motion Diversity			User Study
Method	$\text{FID}_k \downarrow$	$\text{FID}_g^\dagger \downarrow$	$\text{Div}_k \uparrow$	$\text{Div}_g^\dagger \uparrow$	Beat Align Score ↑	Our Method Wins
Ground Truth	17.10	10.60	8.19	7.45	0.2374	40.0%±25.2%
Li *et al.* [29]	86.43	43.46	6.85*	3.32	0.1607	100.0%±0.0%
DanceNet [44]	69.18	25.49	2.86	2.85	0.1430	92.7%±12.1%
DanceRevolution [15]	73.42	25.92	3.52	4.87	0.1950	84.5%±10.8%
FACT [30]	<u>35.35</u>	<u>22.11</u>	<u>5.94</u>	<u>6.18</u>	<u>0.2209</u>	98.2%±3.9%
Bailando (Ours)	**28.16**	**9.62**	**7.83**	**6.34**	**0.2332**	–

of generated movements following [30]. Regarding to the alignment between music and generated motions, we calculate the average temporal distance between each music beat and its closest dance beat as the Beat Align Score:

$$\frac{1}{|B^m|}\sum_{t^m \in B^m} \exp\left\{-\frac{\min_{t^d \in B^d} \|t^d - t^m\|^2}{2\sigma^2}\right\}, \quad (15)$$

where B^d and B^m record the time of beats in dance and music, respectively, while σ is normalized parameter which is set to be 3 in our experiment.

4.1. Comparison to Existing Methods

We compare our proposed model to several state-of-the-art methods including Li *et al.* [29], DanceNet [44], DanceRevolution [15] and FACT [30]. For each method, we generate 40 pieces of dances in AIST++ test set, and sample the generated dance sequence with length of 20 seconds to compute the evaluation metrics mentioned above. We also calculate the quantitative scores for ground truth data in AIST++ test set and compare it to the generated dances.

The quantitative results are shown in Table 1. According to the comparison, our proposed model consistently performs favorably against all the other existing methods on all evaluations. Specifically, our method improves 7.19 (20%) and 12.49 (56%) than the best compared baseline model FACT on FID_k and FID_g, respectively, and even achieves a better FID_g score than the ground truth (9.62 v.s. 10.60). If look closely to the metrics on these two kinds of features, the kinetic feature is defined on motion velocities and energies, which reflects the physical characteristics of dance, while the geometric feature is defined based on multiple man-made templates of movements, which reflects the quality of choreography. The superiority of our method on both dance quality metrics reveals that *Bailando* not only synthesizes more real-like motions than the compared baseline methods, but also achieves outstanding performance on organizing the movements to dance via the proposed actor-critic GPT scheme with learned choreographic memory. Meanwhile, *Bailando* can generate dance with high choreographic diversity instead of converging to few templates, and also achieves improvement on the correlation between music and motion.

User Study. To further understand the real visual performance of our method, we conduct a user study among the dance sequences generated by each compared method and the ground truth data in AIST++ test set. The experiment is conducted with 11 participants separately. For each participant, we randomly play 50 pairs of comparison videos with a length of around 10 seconds, where each pair contains our result and one competitor's in the same music, and ask the participant to indicate "*which one is dancing better to the music*". The statistics are shown in Table 1. Notably, our method significantly surpasses the compared state-of-art methods with at least 84.5% winning rate. Even in comparison to the ground truth, 40% of our generated dance is voted as the better in average. According to the feedback from participants, our generated dance is more "stable to the rythm" with "higher diversity", while the reason why our method is still not as good as real dances is mainly due to "lacking of long-term regularity and subjective beauty". A detailed winning rate distribution on styles of dance can be referred to the supplementary file.

4.2. Ablation Studies

We conduct ablation studies on the pose VQ-VAEs and the motion GPT, respectively. The quantitative scores are shown in Table 2. The visual comparisons of this study can be also referred to the supplementary video.

Pose VQ-VAE. We explore the effectiveness of the following components: (1) the up-lower half body separation, (2) the global velocity prediction branch, and (3) the velocity-and-acceleration loss used in $\mathcal{L}_{rec}$. We train three variant models without each of the three components, respectively. The motion quality measured for VQ-VAEs is on reconstructed results of ground truth of AIST++ test set. As shown in Table 2, the FID_k and FID_g values for variant "w/o. upper/lower" become worse by 12.98 (46%) and 3.22 (25%), respectively. The VQ-VAE trained on whole body

Table 2: **Ablation study on AIST++ test set** Experiments are conducted on pose VQ-VAE and GPT, respectively.

	Method	$FID_k \downarrow$	$FID_g \downarrow$	BAS ↑
Pose VQ-VAE	Ground Truth	17.10	10.60	–
	w/o. upper/lower	41.21	15.85	–
	w/o. global vel.	70.95	18.52	–
	w/o. vel./acc. loss	30.91	11.87	–
	full pose VQ-VAE	28.23	12.63	–
GPT	w/o. quantization	42.71	147.28	–
	w/o. cross-cond. att.	37.41	15.52	–
	w/o. actor critic	28.75	11.82	0.2245
	full actor-critic GPT	28.16	9.62	0.2332

cannot reconstruct the dancing pose of test set effectively. Therefore, the separate representations of upper-lower half bodies are necessary to enlarge the range of poses that the choreographic memory can cover. As to the global velocity branch, the motion quality scores of "w/o. global vel." sharply drops 42.72 (151%) and 5.89 (47%), respectively, which shows the isolated velocity prediction is critical for representing the dance movement. For "w/o. vel./acc. loss" variant, the FID_k is worsened by 2.68. Although the FID_g value of "w/o. vel./acc. loss" is slightly improved by 0.76, the model produces strong motion jitters if without adopting vel./acc loss for training in the supplementary video.

Motion GPT. For the proposed actor-critic GPT, first, we explore the effect of quantized choreography memory by training a variant GPT directly regress to the encoding features of 3D joint sequence via an L_2 Loss. As shown in Table 2, the FID_g drops 135.41 for variant "w/o. quantization" (compared to "w/o. actor critic", same below), while the generated dance sequences contain frequent jitters in vision, which shows the quantization of dancing positions is essential to our proposed framework. Second, to test the effectiveness of the proposed cross-conditional causal attention, we substitute it to causal attention, and train two motion GPTs for upper and lower half bodies separately. The motion quality scores of "w/o. cross-cond. att." drop 8.66 (30%) and 3.70 (31%), respectively. The main reason for the poor performance is that the generated dances of contain frequent asynchronization of upper and lower half bodies, while the proposed cross-conditional attention layer can effectively prevent such situations via the interaction of information between the half bodies. At last, we compare the motion quality and music-motion consistency between the model with (denoted as "full actor-critic GPT") and without actor critic finetuning (denoted as "w/o. actor critic"). After the actor-critic learning, the beat-align score (BAS) of motion GPT increases from 0.2245 to 0.2332, proving the effectiveness of reinforcement learning scheme with proposed beat-align reward. Meanwhile, by constraining the consistency with music, the actor-critic finetuning process can also enhance the motion quanlity on choreography and saliently improves the FID_g score by 2.20 (19%).

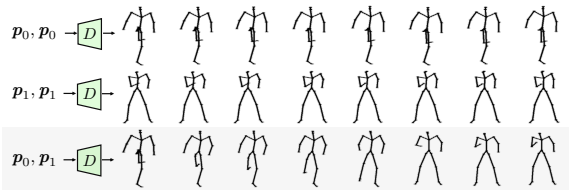


Figure 7: **Interpretability of choreographic memory code.** The sequence of single code is decoded to static pose, while the sequence of two various codes is decoded to smooth transition between two poses, which means each code represents a dancing-style pose and the decoder links poses of different codes to movements.

4.3. Interpretability of Choreographic Memory

In this work, we propose to summarize meaningful dancing units into the codebook via pose VQ-VAE in an unsupervised manner. To understand what kind of dance unit is learned in the choreographic memory, we visualize the latent codes and find each code represents a unique 3D dancing-style pose. As revealed in Figure 7, the first and the second rows are 3D poses decoded from $\boldsymbol{p}_0 = [4, 4]$ and $\boldsymbol{p}_1 = [5, 5]$, respectively, where the former one is doing right leg lifting and the latter is right bicep curl. The decoded pose will keep static for repeating codes, and will make smooth transition between postures of different codes. As shown in the third row of Figure 7, the decoded 3D poses of $[\boldsymbol{p}_0, \boldsymbol{p}_1]$ starts with the posture of $\boldsymbol{p}_0$, while gradually putting down the leg and blending the arm towards the pose of $\boldsymbol{p}_1$. Furthermore, for arbitrary combination of learned choreographic memory codes, the decoders can synthesize fluent movement based on the represented dance positions, which can be referred to the supplementary video. With such characteristic, the choreography process becomes interpretable in proposed *Bailando* as a process of selecting and sorting the quantized dance positions from the learned choreographic memories, instead of a black box as most previous works.

5. Discussion and Conclusion

In this paper, we address the spatial and temporal challenges of 3D dance generation by proposing a novel framework named *Bailando*, which is composed of a choreographic memory to address the spatial constraint by encoding and quantizing dancing-style poses, and an actor-critic GPT to realize the temporal coherency with music that translates and aligns various motion tempos and music beats. Experiments on the standard benchmark (*i.e.*, AIST++ dataset) along with user studies show that *Bailando* achieves state-of-the-art performance both qualitatively and quantitatively.

Acknowledgement. This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-PhD-2022-01-031). This research is conducted in collaboration with SenseTime. This work is supported by NTU NAP and A*STAR through the Industry Alignment Fund - Industry Collaboration Projects Grant. We thank Ruilong Li, Shan Yang and Zhiyuan Chen for their help in this work.

References

[1] Mixamo. https://www.mixamo.com/.

[2] Hyemin Ahn, Jaehun Kim, Kihyun Kim, and Songhwai Oh. Generative autoregressive networks for 3d dancing move synthesis from music. *IEEE Robot. and Automat. Letters*, 5:3501–3508, 2020.

[3] Omid Alemi, Jules Françoise, and Philippe Pasquier. Groovenet: Real-time music-driven dance movement generation using artificial neural networks. *Networks*, 8(17):26, 2017.

[4] Okan Arikan and David A Forsyth. Interactive motion generation from examples. *ACM TOG*, 21(3):483–490, 2002.

[5] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *arXiv preprint arXiv:2106.01345*, 2021.

[6] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021.

[7] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

[8] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021.

[9] João Pedro Klock Ferreira, Thiago M. Coutinho, Thiago L. Gomes, José Francisco Neto, Rafael Azevedo, Renato Martins, and Erickson R. Nascimento. Learning to dance: A graph convolutional adversarial network to generate realistic dance motions from audio. *Comput. Graph.*, 94:11–21, 2021.

[10] Satoru Fukayama and Masataka Goto. Music content driven automated choreography with beat-wise motion connectivity constraints. *SMC*, pages 177–183, 2015.

[11] Shiry Ginosar, Amir Bar, Gefen Kohavi, Caroline Chan, Andrew Owens, and Jitendra Malik. Learning individual styles of conversational gesture. *CVPR*, 2019.

[12] Deepak Gopinath and Jungdam Won. fairmotion - tools to load, process and visualize motion capture data. Github, 2020.

[13] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 30, 2017.

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

[15] Ruozi Huang, Huang Hu, Wei Wu, Kei Sawada, Mi Zhang, and Daxin Jiang. Dance revolution: Long-term dance generation with music via curriculum learning. In *ICLR*, 2021.

[16] Michael Janner, Qiyang Li, and Sergey Levine. Reinforcement learning as one big sequence modeling problem. In *NeurIPS*, 2021.

[17] Yanghua Jin, Jiakai Zhang, Minjun Li, Yingtao Tian, Huachun Zhu, and Zhihao Fang. Towards the automatic anime characters creation with generative adversarial networks. *arXiv preprint arXiv:1708.05509*, 2017.

[18] Chen Kang, Zhipeng Tan, Jin Lei, Song-Hai Zhang, Yuan-Chen Guo, Weidong Zhang, and Shi-Min Hu. Choreomaster: Choreography-oriented music-driven dance synthesis. In *SIGGRAPH*, 2021.

[19] Hsuan-Kai Kao and Li Su. Temporally guided music-to-body-movement generation. *ACM MM*, 2020.

[20] Andrej Karpathy. https://github.com/karpathy/minGPT, 2020.

[21] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014.

[22] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *NeurIPS*, 2000.

[23] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. In *SIGGRAPH*. 2008.

[24] Alexis Lamouret and Michiel van de Panne. Motion synthesis by example. In *Comput. Animat. and Simulat.* 1996.

[25] Hsin-Ying Lee, Xiaodong Yang, Ming-Yu Liu, Ting-Chun Wang, Yu-Ding Lu, Ming-Hsuan Yang, and Jan Kautz. Dancing to music. In *NeurIPS*, 2019.

[26] Jehee Lee, Jinxiang Chai, Paul SA Reitsma, Jessica K Hodgins, and Nancy S Pollard. Interactive control of avatars animated with human motion data. In *Annual Conf. on Comput. Graph. and Interactive Tech.*, 2002.

[27] Minho Lee, Kyogu Lee, and Jaeheung Park. Music similarity-based approach to generating dance motion sequence. *Multimedia tools and applications*, 62(3):895–912, 2013.

[28] Buyu Li, Yongchi Zhao, and Lu Sheng. Danceformer: Music conditioned 3d dance generation with parametric motion transformer. In *AAAI*, 2022.

[29] Jiaman Li, Yihang Yin, Hang Chu, Yi Zhou, Tingwu Wang, Sanja Fidler, and Hao Li. Learning to generate diverse dance motions with transformer. *ArXiv*, abs/2008.08171, 2020.

[30] Ruilong Li, Shan Yang, D A Ross, and Angjoo Kanazawa. Ai choreographer: Music conditioned 3d dance generation with aist++. In *ICCV*, 2021.

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

[32] Meinard Müller, Tido Röder, and Michael Clausen. Efficient content-based retrieval of motion capture data. In *SIGGRAPH*, pages 677–685. 2005.

[33] Kensuke Onuma, Christos Faloutsos, and Jessica K Hodgins. Fmdistance: A fast and effective distance function for motion capture data. In *Eurographics*, pages 83–86, 2008.

[34] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. 2019.

[35] Xuanchi Ren, Haoran Li, Zijian Huang, and Qifeng Chen. Self-supervised dance video synthesis conditioned on music. In *ACM MM*, 2020.

[36] Guofei Sun, Yongkang Wong, Zhiyong Cheng, Mohan S Kankanhalli, Weidong Geng, and Xiangdong Li. Deepdance: music-to-dance motion choreography with adversarial learning. *TMM*, 23:497–509, 2020.

[37] Taoran Tang, Jia Jia, and Hanyang Mao. Dance with melody: An lstm-autoencoder approach to music-oriented dance synthesis. *ACM MM*, 2018.

[38] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *NeurIPS*, 2017.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, volume 30. Curran Associates, Inc., 2017.

[40] Nelson Yalta, Shinji Watanabe, Kazuhiro Nakadai, and Tetsuya Ogata. Weakly-supervised deep recurrent neural networks for basic dance step generation. In *IJCNN*, 2019.

[41] Sijie Yan, Zhizhong Li, Yuanjun Xiong, Huahan Yan, and Dahua Lin. Convolutional sequence generation for skeleton-based action synthesis. In *ICCV*, 2019.

[42] Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*, 2021.

[43] Zijie Ye, Haozhe Wu, Jia Jia, Yaohua Bu, Wenting Chen, Fanbo Meng, and Yanfeng Wang. Choreonet: Towards music to dance synthesis with choreographic action unit. *ACM MM*, 2020.

[44] Wenlin Zhuang, Congyi Wang, Si-Yu Xia, Jinxiang Chai, and Yangang Wang. Music2dance: Music-driven dance generation using wavenet. *ArXiv*, abs/2002.03761, 2020.