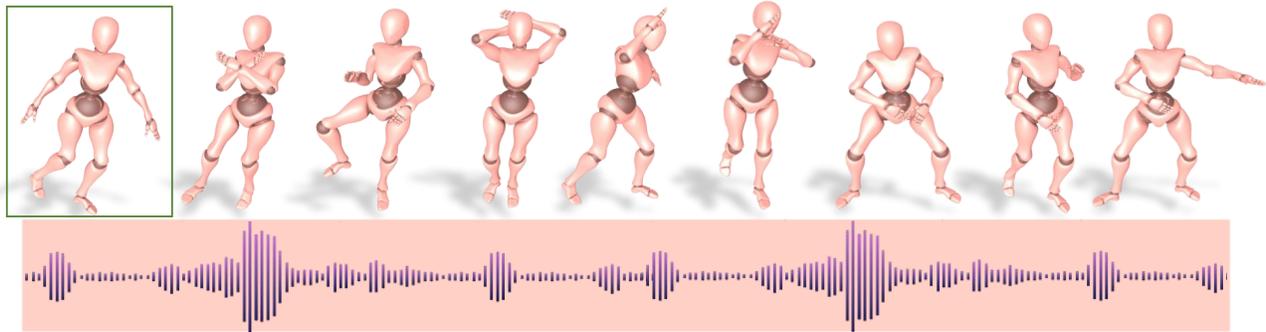


# Learning to Generate Diverse Dance Motions with Transformer

Jiaman Li<sup>1,2</sup> Yihang Yin<sup>3</sup> Hang Chu<sup>4,5</sup> Yi Zhou<sup>1</sup> Tingwu Wang<sup>4,5</sup> Sanja Fidler<sup>4,5</sup> and Hao Li<sup>6</sup>

<sup>1</sup>University of Southern California <sup>2</sup>USC Institute for Creative Technologies <sup>3</sup>Beihang University  
<sup>4</sup>University of Toronto <sup>5</sup>Vector Institute <sup>6</sup>Pinscreen



**Figure 1:** Given novel music (shown in the second row), our model generates diverse dance motions following beats (shown in the first row). The green box shows the initial pose for dance motion synthesis.

## Abstract

With the ongoing pandemic, virtual concerts and live events using digitized performances of musicians are getting traction on massive multiplayer online worlds. However, well choreographed dance movements are extremely complex to animate and would involve an expensive and tedious production process. In addition to the use of complex motion capture systems, it typically requires a collaborative effort between animators, dancers, and choreographers. We introduce a complete system for dance motion synthesis, which can generate complex and highly diverse dance sequences given an input music sequence. As motion capture data is limited for the range of dance motions and styles, we introduce a massive dance motion data set that is created from YouTube videos. We also present a novel two-stream motion transformer generative model, which can generate motion sequences with high flexibility. We also introduce new evaluation metrics for the quality of synthesized dance motions, and demonstrate that our system can outperform state-of-the-art methods. Our system provides high-quality animations suitable for large crowds for virtual concerts and can also be used as reference for professional animation pipelines. Most importantly, we show that vast online videos can be effective in training dance motion models.

## CCS Concepts

- **Computing methodologies** → Motion Generation;

## 1. Introduction

Due to the ongoing COVID-19 pandemic, an entire global live events industry is being shut down. Interactive Vtuber performances and virtual music concerts that take place in online gaming platforms are becoming increasingly popular. Examples include the widely popular holographic and VR concerts with Hatsune Miku, or the virtual rap event in Fortnite, performed by a digital avatar of Travis Scott, hosting tens of millions of viewers in real-time in a massive multiplayer online setting. The ability to generate complex and believable dance motions through music alone could impact a broad range of interactive and immersive applications in entertainment. In existing production settings, dance animations are typi-

cally generated through carefully choreographed performances and they are often captured using complex motion capture systems. Intensive manual labor is generally required for data clean up and animation refinement, which results in long and expensive production cycles.

More recently, data-driven motion synthesis methods [HSK16, LZX\*18] have been introduced to scale the production of animation without the need of actual performers. However, most of these techniques are based on a very limited diversity of motion, e.g. the popular CMU mocap dataset [LLC] in [LZX\*18] containing only two kinds of dances with a duration of less than one hour. These sequences are often very short and difficult to expand. Furthermore,

the movements are often monotonous with repetitive patterns and contain little variation and diversity. Therefore, models trained on these mocap data show limited generalization capabilities for realistic dance motions.

With recent advances in learning-based motion synthesis, several deep learning-based methods have been introduced [HSK16, LZX\*18]. These techniques formulate the problem as a long-term prediction task [LZX\*18] or as an audio-to-pose sequence translation problem [SDSKS18, AFP17, ACI\*17]. These regression-based methods fail to predict highly diverse and complex motions even using advanced training strategies via RNN models [LZX\*18] as they are deterministic. In particular, these models are designed to predict the successors given the current status instead of generating a novel motion according to a database distribution. These methods are not suitable for music-oriented dance motion synthesis which usually expects more than one possible motions given the same music. We further note that there is no benchmark or evaluation metric other than visual inspection for all the above methods.

In this work, we introduce a novel system that can synthesize diverse dance motions by learning from a large-scale dataset with a comprehensive set of highly diverse dance movements. To achieve this goal, we face three major challenges: (1) it is difficult to physically collect a large-scale dance motion dataset with sufficient diversity. Although motion/performance capture methods can provide high-precision data, they require dedicated devices, professional performers, and a tedious clean-up process; (2) existing regression-based models cannot handle the diversity of dance movements; (3) long-term temporal consistency and motion coherency have to be preserved in the generated sequence.

To address the above issues, (1) we take advantage that a large number of dance videos are available online. We download thousands of dance videos, and use cutting edge techniques for 2D pose detection, tracking, and 3D pose estimation to recover the dance sequence. Our large-scale dance motion dataset consists of 50 hours of synchronized music and dance pose sequences. (2) Using this dataset, we propose a conditional auto-regressive generative model to learn the motion distribution, along with Transformers [VSP\*17] as the main architecture for capturing extended time dependency. We formulate the output in each timestep as a categorical distribution using the discrete pose representations inspired by the success of discrete audio representation used in WaveNet architecture [ODZ\*16]. The discrete pose representation enables us to model the next step’s pose distribution and sample diverse poses at inference. Our model not only outperforms previous models for important evaluation metrics but also enables generating diverse dances with new music, demonstrating better modeling and generalization capabilities. (3) Besides, we propose several evaluation metrics from different perspectives to better judge whether the motion synthesis is satisfactory or not. We first use a Bullet-based [C\*13] virtual humanoid to evaluate the feasibility of generated pose sequences. Then, inspired by the motion-beat analysis approach in [KPS03], we introduce an effective and automatic metric for evaluating whether the dance movements follow the beat properly. In addition to physical plausibility and beat consistency, we also provide a metric for dance variation to measure the diversity in the synthesized results.

By testing our model using different settings and comparing it with two main baseline techniques, including acLSTM [LZX\*18] and ChorRNN [CFCF16] in a non-audio setting, we show that our real-time method can generate more diverse and realistic dance motions than existing techniques. We also show that when compared with LSTM architectures, our Transformer model is also more efficient to train. Finally, we demonstrate the effectiveness of our model and proposed evaluation metrics using a perceptual study. Our main contributions include:

1. An end-to-end real-time system for dance motion synthesis that uses highly complex and diverse motion data obtained from Internet videos. We also introduce an efficient and scalable data collection pipeline.
2. A novel two-stream motion transformer model with discrete pose representation to model the motion distribution and to capture long-term dependencies, which can be conditioned on music for diverse dance motion synthesis.
3. Several effective evaluation metrics to assess the quality of synthesized dance motions.

## 2. Related Work

Dance motion synthesis is a highly interdisciplinary problem and we review the most relevant work here.

### 2.1. Motion Synthesis

Motion synthesis has been an actively studied problem in both, the computer graphics and the computer vision communities. Typical methods rely on representation learning techniques such as auto-encoders, to embed motions into a low-dimensional space. Convolutional auto-encoders have been used to learn valid motion representation termed as the motion manifolds [HSKJ15], such that corrupted motion or missing-marker data can be easily recovered. High-level, user-friendly parameters for motion control have also been explored [HSK16]. In particular, they first specify character trajectory and foot contact information, then conditionally synthesize the human motion. A language-guided motion generation framework [LWC\*18] has been proposed to generate realistic motions from natural language descriptions.

Motion synthesis can be also viewed as a long-term human motion prediction. A number of works use recurrent neural networks to address this problem. In the work of Martinez et al. [MBR17], practical strategies including adding residual blocks and introducing sampling during training to improve RNN learning. Auto-conditioned RNN [LZX\*18] takes both ground truth and model prediction as input with a specific alternating interval to train the sequential model, showing the potential of generating motion sequences with long, or even unlimited future duration. QuaterNet [PGA18] conducts extensive experiments to demonstrate the effectiveness of quaternion representation. Dance generation can be regarded as a special case of motion synthesis, while dance movements are more complex compared to usual motions such as walking. It is also important to ensure the coordination between music and motions.

### 2.2. Choreography Learning

The creative process of choreography requires rich dancing experiences, which has inspired the development of data-driven ap-

proaches. A common approach of predicting dance movements from audio is to formulate the task as a translation problem. In particular, this approach enforces a one-to-one mapping from music to dance, which does not generalize well beyond training songs. GrooveNet [AFP17] has been the first work to exploit RNNs to model the correspondence between audio features and motion capture data. Based on music content analysis with motion connectivity constraints, a probabilistic framework [FG15] has been proposed. It mainly focuses on exploring and identifying the major contributing choreographic factors. Moreover, auto-encoder [ACI\*17] has been exploited for mapping music features to a latent space, and to generate dance pose sequences. Essentially, this formulation still falls into the audio to dance mapping category. Similar to previous work in choreography, Audio to Body Dynamics [SDSKS18] uses time-delayed RNNs to learn a mapping from audio to hand key points. The common drawbacks of these works lies in the difficulty of generating diverse pose sequences given audio information only. We argue that dance movements are more complex, hence less suitable to such deterministic formulation. This is because the same music can induce various kinds of dance motions, and the aesthetics of dance is strongly tied to the diversity of motions. Other work solve the task from a generative modeling perspective, ChorRNN [LKL18b], which introduces mixture-density RNNs to generate pose sequences and is particularly promising. Nevertheless, their approach is motion-only and does not take music information into account. In this paper, we treat dance motion synthesis as a conditional generative task. This allows us to not only promote the diversity of generation, but also take music as conditions to ensure the consistency with music features.

### 3. Methods

In this section, we present the Two-Stream Motion Transformer (TSMT) model. It processes the pose sequence and music context separately and then fuses the two streams together to predict the next motion.

#### 3.1. Problem Formulation

Inspired by the recent success of the auto-regressive model [ODZ\*16], we formulate our problem as an auto-regressive generative model conditioned on both music and past motions for synthesizing realistic and self-coherent dance motions. We denote the sequence of 3D dance motions as  $\mathbf{X}=\{\mathbf{x}_1, \dots, \mathbf{x}_T\}$ , and the sequence of audio features as  $\mathbf{A}=\{\alpha_1, \dots, \alpha_T\}$ .  $T$  is the number of frames in the sequence. For each time step,  $\mathbf{x}_t \in \mathbb{R}^{3n}$  and  $\alpha_t \in \mathbb{R}^m$ , where  $n$  is the number of body joints and  $m$  is the dimension of audio features. We model the joint conditional probability as

$$p(\mathbf{X}) = \prod_{t=1}^T p(\mathbf{x}_t | \alpha_t, \dots, \alpha_1, \mathbf{x}_{t-1}, \dots, \mathbf{x}_1) \quad (1)$$

where the motion stream and the audio stream are both encoded by neural networks.

#### 3.2. Motion and Audio Representation

##### 3.2.1. Motion

Unlike acLSTM [LZX\*18] and ChorRNN [LKL18b] which represent poses as deterministic coordinates or the distributions res-

spectively, we represent the continuous value of joint coordinates as discrete categories. For each dimension of the 3D pose  $\mathbf{x}_t$ , we perform uniform discretization into 300 constant intervals and obtain  $3n$  300-dimensional one-hot vectors. To reduce memory cost, we then transform each one-hot vector into a  $D^E$ -dimensional embedding vector with a shared learnable matrix of size  $[D^E, 300]$ . This converts the motion sequence into a tensor of size  $[T, 3n, D^E]$ . We merge the latter two axes of motion embedding and input it to a temporal-wise fully connected feed-forward layer to obtain a sequence of vectors with  $D^M$  channels. Following Vaswani et al. [VSP\*17], we also compute a  $D^M$ -dimensional positional embedding sequence with sine and cosine functions to encode the temporal information. We add the positional embedding sequence to the motion embedding sequence. This forms our final motion representation  $\mathbf{X}'$  of size  $[T, D^M]$ .

##### 3.2.2. Audio

For the audio data at each time step, we directly use the continuous 13-dimensional MFCC vector concatenated with its temporal derivatives into a 26-dimensional feature vector. We embed the binary one-hot beat signal into a 30-dimensional vector, resulting different embedding vectors at beat and non-beat positions. Similar to motion, we feed the audio representation into a 1D convolution and add to the positional embedding. The output is denoted as  $\mathbf{A}'$ .

#### 3.3. Two-Stream Motion Transformer (TSMT)

##### 3.3.1. Transformer

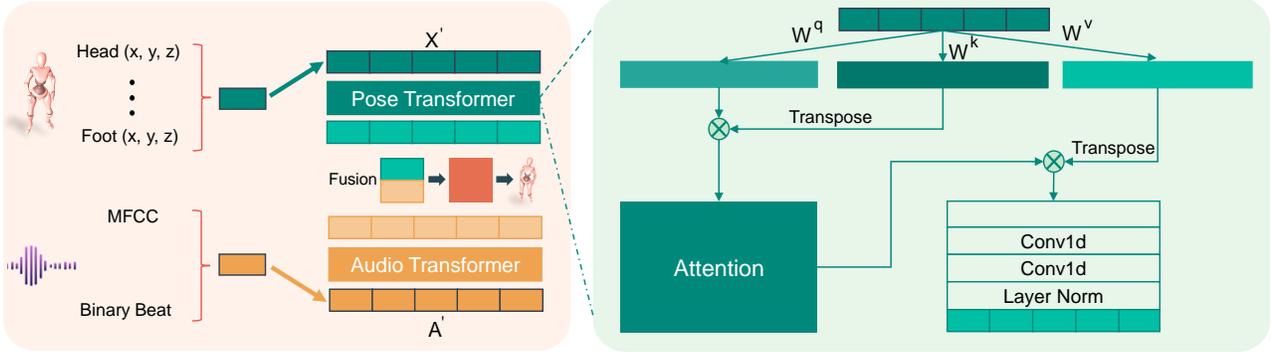
We adopt the Transformer [VSP\*17] architecture, harnessing and exploiting its power in modeling sequential data. Transformer consists of multiple blocks, each block is further composed of multiple heads of self-attention sub-layers as well as position-wise feed-forward sub-layers. Obtained from the previous step, our input  $\mathbf{X}'$  is a matrix of size  $[T, D^M]$ . We first transform the input sequence into three matrices, namely keys  $\mathbf{K}=\mathbf{X}'\mathbf{W}^K$ , queries  $\mathbf{Q}=\mathbf{X}'\mathbf{W}^Q$ , and values  $\mathbf{V}=\mathbf{X}'\mathbf{W}^V$ , where  $\mathbf{K}, \mathbf{Q}, \mathbf{V}$  are  $[T, D]$  matrices. We split each matrix into multiple heads where each head is  $[T, D_i]$ , with  $D=\sum_i D_i$ . For each head, we compute the scaled-dot attentional features as

$$\mathbf{Z}_i = \text{Softmax}\left(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{D_i}}\right) \mathbf{V}_i \quad (2)$$

We concatenate all the  $\mathbf{Z}_i$  to form  $\mathbf{Z}$  with size  $[T, D]$ , then feed it into two position-wise 1D convolution layers followed by layer normalization [VSP\*17]. This forms one Transformer block. We use multiple such blocks with residual connections between the blocks. We refer readers to [VSP\*17] for more details about the Transformer architecture.

##### 3.3.2. TSMT

The essence of dance is the manifestation of musicality in physical forms. Musicality takes form of multiple components such as vocal, bass, snare, keyboard, hi hat, drum, and other sound effects. A key element is carefully paying attention to different layers of music. Therefore, we use a pose-stream transformer to capture dance history, an audio-stream transformer to extract music context, and fuse these two streams together to predict the next pose as shown in Figure 2. We denote the output of pose transformer as  $\mathbf{Z}^X = \{\mathbf{z}_1^X, \dots, \mathbf{z}_T^X\}$ ,



**Figure 2:** Overview of our TSMT model. Left shows an example time step of pose and audio embedding. Middle shows our Two-Stream Motion Transformer including a pose transformer and an audio transformer. Each time step is followed by a late fusion module between two streams, which predicts the pose in the next time step. Right shows a detailed sub-layer composition inside a transformer block.

	$D^E$	$D_i$	$D^M$	block	layer	head
Pose	5	128	256	4	4	4
Audio	-	32	64	2	2	2

**Table 1:** Model details.  $D^E$  for embedding dimension.  $D_i$  for key, query, value dimension of each head, and  $D^M$  for overall feature dimension.

and the output of audio transformer as  $\mathbf{Z}^A = \{\mathbf{z}_1^A, \dots, \mathbf{z}_T^A\}$ . We compute the discrete representation of the pose at next time step from  $\mathbf{Z}^X$  and  $\mathbf{Z}^A$  and predict the final pose as,

$$\log p(\mathbf{x}_t) = \text{Softmax}(\mathbf{W}^X \mathbf{z}_{t-1}^X + \mathbf{W}^A \mathbf{z}_t^A) \quad (3)$$

During training, we can efficiently compute all the time steps in parallel, with a mask applied to the attention to ensure each step only attend to its past. During inference, we sample from the log-likelihood at the new time step.

### 3.4. Implementation Details

The model consists of multiple blocks with multiple layers, each containing the multi-head self-attention and position-wise feed-forward layers as described above. We list important model parameters in Table 1. To train the models, We use Adam optimizer with mini-batches of size 32. We set the initial learning rate as  $10^{-4}$  with 0.3 decay rate after 200 epochs. The global motion is inferred by the Global Path Predictor in Zhou et al. [ZLB\*20].

## 4. Dataset

We collected large amounts of high-quality videos from various dance channels on Youtube, and extracted 3D pose sequences with synchronized audios.

### 4.1. Dataset Collection

We started by manually selecting popular YouTube dance studio channels and downloading both videos and tags. Our next-step data processing pipeline could be divided into four stages: 1) Quick annotation of dancing segments from untrimmed YouTube videos; 2) 2D pose detection and pose tracking; 3) Simple manual cleaning of correct tracking results; 4) 3D pose estimation from 2D and post-processing. The first and third steps were introduced mainly because the Tubers edited the dance videos by inserting non-dance

contents. If the video sources only contain clean dance clips, our pipeline would be fully automatic without manual annotation.

### 4.2. Video Statistics and Trimming

We downloaded all the videos from five popular street dance channels and obtained 3809 videos in total. We filtered out the irrelevant contents e.g. dancer’s daily life and trimmed the dance segments from the original videos. For each video, we annotated the start and end time of dance performance as shown in Figure 3(a). The statistics of original and trimmed video segments are shown in Table 2 and Figure 3(c)(d).

### 4.3. 2D Pose Detection and Tracking

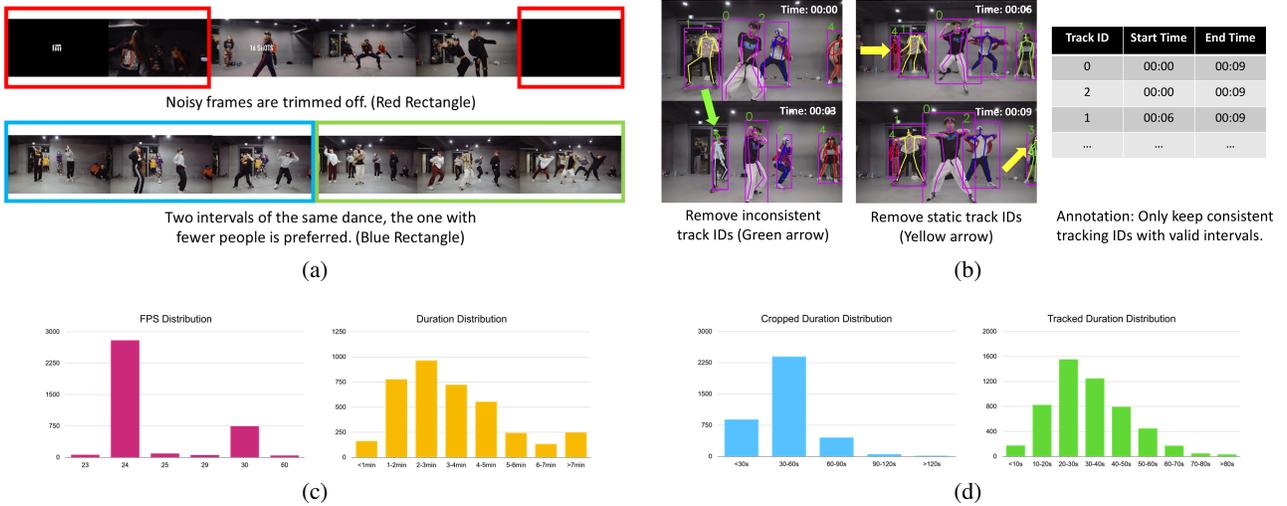
We use YOLO-V3 [RF18], SimplePose [XWW18] and LightTrack [NH19] to detect humans, estimate 2D pose, and track human pose sequences respectively. Since the videos often contain multiple dancers and audiences, we kept top-5 largest human detection bounding boxes to reduce computation cost.

### 4.4. Track Cleaning

Since our collected online videos are completely unconstrained, we observe two main issues: First, some tracks are audiences instead of dancers. Second, there are incorrect track id exchanges between dancers, which leads to the pose discontinuity. We performed manual annotations on pose sequences to reduce these types of noises. We ask volunteers to watch pose tracking visualization videos, and mark the correct start and end times for each track ids as shown in Figure 3(b). We notice that imperfect tracking usually happens during group formation changes, where dancers occasionally occlude each other. However, this does not affect the overall data quantity and distribution because most correct tracks have sufficient durations. Figure 3(d) shows the statistics after this data cleaning step.

### 4.5. 3D Pose Estimation and Jitter Removal

We applied VideoPose3d [PFGA19] to convert 2D pose sequences into 3D with 17 joints. Upon examining the results, we observed frequent motion jitters to be the main issue. Thus, we used Hodrick-PreScott (HP) filter [HP97] to remove these jitters. HP filter separates a time-series  $\mathbf{X} = \{x_t\}$  into a trend component and a cyclical



**Figure 3:** Our YouTube-Dance3D dataset. Figure (a) shows examples of video trimming. Figure (b) shows an example of valid pose tracking annotation. Figure (c) shows the distribution of original video FPS and duration. Figure (d) shows the duration distribution for cropped videos and tracked videos.

Video Source	Video	Min.	Trim Seg.	Trim Min.	Track IDs	Track Min.
Urban Dance Camp	197	400	193	173	9.1	201
Mat	302	1865	462	241	8,7	248
Movement Lifestyle	478	1454	217	144	11.9	185
Snowglobe	1184	3568	873	654	10.2	727
One Million Dance	1648	6301	2075	1494	14.5	1641
Total	3809	13588	3820	2707	12.4	3002

**Table 2:** Left shows statistics of our dataset: the number of videos, duration in minutes, trimmed segments, duration after trimming, average number of track IDs per video, and total duration of all tracks. Right shows a comparison of our dataset to previous datasets.

	Source	Min.	Dancer	Audio	3D	Public
ChorRNN [LKL18b]	MoCap	300	1	✗	✓	✗
GrooveNet [AFP17]	MoCap	24	1	✓	✓	✗
RobotKinect [ACI*17]	MoCap	-	4	✓	✓	✗
MelodyDance [TJM18]	MoCap	94	-	✓	✓	✓
MikuDance [YLX*19]	Game	600	-	✓	✓	✗
YT2D [LKL18a]	YouTube	376	-	✓	✗	✗
DanceToMusic [LYL*19]	YouTube	4260	-	✓	✗	✗
Ours	YouTube	3002	-	✓	✓	✓

component, i.e.  $x_t = x_t^{trd} + x_t^{cyc}$ . The components are determined by minimizing a quadratic loss function,

$$\min_{\{x_t^{trd}\}} \sum_t [x_t^{cyc}]^2 + \lambda [x_t^{trd} - 2x_{t-1}^{trd} + x_{t-2}^{trd}]^2 \quad (4)$$

We applied this filter to the 3D coordinates of each joint separately, with  $\lambda=1$  which empirically produces better result on our data. All poses are bicubic interpolated into 24-fps to ensure consistent framerates.

#### 4.6. Audio Processing

Mel-Frequency Cepstral Coefficients (MFCC) are effective audio features widely used in various audio related tasks [SDSKS18]. We use LibROSA [MRL\*15] to compute music features including MFCC and time intervals between beats. The audio are standardized with 44.1Khz sample rate with EBU R128 loudness normalization via ffmpeg-normalize, after which features are extracted at 24-fps.

#### 4.7. Dataset Analysis

Compared to existing dance datasets, our dataset has not only larger scale, but also higher quality and bigger variety. As given In Table 2, the scale of YouTube-Dance3D exceeds the previous

largest YT2D [LKL18a] by more than a magnitude order and comparable to a concurrent pose2D dataset introduced in [LYL\*19]. Moreover, our dataset possesses higher diversity due to the choice of urban dance, which by nature emphasizes choreography and variation. This is in contrast to other datasets that contain dances with obvious repetitive motion patterns e.g. Salsa, Tango, Ballet, and Waltz. Additionally, our data processing pipeline enables inexpensive data collection, making future expansion possible given new sources of dance video. We split data into approximately 80% training and 20% validation. For the convenience of model training, we divide data into segments of 20s, with 10s overlap between two consecutive segments. Each segment contains 480 frames. This results in 9561 training segments and 2136 validation segments.

## 5. Experiments

We first describe evaluation metrics including physical plausibility, beat consistency and generation diversity. Next, we show quantitative and qualitative results.

### 5.1. Metrics

Automatic evaluation metric has been a known challenge in many generative tasks. Recently [YLX\*19, LYL\*19], motion generation evaluation receives more attention, where FID-based metrics have

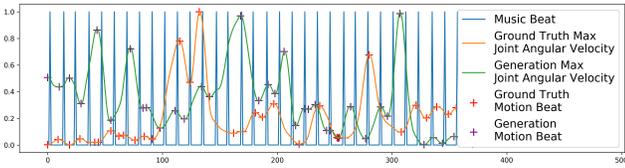


Figure 4: An example of detected music beats and motion beats.

been explored. We share the similar insight on the aspect of evaluating generation diversity. Moreover, we further introduce to use a humanoid physics simulator to evaluate dance plausibility and a new beat consistency metric.

### 5.1.1. Physical Plausibility

We measure the ratio of implausible frames that can not be executed by the humanoid inside Bullet simulator. Concretely, we measure two types of pose invalidity: 1) *Authenticity* is the ratio of frames where none of the joints exceeded its rotation limit. This ensures the pose is statically plausible, as can be performed by a normal human body. 2) *Coherence* is the ratio of frames where the angular velocity of all joints stay within a realistic range. This ensures the motion between poses is dynamically plausible, preventing abnormal behaviour such as twitching.

### 5.1.2. Beat Consistency

A good dancer knows to express their perception of beat by periodically changing their moves, in other words, to accompany musical beats with their motion beats. We first extract motion beats from poses. Then we measure the similarities between generated motion beats and ground-truth motion beats. We extract motion beats using the method of Kim et al. [KPS03], which detects zero-crossings of the joint angular acceleration. For two beats to match, we allow a flexibility of 2 frames. We compute *precision*, *recall*, and *F-score* between two beat sequences.

### 5.1.3. Diversity

The complexity of choreography reflects in the composition of diverse body movements. We measure the diversity aspect of generated dances via four aspects. 1) *Fr chet Inception Distance (FID)* This refers to the default usage of FID [HRU\*17], measuring the difference between ground truth and generation feature distribution. 2) *Inter-sequence Diversity (A-seq-D)* We generate a large number of pose sequences, from which pairs of sequences are randomly selected. For each pair, we measure the L2 distance between their feature vectors. We use the average feature distance as the A-seq-D score. 3) *Intra-sequence Diversity (I-seq-D)* Within a pose sequence, we divide it into chunks, and compute the feature distance among all possible pairs. This distance is averaged over all pairs from all sequences as the I-seq-D score. 4) *Same-music Diversity (S-music-D)* We generate multiple sequences given the same music, and compute the feature distances between these generations. We average this over all music as the S-music-D score.

We obtain perceptual features of the dance with a dance style classifier. We first divide our dataset into 5 categories based on the YouTube channel name, with balanced size between classes. Then we train a 2-block transformer with classification outputs, obtaining

	Coherence $\uparrow$	Authenticity $\uparrow$	FID $\downarrow$	A-seq-D $\uparrow$	I-seq-D $\uparrow$
Ground-Truth	1	1	0	32.84	10.76
acLSTM	<b>0.9995</b>	<b>0.9998</b>	3.94	12.99	3.55
ChorRNN-5	0.83	0.75	2.56	30.48	<b>21.26</b>
DLSTM	0.94	0.88	2.47	29.07	12.06
TSMT-noaudio	<b>0.97</b>	<b>0.96</b>	<b>0.53</b>	<b>32.52</b>	7.98

Table 3: Comparing different methods at the non-audio setting.

61.0% top-1 classification accuracy and 71.5% top-2 classification accuracy.

## 5.2. Comparisons

Previous work mainly focus on motion synthesis without audio input. Therefore, we first compare our model with them in non-audio setting. Then we compare variations of audio-enabled models.

### 5.2.1. acLSTM [LZX\*18]

acLSTM [LZX\*18] is a widely-used model in dance motion synthesis from mocap data. It introduces an interval when ground-truth and model samples are used in the training process. This address the motion freeze issue of standard teacher-forcing training, which enables generating unlimited sequence length by learning from small amount of data. The model is a 3-layer LSTM with a hidden dimension of 1024. We follow the same settings as described in [LZX\*18] to train on our dataset. acLSTM is essentially a deterministic model, we evaluate its generative ability by feeding it additional information of different initial pose sequences with a length of 10.

### 5.2.2. ChorRNN [CFCF16]

ChorRNN [CFCF16] is a mixture density model based on a 3-layer LSTM. In each time step, ChorRNN predicts a distribution of pose instead of deterministic coordinates. Since their code is not public, we re-implement their model and experiment with different number of mixtures.

## 5.3. Results

### 5.3.1. Quantitative Results

We first report experimental results with the non-audio setting in Table 3. We generate for each model 1000 pose sequences for evaluation. All model use a random initial first pose to reduce the possible noise brought by generation in the first step. It can be seen that for acLSTM [LZX\*18], although it has the highest Coherence and Authenticity scores, it is unable to generate diverse dances as shown in the FID and Diversity scores. For ChorRNN [LKL18b], although it has the highest intra-sequence diversity, many of its generations are hardly valid human poses. This can be seen from its low Coherence and Authenticity scores. We further experiment different number of mixtures which consists of its most important parameters. From the results in Table 4, the high FID scores show that none of the settings can generate both realistic and diverse dances. Regarding Coherence and Authenticity, our model scores comparably to acLSTM, while being able to generate more diverse motions close to the ground truth distribution. In the audio-enabled setting, We show baseline and ablation study results in Table 5. Single Transformer model refers to the baseline model that directly

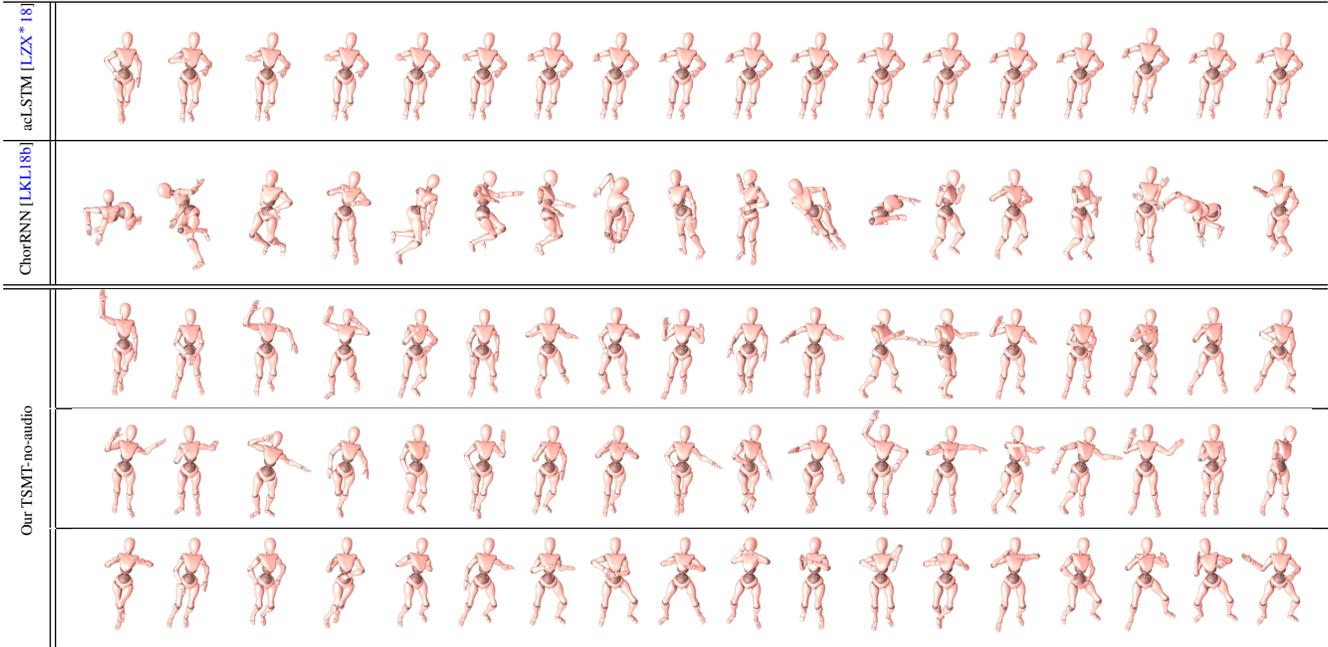


Figure 5: Qualitative comparison. Our model generates plausible, realistic, and diverse dances.

concatenates audio and pose data as input to a single-stream transformer. We also explore the effects of using different combinations of audio information for each model. We observe that all of our models have high Coherence and Authenticity scores. The beat-only models achieve higher diversity scores. This is because of less constraints imposed by the audio input.

5.3.2. Qualitative Results

We show qualitative results for acLSTM, ChorRNN and our proposed TSMT-noaudio Model in Figure 5. All results are from a generation of 20 seconds at the non-audio setting, screen captured at the same time interval. It can be seen that acLSTM tends to quickly freeze to a generic static pose. ChorRNN generates invalid poses that can hardly be performed by any dancer. Our model is able to generate valid and diverse poses, which is consistent with the quantitative evaluation metric scores as reported in Table 3.

5.3.3. Computation Time

We test the computation time of our transformer model and baseline models in the non-audio setting. We use a single GPU GTX 1080 to perform the running time evaluation. For training time, we report the average time per batch from our training log. For testing time, we generate 10 sequences with each model, each with 480 time steps. Results are shown in Table 6. It can be seen that our transformer-based model outperforms LSTM-based model in terms of training efficiency, while remaining real-time (24-fps) in testing.

5.4. Human Evaluation

We conduct human evaluation to verify whether our proposed automatic metrics are consistent with human judgement. We make use of Amazon Mechanical Turk (AMT) and pay workers to perform a crowd-sourced evaluation. We restrict to US-based workers who have at least 85% acceptance score.

	Coherence↑	Authenticity↑	FID↓	A-seq-D↑	I-seq-D↑
ChorRNN-1	<b>0.94</b>	<b>0.94</b>	8.42	<b>40.39</b>	17.32
ChorRNN-5	0.83	0.75	<b>2.56</b>	30.48	<b>21.26</b>
ChorRNN-10	0.75	0.54	17.30	28.35	17.99
ChorRNN-20	0.74	0.55	17.29	24.88	16.66

Table 4: Effect of mixture component number for ChorRNN [LKL18b]

5.4.1. Physical Plausibility and Beat

We first obtain Authenticity, Coherence, and Beat scores computed with our automatic metrics. Then for each metric, we divide the score into three levels of high, middle and low. We randomly sample 60 pairs from three different level combinations, namely high-mid, high-low, and mid-low. Then we resort to workers and ask them to select the better one from each pair. Each test contains 20 evaluation pairs and 3 validation pairs. The validation pairs contain a ground-truth and an artificially noised motion sequence. We use this as a hidden test to filter out the workers who are inattentive or intentionally abusing. We take the answer from AMT workers via majority-voting, compare with known answer decided by actual scores in each pair and calculate the consistency between two answers. We observe that Authenticity and Coherence have high consistency with human evaluations which verify the effectiveness of our physical plausibility metrics. However, the consistency for beats is relatively low for High-Mid and Mid-Low tests, while High-Low test has higher consistency. We believe that the average person does not have high sensitivity for beats as machine measurements do. In other words, some of AMT workers might be incapable of distinguishing whether the dance follows the beat or not when the differences are small. Further studies are needed in the future for deeper beat analysis.

	Coherence $\uparrow$	Authenticity $\uparrow$	Beat $\uparrow$	FID $\downarrow$	A-seq-D $\uparrow$	I-seq-D $\uparrow$	S-music-D $\uparrow$
Ground-Truth	1	1	1	0	32.84	10.76	0
Single Stream Transformer + Beat	0.97	0.92	0.447	0.56	<b>41.95</b>	<b>11.09</b>	17.30
Single Stream Transformer + MFCC	0.96	0.92	0.445	0.43	38.69	9.69	16.96
Single Stream Transformer + Beat + MFCC	0.97	0.93	0.439	1.27	38.86	10.88	16.00
Two Stream Transformer + Beat	0.97	0.93	<b>0.451</b>	1.43	40.90	9.49	<b>18.11</b>
Two Stream Transformer + MFCC	0.96	0.92	0.430	1.46	33.37	9.41	16.07
Two Stream Transformer + Beat + MFCC	0.97	0.93	0.449	<b>0.21</b>	36.44	10.02	16.16

Table 5: Comparing different methods at the audio-enabled settings.

	acLSTM	ChorRNN	DLSTM	Our TSMT
Train per Batch	2.40s	0.31s	0.39s	<b>0.23s</b>
Test per Step	<b>0.55ms</b>	62.5ms	15.7ms	19.4ms

Table 6: Training and testing time comparison.

	High-Low	High-Mid	Mid-Low	Total
Authenticity	0.9	1.0	0.65	0.85
Coherence	0.95	0.95	0.9	0.933
Beat	0.65	0.55	0.4	0.533

Table 7: AMT user study result to evaluate the consistency between automatic metrics and human judgement.

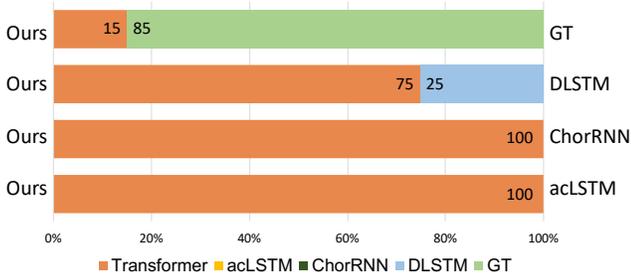


Figure 6: AMT user study on overall generation quality.

### 5.4.2. Overall Quality

We randomly select 200 pairs generated by acLSTM, ChorRNN, Discrete-LSTM, and our TSMT model (non-audio setting) in different combinations, and ask workers to pick the more preferable one with better quality. A comparison between our result and ground-truth is also included. Similarly, we use validation questions to filter out noisy annotations. Figure 6 shows the pair-wise comparison results. It can be seen that our model is obviously better than acLSTM and ChorRNN baselines, also superior than LSTM with discrete representation. However, there is still a gap between our synthetic motions and the ground truth, and the main reason is that our synthetic motion sequences are based on sampling in each timestep, there is a chance that low probability pose gets sampled which introduce noise to the sequence generation. It is possible to eliminate this type of noise by applying some constraints during pose sampling.

## 6. Discussion and Conclusion

We proposed a complete system for dance motion synthesis from audio input, and we have shown that we can handle highly diverse dance movements using widely available online dance videos as training. We have also introduced a new large-scale motion dataset,

as well as new evaluation metrics in terms of quality, diversity and musicality. Our proposed conditional generative model also outperformed existing methods. We also conducted a study that indicates the effectiveness of our model and proposed metrics.

### 6.1. Limitation and Future Work

Since our data is collected from videos, the number of joints depends on 3D pose estimation method which usually does not take finger animations into account. An interesting future direction is to extract finger joints and facial expressions as well from videos so that we are able to produce expressive dance data. Furthermore, we are interested in using motion capture data to denoise and improve the quality of 3D pose sequence extracted from online videos.

We have used audio representations such as MFCC features and beat, but according to professional dancers and choreographers, they tend to also follow additional musical layers including bass, lyrics, etc. The exploration for more complex audio features is therefore particularly intriguing, as well as the analysis of interactions between dancers and crowds.

## 7. Acknowledgements

This research was funded by in part by the ONR YIP grant N00014-17-S-FO14, the CONIX Research Center, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, the Andrew and Erna Viterbi Early Career Chair, the U.S. Army Research Laboratory (ARL) under contract number W911NF-14-D-0005, Adobe, and Sony. We thank Yajie Zhao, Mingming He for proof reading, Zhengfei Kuang for editing the demo, Marcel Ramos and Kalle Bladin for the character rigging and rendering, and Pengda Xiang for the data annotation.

## References

- [ACI\*17] AUGELLO A., CIPOLLA E., INFANTINO I., MANFRE A., PILLATO G., VELLA F.: Creative robot dance with variational encoder. *arXiv:1707.01489* (2017). 2, 3, 5
- [AFP17] ALEMI O., FRANÇOISE J., PASQUIER P.: Groovenet: Real-time music-driven dance movement generation using artificial neural networks. *networks* 8, 17 (2017), 26. 2, 3, 5
- [C\*13] COUMANS E., ET AL.: Bullet physics library. *Open source: bulletphysics.org* 15, 49 (2013), 5. 2
- [CFCF16] CRNKOVIC-FRIIS L., CRNKOVIC-FRIIS L.: Generative choreography using deep learning. *arXiv:1605.06921* (2016). 2, 6
- [FG15] FUKAYAMA S., GOTO M.: Music content driven automated choreography with beat-wise motion connectivity constraints. *Proceedings of SMC* (2015), 177–183. 3

- [HP97] HODRICK R. J., PRESCOTT E. C.: Postwar us business cycles: an empirical investigation. *Journal of Money, credit, and Banking* (1997), 1–16. 4
- [HRU\*17] HEUSEL M., RAMSAUER H., UNTERTHINER T., NESSLER B., HOCHREITER S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS* (2017). 6
- [HSK16] HOLDEN D., SAITO J., KOMURA T.: A deep learning framework for character motion synthesis and editing. *TOG* 35, 4 (2016), 138. 1, 2
- [HSKJ15] HOLDEN D., SAITO J., KOMURA T., JOYCE T.: Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia* (2015), p. 18. 2
- [KPS03] KIM T.-H., PARK S. I., SHIN S. Y.: Rhythmic-motion synthesis based on motion-beat analysis. In *TOG* (2003), vol. 22, pp. 392–401. 2, 6
- [LKL18a] LEE J., KIM S., LEE K.: Listen to dance: Music-driven choreography generation using autoregressive encoder-decoder network, 2018. [arXiv:arXiv:1811.00818](https://arxiv.org/abs/1811.00818). 5
- [LKL18b] LEE J., KIM S., LEE K.: Listen to dance:music-driven choreography generation using autoregressive encoder-decoder network. [arXiv:1811.00818](https://arxiv.org/abs/1811.00818) (2018). 3, 5, 6, 7
- [LLC] LLC. M.: Cmu graphics lab motion capture database. <http://mocap.cs.cmu.edu>. 1
- [LWC\*18] LIN A. S., WU L., CORONA R., TAI K., HUANG Q., MOONEY R. J.: Generating animated videos of human activities from natural language descriptions. *Learning 2018* (2018). 2
- [LYL\*19] LEE H.-Y., YANG X., LIU M.-Y., WANG T.-C., LU Y.-D., YANG M.-H., KAUTZ J.: Dancing to music. In *NeurIPS* (2019). 5
- [LZX\*18] LI Z., ZHOU Y., XIAO S., HE C., HUANG Z., LI H.: Auto-conditioned recurrent networks for extended complex human motion synthesis. *ICLR* (2018). 1, 2, 3, 6, 7
- [MBR17] MARTINEZ J., BLACK M. J., ROMERO J.: On human motion prediction using recurrent neural networks. In *CVPR* (2017). 2
- [MRL\*15] MCFEE B., RAFFEL C., LIANG D., ELLIS D. P., MCVICAR M., BATTENBERG E., NIETO O.: librosa: Audio and music signal analysis in python. 5
- [NH19] NING G., HUANG H.: Lighttrack: A generic framework for online top-down human pose tracking. [arXiv:1905.02822](https://arxiv.org/abs/1905.02822) (2019). 4
- [ODZ\*16] OORD A. V. D., DIELEMAN S., ZEN H., SIMONYAN K., VINYALS O., GRAVES A., KALCHBRENNER N., SENIOR A., KAVUKCUOGLU K.: Wavenet: A generative model for raw audio. [arXiv:1609.03499](https://arxiv.org/abs/1609.03499) (2016). 2, 3
- [PFGA19] PAVLLO D., FEICHTENHOFER C., GRANGIER D., AULI M.: 3d human pose estimation in video with temporal convolutions and semi-supervised training. *CVPR* (2019). 4
- [PGA18] PAVLLO D., GRANGIER D., AULI M.: Quaternet: A quaternion-based recurrent model for human motion. In *BMVC* (2018). 2
- [RF18] REDMON J., FARHADI A.: Yolov3: An incremental improvement. [arXiv:1804.02767](https://arxiv.org/abs/1804.02767) (2018). 4
- [SDSKS18] SHLIZERMAN E., DERY L., SCHOEN H., KEMELMACHER-SHLIZERMAN I.: Audio to body dynamics. *CVPR* (2018). 2, 3, 5
- [TJM18] TANG T., JIA J., MAO H.: Dance with melody: An lstm-autoencoder approach to music-oriented dance synthesis. In *ACM Multimedia* (2018), pp. 1598–1606. 5
- [VSP\*17] VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER Ł., POLOSUKHIN I.: Attention is all you need. In *NIPS* (2017). 2, 3
- [XWW18] XIAO B., WU H., WEI Y.: Simple baselines for human pose estimation and tracking. In *ECCV* (2018). 4
- [YLX\*19] YAN S., LI Z., XIONG Y., YAN H., LIN D.: Convolutional sequence generation for skeleton-based action synthesis. In *ICCV* (2019). 5
- [ZLB\*20] ZHOU Y., LU J., BARNES C., YANG J., XIANG S., ET AL.: Generative tweening: Long-term inbetweening of 3d human motions. [arXiv preprint arXiv:2005.08891](https://arxiv.org/abs/2005.08891) (2020). 4