Computational Visual Media
https://doi.org/10.1007/s41095-0xx-xxxx-x

Research Article

Foot-constrained Spatial-Temporal Transformer for Keyframe-based Complex Motion Synthesis

Hao Li[1,2], Ju Dai(✉)[1], Rui Zeng[1,2], Junxuan Bai[3], Zhangmeng Chen[1,2], Junjun Pan(✉)[1,2]

Abstract Keyframe-based motion synthesis hold significant effects in games and movies. Existing methods for complex motion synthesis often produce foot sliding, which results in low quality movements. In this paper, we analyze the cause of the sliding issue attributed to the mismatch between root trajectory and motion postures. To address the problem, we propose a novel spatial-temporal transformer network conditioned on foot contact information for keyframe-based motion synthesis. Specifically, our model mainly compromises a spatial-temporal transformer encoder and two decoders for learning motion sequence features and predicting motion postures and foot contact states. A novel mixed embedding, which consists of keyframes and foot contact constraints, is incorporated into the model to facilitate network learning from diversified control knowledge. To generate matched root trajectory with motion postures, we design a differentiable root trajectory reconstruction algorithm to construct root trajectory based on the decoder outputs. Qualitative and quantitative experiments on the public LaFAN1, Dance, and Martial Arts datasets demonstrate the superiority of our method in generating complex motion synthesis. It can satisfactorily address the foot sliding problem compared with the existing most advanced methods.

Keywords Motion transition, Transformer, Foot sliding, Foot constraint, Mix embedding

1 Introduction

Character animation is one of the important research topics in the computer graphics [1, 2]. Animation based on keyframe interpolation dramatically alleviates the laborious workloads of animators. Although deep learning-based motion synthesis [3, 4] has apparent advantages over the traditional linear interpolation [5], it still cannot address well with the sliding problem during the complex motion generation. Tedious post-processing is needed for satisfactory animation production [6, 7]. Therefore, it is of great research value for the end-to-end high quality generation of complex motions.

Previous endevors [3, 8] often separately decode the root trajectory and motion postures (joint rotation angles) from the hidden motion representations. Such a strategy disregards the tight dependence of root trajectories on the changes in motion movements. As a result, the generated motions often lead to mismatches between root trajectories and motion postures, which cause the foot sliding problem. Therefore, post-processing strategies based on physical constraint-based pose correction [9, 10] and modification of root joint translations [11, 12] are used to eliminate sliding. Essentially, post-processing is a secondary editing of the motion to obtain a matched consistency between the root trajectory and motion postures [13]. Although it can produce visually natural motion movements, it is challenging to entangle in the end-to-end optimized neural networks, and the process is tedious. Inspired by the post-processing algorithms [10–12], we believe that the solution for sliding lies in guaranteeing the matching between the root trajectory and motion postures. We do not attempt to forecast the root trajectory directly. On the contrary, the root trajectory is inferred based on the predicted motion postures and foot contact information to ensure matching consistency.

In this paper, to generate robust complex motions and solve the foot sliding problem without post-processing, a novel spatial-temporal transformer network for keyframe-based complex motion synthesis conditioned on foot contact information is proposed. Our model compromises a spatial-temporal transformer encoder and two decoders for learning motion sequence features and predicting motion postures and foot contact states. Besides encoding motion sequence

1 Peng Cheng Laboratory, Shenzhen, China. E-mail: daij@pcl.ac.cn (Ju Dai)

2 Beihang University, Beijing, China. E-mail: haolirj@buaa.edu.cn (Hao Li), zrsammy@buaa.edu.cn (Rui Zeng), zhmchen@buaa.edu.cn (Zhangmeng Chen), pan_junjun@buaa.edu.cn (Junjun Pan).

3 Capital University of Physical Education and Sports, Beijing, China. E-mail: baijunxuan@cupes.edu.cn (Junxuan Bai).

information, a novel mixed embedding, which consists of keyframe and foot contact constraints, is incorporated to facilitate network learning from diversified control knowledge. Furthermore, unlike most previous endeavors separately predicting the information of root trajectory and motion postures, we present a differentiable root trajectory reconstruction algorithm to infer matched root trajectory based on the predicted motion postures and foot contact states. Qualitative and quantitative experiments on the public LaFAN1, Dance, and Martial Arts datasets demonstrate the superiority of our method in generating robust motion movements. It can adequately handle the foot sliding issue compared with state-of-the-art methods.

Our main contributions are summarized as follows:

- We analyze the causes of foot sliding in motion generation and design a root trajectory reconstruction algorithm that can be embedded and optimized in our network to eliminate sliding issues.
- We propose a novel spatial-temporal transformer network for keyframe-based character animation, which is conditioned on foot contact states to perform high-quality and diversified motion synthesis.
- We achieve better performance over state-of-the-art methods in experiments on public datasets, demonstrating the superiority of our model for generating both simple and complex motions.

2 Related Work

2.1 Motion completion

Motion completion becomes a prevailing hotspot in computer graphics and multimedia [14, 15], which can be regarded as generating motion transition based on keyframes. As a typical 3D animation [16, 17], motion completion needs to consider the motion postures of all frames and the position information of keyframes. In the early stage, linear interpolation [18] between keyframes is usually used to generate motion inbetweening. In addition, probability-based models such as Gaussian process [19] and Markov model [20] are introduced into motion completion tasks. When dealing with complex data, however, these methods have difficulties producing high-quality motions.

As a powerful learning technique, recent years have witnessed significant growth of deep learning in motion completion [21, 22]. For example, Harvey and Pal [8] present the Recurrent Transition Networks (RTN) for transition generation, where each autoregressive inputs contain the position of keyframes and their offsets to generate transition movements. Later, Harvey *et al.* [3] extend the RTN model based on adversarial learning and design the novel time-to-arrival embedding and scheduled target noise vector into the network to achieve robust transition generation. Duan *et al.* [4] formulate a non-autoregressive Transformer-based network as the backbone and leverage the mixed embedding of keyframe information and temporal relationships to reconstruct motion movements between keyframes. Kaufmann *et al.*[23] propose an end-to-end trainable convolutional autoencoder to fill the unknown frames. Still, the method is challenging to ensure that the bone length is consistent and inevitable foot sliding.

Different from previous methods, we synthesize motion conditioned on foot contact constraints and embed a differentiated root trajectory reconstruction algorithm into our model to circumvent foot sliding in motion generation.

2.2 Motion sliding correction

Human motion has intrinsically highly intricate and arbitrary [24]. The traditional interpolation algorithm is challenging to benefit the completion of complex actions [25]. The most obvious effect is the inevitable foot-sliding in animation production. At present, the most straightforward procedure is to post-process the generated animation. For example, Kovar *et al.* [13] leverage inverse kinematics (IK) modification to construct bone length and rotation angle information, so as to make the generated animation conform to the foot constraints. Nevertheless, such strategy requires re-correction for each frame, which greatly increases the animator's workloads.

Currently, endeavors based on deep learning methods leverage complex and powerful neural networks to represent human motions [26], which still needs to eliminate the sliding issue. To obtain high-quality motion movements, Harvey *et al.* [3] attempt to label and predict the contact foot in the training phase, hoping to diminish the sliding phenomenon. Lee *et al.* [27] introduce a contact loss function to penalize foot sliding in training to deal with foot contact of non-periodic motion. Pan *et al.* [7] supervise the speed of footsteps in training to constrain the network to produce more realistic motion. Zou *et al.* [28] propose a neural network-based detector for localizing ground contact events on human feet by imposing physical constraints to optimize the dynamics of the entire motion. Although these methods can inhibit the occurrence of sliding, they can not completely eradicate its occurrence. Another representative method, Wang *et al.* [29] resort to the adversarial insights of the Generative Adversarial Network (GAN) to improve generated motion quality in the generator. However, it has only verified the effectiveness of simple periodic movements, unable to handle challenging dance motions and other complex activities.

In summary, existing methods independently predict the root trajectory and motion posture information, neglecting their immediate relevance. Thus the mismatch between root trajectory and motion posture results in the foot sliding problem. To address the issue, our model focuses on reconstructing motion postures and foot contact states based on a matched root trajectory inferred with our proposed root trajectory reconstruction algorithm.

2.3 Motion control

Motion control is a typical conditional motion generation task, which is usually driven by user-defined time-intensive external signals [30, 31]. Motion graph is the most classic control method in traditional animation generation control [32–34]. It transforms the motion dataset into a graph structure, finds the path to the desired state in the expected time through random search in the motion graph, and interactively edits the motion to generate controllable complex movements. Cai *et al.* [35] convert motion control into a maximum posterior probability problem, synthesizing motion based on a statistical dynamic model through prior knowledge and control constraints. Other statistical models such as Gaussian Processes [36] and Gaussian Process Latent Variable Models [37] are also utilized for motion control.These methods have expensive runtime calculations and the generated actions are very scripted.

At present, control based on deep learning has gradually dominated the area. Convolutional autoencoder [38] and recurrent neural networks [39] are often used in this problem. Daniel *et al.* [40] generate the regression of the current frame control parameters to the corresponding character state through the phase function based on the character state. Pan *et al.* [7] introduce the root trajectory control into the RTN generation model to achieve motion completion based on root trajectory control. Similarly, the work based on reinforcement learning is further incorporated into the physical rules to generate higher-quality sports. Kevin *et al.* [41] conduct the stable control of characters in walking and other movements through the control strategy of reinforcement learning.

Nevertheless, the above control methods focus on modeling the relationship between the character's state and control signals, which rarely consider the sliding in the generated motion. On the contrary, we focus on enhancing motion quality and avoiding foot-sliding to attain end-to-end high-quality motion generation. Therefore, We introduce foot contact states as the control signal to complete the motion completion by giving the foot contact state between keyframes.

3 Methodology

3.1 Overview of the proposed network

The core of our work is to generate expected stable and robust motion transitions given user-defined keyframes and foot contact states. The overview of the proposed network is illustrated in Figure 1. Our model mainly consists of an Spatio-Temporal Transformer (ST-Transformer) encoder and two decoders. The ST-Transformer involves a spatial transformer to model the skeleton joint relationships and a temporal transformer encoder to establish the temporal structure dependencies. The decoders comprise a contact decoder and the state decoder, which are respectively responsible for decoding the foot contact information and reconstructing the unknown motion postures. For the root trajectory, we design the Root Trajectory Reconstruction (RTR) block to reason it based on foot contact state and motion posture knowledge, which can competently bypass the foot-sliding issue, especially in complex motion synthesis.

Given keyframes, we first compliment the unknown frames between keyframes using the spherical linear interpolation algorithm [25]. The interpolated motion sequence is later fed into a linear embedding to convert into feature space. We leverage the spatial and temporal transformer blocks to process the embedded representation sequentially. Since the transformer has permutation invariant characteristics, we incorporate Spatial Position Embedding (S-PE) and Temporal Position Embedding (T-PE) into the spatial and temporal transformers to distinguish different spatial joints and temporal frames. We also introduce a novel Mix-Embedding to signify keyframe information and control conditions to enhance the expression ability of the network.

3.2 Mix embedding

For motion sequences obtained by spherical linear interpolation based on keyframes, the differences between interpolated motion frames and ground truth (GT) frames are usually slight when near keyframes and larger when far away from keyframes. Therefore, our network aims to predict the residuals between interpolated frames and GT values to complete transition movements. To enable our model to distinguish between keyframes and intermediate frames, we introduce keyframe embedding to accomplish motion completion for intermediate frames better. Similarly, to indicate the character's foot contact state and make the generated motion conform to user-defined contact states, we also encode the foot contact state for each frame.

To this end, we design a mix embedding $\mathbf{E}_m$. It consists of keyframe embedding $\mathbf{E}_{kf}$ and foot contact embedding $\mathbf{E}_{fc}$,

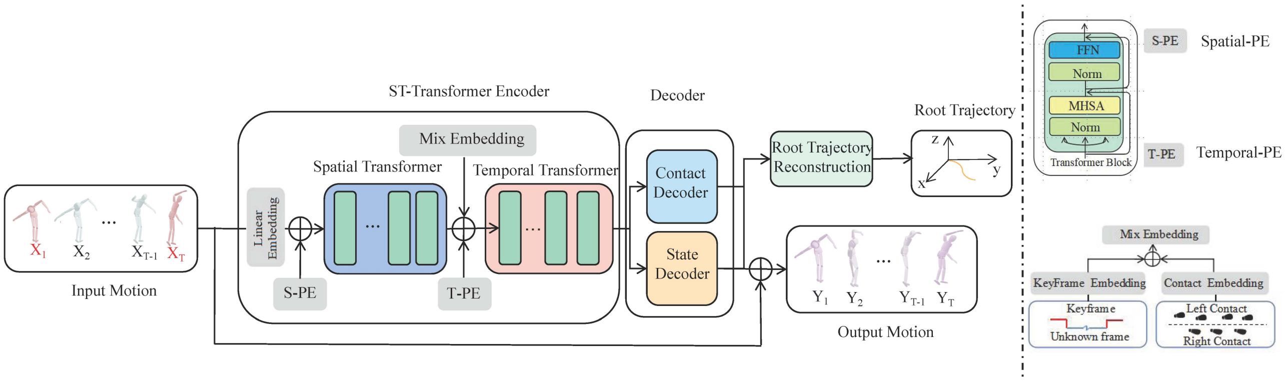


Fig. 1 Overview of the proposed framework. We use ST-Transformer to extract the spatial-temporal features and decode the foot contact state and motion posture of the input motion. The root trajectory is inferred from the motion posture and foot contact state.

where $\mathbf{E}_{kf}$ characterizes keyframe information of the motion sequence and $\mathbf{E}_{fc}$ determines the foot contact states for all frames. $\mathbf{E}_m$ is defined as follows:

$$\mathbf{E}_m = \mathbf{E}_{kf} + \mathbf{E}_{fc}, \tag{1}$$

where $\mathbf{E}_{kf} \in \mathbb{R}^{T\times d}, \mathbf{E}_{kf} = \hat{\mathbf{S}}_k \mathbf{W}_{kf}$, $\mathbf{E}_{fc} \in \mathbb{R}^{T\times d}, \mathbf{E}_{fc} = \hat{\mathbf{S}}_c \mathbf{W}_{fc}$, T is the total number off frames, and d is the temporal embedded dimension. $\hat{\mathbf{S}}_k \in \mathbb{R}^{T\times 2}$ and $\hat{\mathbf{S}}_c \in \mathbb{R}^{T\times 2}$ are essentially one-hot encoding, where $\hat{\mathbf{S}}_k$ tells keyframes from unknown frames (keyframe = [1,0], unknown frame = [0,1]), $\hat{\mathbf{S}}_c$ is the given ground truth foot contact state (right foot contact = [1,0], left foot contact = [0,1]). $\mathbf{W}_{kf} \in \mathbb{R}^{2\times d}$ and $\mathbf{W}_{fc} \in \mathbb{R}^{2\times d}$ are learnable weight matrixes.

3.3 Transformer block

We leverage the ST-transformer encoder to extract spatio-temporal features for the input motion sequence. The encoder consists of multiple spatial and temporal transformer blocks to extract sequence features for motion synthesis. Each transformer block is composed of the multi-head self-attention (MHSA) and the feed-forward network (FFN) [42].

The MHSA unit is used to establish relationships of input tokens adaptively. Given $\mathbf{X} \in \mathbb{R}^{n\times c}$ with n tokens and embedding dimension c, we first map it into query $\mathbf{Q} \in \mathbb{R}^{n\times c}$, key $\mathbf{K} \in \mathbb{R}^{n\times c}$ and value $\mathbf{V} \in \mathbb{R}^{n\times c}$ through learnable matrices $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$. We split $\mathbf{Q}, \mathbf{K}$ and $\mathbf{V}$ into H heads and leverage the attention mechanism to transform $\mathbf{X}$. The H heads are processed in parallel, and the concatenated outputs are further transformed through a learnable matrix $\mathbf{W}_o \in \mathbb{R}^{c\times c}$. The processing of MHSA is as follows:

$$\mathrm{MHSA}(\mathbf{X}) = \mathrm{Concat}(\mathbf{H}_1, ..., \mathbf{H}_h, ..., \mathbf{H}_H)\mathbf{W}_o, \tag{2}$$

where $\mathbf{H}_h$ denotes the h-th ($h \in [1, ..., H]$) head attention operation for splited query $\mathbf{Q}_h$, key $\mathbf{K}_h$ and value $\mathbf{V}_h$:

$$\mathbf{H}_h = \mathrm{Softmax}(\mathbf{Q}_h \mathbf{K}_h^T / \sqrt{c/H})\mathbf{V}_h. \tag{3}$$

The FFN unit is composed of two fully-connected (FC) layers for feature transformation and adding non-linearity to the model. Its formulation can be expressed as:

$$\mathrm{FFN}(\mathbf{X}) = \sigma(\mathbf{X}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \tag{4}$$

where $\mathbf{W}_1 \in \mathbb{R}^{c\times c1}$ and $\mathbf{W}_2 \in \mathbb{R}^{c1\times c}$ are the learnable matrices, and $\mathbf{b}_1$ and $\mathbf{b}_2$ are the the corresponding bias terms.

Since both the spatial and temporal transformer encoders have L transformer blocks. The whole processing for a transformer encoder can be written as follows:

$$\mathbf{Z}^{(l+1)} = \mathrm{MHSA}(\mathrm{LN}(\mathbf{X}^{(l)})) + \mathbf{X}^{(l)}, \quad l \in [1, ..., L], \tag{5}$$

$$\mathbf{X}^{(l+1)} = \mathrm{FFN}(\mathrm{LN}(\mathbf{Z}^{(l+1)})) + \mathbf{Z}^{(l+1)}, l \in [1, ..., L], \tag{6}$$

where $\mathbf{X}^{(l)}$ and $\mathbf{X}^{(l+1)}$ is the input and output of l-th transformer block, $\mathbf{Z}^{(l+1)}$ is the output of MHSA. $\mathrm{LN}(\cdot)$ refers to layer normalization (LN) layer.

3.4 Root trajectory reconstruction

Basically, a motion sequence can be decomposed into the trajectory of the root joint and rotation angles of all joints. The root joint position characterizes the motion position of an arbitrary frame, while the rotation angles describe the motion postures of different frames. Essentially, the sliding is produced because of the mismatch between the root trajectory and the generated motion postures. In previous methods [3, 7], the root trajectory and motion posture are usually directly regressed without regarding the tight relationship between them, yielding severe sliding issue during motion generation.

We believe that the adaptation of root trajectory and motion posture is paramount in eliminating sliding. Therefore, in motion reconstruction, we do not directly predict the root trajectory from the hidden state but reconstruct it through the foot contact state and motion posture. To this end, we formulate the root trajectory reconstruction (RTR) algorithm to infer the root trajectory matching the foot contact state and motion posture. The whole process of RTR is formulated as:

$$\mathbf{R} = \mathrm{RTR}(\mathbf{S}_c, \mathbf{Y}), \tag{7}$$

where $\mathbf{R}$ denotes the reconstructed root trajectory in the world coordinate system, $\mathbf{S}_c$ refers to the predicted foot contact state, and $\mathbf{Y}$ is the predicted joints quaternions of motion sequence in the relative coordinate system.

To be specific, we first utilize forward kinematics (FK) to calculate joint positions in relative coordinate system. We define the relative coordinate system as the coordinate system with the root joint as the origin:

$$\mathbf{P} = \mathrm{FK}(\mathbf{O}_r, \mathbf{Y}), \tag{8}$$

where $\mathbf{O}_r$ refers to the origin of the relative coordinate system and $\mathbf{P}$ signifies the inferred positions of all joints except the root joint in the relative coordinate system.

Then, we calculate the contact foot displacement for each frame based on the position of the contact foot. The process is displayed in Figure 2 (b). Its formula is described as:

$$\Delta\mathbf{P}_{c,t} = \mathbf{P}_{c,t+1} - \mathbf{P}_{c,t}, \tag{9}$$

where $\mathbf{P}_{c,t+1}$ and $\mathbf{P}_{c,t}$ are the contact foot positions of frame $t+1$ and t, $\Delta\mathbf{P}_{c,t}$ is the contact foot displacement at frame t.

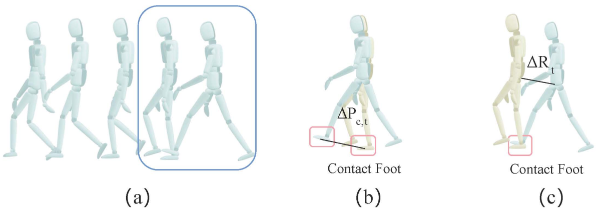


Fig. 2 The proposed RTR algorithm. (a) is the foot contact states of ground truth in world coordinate system, (b) is the predicted motion postures in relative coordinate system, (c) is the root trajectory in world coordinate system inferred by RTR.

As illustrated in Figure 2 (b), in relative coordinate system, the contact foot displacement is $\Delta\mathbf{P}_{c,t}$. While in world coordinate system, the contact foot is not movable in Figure 2 (c). We can complete the transition from the relative coordinate system to the world coordinate system by moving the origin of the relative coordinate system. Due to the relativity of motion, the displacement of the origin of the relative coordinate system subtracting the same value can compensate for the displacement of the contact foot. Therefore, it can be inferred that the displacement of the root joint $\Delta\mathbf{R}_t$ is:

$$\Delta\mathbf{R}_t = -\Delta\mathbf{P}_{c,t}, \tag{10}$$

$$\mathbf{R}_{t+1} = \mathbf{R}_t + \Delta\mathbf{R}_t, \tag{11}$$

where $\mathbf{R}_{t+1}$ is the root joint position at frame t. We eliminate foot sliding in the world coordinate system by using the root joint displacement to compensate for the displacement of the contact foot in the relative coordinate system.

3.5 Neural network

Our framework is shown in Figure 1, an encoder-decoder architecture for motion generation. The encoder concentrates on dissecting spatial-temporal relationships and learning powerful features for input motion sequences. The decoder strives to predict motion posture and foot contact state information. In short, our network mainly contains four modules: spatial-transformer, temporal-transformer, state decoder, and contact decoder. We elaborate on each module in detail.

Spatial-transformer. The spatial-transformer is leveraged to model spatial relationships among skeleton joints within each frame. It is constructed by stacking L-layer transformers. The whole network input $\mathbf{X}_{slerp} = \{\mathbf{X}_1, ..., \mathbf{X}_T\} \in \mathbb{R}^{T\times J\times 4}$ is the Slerp interpolation of joint quaternions between keyframes, where 4 is the dimension of the quaternion, T is the number of frames, and J is the number of joints. We consider each joint as a token and embed them into a c-dimensional space through the matrix $\mathbf{W_s} \in \mathbb{R}^{4\times c}$. Following [43], a learnable spatial position embedding $\mathbf{E_s} \in \mathbb{R}^{J\times c}$ is added to the projected tokens, that is:

$$\mathbf{X}_{si} = \mathbf{X}_{slerp}\mathbf{W_s} + \mathbf{E}_s, \tag{12}$$

where $\mathbf{X}_{si}$ serves as the input to the spatial-transformer.

For each layer in spatial-transformer module, we implement Eq. (5) and Eq. (6) untill all the L blocks done. We denote the output of the spatial-transformer module as $\mathbf{H}_{so} \in \mathbb{R}^{T\times J\times c}$.

Temporal-transformer. The temporal-transformer concentrates on establishing temporal relationships between different frames for the input motion sequence. It is also stacked by L transformer blocks. We flatten the output $\mathbf{H}_{so}$ as $\mathbf{H}_t \in \mathbb{R}^{T\times d}$ $(d = J\times c)$. Before feeding $\mathbf{H}_t$ into the temporal-transformer, we incorporate the learnable temporal position embedding $\mathbf{E}_t \in \mathbb{R}^{T\times d}$ and the mix embedding $\mathbf{E}_m \in \mathbb{R}^{T\times d}$. Therefore, the input of the temporal-transformer $\mathbf{X}_{ti}$ can be described as:

$$\mathbf{X}_{ti} = \mathbf{H}_t + \mathbf{E}_t + \mathbf{E}_m. \tag{13}$$

The temporal-transformer module follows the same steps as the spatial-transformer, and the output is $\mathbf{X}_{to} \in \mathbb{R}^{T\times d}$.

State decoder. The state decoder is utilized to predict motion posture residuals, which will be later added to the input motion sequence to obtain the reconstructed motion movements. Specifically, the state decoder consists of a fully connected layer to transform the extracted spatio-temporal features $\mathbf{X}_{to}$ into posture residuals $\mathbf{X}_{res} \in \mathbb{R}^{T\times J\times 4}$:

$$\mathbf{X}_{res} = \mathbf{X}_{to}\mathbf{W}_{res} + \mathbf{b}_{res}, \tag{14}$$

where $\mathbf{W}_{res}$ and $\mathbf{b}_{res}$ are the learnable weight matrix and bias. Thus, the completed motion postures $\mathbf{Y}$ can be achieved:

$$\mathbf{Y} = \mathbf{X}_{slerp} + \mathbf{X}_{res}. \tag{15}$$

Contact decoder. The contact decoder is designed to predict the foot contact states $\mathbf{S}_c \in \mathbb{R}^{T\times 2}$. Unlike previous methods that predict motion postures and root trajectory in parallel, neglecting the close relationship between them, we infer the root joint trajectory based on the predicted foot contact states and joint angles. Our contact decoder is also composed of a fully connected layer, its operation is similar as the state decoder:

$$\mathbf{S}_c = \mathbf{X}_{to}\mathbf{W}_{con} + \mathbf{b}_{con}, \tag{16}$$

where $\mathbf{W}_{con}$ and $\mathbf{b}_{con}$ are the learnable weight matrix and bias for the contact decoder.

3.6 Motion control

Root trajectory control is one of the most used control constraint in previous works [7, 8]. However, if the root trajectory generated according to the control signal does not match the generated motion posture, then the produced animation is usually accompanied by sliding and unreality.

In this work, our control signals are the user-defined contact state and keyframes. In training phase, we train the network with the foot contact states and motion postures of ground truth as supervision. In testing phase, the foot contact signal is used as the network input, and the network predicts the corresponding foot contact state $\mathbf{S}_c$ and motion posture $\mathbf{Y}$ through the contact decoder and state decoder. Finally, We use the root trajectory reconstruction algorithm to inference the corresponding trajectory. Our model implicitly controls the root trajectory by manipulating the foot contact state and ensures the generated root trajectory matches motion postures with better visual effects.

3.7 Loss function

For the proposed neural network, the output is the quaternion of the joint and the foot contact state. Given motion posture sequence $\hat{\mathbf{Y}} = \{\hat{\mathbf{Y}}_1, ..., \hat{\mathbf{Y}}_T\}$ and foot contact sequence $\hat{\mathbf{S}}_c = \{\hat{\mathbf{S}}_{c,1}, ..., \hat{\mathbf{S}}_{c,T}\}$, we employ the mean square error and cross-entropy losses to reconstruct motion postures and predict foot contact state. The two main losses ensure our model generates motion movements to satisfy the user-defined constraints:

$$\mathcal{L}_{rec} = \frac{1}{T}\sum_{t=1}^{T} \|\hat{\mathbf{Y}}_t - \mathbf{Y}_t\|_2, \tag{17}$$

$$\mathcal{L}_{con} = -\frac{1}{T}\sum_{t=1}^{T}(\hat{\mathbf{S}}_{c,t}*\log(\mathbf{S}_{c,t})+(1-\hat{\mathbf{S}}_{c,t})*\log(1-\mathbf{S}_{c,t})). \tag{18}$$

In addition, we also incorporate two auxiliary losses *i.e.*, joint position loss $\mathcal{L}_p$ and root trajectory loss $\mathcal{L}_r$, to facilitate our network optimization:

$$\mathcal{L}_p = \frac{1}{T}\sum_{t=1}^{T} \|\hat{\mathbf{P}}_t - \mathbf{P}_t\|_2, \tag{19}$$

$$\mathcal{L}_r = \frac{1}{T}\sum_{t=1}^{T} \|\hat{\mathbf{R}}_t - \mathbf{R}_t\|_2, \tag{20}$$

where $\hat{\mathbf{P}}_t$ and $\mathbf{P}_t$ are the ground truth and predicted joint positions for frame t in relative coordinate system, while $\hat{\mathbf{R}}_t$ and $\mathbf{R}_t$ are the ground truth and predicted root joint positions for frame t in world coordinate system.

In summary, the whole loss during training is described as:

$$\mathcal{L} = \mathcal{L}_{rec} + \mathcal{L}_{con} + \lambda * (\mathcal{L}_p + \mathcal{L}_r), \tag{21}$$

where λ balances the weights of the main and auxiliary losses and is empirically set as $1e-5$ with extensive experiments.

4 Experiments

4.1 Datasets and protocols

LaFAN1. The LaFAN1 dataset [3] is a widely used public dataset to evaluate motion transition quality. It records $496,672$ motion frames using MOCAP sampled at 30Hz, where five subjects are asked to perform a variety of actions with large global displacements. Following the protocol in [3], we choose the motion frames of subject 5 as the testing set and other subjects as the training set. Considering the long motion sequence of each segment, the dataset is divided with sliding windows of fixed lengths. Specifically, we set the window length to 50 frames with an offset of 20 frames for the training set, while the window length is set to 65 frames with 25 frames overlapping for the testing set. Thus, we possess 20,212 motion clips for training and 2,232 clips for testing. Qualitative and quantitative experiments are performed the LaFAN1 datasets to evaluate our model.

Dance. The dance dataset is collected by Aristidou *et al.* [44]. According to the division of Pan *et al.* [7], there are 123 pieces of contemporary dance and 93,347 motion frames in total. Among them, 98 segments with 93,347 frames are used as the training set and 25 segments with 20,897 frames are used as the test set. Following Pan *et al.* [7], we also resort to a fixed sliding window to divide the dance dataset to facilitate experiments. For both the training and testing sets, the sliding window's length is 50 frames, and the offset is 10 frames. We conduct qualitative and quantitative experiments on this dataset and make state-of-the-art methods.

Martial Arts. We employ the martial arts data gathered by zhou *et al.* [6], which originated from the publicly available CMU motion capture dataset ①. The Martial Art dataset consists of 10 actors performing contemporary martial, resulting 10 segments with a total number of 29,183 motion frames. In experiments, we take the last nine actors, 23,066 frames, as the training set and the first actor, 6,117 frames, as the testing set. Similarly, we use a sliding window to divide the dataset with the window length and offset number as 50 frames and 10 frames. Since no quantitative results are reported for the dataset, we only implement qualitative control experiments and make comparisons.

Evaluation protocols. We follow Harvey *et al.* [3] and utilize L2Q, L2P and NPSS as our evaluation metrics. The L2Q characterizes the average L-2 distances of the quaternions between the predicted motion postures and their ground truth. Similarly, the L2P expresses the average L2 distances of the relative positions. L2Q and L2P can be described as:

$$L2Q = \frac{1}{|D|}\frac{1}{T}\sum_{s\in D}\sum_{t=1}^{T}\|\hat{\mathbf{Y}}_t - \mathbf{Y}_t\|_2, \tag{22}$$

$$L2P = \frac{1}{|D|}\frac{1}{T}\sum_{s\in D}\sum_{t=1}^{T}\|\hat{\mathbf{P}}_t - \mathbf{P}_t\|_2, \tag{23}$$

where s is a transition sequence of the test set D, $|D|$ is size of D, and T is the transition length.

The NPSS, proposed by Gopalakrishnan *et al.* [45], is a variant of L2Q, which computes the Normalized Power Spectrum Similarity and is based on the angular frequency distance between the prediction and the ground truth. The detailed formula can be found in [45].

4.2 Implementation details

In our network, the encoder is stacked with 4 spatial transformer and 4 temporal transformer layers, *i.e.,* $L = 4$. For the spatial transformer, we embed the quaternion into the 32-dimensional space. In the temporal transformer, the embedding dimension is given by the product of the number of joints and the spatial embedded dimension. The head number for the MSHA layer is set as 8. Considering that the joint quaternion values vary between 0 and 1, in contrast, the range of values of the absolute coordinate positions of the root joint and the other joints' relative coordinate positions changes significantly, we normalize them both and normalize their values to between 0 and 1 respectively to accelerate network learning. Since some small sliding still exists in the raw data, we reconstruct the root trajectory of the raw dataset with our proposed RTR algorithm to reduce the adverse effect in the training process. Adam optimizer is leveraged to train our network, and we set the learning rate as 0.001 and the maximum training times as 1000. The whole framework is implemented on the PyTorch platform.

4.3 Comparisons with state-of-the-arts

Experiment on LaFAn1 dataset. We implement quantitative experiment on the LaFan1 dataset and make comparisons with the linear interpolation, as well as deep neural networks proposed by Harvey *et al.* [3], Pan *et al.* [7] and Duan *et al.* [4]. The L2Q, L2P, and NPSS assessment metrics for the LAFAN1 dataset are evaluated. During the test, we only give the starting and ending frames of the keyframes and report the results of transition length of 5 frames, 30 frames, and 45 frames in Table 1. We can observe that our method can achieve outstanding performance on either long-term or short-term motion transition, demonstrating the strength and advantages of our network compared with the state-of-the-art methods on the LaFAn1 dataset.

Experiment on Dance dataset. We also conduct quantitative analysis on the Dance dataset. Motion movements in this dataset are more complex and challenging. We contrast with the linear interpolation, as well as state-of-the-art methods proposed by Harvey *et al.* [3], Pan *et al.* [7] and Duan *et al.* [4], and report the L2P, L2Q, and NPSS results in Table 1. Motion performance with transition lengths of 10 frames, 30 frames, and 50 frames are assessed. From Table 1, we can notice that our approach confirms impressive benefits for the three metrics compared with corresponded methods in long-term motion transition, *i.e.,* 30 and 50 frames. However, in the short-term transition of 10 frames, our results perform worse than the direct interpolation strategy. The same suboptimal performance can be found in the works of Harvey *et al.* [3], Pan *et al.* [7] and Duan *et al.* [4]. We suppose that this is because the motion changes almost linearly in a short enough period. Nevertheless, experiments on the Dance dataset validate the significantly superior of our method compared with the most advanced endeavors.

4.4 Ablation study

We implement ablation studies on LaFAN1 and Dance datasets to validate the effectiveness and superiority of different components of our model. The network with slerp interpolation motion as input and the posture reconstruction loss as supervision is regarded as the baseline. Experiment results of L2Q, L2P, and NPSS on the two datasets are illustrated in Table 2.

① http://mocap.cs. cmu.edu/.

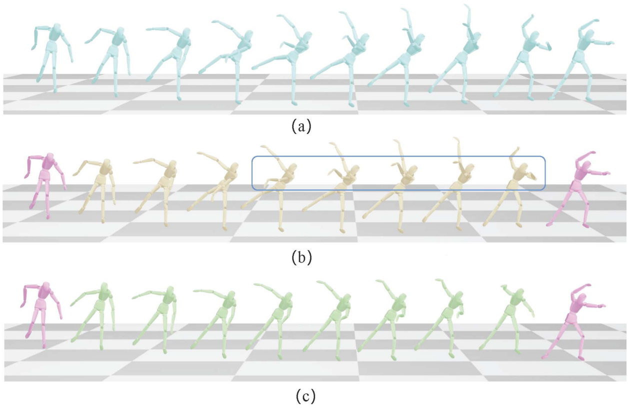
Table 1 Comparisons with state-of-the-art methods on LaFAN1 and Dance datasets. The best results are denoted in bold font.

Method	LaFAN1									Dance								
	L2Q (↓)			**L2P (↓)**			**NPSS (↓)**			**L2Q (↓)**			**L2P (↓)**			**NPSS (↓)**		
	5	30	45	5	30	45	5	30	45	10	30	50	10	30	50	10	30	50
Interpolation	0.22	0.98	1.25	0.37	1.84	2.19	0.0023	0.2013	0.4493	**0.31**	0.68	1.15	**0.87**	3.68	5.27	**0.0141**	0.1605	0.4205
Harvey *et al.* [3]	0.17	0.69	0.94	0.28	1.66	2.01	0.0020	0.1328	0.3311	0.49	0.54	0.96	1.56	1.95	4.75	0.0721	0.1313	0.3342
Pan *et al.* [7]	0.19	0.64	0.74	0.31	1.63	1.87	0.0021	0.1256	0.2579	0.36	0.52	1.07	0.98	1.91	5.17	0.0194	0.1063	0.3831
Duan *et al.* [4]	0.14	0.61	0.72	0.27	1.62	1.82	0.0016	0.1222	0.1431	0.33	0.48	0.86	0.95	1.78	4.46	0.0168	0.0719	0.2991
Ours	**0.07**	**0.46**	**0.58**	**0.21**	**1.37**	**1.69**	**0.0006**	**0.0494**	**0.0991**	0.32	**0.42**	**0.79**	0.91	**1.44**	**4.12**	0.0145	**0.0612**	**0.2447**

Table 2 Ablation study on LaFAN1 and Dance datasets. The best results are denoted in bold font.

Method	Components	LaFAN1									Dance								
		L2Q (↓)			**L2P (↓)**			**NPSS (↓)**			**L2Q (↓)**			**L2P (↓)**			**NPSS (↓)**		
		5	30	45	5	30	45	5	30	45	10	30	50	10	30	50	10	30	50
1	Linear	0.11	0.52	0.64	0.26	1.56	1.75	0.0007	0.0588	0.1231	0.37	0.48	0.84	0.99	1.80	4.39	0.0195	0.0763	0.2871
2	Slerp (Baseline)	0.09	0.50	0.61	0.25	1.53	1.74	0.0007	0.0576	0.1143	0.35	0.47	0.82	0.96	1.79	4.31	0.0173	0.0732	0.2707
3	$+\mathbf{E}_m$	0.08	0.49	0.60	0.25	1.53	1.73	0.0006	0.0579	0.1102	0.34	0.43	0.81	0.94	1.68	4.21	0.0151	0.0688	0.2671
4	$+\mathcal{L}_{con}$	0.08	0.46	0.58	0.23	1.51	1.72	0.0006	0.0558	0.1091	0.33	0.43	0.80	0.93	1.67	4.17	0.0150	0.0647	0.2576
5	$+\mathcal{L}_p$	0.08	**0.45**	**0.55**	0.21	1.38	**1.68**	0.0006	**0.0492**	0.0993	0.32	0.42	0.80	0.92	1.45	4.14	0.0147	0.0631	0.2458
6	$+\mathcal{L}_r$	**0.07**	0.46	0.58	**0.21**	**1.37**	1.69	**0.0006**	0.0494	**0.0991**	**0.32**	**0.42**	**0.79**	**0.91**	**1.44**	**4.12**	**0.0145**	**0.0612**	**0.2447**

Effectiveness of Slerp interpolation. In our method, we utilize the interpolation algorithm to fill the unknown frames between keyframes to obtain the complete sequence as the network input. The quality of the filled frames significantly impacts the performance of the model learning. Unlike traditional linear interpolation, we leverage the Slerp [25] algorithm to synthesize the missing frames. To verify its superiority, we conduct experiments to compare the linear and Slerp interpolation strategies. It can be clearly observed from the first two rows in Table 2 that Slerp demonstrates substantial error reductions for the L2Q, L2P, and NPSS criteria on the LaFAN1 and Dance datasets. We believe that the performance enhancement attribute to the Slerp interpolation is more suitable for the quaternion representation of motion postures.


(a)

(b)

(c)

Fig. 3 Ablation experiments of mix embedding. (a) is ground truth motion. (b) is the generated motion with mix embedding. (c) is the generated motion without mix embedding.

Effectiveness of mix embedding. The mix embedding $\mathbf{E}_m$ contains the keyframe embedding and the foot contact embedding, respectively, responsible for distinguishing between keyframes and unknown frames, as well as different foot contact states. To validate the effectiveness of $\mathbf{E}_m$, we incorporate it into the baseline network and display the results in Table 2. It can be summarized that with $\mathbf{E}_m$, the errors of evaluation indicators, *i.e.*, L2Q, L2P, and NPSS, have decreased significantly on LaFAN1 and Dance datasets. Motion generation comparisons with and without $\mathbf{E}_m$ are further exhibited in Figure 3. We can summarize that the mix embedding improves the quality of the generated motion. The performance gains demonstrate the positive effect and superiority of considering the mix embedding characteristic.

Effectiveness of foot contact loss. The foot contact loss $\mathcal{L}_{con}$ is leveraged to synthesize motion movements towards defined constraints. To verify its effect, we further implement an experiment to add the loss term. As shown in Table 2, with the proposed $\mathcal{L}_{con}$, method 4 achieves much better performance, especially on the long-term motion completion compared with method 3 for the L2Q, L2P, and NPSS metrics. The reason for the improvements is that $\mathcal{L}_{con}$ can guarantee accurate foot contact states prediction, so as to ensure stable and high-quality motion generation without foot sliding issues.

Effectiveness of joint position loss. The foot contact loss $\mathcal{L}_p$ is used to assist model learning and ensure motion transition robustness. By comparing methods 4 and 5 in Table 2, we can observe that incorporating $\mathcal{L}_p$, the L2P error has dropped significantly. This may be due to the accumulated error of rotation angle in local coordinates, which leads to the increase of position error at the end joint. We also compare the motion synthesis performance of with and without $\mathcal{L}_p$

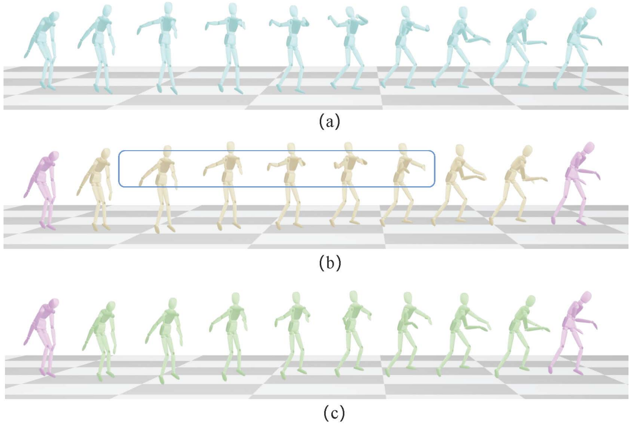


Fig. 4 Ablation experiments of joint position loss. (a) is ground truth motion. (b) is the generated motion with joint position loss. (c) is the generated motion without joint position loss.

with ground truth and display the results in Figure 4. It can safely conclude that the addition of $\mathcal{L}_p$ enhances the motion generation effects of the end-effector joints. The performance improvements demonstrate the superiority of applying joint position loss for restraining motion movements.

Effectiveness of root trajectory loss. The root trajectory loss $\mathcal{L}_r$ aims at generating motion that corresponds to the ground truth motion trajectory. It also optimizes the position information. In Table 2, it can be concluded that with the auxiliary loss $\mathcal{L}_r$, method 6 reduces the errors of L2Q and NPSS for different length dance transitions compared with method 5. In contrast, the L2P loss is being optimized for both the LaFAN 1 and Dance datasets. The results demonstrate the effectiveness of root trajectory loss.

4.5 Motion synthesis

We visualize the motion synthesis results of Slerp interpolation [25], Harvey *et al.* [3], Pan *et al.* [7], Duan *et al.* [4] and our method for the LaFAN1, Dance and Martial Arts datasets. We randomly select the periodic walk motion as well as the complex non-periodic dance performance and martial arts movements for visualization. The generated transition sequences of 30 frames are drawn every three frames. The motion completion results for given keyframes are exhibited in Figure 5. It can be observed from Figure 5 (a) that for simple periodic action, *i.e.*, walking, the intermediate frames generated by our method are more proximate to the ground truth. For more complex movements such as, dance in Figure 5 (b) and martial arts in Figure 5 (c), Harvey's and Pan's methods appear to deviate from the target keyframe, and Duan's method performs more unnaturally in the later stages of motion generation, while our model can still generate complex movements with high performance.

Furthermore, the pure image visualization results can not intuitively display the sliding problem well. We have additionally provided supplementary video material [①]. It can be clearly seen from the video materials that the Slerp [25], Harvey *et al.* [3], Pan *et al.* [7], Duan *et al.* [4]'s methods somewhat have serious sliding problems no matter in simple or complex movements. On the contrary, our method has solved this problem well and yields higher fidelity and quality motion movements for diversified actions.

4.6 Depth analysis

Foot sliding discussion. Sliding is challenging to evaluate quantitatively, so we randomly choose a segment of motion generated by Harvey *et al.* [3], Pan *et al.* [7], Duan *et al.* [4] and our method and depict the synthesized motions in Figure 6. The transition segment is 10 frames, and visualization is performed every three frames. As shown in Figure 6, we have marked the foot contact states with light blue rectangular boxes. Compared to the ground truth (GT) foot contact state, it can be observed that the contact foot, which should be supposed to maintain the same, have shifted in Harvey's, Pan's, and Duan's approaches. While the reconstructed root trajectory in our model matches the generated motion postures, so the resulting movements exhibit satisfactory results.

Same keyframes with different contact foots. The control constraint of our method involves the contact foot states. Given the same keyframes with different foot contact states, diversified motion postures and disparate root trajectories can be generated. We implement experiments and illustrate the results in Figure 7, where three different foot contact signals are specified in Figure 7 (a), and the generated motions and corresponding root trajectories are drawn in Figure 7 (b) and (c). We change the control signals every 10 frames and visualize it every 2 frames. It can be observed that the induced transitions not only satisfy the corresponding control information but also guarantee diversity with no sliding.

Same contact foots in different keyframes. Apart from the foot contact constraint, the keyframe is also a significant control signal. We further visualize the generated motion transition given the same contact foot states in different keyframes in Figure 8 (a) and (b). It can be observed that although having the same foot contacts, different keyframes lead to entirely different motion movements and root trajectories. The results demonstrate that the keyframes are critical in the motion generation process.

[①] https://www.youtube.com/watch?v=7Nom_5ljL38

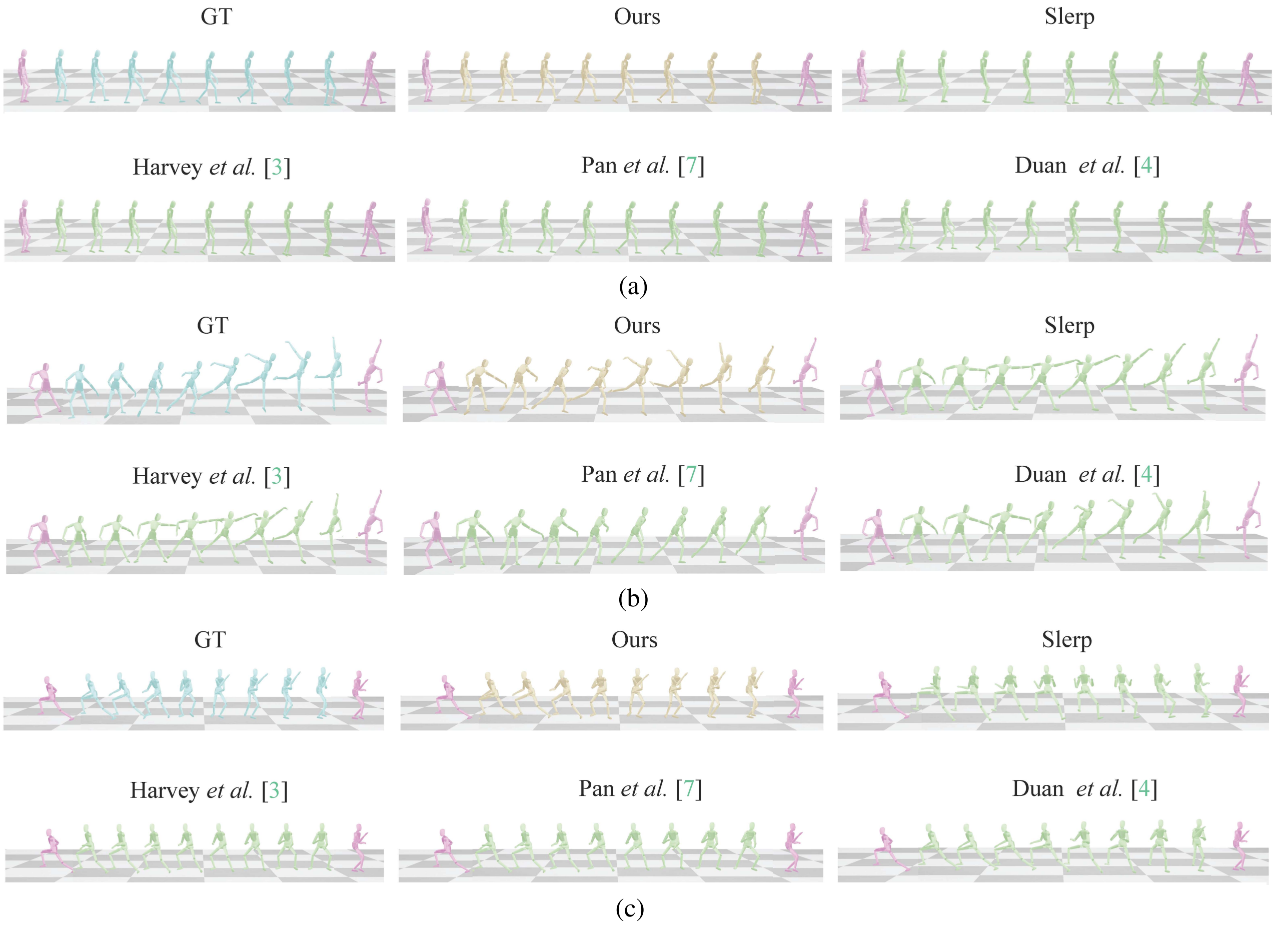


Fig. 5 Motion transition results on the (a) LaFAN1, (b) Dance, and (c) Martial Arts datasets. Purple skeletons represent keyframes.

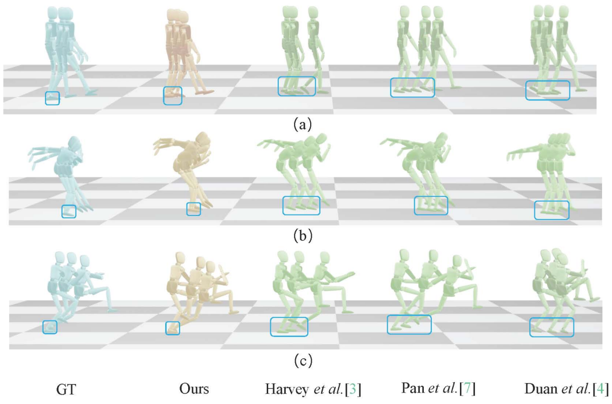


Fig. 6 Visualization of the foot sliding issues on the (a) LaFAN1, (b) Dance, and (c) Martial Arts datasets.

Feet touching the ground. In our model, to simplify learning, we only assume two contact states, *i.e.*, left foot touching the ground and right foot touching the ground. However, it is common for both feet to touch the ground in practice. When the case is triggered, we randomly choose the left foot contact or right foot contact state to infer the motion posture of the next frame, and the results are illustrated in Figure 9. We can observe that whether we authorize the left or right foot contact with the ground has little effect on the generated motion postures. Correspondingly, the inferred root trajectories are consistent with the ground truth. The qualitative visualization results demonstrate that the simplified treatment of the feet touching the ground is adequate.

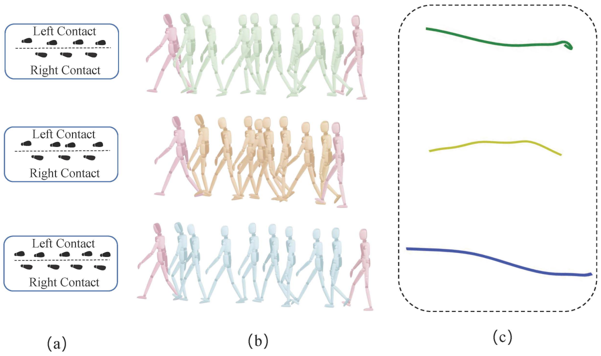


Fig. 7 Motion transition under the same keyframes with different foot contacts. (a) is control signals. (b) is the generated motions. (c) is the corresponding root trajectories.

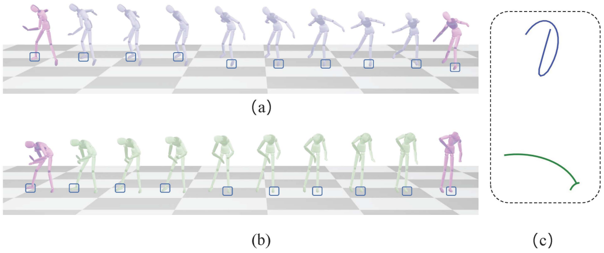


Fig. 8 Motion synthesis under the same contact foots in different keyframes. (a) and (b) are the generated motions. (c) is the corresponding root trajectories.

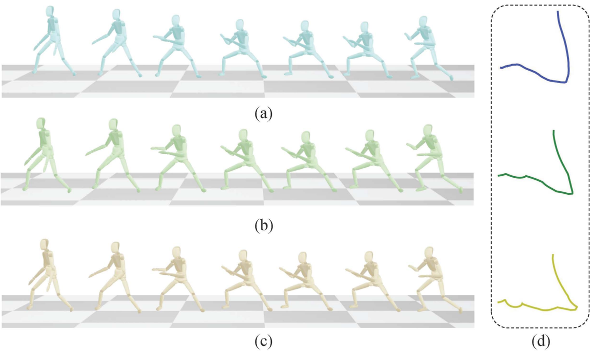


Fig. 9 Feet touching the ground. (a) is ground truth motion. (b) is the generated motion with left foot regarded as the contact foot. (c) is the generated motion with right foot regarded as the contact foot. (d) is the corresponding root trajectories.

4.7 Limitations

Our method can skillfully manage the foot sliding issue in motion generation based on the foot constraint. To constrain the network optimization and synthesize satisfactory motions, we define two contact states, *i.e.*, left foot contact with the ground and right foot contact with the ground. The proposed RTR algorithm performs motion transition well when either one of the feet on the ground. While both feet touch the ground, the RTR algorithm achieves consistent performance by randomly selecting one of the defined states. However, such strategy might cause artifacts due to the lack of contact feet when both feet off the ground. Although forcing the left or right foot contact signal can be leveraged to produce motions, the generated movements perform weirdly with always one foot in contact states. We display the failure case in Figure 10. In future, we will consider an auxiliary inference strategy based on physical models to assist in the reconstruction of root trajectory for the special case of feet off the ground.

Last but not least, similar to existing data-driven methods [3, 4, 7], it is conceivable that unsatisfactory outcomes produced with our model when synthesizing motions deviate significantly from the training set. Such an annoyance can be alleviated by expanding the diversity and scale of the dataset or increasing the network generalization capability.

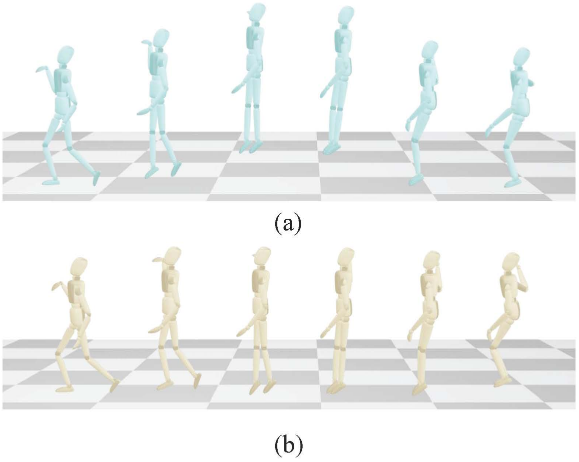


Fig. 10 Feet off the ground. (a) is the ground truth motion. (b) is the generated motion by our method.

5 Conclusion

Analyzing existing motion synthesis methods, we have found that the foot sliding problem inevitably emerges when generating complex motion. The reasons are the mismatch between the root trajectory and motion posture, which is usually caused by the separated prediction process in the model design. Aiming to yield high-quality and high-fidelity complex motion transitions, we propose a novel spatial-temporal transformer network conditioned on foot contact information. In addition, a differentiable root trajectory reconstruction (RTR) algorithm is incorporated into our model. Unlike isolated decoding motion postures and root trajectory techniques, the RTR algorithm reconstructs the root trajectory leveraging the predicted motion postures and foot contact states. Ablation experiments and depth analysis validate the effectiveness of different modules of our model in generating stable and robust high-quality motion movements. Furthermore, qualitative and quantitative comparisons with the existing advanced methods on the public LaFAN1, Dance, and Martial Arts datasets demonstrate the superiority of our approach in addressing the foot sliding problem. In the future, we will focus on feet off the ground issue and promote the network generalization ability. The source code will be available at https://github.com/wslh852/Keyframe-based-Complex-Motion-Synthesis.git after the paper is accepted.

Acknowledgments. This research is supported by National Key R&D Program of China (No. 2022ZD0115902), National Natural Science Foundation of China (Nos. 62102208, 62272017, U20A20195, 62172437).

Declaration of competing interest. The authors have no competing interests to declare that are relevant to the content of this article.

Electronic Supplementary Material. We provide a supplementary video to describe our implementation and show visual comparisons of our method with state-of-the-art methods.

References

[1] Qin J, Zheng Y, Zhou K. Motion In-Betweening via Two-Stage Transformers. *ACM Transactions on Graphics*, 2022, 41(6): 1–16.

[2] Kim J, Byun T, Shin S, Won J, Choi S. Conditional motion in-betweening. *Pattern Recognition*, 2022, 132: 108894.

[3] Harvey FG, Yurick M, Nowrouzezahrai D, Pal C. Robust motion in-betweening. *ACM Transactions on Graphics*, 2020, 39(4): 60:1–60:12.

[4] Duan Y, Lin Y, Zou Z, Yuan Y, Qian Z, Zhang B. A Unified Framework for Real Time Motion Completion. In *Association for the Advancement of Artificial Intelligence*, 4459–4467.

[5] Roberts R, Lewis JP, Anjyo K, Seo J, Seol Y. Optimal and interactive keyframe selection for motion capture. *Computational Visual Media*, 2019, 5(2): 171–191.

[6] Zhou Y, Li Z, Xiao S, He C, Huang Z, Li H. Auto-Conditioned Recurrent Networks for Extended Complex Human Motion Synthesis. In *International Conference on Learning Representations*, 2018.

[7] Pan J, Wang S, Bai J, Dai J. Diverse Dance Synthesis via Keyframes with Transformer Controllers. *Computer Graphics Forum*, 2021, 40(7): 71–83.

[8] Harvey FG, Pal C. Recurrent transition networks for character locomotion. In *SIGGRAPH Asia Technical Briefs*, 2018, 1–4.

[9] Prazák M, Hoyet L, O'Sullivan C. Perceptual evaluation of footskate cleanup. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2011, 287–294.

[10] Harrison J, Rensink RA, van de Panne M. Obscuring length changes during animated motion. *ACM Transactions on Graphics*, 2004, 23(3): 569–573.

[11] Lyard E, Magnenat-Thalmann N. A simple footskate removal method for virtual reality applications. *The Visual Computer*, 2007, 23(9-11): 689–695.

[12] Smith HJ, Cao C, Neff M, Wang Y. Efficient Neural Networks for Real-time Motion Style Transfer. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2019, 2(2): 13:1–13:17.

[13] Kovar L, Schreiner J, Gleicher M. Footskate cleanup for motion capture editing. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2002, 97–104.

[14] Zhang H, Starke S, Komura T, Saito J. Mode-adaptive neural networks for quadruped motion control. *ACM Transactions on Graphics*, 2018, 37(4): 145:1–145:11.

[15] Ruiz AH, Gall J, Moreno F. Human Motion Prediction via Spatio-Temporal Inpainting. In *IEEE International Conference on Computer Vision*, 2019, 7133–7142.

[16] Tang X, Wang H, Hu B, Gong X, Yi R, Kou Q, Jin X. Real-time controllable motion transition for characters. *ACM Transactions on Graphics*, 2022, 41(4): 137:1–137:10.

[17] Xiao Y, Lai Y, Zhang F, Li C, Gao L. A survey on deep geometry learning: From a representation perspective. *Computational Visual Media*, 2020, 6(2): 113–133.

[18] Xing J, Hu W, Zhang Y, Wong T. Flow-aware synthesis: A generic motion model for video frame interpolation. *Computational Visual Media*, 2021, 7(3): 393–405.

[19] Wang JM, Fleet DJ, Hertzmann A. Gaussian Process Dynamical Models. 2005, 1441–1448.

[20] Lehrmann AM, Gehler PV, Nowozin S. Efficient Nonlinear Markov Models for Human Motion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, 1314–1321.

[21] Wang H, Ho ESL, Shum HPH, Zhu Z. Spatio-Temporal Manifold Learning for Human Motions via Long-Horizon Modeling. *IEEE Transactions on Visualization and Computer Graphics*, 2021, 27(1): 216–227.

[22] Tang X, Wang H, Hu B, Gong X, Yi R, Kou Q, Jin X. Real-time controllable motion transition for characters. *ACM Transactions on Graphics*, 2022, 41(4): 137:1–137:10.

[23] Kaufmann M, Aksan E, Song J, Pece F, Ziegler R, Hilliges O. Convolutional Autoencoders for Human Motion Infilling. In *International Conference on 3D Vision*, 2020, 918–927.

[24] Martinez J, Black MJ, Romero J. On Human Motion Prediction Using Recurrent Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, 4674–4683.

[25] Leeney M. Fast quaternion slerp. *International Journal of Computer Mathematics*, 2009, 86(1): 79–84.

[26] Ling HY, Zinno F, Cheng G, van de Panne M. Character controllers using motion VAEs. *ACM Transactions on Graphics*, 2020, 39(4): 40.

[27] Lee M, Lee K, Park J. Music similarity-based approach to generating dance motion sequence. *Multimedia Tools and Applications*, 2013, 62(3): 895–912.

[28] Zou Y, Yang J, Ceylan D, Zhang J, Perazzi F, Huang J. Reducing footskate in human motion reconstruction with ground contact constraints. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2020, 448–457.

[29] Wang Z, Chai J, Xia S. Combining Recurrent Neural Networks and Adversarial Training for Human Motion Synthesis and Control. *IEEE Transactions on Visualization and Computer Graphics*, 2021, 27(1): 14–28.

[30] Holden D, Saito J, Komura T. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics*, 2016, 35(4): 138:1–138:11.

[31] Agrawal S, van de Panne M. Task-based locomotion. *ACM Transactions on Graphics*, 2016, 35(4): 1–11.

[32] Kovar L, Gleicher M, Pighin FH. Motion graphs. *ACM Transactions on Graphics*, 2002, 21(3): 473–482.

[33] Beaudoin P, Coros S, van de Panne M, Poulin P. Motion-motif graphs. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2008, 117–126.

[34] Shen Y, Wang H, Ho ESL, Yang L, Shum HPH. Posture-based and action-based graphs for boxing skill visualization. *Computers Graphics*, 2017, 69: 104–115.

[35] Chai J, Hodgins JK. Constraint-based motion optimization using a statistical dynamic model. *ACM Transactions on Graphics*, 2007, 26(3): 8–es.

[36] Min J, Chai J. Motion graphs++: a compact generative model for semantic motion analysis and synthesis. *ACM Transactions on Graphics*, 2012, 31(6): 153:1–153:12.

[37] Wang JM, Fleet DJ, Hertzmann A. Gaussian Process Dynamical Models for Human Motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008, 30(2): 283–298.

[38] Holden D, Saito J, Komura T, Joyce T. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia technical briefs*, 2015, 18:1–18:4.

[39] Lee K, Lee S, Le J. Interactive character animation by learning multi-objective control. *ACM Transactions on Graphics*, 2018, 37(6): 180:1–180:10.

[40] Holden D, Komura T, Saito J. Phase-functioned neural networks for character control. *ACM Transactions on Graphics*, 2017, 36(4): 42:1–42:13.

[41] Bergamin K, Clavet S, Holden D, Forbes JR. DReCon: data-driven responsive control of physics-based characters. *ACM Transactions on Graphics*, 2019, 38(6): 206:1–206:11.

[42] Xu Y, Wei H, Lin M, Deng Y, Sheng K, Zhang M, Tang F, Dong W, Huang F, Xu C. Transformers in computational visual media: A survey. *Computational Visual Media*, 2022, 8(1): 33–62.

[43] Zheng C, Zhu S, Mendieta M, Yang T, Chen C, Ding Z. 3D Human Pose Estimation with Spatial and Temporal Transformers. In *IEEE International Conference on Computer Vision*, 2021, 11636–11645.

[44] Aristidou A, Zeng Q, Stavrakis E, Yin K, Cohen-Or D, Chrysanthou Y, Chen B. Emotion control of unstructured dance movements. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 9:1–9:10.

[45] Gopalakrishnan A, Mali AA, Kifer D, Giles CL, II AGO. A Neural Temporal Model for Human Motion Prediction. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, 12116–12125.