International Journal of Machine Learning and Cybernetics (2021) 12:2255–2267
https://doi.org/10.1007/s13042-021-01304-w

ORIGINAL ARTICLE

Hierarchical learning recurrent neural networks for 3D motion synthesis

Dongsheng Zhou[1,2] · Chongyang Guo[1] · Rui Liu[1] · Chao Che[1] · Deyun Yang[3] · Qiang Zhang[1,2] · Xiaopeng Wei[2]

Received: 11 May 2020 / Accepted: 8 March 2021 / Published online: 29 March 2021

Abstract
Three-dimensional human motion synthesis is one of the key technologies in the field of computer animation and multimedia applications. It is well known that the human body's own motion is full of strong personality, emotion, and high-dimensional characteristics, leading to the automatic synthesis of diverse and lifelike 3D human motion data continues to be a challenging task. Facing the challenge, this paper proposes a human motion synthesis framework based on hierarchical learning recurrent neural networks (HL-RNN). The framework includes a low-level network and a high-level network, which are used to extract the path information of the movement and the spatio-temporal relationship of the human bone structure, respectively. Then, after fusion, motions that satisfy the path constraints could be generated. This method can not only synthesize high-quality human movements that follow a specified trajectory, but also synthesize smooth transitions between various movements, and can also be used to synthesize data of different motion styles. Compared with some latest methods, experiments showed that the proposed method can significantly improve the quality and generalization performance of motion synthesis.

Keyword Hierarchical learning · Recurrent neural network · Motion synthesis · Motion transition

1 Introduction

The three-dimensional human motion capture device is a high-tech device for accurately measuring the movement of the human body in three-dimensional space. The device mainly employs the multi-video principle and computer graphics technology to obtain three-dimensional dynamic data of human joint points, and then uses the physiological topological structure of the human body to reconstruct the human motion data set. This data set has a wide range of application values, and can be widely used in computer animation [1], virtual reality [2, 3], security monitoring [4, 5], human–robot interaction [6, 7] and other fields related to human motion analysis. On the other hand, the contradiction between the personalization of human motion and the need for data commonality limits the reusability of human motion data. Due to economic, time and other factors, the acquisition of new motion data requires high costs, and it is easy to further cause a lot of redundancy of the same type of data. Therefore, how to effectively reuse existing motion data has become one of the key issues urgently to be solved in academic and engineering fields.

Human motion synthesis technology uses existing dataset to synthesize a variety of new motions that meet demand. This technology can not only solve the data reuse problem mentioned above, but also break through the hardware barriers in many fields related to human motion analysis [8]. At the same time, the technology has also emerged in some typical representative fields of AI, such as natural human–robot interaction [9–11] and autonomous driving [12, 13]. As one of the basic supporting technologies for machine understanding, analysis, and prediction of human behavior, this technology has shown a wider and more vital research and application value, and has attracted more and more attention from researchers.

However, the existing human motion synthesis methods still have the challenge of generalization performance and synthesis accuracy to be improved, and it is difficult to

✉ Dongsheng Zhou
zhouds@dlu.edu.cn

[1] Key Laboratory of Advanced Design and Intelligent Computing (Ministry of Education), School of Software Engineering, Dalian University, Dalian 116622, China

[2] School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China

[3] School of Information Science and Technology, Taishan University, Shandong, Tai'an 271000, China

smoothly transition different types of motions. This paper proposes a multi-purpose human motion synthesis framework to solve the above problems. In summary, the main innovations of this article are:

This paper proposes a multi-purpose general human motion synthesis method, which can,

1. Synthesize a human motion sequence that follows the user's input trajectory.
2. Synthesize a smooth transition between two different types of motion sequence.
3. Able to motion style conversion.

The rest of this paper is organized as follows. In Sect. 2, some related works are briefly introduced by category. In Sect. 3, the motion synthesis algorithm based on HL-RNN is proposed. In Sect. 4, experiments and analysis are illustrated. Finally in Sect. 5, the conclusion of this paper is summarized. Also, some challenges and future work are presented.

2 Related work

Due to high research difficulty, practicality and commercial value, the technology of human motion synthesis has become one of the focuses, both in academic and application fields. This section outlines some representative methods of motion synthesis technology, mainly including three types: optimization-based methods [14–17], deep-learning-based methods [18–27], and reinforcement-learning-based methods [28–30]. The details are as follows:

2.1 Optimization-based methods

Synthesizing new movements that satisfy constraints and control conditions is one of the most important goals of motion synthesis technology. The quantitative measure is the minimum error between the composite value and the reference value. That is, from a mathematical point of view, the motion synthesis problem can be converted into an optimal value solving process that takes into account various constraints. Although the optimization-based method belongs to the traditional idea, in recent years, researchers have still proposed some representative new methods.

Alla et al. [14] improved the optimized search algorithm of motion maps, and proposed a suboptimal solution method, which improved the search efficiency of the algorithm and the fidelity of the synthesized motion. Levine et al. [15] used a connected prior Gaussian process latent variable model to learn the transition between different motion sequences. According to the control signal, the model can synthesize actions such as walking and hitting. However, due to the complex modeling process, this model is not suitable for processing large motion data sets. Mahmudi et al. [16] proposed a multimodal search tree method which can control the character to generate various tasks, such as opening door and taking book from bookshelves. Kang et al. [17] used the Gaussian process to estimate the end effector position of the character in the motion graph, and plan the next motion trajectory. This method can generate the climbing locomotion of the limbs contacting the object, while automatically avoid obstacles.

2.2 Deep-learning-based methods

Deep learning is based on data-driven neural networks suitable for processing massive amounts of complex data. Logically speaking, deep learning is very suitable for processing human motion data which has the characteristics of high dimensions, complex nonlinear dynamics, large amount of data and so on. In fact, using deep learning techniques to construct human motions has become one of the current research hotspots.

In recent years, Holden et al. [18] have a few outstanding contributions in motion synthesis and editing, and proposed a motion editing and synthesis framework based on manifold learning and convolutional neural networks. The network used manifold learning to extract low dimensional spatial information of motion data and edited motion in the hidden layer. At the same time, the feedforward neural network mapped the advanced control signals into the manifold space, generated actions that followed the specified trajectory, and completed the style conversion. Then, Holden et al. [19] proposed an impressive work about control motion. They employed the contact information between the footsteps and the ground as a dynamic weight to synthesize the motions that adapted to various terrain conditions in real time. Hwang et al. [20] trained a physical model based on an inverted pendulum, which can simulate human running, walking and other actions in real time under the control of signals. The model has the advantage of interaction with the environment and can avoid obstacles without any other inputs. However, the vividness of the motions synthesized still needs to be further improved. Habibie et al. [21] used a convolutional neural network (CNN) to extract motion control signals. The model combined with a variational encoder structure which can generate motions that satisfy the control signals. However, the process is a little complicated because that the control signals needs to be manually labeled. Li et al. [22] focused on solving the problem of long time motion generation and proposed a method named auto-conditioned recurrent neural network (acRNN) which based on ac-LSTM architecture. Theoretically, the method can generate infinitely long motion sequences. Wang et al. [23] proposed a training model based on a combination of

recurrent neural networks and adversarial learning, which can not only synthesize and control the character's motion path in an online or offline manner, but eliminate the noise in the original motion as well. Harvey et al. [24] focused on the use of neural networks to handle transitional movements, and proposed a recurrent transition network (RTN) based on LSTM network. The model did not require any gait, phase or contact labels, and could generate transition frames only based on the previous motion data and target state. Gopalakrishnan et al. [25] proposed a two-layer GRU network. The top-level network extracts one-hot encoding from the initial samples as the action labels, and the lower-level network combines the labels and the input motion to predict the long-term motion trend of a single action. Battan et al. [26] proposed GlocalNet, which captures the global dependence between different kinds of poses from sparsely sampled poses, and then uses an autoregressive network to learn dense motion trajectories, thereby synthesizing two different kinds of motions. Zhao et al. [27] proposed a novel method of human motion synthesis, which uses the abstract modeling capabilities of Bayesian networks and combines the generation of adversarial networks to synthesize motion data. The Bayesian network makes up for the shortcomings of the generative adversarial network that is easy to overfit and rely on data-driven. At the same time, because the Bayesian network parameterizes the motion data, the network only needs a small amount of training time to synthesize realistic sports. Data, and can repair missing joint data.

2.3 Reinforcement-learning-based methods

Reinforcement learning considers the interaction between the agent and the external environment, and uses the interactive feedback to enhance its own learning ability, which has a very broad application prospect. By receiving feedback from the environment, it can interact with the actual environment in real time and quickly learn tasks to meet the goal. Therefore, reinforcement learning methods are also widely used in human motion synthesis considering intelligent interaction.

Peng et al. [28] proposed a two level reinforcement learning model. This model combines high level navigation function with low level walking and balance control targets. The biped physical character generated by this model can pass through the obstacles, independently plan paths, and complete target navigation tasks. However, the posture of the physical model is quite different from a real person and needs to be improved. Then, Peng et al. [29] combined data-driven skeletal motion with reinforcement learning physics models to drive the model to learn motion clip data, which could imitate a broad of example motion clips and generate realistic motion. Merel et al. [30] proposed a multilayer reinforcement learning model based on adversarial imitation. The method learned motion capture sequences by constructing a discriminator, which interacted with the environment as a reward function of the network. Using this method, the agent could complete various special tasks, such as kicking a ball, avoiding obstacles.

In summary, although the optimization based methods can synthesize motions that satisfied the constraints, the modeling process is complex and it is difficult to handle large dataset. Reinforcement learning can interact with the external environment, but they are still limited by complex modeling processes and can only generate limited types of motion. In contrast, deep learning-based methods can handle large data sets with various forms of motion, and can encode complex motion data into small, fixed-size networks. The above advantages have gradually made it a research focus in the field of motion synthesis.

3 Method

This paper proposes a human motion synthesis framework method based on hierarchical learning recurrent neural network (HL-RNN). The method defines a synthesis task as a sequence of movements of a character given a trajectory and a speed of movement. The difference is that we divide the synthetic motion path and the synthetic action sequence into two parts, the purpose is to receive the user's input path and synthesize a motion sequence that meets the user's input. And, the whole pipeline of the method is shown in Fig. 1, in which consists of two parts. The structure in the red dash line box is the first part of the framework, which is the low level motion information extraction network. The structure in the yellow dash line box is the second part of the framework, which is a high level motion synthesis network. More details of the two parts are as follows.

3.1 Low level motion information extraction network

The function of this part of the network is to extract the motion trajectory parameters. The motion trajectory parameters are used as extra information to synthesize the character's motion and control the motion path. The network consists of two layers of GRU [31] units and a fully connected layer.

The network represents the character trajectory information as a piecewise linear spline [32]. The network receives the curvature and average speed $\{c_i, b_i\}$ of each frame as input, and extracts parameters related to the motion path on the trajectory through the network. The motion parameters of the frame i can be expressed as $e_i = \{d_i, f_i, q_i, g_i\}, e_i \in R^6$. During training, the motion parameters of each frame of the skeleton data need to be pre-calculated. The process is as follows.

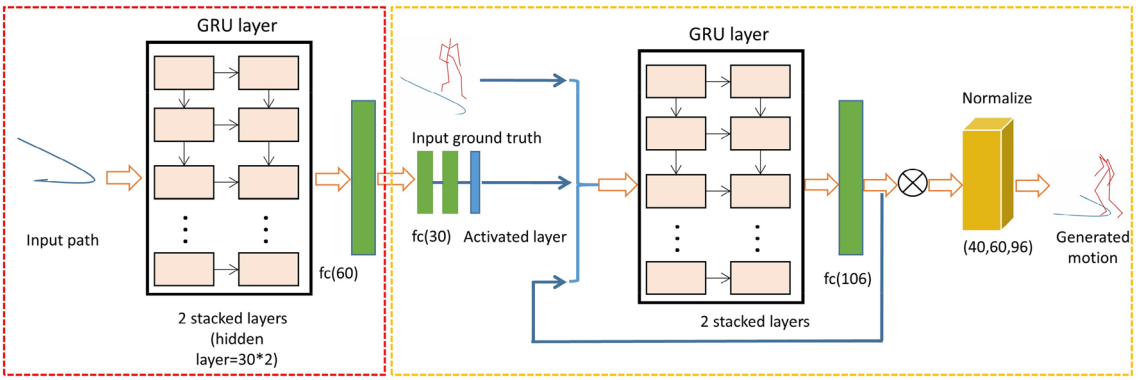

Fig. 1 The structure diagram of hierarchical learning recurrent neural network (HL-RNN) for human motion synthesis

First, define the function as equation below:

$$f(X) = \tan^{-1}\left(\frac{x_2}{x_1}\right) \tag{1}$$

where, $X = (x_1, x_2), X \in R^2$.

$$q_i = \frac{p_{i+1} - p_i}{\|p_{i+1} - p_i\|_2} \tag{2}$$

where, q_i represents the offset of the character in frame i, $q_i \in R^2$. p_i is the world coordinate position of the x and y axis of the character's root joint at frame i, $p_i \in R^2$.

$$c_i = f(q_i) \tag{3}$$

where, c_i is the curvature feature for input.

$$s_i = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{\|p_{i+1} - p_i\|_2^2}{2\sigma^2}) \tag{4}$$

where,s_i represents the instantaneous velocity of the character's root joint after passing the Gaussian filter.

$$b = \frac{\sum_1^L \|s_i \cos\theta_i, s_i \sin\theta_i\|_2}{L} \tag{5}$$

where, b represents the average speed of a character. L is the length of a motion sequence. θ_i represents the contact information of the foot joints, and we defined that $\theta_i = 2\pi$ when the left foot touches the ground, and $\theta_i = \pi$ when the right foot touches the ground.

$$d_i = \{\cos\delta_i, \sin\delta_i\} \tag{6}$$

where, d_i indicates the direction of the character's movement in each frame, δ_i is the Euler direction angles of x and y axis.

$$f_i = \|s_i \cos\theta_i, s_i \sin\theta_i\|_2 \tag{7}$$

where,f_i represents the local velocity feature of the character, which uses the instantaneous velocity of the root joint to calculate the motion characteristics of the footstep.

$$\beta_i = \frac{\{s_i \cos\theta_i, s_i \sin\theta_i\}}{\|s_i \cos\theta_i, s_i \sin\theta_i\|_2} \tag{8}$$

where,β_i represents motion parameters of a complete sequence.

$$g_i = f(\beta_{i+1}) - f(\beta_i) \tag{9}$$

g_i represents the step frequency of a character calculated by using the difference method.

$$loss(e_i, \hat{e}_i) = \frac{1}{n}\sum_{i=1}^{n} |e_i - \hat{e}_i| \tag{10}$$

where, $\hat{e}_i$ represents the predicted motion parameters, and e_i represents the real motion parameters. As mentioned above, during the training process, the network took the curvature and average speed of each step $\{c_i, b_i\}$ as inputs to predict the motion parameter e_i at each moment of the trajectory. By comparing $\hat{e}_i$ and e_i, the back-propagation algorithm [33] was employed to gradually optimize the prediction error of the network. Inspired by Ref. [34], the loss function can be defined as the Mean Absolute Error (MAE), as shown in formula (10). The smaller the value of *loss*, the closer $\hat{e}_i$ to e_i, that is to say, the extracted motion parameters will be more accurate.

3.2 High level motion synthesis network

The second part of the network is a high level motion synthesis network, as shown by the yellow box in Fig. 1. A two layer GRU network was used to synthesize the motion sequence. Similar with [35], the network transformed the skeleton information represented by the three dimensional rotation into quaternion space. The ith frame skeleton

information could be expressed as x_i, where $x_i \in R^4$, which represented the skeleton rotation of the quaternion space. The input control parameters could be expressed as E_i, where $E_i \in R^6$. E_i is the control parameter after the output of the low level network and calculated by the high level network. The calculation process is shown as follows:

$$\begin{cases} E_i^1 = s_i \cos\theta \\ E_i^2 = s_i \sin\theta \\ E_i^3 = s_i q_i^0 \\ E_i^4 = s_i q_i^1 \\ E_i^5 = \cos\delta_i \\ E_i^6 = \sin\delta_i. \end{cases} \tag{11}$$

Among them, E_i^1 represents the first feature in the ith frame control parameters, and θ is the contact information of the foot joint.

The network also extracted the skeleton height and average speed at the previous moment as additional transformation parameters expressed as T_i, where $T_i \in R^2$. The skeleton rotation, control parameters and transformation parameters were input into the GRU network. Then, the motion sequence could be synthesized following the input path. The process can be expressed as follow:

$$x_{k+1} = P(\{x_1, E_1, T_1\}, \{x_2, E_2, T_2\}, \ldots, \{x_k, E_k, T_k\}, \theta) \tag{12}$$

where, θ represents the weight parameters of the network.

In the training stage, the control parameters for each frame of motion in the data set were first calculated, and merged to the input skeleton information. Then, based on the existing motion of k frames, a new motion of n frames could be synthesized, where the values of k and n are not necessarily equal, depending on the specific task. So, an input with motion of length $k + n$ can be expressed as:

$$x_{input} = (\{x_1, E_1, T_1\}, \{x_2, E_2, T_2\}, \ldots, \{x_{k+n}, E_{k+n}, T_{k+n}\}). \tag{13}$$

The output motion can be expressed as:

$$y_{output} = (x_{k+1}, x_{k+2}, \ldots, x_{k+n}). \tag{14}$$

In the final, the network back-propagation algorithm [33] was used to further reduce the error between the output motion and the ground truth.

In the running stage, the flow of the algorithm for synthesizing motion sequences was the same as the training stage. It is worth noting that because the network provides additional control parameters at each step, it can overcome the problem of motion divergence caused by error accumulation.

4 Experiments and analysis

In order to verify the performance of the proposed method, four sets of experiments were designed to evaluate the quality and diversity of synthetic motion. The first set of experiments aimed to evaluate the accuracy of extracting motion information from the low level network using four motions: kick, punch, wheel and jog. In the second set of experiments, the joint position errors and animation effects of the four motion sequences were evaluated quantitatively and qualitatively. The third set of experiments demonstrated that the method could generate realistic and smooth transition animations by synthesizing diversified transition motion under the given path. The fourth set of experiments focused on verifying the potential application direction, and the proposed method was mainly used for transfer the motion styles.

4.1 Experimental setup

Our experiments used the dataset from Ref. [18], which contains multiple online large-scale motion databases, including various sequences of running, walking, kicking, rolling, etc. Similar to the pre-processing method in Ref. [34], we first normalized the data, that is, relocated the joint points to a skeleton structure with a uniform scale and joint length. Then, the dataset was down-sampled to 30 Hz, and the amount of data was doubled by mirroring. Finally, the fixed joint points were removed, ensuring that each data has 26 joint points. The Pytorch 1.0 framework was used as the software support platform. One NVIDIA Quadro K6000 was used as the core computing hardware which has 12 GB memories and 384bit memory bandwidth.

4.2 Network parameter setting

The motion information extraction network uses a two-layer GRU network with 128 hidden units. The learning rate of the network is 0.001, the decay learning rate of each generation is set to 0.999, the batch size is 40, and the number of network iterations is 8000 generations.

Inspired by Ref. [36], and after experiments, we found that compared to the LSTM network recently most used in human motion data processing, the GRU calculation cost is lower, but as the number of network layers increases, the GRU calculation overhead will increase significantly, and does not bring better synthesis results with the increase of network depth. The experimental results are shown in Fig. 2. We compared the effects of different layers of motion synthesis networks on the results. The experiment uses different number of layers to train the network, and compares the total error between the four motion generated kick, punch,

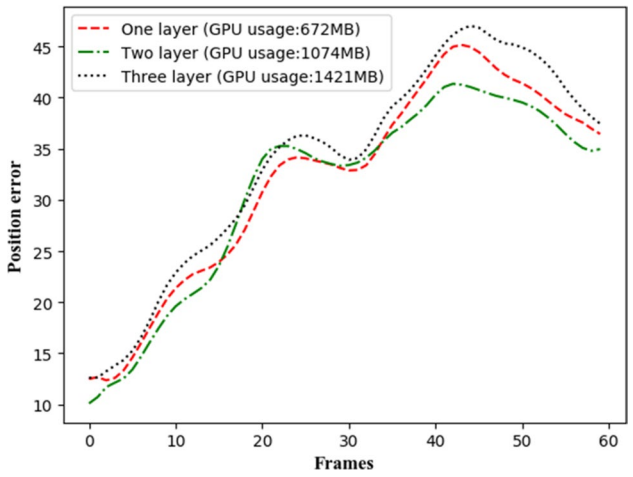

Fig. 2 Comparison of training result of different layer networks from 1 layers to 3 layer

wheel, and jog with the ground true and the GPU load during training. Experimental results show that stacking multi-layer GRU networks cannot improve the performance of the network, and multi-layer networks lead to excessive training parameters, increasing GPU load and training time. Therefore, the synthetic network in this paper uses a two-layer GRU network with 1024 hidden units. The learning rate of the network is 0.001, the batch size is 40, and the number of network iterations is also 8000 generations. Both networks use Adam [37] optimizer to optimize the training process.

4.3 Experimental results

1. Motion information extraction

   Based on our method and the method in Ref. [34], the experiments were first performed to analyze and compare the accuracy of the extracted motion information. The f-parameter of motion features $e_i = \{d_i, f_i, q_i, g_i\}$ was selected for comparison experiments, which could well describe the dynamic performance of the generated motion. It could comprehensively reflect the order in which the motioned feet touch the ground and the speed characteristics of each frame of the character. And, four representative motions were used, including kick, punch, roll and jog. The results are illustrated in Fig. 3.

   The figure above shows the accuracy curve of footstep speed characteristics obtained by using low-level information extraction network. The red curve is the result by using our method, the black one is based on Ref. [34], and the green one represents the ground truth. It is easy to see that the red curve is closer to the curve of the ground truth. That's means our method has lower error, better effect, and can better reflect the dynamic characteristics of character movement. More details of error values are shown in Table 1. We compared the

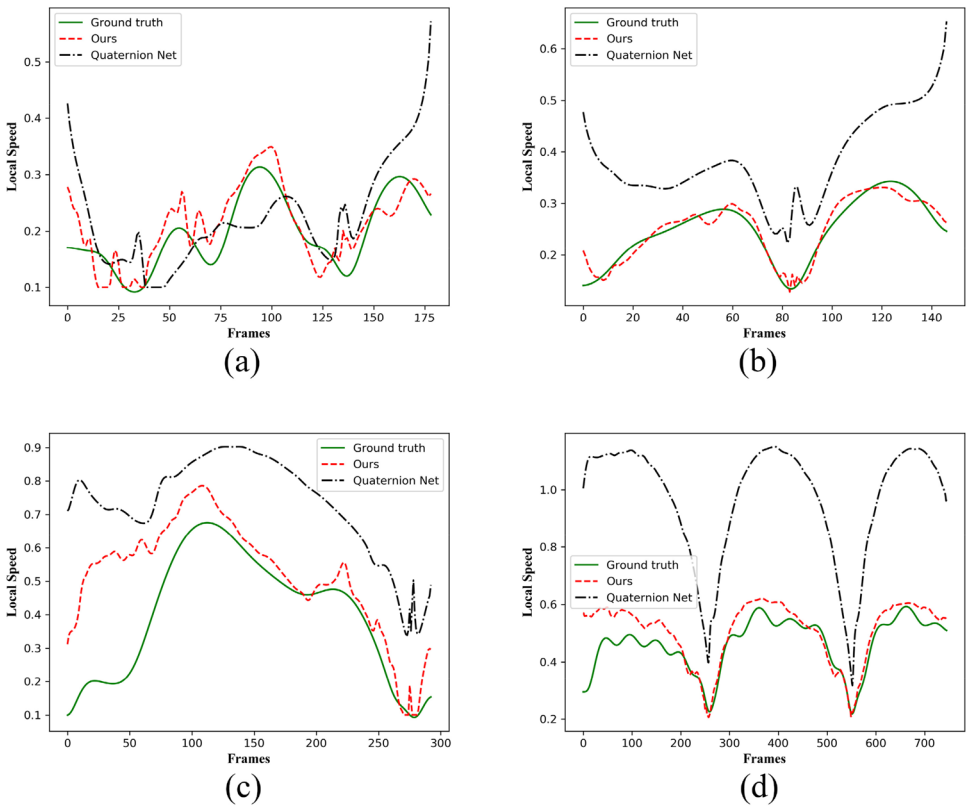

Fig. 3 Comparison of accuracy of low level network extraction of motion information, **a** kick, **b** punch, **c** wheel, **d** jog

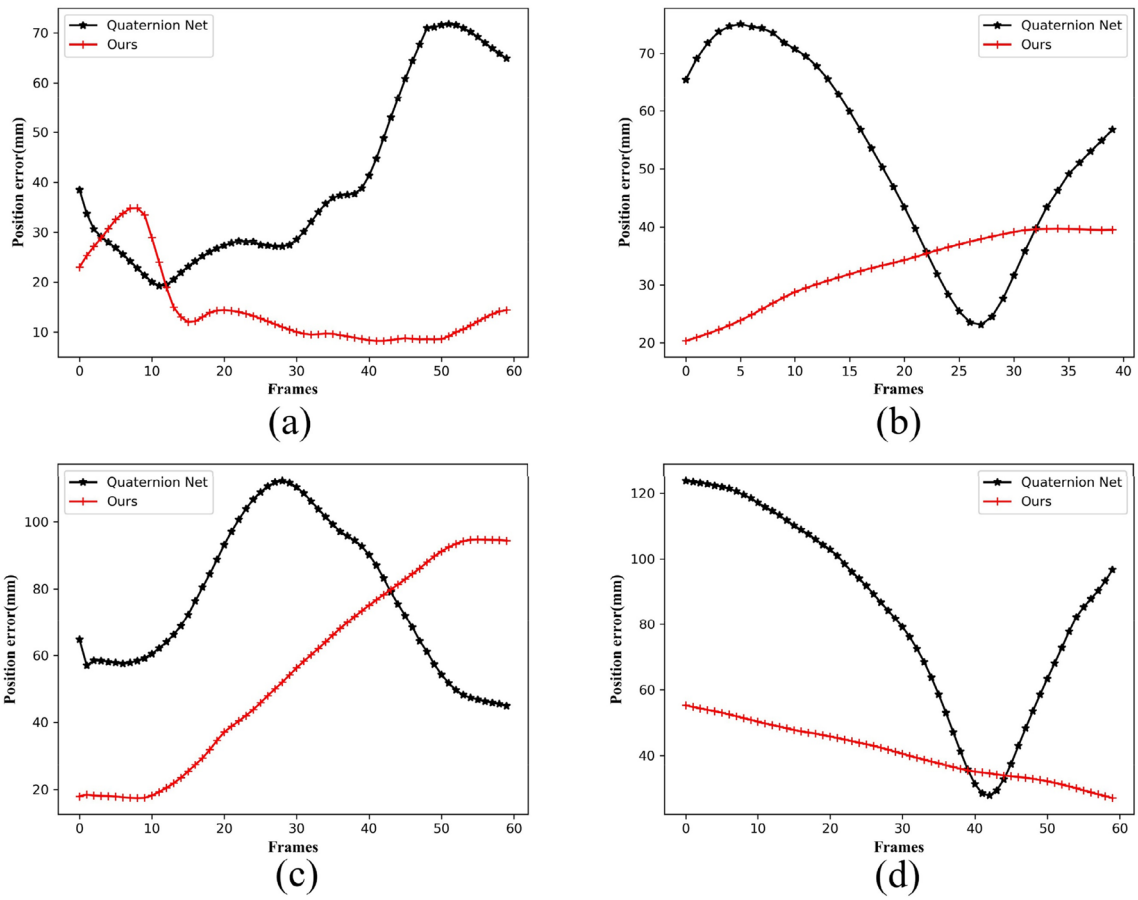

Fig. 4 Comparison of errors in synthetic motion of high level network, **a** kick, **b** punch, **c** wheel, **d** jog

errors of the motion features extracted within 60 frames using the above two methods. The bold values in black represent smaller error values. As can be seen from the Table 1, most of the motion features extracted by our method have smaller errors.

2. Motion synthesis

Next, the joint position errors of the four synthesized motions (kick, punch, roll and jog) were compared experimentally. The generated motion sequences were compared with the real joint positions in the database, and the position error of each frame could be calculated, as shown in Fig. 4.

Experiments have found that for movements with relatively small changes in character displacement, such as kick and box, our method could generate a motion sequence with a high degree of fit to the original motion

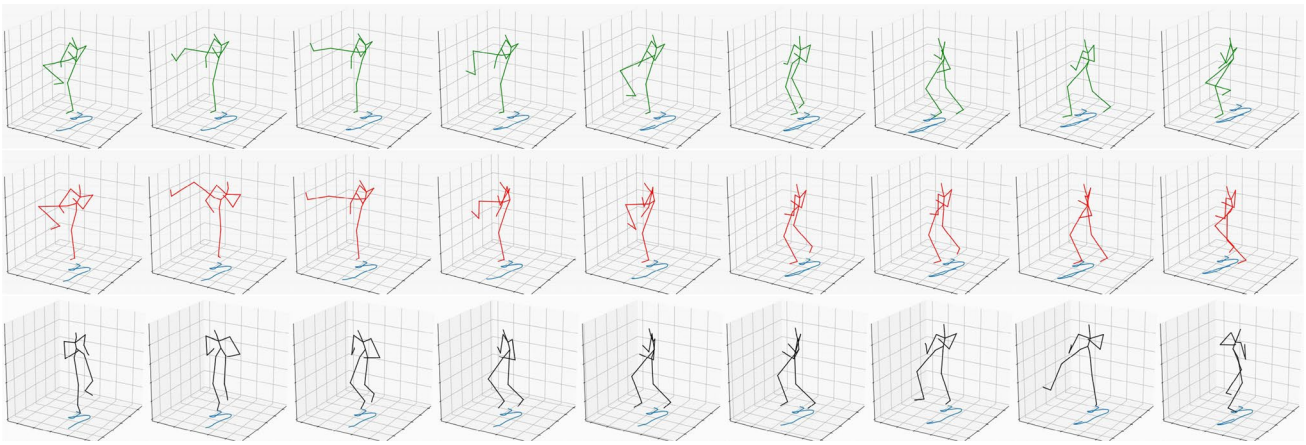

Fig. 5 Animation effect of synthetic kick motion

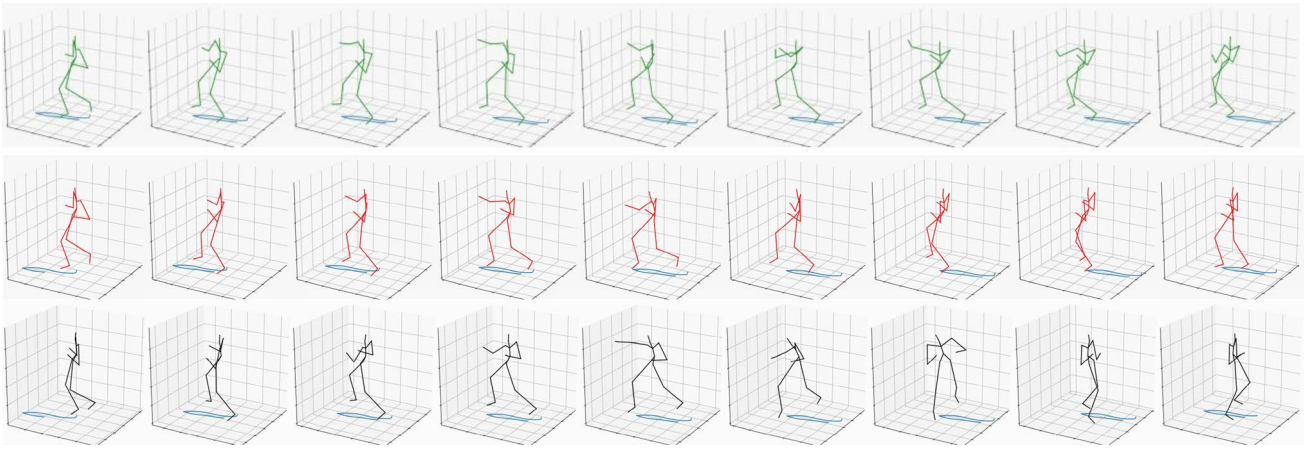

Fig. 6 Animation effect of synthetic punch motion

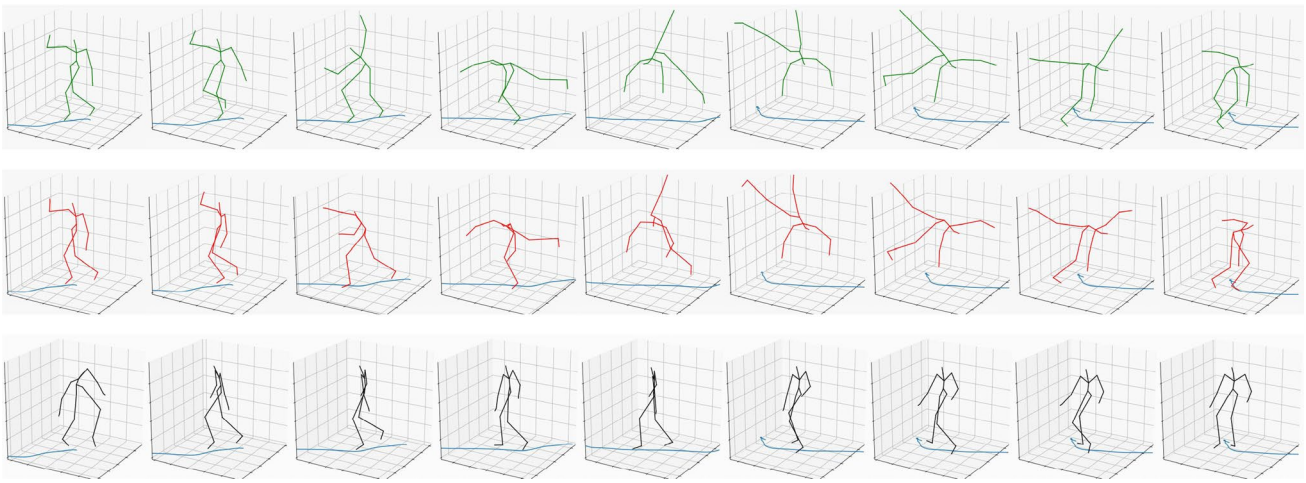

Fig. 7 Animation effect of synthetic wheel motion

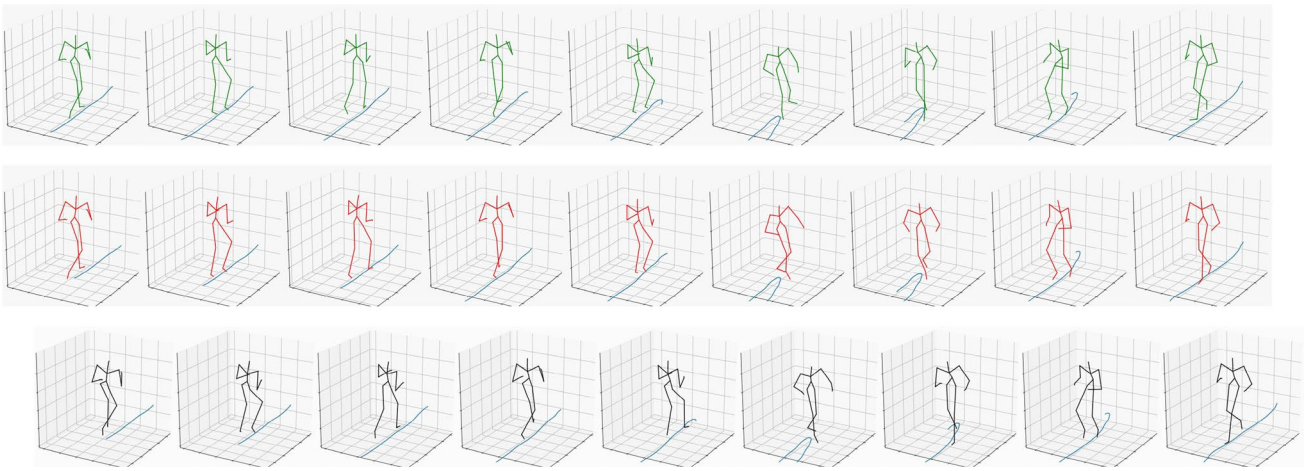

Fig. 8 Animation effect of synthetic jog motion

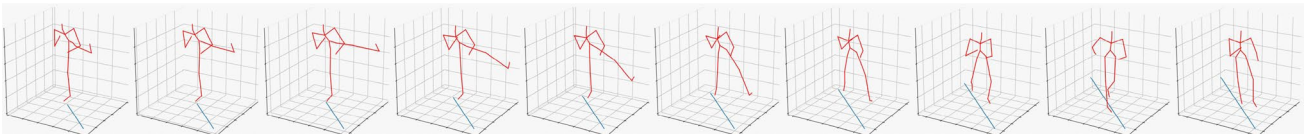

Fig. 9 Effect of synthetic kick to run motion transition

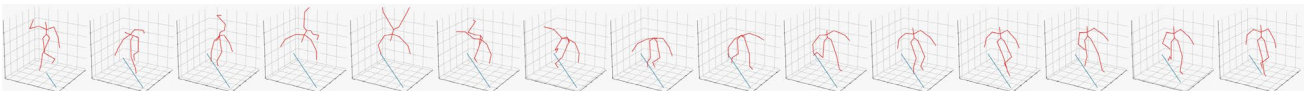

Fig. 10 Effect of synthetic wheel to run motion transition

data. However, the same types of motions synthesized by the method in Ref. [34] have irregular rotation and sliding steps, resulting in insufficient smoothness and fluency of the whole motion. The comparisons of visualized animation sequences are shown in Figs. 5 and 6. The compared animations have a same timeline, and the first row of green is the real motion sequence, the second row of red is the motion sequence generated based on our method, the third row of black is the motion sequence generated based on the method in Ref. [34]

As shown in Fig. 7, for the types of motion with relatively large amplitude and complexity, such as wheel, our method could also generate motion data with a high degree of fit to the original data. However, the method based on Ref. [34] did not perform well, and the generated motion appeared ambiguous. The input raw data is a wheel, but the generated motion is a jog.

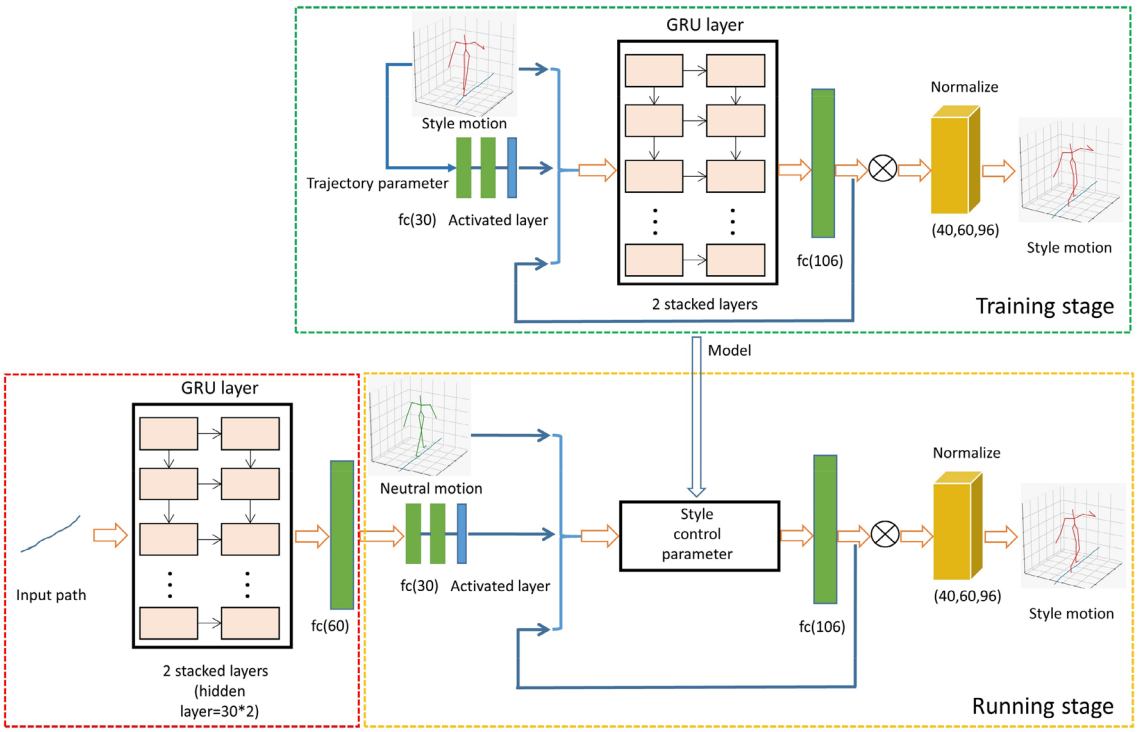

Fig. 11 Architecture of motion style transformation. The green dashed box in the upper part is the training process of the network, the red dashed box in the lower part is used to extract the trajectory input by the user, the yellow box is the test part, and the blue arrow indicates the trained network parameters

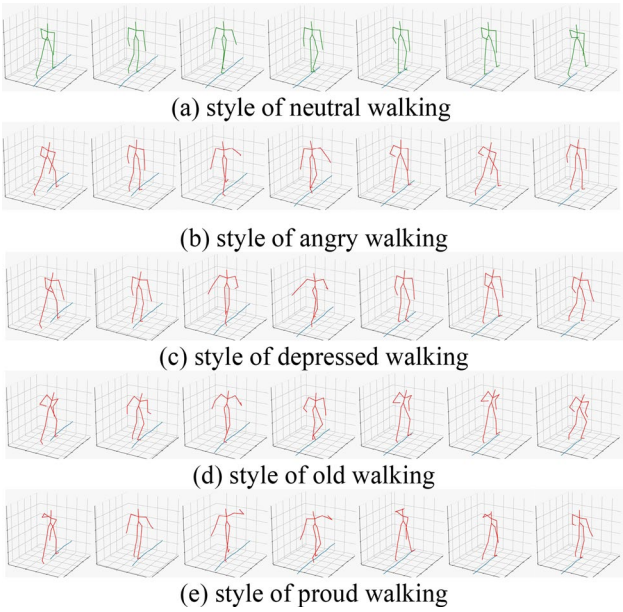

Fig. 12 Some synthetic motions with different emotional styles

In addition, for motions with smaller amplitude and complexity, such as the jog, as shown in Fig. 8, both our method and Ref. [34] could generate smooth motions. However, our method has a higher degree of fit with real values, lower joint errors, and is closer to the characteristics of real motions.

3. Transitional motion generation

Based on the proposed method, we also conducted a combination experiment based on two different types of motions, the core of which was to generate data in the transition phase.

Monotonic regular movements such as walking and running were defined as simple forms of movement, while violent and changeable movements, such as wheeling, jumping, kicking, etc., were belong to complex forms. Combining simple and complex forms of motion, whether the synthetic transition motion is natural, real and smooth is one of the research difficulties. We performed two sets of experiments. The first set was a combination of kicking and running, as shown in Fig. 9. The second is a combination of wheeling and running, as shown in Fig. 10. Experiments show that our method performs well, and the resulting transitional motion is natural and smooth.

In the transition animation generated from the kicking motion to the running motion, the model could automatically generate a smooth transition frame without any manual editing process, as shown in the black box of Fig. 9. In the transition animation from the rolling to the running movement, the generated transition frames show the stumbling state of the human body due to standing instability after somersaults. This action vividly reflects the process of the body's spontaneous adjustment of the balance in motion from the details, as shown in the black box of Fig. 10. It is worth mentioning that the above synthetic actions are not included in the existing data set, and are entirely new actions generated based on our proposed method.

4. Motion styles

We also tried to use the presented method for motion style conversion. We defined the task to generate an emotional motion from a neutral motion under a given motion trajectory.

The whole detailed architecture of the method is shown in Fig. 10. The training phase is shown in the green box part of Fig. 11. We use emotional walking motion data as the training set to train high-level motion synthesis networks to synthesize motion sequences with emotions. The lower part in Fig. 11 is the testing phase. In the testing phase, the user can input the specified motion trajectory, and the motion trajectory is extracted by the low-level network and then input to the high-level network. Then, the normal emotional motion data is combined with the trajectory parameters extracted by the low-level network into the trained high-level network, and the network can synthesize a motion sequence with emotion that moves according to a predetermined trajectory.

The low-level network was still used for extracting the parameters of motion trajectory. The high-level network was used for training the weight parameters which could reflect the skeleton spatio-temporal relationship information of style movements. In the running stage, the high-level network takes the neutral prefix length as input, and combined the trajectory parameters from the low-level network to synthesize motions with different emotional styles. The control parameters have a restraining effect on the high-level network when generating motion, so our method could avoid the accumulation of errors during synthesis, and could suppress the undesirable tendency of motion freezing or divergence. In addition, due to the full use of the RNN's memory, compared with the CNN based method [38], our method does not need a style loss function to constrain the emotional style features of the generated motion, thereby could reduce the complexity of the network.

Based on the motion data with neutral emotion, the network can generate emotional data such as angry walking style, depressed walking style, old walking style and proud walking style, as shown in Fig. 12.

5. Ablative study

In our network, the input control parameters of the high-level network are one of the important factors that determine the network synthesis performance. There-

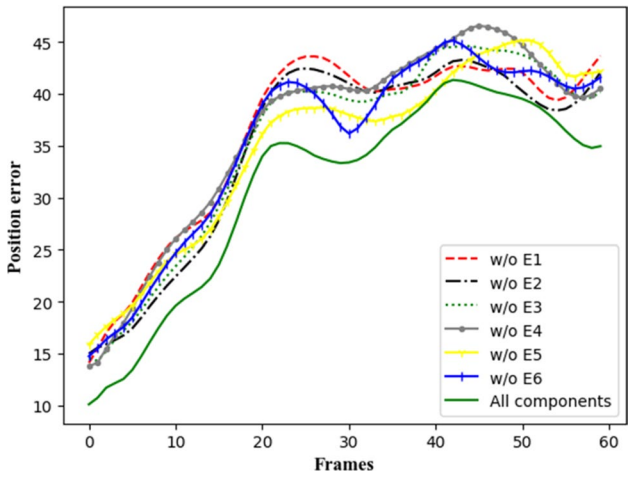

Fig. 13 The position error comparison of the motion generated after ablation of different control parameters. "w/o" means to ablate the corresponding components, "all components" means complete control parameters

fore, we verify the rationality of the network parameter settings by ablating different control parameters. The experimental results are as follows:

In the ablation test, any one of the control parameters in the high-level motion synthesis network is sequentially eliminated, and then the average error of the four motion sequences of kick, punch, wheel, jog is calculated to verify the influence of the control parameters on the network performance. In Fig. 13, it can be seen that eliminating any control parameter will result in a larger error in the generated motion. In fact, the lack of any components will cause ambiguity in the motion generated by the network. For example, the punching motion sequence generated in Fig. 14 will be mixed with the kicking motion. This is because the network generated motion requires control parameters to constrain the

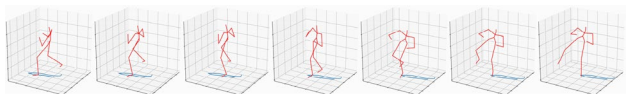

Fig. 14 In the absence of the control parameter E1, the generated punching motion will be distorted into a kicking motion, resulting in ambiguity

motion of the foot. The type of motion generated, the lack of any components will cause the network to misjudge and fail to correctly generate the required motion form.

5 Conclusion and future work

This paper introduces an algorithm for motion synthesis and motion style conversion. The framework adopts the idea of hierarchical learning and builds a model including low-level and high-level networks based on recurrent neural networks. In which, a low-level network is used to extract motion trajectory parameters, and a high-level network is used to learn the spatio-temporal relationship of the skeleton data, and can complete motion synthesis. Experiments show that the presented method can not only be used to synthesize motions that follow an input trajectory, but also can be used to generate transitions between two different types of motions. It can also be used to synthesize motion sequences with different emotional styles. Compared with the existing method, the effectiveness of the proposed method was proved.

Our method also has some shortcomings. For example, a slipping phenomenon may occur when synthesizing a rolling motion, which may be caused by the lack of joint constraints. In the future, in addition to continuing to improve the above-mentioned shortcomings, our work will

Table 1 Comparison of accuracy errors

	Frames					
Methods	10	20	30	40	50	60
Kick						
Ours	**0.008**	0.048	**0.003**	0.025	**0.040**	**0**
Pavllo's	0.088	**0.005**	0.048	**0.008**	0.078	0.038
Punch						
Ours	**0.012**	**0.017**	**0.007**	**0.006**	**0.018**	**0.011**
Pavllo's	0.206	0.121	0.092	0.078	0.083	0.095
Wheel						
Ours	**0.252**	**0.348**	**0.373**	**0.393**	**0.358**	**0.326**
Pavllo's	0.647	0.557	0.522	0.522	0.478	0.377
Jog						
Ours	**0.258**	**0.208**	**0.150**	**0.114**	**0.107**	**0.089**
Pavllo's	0.798	0.761	0.685	0.644	0.640	0.654

focus on the interaction between motion and the environment, that is, the constraints imposed by the environment (such as terrain and obstacles) will be fully considered to synthesize the corresponding motion.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s13042-021-01304-w.

Acknowledgement This work was supported by the Key Program of NSFC (Grant No. U1908214); Special Project of Central Government Guiding Local Science and Technology Development (Grant No. 2021JH6/10500140); the Program for the Liaoning Distinguished Professor; the Program for Science and Technology Innovation Fund of Dalian (Grant No. 2020JJ25CY001); the Project for Technology Innovation Team of Liaoning Province and Dalian University.

References

1. Lee K, Lee S, Lee J (2018) Interactive character animation by learning multi-objective control. ACM Trans Graph 37(6):1–10
2. Oh J, Lee Y, Kim Y, Jin T et al (2016) Hand contact between remote users through virtual avatars. In: Proceedings of the 29th International Conference on Computer Animation and Social Agents, pp 97–100
3. Cao Z, Gao H, Mangalam K et al (2020) Long-term human motion prediction with scene context. In: Proceedings of European Conference on Computer Vision, pp 387–404
4. Jain A, Zamir AR, Savarese S et al (2016) Structural-RNN: deep learning on spatio-temporal graphs. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 5308–5317
5. Adeli V, Adeli E, Reid I et al (2020) Socially and contextually aware human motion and pose forecasting. IEEE Robot Autom Lett 5(4):6033–6040
6. Gui L, Zhang K, Wang Y et al (2018) Teaching robots to predict human motion. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 562–567
7. Ding W, Hu B, Liu H et al (2020) Human posture recognition based on multiple features and rule learning. Int J Mach Learn Cybern 11(11):529–2540
8. Klaus F (2015) From motion capture to performance synthesis: A data based approach on full-body animation. Aalto University publication series Doctoral Dissertations
9. Butepage J, Kjellstrom H, Kragic D (2018) Classify, predict, detect, anticipate and synthesize: Hierarchical recurrent latent variable models for human activity modeling. CoRR
10. Wang Y, Che W, Xu B (2017) Encoder–decoder recurrent network model for interactive character animation generation. Visual Comput 33(6–8):971–980
11. Ondras J, Celiktutan O, Bremner P, Gunes H (2020) Audio-driven robot upper-body motion synthesis. IEEE Trans Cybern. https://doi.org/10.1109/TCYB.2020.2966730
12. Du X, Vasudevan R, Johnson-Roberson M (2019) Bio-LSTM: A biomechanically inspired recurrent neural network for 3-D pedestrian pose and gait prediction. IEEE RA-L 4(2):1501–1508
13. Kim W, Ramanagopal MS, Barto C et al (2018) PedX: benchmark dataset for metric 3D pose estimation of pedestrians in complex urban intersections. IEEE Robot Autom Lett 4(2):1940–1947
14. Safonova A, Hodgins JK (2008) Artificial Intelligence Techniques for Computer Graphics. Springer, Berlin, Heidelberg
15. Levine S, Wang JM, Haraux AZ et al (2012) Continuous character control with low-dimensional embeddings. ACM Trans Graph 31(28):1–10
16. Mahmudi M, Kallmann M (2015) Multi-modal data-driven motion planning and synthesis. In: Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games, pp 119–124
17. Kang C, Lee S (2017) Multi-contact locomotion using a contact graph with feasibility predictors. ACM Trans Graph 36(2):1–14
18. Holden D, Saito J, Komura T (2016) A deep learning framework for character motion synthesis and editing. ACM Trans Graph 35(4):1–11
19. Holden D, Komura T, Saito J (2017) Phase-functioned neural networks for character control. ACM Trans Graph 36(4):1–13
20. Hwang J, Kim J, Suh IH et al (2018) Real-time locomotion controller using an inverted-pendulum-based abstract model. Comput Graph Forum 37(2):287–296
21. Habibie I, Holden D, Schwarz J et al (2017) A recurrent variational autoencoder for human motion Synthesis. In: Proceedings of 28th British Machine Vision Conference, pp 1–12
22. Li Z, Zhou Y, Xiao S et al (2018) Auto-conditioned recurrent networks for extended complex human motion synthesis. In: Proceedings of International Conference on Learning Representations. arXiv preprint arXiv:1707.05363
23. Wang Z, Chai J, Xia S (2021) Combining recurrent neural networks and adversarial training for human motion synthesis and control. IEEE Trans Visual Comput Graphics 27(1):14–28
24. Harvey FG, Pal C (2018) Recurrent transition networks for character locomotion. In: SIGGRAPH Asia 2018 Technical Briefs, pp 1–4
25. Gopalakrishnan A, Mali A, Kifer D et al (2019) A neural temporal model for human motion prediction. In: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 12116–12125
26. Battan N, Agrawal Y, Rao SS, Goel A, Sharma A et al (2021) GlocalNet: Class-aware Long-term Human Motion Synthesis. In: IEEE Winter Conference on Applications of Computer Vision, January 5–9, Virtual
27. Zhao R, Su H, Ji Q (2020) Bayesian adversarial human motion synthesis. In: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 6225–6234
28. Peng XB, Berseth G, Yin K, Panne MVD (2017) DeepLoco: dynamic locomotion skills using hierarchical deep reinforcement learning. ACM Trans Graph 36(4):1–13
29. Peng XB, Abbeel P, Levine S et al (2018) DeepMimic: example-guided deep reinforcement learning of physics-based character skills. ACM Trans. Graph 37(4):1–14
30. Merel J, Tassa Y, Srinivasan S et al (2017) Learning human behaviors from motion capture by adversarial imitation. arXiv preprint arXiv:1707.02201
31. Cho K, Merriënboer BV, Gulcehre C et al (2014) Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing, pp 1724–1734
32. Stoer J, Bulirsch R (1980) Introduction to Numerical Analysis. Springer, New York
33. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323(9):533–536
34. Pavllo D, Feichtenhofer C, Auli M et al (2019) Modeling human motion with quaternion-based neural networks. Int J Comput Vis 128(4):855–872
35. Pavllo D, Grangier D, Auli M (2018) QuaterNet: a quaternion-based recurrent model for human motion. In: Proceedings of British Machine Vision Conference, pp 188

36. Martinez J, Black MJ, Romero J (2017) On human motion prediction using recurrent neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 2891–2900
37. Kingma D, Ba J (2015) Adam: a method for stochastic optimization. In: Proceedings of International Conference on Learning Representations.arXiv preprint arXiv:1412.6980
38. Holden D, Habibie I, Kusajima I et al (2017) Fast neural style transfer for motion data. IEEE Comput Graphics Appl 37(4):42–49