Research on character Action Control Method Based on Multi Phase-Functioned Neural Network and State Machine

Zhe Wang, Xin Lei and Jing Li*
Naval University of Engineering Ordnance Engineering College Naval University of Engineering Wuhan, China
*Corresponding author’s e-mail: heeu8@tzxrlzy.com

Abstract. Aiming at the situation that the phase-functioned neural network can’t produce natural action or interact with the world poorly in some specific scenes, this paper proposes to divide different states according to the phase information of characters, and train several independent networks with different weights to predict the next frame state of characters. The experimental results show that, while the complexity of the model is reduced, the action generated by this method is more natural and fluent, which has good application value.

1. Introduction
With the development of virtual reality technology, virtual human is more and more widely used in military, manned spaceflight, games and other fields. The character control of virtual human involves the requirements of computer graphics, kinematics and dynamics, artificial intelligence and other disciplines. Because the control of virtual human needs to meet the requirements of real-time, realistic, smooth and so on, the real-time control of virtual character has always been a hot topic and challenge in the field of study.

The real-time control of virtual character has always been a hot topic and challenge in the field of study. The character control of virtual human first needs to have a large number of high-quality motion capture data as the basis, and then the application of efficient algorithms to make virtual human move according to the trajectory. Because of the large amount of data, when the real-time performance of the algorithm is not high, the output action may be stiff and unnatural. Therefore, domestic and foreign scholars have conducted a lot of research on the application of deep learning in the extraction of character-related information and automatic action generation, and achieved a lot of practical results. Taylor[1], Fragkiadaki[2] et al, used autoregressive models to predict character poses. Although this kind of method is more extensible and efficient than the traditional method, it also has the problem of drift. Holden[3] et al, applied convolutional neural network to map the character information to the whole body movement, but this kind of method has a modest effect when applied to the scene where the real-time character control is related to the information before and after. In 2017, Holden[4] et al, proposed a phase-functioned neural network (PFNN) to predict the state of the character in the next frame, which improved the real-time and lifelike performance of the character control. However, this method is prone to distortion in high continuous action.

In order to solve the problem of character motion distortion, based on the phase-functioned neural network proposed by Holden et al, this paper uses the state machine to determine the character motion state, divides the character state into multiple parts according to the phase information, and trains multiple independent neural networks with different weights for these parts of the motion cycle. In the
process of character control, dynamically changing the network weight instead of fixing the weight, so that the character control can be smooth and realistic in various environments.

2. Establishment of character model based on state machine

2.1. Character Model
During the establishment of the character model, the main data collected include joint velocity and joint position. Joint velocities and joint positions are obtained by capturing geometric and motion data of the character using motion capture techniques, which capture various data on actual obstacles, springboards and platforms in the laboratory[5-6].

There is an action phase field in each set of character data models. This field is the key of phase-functioned neural network processing and is the input parameter of phase neural network. The phase field can be marked by calculating the speed of the heel and toe joints and observing when the lower limit is reached. In order to ensure the accuracy of phase field, it is necessary to confirm the correctness of field annotation.

2.2. State Machine
Through the character model, the character has different animation that performs different actions in virtual space, such as the character can breathe or sway gently when stopped, and walk when commanded. It is very complex to control these animations, especially for complex environment and multi continuous actions, which can easily cause foot sliding or strange unstable actions. Therefore, this paper uses state machine to determine the status of the character, so that it is convenient for character control.

The state machine is a mechanism for representing the state of a thing. The basic idea is to limit a character to a specific available action at a given moment[7]. These available movements depend on the type of control. Typical movements include stopping, walking, running, jumping, etc. These actions are called states. After the collection is complete, the status of each group of data is marked. In this paper, the types of states include idle, walking, fast walking, jogging, jumping, running, and crawling. Status markers are semantic markers of a character's state. They provide a way to disambiguate confusing actions such as fast walking and jogging tracks, providing additional semantic information about the character.

In the scene, the character may be in a "state" of walking, idle, etc. Usually, a character is restricted from entering another state, rather than being able to directly transition from any state to any other state. For example, run and jump is only done when the character is already running and not standing still, so you can never directly switch from idle to run and jump. The type of selection in which a character can move from the current state to the next state is called a state transition. In short, the set of states, the set of transitions, and the variables that remember the current state form the state machine. The state transition diagram set in this system is shown in Figure 1. In the figure, nodes represent states, and arcs (arrows between nodes) represent transitions.

![Figure 1 Character state transition diagram](image)

3. Multi phase-functioned neural network
For the phase neural network proposed in reference [4], a neural network model needs to be trained in the training, but the number of neurons in the layer of the network model is 512, and the training speed
of the model is slow, and there may be delays at the joints of different types of actions, and the character action distortion occurs in complex environments. In order to make the action of the character better adapt to a variety of complex environment and make the action realistic, this paper draws lessons from the idea of state machine, and defines the character of each action, to ensure that the state of the character does not jump and the training speed becomes faster. On the basis of dividing the action into states, the phase information that directly affects the training character of the model is divided into two stages, and two independent neural networks with different weights are trained respectively for these two states, which can reduce the training burden of each network. During the phase transition, the method of overlapping phase averaging is adopted to make the character's actions more realistic. In different states, the neural networks in each state are used to predict the next frame of the character's state, which makes the change of the character's action more smooth than before.

3.1. Simple phase-functioned neural network

In this paper, the neural network model proposed in literature [4] is called simple phase-functioned neural network, which is a special neural network and can realize real-time character control. The network structure of PFNN is shown in Figure 2.

![Character state transition diagram](image)

The phase-functioned neural network consists of a phase function $\Theta$ and a three-layer fully connected neural network $\Phi$. The phase function $\Theta$ contains four parameters, and its input parameter is phase $p$. The weights of fully connected neural network $\Phi$ are trained using the known input and output data. In the prediction process, the phase cycle function $\Theta$ is first used to update the weight of the three-layer fully connected network, and then the user control information, the character state information of the previous frame and the scene geometry information are input into the network to predict the character action, phase change and trajectory change of the next frame.

In this phase-functioned neural network model, the input is $x_i = \{t_i^p, t_i^d, t_i^r, j_i^p, j_i^v, j_i^a, g_{it}^g, g_{it}^g, g_{it}^g\} \in \mathbb{R}^n$. Among it, $t_i^p \in \mathbb{R}^{2l}$ is the sub-sampling window of the trajectory position of the character on the 2D horizontal plane. $t_i^d \in \mathbb{R}^{2l}$ is the trajectory direction of the character on the 2D horizontal plane. $t_i^r \in \mathbb{R}^{3l}$ is left, right and center three sample points trajectory heights. $t_i^r \in \mathbb{R}^{3l}$ is used to describe the trajectory of a status and other information semantic markup. $j_i^p \in \mathbb{R}^{3l}$ is the joint position of the character in the previous frame. $j_i^v \in \mathbb{R}^{3l}$ is the joint velocity in the previous frame, and $j_i^a$ is the number of joints. When the character is in the four states of static, walking, jogging, and jumping, the semantic markup of other information in $g_{it}^g$ is valid.

The output of the model is $y_i = \{t_{i+1}^p, t_{i+1}^d, t_{i+1}^r, j_{i+1}^p, j_{i+1}^v, j_{i+1}^a, r_{i+1}^v, r_{i+1}^a, p, c_i\} \in \mathbb{R}^m$. Among it, $t_{i+1}^p \in \mathbb{R}^{2l}$ is the predicted trajectory position of the character animation for the next frame. $t_{i+1}^d \in \mathbb{R}^{2l}$ is the predicted trajectory direction of the character animation for the next frame. $j_{i+1}^p \in \mathbb{R}^{3l}, j_{i+1}^v \in \mathbb{R}^{3l}$ and $j_{i+1}^a \in \mathbb{R}^{3l}$ are the joint position, joint velocity and joint angle of the current character relative to the standard animation form, respectively. $r_{i+1}^v \in \mathbb{R}$ is the square root of the x-axis velocity based on the direction facing the character. $r_{i+1}^a \in \mathbb{R}$ is the square root of the z-axis velocity with respect to the face direction.
the square root of the upward angular velocity. $\dot{\theta}_i \in \mathbb{R}$ is the phase shift. $c_i \in \mathbb{R}^4$ is the label for the contact between the character’s feet and the ground.

The mathematical expression of the three-layer fully connected neural network is as follows:

$$\Phi(x; \alpha) = W_2 \text{ELU}(W_1 \text{ELU}(W_0x + b_0) + b_1) + b_2$$  \hspace{1cm} (1)

$$\text{ELU}(x) = \max(x, 0) + \exp(\min(x, 0)) - 1$$  \hspace{1cm} (2)

Among it, $\alpha = \{W_0 \in \mathbb{R}^{k \times n}, W_1 \in \mathbb{R}^{k \times h}, W_2 \in \mathbb{R}^{h \times h}, b_0 \in \mathbb{R}^k, b_1 \in \mathbb{R}^h, b_2 \in \mathbb{R}^n\}$.

In the three-layer fully connected network above, the number of neurons in each layer $h$ is 512, and the activation function $\text{ELU}(x)$ is an exponentially modified linear function.

Phase function $\alpha = \Theta(p; \beta)$, the input of the function has phase $p$ and parameter $\beta$. In this paper, Cubic Catmull-Rom Spline is chosen. By using this function, only four control point parameters $\beta=(\alpha_0, \alpha_1, \alpha_2, \alpha_3)$ are given, and then the input phase $p$ is given, then an appropriate weight can be found in the weight space of the neural network, so as to realize the regression from input parameters to output parameters.

The mathematical expression of the phase function is as follows:

$$\Theta(p; \beta) = \alpha_i + w(-\frac{1}{2}\alpha_{i-2} + \frac{1}{2}\alpha_{i-1}) + w^2(\alpha_{i-2} - \frac{5}{2}\alpha_{i-1} + 2\alpha_i - \frac{1}{2}\alpha_{i+1}) + w^3(-\frac{1}{2}\alpha_{i-2} + \frac{3}{2}\alpha_{i-1} - \frac{3}{2}\alpha_i + \frac{1}{2}\alpha_{i+1})$$  \hspace{1cm} (3)

$$w = \frac{4p}{2\pi} \mod 1$$  \hspace{1cm} (4)

$$k_n = \left\lfloor \frac{4p}{2\pi} \right\rfloor + n - 1 \mod 4$$  \hspace{1cm} (5)

3.2. Multi phase-functioned neural network

3.2.1. The phase division

Based on the simple phase-functioned neural network, the paper extends the simple phase-functioned neural network to multiple phases in order to make the character more fluent and realistic. The proposed multi phase-functioned neural network in this paper consists of two 3-layer fully connected networks and two phase functions. Among it, the phase function is divided into two sections according to the left and right feet. $0-\pi$ is the phase function of the first section, representing the stage when the left foot steps to the landing; $\pi-2\pi$ is the phase function of the second section, representing the stage when the right foot steps to the landing. As shown in figure 3. Set the left foot to phase 0 when it moves off the ground and to phase $\pi$ when it lands. Set the left foot to phase 0 as it moves off the ground and to phase $2\pi$ as it lands. Set the minimum cycle to 0.25s for a standing character, with the phase allowing continuous cycles.

![Figure 3 Phase division diagram](image-url)
Since the task of each neural network is reduced by half, the number of neurons is reduced from 512 to 256, which is only responsible for half the regressive task of the original phase. The network structure of the multi phase-functioned neural network model is shown in Figure 4.

![Figure 4](image_url) Structure of multi phase-functioned neural network

The advantages of this approach are that the complexity of each network is reduced, the number of neurons in the hidden layer is reduced, and the training time of the model is also reduced. However, under some behaviors, the fluency of the character's behavior is improved compared with that before. For this reason, the following methods are adopted during the phase transition.

3.2.2. Phase transition phase processing method

During the transition between the two phases of the phase, the animation is not natural because of the difference in the weight of the neural network, so we adopt the average weighting method, that is, the transition zone is set up at the junction of the two phases. In this paper, the transition is set as: in the upper and lower range of the junction, the average value of the two prediction results should be taken as the output. In this way, although the calculation amount during the transition is increased, the transition of action is more natural and smooth, as shown in Figure 5.

![Figure 5](image_url) Phase transition processing method

The transition model is expressed as follows:

\[
\Phi(x; \alpha) = \begin{cases} 
\Phi^-(x; \alpha) & p \in \left[\frac{1}{16}, \frac{15}{16}\right] \\
\Phi^+(x; \alpha) & p \in \left[\frac{17}{16}, \frac{31}{16}\right] \\
\Phi^-(x; \alpha) + \Theta^+(x; \alpha) & p \in \left(\frac{31}{16}, \frac{1}{16}\right) \text{ or } p \in \left(\frac{15}{16}, \frac{17}{16}\right)
\end{cases}
\] (6)

Among it, the input of phase function \( \alpha=(p, \beta) \) is phase p and parameter \( \beta \).

\( \Phi^-(x; \alpha) = W_2 \text{ELU}(W_1 \text{ELU}(W_0 x + b_0) + b_1) + b_2 \) is the neural network model of the first stage.

\( \Phi^+(x; \alpha) = W_2 \text{ELU}(W_1 \text{ELU}(W_0 x + b_0) + b_1) + b_2 \) is the neural network model of the second stage.
3.3. Network implementation

For the image of ith-frame, variables $x_i$, $y_i$, and phase $p_i$ are stored in the matrix, and all data are normalized and reduced according to the weight. In the actual simulation, all the input variables related to joints are reduced to 0.1 times of the original, so as to reduce the influence of variables, obtain the best results, and make the character more realistic.

When training this network, it is necessary to ensure that the corresponding output variable $Y$ can be generated for the given control parameter $x$ and phase parameter $P$. During training, the phase function parameters and loss function are optimized.

$$\text{Cost}(X, Y, P; \beta) = \| Y - \Phi(X; \Theta(P; \beta)) \| + \gamma \|eta\|$$  \hspace{1cm} (7)

In the above equation, the first term represents the mean variance of the regression results, and the second term is the regularization parameter set to ensure that the weight is not too large. In this paper, $\gamma$ is set to 0.01 and Adam random gradient descent algorithm is used to calculate the derivative of loss function.

4. Experimental results and analysis

4.1. Experimental Environment

The experiment was run in Ubuntu system with two Intel Xeon Silver 4214R CPUs, no GPU. The RAM is 128GB (4*32GB DDR4 2933), and theano is used as the deep learning library built on the network.

4.2. Performance Comparison

The proposed method is compared with character control methods such as simple phase-functioned neural network, cyclic codec network and dynamic regression Gaussian process. In comparison, all methods have been trained and the weights of each network are adjusted to make the same memory footprint. The comparison results are shown in Table 1. In the case of the same memory consumption, the performance has been improved compared with the simple phase-functioned neural network, reducing the sliding phenomenon when the left and right foot phase change, and the character's walking is more natural and realistic.

| Method                        | Training time | Running time | RAM consumption |
|-------------------------------|---------------|--------------|-----------------|
| Multi phase-functioned neural network | 30h           | 0.0008s      | 10MB            |
| Simple Phase-functioned neural network | 25h           | 0.0018s      | 10MB            |
| Cyclic codec network         | 9h            | 0.0009s      | 10MB            |
| Dynamic regression Gaussian process | 1h            | 0.219s       | 100MB           |

4.3. Responsiveness

In order to verify the responsiveness of the algorithm, circular and square walking tracks are created in this paper to test the response effect of the characters, as shown in Figure 6. The simple phase-functioned neural network and multi phase-functioned neural network were tested respectively under various trajectories, and the comparative results were shown in Table 2. Parameters $\delta_v$ and $\delta_d$ represent the velocity deviation and direction deviation between the predetermined trajectory and the actual trajectory of the measured character, respectively.
Figure 6 Sample diagram of predetermined and actual tracks

Table 2 Response parameters of different trajectories and different methods

| Walking Trails | Method                                      | $\delta_v$ | $\delta_d$ | Average Deviation | Deviation Reduction Rate |
|----------------|---------------------------------------------|------------|------------|-------------------|--------------------------|
|                | Multi phase-functioned neural network       | 0.5        | 2.0        | 10.50cm           | 23.36%                   |
|                |                                              | 2.0        | 5.0        | 6.32cm            | 21.30%                   |
|                | Simple phase-functioned neural network      | 0.5        | 2.0        | 13.70cm           | —                        |
|                |                                              | 2.0        | 5.0        | 8.03cm            | —                        |
|                | Multi phase-functioned neural network       | 0.5        | 2.0        | 16.98cm           | 20.56%                   |
|                |                                              | 2.0        | 5.0        | 9.08cm            | 19.29%                   |
|                | Simple phase-functioned neural network      | 0.5        | 2.0        | 21.26cm           | —                        |
|                |                                              | 2.0        | 5.0        | 11.25cm           | —                        |

As can be seen from the above table, after the improvement of the multi phase-functioned neural network in this paper, the average deviation is reduced by about 20% when walking according to the predetermined trajectory, and the accuracy of the real-time control of the characters is significantly improved.

5. Conclusion

Based on the study of simple phase-functioned neural network, the state machine is used to determine the character state and transition rules, and the character state is divided into two parts, and the two neural networks are trained, and the weighted average method is used in the phase transition stage to make the character can be natural and smooth in various environments, which has a good application prospect. The training time of model is longer than that of other methods, so reducing the training time is the next research direction.

References

[1] Graham W Taylor, Geoffrey E Hinton. Factored conditional restricted Boltzmann machines for modeling motion style. In Proc. ICML. ACM, 1025–1032.
[2] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. 2015. Recurrent network models for human dynamics. In Proc. ICCV. 4346–4354.
[3] Daniel Holden, Jun Saito, and Taku Komura. 2016. A deep learning framework for character motion synthesis and editing. ACM Trans on Graph 35, 4 (2016). 138-146.
[4] Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-Functioned Neural Networks for Character Control. ACM Trans. Graph. 36, 4 (2017). 42:1–42:13.
[5] Xu, H.B. (2018) Xuni xuexi huanjing zhong jineng juese kongzhi yu renji jiaohu yanjiu [Research on Intelligent Character Control and Human-Computer Interaction in Virtual Learning Environment]. Chongqing youdian daxue.
[6] Luo, L.F., Zou, X.J., Zhang, C., and Xie, L. (2015). Jiyu shishi shuju de juese yundong jianmo yu fangzhen [Model and Simulation of Virtual Character Based on Real-time Sensor Data-driven]. Xitong fangzhen xuebao, 06(4), 677-681.
[7] Lin, H. & Zhang, X. (2019). Jiyu youxian zhuangtaiji de dianji Huoer chuanganqi guzhang zhenduan yu buchang celue [Finite state machine based fault diagnosis and compensation strategy of motors with Hall sensors]. Dongnan daxue xuebao(Ziran kexue ban), 06.