Proceedings of the 2015
IEEE Conference on Robotics and Biomimetics
Zhuhai, China, December 6-9, 2015

Modeling the stroke process in table tennis robot using neural network

Abstract—**To hit incoming balls back to a desired position, it is a key factor for table tennis robot to get racket parameters accurately. For modeling the stroke process, a novel model is built based on multiple neural networks. The input data for neural networks are the ball velocity differences during the stroke, and racket parameters are the output data. To reduce the influences from the invalid data, a neural network based on each empirical data is established. The training data are clustered based on the empirical data. The way of choosing a neural network to compute the racket parameters depends on the comparison between the new coming data and the empirical data. Moreover, a novel way based on a binocular vision system to verify the stroke model is proposed. Experimental results have showed that the stroke model created via the proposed method is applicable and the verification method is effective.**

I. INTRODUCTION

Since the first table tennis robot was born in 1983 [1], more and more researchers are interested in the table tennis robot system, because it is a good platform for researching, which contains automatic control, computer vision and others. The table tennis robot develops rapidly over the last few years.

In 1988, Andersson took a 6-degree-of-freedom (DOF) PUMA260 as the arm for table tennis robot and four cameras working at 60Hz as the vision system to track the flying ball [2]. However, for ensure that the robot played balls well, this robot had a narrow table which was 2 meter in length and 0.5 meter in width, and it added frames to make sure that the ball must pass. In 2003, Acosta built a robot based on the PC with the goal of low cost [3]. The robot had one camera to extract the ball's trajectory, and it had two rackets attached to a 2-DOF structure. Since the strength of the robot hitting the balls was small, the table for playing balls was also small.

The first table tennis robot built by the standard table-tennis rules was from the Osaka University [4]-[6]. Miyazaki constructed a robot with a 4-DOF arm and a stereo vision system. The manipulator mounted on the table could move in the horizontal plane and had two rotations in lateral and up-down. And the control system combined the motion control card with PC. Muelling and Kober designed a robot with a 7-DOF arm (Barrett WAM) to imitate how the human played balls [7]-[8]. The vision system was made up by two high-speed cameras. By emulating human behaviors, the robot could perform like a human.

The vision and the racket control systems are two main challenges for the table tennis robot. As for the vision system, many good algorithms to track the ball were proposed [9]. With the help of high-speed cameras, the time to detect the ball becomes shorter and shorter. On the other hand, for hitting the ball to a desired position, the racket control should be precise. What's more, the racket trajectory is helpful to track the flying ball. Chen predicted the trajectory of the spinning ball successfully based on the motion of the racket [10].

Since the contact time between the racket and the ball during the stroke process is too short and the plastic surface of the racket is flexible, it is difficult to create a good physical model for the stroke process. What's more, it is not easy to obtain the racket parameters from the complex stroke process. Therefore, using the empirical data to model the stroke process would be a better and simpler way. Some researchers had used the empirical data to control the racket in the stroke process. Miyazaki used the empirical data to control the racket based on the locally weighted regression (LWR) [11], and Huang combined the LWR and the FCMAC learning algorithm to compute the racket parameters [12].

To control the racket better, this paper proposes a novel stroke method based on the neural network. In the section II, we introduce the table tennis robot system specifically. In the section III, a list of neural networks according to the empirical data is created for modeling the stroke process. In the section IV, a new way is proposed to verify the stroke model. The experiments and results are given in the section V. Finally, a brief conclusion is given in section VI.

II. THE SYSTEM HARDWARE

Our table tennis robot mainly contains two parts: a five-DOF motion mechanism to hit balls and two binocular vision systems. One of the binocular vision systems has two high-speed cameras to track the flying ball, which connects to the computer by the router. The other directly connecting to the computer is to obtain the trajectory of the racket. As shown in Figure 1, the racket is fixed on the motion mechanism and the five degrees of freedom includes three degrees in the horizontal plane and vertical direction, and the two degrees in lateral rotation and up-down rotation. In the table tennis robot system, there are four coordinate systems established: the world frame $\{O_W\}$, the racket frame $\{O_r\}$, and the two camera frames $\{O_A\}$ and $\{O_B\}$. The world frame $\{O_W\}$ coincides with the camera frame $\{O_B\}$. In the frame $\{O_W\}$, the $X_W Y_W$ plane is the table plane, and the origin point lies in the middle of the table short side. In the frame $\{O_W\}$, the Y_W-axis is parallel to the middle white line on the table plane, pointing toward the human side. The direction of Z_W-axis is upward and perpendicular to the table plane. In the camera frame $\{O_A\}$, the Z_A-axis and X_A-axis have the same directions with the Z_W-axis and X_W-axis, and one point on the table plane in the human side is chosen as the origin point of the frame $\{O_A\}$. In the racket frame $\{O_r\}$, for describing the racket motion easier, the racket's yaw axis for

* This work is supported by the National Natural Science Foundation of China under Grant 61305024, 61273337, 61227804 and 61333016. It is also supported by the Beijing Natural Science Foundation under Grant 3141002.

Kun Zhang, Zaojun Fang, and Jianran Liu, Min Tan are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (Zhang e-mail:zhangkun_ia@163.com).

the lateral rotation is considered as the X_r-axis, and the X_r-axis points towards the handle of the racket. The direction perpendicular to the racket plane is Z_r-axis.

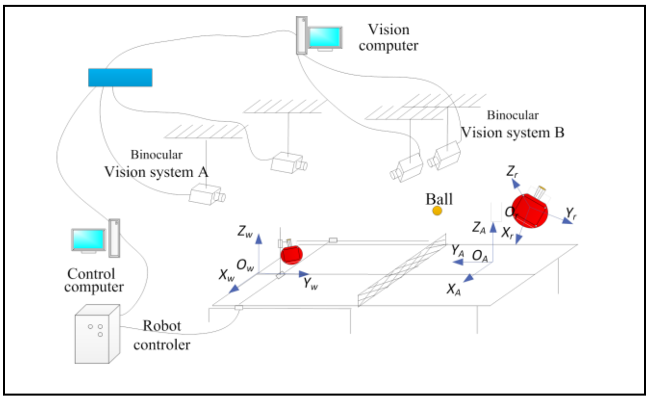


Figure 1. The table tennis robot and its coordinate systems

III. MODELLING THE STROKE PROCESS

A round that a person plays the ball with the table tennis robot contains three main procedures, as shown in Figure 2.

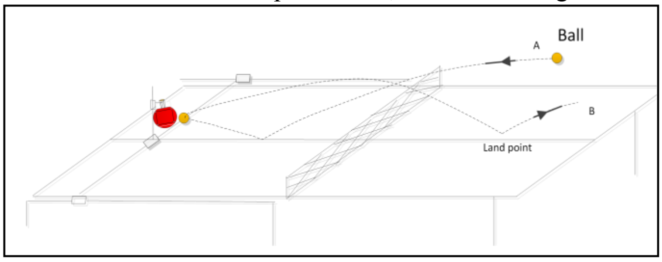


Figure 2. The model of ball flight

1. The ball flies from the human side to the robot.
2. The table tennis robot controls the racket to hit the ball back.
3. The ball flies back toward the human side.

We define that the trajectory A is the trajectory of the ball flying toward the table tennis robot. It means the beginning of the round. And the trajectory B represents the trajectory of ball flying after the stroke process. The racket parameters in our table tennis robot include the yaw angle θ_y, the pitch angle θ_p and the velocity V_r (V_{rx}, V_{ry}, V_{rz}), which influence the results about the land point. We also define V_{in} $(V_{inx}, V_{iny}, V_{inz})$ as the ball's velocity in the trajectory A, and V_{out} $(V_{outx}, V_{outy}, V_{outz})$ is the ball's velocity in the trajectory B. Because of the difficulty in measuring the angular velocity of the ball, we only study the ball without spinning in this paper.

The LWR is a good method to fit a local model with the empirical data weighted by a distance function [13]. The importance of the stored data reflects on the weighted distance function, and the nearer data make bigger contributions to the local model. Therefore, since the stroke process is complex, the stroke model based on the LWR algorithm is applicable. However, when the number of the stored data is huge, it is difficult to train the model using the LWR on real time.

The neural network is also a good way to model some complex models [14], and it can help us find some characters about the unknown model. Nevertheless since the training rate is too low, many neural networks have to be trained offline for the model. What's more, the amounts of layers and nodes in a neural network have to be decided depend on our experience. It is uncertainly to get a perfect model based on the neural network. Moreover, the neural network is related to all the memory data, which contain many invalid data in an actual condition.

According to the condition of the racket motion, we choose the ball velocity difference during the stroke ($\Delta V_b = V_{out} - V_{in}$) as the input value in a neural network, and we set a restriction: $V_{rx} = V_{rz} = 0$. In this case, the racket model is simpler and easier to be investigated. It is also known that the stroke model based on empirical data is a local model. Thus, one important thing before creating a neural network is to cluster the experience data in order to reduce influences from some invalid data.

A. Data clustering and the neural network creation

We define that the experience data are $\{(\Delta V, \theta_y, \theta_p, V_{ry})\} = \{(\Delta V_1, \theta_{y1}, \theta_{p1}, V_{ry1}), \dots, (\Delta V_i, \theta_{yi}, \theta_{pi}, V_{ryi})\} i = 1,2, \dots, N$, where N is the quantity of the experience data.

$$Dis_j^i = \left\| \Delta V_j - \Delta V_i \right\|, \{j = 1,2, \dots .K\}, K < N \qquad (1)$$

According to (1), we could obtain an empirical data set $\{S_i\} = \{ (\Delta V_1^i, \theta_{y1}^i, \theta_{p1}^i, V_{ry1}^i), \dots, (\Delta V_j^i, \theta_{yj}^i, \theta_{pj}^i, V_{ryj}^i) \}$ for the empirical data $(\Delta V_i, \theta_{yi}, \theta_{pi}, V_{ryi})$. In the data set $\{S_i\}$, each Dis_j^i is smaller than others in the whole empirical data compared with the empirical data $(\Delta V_i, \theta_{yi}, \theta_{pi}, V_{ryi})$.

$$Dis_j^i = \left\| \Delta V_j - \Delta V_i \right\|, \{j = 1,2, \dots .K\}, K < N$$

where K is the amount of the empirical data which one data set has. Using the data set $\{S_i\}$, we could train a BP neural network $\{Net_i\}$ offline and gain the parameters about the neural network. The input value are $\{\Delta V_1^i, \dots, \Delta V_j^i\}$, and the target value in the neural network are the racket parameters $\{(\theta_{y1}^i, \theta_{p1}^i, V_{ry1}^i), \dots, (\theta_{yj}^i, \theta_{pj}^i, V_{ryj}^i)\}$. So we can obtain N BP neural network based on each empirical data. The amounts of the layers and nodes in each neural network are the same, there are n layers and the ith layer has m_i nodes. Finally a list of the neural network is formed.

B. Computation of the racket parameters

The table tennis robot initializes the neural network parameters before starting to play balls. The land point, the trajectory A and the ball's parameters (ΔV_{target}) can be computed by the physical model of the flying ball before the racket strikes the ball [15]. Then the rule to choose the neural network is given by (2).

$$Dis = min \left\| \Delta V_i - \Delta V_{target} \right\|, \{i \in 1,2, \dots .N\} \qquad (2)$$

According to the comparison between the new ball's parameters and the whole empirical data, we determine that the target network is the $\{\mathrm{Net}_i\}$ whose Dis is minimum, then we can get the racket parameters $(\theta_{ytarget}, \theta_{ptarget}, V_{rtarget})$ as the input value ΔV_{target} to the $\{Net_i\}$.

C. Update of the clustered data

After the table tennis robot hits the ball many times, there would be many new empirical data produced. When a new data (ΔV_{new}) appears, the table tennis robot would compute a new racket parameters($\theta_{ynew}, \theta_{pnew}, V_{rnew}$), at the same time if the binocular vision system A finds that the land point is consistent with the desired position via racking the trajectory of the ball on real time, the table tennis robot would consider the data as a correct and useful data. The table tennis robot system would record and save the data to the empirical data.

When the stroke process finishes, the neural network list will be updated offline according to the new empirical data, the steps are as follows.

Step 1: For each old empirical data ($\Delta V_i, \theta_{yi}, \theta_{pi}, V_{ryi}$), find out the most similar empirical data based on the function (2) and record the Dis_{min}^{i}. $\{i = 1,2, \dots, N\}$

Step 2: Find out the most similar experience data for the new empirical data between those existing empirical data, and record the Dis_{min}^{new}.

We define a threshold Dis_{thre} as the similar degree for the experience data.

If $min\{Dis_{min}^{i}\} < Dis_{thre}$, and $Dis_{min}^{new} > Dis_{thre}$, then delete the Net_i and create a new neural network Net_{new} according to the new experience data ΔV_{new}. The other parameters of the neural networks would be updated because of the new empirical data coming. In this way, the amount of the neural networks does not change, but the number of the whole empirical data is increasing.

Finally, the experience data, by which the neural networks are created, would be distributed uniformly.

IV. THE WAY TO VERIFY THE MODEL

There are some special cases when the table tennis robot is in motion. For example, sometimes the table tennis robot could not reach the desired position on our commands. In this case, we don't know the practicability of the stroke model built based on the neural network. Thence, a good way is demanded to verify the stroke model and check out whether the racket motion is suitable for the parameters ($\theta_{ytarget}, \theta_{ptarget},, V_{rtarget}$) or not. The key of the way is that it needs to reflect the moment when the racket hits balls. As shown in the Figure 2, if we can obtain the intersection point of the trajectory A and B, the moment when the racket hits balls can be known.

A. Computation of the velocity of the incoming ball

During the ball flying, there are three main forces on the ball, which are the air resistance, and gravity and the Magnus force [16]. The Magnus force depends on the angular velocity of the ball. Because in this paper, we just study the ball without spinning, the Magnus force can be ignored. The force can be computed as (3) and (4).

$$\overrightarrow{F_D} = -\frac{1}{2}\rho S C_D \|\vec{V}\|\vec{V} \tag{3}$$

$$\overrightarrow{F_G} = [0,0,-mg]^T \tag{4}$$

where $\overrightarrow{F_G}$ is the gravity force and $\overrightarrow{F_D}$ is the air resistance. ρ is the air density, S is the effective cross-sectional ball area, C_D is the drag coefficient, m is the mass of the ball, g is the gravity accelerator. The status vector of the ball in the flight is $[x_k, y_k, z_k, V_{xk}, V_{yk}, V_{zk}]$, according to the force, the trajectory of the ball flight can be described by (5) and (6).

$$m\vec{a} = \overrightarrow{F_G} + \overrightarrow{F_D} \tag{5}$$

$$\begin{bmatrix} x_k \\ y_k \\ z_k \\ V_{Xk} \\ V_{yk} \\ V_{zk} \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \\ V_{Xk-1} \\ V_{yk-1} \\ V_{zk-1} \end{bmatrix} + \begin{bmatrix} \vec{V}_{xk-1} \\ \vec{V}_{yk-1} \\ \vec{V}_{zk-1} \\ -\frac{1}{2}\rho S C_D \|\vec{V}_{xk-1}\|\vec{V}_{xk-1} \\ -\frac{1}{2}\rho S C_D \|\vec{V}_{yk-1}\|\vec{V}_{yk-1} \\ -\frac{1}{2}\rho S C_D \|\vec{V}_{zk-1}\|\vec{V}_{zk-1} - g \end{bmatrix} * \Delta T \tag{6}$$

where k = 1,2,…, and the time interval ΔT for an iteration step is very small. The binocular vision could sample the positions and times of the ball in the trajectory A. Then the initial status vector $[x_0, y_0, z_0, V_{x0}, V_{y0}, V_{z0}]$ can be computed through the sampled data. We can use a second-order polynomial to fit the sampled data, as given in (7).

$$\begin{cases} x_i = a_1 {t_i}^2 + b_1 t_i + c_1 \\ y_i = a_2 {t_i}^2 + b_2 t_i + c_2 \\ z_i = a_3 {t_i}^2 + b_3 t_i + c_3 \\ V_{xi} = a_1 t_i + b_1 \\ V_{yi} = a_2 t_i + b_2 \\ V_{zi} = a_3 t_i + b_3 \end{cases} \tag{7}$$

where $a_1, b_1, c_1, a_2, b_2, c_2, a_3, b_3, c_3$ are the coefficients of the polynomial which could be computed via the least square method (LSM) with the sampled data. So the whole status of the ball in the trajectory A can be acquired through the iteration. If the strike doesn't exist, the ball would reach the table plane follow the trajectory A.

B. Computation the velocity of the ball after hitting

Because the time interval ΔT is very small and the velocities of the ball are continuous, we can assume that the $[V_{xk}, V_{yk}, V_{zk}] \approx [V_{xk-1}, V_{yk-1}, V_{zk-1}]$. Therefore, we can approximately reverse the iteration as presented in (8).

$$\begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \\ V_{Xk-1} \\ V_{yk-1} \\ V_{zk-1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ z_k \\ V_{Xk} \\ V_{yk} \\ V_{zk} \end{bmatrix} + \begin{bmatrix} -\vec{V}_{xk} \\ -\vec{V}_{yk} \\ -\vec{V}_{zk} \\ \frac{1}{2}\rho S C_D \|\vec{V}_{xk}\|\vec{V}_{xk} \\ \frac{1}{2}\rho S C_D \|\vec{V}_{yk}\|\vec{V}_{yk} \\ \frac{1}{2}\rho S C_D \|\vec{V}_{zk}\|\vec{V}_{zk} + g \end{bmatrix} * \Delta T \tag{8}$$

In the same way with the calculation of the velocity V_{in}, the status in the whole trajectory B can be obtained.

C. Computation of the intersection point

$[P_i^A, t_i^A, V_i]$ is defined as the status vector of the *i*th point in the trajectory A and $[P_j^B, t_j^B, V_j]$ as the status vector of the *j*th point in the trajectory B. In an ideal condition, it can be found out that there is an intersection point of the trajectory A and the trajectory B as Figure 2. However, due to the sampling error from the vision system A, the positions and times of the sampled data and the iteration functions deviate from the ideal cases. Thus, we maybe cannot get the intersection point directly. The distances between two points $[P_i^A, t_i^A, V_i]$ and $[P_j^B, t_j^B, V_j]$ with the same time stamp are not the shortest distance between the trajectory A and B, and the two points $[P_i^A, t_i^A, V_i]$ and $[P_j^B, t_j^B, V_j]$ whose distance is the shortest have a huge difference in the time. In this respect, we need to fuse the information from the distance and time with the rule as presented in (9).

$$\emptyset(i,j) = min\{\alpha_1 \left\|P_i^A - P_j^B\right\| + \alpha_2(t_i^A - t_j^B)\} \quad (9)$$

where α_1, α_2 decided by the experience are the weights of the information in the distance and time. The status vector $[P^C, t^C]$ of the intersection point can be obtained as shown in (10).

$$[P^C, t^C] = [(P_i^A + P_j^B)/2, (t_i^A + t_j^B)/2] \quad (10)$$

Then the input value of the neural network is born by (11).

$$\Delta V = V_j - V_i \quad (11)$$

D. *Computation of the racket pose*

After obtaining the input of the neural network, we still need to know the output data: the racket pose. The racket pose can be obtained by the binocular vision system B. To obtain the racket pose, we draw a mark on the racket in advance, as given in Figure 3.

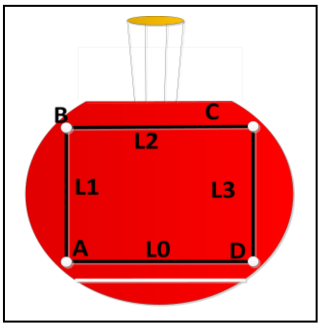


Figure 3. The racket in the end of the maniulator

The procedure of getting the racket pose is introduced in [17] as shown in Figure 4. The steps are as follow:

Step 1: After the binocular vision system B captures the images of the racket, we can extract the racket according to the red region, because the red points are special in the HSV color space. At the same time, we can get some information about the light situation from the red region's RGB values.

Step 2: The white line can be detected from the red region because the V values of the white in the HSV color space are maxima, and the S values are small. Simultaneously, the information in RGB color space about the white points can be obtained.

Step 3: With the help of the slope of the white line, the four black lines can be estimated with the rule that the sums of R,G,B value of the black points are minima during scanning the red region in line.

Step 4: The four corners can be computed by the four black lines, and depend on the information gained in the step 2, we can verify whether the four corners are right or not.

Step 5: After the four corners obtained, the coordinates of the four corners in the frame $\{O_W\}$ can be computed via the three-dimensional (3D) reconstruction. There are $[x_i, y_i, z_i]$, $\{i = \mathrm{A, B, C, D}\}$.

Step 6: In most conditions, the origin point in the frame O_W isn't in the racket plane when the racket strikes the ball, so the plane of the racket can be described as (12) .

$$ax_i + by_i + cz_i + 1 = 0, \{i = \mathrm{A, B, C, D}\} \quad (12)$$

where *a, b, c* are the coefficients of the racket plane. The coefficients can be computed via the LSM algorithm. Then the racket pose can be calculated with the function (13).

$$a * tan\theta_p = b, c * sin\theta_p = b * tan\theta_y \quad (13)$$

Step 7: The velocity of the center point can be acquired by the function (14).

$$V_{ry} = (\sum y_i)/(4 * \Delta t), \{i = \mathrm{A, B, C, D}\} \quad (14)$$

where Δt is the time of the vision system capturing two images. The velocities of other points in the racket plane can be considered as equal to the velocity of the center point on the table plane.

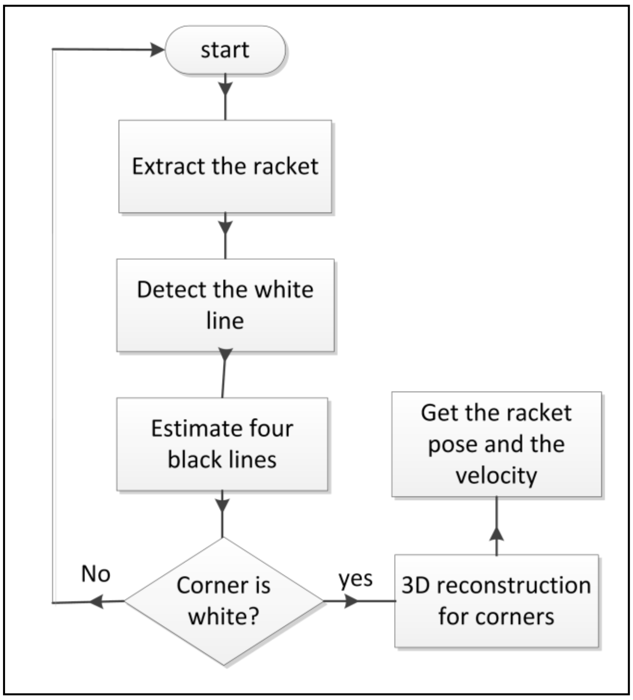


Figure 4. The precedure to get the racket pose

Along with the velocity of the racket and the racket pose obtained, the goal of getting the output data for the neural network is achieved. Finally, if the time stamps about the input data and output data are checked, the model for the stroke process can be verified.

V. EXPERIMENTS AND RESULTS

A. *Experimental System*

The experimental system was consisted of a 5-DOF manipulator and two binocular vision systems A and B, as shown in Fig.5. The cameras with high speed in the vision system A were VC 4458, and the cameras in vision system B were GC660C, they were all connected with the computer and the router through internet cables. The computer used for the vision systems was with 3.40- G internal frequency, and 4.0- G RAM working in Window XP.

Figure 5. The experimental table tennis system

B. *Experiment results about the model created via the proposed method*

Before creating the stroke model, we acquired many empirical data. The amount of those data was 1800, 1500 of those data were used to create neural networks and train the model, the rest were used to test the stroke model.

With a lot of tests, we decided that each BP neural network had 3 hidden layers. The first hidden layer had 10 nodes, the second was 5 nodes, and the third hidden layer was 3 nodes, as shown in the Figure 6. For each neural network, the size of a data set for training the neural network was 1000.

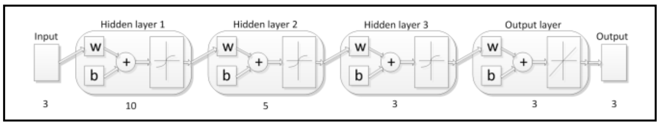
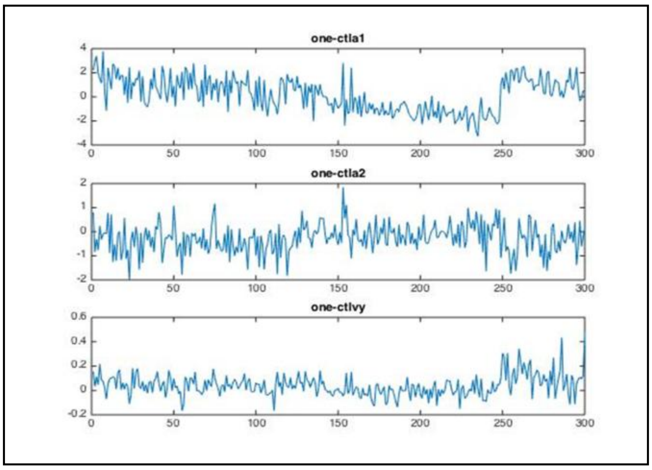


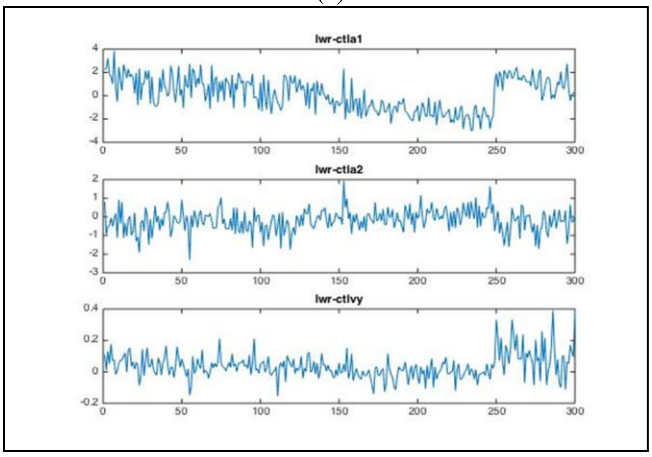
Figure 6. The BP neural network model

Finally, there were 1500 neural networks corresponding to the empirical data. The results from 300 empirical data for testing the stroke model created by the proposed method were shown in the table I and Figure 7, and the model created by the method which just used one neural network to train all the empirical data and the model created in the LWR algorithm were tested.

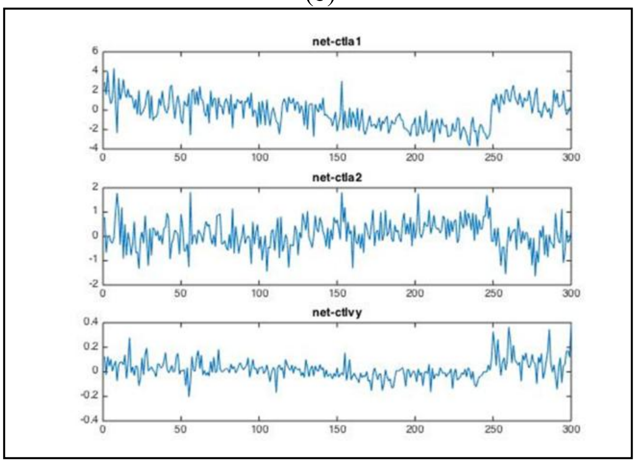
The result showed that the proposed method was nearly to the LWR algorithm and obviously better than the method which just had one neural network.



(a)



(b)



(c)

Figure 7. The errors of 300 experience data as test data in diffrence methods, (a) the method with one neural network, (b) the LWR algorithm, (c) the proposed method

TABLE I. THE RESULTS FOR THE RACKET MODEL

Method	**The method proposed**			**Our LWR**			**One neural network**		
variables	θ_p	θ_y	V_{ry}	θ_p	θ_y	V_{ry}	θ_p	θ_y	V_{ry}
average	-0.1877	0.0888	0.0213	0.1176	-0.1873	0.0362	0.4204	-0.1594	0.0449
variance	2.2622	0.3652	0.0070	2.0516	0.3608	0.0065	3.5488	0.4897	0.0042

C. Experiment results about the way to verify the racket model

In the experiments, when the table tennis robot was competing with a human, the vision system sampled the data, and the procedure could be reappeared via the proposed way. The velocities of the ball before and after the strike could be obtained, simultaneously we could acquire the racket poses and the velocities of the racket with the vision system B. The whole flight process was shown in the Figure 8.

$$\Delta V = V_{out} - V_{in} = [\, 0.1944, 6.0170, 1.5964] \text{ (m/s)}$$

Then we could get the results calculated by the three methods when the input data was ΔV, the result was shown in table II.

It was found that the results obtained by the method proposed and the LWR algorithm were close to the real situation, the racket model created by the method proposed was practical for the table tennis.

Due to the errors of detecting the ball positions via the binocular vision system A, the velocities of the ball and the racket pose were not absolutely precisely, and because the time to obtain the racket pose was long, and the time stamps were hard to be verified, we had to simulate the stroke process offline. The results showed that the verification method could appropriately reflect the situation about the stroke moment.

TABLE II. THE RESULT CALCULATED WITH THE INPUT DATA

The method	The result
The method proposed	[-0.3779,-2.0314,1.8113]
The LWR algorithm	[-0.3916,-2.0436,1.8709]
The method with one network	[0.6455,-2.0050,1.7134]
The way for verification	[-0.1023,-2.0334,1.8601]

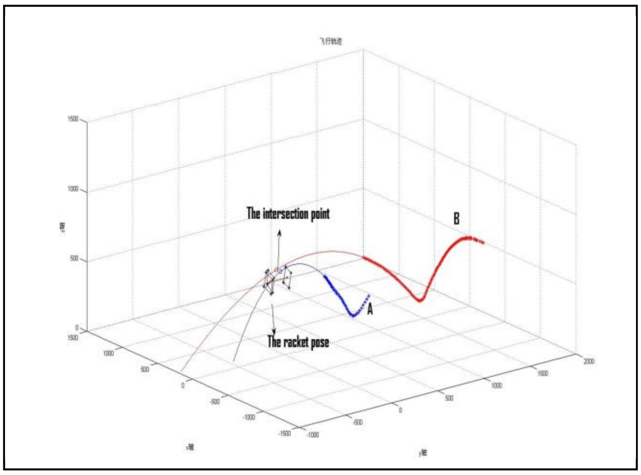


Figure 8. The whole flight estimation

VI. CONCLUSION

In this paper, a novel stroke model created via the neural network was presented. Depending on the empirical data clustering for reducing the influences from the valid training data, a list of the neural networks was established. The proposed way, which could represent the whole procedure of the flying ball and the stroke moment, would help us to verify the applicability of the stroke model. The experiments and results showed the practicability of the proposed model and the effectiveness of the verification method. In the future, we will find out better models for the stroke process.

REFERENCES

[1] R.L. Andersson, "A robot ping-pong player: experiment in real time intelligent control." Cambridge, MA, USA: MIT Press 1988.

[2] R.L. Andersson, "Aggressive trajectory generator for a robot ping-pong player," *IEEE Control Systems Magazine* , vol.9, no.2, pp.15-21, Feb. 1989

[3] L. Acosta, J. J. Rodrigo, J. A.Méndez, G. N. Marichal, M. Sigut, "A PC-based low-cost ping-pong robot." *IEEE Robotics and Automation Magazine*, *10*, 44-52, 2003.

[4] M. Takeuchi, F. Miyazaki, M. Matsushima, M. Kawatani, T. Hashimoto, "Dynamic dexterity for the performance of "wall-bouncing" tasks," *IEEE International Conference on Robotics and Automation, Proceedings. ICRA* pp.1559-1564 vol.2, 2002.

[5] M. Matsushima, T. Hashimoto, F. Miyazaki, "Learning to the robot table tennis task-ball control & rally with a human," *IEEE International Conference on Systems, Man and Cybernetics*, pp.2962-2969 vol.3, 5-8 Oct. 2003.

[6] F Miyazaki, M Matsushima, T Hashimoto, M Takeuchi, "A table tennis robot controls a ball and rallies with a human being," *Advanced in Robot Control.* Berlin: Springer Berlin Heidelberg, pp.317-341, 2006.

[7] K Mülling, J. Kober, J. Peters, "A biomimetic approach to robot table tennis," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* pp.1921-1926, 18-22 Oct. 2010.

[8] K Mülling, J. Kober, J. Peters, "Simulating human table tennis with a biomimetic robot setup." *From Animals to Animats 11*. Springer Berlin Heidelberg, 2010. 273-282.

[9] J. Liu, Z. Fang, K. Zhang, M. Tan, "Improved high-speed vision system for table tennis robot," *IEEE International Conference on Mechatronics and Automation (ICMA),* pp.652-657, 3-6 Aug. 2014.

[10] G. Chen, D. Xu, Z. Fang, Z. Jiang; M. Tan, "Visual Measurement of the Racket Trajectory in Spinning Ball Striking for Table Tennis Player," *IEEE Trans. Instrumentation and Measurement,* vol.62, no.11, pp.2901-2911, Nov. 2013.

[11] M. Matsushima, T. Hashimoto, M. Takeuchi, and F. Miyazaki, "A learning approach to robotic table tennis," *IEEE Trans. Robot.*, vol. 21, no. 4, pp. 767–771, Aug. 2005.

[12] Y.Huang, D. Xu, M. Tan, H. Su, "Adding Active Learning to LWR for Ping-Pong Playing Robot," *IEEE Trans Control Systems Technology,* vol.21, no.4, pp.1489-1494, July 2013.

[13] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning," *Artificial Intell. Rev.*, vol. 11, nos. 1–5, pp. 11–73, 1997.

[14] D. E. Rumelhart, G. E. Hinton, R. J. Williams, "Learning representations by back-propagating errors". *Cognitive modeling*, *5*, 3. 1988

[15] Z. Zhang, D. Xu, and M. Tan, "Visual measurement and prediction of ball trajectory for table tennis robot," *IEEE Trans. Inst. Meas.*, vol. 59, no. 12, pp. 3195–3205, Dec. 2010.

[16] Wikipedia: Magnus Effect. [Online]. Available: http://en.wikipedia.org/wiki/Magnus_effect

[17] Kun, Zhang, et al. "An adaptive way to detect the racket of the table tennis robot based on HSV and RGB." *Control Conference (CCC), 2015 34th Chinese*. IEEE, 2015.