IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 8, 2021, accepted June 18, 2021, date of publication June 29, 2021, date of current version July 21, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3093340

Ball Motion Control in the Table Tennis Robot System Using Time-Series Deep Reinforcement Learning

LUO YANG[1,2], HAIBO ZHANG[2], XIANGYANG ZHU[1], AND XINJUN SHENG[1], (Member, IEEE)
[1]School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
[2]Shanghai Pongbot Technology Company Ltd., Shanghai, Pudong 201206, China

Corresponding author: Xinjun Sheng (xjsheng@sjtu.edu.cn)

This work was supported by the Science and Technology Commission of Shanghai Municipality under Grant 18JC1410400.

ABSTRACT One of the biggest challenges hindering a table tennis robot to play as well as a professional player is the ball's accurate motion control, which depends on various factors such as the incoming ball's position, linear, spin velocity and so forth. Unfortunately, some factors are almost impossible to be directly measured in real practice, such as the ball's spin velocity, which is difficult to be estimated from vision due to the little texture on the ball's surface. To perform accurate motion control in table tennis, this study proposes to learn a ball stroke strategy to guarantee desirable ''target landing location'' and the ''over-net height'' which are two key indicators to evaluate the quality of a stroke. To overcome the spin velocity challenge, a deep reinforcement learning (DRL) based stroke approach is developed with the spin velocity estimation capability, through which the system can predict the relative spin velocity of the ball and stroke it back accurately by iteratively learning from the robot-environment interactions. To pre-train the DRL-based strategy effectively, this paper develops a virtual table tennis playing environment, through which various simulated data can be collected. For the real table tennis robot implementation, experimental results demonstrate the superior performance of the proposed control strategy compared to that of the traditional aerodynamics-based method with an average landing error around 80 mm and the landing-within-table probability higher than 70%.

INDEX TERMS Ball motion control, reinforcement learning, spin velocity estimation, table tennis robot.

I. INTRODUCTION

The sports have become a frontier of robot application area since the success of AlphaGo and AlphaZero [1]. With the development of both the hardware and the algorithm, the ultimate goal of a sport robot switched from playing with human to playing the role of a professional human player. In such a case, the sport robot should be able to act as a humanoid system both physically and intelligently. For example, the table tennis robot studied in this research has been designed like a humanoid system, which can sense the position, speed and spin velocity of the incoming ball, and then make the decision of stroke strategy followed by the controlling the robot to stroke back the ball. In such a case, building an accurate physical model of the table tennis environment is essential for designing the robot control system [2], [3]. Table tennis can be considered as a complicated physical system including the ball flying physical model, collision model, racket hit model. Therefore, we have to build an entire physical system that combines all of above models for robot controller design. The accuracy of the model decides the control performance of the table tennis robot [4].

Other than the modeling work, the state observation is another challenge of the robot control system design. Fully observing the environment changes is almost impossible for the table tennis game, because of the high speed of ball flying and spinning. Based on our testing result, during the table tennis game, the ball flying speed and spin velocity can reach as high as 25 m/s and 300 rad/s respectively. Although many studies have been made on the ball tracking and trajectory prediction of the table tennis [3]–[5], seldom attempts had been made for the ball's spinning velocity estimation.

The associate editor coordinating the review of this manuscript and approving it for publication was Shun-Feng Su.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

Although obtaining an accurate spin velocity is a crucial step for the stroke strategy decision of the table tennis robot to play and compete with human players, it is very difficult to capture the spin velocity of the ball using a regular vision based method. From the point of view of the human player, the spin type (such as the top, back and side spins) and the velocity usually are judged by observing the swing movement of the opponent player empirically with a rough spin velocity of the table tennis ball.

Several attempts have been made to estimate the spin velocity of the table tennis ball, and generally can be divided into two categories. The first one is to use the high-speed camera to record the trajectory of the table tennis and identify the trademark or other signs on the ball [6]–[8]. Because of the little texture on the surface of the ball (almost the uniform color), even a high-speed camera was used, a relatively big estimation error exists.

Other than the ones used the trademark on the surface of the ball, artificial markers are also used to increase of estimation accuracy of the spin velocity. For example, Liu *et al.* drew several marker points on the ball and obtained the pose information in real-time through analyzing the images captured by a high-speed camera. They used a nearest-neighbor interpolation method to estimate the pixel difference between two adjacent frames, and obtained the spin information with the help of the conjugate gradient method [6]. In [7], combining the process of recognition, segmentation and center point extraction, the spin velocity of the table tennis ball was calculated with a controlled error. However, in real practice, drawing any marker on the ball is not allowed during the table tennis game, and that makes this marker based method only work in the laboratory but not the game court. Therefore, a markerless estimation method is needed for the spin velocity estimation in real practice.

Another way to estimate the spin velocity is to analyze the incoming trajectory of the ball [3], [9]–[11]. Nakashima *et al.* obtained the spin information of table tennis by minimizing the error between the actual trajectory and theoretical trajectory, which was derived based on an aerodynamic model of the table tennis [9], [10]. The position information of the table tennis was used to estimate the spin velocity directly, followed by a descending simplex method to minimize the error. However, the accuracy of this method is based on the accurate trajectory estimation which is difficult to be obtained in real practice. Therefore, how to estimate the spin velocity accurately is still a great challenge of the motion control of the table tennis robot.

To date, more and more intelligent methods have been implemented to enhance robotic automation, thanks to the rapid development of artificial intelligence [30]. It has been adopted as effective approaches for controlling the robots that can play table tennis. For instance, Y. Zhu *et al.* proposed to utilize Monte-Carlo based optimization method to let robot learn how to return a ball [31]; O. Koc et al, developed an intelligent on-line optimal trajectory generator to assist the table tennis robots [32]. Different from the traditional neural network-based method [30], whose approximating capability is constrained by the limited number of neurons, or the supervised learning methods that require ground truth labels [33], in this study, a deep reinforcement learning (DRL) algorithm has been developed to estimate the ball spin velocity through interacting with the environment continuously. The DRL algorithms have been proved effectively to allow agents to accomplish various tasks in the research areas of robotics [13]. For example, the Deep Deterministic Policy Gradient (DDPG) algorithm that combines the deep network and Actor-Critic frame, as well as an off-policy model-free algorithm, have been widely used in robot applications, such as motion control for biped robot [13] and inserting a clip into a rigid object [14]. Recently, there are a number of algorithm improvements on the DDPG which increase either the sample efficiency [15]–[21] or practicality [22]. The Twin delayed DDPG algorithm [20] is used to complete the usual manipulations such as reaching, pushing and pick-and-place only with the pixel input and the reward shaped by image difference [12]. In addition, maximum entropy reinforcement learning framework is combined with Actor-Critic frame which overcomes the explore problem to a great extent and thus enhance the agent to complete a more complex task such as the humanoid robot with 21-dimension of freedom [23]. In [24], an asymmetric Actor Critic algorithm [22] was used to bridge the 'sim-to-real' gap, and control a five-fingered humanoid hand to manipulate a block from an initial to a goal configuration only with simulation interaction. In this paper, a modified DDPG is designed as the backbone network to estimate the spin velocity of the table tennis ball.

Different from regular usage of DRL, which outputs the action, we use it to predict the spin velocity of the ball from the incoming flying trajectory through interacting with the environment. This is more like the traditional deep learning (DL) based method, but the difference is the DRL does not need the supervised label of the spin velocity which is required for the DL which is hardly obtained as mentioned above. It should be noted that instead of absolute spin velocity, the output of the DRL network is a relative spin velocity which includes the environment factors. This is because the flying trajectory of the ball is affected by various factors like the moving speed, spin velocity, gravity, buoyancy, air resistance and Magnus force, which is a highly coupled physical system. We encoded all the environment various other than the physical model into the relative spin velocity which can be learned from the DRL algorithm directly and used that as one of the inputs of our stroke strategy. This ensures the accuracy of the motion control of the table tennis ball.

Compared to the existing researches (e.g., [28] and [29]), which are unable to return a marker-free spinning table tennis ball, the DRL-based method proposed in this paper significantly improve the capability of handling this situation. Contributions of this research can be summarized as following: 1) It is the first time that DRL method is implemented to estimate spin velocity of a flying table tennis ball. Compared to the methods by employing extra cameras and markers,

the proposed DRL method provides a relatively low cost and marker-free solution. 2) The developed table tennis robot, with DRL-based flying ball spin velocity estimation capability, is one of the best robots that can play table tennis, which significantly enhances the success rate of returning a ball when it presents a spin motion.

This paper is organized as follows. A dynamic model of the table tennis ball is built in Section II, The model includes the physical ball flying model of table tennis, a ball-table collision model, and a ball-racket collision model. A DRL-based table tennis ball stroke strategy is designed in Section III. In Section IV, the effectiveness of the algorithms is verified in the simulation environment. Experimental implementation and test results of the proposed method conducted by a physical robot platform are demonstrated in Section V. Finally, we conclude this paper in Section VI.

II. DYNAMIC BEHAVIOR MODELING OF THE TABLE TENNIS

The dynamic behavior of the table tennis is a critical part of designing the control system of the table tennis robot. Usually, a more accurate model always comes with less controller design work. In this section, a dynamic model of the table tennis has been built ahead of the controller design. The entire table tennis environment includes a flight model of the incoming table tennis ball, a ball-table collision model, and a ball-racket collision model as follows.

1) Ball Flying Model: It is a physical flying trajectory model of the incoming ball. The input of this model includes the initial position, flying velocity, and the spin velocity of the ball, and the output of this model is the flying trajectory of the ball before colliding with the table.
2) Collision Model: It is a physical collision model between a ball and a table. The collision model is designed to obtain the trajectory and the spin velocity of the ball after the collision with the table. The input and output of this model are the trajectory and spin velocity of the ball before and after the collision with the table respectively.
3) Racket Hit Model: It is a physical hit model between a ball and a packet. Given the expected return point and the expected height above the net, the batting parameters are calculated by the algorithm.

Before the model building work, the coordinates of the table tennis environment are defined as shown in Fig.1. The world coordinate system coordinate is set as the Z-axis (Z_w) perpendicular to the table, Y-axis (Y_w) parallel to the long side of the table, and the origin of the coordinate at the center point of the table. For the racket coordinate system, the Z-axis (Z_R) is set as the normal direction of the racket, and the Y-axis (Y_R) is along the direction of the racket handlebar and points to the opposite direction of the racket handlebar. The origin of the coordinates is the center of the racket surface.

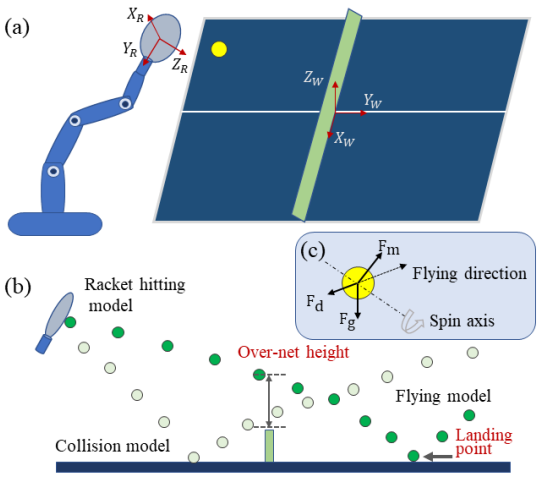

FIGURE 1. World coordinate and racket coordinate.

A. PHYSICAL ROTATION FLIGHT MODEL OF TABLE TENNIS BALL

The table tennis ball, a typical example of rotational flying objects, has a small mass with the high moving speed. The trajectory of the ball can be affected by both the flying and spin velocity. It can be seen that the comprehensive force of the rotary table tennis ball is non-linearly related to the state of motion, and the motion model has a high order nonlinear characteristics.

A spinning table tennis ball is mainly subject to gravity F_g, air resistance F_d, and Magnus force F_m in the air, as shown in Fig. 1(c). The flying velocity of the table tennis ball at the time t in the world coordinate system is defined as $V(t) = [v_x(t), v_y(t), v_z(t)]^T$, and the spin velocity is $W = [w_x, w_y, w_z]^T$. These forces can be described as follows.

$$\begin{aligned} F_g &= -m[\,0 \quad 0 \quad g\,]^T, \\ F_d &= -\frac{1}{2} C_d \rho A \|V(t)\| V(t), \\ F_m &= \frac{1}{2} C_m \rho r A (W \times V(t)), \end{aligned} \tag{1}$$

where m denotes the mass of standard table tennis ball, g is the acceleration of gravity, C_d represents the coefficient of air resistance, C_m denotes the Magnus force coefficient, and ρ represents the air density under standard conditions. A stands for the cross-sectional area of table tennis, and r is the radius of the table tennis ball. $\|V(t)\|$ is the 2-norm of $V(t)$, and $W \times V(t)$ is described as the cross of the spin velocity and the linear velocity. Notice that when the ball is flying in the air, the spin acceleration is small, so the spin velocity can be treated as a constant W.

It can be seen from the Eq. (1) that the magnitude of the air resistance is proportional to the square of the flying velocity. The proportionality factor is determined by the air resistance coefficient, air density and the cross-sectional area of the table tennis ball. The magnitude of the Magnus force is proportional to the outer product of rotation and flight speed,

and the direction of the Magnus force is perpendicular to both rotation speed and flight speed.

According to Newtonian mechanics, the acceleration of the flying ball can be described as (2),

$$\dot{V}(t) = -\frac{1}{2m}C_d\rho A\,\|V(t)\|\,V(t) + \frac{1}{2m}C_m\rho A(W \times V(t)) - [0,\ 0,\ g]^T, \quad (2)$$

and the discrete motion model of a moving ball can be derived as:

$$\begin{bmatrix} x^{k+1} \\ y^{k+1} \\ z^{k+1} \\ v_x^{k+1} \\ v_y^{k+1} \\ v_z^{k+1} \end{bmatrix} = \begin{bmatrix} x^k \\ y^k \\ z^k \\ v_x^k \\ v_y^k \\ v_z^k \end{bmatrix} + \begin{bmatrix} v_x^k \\ v_y^k \\ v_z^k \\ -k_d\|V^k\|v_x^k + k_m(w_y v_z^k - w_z v_y^k) \\ -k_d\|V^k\|v_y^k + k_m(w_z v_x^k - w_x v_z^k) \\ -k_d\|V^k\|v_z^k + k_m(w_x v_y^k - w_y v_x^k) - g \end{bmatrix} T_s. \quad (3)$$

where $k_d = -\frac{1}{2m}C_d\rho A$, $k_m = -\frac{1}{2m}C_m\rho A r$, the superscript k and $k+1$ denote the iteration step. The iteration period used to generate the virtual data in this paper is defined as T_s. To generate versatile data for the simulation study, the initial flight velocity of the ball is set between the minimum flight velocity V_{min}, and the maximum V_{max} randomly. Similarly, the initial rotation velocity is selected randomly between W_{min} and W_{max}.

B. BALL-TABLE COLLISION MODEL

The collision between the table tennis ball and the table shows two distinct behaviors due to the generated sliding friction and rolling friction, which is induced by the angle between the incident velocity direction and the normal direction of the table local surface. Therefore, the collision model between the ball and table is established based on the sliding friction model and the rolling friction model according to the incoming ball flying status. When the incoming ball's angle is greater than the critical friction angle, sliding friction will occur between the ball and the table. Otherwise, rolling friction occurs between the ball and the table. The velocity relationship before and after collision with the table can be described as follows,

$$\begin{aligned} V^{j+1} &= A_v V^j + B_v W^j, \\ W^{j+1} &= A_w V^j + B_w W^j, \end{aligned} \quad (4)$$

where V^{j+1} and W^{j+1} represent the linear and spin velocities after rebounding, respectively, while V^j and W^j denote the linear and angular velocities before rebounding. A_v, B_v, A_w and B_w are coefficient matrices about case dependent parameter a, and they can be generally represented as follows [25],

$$\begin{aligned} A_v &= \begin{bmatrix} 1-a & 0 & 0 \\ 1 & 1-a & 0 \\ 0 & 0 & -e_n \end{bmatrix}, \\ B_v &= \begin{bmatrix} 0 & ar & 0 \\ -ar & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\ A_w &= \begin{bmatrix} 0 & -\frac{3a}{2r} & 0 \\ \frac{3a}{2r} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\ B_w &= \begin{bmatrix} 1-\frac{3a}{2} & 0 & 0 \\ 0 & 1-\frac{3a}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \end{aligned} \quad (5)$$

where r denotes the radius of the table tennis ball, and the case-dependent parameter a can be described as:

$$a = \begin{cases} \mu(1+e_n)\frac{|v_{bz}|}{|v_{bt}|}, & V_s > 0 \\ \frac{2}{5}, & V_s \le 0, \end{cases} \quad V_s = 1 - \frac{5}{2}\mu(1+e_n)\frac{|v_{bz}|}{|v_{bt}|}. \quad (6)$$

In (6), $|v_{bz}|$ denotes the velocity magnitude along the axis of the world coordinate and $|v_{bt}|$ represents that in the horizontal plane of the world coordinate. When $V_s > 0$, a is calculated as the sliding friction coefficient; otherwise, a is treated as the rolling friction coefficient. In the representation of V_s, e_n indicates the coefficient of elastic recovery between the ball and the table, and μ represents the friction ratio between the ball and the table.

C. BALL-RACKET COLLISION MODEL

The collision model between the table tennis ball and the racket is an incomplete elastic collision model, which satisfies the following calculation formula [26],

$$\begin{bmatrix} v_1 - V_R \\ w_1 \end{bmatrix} = R_{RRM}(\alpha, \beta)\begin{bmatrix} v_0 - V_R \\ w_0 \end{bmatrix}, \quad (7)$$

where v_0, v_1 represent the linear velocity of the flying ball before and after collision with the racket, respectively; w_0, w_1 denote the associated spin velocity before and after the collision, respectively; V_R denotes the linear velocity of the racket and R_R, represented as (9), is the transformation matrix of the racket coordinate system relative to the world coordinate system, with α and β being the yaw and pitch angles of the racket,

$$R_{RRM}(\alpha, \beta) = \begin{bmatrix} R_R & 0 \\ 0 & R_R \end{bmatrix}\begin{bmatrix} A_{vv} & A_{vw} \\ A_{wv} & A_{ww} \end{bmatrix}\begin{bmatrix} R_R & 0 \\ 0 & R_R \end{bmatrix}^T, \quad (8)$$

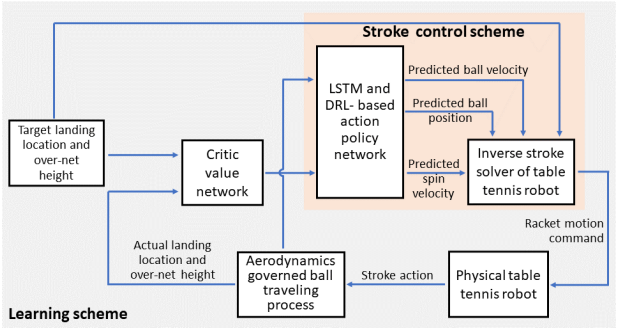

FIGURE 2. Schematic diagram of the proposed LSTM and DRL-based table tennis robot control method.

with R_R, A_{vv}, A_{wv} having the following expressions [26],

$$R_R = \begin{bmatrix} \cos\beta & \sin\beta\sin\alpha & \sin\beta\cos\alpha \\ 0 & \cos\alpha & -\sin\alpha \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix}, \quad (9)$$

$$\begin{aligned} A_{vv} &= \mathrm{diag}(1-k_v, 1-k_v, -e_r), \\ A_{ww} &= \mathrm{diag}(1-k_w r^2, 1-k_w r^2, 1), \\ A_{vw} &= k_v r S_c, \\ A_{wv} &= -k_w r S_c, \end{aligned} \quad (10)$$

where r denotes the radius of the ball, k_v represents the tangent recovery coefficient of the racket, e_r represents the normal recovery coefficient of the racket, k_w denotes the rotation conversion coefficient of the racket, and S_c is denoted as (11),

$$S_c = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \quad (11)$$

To enhance the efficiency of batting strategy learning, this paper develops a virtual table tennis robot environment according to the established flying ball model, to carry out the batting policy training and strategy verification (details in Section 4).

III. LSTM AND DRL-BASED TABLE TENNIS BALL STROKE STRATEGY

To precisely stroke back the incoming ball with a desired flying trajectory, the LSTM and DRL-based batting strategy is developed in this section, the schematic diagram is shown in Fig. 2. In detail, to stroke back a ball desirably, the method consisted of LSTM and DRL networks is employed to estimate the relative spin velocity of an incoming ball, which is then served to the following inverse stroke solver to generate the stroke action, and the integrated control method is named "Stroke control scheme". In order to provide a spin velocity in advance, the DRL method takes the incoming ball trajectory estimation generated by the LSTM module as the input, which can predict the ball flying trajectory precisely. The critic value network is employed to evaluate and train the "Stroke control scheme", which is also named as "Learning scheme" (as shown in Fig. 2).

The stroke action, which is determined by the parameter $[v_{R,x}, v_{R,y}, v_{R,z}, \alpha, \beta]$ (with $v_{R,*}, \alpha, \beta$ representing the linear velocity components, pitch and yaw of the racket, respectively) is generated according to the incoming ball trajectory, the desired landing location and the ideal over-net height. To determine the bat velocity and the orientation given the linear and spin velocity of the incoming ball, the inverse dynamics of the ball should be solved first.

A. DESIRED RACKET MOTION DERIVATION

To determine the motion of the racket, the flight state of the ball after collision with the racket should be formulated first. The purpose of the flight state estimation is to obtain the expected linear velocity $V_h = [v_x, v_y, v_z]$ after stroke according to the expected hit point $p_h = [x_h, y_h, z_h]$, the expected landing point $p_d = [x_d, y_d, z_d]$, and the expected point $p_n = [x_n, y_n, z_n]$ for the ball passing the net. To lower the calculation burden for real time implementation, the spin velocity of the ball after the stroke is set to zero, in other words, the robot tries to conquer the spinning motion of the incoming ball for stroking it back accurately. The flying trajectory of the return ball can be approximated as a parabolic curve. According to the three non-overlapping points (rebounding, net-passing and landing), a quadratic curve $z = dy^2 + by + c$ can be uniquely determined, where coefficients d, b and c can be calculated as follows,

$$\begin{aligned} d &= \frac{(y_d - y_h)(z_h - z_n) - (y_h - y_n)(z_d - z_h)}{(y_h - y_n)(y_n - y_d)(y_d - y_b)}, \\ b &= \frac{z_h - z_n}{y_h - y_n} - d(z_h + z_n), \\ c &= z_h - dy_h^2 - by_h. \end{aligned} \quad (12)$$

The quadratic coefficients determine the initial flight direction of the ball, to obey the physical laws, $d < 0$ has to be guaranteed while determining the flying trajectory, otherwise the expected landing point and the height above the net have to be altered to ensure that the expected trajectory conforms to the physical laws.

Based on the setting of uniform motion in y direction (effect of the air resistance is negligible for relatively low flying speed), the speed in y direction can be obtained as $v_y = \sqrt{g/(2d)}$. Flight time of the entire trajectory is calculated as $t_d = (y_d - y_h)/v_y$.

According to the uniform speed movement in x direction and the uniform acceleration movement in z direction, the expected speed in x direction and z direction with respect to the world coordinate system can be obtained as follows,

$$\begin{aligned} v_x &= \frac{x_d - x_h}{t_d}, \\ v_z &= \frac{z_d - z_h}{t_d} - \frac{g t_d}{2}. \end{aligned} \quad (13)$$

To realize the expected linear velocity $V_h = [v_x, v_y, v_z]$ obtained above for the return ball, the robotic bat should perform a desired stroke which is determined by the bat linear velocity V_R and orientation $[\alpha, \beta]$. According to the

ball-racket collision model, the expression of racket velocity can be obtained as follows [26],

$$V_R = v_0 + R_R(I - A_{vv})^{-1}(R_R^T(v_1 - v_0) - A_{vw}R_R^T w_0). \quad (14)$$

Substituting (14) into (7), the collision model of table tennis ball and racket can be further obtained as the expression of spin velocity before and after the rebounding,

$$\begin{aligned} w_1 &= S(v_1 - v_0) + R_R(A_{wv}(I - A_{vv})^{-1}A_{vw} + A_{ww})R_R^T w_0 \\ &= S(v_1 - v_0) + w_0, \end{aligned} \quad (15)$$

which is derived based on the constraint (16),

$$A_{wv}(I - A_{vv})^{-1}A_{vw} + A_{ww} = I, \quad (16)$$

and S has the following expression,

$$S = -R_R A_{wv}(I - A_{vv})^{-1}R_R^T = \frac{k_w r}{k_v} S_s(\alpha, \beta) \quad (17)$$

with $S_s(\alpha, \beta)$ being denoted as

$$S_s(\alpha, \beta) = \begin{bmatrix} 0 & \cos\alpha\cos\beta & \sin\alpha \\ -\cos\alpha\cos\beta & 0 & \cos\alpha\sin\beta \\ -\sin\alpha & -\cos\alpha\sin\beta & 0 \end{bmatrix}. \quad (18)$$

By defining $\xi = \frac{k_w r}{k_v}(v_1 - v_0)$ and $\eta = w_1 - w_0$, (15) can be simplified as $\eta = S_s(\alpha, \beta)\xi$. $S_s(\alpha, \beta)$ is an antisymmetric matrix, therefore the linear velocity difference and the spin velocity difference before and after rebounding satisfies the orthogonal relationship in (19),

$$(v_1 - v_0)^T(w_1 - w_0) = 0. \quad (19)$$

Performing expansion to $\eta = S_s(\alpha, \beta)\xi$, and the following expressions can be obtained,

$$\eta_x = \xi_y \cos\alpha \cos\beta + \xi_z \sin\alpha, \quad (20)$$
$$\eta_y = -\xi_x \cos\alpha \cos\beta + \xi_z \cos\alpha \sin\beta, \quad (21)$$
$$\eta_z = -\xi_x \sin\alpha - \xi_y \cos\alpha \sin\beta. \quad (22)$$

By combining (20) and (22), the following expression can be obtained,

$$\xi_y^2 cos\alpha^2 = (\eta_x - \xi_z sin\alpha)^2 + (-\eta_z - \xi_x sin\alpha)^2, \quad (23)$$

which can be simplified as $psin\alpha^2 + 2qsin\alpha + n = 0$, where $p = \xi_x^2 + \xi_y^2 + \xi_z^2$, $q = \xi_x\eta_z - \xi_z\eta_x$, $n = \eta_x^2 + \eta_z^2 - \xi_y^2$. After solving the quadratic equation about the sine function to get the value α, the value of β, and the racket linear velocity V_R can be further calculated according to (14). In summary, once the linear and spin velocity of the incoming ball is obtained, the action of the racket can be determined according to the desired landing location and the location while the ball passing the net.

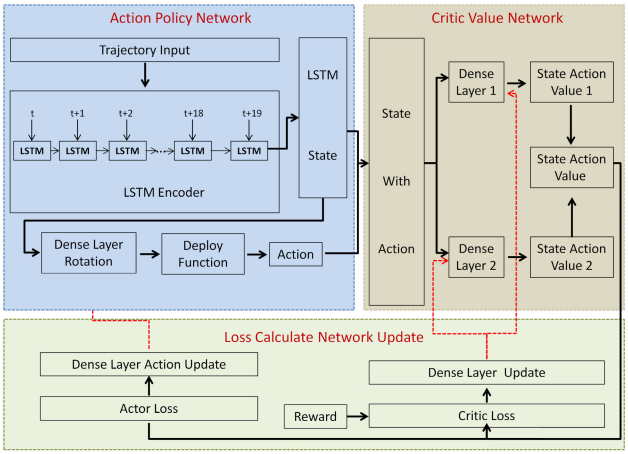

FIGURE 3. Structure of deep reinforcement learning algorithm.

B. SCHEMATIC OF THE LSTM AND DRL-BASED STROKE STRATEGY

To precisely stroke back the ball, the incoming trajectory of the ball is employed in this study, which contains the information of its linear velocity and the spin velocity. Compared to the linear velocity, the spin velocity is more difficult to estimate and usually contains uncertainty. Besides, the aforementioned desired racket moving strategy only considers the uniform flying velocity case and neglects the complicated aerodynamics of the ball, which brings more uncertainties to the racket control system. Therefore, this paper employs the DRL-based method, which has been proven to be sufficiently robust, to estimate the incoming ball status and to rectify the desired racket motion. The structure of the DRL-based racket control strategy is shown in Fig. 3, where one can see the robotic racket control strategy (action policy network) is basically comprised of the LSTM network [34] and the deploy function (the racket motion generator). For the real-time implementation, the LSTM is employed to estimate the incoming ball status, and the deploy function is utilized to generate racket motion.

For every episode, the first 20 equal-time-interval table tennis ball trajectory spatial positions generated in the virtual table tennis robot environment are served as the input. Then, the LSTM network (pre-trained in trajectory prediction) encodes the input to an LSTM state (50-dimension vector) which is served as the state in the common DRL setting. The five-dimension action vector consists of three-dimension linear velocity components and two-dimension pitch and yaw of the racket, which is predicted by a deploy function. The deploy function is predicted by a 3 layer dense actor network. The critic value network takes the LSTM state vector and the action as the input, and outputs the state action value Q. To overcome the overestimation bias in Actor-Critic, a twin critic network frame is adopted here, where two value estimation deep neural networks with the same structure (3 layer dense network) and different parameters are used to calculate the Q and the minimum Q is taken as the final state-action value. The reward function we used in the DRL algorithm is

shown as follow:

$$Reward = \begin{cases} -1 \quad \text{(fail)}, \\ \max(1 - distance\_loss - height\_loss, -1) \quad \text{(success)}, \end{cases} \tag{24}$$

where *distance_loss* and *height_loss* represent the deviation loss of the space position of the actual ball return point and the height across the net, respectively. The *distance_loss* and *height_loss* of the space position of the return ball is calculated by the following expression:

$$distance\_loss = \begin{cases} a_0 \|\Delta P\|_2^2, & \text{s.t. } \|\Delta P\|_2 < c_1, \\ a_1 \|\Delta P\|_2 + b_1, & \text{s.t. } c_1 \le \|\Delta P\|_2 < c_2, \\ \min(a_2 \|\Delta P\|_2 + b_2, 1), & \text{s.t. } \|\Delta P\|_2 \ge c_2, \end{cases} \tag{25}$$

where $\Delta P = P_{rebound\_real} - P_{rebound\_set}$.

$$height\_loss = \begin{cases} d_0 |\Delta H|^2, & \text{s.t. } \|\Delta H\| < f_1, \\ d_1 \|\Delta H\| + e_1, & \text{s.t. } f_1 \le \|\Delta H\| < f_2, \\ \min(d_2 \|\Delta H\| + e_2, 1), & \text{s.t. } \|\Delta H\| \ge f_2, \end{cases} \tag{26}$$

where $\Delta H = H_{real} - H_{set}$.

The virtual table tennis robot uses the predicted table tennis spin velocity (Action), preset $P_{rebound\_set}$ and the net height H_{set} of the table tennis ball return ball. The actual table tennis ball return point $P_{rebound\_real}$ and the ball are obtained through the state estimation of the trajectory and the stroke strategy. The height H_{real} is calculated by using the difference between the preset and actual ball return points. The following constraints should be satisfied to preserve continuity of *deviation_loss* and *distance_loss*, respectively.

$$\begin{cases} a_0 c_1^2 = a_1 c_1 + b_1, \\ a_1 c_2 + b_1 = a_2 c_2 + b_2, \\ a_2 > a_1 > 0, \\ 0 < c_1 < c_2 < 1. \end{cases} \tag{27}$$

$$\begin{cases} d_0 f_1^2 = d_1 f_1 + e_1, \\ d_1 f_2 + e_1 = d_2 f_2 + e_2, \\ d_2 > d_1 > 0, \\ 0 < f_1 < f_2 < 1. \end{cases} \tag{28}$$

As the DRL algorithm in our setting is used to solve one step decision problem, some notations are also different from the common setting (Markov Decision Process, MDP) [35], [36]. The detail differences are list as follows: 1) $Q - target = Reward$ instead of $Q - target = Reward + Q \max_{a'}(s', a')$. 2) $TD - error = Q(s, a)$–$Reward$ instead of $TD - error = Q(s, a)$–$(Reward + Q \max_{a'}(s', a')$. The experiment result shows that the algorithm is still convergent to good result at this setup.

TABLE 1. Parameter setting of the virtual table tennis environment.

m	Mass of ball	$0.0027kg$
g	Acceleration of gravity	$9.802m/s^2$
C_d	Coefficient of air resistance	0.5
C_m	Magnus force coefficient	1.0
ρ	Air density	$1.29kg/m^3$
A	Cross-sectional area of ball	$0.001256m^2$
r	Radius of ball	$0.02m$
T_s	Iteration period	$0.008s$
V_{min}	Minimum linear velocity	$[-10, 0, -5]^T m/s$
V_{max}	Maximum linear velocity	$[10, 15, 5]^T m/s$
e_n	Coefficient of elastic recovery	0.93
μ	Coefficient of friction	0.25
k_v	Tangent recovery coefficient	0.615
e_r	Normal recovery coefficient	0.73
k_w	Rotation conversion coefficient	2570

IV. SIMULATION VERIFICATION THROUGH VIRTUAL ENVIRONMENT

A. SETUP OF THE VIRTUAL ENVIRONMENT

In order to evaluate the performance of the stroke strategy quantitatively, we randomly generate 50,000 sets of virtual table tennis ball trajectory data in the established virtual environment. Configuration of the virtual environment is set according to Table. 1, parameters of which are determined according to the physical environment [25], [26]. The virtual environment is running on a computer with Ubuntu 16.04 OS installed. The number of LSTM states is set to 50, the number of the fully connected layers is set to 3, and the total number of parameters of the proposed networks is 22981, optimized using the method Adam [27], in which, the learning rate is set to 0.001, the first order moment coefficient is set to 0.9, and the second order moment coefficient is set to 0.999, empirically.

During the policy training process, the state of the stroke strategy is set to the first 20 points of the flying trajectory of the table tennis ball with or without noise, to mimic the ball position estimation error using the LSTM network in the real world. The action value is generated based on the estimated spin velocity before the ball hits the racket. Following the action generated by the DRL-based strategy, the racket performs the stroke accordingly. After the stroke, the reward value is obtained by calculating the deviation of the landing location of the ball and the height while it is passing the net.

To fully train the ball stroke strategy, the linear and spin velocity of the incoming ball are selected from $[V_{min}, V_{max}]$ and $[W_{min}, W_{max}]$, randomly. First, the ground truth virtual flying ball motion data is utilized to verify the proposed DRL strategy. Then, the random noise is added to the randomly generated virtual ball flight trajectory to mimic the measurement error of the flying ball position in the real world (Note that, the ball flying velocity is estimated according to the ball location in the adjacent images, thus only the position measurement error is considered in this paper). From these simulation tests, the relation between the measurement accuracy and the DRL strategy performance can be revealed.

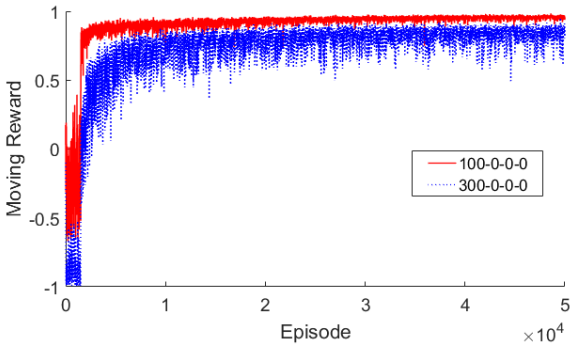

FIGURE 4. Moving reward gets improved during the training process with $W_x, W_y, W_z \in [-100, 100]$ rad/s (group-100-0-0-0) and $W_x, W_y, W_z \in [-300, 300]$ rad/s (group-300-0-0-0), respectively.

B. COMPARATIVE RESULTS OF THE DRL-BASED STRATEGY TRAINED WITH GROUND TRUTH DATA

To study the spin velocity influence on the performance of the DRL strategy, it is assumed that the linear velocity $V = [v_x, v_y, v_z]$ can be estimated exactly (noise range is set to zero for each component of V). Two groups of training data with W components selected from two predefined ranges, are randomly generated for comparison. The two groups of data are named as group-100-0-0-0 and group-300-0-0-0 to reflect the data property, e.g. group-100-0-0-0 indicates each speed of the spin velocity is selected from range $[-100, 100]$rad/s, and noise ranges for linear velocity component v_x, v_y, v_z are all set to zero.

As shown in Fig. 4, the moving rewards of the two cases keep improving during the training process at the beginning stage. After a certain number of episodes, it is hard to see obvious improvement of the DRL strategies, indicating these policies converge to the local optimal solutions. It is noted that the data with lower spin velocity (group-100-0-0-0) converges faster than that with higher spin velocity (group-300-0-0-0). Besides, the final moving reward of group-300-0-0-0 is a little lower than that of the group-100-0-0-0 with lower variance. This phenomenon should be attributed to the fact that fast spin velocity is harder to be conquered, leading to slower success experience accumulation, lower performance reward (regarding the landing accuracy and the flying height accuracy while the ball is passing the net) and higher uncertainty.

After 50, 000 training episodes, ideal reward convergence can be achieved for both policies with different training settings. The typical testing results (1000 episodes) are shown in Fig. 5. All the data are counted on the case that the ball lands within the table, and the probability of this case is denoted as η. For the group-100-0-0-0 policy, $\eta = 100\%$, and for the group-300-0-0-0 policy, $\eta = 99.5\%$. The distributions of the x/y-axis difference between actual landing location and desired location are illustrated in Fig. 5(a), (b) and (d), (e), respectively. One can see the mean landing error is pretty close to zero (at the *mm* level) for the two cases, which is ideal. When it comes to the standard deviation of landing error, group-300-0-0-0's is larger than that of the

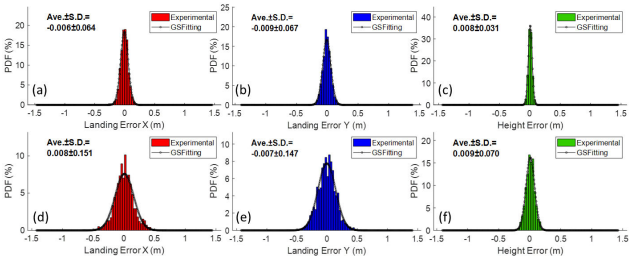

FIGURE 5. Testing results of the stroke strategy for flying ball with randomly selected spin velocity from [−100,100] rad/s and [−300,300] rad/s, respectively. (a)-(c) landing location deviation along the x/y-axis and the height deviation while the ball passing the net (policy trained with group-100-0-0-0 data), (d)-(f) performance of the policy trained with group-300-0-0-0 data.

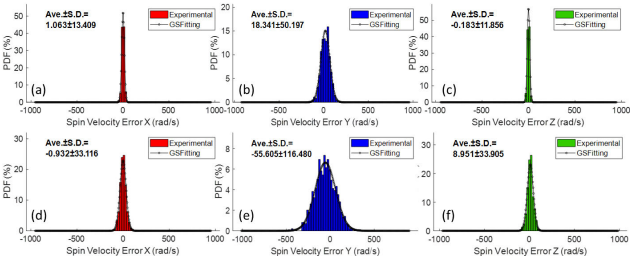

FIGURE 6. Spin velocity estimation performance for the policy trained with group-100-0-0-0 data and group-300-0-0-0 data, respectively. (a)-(c) spin velocity estimation error along the x, y, z-axis (policy trained with group-100-0-0-0 data), (d)-(f) performance of the policy trained with group-300-0-0-0 data.

group-100-0-0-0, indicating faster spin velocity is more difficult for the DRL policy to tackle. Fig. 5(c) and (f) illustrate the passing height performance while the ball flying across the net with the two DRL strategies. Similar to the landing performance, the standard deviation of group-300-0-0-0 is larger than that of group-100-0-0-0, and this should be attributed to the same reason: unconquered spin velocity generates unexpected linear velocity during rebounding, and it leads to a biased flying trajectory.

To directly demonstrate the estimation accuracy of the spin velocity from the table tennis ball flying trajectory, 1, 000 sets of random trajectory data are selected as the input of the trained DRL strategy, and the estimated ball spin velocity is compared with its theoretical value to produce the deviation distributions in three dimensions. The estimation error in three dimensions is shown in the Fig. 6, where for the group 100-0-0-0 (as shown in Fig. 6(a), (b) and (c)), the estimation error is mostly within ± 40 rad/s ($\pm 3\sigma$ range) for both x and z-axis, and the estimation error is mostly within ± 150 rad/s about y-axis. The reason for the larger error about y-axis should be attributed to that the main velocity of the table tennis ball is usually in y direction, thus the spin speed about y-axis has the weakest effect on the trajectory, which makes it more difficult to obtain during DRL strategy training. Similar results can also be found in the distributions of group-300-0-0-0 (as shown in Fig. 6(d), (e) and (f)). Consistently, the estimated spin velocity of group-300-0-0-0 shows a much wider standard deviation, leading to a lower accuracy of ball motion control.

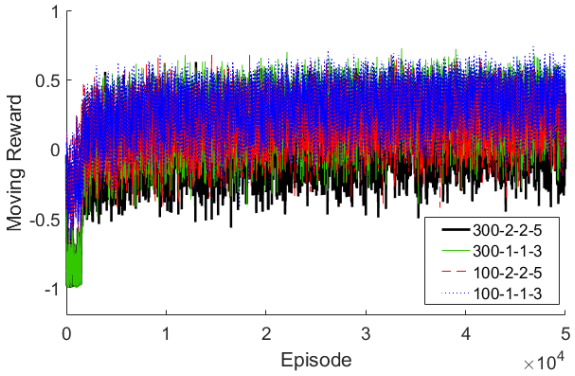

FIGURE 7. Moving rewards keep improving during the training process with the four different training settings.

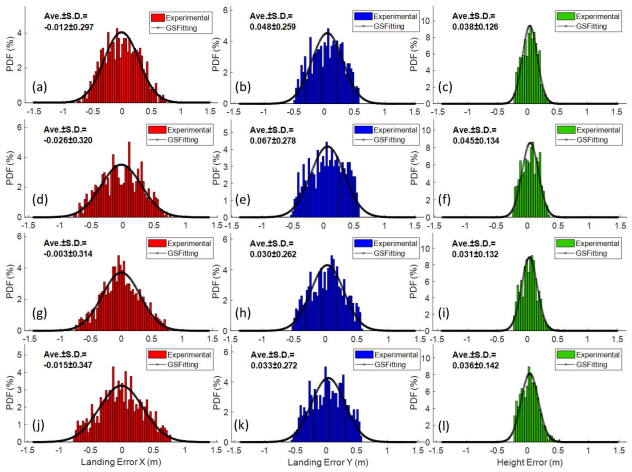

FIGURE 8. Testing performance comparison of the DRL-based stroke strategy with four different noise settings, (a)-(c) landing location deviation along the x/y-axis and the height deviation while the ball passing the net (policy trained with group-100-1-1-3 data), (d)-(f) performance of the policy trained with group-100-2-2-5 data, (g)-(i) performance of the policy trained with group-300-1-1-3 data, and (j)-(l) performance of the policy trained with group-300-2-2-5 data.

C. COMPARATIVE RESULTS OF THE DRL STRATEGY TRAINED WITH ADDITIVE NOISY DATA

As mentioned, the linear velocity estimation of an incoming ball is performed by comparing its location appeared in the two adjacent images, and the spin velocity is also evaluated based on the ball locations sequence latently. However, it is impossible for a measurement system to obtain an error-free position of the flying ball in real implementation. Therefore, random noise is added to the estimated position of the flying ball in the virtual environment. It is assumed that the additive noise conforms to the uniform distribution with probability density function expressed as (29),

$$f(u) = \begin{cases} \dfrac{1}{b-a}, & a \le u \le b, \\ 0, & \text{else}, \end{cases} \tag{29}$$

where u denotes the selected random value, and a, b represents the lower and upper boundary, respectively.

To quantitatively characterize the noise effect on the training and testing process of the DRL-based strategy, different noise settings are considered in this study. There are four different cases, denoted by group-100-1-1-3, group-100-2-2-5, group-300-1-1-3 and group-300-2-2-5, respectively. For instance, group-100-1-1-3 represents the case that components of the spin velocity and linear velocity estimation error are randomly selected from $[w_x, w_y, w_z] \in [-100, 100]$rad/s, $v_x \in [-1.0, 1.0]$mm/s, $v_y \in [-1.0, 1.0]$mm/s and $z \in [-3.0, 3.0]$mm/s, respectively. Same as the noise-free study, 50, 000 training episodes were performed to the DRL strategies with the four settings above. The reward curves obtained during the training process are illustrated in Fig. 7.

As shown in Fig. 7, the reward curves of the DRL method of the four different settings keep climbing along with training episodes, and they converge quickly at the early stage, similar to that of the noise-free case. However, the added random estimation errors influence the DRL performance seriously, which increases the variance and lowers the final reward, indicating a larger trajectory deviation for the rebounded ball. Based on the training performance shown in Fig. 7, it can be seen that the larger spin velocity and heavier noise lead to worse training performance. By comparing the similar performance of the group-100-2-2-5 policy and that of the group-300-1-1-3 strategy, it can be concluded that the proposed DRL strategy is more sensitive to position estimation error in contrast to the range effect of spin velocity, since spin velocity range altered by two folds while the position error changed by around one fold.

To conduct a detailed performance comparison, the landing and net passing errors of the four cases are illustrated in Fig. 8. For the policy trained with group-100-1-1-3 data, the ball-inside-table probability is $\eta = 89\%$; for group-100-2-2-5, $\eta = 83.5\%$; for group-300-1-1-3 $\eta = 83.3\%$, and for the policy trained with group-300-2-2-5 data, $\eta = 73.9\%$, indicating lowest performance of the policy trained with group-300-2-2-5 data. However, when it comes to the variance of error distributions shown in Fig. 7, the four settings demonstrate similar performances, which are far behind the performances of policies trained with ground truth data (as shown in Fig. 5). Specifically, for the x-axis, the standard deviation of the landing location error is around 0.32 m; for y-axis, it is around 0.26 m, and for the net passing error, it is around 0.13 m. These testing results may be explained as that once the DRL strategy can correctly estimate the incoming ball rough status, it can perform robust action to stroke the ball to the desired area.

To illustrate the accuracy of the DRL strategy in estimating the spin velocity of the ball with noise, the distributions of the error between the estimated and the actual spin velocity are shown in Fig. 9. From the estimation error during the test one can see the standard deviation in the direction x and the direction z is around 42 rad/s and 54 rad/s, which is uniform for the four noise settings and demonstrates the robustness of the DRL-based strategy. When it comes to the y direction, the estimation error standard deviation increases by around 1 fold for group-100-*-*-* settings and about 2 folds for

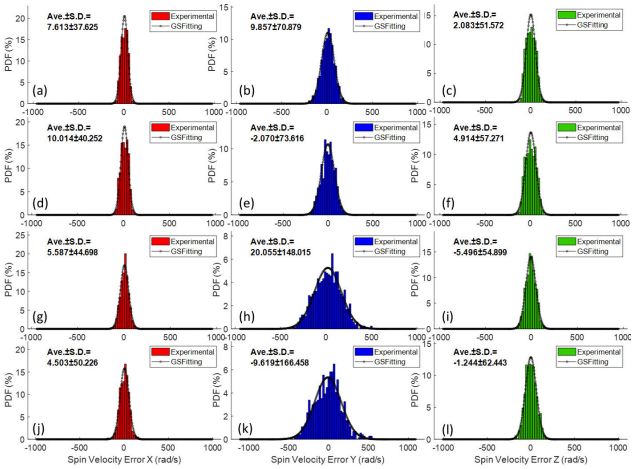

FIGURE 9. Spin velocity estimation performance for the policy trained with the four noise settings, (a)-(c) spin velocity estimation error along the x, y, z-axis (policy trained with group-100-1-1-3 data), (d)-(f) performance of the policy trained with group-100-2-2-5 data, (g)-(i) performance for group-300-1-1-3, and (j)-(l) performance for group-300-2-2-5.

TABLE 2. Main parameters of table tennis robot.

DOFs	6
TCP Max Speed	10m/s
Reaching Area	Whole Table
Smash Time	0.15s
Stereo Vision FPS	200Hz
Prediction Precision	0.02m
Ball Recognition Rate	99.9%

group-300-*-*-* settings. Although the spin velocity estimation error of y direction is wider and the robustness is lower, it has a weak influence on the flying ball trajectory.

To summarize, compared with the influence from the larger spin velocity range, the position estimation error of the flying ball is much more detrimental for the proposed DRL-based strategy, leading a robot to stroke the ball outside the table more frequently.

V. EXPERIMENTAL STUDY BASED ON THE TABLE TENNIS PLAYING ROBOT PLATFORM

A. SETUP OF THE TABLE TENNIS ROBOT PLATFORM

In order to verify the effectiveness of the proposed method in the real implementation, the virtual data-based pre-trained DRL strategy is applied to the table tennis robot developed by the research team, and table tennis balls with various types of spin velocity are served to the table tennis robot by a serving robot. Specifications of the table tennis robot are listed in Tab. 2. In this study, the stereo vision FPS is set to 200 Hz, which is triggered by a hardware counter. Based on this setup, the incoming ball motion can be captured and processed within 5 ms.

As shown in Fig. 10, the experimental platform consists of a standard table for playing table tennis, a table tennis robot and a serving robot. The high speed binocular stereo vision module in the table tennis robot can capture the scene images and estimate the position of the flying ball, and an RNN module is employed to predict the trajectory of the flying ball

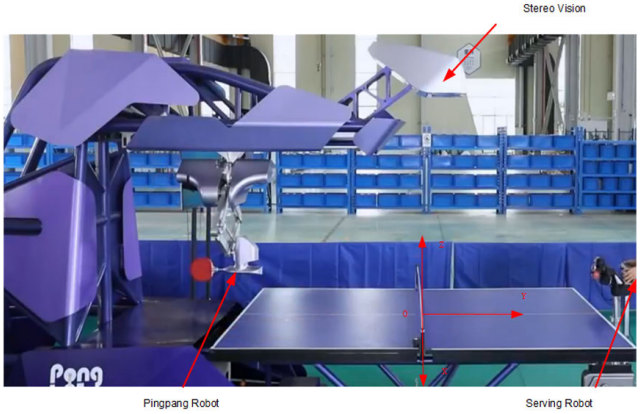

FIGURE 10. Experimental platform for testing table tennis ball stroke strategies in a self-made table tennis robot system. (Details of testing experiments can be found in the supplementary material: video1).

in real-time. A serving robot is used for serving balls to the table tennis robot with various spin velocities continuously. A world coordinate system for the vision system is defined the same as that shown in Fig. 1.

B. COMPARATIVE TESTS OF THE TABLE TENNIS ROBOT

To verify the effectiveness of the proposed method, the table tennis ball striking strategy proposed in this paper is implemented in the table tennis robot, and two sets of experiments were carried out. One of the two settings uses the proposed DRL-based strategy, comprised of time-series-based DDPG policy and racket motion control module, to estimate the spin velocity of the incoming ball in real time and performs the stroke action. The other setting employs only the racket motion control module and sets spin velocity to a constant value, specifically, 80 rad/s (average spin speed of the incoming balls) for this test. The serving robot is controlled to serve various types of spin to the table tennis robot, including up spin, backspin and sidespin. A stereo vision system is used to capture the landing position and the height of the ball while it is passing the net after stricken back by the table tennis robot. The DRL-based method is fine-trained based on the pre-trained network when implemented in the table tennis robot. During the fine-training process, the reward is calculated based on the difference between expected and actual landing position and passing-net height captured in real time. The training result is shown in Fig. 11, where one can see the lower boundary of the reward has been improved by the fine training.

During the experiment, for the traditional method, 100 tests were conducted using the serving robot, and for the DRL-based method, 292 tests were carried out. The reason for more testing sets on the DRL-based method is that the neural networks-based method is non-transparent compared with the traditional dynamics-based approach, thus it needs more tests to perform fare comparison. For the traditional method, there are only 16 times having the ball landing within the table ($\eta = 16\%$); for the DRL-based method, there are 210 times, which reaches $\eta = 71.9\%$.

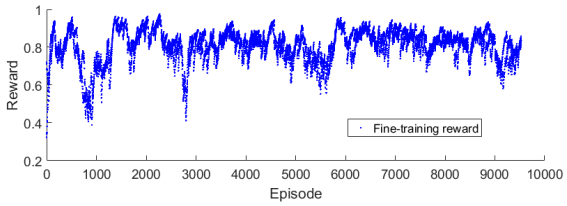

FIGURE 11. Moving reward curve of the fine training using the table tennis robot.

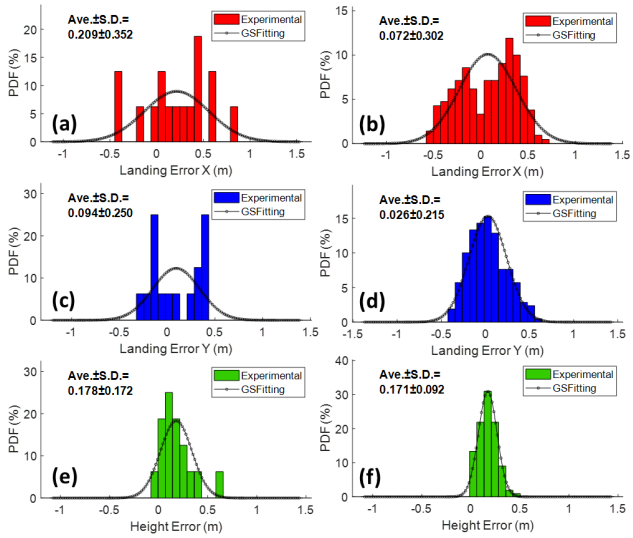

FIGURE 12. Testing performance comparison of table tennis robot with two settings, (a)-(c) landing location deviation along the x/y-axis and the passing-net height error of the traditional method, (d)-(f) performance of the proposed DRL-based ball stroke strategy.

Comparison is further performed to the landing-within-table data. The obtained deviation distributions for both settings are shown in Fig. 12, where one can see the DRL-based method demonstrates better performance in terms of the mean value and the standard deviation of the landing error shown in Fig. 12(a) and (d). For the passing-height accuracy, though the distribution of the DRL-based method is narrower than that of the traditional method, the mean values show obvious positive bias for both methods. This is because the DRL-based strategy prefers to stroke the ball a little higher to avoid hitting the net when there is environmental uncertainty in the real world. Otherwise, this episode is prone to get the lowest reward as it has a larger chance to hit the net. A similar reason to the traditional method, the higher-flying balls have a better chance to pass the net, thus it shows obvious positive bias in the distributions of Fig. 12(c) and (f).

VI. CONCLUSION

A DRL-based table tennis ball playing strategy considering the spin velocity of the incoming ball is proposed in this paper. To fulfill the training process of the DRL-based policy, a virtual table tennis robot environment is developed, which is utilized to meet the data diversity requirement of DRL type methods. Based on the virtual environment, different noise settings were adopted to train the DRL method, and associated analysis on the influence of noise on the accuracy of landing and height over net is carried out. The simulation tests reveal that the ball's position estimation error is as detrimental as that of the spin velocity estimation error. The experimental results show that the proposed motion control approach can effectively stroke back the ball to the desired area. Compared with the traditional method, the DRL-based method shows significant improvement in playing back the ball (e.g. landing-within-table probability: 70% *vs*. 16%, x-axis average landing deviation: 72 mm *vs*. 209 mm). Moreover, the proposed method does not require making additional markers on a ball, thus the simple flying ball trajectory-based technique makes practical implementations more possible than traditional ways.

In the future, we are going to focus on improvement of the returning ball strategy, which will employ faster and more accurate flying ball behavior prediction methods, and this will be able to save more time for the robot physical system to execute precise stroke. Besides, player motion analysis-based returning ball strategy will be investigated to get better chance of surpassing highly skilled human players.

REFERENCES

[1] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

[2] S. Gomez-Gonzalez, G. Neumann, B. Schölkopf, and J. Peters, "Adaptation and robust learning of probabilistic movement primitives," *IEEE Trans. Robot.*, vol. 36, no. 2, pp. 366–379, Apr. 2020.

[3] H.-I. Lin, Z. Yu, and Y.-C. Huang, "Ball tracking and trajectory prediction for table-tennis robots," *Sensors*, vol. 20, no. 2, p. 333, Jan. 2020.

[4] S. Gomez-Gonzalez, S. Prokudin, B. Schölkopf, and J. Peters, "Real time trajectory prediction using deep conditional generative models," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 970–976, Apr. 2020.

[5] I. Kovacs, A. McClinton, C. Rauenzahn, and W. Liu, "Table tennis training results with robot: Spin rate and hitting speed in forehand loop-drives: 3479: Board# 167 June 1 8:00 am-9:30 am," *Med. Sci. Sports Exerc.*, vol. 51, no. 6, pp. 956–957, 2019.

[6] C. Liu, Y. Hayakawa, and A. Nakashima, "A registration algorithm for on-line measuring the rotational velocity of a table tennis ball," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, Dec. 2011, pp. 2270–2275.

[7] J. Tebbe, L. Klamt, Y. Gao, and A. Zell, "Spin detection in robotic table tennis," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 9694–9700.

[8] S. Gomez-Gonzalez, Y. Nemmour, B. Schölkopf, and J. Peters, "Reliable real-time ball tracking for robot table tennis," *Robotics*, vol. 8, no. 4, p. 90, 2019.

[9] A. Nakashima, T. Okamoto, and Y. Hayakawa, "An online estimation of rotational velocity of flying ball via aerodynamics," in *Proc. IFAC World Congr.*, 2014, pp. 1–6.

[10] A. Nakashima, Y. Ogawa, C. Liu, and Y. Hayakawa, "Robotic table tennis based on physical models of aerodynamics and rebounds," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, Dec. 2011, pp. 2348–2354.

[11] Y. Huang, D. Xu, M. Tan, and H. Su, "Trajectory prediction of spinning ball for ping-pong player robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 3434–3439.

[12] A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine, "Visual reinforcement learning with imagined goals," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9191–9200.

[13] X. Wu, S. Liu, T. Zhang, L. Yang, Y. Li, and T. Wang, "Motion control for biped robot via DDPG-based deep reinforcement learning," in *Proc. WRC Symp. Adv. Robot. Autom. (WRC SARA)*, Aug. 2018, pp. 40–45.

[14] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," 2017, *arXiv:1707.08817*. [Online]. Available: http://arxiv.org/abs/1707.08817

[15] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5048–5058.

[16] I. Popov, N. Heess, T. Lillicrap, R. Hafner, G. Barth-Maron, M. Vecerik, T. Lampe, Y. Tassa, T. Erez, and M. Riedmiller, "Data-efficient deep reinforcement learning for dexterous manipulation," 2017, *arXiv:1704.03073*. [Online]. Available: http://arxiv.org/abs/1704.03073

[17] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018, *arXiv:1801.01290*. [Online]. Available: http://arxiv.org/abs/1801.01290

[18] Y. Hou, L. Liu, Q. Wei, X. Xu, and C. Chen, "A novel DDPG method with prioritized experience replay," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2017, pp. 316–321.

[19] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, D. TB, A. Muldal, N. Heess, and T. Lillicrap, "Distributed distributional deterministic policy gradients," 2018, *arXiv:1804.08617*. [Online]. Available: http://arxiv.org/abs/1804.08617

[20] S. Fujimoto, H. Van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.

[21] T. Xu, Q. Liu, L. Zhao, and J. Peng, "Learning to explore with meta-policy gradient," 2018, *arXiv:1803.05044*. [Online]. Available: http://arxiv.org/abs/1803.05044

[22] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," 2017, *arXiv:1710.06542*. [Online]. Available: http://arxiv.org/abs/1710.06542

[23] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1329–1338.

[24] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, and J. Schneider, "Learning dexterous in-hand manipulation," *Int. J. Robot. Res.*, vol. 39, no. 1, pp. 3–20, 2020.

[25] A. Nakashima, Y. Ogawa, Y. Kobayashi, and Y. Hayakawa, "Modeling of rebound phenomenon of a rigid ball with friction and elastic effects," in *Proc. Amer. Control Conf.*, Jun. 2010, pp. 1410–1415.

[26] C. Liu, Y. Hayakawa, and A. Nakashima, "Racket control and its experiments for robot playing table tennis," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2012, pp. 241–246.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[28] Y. Zhang, R. Xiong, Y. Zhao, and J. Wang, "Real-time spin estimation of ping-pong ball using its natural brand," *IEEE Trans. Instrum. Meas.*, vol. 64, no. 8, pp. 2280–2290, Aug. 2015.

[29] W. Zhang, J. Li, Q. Huang, Z. Yu, X. Chen, G. Ma, L. Meng, Y. Liu, S. Zhang, F. Meng, W. Zhang, and J. Gao, "System design of a 9-DOF robot capable of fast and flexible rally task," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, Aug. 2014, pp. 1428–1433.

[30] Y. Ren, Z. Zhao, C. Zhang, Q. Yang, and K.-S. Hong, "Adaptive neural-network boundary control for a flexible manipulator with input constraints and model uncertainties," *IEEE Trans. Cybern.*, early access, Oct. 1, 2020, 10.1109/TCYB.2020.3021069.

[31] Y. Zhu, Y. Zhao, L. Jin, J. Wu, and R. Xiong, "Towards high level skill learning: Learn to return table tennis ball using monte-carlo based policy gradient method," in *Proc. IEEE Int. Conf. Real-Time Comput. Robot. (RCAR)*, Aug. 2018, pp. 34–41.

[32] O. Koç, G. Maeda, and J. Peters, "Online optimal trajectory generation for robot table tennis," *Robot. Auton. Syst.*, vol. 105, pp. 121–137, Jul. 2018.

[33] H. Zhang, Z. Fu, and K.-I. Shu, "Recognizing ping-pong motions using inertial data based on machine learning classification algorithms," *IEEE Access*, vol. 7, pp. 167055–167064, 2019.

[34] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," 2015, *arXiv:1506.04214*. [Online]. Available: http://arxiv.org/abs/1506.04214

[35] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," 2018, *arXiv:1805.12114*. [Online]. Available: http://arxiv.org/abs/1805.12114

[36] Y. Luo, K. Dong, L. Zhao, Z. Sun, E. Cheng, H. Kan, C. Zhou, and B. Song, "Calibration-free monocular vision-based robot manipulations with occlusion awareness," *IEEE Access*, vol. 9, pp. 85265–85276, 2021.

LUO YANG received the M.E. degree from the East China University of Science and Technology. He is currently pursuing the Ph.D. degree in mechanical engineering with Shanghai Jiao Tong University. As a member of SIASUN Company Ltd., he leads the Research Team to develop the collaborative robot and sport robot system. His research interests include robotics and deep reinforcement learning.

HAIBO ZHANG received the M.E. degree from China Jiliang University. As a member of Pongbot Company Ltd., he is currently cin charge of developing the table tennis robot system. His research interests include robotics and computer vision.

XIANGYANG ZHU received the B.S. degree in automatic control engineering from the Nanjing Institute of Technology, Nanjing, China, in 1985, and the M.Phil. degree in instrumentation engineering and the Ph.D. degree in automatic control engineering from Southeast University, Nanjing, in 1989 and 1992, respectively.

From 1993 to 1994, he was a Postdoctoral Research Fellow with the Huazhong University of Science and Technology, Wuhan, China. He joined the Department of Mechanical Engineering, Southeast University, as an Associate Professor, in 1995. Since June 2002, he has been with the School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China, where he is the Changjiang Chair Professor, the Director of the State Key Laboratory of Mechanical System and Vibration, and the Director of the Robotics Institute. His current research interests include robotic manipulation planning, human–machine interfacing, and biomechatronics. He received the National Science Fund for Distinguished Young Scholars Award, in 2005.

XINJUN SHENG (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees from the School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2000, 2003, and 2014, respectively. He was a Visiting Scientist with Concordia University, Montreal, QC, Canada, in 2012. He is currently an Associate Professor with the School of Mechanical Engineering, Shanghai Jiao Tong University. His current research interests include robotics and biomechatronics. He is a member of the IEEE Robotics and Automation Society, IEEE Engineering in Medicine and Biology Society, and IEEE Industrial Electronics Society.