Improved high-speed vision system for table tennis robot

Jianran Liu , Zaojun Fang, Kun Zhang, and Min Tan
State Key Laboratory of Management and Control for Complex Systems
Institute of Automation, Chinese, Academy of Sciences
Beijing 100190, China.
Email: {liujianran2012,zaojun.fang, zhangkun2012, min.tan}@ia.ac.cn

***Abstract*—In this paper a high-speed vision system for table tennis robot is designed. The system architecture is designed firstly. Then a ball detection algorithm is proposed to improve the image processing speed up to 200 frames per second. Besides, a novel ball trajectory reconstruction algorithm is established, in which the cameras in the stereo vision system needn't be synchronized. Experimental results verify the effectiveness and accuracy of the proposed method.**

***Index Terms*—High-speed stereovision, table tennis robot, target recognition, visual measurement, visual tracking.**

I. Introduction

Table tennis robot has been attracting many researchers' attentions. It is a complex system which involves high speed vision procession, model identification and motion control. Since 1980s, many research groups have focus on this question and quite a lot of works have been done[1–7]. However, in the vision systems of the table tennis robots mentioned above, the image processing speed is limited below 100 frames per second (FPS), which is far from dealing with the practical human-playing situation. When playing with a human, the ball can fly as fast as 20m/s. Since the length of the table is only 2m, the robot has only less than 0.1s to react to the coming ball. A more specific condition is that when dealing with coming ball with rotation, more data is needed to improve the robustness and accurateness of the measuring system. Thus, the vision system with higher speed is required to improve the performance of the existing table tennis robot.

In order to increase the vision processing speed and get data more efficiently, some problems need to be solved. Firstly, the image processing algorithm needs to be improved in order to reduce its time complexity. Secondly, the short processing time for each image requires short shutter time, which will lead to poor image quality. The reason is that when the shutter time is short, little light would get into the lens, then the signal to noise ratio would decrease.

Unfortunately, the commonly used moving object detection and tracking algorithms can not overcome the obstacles listed above. For example, given that the images are severely unstable, we can not get the accurate ball positions through the simple algorithms like adjacent frame difference. Meanwhile, though we can get relatively accurate and robust results through complex algorithms like Gaussian Mixture Model (GMM), we can not afford the considerable time complexity[8].

Another point that can be improved is the algorithm for three dimensional (3-D) ball trajectory reconstruction. In traditional 3-D trajectory reconstruction algorithm, it's critical to ensure that two cameras are synchronized, which means that only when each camera detects the ball at exactly the same time can we get the 3-D position of the ball. This restriction would weaken the vision system both in real-time performance and in validity of the trajectory prediction.

In this paper, we will present a vision system in which the problems mentioned above are solved. In section II, the vision system structure is described from hardware and software aspects. In section III, the ball detection algorithms would be proposed. Then a new 3-D trajectory reconstruction algorithm would be introduced in section IV. Experiments are given to verify the effectiveness of the algorithms in section V. Finally, this paper is concluded in section VI.

II. System architecture

A. Hardware Architecture

The proposed stereovision system is improved from [1], which contains two smart cameras and a personal computer (PC). The communication between each part of the system is based on TCP/IP protocol. The scheme diagram is shown in Fig. 1.

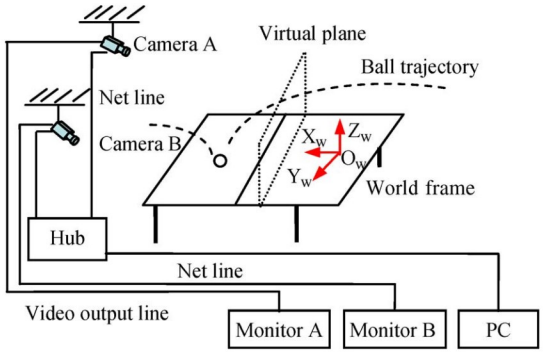

Fig. 1. System hardware architecture [1]

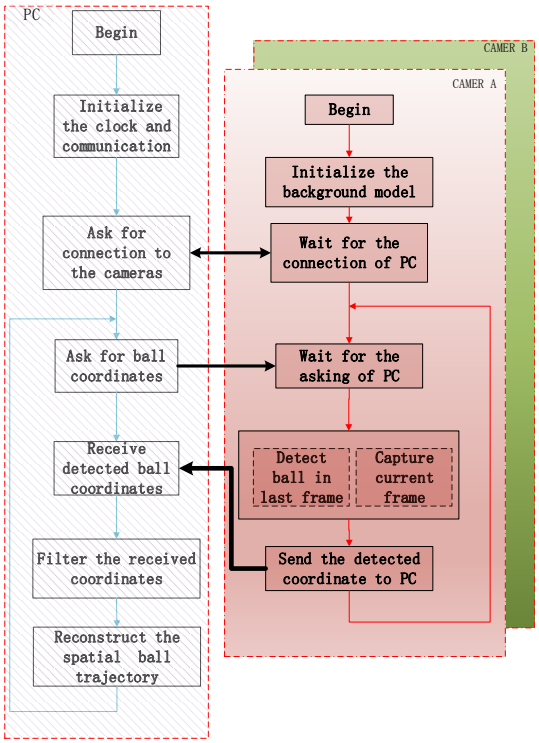

Fig. 2. The flow chart of the software

B. Software Architecture

As described in Section I, the 3-D reconstruction algorithm proposed in this paper does not ask for synchronization. The hand-shaking between two cameras is no longer needed. So the software architecture is greatly simplified from [1]. The cameras process the images and detect the positions of the ball independently. Given that the smart camera has two kernels, the capturing of new image and processing of last image can be handled in parallel, which would improve the real-time performance of the system. The results from both cameras are sent to the PC for reconstruction the 3-D trail of the ball. The flowchart of the software is shown in Fig. 2. Before the running of robot, the connection between PC and cameras would be established first. During the system running, the PC would keep asking for new ball coordinates from the cameras. Meanwhile, once the cameras receive the asking from PC, they would try to find out the ball coordinate in the captured frame, and send the results back to the PC. Then the PC would reconstruct the 3-D trajectory with received data.

III. Ball Detection Algorithm

To achieve the processing speed as high as 200 FPS, a set of algorithms for ball recognition and tracking is proposed. The algorithms are improved from the Single Gaussian model [9], so they share the same assumption that

$$I(x,y,k) \sim N\left(u(x,y), \delta^2(x,y)\right) \tag{1}$$

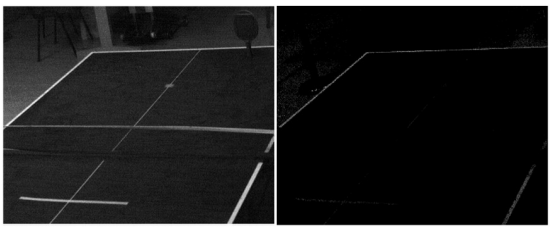

Fig. 3. background model. (a) is I_u and (b) is I_D

where $I(x,y,k)$ is the gray value of the pixel at (x,y) in the $k-th$ image, and $u(x,y)$, $\delta^2(x,y)$ are the corresponding expectation and variance. Based on this assumption, the details of the algorithms are described as follows.

A. Initialization of the background model

At the initialization stage, 15 pictures of clean scene without balls and players are taken. Then the initial $u(x,y)$ and $\delta^2(x,y)$ are estimated through the mean and second moment of each pixel. Image I_u and I_D are created with $u(x,y)$ and $\delta^2(x,y)$ as their pixel brightness at (x,y). An example is presented in Fig. 3.

B. Rough locating the ball

In order to detect the areas with moving objects effectually, adjacent frame difference is used for rough locating the ball.

According to (1), we have

$$\begin{cases} I(x,y,k) \sim N\left(u(x,y), \delta^2(x,y)\right) \\ I(x,y,k-1) \sim N\left(u(x,y), \delta^2(x,y)\right) \end{cases} \tag{2}$$

So the adjacent frame difference can be given as

$$dI(x,y,k) = I(x,y,k) - I(x,y,k-n) \tag{3}$$

As we can see, $dI(x,y,k)$ is a random variable whose variance can be written as

$$\begin{aligned} D(dI(k)) &= D(I(k) - I(k-n)) \\ &= D(I(k)) + D(I(k-n)) - 2cov(I(k), I(k-n)) \\ &= 2\delta^2 - 2cov(I(k), I(k-n)) < 2\delta^2 \end{aligned} \tag{4}$$

Though it cannot been ensured that $dI(x,y,k)$ obeys normal distribution, it is verified by experiments that the rough position of moving objects can be labeled by binarizing the image dI with a threshold $3\sqrt{D(dI)}$, i.e.,

$$I_b(x,y) = \begin{cases} 1, & |dI(x,y,k)| > 3\sqrt{2}\delta(x,y) \\ 0, & |dI(x,y,k)| \le 3\sqrt{2}\delta(x,y) \end{cases} \tag{5}$$

where I_b is the possible zone of moving objects.

Fig. 4 is an example of I_b , from which it can be observed that besides the ball, other moving objects such as moving human body are also detected. Since most of the moving objects would not move as fast as the ball, the disturbance trace would be much narrower compared to the trace of the ball. Hence such distribution can be ruled out simply by morphological processing:

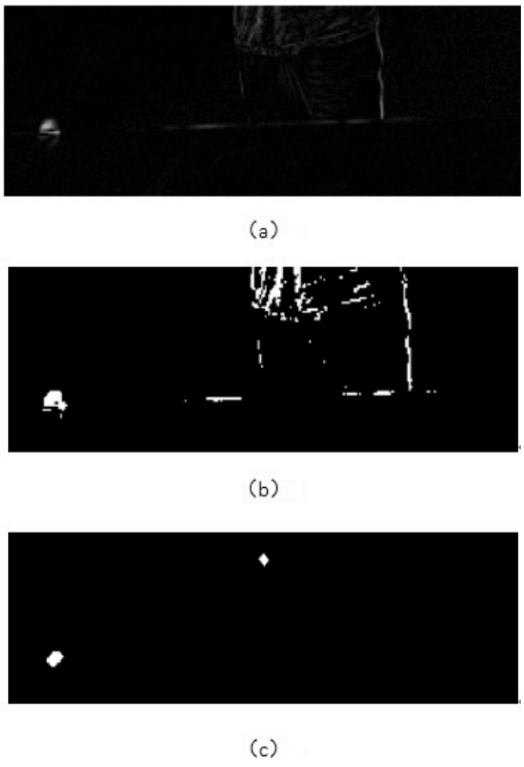

Fig. 4. Results of the ball rough locating . (a) Adjacent frame difference result . (b) Possible zone of the ball . (c) Rough position of the ball

$$I_r = (I_b \oplus B) \ominus C \tag{6}$$

where B is a pixel while C is a square with the size of 3×3 in pixel, and binary image I_r indicates the rough position of the ball in the image.

C. Accurate locating the ball

In section III-B we get the rough position of the ball, which is denoted as P_r . Then the accurate position of the ball is searched in a square window around P_r with the size of $R_w \times R_w$ in pixel.

We define dI_b as the difference of the current image I and the background expectation I_u. Then we binaries dI_{bkg} with threshold $3\sqrt{I_D}$ as follows:

$$dI_{bin}(x,y) = \begin{cases} 1, & |dI_b(x,y)| - 3\sqrt{I_D(x,y)} > 0) \\ 0, & others \end{cases} \tag{7}$$

In the image dI_{bin} , the ball is supposed to be a continuous block with circular contour and with area in a certain range. Based on this, it can be judged that whether we get the exact window which contains the ball through some verification.

We define a set W as

$$W = (x,y)|dI_{bin}(x,y) = 1 \tag{8}$$

then we have

$$\begin{cases} \bar{x} &= \frac{1}{N} \sum\limits_{(x,y)\in W} x \\ \bar{y} &= \frac{1}{N} \sum\limits_{(x,y)\in W} y \end{cases} \tag{9}$$

$$\begin{cases} D_x = \frac{1}{N} \sum\limits_{(x,y)\in W} (x-\bar{x})^2 \\ D_y = \frac{1}{N} \sum\limits_{(x,y)\in W} (y-\bar{y})^2 \end{cases} \tag{10}$$

where N is the size of W, $(\bar{x}, \bar{y})$ is the mean position of W, D_x and D_y are the distribution variances of W. The accurate position of the ball is $(\bar{x}, \bar{y})$ if the following inequations are satisfied.

$$\begin{cases} T_l < N < T_h \\ D_x < T_x, D_y < T_y \end{cases} \tag{11}$$

D. Dynamic tracking and background update

To further improve the real-time performance of the image processing, the dynamic window technique described in [1] is used to track the ball.

To ensure the robustness of the algorithm, the background needs to be updated as described in [9]. However, updating the model in every frame costs too much computation. To solve this problem, the model is updated only when the following two conditions are satisfied:

a) The system fails to detect the ball for successive N frames.

b) The time gap between two updated frames is more than 1 s.

E. Ball detecting algorithm scheme

The scheme of ball detecting algorithm is presented in Fig. 5. Firstly the searching window would be chosen, then a list of doubtable rough ball coordinates in the window would be established with the method proposed in section III-B. For each element in the list the algorithm proposed in section III-C would be applied to get the accurate ball location.

IV. Space reconstruction of ball trajectory

A new 3-D trajectory reconstruction algorithm is proposed in this paper, which does not need to ensure the synchronization of the two cameras. In this section the geometry explanation of the theory is introduced first, then the mathematic deduce about the algorithm is presented. At last, some improvements for applying this algorithm to the table tennis robot system is discussed.

A. Geometry explanation of the algorithm

In Fig. 5, the green line is the trace of a flying ball, O_1 and O_2 are the optic center of the two cameras. $P_1 \sim P_6$ are the ball positions in the different time, $P_1' \sim P_6'$ are at the corresponding image coordinates detected by the two cameras. Given the parameters of the cameras and detected $P_1' \sim P_6'$, we can get the $3D$ straight lines $l_1 \sim l_6$. Suppose that in a

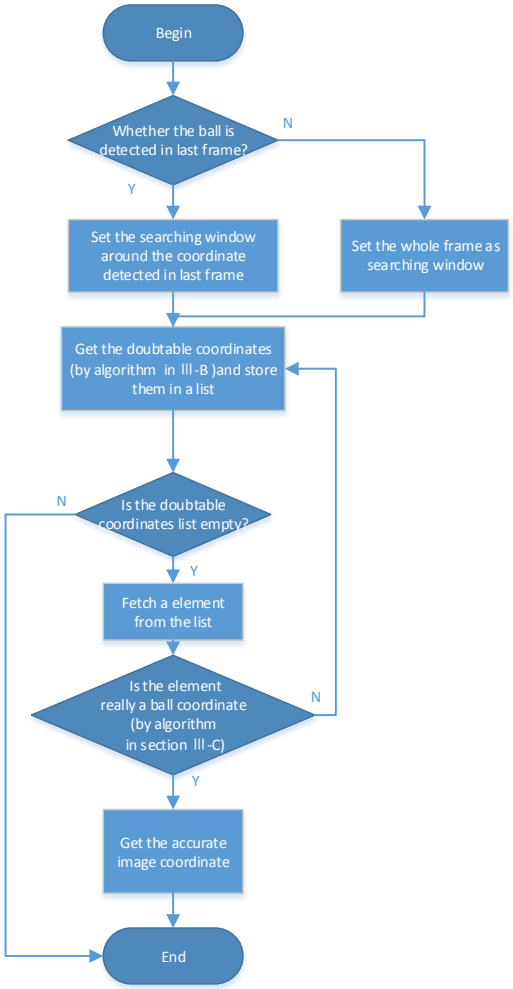

Fig. 5. The scheme of Image processing algorithm

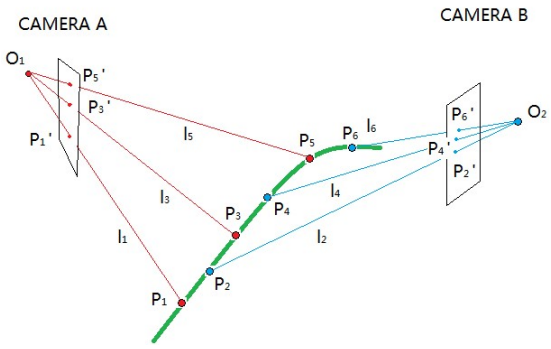

Fig. 6. Schematic plot of a ball cross the view filed of the cameras

short time period, the trace of the ball, can be approximated as a n-order curve as follows:

$$\begin{aligned} P(t) &= \begin{bmatrix} P_x(t) \\ P_y(t) \\ P_z(t) \end{bmatrix} = \begin{bmatrix} a_0 + a_1 t + \cdots + a_n t^n \\ b_0 + b_1 t + \cdots + b_n t^n \\ c_0 + c_1 t + \cdots + c_n t^n \end{bmatrix} \\ &= \begin{bmatrix} a_0 & a_1 & \cdots & a_n \\ b_0 & b_1 & \cdots & b_n \\ c_0 & c_1 & \cdots & c_n \end{bmatrix} \begin{bmatrix} 1 \\ t \\ \vdots \\ t_n \end{bmatrix} \\ &= F \cdot T(t) \end{aligned} \tag{12}$$

where t denotes time, F is the coefficient matrix. Given the fact that the trace of the ball must intersects $l_1 \sim l_6$ at points $P_1 \sim P_6$, the reconstruction of the ball trace can be transformed as

$$F_o = arg \min_F \sum D_i$$

where D_i is the distance between $P(t)$ and l_i at the moment t_i.

B. Mathematic deduce

According to Section IV-A, its theoretically possible to reconstruct the ball trace without synchronizing the two cameras. Suppose the parameters of the two cameras are given as

$$M_1 = \begin{bmatrix} m_{11} \\ m_{12} \\ m_{13} \end{bmatrix}, M_2 = \begin{bmatrix} m_{21} \\ m_{22} \\ m_{23} \end{bmatrix} \tag{13}$$

where m_{ij} are 1×3 vectors.

We can get the 3-D position of the object by the least square method (LSM) based on the following equations:

$$\begin{cases} \overline{u_1} = \dfrac{A_{11}}{A_{13}}, \ v_1 = \dfrac{A_{12}}{A_{13}} \\ \overline{u_2} = \dfrac{A_{21}}{A_{23}}, \ v_2 = \dfrac{A_{22}}{A_{23}} \end{cases} \tag{14}$$

where $A_{ij} = m_{ij}P$, P is the 3-D position of the object, (u_1, v_1) and (u_2, v_2) are the detected image positions by the two cameras, respectively [10].

Now consider the situation that Camera A detects the ball and gets its image position $(u_1(t_1), v_1(t_1))$ at the moment t_1. By combining the (12) and (14), we have

$$\begin{cases} \overline{u}_1 = \dfrac{m_{11}FT(t_1)}{m_{13}FT(t_1)} \\ \overline{v}_1 = \dfrac{m_{12}FT(t_1)}{m_{13}FT(t_1)} \end{cases} \tag{15}$$

Similarly, if Camera B detects the ball at different moment t_2, we would have

$$\begin{cases} \overline{u}_2 = \dfrac{m_{21}FT(t_2)}{m_{23}FT(t_2)} \\ \overline{v}_2 = \dfrac{m_{22}FT(t_2)}{m_{23}FT(t_2)} \end{cases} \tag{16}$$

From (15) and (16), it can be seen that as long as one of the two cameras detects the ball, we can get one more equation like (15) or (16).

Since our goal is to figure out F which has $3(n+1)$ variables, we need $3(n+1)$ independent equations at least. This can be satisfied if the two cameras detect the ball for more than $3(n+1)/2$ times in a period short enough. F is computed via LSM after enough sampling points are captured.

C. Applying the algorithm to the table tennis robot

1) Choice of n and using of data : Though bigger n means more accuracy, more sampling points need to be captured, which would cost much time. However, the algorithms proposed above are all based on (12), which is only valid when the sampling points are gathered in a relatively short time. Considering this, n is chosen to be 1, and F is calculated by the newest image coordinates detected in the past $100ms$.

2) Image coordinates preprocessing: The image coordinates of the ball may contain errors caused by image processing and disturbance. So the data from smart cameras needs to be preprocessed.

Since n is chosen to be 1, $T(t)$ in (12) can be written as $T(t) = [1 \quad t]^T$. Apply this to (15), we have

$$u_1(t) = \frac{k_{11} + k_{12}t}{k_{31} + k_{32}t}, \quad v_1(t) = \frac{k_{21} + k_{22}t}{k_{31} + k_{32}t} \tag{17}$$

where

$$\begin{cases} [k_{11} \quad k_{12}] = m_{11}F \\ [k_{21} \quad k_{22}] = m_{12}F \\ [k_{31} \quad k_{32}] = m_{13}F \end{cases} \tag{18}$$

Equation (17) can be transformed as a homogeneous equation as

$$\begin{bmatrix} u_1 & u_1t & -t & -1 \\ v_1 & v_1t & -t & -1 \end{bmatrix} \begin{bmatrix} k_{31} & k_{31} \\ k_{32} & k_{32} \\ k_{12} & k_{22} \\ k_{11} & k_{21} \end{bmatrix} = [0 \quad 0] \tag{19}$$

When the trajectory of the ball can be approximated as (12), its corresponding image coordinates in one camera should be roughly subject to (19).

For each camera an image coordinates queue is created whose length is fixed. Once a new ball coordinate is detected, the queue is updated and verified using (19) in $RANSAC$ method. Only when the deviation is acceptable would the data in the queue be used for the algorithm proposed in IV-B, otherwise it would be given up and a new data is waited for.

V. Experiments and results

To validate the performance of proposed methods, a robot vision system is built. In the system the ball detection is realized in two smart camera, while the 3-D trajectory reconstruction is

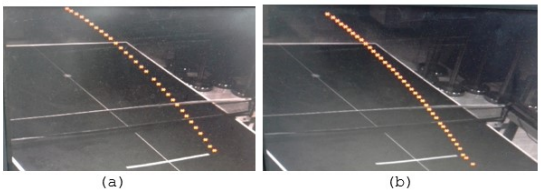

Fig. 7. Ball detection results using two methods, (a) the method proposed in [1] (b) the method proposed in this paper

A. Ball detection in real time

To do the comparison experiments using two methods proposed in [1] and this paper, similar ball trajectories were generated by a ball serve machine. Successive ball image coordinates were computed when it flied from one court of the table to the other using the algorithm proposed in this paper and that proposed in [1], respectively. The result is presented in Fig. 7. It can be observed that for similar trajectories ,the number of coordinates in our algorithm is about 1.5 times that in the algorithm proposed in [1] got, which means better real-time performance.

In the experiments, the image processing time for global search was about $10ms$ and that for local search with dynamic window was reduced to about $5ms$ as expected. Meanwhile the accuracy is well guaranteed. The real time performance and the image processing accuracy can meet the requirements of the table tennis robot.

B. 3-D trajectory reconstruction

To verify the trajectory reconstruction algorithm, a complete trajectory was recorded.

The output of the trajectory reconstruction algorithm is a series of trajectory coefficients, i.e., a series of F. To make the results more intuitive, we figured out the spatial coordinates and velocities of the ball according to the obtained F and the corresponding t with (12). The results are presented in Fig. 8. From Fig. 8 it can be seen that in most situations the measured coordinates were smooth and the velocities well matched with the future trajectory.

Near the rebounding point the deviation clearly increased. This is because the real trajectory suddenly changed in the rebounding point, leading to failure of (12). This defect can be avoided by abounding the coordinates near the table.

VI. Conclusion

In this paper, a high-speed vision system for table tennis robot has been built. To enhance its performance, a new ball detection algorithm is proposed, which improves the image-processing speed up to $200FPS$. Meanwhile, a new algorithm for ball trajectory reconstruction is proposed, by which we can obtain the spatial trajectory of the flying ball without synchronization of the two cameras. Experiments show that the ball detection algorithm is efficient and robust, and the trajectory reconstruction algorithm achieves satisfied accuracy.

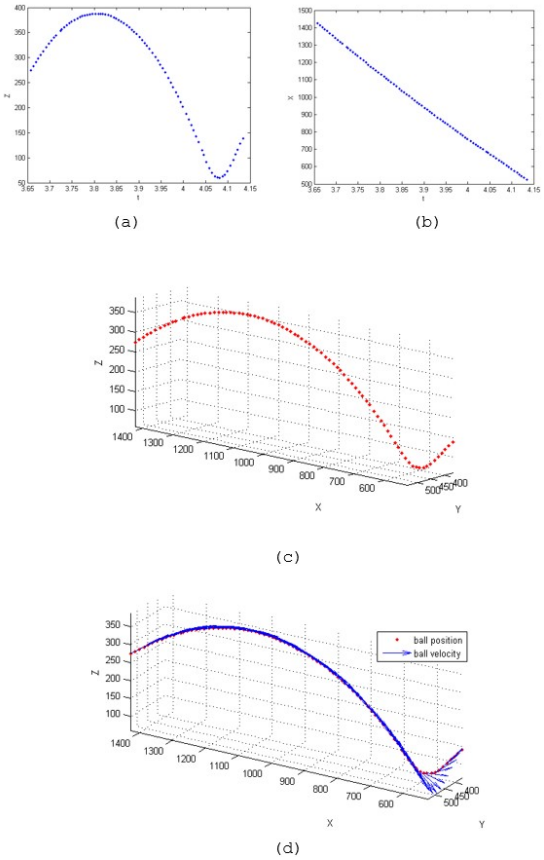

Fig. 8. Trajectory reconstruction results. (a) t-X curve, (b) t-Z curve, (c) spatial coordinates of the ball trajectory, (d) spatial velocities of the ball trajectory

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China under Grant 61305024, Grant 6137337, Grant 61227804 and Grant 61333016. It was also supported by the Beijing Natural Science Foundation under Grant 3141002.

REFERENCES

[1] Z. Zhang, D. Xu, and M. Tan, "Visual measurement and prediction of ball trajectory for table tennis robot," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 12, pp. 3195–3205, 2010.

[2] K. Mülling, J. Kober, and J. Peters, "A biomimetic approach to robot table tennis," *Adaptive Behavior*, vol. 19, no. 5, pp. 359–376, 2011.

[3] M. Matsushima, T. Hashimoto, M. Takeuchi, and F. Miyazaki, "A learning approach to robotic table tennis," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 767–771, 2005.

[4] C. Lai and T. J. Tsay, "Self-learning for a humanoid robotic ping-pong player," *Advanced Robotics*, vol. 25, no. 9-10, pp. 1183–1208, 2011.

[5] R. L. Anderson, *A robot ping-pong player: experiment in real-time intelligent control*. MIT press, 1988.

[6] Y. Huang, D. Xu, M. Tan, and H. Su, "Trajectory prediction of spinning ball for ping-pong player robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2011, pp. 3434–3439.

[7] L. Acosta, J. Rodrigo, J. A. Mendez, G. Marichal, and M. Sigut, "Ping-pong player prototype," *IEEE Robotics & Automation Magazine*, vol. 10, no. 4, pp. 44–52, 2003.

[8] A. Mittal and N. Paragios, "Motion-based background subtraction using adaptive kernel density estimation," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004*, vol. 2. IEEE, 2004, pp. II–302.

[9] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780–785, 1997.

[10] D. A. Forsyth and J. Ponce, *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.