Proceedings of the 34th Chinese Control Conference
July 28-30, 2015, Hangzhou, China

# An Adaptive Way to Detect the Racket of the Table Tennis Robot Based on HSV and RGB

ZHANG Kun, FANG ZaoJun, LIU JianRan, and TAN Min

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

**Abstract:** To detect the racket pose faster and easier, a new method combining the HSV and RGB color spaces are presented. The purpose of using the HSV color space is to get enough information in the RGB color space. By detecting the mark on the racket in real-time, the racket pose could be obtained. The experimental results show that the proposed method has a great advantage on the robustness of corner extraction of the rectangle on the racket.

**Key Words:** racket pose, HSV, RGB, self-adaptive method

## 1 Introduction

Recently, many researchers are interested in the table tennis robot, because it is a great model containing many important technologies, especially the machine vision and automatic control. The table tennis robot was originated in 1983. In the early period, researchers constructed some simple mechanisms to achieve the goal of hitting the balls and striking with human [1-4]. With the development of technology, the table tennis robots are more complex, and they can complete more difficult goals, such as imitation form human motions, learning how to play by themselves and so on. However, there are still many tasks to be solved. For example, the balls need to be detected from some complex backgrounds in a shorter time, since all the flying balls just leave little time for the robot [5]. Many researchers increase the acquisition frame rate of the vision system or develop new algorithms to solve this problem.

In order to make table tennis robots perform better, the racket draws attentions of the researchers. Controlling the racket well means that the ball is returned better, Yang [6] proposed a good method on the basis of the physical model. Even in the spinning ball striking, the trajectory of the racket is very helpful [7].

Unfortunately, during the moment hitting the process of the balls, the racket pose is always changing, which results in that the irradiating light on the racket is not stable. Additionally, the racket has a certain degree of reflection. Due to the fast movement of the racket, the images need to be processed in a short time. In this respect, Wang extracted the racket according to the HSV components [8], and Chen used a wide-range and multi-condition threshold method to segment the racket [9].

As the racket region extracted, the racket pose could be estimated, Ren adopted the P4P (Perspective 4 point) algorithm to calculate the racket pose [10]. Hiroshi used multiple cameras directly to acquire the racket pose [11]. There are still many problems that need to be solved. For example, the grey value thresholds on the image segmentation should be designed more robust and the racket pose needs to be extracted more stable.

To track the racket trajectory easier, this paper mainly focuses on the racket pose. A more adaptive method is proposed. In section II, we introduce the table tennis system. The new method combining the HSV and the RGB color spaces is presented in the section III. In the section IV, the racket pose calculated depends on the P4P and orthogonal iteration (OI) algorithms. The experiments and the results are given in section V. Finally, a brief conclusion is given in section VI.

## 2 The System Hardware

The table tennis robot mainly contains three parts: 5DOF motion mechanism hitting the balls, a binocular vision system tracking the balls with high speed and a monocular vision system computing the racket trajectory, as shown in Fig. 1.

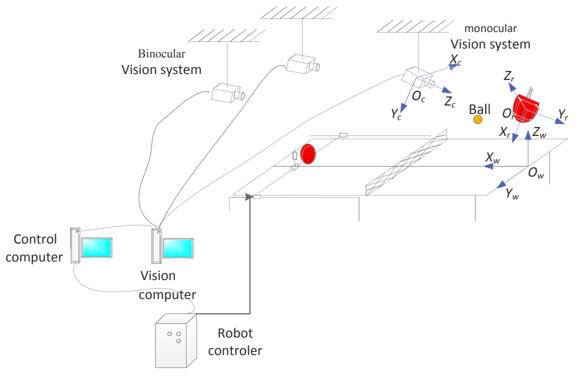

Fig.1 The table tennis robot system

Three coordinate frames are established, i.e., the world frame $O_w$ the camera frame $O_c$ and the racket frame $O_r$. We choose the table plane as the $X_wY_w$ plane of the frame $O_w$, and the positive direction of the $X_w$-axis is parallel to the long edge of the table, pointing away from the player. The direction perpendicular to the desktop and upward is the positive direction of $Z_w$-axis. The frame $O_c$ is established on the optical center of the camera, the direction of the optical axis is selected as the $Z_c$-axis, and the $X_c$-axis is parallel to the horizontal direction. In the frame $O_r$, the original point is the center of the racket, the plane of racket is the $X_rY_r$ plane, the

This work was supported by the National Natural Science Foundation of China under Grant 61305024, Grant 61273337, Grant 61227804, Grant 61271432 and Grant 61333016. It was also supported by the Beijing Natural Science Foundation under Grant 3141002.

$X_r$-axis is from the center of racket to the hand shank, the normal direction of the racket is the $Z_r$-axis.

To acquire the racket pose in frame $O_w$, the racket pose relative to the frame $O_c$ must be known as follows

$$ {}^{W}T_{r} = {}^{c}T_{r} * {}^{c}T_{W}^{-1} \tag{1} $$

where ${}^{w}T_r$ is the racket pose relative to the frame $O_w$, and ${}^{c}T_r$ is the racket pose relative to the frame $O_c$, ${}^{c}T_{w}^{-1}$ is the inversion of the transformation matrix from the frame $O_w$ to the frame $O_c$, which can be calculated after the camera calibration.

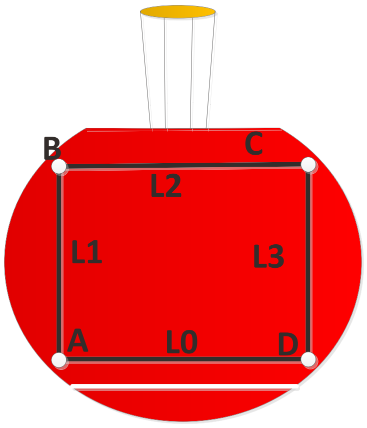

Fig.2 The mark on the racket

To obtain ${}^{w}T_r$, we draw a mark on the racket in advance, as shown in Fig.2. In the red region of the racket, there are a black rectangle and a white line. The white line is parallel to the long side of the rectangle. The corners A,B,C,D are painted as white. The main task of this paper is to extract the corners on the racket and calculate the racket pose based on the corners' coordinates in the frame $O_w$.

## 3 Extraction of the Corners

### 3.1 Extraction of the Red Region and the White Line

In the circumstances of a clean background, the red region of the racket is a special characteristic. However, the racket pose always changes and the light conditions vary widely. Moreover, the plastic surface would reflect the light. It's hard to extract the region just from the RGB color space. Considering the HSV color space is more sensitive to the color than the RGB color space, the red region and the white line would be extracted combining the two color spaces.

In HSV color space as shown in Fig.3, the value of the H component in red area is near 0. It is easy to choose a suitable threshold in the H component, For the S component, it is similar. The rough red region could be detetcted as defined in (2) to (4)

$$ P_1 = \{(i,j)/H(i,j) < T_H\} \tag{2} $$

$$ P_2 = \{(i,j) \mid S(i,j) > T_S\} \tag{3} $$

$$ P_R = P_1 \cap P_2 \tag{4} $$

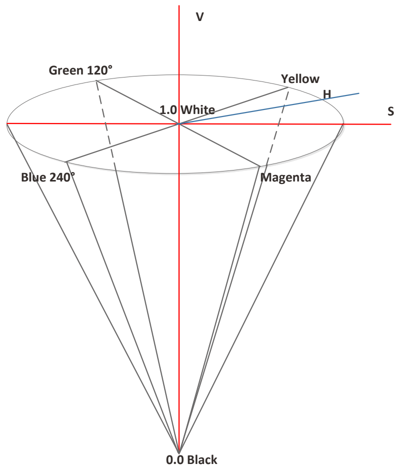

Fig.3 The HSV color space

where $P_R$ is the set of the red points, $H(i,j)$ and $S(i,j)$ are the H,S component values of the pixel point $(i,j)$ respectively. $T_H$, $T_S$ are the threshold of the H and S components. The rough center of the racket $(X_r, Y_r)$ can be obtained via red points' coordinates.

With the red region computed, the light situation from red region's RGB value is determined. The thresholds of the R, G, B components to detect red points are the median of the R, G, B components in $P_R$ as follows

$$ T_R^r = \mathrm{median}\{R(i,j) \mid (i,j) \in P_R\} \tag{5} $$

$$ T_G^r = \mathrm{median}\{G(i,j) \mid (i,j) \in P_R\} \tag{6} $$

$$ T_B^r = \mathrm{median}\{B(i,j) \mid (i,j) \in P_R\} \tag{7} $$

where $T_R^r$, $T_G^r$, $T_R^r$ are the thresholds of the R,G,B components relative to the red points.

Using the similar way to extract the white line, in the HSV color space, the S value of the white approximates 0 and the V value is 1, it's nothing to do with the H value. It is also easy to choose good thresholds for the V component and the S component, as given in (8) to (10)

$$ P_1' = \{(i,j) \mid S(i,j) < T_S'\} \tag{8} $$

$$ P_2' = \{(i,j) \mid V(i,j) > T_V'\} \tag{9} $$

$$ P_{WA} = P_1' \cap P_2' \tag{10} $$

where $P_{WA}$ is the rough set of the white line.

Based on that the white line is in the red region and the variation values between the G and B components of the white points are the maximums in a single direction, the valid white points could be determined from (11) to (16).

$$ P_{WP}^{H\lambda} = \{(i,j) \mid (i-\delta_\lambda, j) \in P_R, (i+\delta_\lambda, j) \in P_R, (i,j) \in P_{WA}\} $$
$$ P_{WP}^{V\lambda} = \{(i,j) \mid (i, j-\delta_\lambda) \in P_R, (i, j+\delta_\lambda) \in P_R, (i,j) \in P_{WA}\} \tag{11} $$

$$ WP_{WG}^{V} = \{(i,j) \mid \arg\max(\sum_{n=0}^{\alpha} G(i, j+n-\alpha) C^T(n)) \\ ,(i,j) \in (P_{WP}^{H\lambda} \cup P_{WP}^{V\lambda})\} \tag{12} $$

$$ WP_{WB}^{V} = \{(i,j) / \arg\max(\sum_{n=0}^{\alpha} B(i, j+n-\alpha) C^T(n))\} \\ , (i,j) \in P_{WP}^{H\lambda} \cup P_{WP}^{V\lambda}\} \tag{13} $$

$$WP_{WG}^{H} = \{(i,j) \mid \arg\max(\sum_{n=0}^{\alpha} G(i+n-\alpha, j)C^{T}(n)) \\ ,(i,j) \in P_{WP}^{H\lambda} \cup P_{WP}^{V\lambda}\} \quad (14)$$

$$WP_{WB}^{H} = \{(i,j) / \arg\max(\sum_{n=0}^{\alpha} B(i+n-\alpha, j)C^{T}(n)) \\ ,(i,j) \in P_{WP}^{H\lambda} \cup P_{WP}^{V\lambda}\} \quad (15)$$

$$\begin{cases} \delta_{\lambda} = V_t, if \lambda = 1 \\ \delta_{\lambda} = V_s, others \end{cases} \quad (16)$$

where the values of $V_t$, $V_s$ ($V_t > V_s$ ) depend on the real condition, $P_{WP}^{H\lambda}$, $P_{WP}^{V\lambda}$ are the set whose points have red points around in the horizontal direction and vertical direction. $C^T(n)$ is a 1-D line detection operator [9], $WP_{WG}^{V}$, $WP_{WB}^{V}$, $WP_{WG}^{H}$, $WP_{WB}^{H}$ are the sets whose points have a max range between the G and B components in the horizontal and vertical direction.

The points in the white line satisfy the principle defined as (17)

$$P_W = WP_{WG}^{V} \cap WP_{WB}^{V} \cap WP_{WG}^{H} \cap WP_{WB}^{V} \quad (17)$$

As the set of the white points obtained, firstly the rough white line is extracted with the Optimized Hough Transform algorithm. Secondly, the RANSAC algorithm is used to reduce the noise. In the end, the white line function can be calculated by the least square line fitting algorithm.

$$y = K_W x + L_W \quad (18)$$

The distance from the center point ($X_r$, $Y_r$ ) to the white line is $D_w$. We plot a square, whose center is the center point ($X_r$, $Y_r$ ) with sides of $D_w$, as the interested area $P_{ROI}$. The thresholds of the R, G, B components relative to white points form the white line are determined as follows.

$$T_R^{w} = \mathrm{median}\{R(i,j) \mid (i,j) \in P_W\} \quad (19)$$

$$T_G^{w} = \mathrm{median}\{G(i,j) \mid (i,j) \in P_W\} \quad (20)$$

$$T_B^{w} = \mathrm{median}\{B(i,j) \mid (i,j) \in P_W\} \quad (21)$$

### 3.2 Detection of the Four Black Line

The four black lines would be represented as shown in (22) and (23)

$$L_i\!: y = K_W x + L_i,\ i = 0,2 \quad (22)$$

$$L_i\!: -K_W y = x + L_i,\ i = 1,3 \quad (23)$$

In the same region and light, the values of the R, G, B components relative to black points are less than the red points. We could make use of $T_R^r$, $T_G^r$, $T_B^r$ to detect the rough black lines.

$$P_G^B = \{(i,j) \mid G(i,j) < offset_G + T_G^r, (i,j) \in P_{ROI}\} \quad (24)$$

$$P_B^B = \{(i,j) \mid B(i,j) < offset_B + T_B^r, (i,j) \in P_{ROI}\} \quad (25)$$

$$P_R^B = \{(i,j) \mid R(i,j) < offset_R + T_R^r, (i,j) \in P_{ROI}\} \quad (26)$$

$$P_B = P_R^B \cap P_G^B \cap P_B^B \quad (27)$$

where offset is the allowance to the $T_R^r$, $T_G^r$, $T_B^r$.

The black lines are detected based on the same principles and algorithms as the white line. The main steps include finding the rough black points in the ROI and the verification of the principle that the two sides of the black points in the ROI are red points and clearing up the set of black points.

In the ROI, the R, G, B components of the black points are less than the R, G, B components of the red points around. Thus, the rough black points could be found out with the principle that the two sides of the rough black points must be red points. Because the black lines are about 2mm wide, when the camera is far away from the racket, the black lines in the images are fuzzy. In order to find out the black lines precisely, all the points in the black lines are selected from the principle that the sum value of the RGB component in one-dimensional are minimum.

The four black lines are distinguished by the relative positions of the center point and the white line. The line $L_0$ and the line $L_2$ are parallel to the white line, and $L_0$ is in the middle of the center of the racket and the white line, the line $L_2$ is not. According to the slop of the white line and the center point of the racket, the line $L_1$ and $L_3$ could be detected. When the absolute value of $K_w$ is less than 1and the white line is under the center point of the racket, the location of the line $L_3$ is at the center point's right, and the left of the center point is the line $L_1$. Having the functions of the four black lines, the coordinates of the corners A, B, C, D relative to the camera frame could be calculated.

### 3.3 Optimization of the Corners

To ensure the accuracy of the corners' coordinates, it is verified by three steps: smooth the points around the corners, pick out the points with the max sum value of all the R, G, B components and verify that the picked points is satisfied the thresholds $T_R^W$, $T_R^W$, $T_B^W$. The smoothing operator is as shown in (28)

$$\omega = \frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad (28)$$

## 4 Estimation of the Racket Pose

Based on the four coplanar and non-collinear points' coordinates, the racket pose could be calculated via the P4P algorithm. In order to guarantee the orthogonality of the orientation matrix of the racket pose, OI algorithm is used to optimize the matrix. The initial value of the OI is from the result calculated by the P4P algorithm.

The information about the camera could be described as

$$T_{in} = \begin{bmatrix} k_x & 0 & u_0 \\ 0 & k_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (29)$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = T_{in} \begin{bmatrix} x_c / z_c \\ y_c / z_c \\ 1 \end{bmatrix} \quad (30)$$

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = {}^{c}T_r \begin{bmatrix} x_r \\ y_r \\ z_r \\ 1 \end{bmatrix} \quad (31)$$

Where $T_{in}$ is the intrinsic model of the camera, and $(u,v)$ are the coordinates in the images, $(x_c, y_c, z_c)$ are the corners coordinates in frame $O_c$, $(x_r, y_r, z_r)$ are the corners coordinates in frame $O_r$, the intrinsic model of the camera can be obtained by camera calibration, and the matrix ${}^cT_r$ is the ultimate goal to achieve.

After the racket pose relative to the frame $O_c$ is calculated, the racket pose relative to the frame $O_w$ could be obtained as the formula (1). When getting a pose of the racket from an image recorded by the camera, we can track all the trajectory of the racket from a video.

## 5 Experiments and Results

To verify the method of this paper reliable, experiments were well conducted. The monocular vision system was made up of a Lenovo computer and a camera GC660C, which could grab images in a speed of 119 fps.

Firstly, we captured some images to detect the corners, where the poses of the racket were different. As shown in Fig.4, some extreme racket poses could be extracted via the method of this paper. From the results, it could be seen that if the thresholds were not computed well using the method in this paper, the method in [9] doesn't work well.

Secondly, randomly selecting multiple images from a video, we attempted to extract the corners from those images via the two methods. The results are shown in Table I, we could see that there were some errors in the two methods, and some errors were relatively large, a special reason causing the result is that some racket poses are so extreme that the results aren't ideal. Other reasons include that via the method in [9] we didn't change the thresholds and the centers of the corners sometimes were in the middle of two pixels.

Finally, we rotated the racket 180° about the $Y_w$-axis, the beginning position was that the racket was parallel to the $X_wY_w$ plane, only the pitch angle changed from 0° to 180°, The result which showed that we got the corners coordinates successful is shown in the Fig.5.

Without changing any thresholds, the angle range was from 64.5° to 168.8°, but without changing any thresholds, the result with the method in [9] was only 60°, it could be seen from the result that the proposed method had great adaptive.

### 5.1 Conclusion

A method combining the HSV and RGB color spaces is presented to extract the racket pose. The experimental results indicate that the method proposed is more adaptive, it adapts to some extreme conditions, and it does not need to change the thresholds for satisfying the light condition. With this method, it is easier to calculate the racket pose in real-time. However, the experiments show that the method needs to be improved so that we could obtain all the racket poses. After all, in some extreme situations, the red region of the racket may disappear in the image. Moreover, the result depends on the color of the target. Therefore, there are many problems that need to be solved, and another goal to achieve is to track the complete trajectory of the racket.

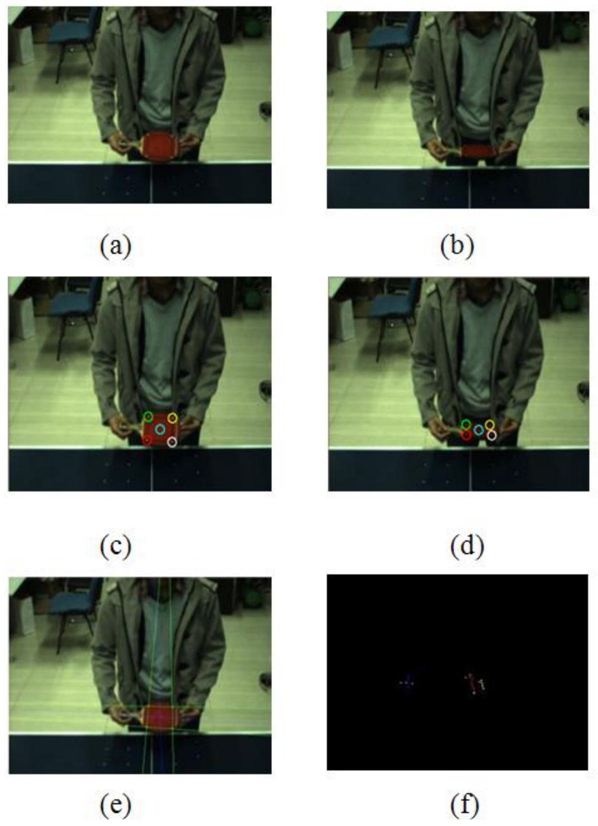

Fig.4. The corner detection results in the different racket poses. (a),(b) the origin image, (c),(d) Results with the proposed method. (e), (f) Results with the method in [9].

Table 1: The real data and the extracted coordinates in two method

| Images | Feature points | Real data | The proposed method | The method in [9] |
|---|---|---|---|---|
| 1 | A | (399,373) | (401,371) | (395,379) |
| | B | (336,372) | (336,372) | (339,370) |
| | C | (341,326) | (341,327) | (338,339) |
| | D | (402,330) | (402,330) | (400,345) |
| 2 | A | (397,379) | (398,380) | (396,378) |
| | B | (335,377) | (334,378) | (335,377) |
| | C | (341,323) | (341,323) | (340,328) |
| | D | (401,326) | (401,327) | (401,328) |
| 3 | A | (395,381) | (395,381) | (396,380) |
| | B | (335,381) | (335,383) | (335,379) |
| | C | (340,321) | (340,321) | (340,322) |
| | D | (401,323) | (401,323) | (401,322) |
| 4 | A | (396,378) | (396,378) | (397,376) |
| | B | (336,375) | (336,375) | (337,372) |
| | C | (340,311) | (341,311) | (340,309) |
| | D | (399,312) | (400,312) | (400,314) |
| 5 | A | (399,365) | (399,365) | (399,365) |
| | B | (338,363) | (338,363) | (339,364) |
| | C | (340,318) | (340,318) | (339,319) |
| | D | (398,320) | (398,320) | (399,321) |

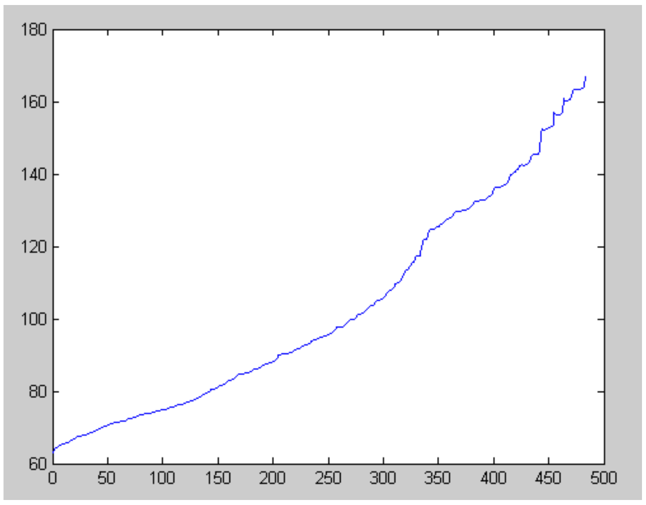

Fig.5 The angle which can detect the corners

## References

[1] R. Anderson, “A robot ping-pong player: experiment in real time intelligent control,” Cambridge, MA: MIT Press, 1988.

[2] R. Anderson, “Dynamic sensing in a ping-pong playing robot,” IEEE Transactions on Robotics and Automation, vol. 5, no. 6, pp. 728-739, Dec. 1989.

[3] L. Acosta, J. J. Rodrigo, J. A. Méndez, G. N. Marichal, and M. Sigut, “Ping-pong player prototype,” IEEE Robotics & Automation Magazine, vol. 10, pp. 44-52, Dec. 2003.

[4] 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, Aug. 2005.

[5] J. R. Liu, Z. J. Fang, K. Zhang, “Improved high-speed vision system for table tennis robot,” in International Conference on Mechatronics and Automation(ICMA), 2014

[6] P. Yang, D. Xu, H. Wang, and J. Zhang, “Design and motion control of a ping pong robot,” Proc. Of the 8th World Congress on Intelligent Control and Automation, pp. 102-107, 2010.

[7] C. Liu, Hayakawa, Y., & Nakashima, A. (2012, October). Racket control for robot playing table tennis ball. In Control, Automation and Systems (ICCAS), 2012 12th International Conference on (pp. 1427-1432). IEEE.

[8] Q. Z. Wang, X. Q. Zhang, D. Xu, “Human behavior imitation for a robot to play table tennis,” in Control and Decision Conference(CCDC), 2012 24th Chinese. IEEE, pp. 1482-1487, 2012.

[9] G. D Chen, D. Xu, et al. "Visual Measurement of the Racket Trajectory in Spinning Ball Striking for Table Tennis Player." IEEE Transactions on Instrumentation and Measurement 2013

[10] Y.Q. Ren, D. Xu, M. Tan, “Pose estimation of racket pad for ping pong robot,” Proceedings of 2010 8th World Congress on Intelligent Control and Automation, IEEE, pp. 1017-1021, 2010.

[11] H. OHYA, H. SAITO, “Tracking racket face in tennis serve motion using high-speed multiple cameras,” In CiteSeer, unpublished