Abstract — We propose a method of controlling the paddle so as to return the ball to a desired point on the table with a specified flight duration. The proposed method consists of the following three input-output maps implemented by means of Locally Weighted Regression (LWR): (1) A map for predicting the impact time of the ball hit by the paddle and the ball position and velocity at that moment according to input vectors describing the state of the incoming ball, (2) A map representing a change in ball velocities before and after impact and (3) A map giving the relation between the ball velocity just after impact and the landing point and time of the returned ball. We also propose a novel control scheme based on iterative learning control to accurately achieve the stroke movement of the paddle as determined by using these maps.

Keywords: Dynamic manipulation, Table tennis robot, Input-output map, LWR, Iterative learning control.

1 Introduction

To perform tasks during intermittent interactions between robot and environment, the robot must be able to adjust the strength and timing of interactions during execution of the tasks. These are called hybrid (mixed continuous and discrete) control problems by Burridge et al, who exemplified them as hopping, catching, hitting, and juggling as typical examples [1]. These tasks have attracted the attention of researchers also in the area of experimental psychology in the sense that they require both receptor anticipation of the environmental situation and effector anticipation of internal movement processes[2]. However, it is difficult to find general approaches that are sufficiently tractable for the robot to perform these tasks. In this paper, we focus on the table tennis task that involves the intermittent nature of the robot-ball interaction in order to explore the hybrid control problems.

Andersson constructed a sophisticated robot system that could play table tennis against humans[3]. An expert controller he developed performs task planning, updates the plan, and improves its quality as the sensor data change, based on physical models and the exception handling mechanism. Andersson’s approach makes full use of the human knowledge as explicit models of the task and the environment, and the task performance depends on the system creator’s knowledge. In other words, the robot system could not improve its skills through practice or experience.

Researchers in the area of sports science have proposed hypotheses on the nature of the human internal processes to execute complex tasks such as the table tennis stroke. For example, Ramanantsoa proposed simplifying procedures of the table tennis stroke based on Bernstein’s hypothesis that expert players limit the degrees of freedom in their movements when planning and performing a shot[4]. The core idea of the procedures he proposed is to identify and reach virtual targets, the point at which the ball should be struck and the paddle velocity just before hitting the ball.

Motivated by Ramanantsoa’s idea, we constructed a robot system that performs the table tennis task, in which virtual targets are predicted using input-output maps implemented efficiently by means of a k-d tree (short for k-dimensional tree)[5]. The paddle approaches these targets by using a visual feedback control scheme similar to the mirror law proposed by Koditschek[6]. However, the trajectory of the ball hit by the paddle is uniquely determined depending on the traveling distance of the ball specified beforehand. In other words, this method is not capable of controlling the speed of the returned ball.

In this paper, we propose a method of controlling the paddle so as to return the ball to a desired point on the table with a specified duration of flight. The proposed method consists of the following three input-output maps implemented by means of Locally Weighted Regression (LWR)[7]:
(1) A map for predicting the impact time of the ball hit by the paddle and the ball position and velocity at that moment according to input vectors describing the state of the incoming ball
(2) A map representing a change in ball velocities before and after the impact
(3) A map giving the relation between the ball velocity just after the impact and the landing point and time of the returned ball

We also propose a novel control scheme based on the iterative learning control[8] to accurately achieve the stroke movement of the paddle determined by using these maps.

An overview of the table tennis robot system is presented in the next section. The third section proposes a novel control scheme for precisely tracking the trajectory of the paddle arbitrarily given as a function of time. The above-mentioned input-output maps are explained in detail in the fourth section. After that, experimental results including rallies with a human opponent are reported to demonstrate the effectiveness of our approach.

2 Table tennis system

Although Andersson stated that five degrees of freedom are required for the paddle attached to the robot to execute the table tennis task, the minimum number of the degrees of freedom is four, two for its position in a horizontal plane and two for its attitude because every incoming ball rebounds on a table before being hit during the game. In this paper, we adopt a robot system with the minimum number of degrees of freedom (four) to demonstrate only the effectiveness of our approach to the robot table tennis task, without considering the use of redundant degrees of freedom.

Figure 1 illustrates the table tennis robot system we developed. The robot is driven by four electric motors, motors 1 and 2 for the motion in a horizontal plane and motors 3 and 4 for the paddle attitude. The 155[mm] square paddle moves in parallel with the table at a height of 195[mm]. A stereo vision system (Quick MAG System 3 : OKK inc.) extracts the ball’s location from the image every 1/60[sec].

3 Ball events in one stroke

To make the following explanation clear, we define “ball events” by stating conditions of a ball, which are given below and in Figure 2.

3.1 Definition of ball events

Event-(s) Hit by the opponent
Event-(m) Passing through a virtual plane for the measurement
Event-(l) Bouncing on the robot’s court
Event-(h) Hit by the robot
Event-(r) Bouncing on the opponent’s court

Numbers 1 and 2 in Figure 2 mean “before” and “after” the bounce.

The task starts when an opponent player or a pitching machine hits the ball (s) and the ball passes through a virtual plane (m) that is set to measure the motion of an incoming ball. After the ball flies over the net, it bounces in the robot’s court (l1, l2) and is hit by the robot (h1, h2). The ball is then returned and bounces in the opponent’s court again (r1, r2).

3.2 Stroke movement

The table tennis task can be divided into three sub-tasks as shown in Figure 3. The hitting task (A) is to return the incoming ball and the returning task (B) is to return the paddle to the initial position to prepare for the next hitting. In the waiting task (C), which has no paddle movement, the robot prepares for the next hitting task.

We divide the waiting task into three parts. In C0, the system updates input-output maps (explained in section 5) and continues monitoring the ball to find that the event m occurs. In C1, the system predicts everything required to hit the ball and determines the hitting time and motion. In C2, the system generates the hitting trajectory and motion commands based on the prediction made in C1.
4.1 Motion trajectory

Assuming the desired motion (position) trajectory of the paddle in each axis is given as a fifth polynomial of time, the velocity is given as

\[ y_d(t) = c_1 t^4 + c_2 t^3 + c_3 t^2 \]  \hspace{1cm} (1)

where the coefficients \(c_1, c_2\) and \(c_3\) are determined by the following equation;

\[
\begin{bmatrix}
\frac{1}{4} T_L^5 & \frac{1}{4} T_L^4 & \frac{1}{4} T_L^3 \\
T_L^4 & T_L^3 & T_L^2 \\
4T_L^3 & 3T_L^2 & 2T_L
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
c_3
\end{bmatrix}
= \begin{bmatrix}
X_L \\
V_L \\
0
\end{bmatrix} \hspace{1cm} (2)
\]

where we assume that the initial states (position, velocity and acceleration) are all 0 and the final states (at \(t = T_L\)) are \((X_L, V_L, 0)\).

4.2 Generating input commands

In the following, we assume that the controlled object is represented as a linear system with an input \(u\) and an output \(y\) (it is possible to extend this to multi-variable systems).

Let \(u_a\) be a learned input for a given trajectory with an output \(y_a\) expressed by coefficients \(c_a = [c_{a1}, c_{a2}, c_{a3}]^T\) which is experimentally obtained by the iterative learning control. Similarly, let \(u_b\) and \(u_c\) be learned inputs for different trajectories with outputs \(y_b\) and \(y_c\) expressed by coefficients \(c_b = [c_{b1}, c_{b2}, c_{b3}]^T\) and \(c_c = [c_{c1}, c_{c2}, c_{c3}]^T\). If the output \(y_d\) of a new trajectory is given by

\[ y_d(t) = k_au_a(t) + k_by_b(t) + k_cy_c(t) \]  \hspace{1cm} (3)

as a linear combination of given outputs, we can obtain the corresponding input \(u\) as a linear combination of the form

\[ u = k_au_a + k_by_b + k_cu_c \]  \hspace{1cm} (4)

where the coefficients \([k_a, k_b, k_c]^T\) are determined by solving

\[
\begin{bmatrix}
c_{a1} & c_{b1} & c_{c1} \\
c_{a2} & c_{b2} & c_{c2} \\
c_{a3} & c_{b3} & c_{c3}
\end{bmatrix}
\begin{bmatrix}
k_a \\
k_b \\
k_c
\end{bmatrix}
= \begin{bmatrix}
c_{d1} \\
c_{d2} \\
c_{d3}
\end{bmatrix} \hspace{1cm} (5)
\]

where \(c_d = [c_{d1}, c_{d2}, c_{d3}]^T\) are coefficients defining the output \(y_d\) and the \(3 \times 3\) matrix on the left side of Eq.(4) has to be nonsingular. It should be noted that this method allows different durations of the motion trajectories (different \(T_L\)s are acceptable).

4.3 Experimental verification

Figure 5 shows an experimental result obtained by applying the proposed method to the actuator on the x axis. A new trajectory is accurately achieved by the input calculated by the proposed method using the inputs and outputs for the three trajectories learned beforehand (see Figure 4).
Next we explain how to achieve an accurate stroke movement and demonstrate an experimental result. This is an application of the proposed method to the table tennis task. As mentioned in Section 3, one stroke motion of the paddle consists of two parts. One is “hitting motion” that starts from a home position (at $t = 0$) and ends at a predicted hit point $X_h$ with a desired velocity $V_h$ and attitude of the paddle (at $t = t_h$). Another is “returning motion” that starts from the final state of “hitting motion” (at $t = t_h$) and ends at the home position (at $t = t_p$). Figure 6 illustrates an experimental result of the stroke motion achieved by using the proposed method. In this stroke motion, the desired velocity at the hit point $X_h = 500[\text{mm}]$ is set $V_h = 200[\text{mm/sec}]$. Hitting motion ends at $t_h = 0.4[\text{sec}]$ and returning motion ends at $t_p = 0.8[\text{sec}]$. From this figure, we can see that a given stroke movement of the paddle is accurately achieved by the input determined using the proposed method.

### Paddle motion decision

Determining the impact time of the ball hit by the paddle and the paddle position and velocity at that moment is most important for performing the table tennis task. This decision has to be made before the impact occurs. Our approach to making the decision is to use empirically acquired input-output maps as opposed to Andersson’s approach based on the human knowledge as explicit models of the task and the environment. These input-output maps are defined below.

[Map 1] – A map for predicting the impact time of the ball hit by the paddle and the ball position and velocity at that moment according to input vectors describing the state of the incoming ball.

[Map 2] – A map representing a change in ball velocities before and after the impact.

[Map 3] – A map giving the relation between the ball velocity just after the impact and the landing point and time of the returned ball.

These maps are implemented by means of LWR.

Learning [map 1] is implemented by storing the measured data of incoming balls hit by a human opponent or a pitching machine. Learning [map 2] and [map 3] are implemented by storing the measured data of balls returned by the paddle using [map 1].

#### 5.1 Learning [map 1]

We consider that the transition of the ball’s state from Event $(m)$ to Event $(h1)$ is expressed as nonlinear functions of the form

$$
dt = f_1(p_{mz}, v_{bmx}, v_{bmy}, v_{bmz}, a_{bmz}) \quad (6)
$$

$$
dx = f_2(p_{mz}, v_{bmx}, v_{bmy}, v_{bmz}, a_{bmz}) \quad (7)
$$

$$
dy = f_3(p_{mz}, v_{bmx}, v_{bmy}, v_{bmz}, a_{bmz}) \quad (8)
$$
\[ v_{bh1x} = f_4(p_{bmz}, v_{bmz}, v_{bmy}, v_{bmz}, a_{bmz}) \] (9)
\[ v_{bh1y} = f_5(p_{bmz}, v_{bmz}, v_{bmy}, v_{bmz}, a_{bmz}) \] (10)
\[ v_{bh1z} = f_6(p_{bmz}, v_{bmz}, v_{bmy}, v_{bmz}, a_{bmz}) \] (11)

where \( dt = t_h - t_m \), \( dx = p_{bhx} - p_{bmz} \), \( dy = p_{bhy} - p_{bmy} \). In these equations, \((p_{bmz}, v_{bmz}, v_{bmy}, v_{bmz})\) and \((v_{bmz}, v_{bmy}, v_{bmz}, a_{bmz})\) are the ball’s position and velocity at Event (m) \((t = t_m)\), \(a_{bmz}\) is the acceleration in direction at Event (m), \((v_{bh1x}, v_{bh1y}, v_{bh1z})\) are the ball’s velocity at Event \((h_1)(t = t_h)\) (See Figure 7). Since it is difficult to represent these functions explicitly, we regard them as input-output maps consisting of empirically acquired data. We call these maps \([map \ 1]\). In the learning phase of \([map \ 1]\), we use the cross-validation error check [7] to store only reliable measured data. In the lookup phase, the impact time *t_h* and the ball position (*p_{bhx}, *p_{bhy}) and velocity (*v_{bh1x}, *v_{bh1y}, *v_{bh1z}) at that moment are determined by \([map \ 1]\) with the aid of LWR and the following relations.

\[ *t_h = t_m + *dt \]
\[ = t_m + f_1(p_{bmz}, v_{bmz}, v_{bmy}, v_{bmz}, a_{bmz}) \] (12)
\[ *p_{bhx} = p_{bmz} + *dx \]
\[ = p_{bmz} + f_2(p_{bmz}, v_{bmz}, v_{bmy}, v_{bmz}, a_{bmz}) \] (13)
\[ *p_{bhy} = p_{bmy} + *dy \]
\[ = p_{bmy} + f_3(p_{bmz}, v_{bmz}, v_{bmy}, v_{bmz}, a_{bmz}) \] (14)

The ball velocities are also predicted similarly.

5.2 Decision of the hitting state of the paddle using \([map \ 2]\) and \([map \ 3]\)

We decide the hitting condition of the paddle from the stored data on the motion of the paddle and ball before and after hitting. These data constitute \([map \ 2]\) and \([map \ 3]\).

5.2.1 Learning \([map \ 2]\) and \([map \ 3]\)

The transition of the ball’s state from Event \((h1)\) to Event \((h2)\) is governed by the collision dynamics between the ball and the paddle. The transition of the ball’s state from Event \((h2)\) to Event \((r)\) is governed by the flight dynamics after hitting. We consider that these transitions are expressed as input-output maps of the form

\[ [V_h, \theta_3, \theta_4] \rightarrow V_{bh12}(= [v_{bh12x}, v_{bh12y}, v_{bh12z}]) \] (15)
\[ V_{bh2} \rightarrow [dt_{hr}, dp_{bhr}, (= [dp_{bhrx}, dp_{bhrx}]) \] (16)

where the \( V_{bh12} \) means a change in velocities just before hitting \( V_{bh1} \) and just after hitting \( V_{bh2} \), that is, \( V_{bh12} = V_{bh2} - V_{bh1} \). \( dt_{hr} \) is the duration of the flight of the returned ball. \( dp_{bhr} \) is the flight distance, that is, \( dp_{bhr} = p_{br} - p_{bh} \). Since the acquired data in the learning phase should be distributed uniformly in the input space, we randomly choose input variables \( V_h \) and
6 Experimental results

In this section, we demonstrate the effectiveness of our approach by presenting experimental results achieved by the table tennis robot.

6.1 Experimental results of the “ball controlling task”

6.1.1 Ball prediction with the [map 1]

We evaluated the prediction accuracy of the ball hit by a human. The pitching machine set behind the robot’s court pitched the ball to the human. The human returned the ball to the robot’s court and the ball trajectory was measured. In this experiment we requested the human hit the ball normally.

The ball states were extracted offline from 300 trajectories data measured in advance. Two hundred ball states were used to construct the database and others were used to evaluate the prediction accuracy.

We calculated the statistical values of the data in a database beforehand and normalized all the data with the standard deviations. The weight matrix of the distance function is a unit matrix $I$, and the band width $h$ is fixed to 0.8 in the LWR [7].

Figures 8 and 9 depict the actual hitting time and position with their respective predicted values. The error in the hitting time is a few milliseconds, and the position errors are within a few centimeters. This means that if the paddle reaches the predicted position at the predicted time, it can hit the ball around the center of the paddle.

The errors in the ball velocities are a few centimeters per second, which is enough to determine the paddle conditions at the impact point.

6.1.2 Map for the decision of the hitting motion

We experimentally acquired [map 2] and [map 3] with which the robot decides the hitting motion.

In the learning phase, the robot hit the balls pitched by a pitching machine with a constant angle, velocity and spin using a paddle driven by the commanded velocities and angles in the range of

$$800 \leq V_h \leq 2000 \text{[mm/sec]}$$

$$-50 \leq \theta_3 \leq 50 \text{[deg]}$$

$$-20 \leq \theta_4 \leq 20 \text{[deg]}$$

Figure 10 illustrates a map representing the relation between $V_{bh12}$ and $V_h$ acquired by learning 120 trajectories of the returned ball, as an example of the acquired maps. Three surfaces corresponds to the cases of $v_{bh12z} = 0, -500$ and $500$ [mm/sec]. Using this map, the robot can determine the paddle velocity that gives the required ball velocity.

Next we evaluated the capability of controlling the flight duration of the returned ball with the acquired maps. We fixed the desired landing point as $x = -1100$ [mm], $y = 300$ [mm]. We also set the desired duration of flight as 0.5 [sec] and 0.7 [sec] alternatively. In this phase, the robot continues to acquire the data of the ball and paddle movements for [map 2] and [map 3] to improve hitting motion as well as in learning phase.
6.2 Experimental results of the rally task with a human

6.2.1 Rally task experiment

We demonstrate the robot rally with a human as an application of our approach (Figure 16). The “rally task” means the table tennis rally that people generally play. We consider it as the repetition of “ball controlling task” described in the previous section.

6.2.2 Experimental results

We demonstrate that the robot can perform the rally task with a human using the proposed method described previously. In the experiment, a human hit a ball toward the robot at random and the robot returned the ball with a fixed duration of flight \((dt_{hr}=0.55[sec])\) to a desired landing point \((p_{rx}=1550[mm], p_{ry}=0.3\times*p_{bhy})\) for the opponent’s easy hitting, where \(*p_{bhy}\) is a predicted impact point.

Figures 17 to 19 present some part of the data acquired in the rally. Figure 17 shows the time history of the ball motion in the \(x\) direction. Figures 18 and 19 show the ball and paddle trajectories on the \(x-y\) plane where the waiting position of the paddle is \(x=900[mm], y=0[mm]\).

We can see that the robot returns the ball to the point the opponent can hit easily by changing the impact point back and forward, right and left.

7 Conclusion

We have described an approach for a robot to perform the table tennis task based on two kinds of memory-based learning, one of which accurately achieves the stroke movement of the paddle and the other of which determines the paddle conditions at the impact point so as to return the ball to a desired landing point with a specified flight duration. Experimental results including
Figure 16: Experimental environment of the “rally task”

Figure 17: Ball motion in the “rally task” (in x direction)

Figure 18: Ball trajectory in the “rally task” (in x-y plane)

Figure 19: Paddle trajectory x in the “rally task”

rallies with a human opponent also have been reported.

References


