A Learning Approach to Robotic Table Tennis

Michiya Matsushima, Takaaki Hashimoto, Masahiro Takeuchi, and Fumio Miyazaki

***Abstract*—We propose a method of controlling a table tennis robot so as to return the incoming ball to a desired point on the table with specified flight duration. The proposed method consists of the following three input–output maps implemented by means of locally weighted regression: 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 feed-forward control scheme based on iterative learning control to accurately achieve the stroke movement of the paddle as determined by using these maps. Experimental results including rallies with a human opponent are also reported to demonstrate the effectiveness of our approach.**

***Index Terms*—Input–output map, iterative learning control, locally weighted regression (LWR), table tennis robot.**

I. 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 gave hopping, catching, hitting, and juggling as typical examples [1]. These tasks have attracted the attention of researchers also in the area of experimental psychology [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]. His approach makes full use of the human knowledge as explicit models of the task and the environment includes exception handling mechanism, 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 planning and performing stroke movements [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-dimensional (k-d) 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 inverse relation between the ball velocity just after the impact and the bouncing point and time of the returned ball.

These maps are implemented by means of LWR [7]. Once a trajectory of the paddle has been planned, it must be executed as accurately as possible. We propose a control scheme based on the iterative learning control (ILC) [8] to compensate for servo errors of the robot controller. This scheme makes it possible to synthesize new inputs without use of iterative operations in learning control given several other inputs that have been correctly learned already.

An overview of the table tennis robot system is presented in Section II. Section III describes "ball events" in one stroke in connection with tasks the robot has to execute. In Section IV, we explain the input–output maps to predict virtual targets and how to plan the trajectory of the paddle. Section V proposes a control scheme based on ILC for precisely tracking the trajectory of the paddle arbitrarily given as a function of time. After that, experimental results including rallies with a human opponent are reported to demonstrate the effectiveness of our approach.

II. Robot Table Tennis

A. Table Tennis System

Fig. 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 [millimeters] square paddle moves in parallel with the table at a height of 195 [millimeters]. A stereo vision system (Quick MAG System 3: OKK, Inc.) whose two cameras are set behind the paddle at $(x, y, z) = (3200, -1500, 2300)$ [millimeters] and (2700, 2000, 1800) [millimeters] described by the coordinate frame in Fig. 1 extracts location of the ball's center of gravity from the image every 1/60 [seconds] using the stereo calculation.

B. Table Tennis Task

Let us explain the task the table tennis robot must do. The robot plays table tennis according to the same rules for humans. The table's width is 1.5 m and the net is 0.16-m high, like human table tennis (Andersson's robot has played against humans according to special rules for robots, scaling down and restricting the area that must be covered by the robots). We consider that it is basically required for the table tennis robot to have the ability to perform the following task:

returning the ball to a desired point on the table with a specified flight duration.

Of course, these terminal conditions of the return trajectory must be chosen freely. In this paper, we focus on this basic task and describe

Manuscript received October 8, 2004. This work was supported by the JSPS. This paper was recommended by Associate Editor G. Oriolo and Editor H. Arai upon the evaluation of the reviewers' comments.

M. Matsushima, T. Hashimoto, and F. Miyazaki are with the Graduate School of Engineering Science, Osaka University, Osaka 560-8531, Japan (e-mail: roadya@robotics.me.es.osaka-u.ac.jp; miyazaki@me.es.osaka-u.ac.jp).

M. Takeuchi is with the Akashi National College of Technology, Hyogo 674-8501, Japan (e-mail: takeuchi@akashi.ac.jp).

Digital Object Identifier 10.1109/TRO.2005.844689

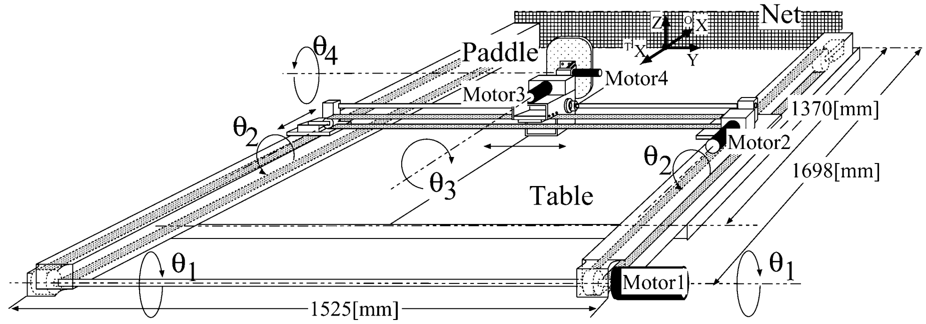

Fig. 1. Table tennis system.

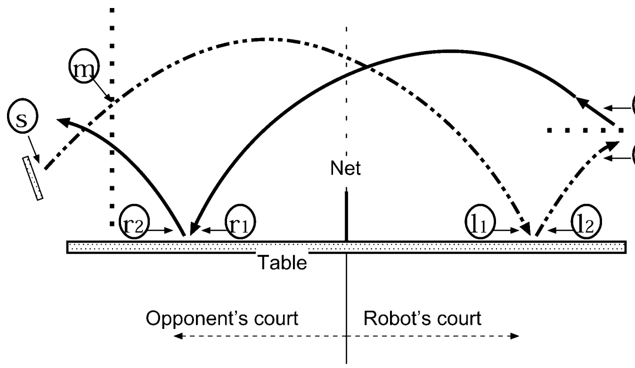

Fig. 2. Definitions of ball events.

how to execute it. The quantitative measure of the robot's performance is the resultant accuracy of the the aim point on the table and the flight duration.

III. 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 Fig. 2.

A. Definition of Ball Events

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

Numbers 1 and 2 in Fig. 2 mean "before" and "after" the bounce. The task starts when an opponent player or a pitching machine hits the ball and the ball passes through a virtual plane (m) that is set near far end of the opponent's court to measure the motion of an incoming ball. After that, it bounces in the robot's court and is hit by the robot $(h1, h2)$. The ball is then returned and bounces in the opponent's court again $(r1, r2)$.

B. Stroke Movement

The table tennis task can be divided into three subtasks as follows.

- A (Hitting Task): To return the incoming ball.
- B (Returning Task): To return the paddle to the waiting position to prepare for the next hitting.
- C (Waiting Task): The robot prepares for the next hitting task without paddle movement.

Moreover, we divide the waiting task into following three parts.

- C_0: The system updates input–output maps (explained in Section IV) and continues monitoring the ball to find that the event m occurs.
- C_1: The system predicts everything required to hit the ball and determines the hitting time and motion.

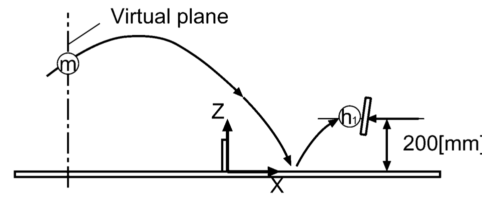

Fig. 3. Map 1.

- C_2: The system generates the hitting trajectory and motion commands based on the prediction in C_1.

C. Ball's State Estimation

In our approach, robot motions are planned based on the ball's state (position, velocity, acceleration) at the event m. The procedure for estimating the ball's state are given below:

1) to store a sequence of the ball's location acquired by the stereo vision system until the ball passes through the virtual plane;
2) to fit a trajectory described by a first-order polynomial of time in each direction of X and Y and a second-order polynomial of time in Z direction, using the least squares algorithm;
3) to calculate the ball's state at the event m using the trajectory obtained in 2.

The same procedure is employed to estimate the ball's state at other events.

IV. Paddle Motion Decision

A. Input–Output Maps

Determining the impact time of the ball hit by the paddle and the paddle position/attitude and velocity at that moment (we call these hitting conditions "virtual targets") 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 correspond to three physical phenomena shown in Figs. 3–5 and are defined below.

- Map 1—A map for predicting the impact time of the ball hit (t_h) by the paddle and the ball's position and velocity at that moment (at event h_1) according to input vectors describing the state of the incoming ball at event m.
- Map 2—A map representing a change in ball velocities before (h_1) and after (h_2) the impact of a ball against the paddle with a velocity (V_h) and attitude (θ_3, θ_4).
- Map 3—A map giving the relation between the ball's velocity just after the impact (h_2) and the landing point (p_r) and time of the returned ball (t_r).

These maps are implemented by means of LWR. We use LWR on memorized data to fit a planar local model at each point an input–output

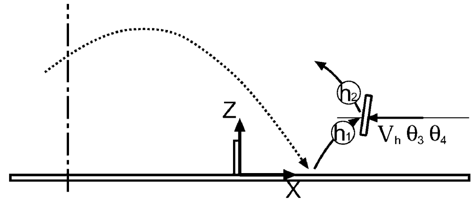

Fig. 4. Map 2.

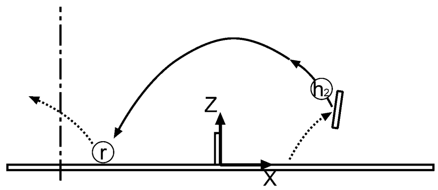

Fig. 5. Map 3.

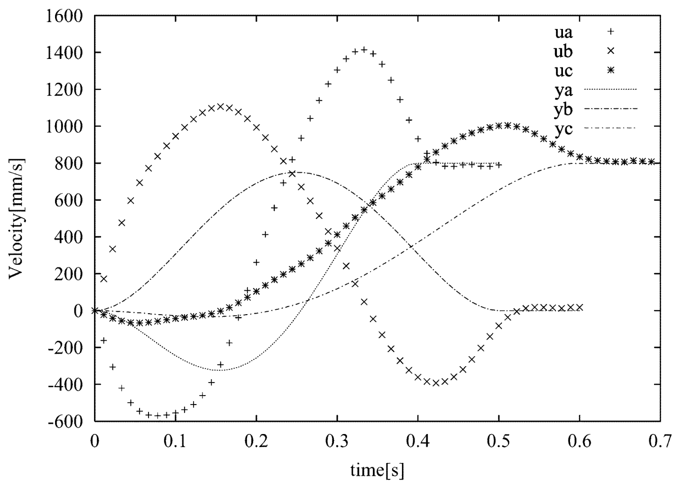

Fig. 6. Patterns A, B, and C (ui: Dearned input. yi: Desired output).

relation is to be predicted. Each data point is weighted by a function of its distance to the desired point in the regression and the model parameters are determined by the least squares algorithm.

Learning [map 1] is implemented by observing the incoming balls hit by a human opponent or a pitching machine and storing the ball's state at the event m and h_1. Once learning [map 1] is completed, the robot can hit almost every kind of the incoming ball using [map 1]. Learning [map 2] and [map 3] are implemented by observing the ball hit by the paddle with a certain attitude and velocity given at random and storing ball velocities before and after the impact, the paddle's attitude and velocity, and the landing point and time of the returned ball. Once learning [map 2] and [map 3] is completed, the control variables V_h and (θ_3, θ_4) are determined using the inverse maps of [map 2] and [map 3]. These inverse maps exist because the physical phenomena shown in Figs. 3 and 4 are governed by the well-known dynamics of collision and flight.

B. Paddle Trajectory

Given virtual targets using the above-mentioned input–output maps, we can define a trajectory of the paddle which attains the designated final position (p), velocity (v), and acceleration (a) corresponding to the virtual targets. We use a fifth-order polynomial as the position trajectory of each axis and set the velocity trajectory of the form

$$v(t) = c_1 t^4 + c_2 t^3 + c_3 t^2 \tag{1}$$

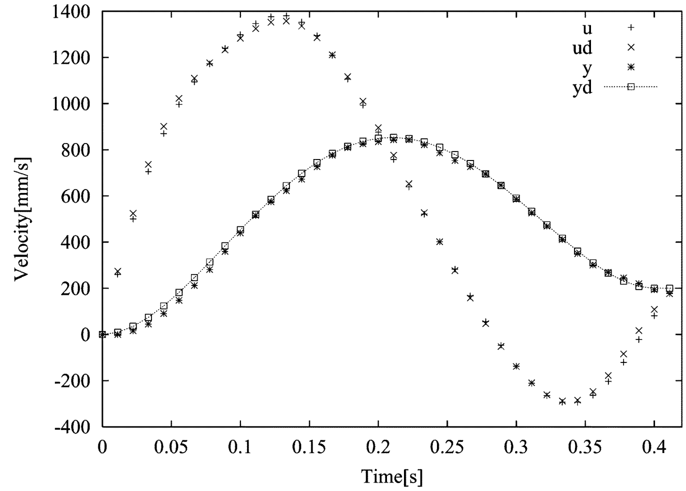

Fig. 7. New input and output trajectories generated by the patterns A, B, and C (u: Input calculated by the proposed method. y: Output corresponding to u. (ud, yd): Desired input/output).

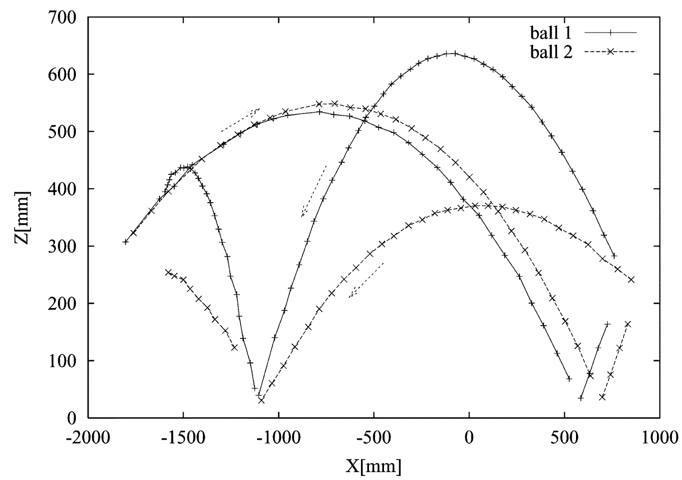

Fig. 8. Ball trajectory in x–y plane (two consecutive hits).

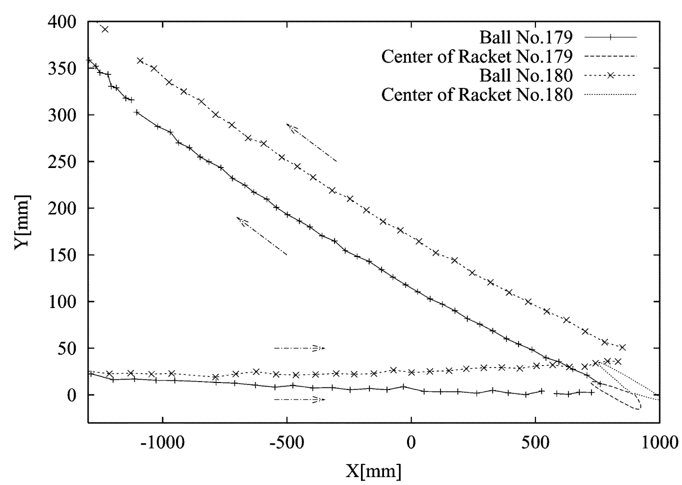

Fig. 9. Ball and paddle trajectories in x–y plane correspondin to Fig. 8.

considering the boundary conditions

$$\begin{aligned} p(0) &= p_i, \quad v(0) = 0, \quad a(0) = 0 \\ p(t_f) &= p_f, \quad v(t_f) = v_f, \quad a(t_f) = 0 \end{aligned} \tag{2}$$

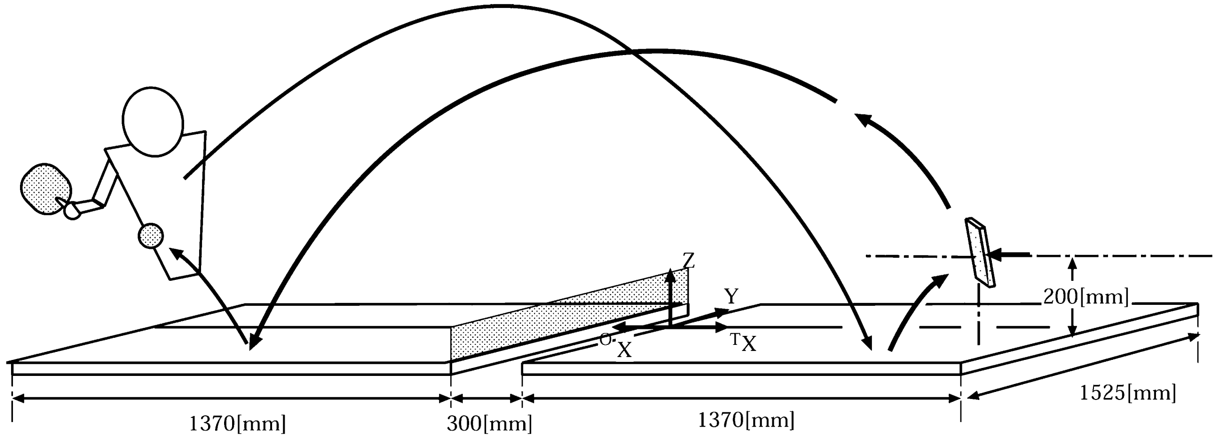

Fig. 10. Experimental environment of the "rally task."

where t_f is the time for the entire trajectory which is predetermined by taking account of the robot's torque limits. As may be seen from these conditions, the paddle motion begins at rest and ends with zero acceleration. The coefficients c_1, c_2, and c_3 are obtained by applying these boundary conditions. After completing the paddle motion to hit a ball, the paddle has to return to the waiting position as soon as possible. We use the same trajectory as (1) reversing the boundary conditions of (2).

V. Generation of Paddle Movement

The table tennis task is successfully performed by accurately achieving the stroke movement of the paddle mentioned in the previous section. The iterative learning control [8] is an effective method to accurately track a desired motion pattern without modeling the dynamics of controlled object. Unfortunately, this method is not directly applicable to the table tennis task because the desired motion patterns of the paddle vary depending on the motion of the incoming ball. In this section, we propose a method of generating a new motion pattern using the results of learning control for several motion patterns provided beforehand.

A. Motion Trajectory

We use the (1) to represent the desired position trajectory of each axis and the coefficients c_1, c_2 and c_3 are determined by the following equation:

$$\begin{bmatrix} \frac{1}{5}t_f^5 & \frac{1}{4}t_f^4 & \frac{1}{3}t_f^3 \\ t_f^4 & t_f^3 & t_f^2 \\ 4t_f^3 & 3t_f^2 & 2t_f \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} p_f \\ v_f \\ 0 \end{bmatrix} \tag{3}$$

where the initial states $(p(0), v(0), a(0))$ are all 0 and the final states (at $t = t_f$) are $(p_f, v_f, 0)$.

B. 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. Let u_a be a learned input for a given trajectory with an output y_a expressed by coefficients $\boldsymbol{c}_a = [c_{a1}, c_{a2}, c_{a3}]^T$ 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 $\boldsymbol{c}_b = [c_{b1}, c_{b2}, c_{b3}]^T$ and $\boldsymbol{c}_c = [c_{c1}, c_{c2}, c_{c3}]^T$. If the output y_d of a new trajectory is given by

$$y_d(t) = k_a y_a(t) + k_b y_b(t) + k_c y_c(t) \tag{4}$$

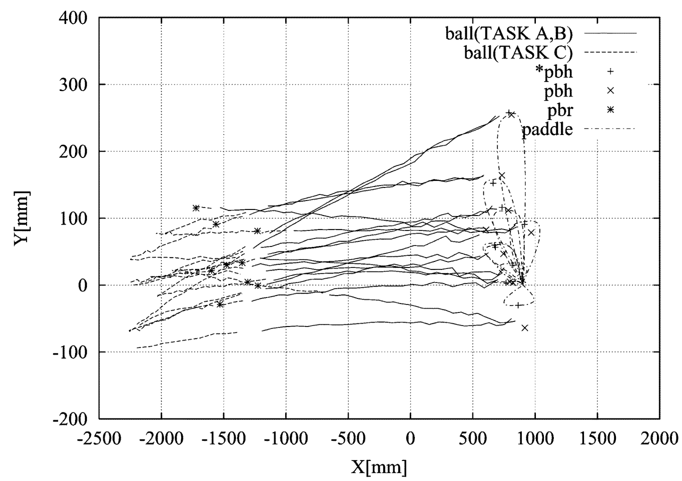

Fig. 11. Ball trajectory in the "rally task" (in x–y plane).

as a linear combination of given outputs, we can obtain the corresponding input $\boldsymbol{u}$ as a linear combination

$$\boldsymbol{u} = k_a \boldsymbol{u}_a + k_b \boldsymbol{u}_b + k_c \boldsymbol{u}_c \tag{5}$$

where the coefficients $[k_a, k_b, k_c]^T$ are determined by solving

$$\begin{bmatrix} c_{a1} & c_{b1} & c_{c1} \\ c_{a2} & c_{b2} & c_{c2} \\ c_{a3} & c_{b3} & c_{c3} \end{bmatrix} \begin{bmatrix} k_a \\ k_b \\ k_c \end{bmatrix} = \begin{bmatrix} c_{d1} \\ c_{d2} \\ c_{d3} \end{bmatrix} \tag{6}$$

where $\boldsymbol{c}_d = [c_{d1}, c_{d2}, c_{d3}]^T$ are coefficients defining the output y_d and the 3×3 matrix on the left side of (6) has to be nonsingular. It should be noted that this method allows different durations of the motion trajectories (different t_fs are acceptable).

C. Experimental Verification

Fig. 7 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 Fig. 6).

This is an application of the proposed method to generate the "hitting motion" that starts from a home position (at $t = 0$) and ends at a predicted t_h with desired virtual targets $(p_h, V_h, \theta_3, \theta_4)$. In Fig. 7, $t_h = 0.4$ [seconds] and $V_h = 200$ [millimeters/second]. From this figure,

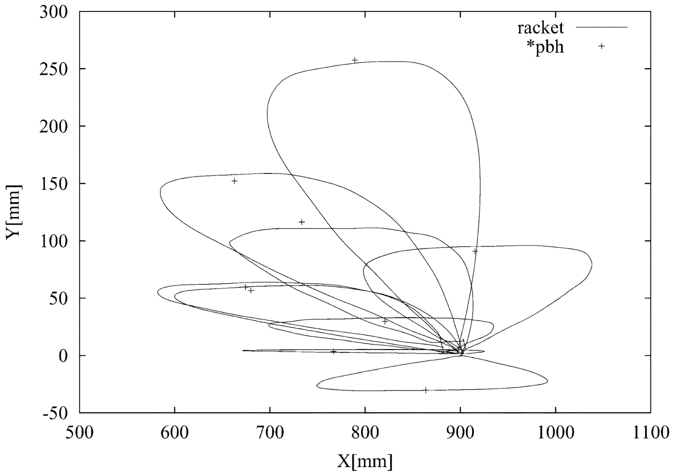

Fig. 12. Paddle trajectory x in the "rally task."

we can see that a given stroke movement of the paddle is accurately achieved by the input determined using the proposed method.

VI. Experimental Results

A. Experimental Results of the "Ball Controlling Task"

We evaluated the capability of controlling the flight duration of the returned ball with the acquired maps. We fixed the desired landing point as $x = -1100$ [millimeters], $y = 300$ [millimeters] and set the desired flight duration as 0.5 [seconds] and 0.7 [seconds], alternatively. We can see from Figs. 8 and 9 that the robot achieves the different ball trajectory with the same landing position by controlling the flight duration.

B. Experimental Results of the Rally Task With a Human

We also demonstrate the robot rally with a human (Fig. 10). The "rally task" means the table tennis rally that people generally play. We consider it as the repetition of "ball controlling task" described above. 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 ($dt_{hr} = 0.55$ [seconds]) to a desired landing point ($p_{rx} = 1550$ [millimeters], $p_{ry} = 0.3 \times {}^*p_{bhy}$) for the opponent's easy hitting, where ${}^*p_{bhy}$ is a predicted impact point.

Figs. 11 and 12 show a part of the ball and paddle trajectories on the x–y plane in the rally where the waiting position of the paddle is $x = 900$ [millimeters], $y = 0$ [millimeters]. 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. You can see a short movie of "rally task experiment" at http://robotics.me.es.osaka-u.ac.jp/MiyazakiLab/ Research/pingpong/.

VII. 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 rallies with a human opponent also have been reported.

References

[1] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek, "Sequential composition of dynamically dextrous robot behaviors," *Int. J. Robot. Res.*, vol. 18, no. 6, pp. 534–555, 1999.

[2] R. A. Schmidt and T. D. Lee, "Motor control and learning a behavioral emphasis," *Human Kinetics*, to be published.

[3] R. L. Andersson, *A Robot Ping-Pong Player: Experiment in Real-Time Intelligent Control*. Cambridge, MA: MIT Press, 1988.

[4] M. Ramanantsoa and A. Duray, "Toward a stroke construction model," *Int. J. Table Tennis Sci.*, no. 2, pp. 97–114, 1994.

[5] F. Miyazaki, M. Takeuchi, M. Matsushima, T. Kusano, and T. Hashimoto, "Realization of table tennis task based on virtual targets," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, Washington, DC, 2002, pp. 3844–3849.

[6] M. Bühler, D. E. Koditschek, and P. J. Kindlmann, "Planning and control of a juggling robot," *Int. J. Robot. Res.*, vol. 13, no. 2, pp. 101–118, 1994.

[7] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning," *Artif. Intell. Rev.*, vol. 11, pp. 11–73, 1997.

[8] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of robots by learning," *J. Robot. Syst.*, vol. 1, no. 2, pp. 123–140, 1984.

The Effects of Partial Observability When Building Fully Correlated Maps

Juan Andrade-Cetto and Alberto Sanfeliu

***Abstract*—This paper presents an analysis of the fully correlated approach to the simultaneous localization and map building (SLAM) problem from a control systems theory point of view, both for linear and nonlinear vehicle models. We show how partial observability hinders full reconstructibility of the state space, making the final map estimate dependent on the initial observations. Nevertheless, marginal filter stability guarantees convergence of the state error covariance to a positive semidefinite covariance matrix. By characterizing the form of the total Fisher information, we are able to determine the unobservable state space directions. Moreover, we give a closed-form expression that links the amount of reconstruction error to the number of landmarks used. The analysis allows the formulation of measurement models that make SLAM observable.**

***Index Terms*—Estimation, localization, mapping, mobile robots, simultaneous localization and map building (SLAM).**

I. Introduction

The study of stochastic models for simultaneous localization and map building (SLAM) in mobile robotics has been an active research topic for over 15 years. One of the main difficulties in providing a robust solution to the SLAM problem resides in the fact that it is a fully correlated state estimation problem. That is, the state space constructed by appending the robot pose and the landmark locations is fully correlated, which is a situation that produces partial observability. Moreover, the modeling of map states as static landmarks yields a partially controllable state vector.

The study of fully correlated estimation due to geometric constraints was originally addressed by Durrant-Whyte [1]. Within the Kalman filter (KF) approach to SLAM, seminal work by Smith and Cheeseman [2] suggested that, as successive landmark observations take place, the

Manuscript received September 9, 2003; revised April 6, 2004 and October 10, 2004. This paper was recommended for publication by Associate Editor W. Chung and Editor S. Hutchinson upon evaluation of the reviewers' comments. This work was supported by the Spanish Council of Science and Technology under Project DPI-2001-2223.

The authors are with the Institut de Robòtica i Informàtica Industrial, 08028 Barcelona, Spain (e-mail: cetto@iri.upc.es).

Digital Object Identifier 10.1109/TRO.2004.842342