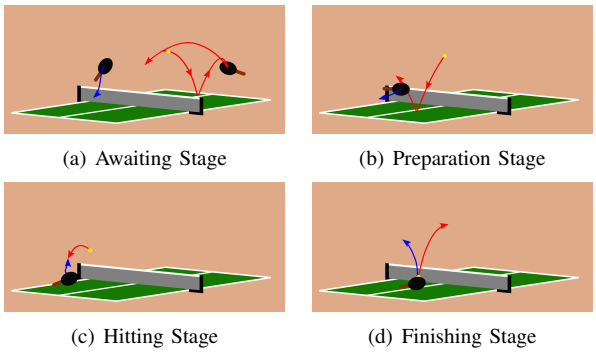
Learning Anticipation Policies for Robot Table Tennis

Zhikun Wang[1,2], Christoph H. Lampert[3], Katharina Mülling[1,2], Bernhard Schölkopf[1], Jan Peters[1,2]

***Abstract*— Playing table tennis is a difficult task for robots, especially due to their limitations of acceleration. A key bottleneck is the amount of time needed to reach the desired hitting position and velocity of the racket for returning the incoming ball. Here, it often does not suffice to simply extrapolate the ball's trajectory after the opponent returns it but more information is needed. Humans are able to predict the ball's trajectory based on the opponent's moves and, thus, have a considerable advantage. Hence, we propose to incorporate an anticipation system into robot table tennis players, which enables the robot to react earlier while the opponent is performing the striking movement. Based on visual observation of the opponent's racket movement, the robot can predict the aim of the opponent and adjust its movement generation accordingly. The policies for deciding how and when to react are obtained by reinforcement learning. We conduct experiments with an existing robot player to show that the learned reaction policy can significantly improve the performance of the overall system.**



(a) Awaiting Stage (b) Preparation Stage

(c) Hitting Stage (d) Finishing Stage

Fig. 1. The four stages of a typical table tennis ball rally are shown with the red curve representing the ball trajectories. Blue trajectories depict the typical racket's movements of players.

I. Introduction

Playing table tennis is a challenging task, particularly for robots. The reasons vary from the robot's deficiencies in perceiving the environment to the hardware limitations that restrict the action planning. Hence, robot table tennis has been used by many groups as a benchmark task for high-speed vision [1], [4], fast movement generation [3], [11], learning [9], [10] and many other subproblems in robotics. For example, a recent approach [11] allowed a Barrett WAM™ robot arm to successfully return 85% of the balls served to a specific region by a ball launcher, but its success rate would have degenerated by an enlargement of the region of incoming balls. This problem also occurs to other robot table tennis players that generate the hitting plan without considering the moves of the opponent. Despite that the robot is faster and more precise than a human being, even a beginner player would have the upper hand simply by choosing regions which the robot cannot reach in time if it bases its actions only on the trajectory of the incoming ball.

A major cause of failure is the robot's limitation of acceleration, which severely restricts its movement abilities. As a result, not every movement can reach the desired *virtual hitting state* [11], i.e., the position, orientation and velocity for the racket at a certain time, and return the ball to the opponent's court. This limitation can best be illustrated using typical human table tennis movements [13], which consist of four stages, as shown in Figure 1. In the awaiting stage, the ball moves towards the opponent and is returned back. The player moves to the *awaiting pose* and stays during this stage. The preparation stage starts when the ball passes the net. The arm swings backwards to a *preparation pose*. A virtual hitting state is produced at the beginning of the hitting stage. The racket moves towards this hitting state and hits the ball at the end of the hitting stage. It follows through in the finishing stage and recovers to the awaiting pose. The duration of the hitting stage is constant for expert players and lasts approximately 80ms. Even against a slower opponent, which allows less than 300ms for the robot's hitting movement, this short time often does not suffice for the racket to reach the hitting state from the preparation position. Therefore, many desired hitting movements are not feasible.

In general, the path between the preparation position and the hitting position influences the chances of successfully returning the ball. Therefore, it is essential to adjust the preparation position based on all available information in order to increase the success probability. This idea coincides with the insight that skilled table tennis players rely heavily on good *anticipation*. In this context, anticipation [2] in striking sports is the abilities to predict the opponent's intention from incomplete cues, and to react based on it. The player maps partial movement of the opponent to a potential *target* to which the opponent plans to return the ball. Therefore, anticipation allows the robot to prepare better for the incoming ball by adjusting the preparation position before it can extrapolate the ball's trajectory. Humans can be trained to obtain the proficiency in tracking the opponent's movement, exploiting prior knowledge to predict the target, and reacting accordingly at the right time. This insight opens the question of whether such perception, prediction and

[1]Max Planck Institute for Intelligent Systems, Spemannstr. 38, 72076 Tübingen, Germany.

[2]Technische Universität Darmstadt, Intelligent Autonomous Systems Group, Hochschulstr. 10, 64289 Darmstadt, Germany.

[3]Institute of Science and Technology Austria, Am Campus 1, IST Austria, A-3400 Klosterneuburg, Austria.

reaction can also be learned and utilized in robotics.

In this paper, we build an anticipation system for robot table tennis players, which allows the robot to decide a preferable preparation pose before the opponent finishes the stroke and, thus, better respond to the incoming ball. We formulate the anticipation process of choosing the optimal preparation pose and time to react as a Markov decision process (MDP) in Section II. The decision making relies on the approaches to perception of the environment and prediction of the opponent's intention, which are presented in Section III. Subsequently, we incorporate the anticipation system into an existing robot table tennis setup and evaluate the performance in Section IV. Experimental results show that the learned policies can significantly improve the performance of this robot player.

II. Learning Reaction Policies

The essence of the anticipation process is decision making based on perception and prediction, of which two fundamental issues have to be addressed for the robot table tennis setup. First, the uncertainties in the prediction and the outcome need to be considered. Given an observed partial opponent's movement, the aim of the opponent can often be predicted. However, this predicted target is subject to uncertainty arising from several sources: the noise in the perception, the lack of adequate prior knowledge for prediction, and the fact that the opponent may still change the target before the racket hits the ball. Furthermore, even if the target is fully revealed, the outcome, i.e., success of returning the ball to the opponent's court, is not deterministic as the underlying dynamics of the robot arm are often too complicated to be modeled precisely at high speed. Therefore, the decision making algorithm should be able to deal with the uncertainties from these sources.

The second fundamental issue is the time-accuracy trade-off. Due to the incremental observations, the prediction accuracy increases while the opponent's racket is moving towards the ball. Hence, there arises a trade-off between prediction accuracy and the time left for the system to respond to the predicted target, e.g., to move the arm from the awaiting pose to the desired preparation pose, as both influence the success probability of the taken action. Also note that once the reaction is triggered, the moving from the awaiting pose to the preparation pose is no longer trivial and the robot can hardly change the preparation pose again. Hence, it is essential to obtain a policy that triggers the best reaction at the right time.

Modeling the problem of how and when to react as a MDP, we learn the optimal policy by reinforcement learning. The uncertainties are naturally encoded in the stochastic transition of the model, and the reaction timing is decided by following the optimal policy. We can learn and improve the policy from accumulated experience. We notice that the idea of considering the optimal stopping problem as decision making in MDPs is related to the previous work in [12].

A. Markov Decision Process Models

We formulate the process of anticipation as a stochastic MDP. The stochastic transition takes the uncertainties of the prediction and the outcome into account. Choosing the optimal reaction time can be transformed into an optimal stopping problem in a stochastic process. The decision that adjusts the preparation position or keeps waiting is made by maximizing the expected future reward it leads to, i.e., the probability of successfully returning the ball in this case.

Specifically, we want to maximize the future expected reward with respect to a policy π: $J(\pi) = \mathbb{E}[\sum_{t=1}^{T} r_t]$. The environment states $\mathbf{s}_t$ with time indices $t = 1 \dots T$ are obtained by the perception and prediction mechanisms with high frequency. At every time step t, the system is in a state $\mathbf{s}_t \in \mathcal{S}$ and can take an action $\mathbf{a}_t \in \mathcal{A}$ in accordance to the policy π, i.e, $\mathbf{a}_t = \pi(\mathbf{s}_t)$. The rewards are given by $r_t = \mathcal{R}(\mathbf{s}_t, \mathbf{a}_t)$ and switching between states is governed by the transition probabilities $\mathcal{P}$ as $\mathbf{s}_{t+1} \sim \mathcal{P}(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$. As the objective function shows, we can view this problem as a finite horizon, episodic MDP.

For our problem, we define the state $\mathbf{s} \in \mathcal{S} = \mathbb{R}^n \bigcup \{\mathbf{s}_0, \mathbf{s}_1\}$ by a set that consists of a continuous n-dimensional space plus two terminal states corresponding to failure and success of returning. The n-dimensional space combines all relevant aspects of the environment, e.g., the state of the ball, the state of the robot arm, and the prediction of the target with associated uncertainty.

In each state, the system has to decide whether to wait for more information or to trigger the reaction and move the robot towards a preparation pose. The action $\mathbf{a} \in \mathcal{A} \subset \mathbb{R}^3$ corresponds to the offset to adjust the racket position from the awaiting position. Taking the action $\mathbf{a} = \mathbf{0}$ means to stay and wait. The system transfers to a new state with an unknown transition model $\mathcal{P}$, determined by the changes of environment until more information of perception and prediction is available. Once there is a non-zero action $\mathbf{a}$, the reaction is triggered and cannot be changed due to the accumulating momentum. Therefore, the process transits to a terminal state indicating whether the striking motion was successful. The reward r_t is only non-zero in those two terminal states, being -1 and 1, respectively.

B. Policy Learning with Function Approximation

An explorative stochastic policy $\pi(\mathbf{a}|\mathbf{s})$ stands for the probability of choosing action $\mathbf{a}$ in state $\mathbf{s}$. We can define the state-action value function $Q^\pi(\mathbf{s}, \mathbf{a})$ of a policy π as

$$Q^\pi(\mathbf{s}, \mathbf{a}) = \mathcal{R}(\mathbf{s}, \mathbf{a}) + \int_{\mathbf{s}' \in \mathcal{S}} \mathcal{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a}) \int_{\mathbf{a}' \in \mathcal{A}} \pi(\mathbf{a}'|\mathbf{s}') Q^\pi(\mathbf{s}', \mathbf{a}') d\mathbf{a}' d\mathbf{s}',$$

which measures the expected future reward when taking action $\mathbf{a}$ in state $\mathbf{s}$ and following the policy π thereafter. In the terminal states, the values of the Q functions are $Q^\pi(\mathbf{s}_0, \mathbf{a}) = -1$ and $Q^\pi(\mathbf{s}_1, \mathbf{a}) = 1$ for all $\mathbf{a}$.

The optimal deterministic policy π^* chooses the action that maximizes the value of its corresponding Q function,

Algorithm 1: The LSPI algorithm iteratively updates the approximation parameters and the optimal policy.

Input : previously obtained samples $\mathcal{D}$
Output: the approximation parameters $\mathbf{w}$ the corresponding policy π

$\mathbf{w}' = 0$;
repeat
  $\mathbf{w} = \mathbf{w}'$;
  $\pi(\mathbf{s}) = \arg\max_{\mathbf{a}} \boldsymbol{\phi}(\mathbf{s}, \mathbf{a})^T \mathbf{w}$;
  Estimate $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{b}}$ according to Eq. (1), (2);
  $\mathbf{w}' = \tilde{\mathbf{A}}^{-1} \tilde{\mathbf{b}}$;
until $\mathbf{w} \approx \mathbf{w}'$;

i.e., $\pi^*(\mathbf{s}) = \arg\max_{\mathbf{a}} Q^{\pi^*}(\mathbf{s}, \mathbf{a})$. The value function of π^* can be written as

$$Q^{\pi^*}(\mathbf{s}, \mathbf{a}) = \int_{\mathbf{s}' \in \mathcal{S}} \mathcal{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a}) \left(\max_{\mathbf{a}' \in \mathcal{A}} Q^{\pi^*}(\mathbf{s}', \mathbf{a}') \right) d\mathbf{s}'.$$

Given the transition model $\mathcal{P}$, the optimal policy can be obtained by iteratively updating first the Q value function then the policy. In practice, integration with respect to the transition probabilities is replaced by sampling. As the above MDPs have continuous state and action spaces, we employ a linear function approximation architecture to make the policy learning and decision making tractable.

We approximate the Q function using a function approximator that is linear in its basis functions, given by $\hat{Q}^{\pi}(\mathbf{s}, \mathbf{a}; \mathbf{w}) = \boldsymbol{\phi}(\mathbf{s}, \mathbf{a})^T \mathbf{w}$, where $\boldsymbol{\phi}(\mathbf{s}, \mathbf{a})$ consists of basis functions $\phi_1(\mathbf{s}, \mathbf{a}) \ldots \phi_k(\mathbf{s}, \mathbf{a})$. To take the two terminal states into account, we include two specific feature functions $\phi_1(\mathbf{s}, \mathbf{a}) = I(\mathbf{s} = \mathbf{s}_0)$ and $\phi_2(\mathbf{s}, \mathbf{a}) = I(\mathbf{s} = \mathbf{s}_1)$ that indicate whether the current state is a terminal state.

The approximation parameters $\mathbf{w}$ are learned with the Least-Squares Policy Iteration (LSPI) algorithm [7]. Given the finite set of samples $\mathcal{D} = \{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}'_i) | i = 1, \ldots, L\}$, the estimates

$$\tilde{\mathbf{A}} = \frac{1}{L} \sum_{l=1}^{L} \boldsymbol{\phi}(\mathbf{s}_i, \mathbf{a}_i) (\boldsymbol{\phi}(\mathbf{s}_i, \mathbf{a}_i) - \boldsymbol{\phi}(\mathbf{s}'_i, \pi(\mathbf{s}'_i)))^T, \quad (1)$$

$$\tilde{\mathbf{b}} = \frac{1}{L} \sum_{l=1}^{L} \boldsymbol{\phi}(\mathbf{s}_i, \mathbf{a}_i) r_i, \quad (2)$$

are used to iteratively update first the approximation parameters $\mathbf{w}$ then the corresponding policy π. As shown in Algorithm 1, LSPI can deal with both continuous and discrete state and action spaces.

The LSPI algorithm learns the policy offline. However, the environment, especially the opponent's behavior may be changing from time to time. As the recent samples are very helpful for making decisions in next plays, it is better to update the policy on-the-fly. Additionally, the learned optimal policy is biased by sample distribution. Therefore, learning the global optimal policy demands sufficient exploration.

Algorithm 2: The online LSTD-Q algorithm learns the policy with sufficient exploration.

Input : previously obtained samples $\mathcal{D}$
Initialize $\mathbf{w}$, π, $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{b}}$ by Algorithm 1;
foreach *encountered state* $\mathbf{s}$ **do**
  Take action $\mathbf{a} = \pi(\mathbf{s})$;
  Randomly change $\mathbf{a}$ with probability ϵ;
  Observe new state $\mathbf{s}'$ and reward r;
  $\mathcal{D} \leftarrow \mathcal{D} \bigcup \{(\mathbf{s}, \mathbf{a}, r, \mathbf{s}'\}$;
  $\mathbf{a}' = \arg\max_{\mathbf{a}'} \boldsymbol{\phi}(\mathbf{s}', \mathbf{a}')^T \mathbf{w}$;
  $\tilde{\mathbf{A}} \leftarrow \tilde{\mathbf{A}} + \boldsymbol{\phi}(\mathbf{s}, \mathbf{a})(\boldsymbol{\phi}(\mathbf{s}, \mathbf{a}) - \boldsymbol{\phi}(\mathbf{s}', \mathbf{a}'))^T$;
  $\tilde{\mathbf{b}} \leftarrow \tilde{\mathbf{b}} + \boldsymbol{\phi}(\mathbf{s}, \mathbf{a}) r$;
  $\mathbf{w} \leftarrow \tilde{\mathbf{A}}^{-1} \tilde{\mathbf{b}}$;
  $\pi(\mathbf{s}) = \arg\max_{\mathbf{a}} \boldsymbol{\phi}(\mathbf{s}, \mathbf{a})^T \mathbf{w}$;

We apply a modified LSTD-Q [7] learning algorithm to make decisions and update the policy, as show in Algorithm 2. It takes actions according to the current policy, and ensures exploration with the ϵ-greedy strategy. The value function and the corresponding policy are continually updated with observed transition and rewards. The learned policy by LSTD-Q converges slowly to the optimal policy given samples $\mathcal{D}$ as it only performs one iteration. In order to compute the optimal policy more efficiently, we execute the LSPI algorithm offline to iteratively update the policy.

III. Perception and Prediction for Robot Table Tennis

The decision making needs the perception of the environment and the prediction of the opponent's target. We develop a vision system[1] to perceive the environment's state with high frequency. Besides the table tennis ball's state, the system tracks the opponent's racket as its orientation and movement direction provide strong indication of the opponent's intention. Based on the information we obtained, Gaussian process regression (GPR) predicts the opponent's target utilizing a *knowledge base* that contains previous experiences.

A. The Vision System

To track the opponent's racket, the vision system employs three Prosilica GE640C cameras mounted above the robot. Their position and direction are chosen so that the opponent can always be seen from every camera and, hence, the racket surface is fully visible from at least two cameras. These cameras are synchronized and calibrated to the coordinate system of the robot. Each camera outputs a stream of frames with frequency of 60Hz, ensuring the possibility of real-time racket tracking. We divide the tracking problem into localizing the racket in each camera and reconstructing its 3D configuration from camera pairs. These problems are both solved in parallel on a multi-core computer.

[1]As it is the first step towards good anticipation, only visual information is used but later we hope to also make use of auditory information.

For each camera, we use linear-chain Condition Random Fields (CRF) [16] for treating the problem of tracking the racket in a sequence of frames. In the frame $\mathbf{I}_t$ indexed by time step t, we compute the most likely racket configuration $\boldsymbol{\theta}_t$ represented by a sub-window with fixed size. We also include the shift of configurations in consecutive frames in the model as the speed of the racket is constrained by the physical motor limits of a human. The joint conditional probability of configurations given N frames is given by

$$P(\boldsymbol{\theta}_{1\ldots N}|\mathbf{I}_{1\ldots N}) = \frac{1}{Z(\mathbf{I}_{1\ldots N})} \exp\left\{\sum_{t=1}^{N} \boldsymbol{\alpha}^T \mathbf{f}(\boldsymbol{\theta}_t, \mathbf{I}_t) + \sum_{t=2}^{N} \boldsymbol{\beta}^T \mathbf{g}(\boldsymbol{\theta}_{t-1}, \boldsymbol{\theta}_t)\right\},$$

where $Z(\mathbf{I}_{1\ldots N})$ is the partition function, vector $\mathbf{g}(\boldsymbol{\theta}_{t-1}, \boldsymbol{\theta}_t)$ measures the differences of the image coordinates between two consecutive configurations, vector $\mathbf{f}(\boldsymbol{\theta}_t, \mathbf{I}_t)$ represents the features of a configuration, and $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are corresponding parameters in the CRF model. We use the local color histogram in the configuration as the features $\mathbf{f}$, with the HSV space quantized into one hundred bins[2].

From ten labeled shots of the racket movements, the parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are learned by maximizing the likelihood of the labeled configurations. Subsequently, the tracking at time t is to find the configuration with the maximal marginal probability $P(\boldsymbol{\theta}_t|\mathbf{I}_{1\ldots t})$, which can be decomposed as

$$P(\boldsymbol{\theta}_t|\mathbf{I}_{1\ldots t}) \propto \exp\left(\boldsymbol{\alpha}^T \mathbf{f}(\boldsymbol{\theta}_t, \mathbf{I}_t)\right) \left\{\sum_{\boldsymbol{\theta}_{t-1}} P(\boldsymbol{\theta}_{t-1}|\mathbf{I}_{1\ldots t-1}) \exp\left(\boldsymbol{\beta}^T \mathbf{g}(\boldsymbol{\theta}_{t-1}, \boldsymbol{\theta}_t)\right)\right\}.$$

As the features are a histogram, we can obtain $\boldsymbol{\alpha}^T \mathbf{f}(\boldsymbol{\theta}_t, \mathbf{I}_t)$ efficiently for all $\boldsymbol{\theta}_t$ using fast convolution. The second factor can be approximately estimated by only considering $\boldsymbol{\theta}_{t-1}$ whose distance from $\boldsymbol{\theta}_t$ is bounded by a constant. Therefore, fast convolution is also applicable. As a result, we can efficiently compute the marginal probability $P(\boldsymbol{\theta}_t|\mathbf{I}_{1\ldots t})$.

From the parameters $\boldsymbol{\alpha}$, we can derive a score for every color. We highlight the shape of the racket by removing all pixels whose scores are below a fixed threshold. An example is shown in Figure 2. Note that the racket may not always be detected correctly, especially when its surface's normal vector is perpendicular to the camera direction, resulting in an incorrect configuration. However, it is still visible from the other two cameras. Hence, the configuration can be corrected in the 3D reconstruction stage, as shown in Figure 2.

We reconstruct the racket's 3D configuration from matched points for every camera pair where the racket is visible. The racket surface has no texture, rendering the matching of the keypoints difficult. Consider that the projection of the racket on an epipolar line [6] forms a line segment. The camera pairs have approximately horizontal epipolar lines. Therefore, for a pair of epipolar lines, we match the left most points with a score higher than the threshold, and the right most points as well. Those pairs of matched points are converted into a set of 3D points.

[2]We group all pixels in 1000 captured images into 100 clusters by k-means, and quantize the HSV space by nearest neighbor mapping.

Fig. 2. The images show the cropped scenes from the cameras of the vision system together with the reconstructed racket surface. We highlight the pixels on the racket surface whose scores are higher than the threshold by the red color. Green dots are matched points from a pair of cameras. Although the racket is not detected in the middle image due to the viewing angle, it can be recovered from the other cameras.

Although noise is inevitable, we expect that the majority of the matched points are correct and, hence, will be on the same plane. We apply RANSAC [5] to robustly estimate the normal vector of the racket surface, and concurrently eliminate outlier points. The procedure can be performed in parallel, with different hypotheses evaluated on different cores. The green dots in Figure 2 correspond to all matched points, from which we can recover the surface of the racket. The surface is projected into the 2D images. Therefore, we can detect and repair the incorrect configuration in a single camera.

In conclusion, we obtain a pipeline that robustly tracks the racket's position and orientation with frequency of 60Hz. Together with a real-time ball tracking system [8], the robot can perceive the state of the ball and the opponent's racket, which yield the information needed for prediction and decision making.

B. Target Prediction

The visual observations provide imperfect and incomplete information that sheds light on the opponent's intention. The obtained data contains the position of the ball, and the position and orientation of the opponent's racket with densely sampled time steps (60Hz). However, errors in the position and orientation are inevitable, which lead to inaccurate velocity estimates and erroneous acceleration estimates. Thus, we cannot precisely determine the hitting point even if the entire striking movement is observed. Moreover, despite the human players' limited acceleration, the desired target can still be changed. Hence, we instead estimate the mean and variance of the predicted target coordinates from the available information before the player hits the ball.

In the used table tennis setup [11], the robot always chooses the hitting point on the virtual hitting plane demonstrated in Figure 3. Therefore, the desired target is the point where the incoming ball's trajectory intersects with the hitting plane. We predict the X and Z coor-

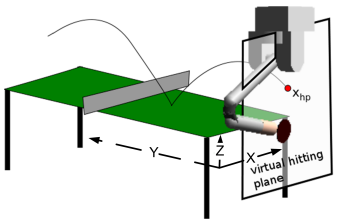
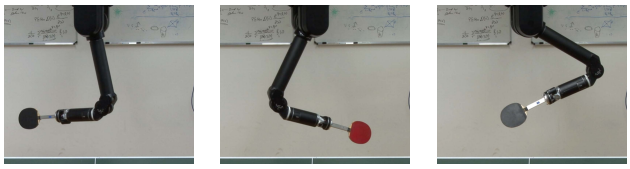


Fig. 3. The image shows the virtual hitting plane and the hitting point.



(a) forehand pose (b) backhand pose (c) middle pose

Fig. 5. Three pre-defined preparation poses are offered by the robot table tennis player. They are optimized for hitting points in different regions.

dinates of the target separately. For each axis, Gaussian process learns a non-parametric mapping from the available information to the distribution of the predicted target. We use the mean and the variance for future decision making.

The initial training data includes 600 opponent's movements along with the resulting hitting points. They serve as a preliminary knowledge base, from which the Gaussian process regression makes predictions. Equipped with the fully automatic vision system, the knowledge base can be incremented as more future plays are recorded, and the performance of prediction can improve simultaneously. However, the increase will result in a higher computational complexity, hence, the time to predict grows as well. To cope with this problem, we can adopt the local and global sparse Gaussian process approximation [15], so that the prediction complexity $\mathcal{O}(B^2)$ is controlled with a budget B.

IV. Experiments

We present the experimental results on the designed anticipation system in this section.

A. Robot Table Tennis Player

We use the existing robot table tennis player [11] to evaluate the effectiveness of the anticipation system. We use the SL framework [14], which consists of a real-world setup and a sufficiently realistic simulation. The setup includes a Barrett WAM$^{\text{TM}}$ arm with seven degrees of freedom that is capable of high speed motion. A racket is attached to the end-effector. Table, racket and ball are compliant with the international rules of human table tennis. In this paper, we use the real-world setup to collect data including opponent's movement and ball's trajectory, as demonstrated in Figure 4. The evaluation is performed in the simulated environment.

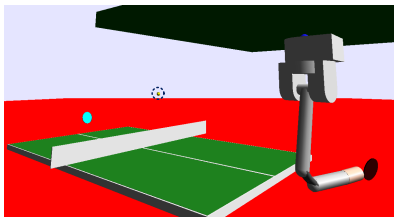


Fig. 4. The image shows the visualized environment information from the perception system, including states of the opponent's racket, the ball and the robot arm.

B. The Preparation Poses

The robot player offers three preparation poses that are designed for optimizing the inverse kinematics, as shown in Figure 5. We use these pre-defined preparation poses to construct the radial basis functions for approximating the action space $\mathcal{A}$. We denote the pre-defined actions, i.e., adjusting to a preparation position, by the set $\mathcal{A}_0$. In any state $\mathbf{s}$, the optimal action $\mathbf{a}^* = \arg\max_{\mathbf{a}} \hat{Q}^\pi(\mathbf{s}, \mathbf{a}; \mathbf{w})$ was chosen by maximizing the approximated Q function. The approximated $\hat{Q}$ function with a specific state $\mathbf{s}$ can be written in the form of

$$\hat{Q}^\pi(\mathbf{s}, \mathbf{a}; \mathbf{w}) = \sum_k w_k \exp\left\{-\frac{\|\mathbf{a} - \mathbf{a}_k\|^2}{2\sigma_a^2} - \frac{\|\mathbf{s} - \mathbf{s}_k\|^2}{2\sigma_s^2}\right\}$$
$$= \sum_{\mathbf{a}'_i \in \mathcal{A}_0} w_i(\mathbf{s}) \exp\left\{-\frac{\|\mathbf{a} - \mathbf{a}'_i\|^2}{2\sigma_a^2}\right\},$$

where $w_i(\mathbf{s}) = \sum_{k:\mathbf{a}_k = \mathbf{a}'_i} w_k \exp\{-\|\mathbf{s} - \mathbf{s}_k\|^2/(2\sigma_s^2)\}$. As the variance σ_a was set to be sufficiently small, the optimal action $\mathbf{a}^*$ will be very close to a pre-defined position $\mathbf{a}'_i$ with the maximal $w_i(\mathbf{s})$. Therefore, we always select actions from the pre-defined set as their striking movements are optimized. This simplification also reduces the decision making problem to determining $\mathbf{a}^* = \arg\max_{\mathbf{a}_k \in \mathcal{A}_0} \hat{Q}^\pi(\mathbf{s}, \mathbf{a}_k; \mathbf{w})$.

We evaluate the performance of exclusively using a single pre-defined preparation pose on a dataset with recorded games from the same opponent player. The dataset is partitioned into a training set of 220 rallies and a testing set of 172 rallies. The outcome of every play in the testing set is estimated from five repetitions in the simulated environment. Every pre-defined pose yields a relatively high success rate in its designated regions. However, the overall rate on the entire dataset is considerably reduced due to its poor performance in the other regions. The three poses are very different such that switching between them is not trivial. Therefore, the player without the anticipation system would achieve a success rate of 53% on the testing set.

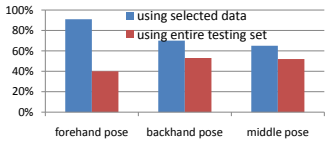


Fig. 6. The columns shows the rates of successfully returning the ball for the three preparation poses. The performance is relatively high on the selected subset whose hitting points are in their respectively designed regions. However, the performance on the entire dataset is significantly reduced.

C. Target Prediction

We evaluate the performance of target prediction on a larger dataset with the recorded striking movements from different players, using leave-one-out cross validation. We predict the hitting point before the opponent hits the ball, with features including position,

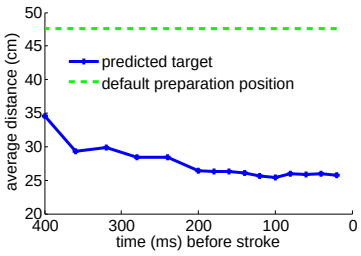


Fig. 7. The average errors before the opponent's stroke, i.e., the distance between the predicted target and the true hitting point on the X-axis, are shown by the blue curve. The green line shows the average distance between the default (middle) preparation position and the true hitting point.

velocity and orientation of the racket and position and velocity of the ball, which are extracted from the perceived information up to a specific instant in time. The prediction of the X coordinate of the target on the hitting plane is evaluated as demonstrated in Figure 7, which shows that the error of prediction decreases as the opponent finishes the stroke. Hence, late reaction benefits from better prediction due to more perceived information.

D. Reaction Policies

However, late reaction will reduce the time to move from the awaiting pose to the desired preparation position, consequently leading to lower probability of successfully returning the incoming ball. Figure 8 displays that the performance on the testing set is decreasing in general while the time for moving to the preparation position reduces. The curves show the probabilities of successfully returning the ball if the robot starts moving to the desired preparation pose after perceiving the environment at a specific time. The results suggest that the target prediction and choosing preparation pose accordingly can significantly improve the probability of successfully returning the ball from 53% to 73%, while the optimistic probability 81% shown by the red curve is the (approximate) upper bound that the algorithm can achieve given the existing setup. However, the timing of deciding the desired preparation pose can be further optimized with a learned policy that trades off the prediction accuracy and time for moving.

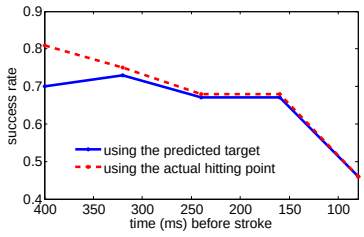


Fig. 8. The probability of successfully returning the ball drops along with the decreasing time for the robot to move from the awaiting pose to the desired preparation pose. The performance shown by the blue curve is by choosing the preparation pose based on the predicted target, while that in the red curve is the optimistic estimate by using the optimal preparation position given the true hitting point.

For each recorded rally in the training set, we simulate the outcome of every combination of the preparation pose and reaction time. We apply Algorithm 1 to learn the optimal policy. The learned policy yields the success rate of 78.5% on the testing set, which is very close to the optimistic upper bound. In comparison to the success rate 73% when the reaction is solely based on prediction, the learned reaction policy achieved a significant improvement. We also test the learned policy on a dataset obtained with different players. The anticipation system improves the success rate from 78% to 82.5%. The improvement is less significant as most of the hitting points are close to the robot and, thus, only using the middle preparation pose already achieves a high success rate on the dataset.

V. Conclusions and Future Work

We presented an approach for learning anticipation policies based on perception of the environment and prediction of the opponent's prediction, which can be used for robot table tennis players and, probably, other robot striking sports. Based on visual observation of the opponent's racket movement, the robot can predict the aim of the opponent and adjust its movement generation accordingly. An optimal policy for deciding how and when to react is learned by reinforcement learning. We conducted experiments with the existing robot player to show that the learned anticipation policy can significantly improve the performance of the overall system.

There are also many interesting directions that we will continue exploring. We will employ more types of sensors, for example microphone arrays and stereo cameras, to better perceive the environment, especially the opponent. As more information can be obtained, the problem of high-dimensional states arises, which requires more efficient algorithms for learning the optimal policy. Moreover, the anticipation mechanism can be also used in other problems where the robot interacts with humans.

References

[1] L. Acosta, J.J. Rodrigo, J.A. Mendez, GN Marichal, and M. Sigut. Ping-pong player prototype. *Robotics & Automation Magazine, IEEE*, 10(4):44–52, 2003.

[2] M. Alexander and A. Honish. Table Tennis: a Brief Overview of Biomechanical Aspects of the Game for Coaches and Players.

[3] L. Ángel, J.M. Sebastián, R. Saltarén, R. Aracil, and R. Gutiérrez. RoboTenis: design, dynamic modeling and preliminary control. In *Advanced Intelligent Mechatronics. Proceedings, 2005 IEEE/ASME International Conference on*, pages 747–752. IEEE, 2005.

[4] H. Fässler, H.A. Beyer, and J. Wen. A robot ping pong player: optimized mechanics, high perfromance 3D vision, and intelligent sensor control. *Robotersysteme*, 6(3):161–170, 1990.

[5] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

[6] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge Univ Pr, 2003.

[7] M.G. Lagoudakis and R. Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

[8] C.H. Lampert and J. Peters. Real-time detection of colored objects in multiple camera streams with off-the-shelf hardware components. *Journal of Real-Time Image Processing*, pages 1–11.

[9] M. Matsushima, T. Hashimoto, M. Takeuchi, and F. Miyazaki. A learning approach to robotic table tennis. *IEEE Transactions on Robotics*, 21(4):767–771, 2005.

[10] F. Miyazaki, M. Matsushima, and M. Takeuchi. Learning to dynamically manipulate: A table tennis robot controls a ball and rallies with a human being. *Advances in Robot Control*, pages 3137–341, 2005.

[11] K. Mülling, J. Kober, and J. Peters. A Biomimetic Approach to Robot Table Tennis. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[12] B. Póczos, Y. Abbasi-Yadkori, C. Szepesvári, R. Greiner, and N. Sturtevant. Learning when to stop thinking and do something! In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 825–832. ACM, 2009.

[13] M. Ramanantsoa and A. Durey. Towards a stroke construction model. *International Journal of Table Tennis Science*, 2:97–114, 1994.

[14] S. Schaal. The SL simulation and real-time control software package. *University of Southern California*, 2007.

[15] E. Snelson and Z. Ghahramani. Local and global sparse gaussian process approximations. In *Artificial Intelligence and Statistics*, volume 11. Citeseer, 2007.

[16] C. Sutton and A. McCallum. An Introduction to Conditional Random Fields for Relational Learning. *Introduction to statistical relational learning*, page 93, 2007.