An Optimal Trajectory Planner for a Robotic Batting Task: The Table Tennis Example

Diana Serra[1], Aykut C. Satici[2], Fabio Ruggiero[1], Vincenzo Lippiello[1] and Bruno Siciliano[1]

[1]*Department of Electrical Engineering and Information Technology, University of Naples Federico II,*
Via Claudio 21, 80125, Naples, Italy
[2]*Electrical Engineering and Computer Science, Massachusetts Institute of Technology,*
32 Vassar Street, Cambridge, Massachusetts, U.S.A.

Keywords: Optimal Trajectory Planning, Table Tennis Robot, Nonprehensile Manipulation.

Abstract: This paper presents an optimal trajectory planner for a robotic batting task . The specific case of a table tennis game performed by a robot is considered. Given an estimation of the trajectory of the ball during the free flight, the method addresses the determination of the paddle configuration (pose and velocity) to return the ball at a desired position with a desired spin. The implemented algorithm takes into account the hybrid dynamic model of the ball in free flight as well as the state transition at the impact (the reset map). An optimal trajectory that minimizes the acceleration functional is generated for the paddle to reach the desired impact position, velocity and orientation. Simulations of different case studies further bolster the approach along with a comparison with state-of-the-art methods.

1 INTRODUCTION

Interest in robotic manipulation without grasping the objects, i.e., *nonprehensile manipulation*, is recently rising. The presence of unilateral constraints increases the complexity in planning and control the robot that has to dynamically drive the manipulated object from the initial to the desired configuration. Nevertheless, as highlighted by (Lynch and Mason, 1999), nonprehensile manipulation allows controlling extra degrees of freedom. The workspace is thus increased since the manipulated object can physically overcome the kinematic limitations of the end-effector by continuously breaking and creating contacts with the robot. Therefore, a nonprehensile manipulation task is in general complex, skillful and dexterous. Normally, it can be undertaken by splitting the complex task in many simpler subtasks, usually called *primitives*, such as rolling, sliding, throwing, catching, pushing and batting. The latter is the one addressed in this paper. Batting combines catching and throwing in a single collision: a practical example is given by the table tennis game. Such a task requires so high velocities and precision that robotic companies take it as an example to display the high performances of their products. For instance, Kuka has chosen the table tennis game to promote its wares in a thrilling commercial spot (Kuka, 2014), showing

the potential abilities of robots. The Omron automation company has also broadcast a video showing its parallel Delta robot playing table tennis and coaching humans at CEATEC Japan 2015 exhibition, (Omron, 2015).

The robotic batting task can be tackled with different approaches. From the artificial intelligence point of view, for instance, the aim is to improve the performance of the robot through experience, while also dealing with temporal issues; from the planning and control points of view, instead, accurate models are researched and investigated for autonomous and fast motion execution; from the perception side, finally, fast and reliable measurements are requested. This paper is focused on planning and control aspects, inherently considering the hybrid nature of the system and real-time constraints.

Generally, a robotic system for a batting task (e.g., table tennis) should be provided by: a vision system to track the motion of the ball, a method to estimate the trajectory of the ball in free flight and its spin, a decision system to choose the paddle configuration to direct the ball towards the desired position on the opposite court, and a trajectory planner for the motion of the paddle.

The scope of this paper is to provide an optimal trajectory planner for the paddle to return the ball to the opposite court at a desired position with an im-

posed spin. At the same time, an improvement of the decision system proposed by (Liu et al., 2012) to choose the configuration of the paddle at the impact time is here described. The proposed algorithm consists of two main phases: the hybrid model solution, computing the state of the paddle at the impact time such that the control objective is satisfied, and the trajectory optimization in SE(3), providing the angular and linear trajectory of the paddle up to the impact time.

The novelties introduced by this manuscript are twofold.

- Firstly, in comparison to the state-of-the-art, the proposed method improves the control accuracy by considering a full aerodynamic model of the ball, and taking into account drag and lift forces. The computation time of the algorithm is also fast enough to guarantee it can suitably be implemented in real-time.

- Secondly, rigorous methods from calculus on manifolds are borrowed to generate an optimal trajectory on SE(3) for the paddle to strike the ball at the impact time.

2 RELATED WORK

(Andersson, 1989a) describes a tracking system and a trajectory generation for a table tennis robot player. The method uses fifth-order polynomials to generate a trajectory for the paddle intercepting the ball. The architecture is designed so that the task level control can monitor the torques of the robot during the movements of the arm, and also adjust the trajectory while the ball is in free flight. (Andersson, 1989b) shows how the processing time of a stereo vision system should be taken into account when it is applied to a quickly changing operating environment. A table tennis robot player prototype equipped with two levels control system is described by (Acosta et al., 2003). The objective of the lower level is to move the center of the paddle to the right position to intercept the ball. The high-level control defines instead the game strategy. This is done by orienting the paddle to return the ball to the desired position on the table. The prototype is capable of returning balls with a good success rate when their velocity is below 5 m/s and under small spin effects. A high-speed trajectory planner is described by (Senoo et al., 2006) and it is applied to the batting task. The authors also consider the robot dynamic model within their framework, and they rely upon a 1kHz high-speed vision system. Nevertheless, they do not go into details about the controller computational efforts, and they use a simplified version

of the reset map in which the spin of the ball after the impact is not considered. Spin is indeed relevant for the table tennis game. Moreover, the trajectory planner they use is based on polynomial primitives, which has been demonstrated to demand high joint accelerations. (Nakashima et al., 2010) show how to obtain the linear and angular ball trajectory information from the visual sensors. On the other hand, (Liu et al., 2012) propose a control method for returning a table tennis ball to a desired position with a desired spin. The method determines the state of the paddle employing the hybrid dynamic model of the ball in free flight as well as the state transition at the impact. In order to determine the control action, an approximated aerodynamic model is considered, providing a satisfactory but less accurate results. (Nakashima et al., 2014) follow up the work of (Liu et al., 2012) by focusing on the nonlinear aerodynamics solution. A finite difference method and a linearization of the discretized model are employed to solve the two boundary values problem so as to reduce the computational time. However, they present fewer case studies than (Liu et al., 2012), while the obtained results are not so easy to reproduce due to a lack of implementation details.

The artificial intelligence research community is applying many learning techniques to the robotic table tennis task. (Matsushima et al., 2005) describe an approach to perform the table tennis task based on learning. Some impressive table tennis games between two humanoid robots are performed using the adaptive trajectory prediction developed by (Zhang et al., 2012). This model involves an offline training of the parameters on the base of the recorded state of the ball. Afterwards, the model parameters are online adapted for estimation and prediction processes. Other approaches reproduce the movements learned from human demonstrations. (Mülling et al., 2010) pursue trajectory generation for the robotic table tennis task from a biomimetic point of view. In the approach introduced by (Mülling et al., 2013), the robot first learns a set of elementary table tennis hitting movements from a human table tennis teacher, and then generalizes those movements in a wider range of situations using a mixture of motor primitives approach. (Huang et al., 2013) propose an active learning approach where the initial parameters related to the paddle are computed through a locally weighted regression method. A simplified hitting scenario task is implemented by (Oubbati et al., 2013), where a robot arm hits a ball rolling on an inclined plane placed in front of the robot. The authors propose a model that autonomously generates and organizes sequences of timed actions. The timing of the move-

ments is controlled by nonlinear oscillators. Their activation and deactivation are coordinated by a hierarchical neural dynamic architecture. (Yanlong et al., 2015) find optimal striking points for a table tennis robot. The choice is based on a reward function, measuring how well the trajectory of the ball and the movement of the paddle coincide. Given the striking point, a stochastic policy over the reward is derived to evaluate prospective striking points sampled from the predicted rebound trajectory. In that approach, the resulting learning method takes into account the amount of experience data and its confidence.

Recently, researchers of aerial robotics have also been interested in the batting problem. A control approach to juggle a ball between either a human and a quadrotor or two quadrotors is showed by (Müller et al., 2011). (Silva et al., 2015) tackle the task of hitting a table tennis ball with a commercial drone. The ball tracking relies only upon the onboard camera and not upon an external motion capture system. The estimation of the hitting point is performed using a variation of regularized kernel regression, where samples are weighted according to their relevance to the task. The decision phase determines the hitting motion from a set of primitives learned from human demonstrations. Finally, (Wei et al., 2015) present a trajectory tracking control strategy for a ball juggling task on a quadrotor, based on the subspace stabilization approach. An optimal trajectory generation method is adopted to obtain a dynamically feasible, minimum jerk trajectory.

3 HYBRID DYNAMIC MODELING

The hybrid dynamics of the ball consists of the free flight aerodynamics and the impact reset map. The former is modeled through Newton's equations of motion, while the latter is a reset of the state, updated according to the impact detection. In order to analytically model the ball dynamics, the work by (Liu et al., 2012) is considered. That work is supported by simulations and experiments in several case studies and follows up a deep study on the hybrid dynamic modeling of the table tennis game, (Nakashima et al., 2010). On the other hand, a first-order dynamics for the paddle is here introduced, assuming that it is possible to directly control its velocity.

It is assumed that a point contact occurs between the ball and the paddle during the impact. Moreover, as long as the paddle is made of rubber, the rebound in the direction normal to the paddle's plane does not affect the motion of the ball in the other directions.
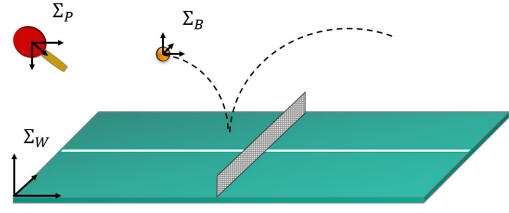

Figure 1: Ball and paddle coordinate systems.

Finally, since the mass of the paddle is usually bigger than the mass of the ball, only the velocity of the ball is considered to be affected by the impact.

According to Figure 1, let Σ_W be the fixed world frame, Σ_P be the frame placed at the center of the paddle, where the z-axis is the outward normal, and Σ_B be the frame placed at the center of the ball. Let $\mathbf{p}_B = \begin{bmatrix} p_{Bx} & p_{By} & p_{Bz} \end{bmatrix}^T \in \mathbb{R}^3$ be the position of the ball, $\mathbf{v}_B = \begin{bmatrix} v_{Bx} & v_{By} & v_{Bz} \end{bmatrix}^T \in \mathbb{R}^3$ be the velocity of the ball, $\boldsymbol{\omega}_B = \begin{bmatrix} \omega_{Bx} & \omega_{By} & \omega_{Bz} \end{bmatrix}^T \in \mathbb{R}^3$ be the spin of the ball, assumed constant during the free flight, $\mathbf{p}_P = \begin{bmatrix} p_{Px} & p_{Py} & p_{Pz} \end{bmatrix}^T \in \mathbb{R}^3$ be the position of the paddle, $\mathbf{v}_P = \begin{bmatrix} v_{Px} & v_{Py} & v_{Pz} \end{bmatrix}^T \in \mathbb{R}^3$ be the velocity of the paddle, $\boldsymbol{\omega}_P = \begin{bmatrix} \omega_{Px} & \omega_{Py} & \omega_{Pz} \end{bmatrix}^T \in \mathbb{R}^3$ be the angular velocity of the paddle, all expressed in Σ_W. Finally, let $\mathbf{R}_P \in SO(3)$ be the rotation matrix of Σ_P with respect to Σ_W.

The continuous ball and paddle dynamics are given by

$$\dot{\mathbf{p}}_B = \mathbf{v}_B, \tag{1a}$$

$$\dot{\mathbf{v}}_B = -\mathbf{g} - k_d ||\mathbf{v}_B|| \mathbf{v}_B + k_l \mathbf{S}(\boldsymbol{\omega}_B) \mathbf{v}_B, \tag{1b}$$

$$\dot{\mathbf{p}}_P = \mathbf{v}_P, \tag{1c}$$

$$\dot{\mathbf{R}}_P = \mathbf{R}_P \mathbf{S}(\boldsymbol{\omega}_P), \tag{1d}$$

where $\mathbf{g} = \begin{bmatrix} 0 & 0 & g \end{bmatrix}^T$ is the gravity acceleration, $||\cdot||$ is the Euclidean norm, $\mathbf{S}(\cdot) \in \mathbb{R}^{3 \times 3}$ is the skew-symmetric matrix operator. k_d and k_l are drag and lift parameters, respectively, and they are typically modelled as

$$k_d = k_d(\mathbf{v}_B, \boldsymbol{\omega}_B) = \frac{\rho \pi r^2 (a_d + b_d f(\mathbf{v}_B, \boldsymbol{\omega}_B))}{2m}, \tag{2a}$$

$$k_l = k_l(\mathbf{v}_B, \boldsymbol{\omega}_B) = \frac{\rho 4 \pi r^3 (a_l + b_l f(\mathbf{v}_B, \boldsymbol{\omega}_B))}{m}, \tag{2b}$$

with

$$f(\mathbf{v}_B, \boldsymbol{\omega}_B) = \frac{1}{\sqrt{1 + \frac{(v_{Bx}^2 + v_{Bx}^2)\omega_{Bz}^2}{(v_{Bx}\omega_{By} - v_{By}\omega_{Bx})^2}}}. \tag{3}$$

The meaning of other parameters, like ρ, r, a_d, a_l, b_d, b_l, and their numerical values employed in simulations are depicted in Section 6, Table 1, where a

standard table tennis ball and a rubber paddle have been considered.

Since the paddle can modify the ball velocity only at the impact time, the control action, represented by the paddle linear and angular velocities, enters the ball dynamics through the reset map. Assuming that the superscripts $-$ and $+$ represent the state before and after the impact time, respectively, the rebound equations are given by

$$\mathbf{v}_B^+ = \mathbf{v}_P + \mathbf{R}_P \mathbf{A}_{vv} \mathbf{R}_P^T (\mathbf{v}_B^- - \mathbf{v}_P) + \mathbf{R}_P \mathbf{A}_{v\omega} \mathbf{R}_P^T \omega_B^-, \tag{4a}$$

$$\omega_B^+ = \mathbf{R}_P \mathbf{A}_{\omega v} \mathbf{R}_P^T (\mathbf{v}_B^- - \mathbf{v}_P) + \mathbf{R}_P \mathbf{A}_{\omega\omega} \mathbf{R}_P^T \omega_B^-, \tag{4b}$$

where the matrices of rebound coefficients are defined as

$$\mathbf{A}_{vv} = \mathrm{diag}(1 - e_v, 1 - e_v, e_r), \mathbf{A}_{v\omega} = -e_v r \mathbf{S}(\mathbf{e}_3), \tag{5}$$

$$\mathbf{A}_{\omega v} = e_\omega r \mathbf{S}(\mathbf{e}_3), \mathbf{A}_{\omega\omega} = \mathrm{diag}(1 - e_\omega r^2, 1 - e_\omega r^2, 1),$$

where $\mathbf{e}_i \in \mathbb{R}^3$ is the unit vector along the ith-axis, $i = \{1, 2, 3\}$, while e_v and e_w are described in Section 6, Table 1.

Notice that (1b) takes into account the spin of the ball: the magnitude of the drag and lift forces and their coefficients change according to the spin, influencing the trajectory of the ball. The values of the components of ω_B determine the kind of the spin, namely: backspin, if $\omega_{By} > 0$; topspin, if $\omega_{By} < 0$; and sidespin, if $\omega_{Bz} > 0$. The effect of the spin of the ball in a table tennis game is not negligible and makes a difference between a serious table tennis player and a novice one. Serious players use spin on both their serves and rallying shots to control the ball and to force errors from their opponents.

4 BATTING PROBLEM SOLUTION

In order to accomplish the task of hitting the table tennis ball and directing it to a desired position on the opposite court with a desired spin, the paddle must intercept the ball with a certain orientation and velocity. These inputs are computed employing the dynamics of the ball in different steps. Firstly, the impact position $\mathbf{p}_B^i = \begin{bmatrix} p_{Bx}^i & p_{By}^i & p_{Bz}^i \end{bmatrix}^T \in \mathbb{R}^3$ and velocity $\mathbf{v}_B^-$ are predicted by assigning the impact time t_i, and the initial position $\mathbf{p}_B^0 = \begin{bmatrix} p_{Bx}^0 & p_{By}^0 & p_{Bz}^0 \end{bmatrix}^T \in \mathbb{R}^3$ and velocity $\mathbf{v}_B^0 = \begin{bmatrix} v_{Bx}^0 & v_{By}^0 & v_{Bz}^0 \end{bmatrix}^T \in \mathbb{R}^3$ of the ball produced by the opponent's hit. This step is accomplished solving forward the model (1a)-(1b). Once
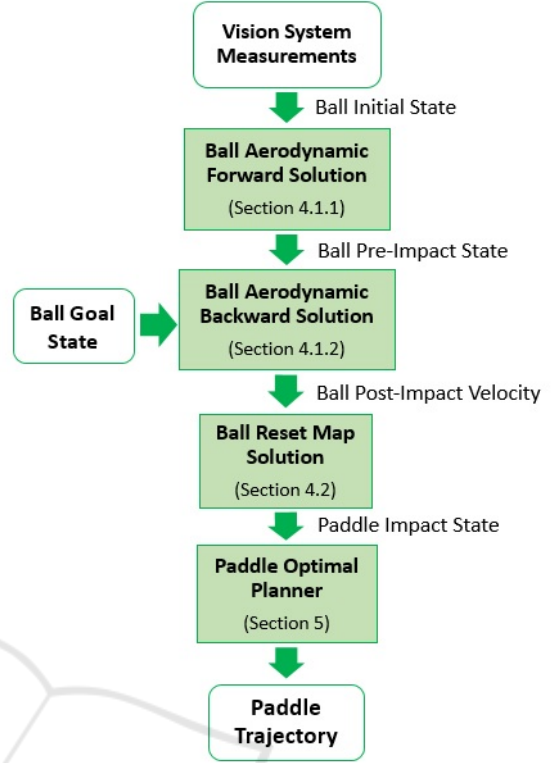

Figure 2: Scheme of the steps to generate the paddle trajectory.

the impact position is predicted, assigning the desired position of the ball on the opposite court $\mathbf{p}_B^d = \begin{bmatrix} p_{Bx}^d & p_{By}^d & p_{Bz}^d \end{bmatrix}^T \in \mathbb{R}^3$, the desired arrival time t_d, and the desired spin of the ball after the impact, $\omega_B^{+d} \in \mathbb{R}^3$, solving backward (1a)-(1b) yields the required velocity of the ball after the impact $\mathbf{v}_B^+ \in \mathbb{R}^3$. Afterwards, the orientation $\mathbf{R}_P^i$ and velocity $\mathbf{v}_P^i$ of the paddle at the impact time may be retrieved, through the reset map (4), once the spin and velocity of the ball before and after the impact are assigned. Lastly, the paddle desired impact configuration is the input of the optimal trajectory planner described in the next section. A scheme of the steps required to generate a suitable paddle trajectory is showed in Figure 2.

4.1 Aerodynamic Model Solution

In order to predict the impact state of the ball and to compute the post-impact velocity of the ball such that it reaches the goal, the equations (1a) and (1b) of the aerodynamic model are employed. However, this model is nonlinear and coupled, thus an analytic solution does not exist. In the literature (Acosta et al., 2003), (Müller et al., 2011) and others have considered a linearized model, or a simplified one for control

purposes to reduce the computations.

The following simplified model of (1a)-(1b) is employed by (Liu et al., 2012) since it has an analytic solution

$$\dot{v}_{Bx} = -c_d|v_{Bx}|v_{Bx}, \qquad \dot{p}_{Bx} = v_{Bx}, \qquad (6a)$$

$$\dot{v}_{By} = -c_d|v_{By}|v_{By}, \qquad \dot{p}_{By} = v_{By}, \qquad (6b)$$

$$\dot{v}_{Bz} = -g, \qquad \dot{p}_{Bz} = v_{Bz}, \qquad (6c)$$

where c_d is a constant drag parameter. In this paper, instead, a numerical solver suitable for real-time process is employed to solve the full aerodynamic model. The proposed algorithm consists in the two minimization problems shown below.

4.1.1 Aerodynamic Forward Solution

The first problem has the aim to predict the impact position and velocity of the ball. By assigning the impact time, the ball impact state is predicted according to the initial position and velocity of the ball produced by the opponent's hit. In detail, the algorithm solves the following minimization problem

$$\min_{\mathbf{p}_B^i,\mathbf{v}_B^-} \left\| \begin{bmatrix} \widetilde{\mathbf{p}}_B^0 \\ \widetilde{\mathbf{v}}_B^0 \end{bmatrix} - \begin{bmatrix} \mathbf{p}_B^0 \\ \mathbf{v}_B^0 \end{bmatrix} \right\|^2, \qquad (7)$$

where $\mathbf{p}_B^i$ and $\mathbf{v}_B^-$ are the optimizing variables, $\widetilde{\mathbf{p}}_B^0 = \widetilde{\mathbf{p}}_B^0(\mathbf{p}_B^i,\mathbf{v}_B^-)$ and $\widetilde{\mathbf{v}}_B^0 = \widetilde{\mathbf{v}}_B^0(\mathbf{p}_B^i,\mathbf{v}_B^-)$ are the position and the velocity of the ball at the initial time, respectively, numerically obtained by backward integrating (1a) and (1b) starting from the optimization variables $\mathbf{p}_B^i,\mathbf{v}_B^-$ at time t_i. In practice, the minimization problem (7) is solved through the *Levembert-Marquardt*'s algorithm that is well suited for real-time computations according to (Lippiello and Ruggiero, 2012) and (Cigliano et al., 2015). To speed up the convergence of the nonlinear estimation algorithm (7), the initial guess for $\mathbf{p}_B^i$ and $\mathbf{v}_B^-$ is calculated analytically from (6). The work by (Liu et al., 2012) assumes instead that the impact position of the paddle with the ball is a-priori known. Such assumption is quite restrictive: hence, here the determination of such impact position is addressed at run-time.

4.1.2 Aerodynamic Backward Solution

In order to compute the velocity $\mathbf{v}_B^+$ of the ball after the impact such that it reaches the goal position $\mathbf{p}_B^d$ at the time t_d, the following minimization problem is solved

$$\min_{\mathbf{v}_B^+} \|\widetilde{\mathbf{p}}_B^d(\mathbf{v}_B^+) - \mathbf{p}_B^d\|^2, \qquad (8)$$

where $\mathbf{v}_B^+$ is the optimization variable and $\widetilde{\mathbf{p}}_B^d(\mathbf{v}_B^+)$ is the position of the ball at time t_d numerically obtained

by forward integrating (1a) and (1b) starting from $\mathbf{p}_B^i$ at t_i computed in Section 4.1.1. The result of the minimization problem is the value of the velocity of the ball after impact so that it reaches the desired position on the opposite court at t_d as close as possible. The initial guess for $\mathbf{v}_B^+$ is calculated analytically from (6). This choice of the initial condition, together with the use of the complete aerodynamic model (1a)-(1b) in (8) to compute $\widetilde{\mathbf{p}}_B^d(\mathbf{v}_B^+)$, guarantees that the value of $\mathbf{v}_B^+$ found by solving (8) is more precise than the one provided by (6), since the latter employs only a simplified model.

4.2 Reset Map Solution

Given the pre-impact velocity of the ball $\mathbf{v}_B^-$ as in Section 4.1.1, and the post-impact one $\mathbf{v}_B^+$ as in Section 4.1.2, the paddle configuration can be now computed. Consider the YX-Euler angles (θ,ϕ) as a parametric representation of the orientation of the paddle, with $\phi \in [-\pi/2,\pi/2]$ and $\theta \in [0,\pi]$, and define $\tilde{\mathbf{v}} = \begin{bmatrix} \tilde{v}_x & \tilde{v}_y & \tilde{v}_z \end{bmatrix}^T = \mathbf{v}_B^+ - \mathbf{v}_B^-$ and $\tilde{\omega} = \begin{bmatrix} \tilde{\omega}_x & \tilde{\omega}_y & \tilde{\omega}_z \end{bmatrix}^T = \omega_B^{+d} - \omega_B^-$. The velocity and orientation of the paddle at the impact time are respectively computed through

$$\mathbf{v}_P^i = \mathbf{v}_B^- + \mathbf{R}_P^i(\mathbf{I}_3 - \mathbf{A}_{vv})^{-1}(\mathbf{R}_P^{iT}\tilde{\mathbf{v}} - \mathbf{A}_{v\omega}\omega_B^-), \quad (9a)$$

$$\mathbf{R}_P^i = \mathbf{R}_Y(\theta)\mathbf{R}_X(\phi), \qquad (9b)$$

where $\mathbf{I}_3 \in \mathbb{R}^{3\times3}$ is the identity matrix, $\mathbf{R}_i(\cdot) \in SO(3)$ is the elementary rotation matrix with $i = \{X,Y\}$, representing the rotation of an angle around the i-axis, and θ,ϕ are such that

$$\tilde{v}_z\cos\phi\sin\theta - \tilde{v}_x\cos\phi\cos\theta = \tilde{\omega}_y, \qquad (10a)$$

$$a\sin^2\phi - 2b\sin\phi + c = 0, \qquad (10b)$$

where $a = e_c^2\|\tilde{\mathbf{v}}\|^2$, $b = e_c\mathbf{e}_2S(\tilde{\mathbf{v}})\tilde{\omega}$, $c = (\mathbf{e}_1 + \mathbf{e}_3)\|\tilde{\omega}\|^2 - e_c\mathbf{e}_2\|\tilde{\mathbf{v}}\|^2$, and $e_c = e_\omega r/e_v$. Notice that the ball motions have to satisfy the proposition given by (Liu et al., 2012) (Proposition 1, Section III-A) to guarantee the existence of a solution.

5 OPTIMAL PADDLE TRAJECTORY PLANNING

In this section, the problem of generating an optimal trajectory for the end-effector of the paddle is tackled. The control design procedure from the previous section provides a desired pose and velocity of the paddle and corresponding desired time instant at which to achieve these. There are many different paths that the paddle can take to fulfill these requirements. A more judicious method would be to plan this path such that

a certain objective function is optimized. Differential geometry offers a way to extend the notion of differentiation from Euclidean space to an arbitrary manifold. Primarily, a short background about differential geometry is here reported, consequently, the theory about the minimum acceleration planner for the paddle is described.

5.1 Differential Geometry Background

In this context, trajectories for which it is possible to specify the initial and final position and velocity of the motion are of interest. Note that, the motion may be specified in either the joint space, which is a torus, or the task space, which is the special Euclidean group of three-dimensions over the reals. At this stage, it is assumed that the path is generated on an arbitrary Riemannian manifold M (do Carmo, 1992). Let $\gamma : (a,b) \to M$ be the path, and $\langle\langle \cdot, \cdot \rangle\rangle$ be the metric on M. Let $f : (-\varepsilon,\varepsilon) \times (a,b) \to M$ be a proper variation of γ; that is, it satisfies

$$f(0,t) = \gamma(t), \quad \forall t \in (a,b)$$
$$f(s,a) = \gamma(a), \quad \text{and} \quad f(s,b) = \gamma(b)$$

Two vector fields are relevant along the path γ. The first one is called the *variation field* and is defined by

$$S_{\gamma(s)} := \frac{\partial f(s,t)}{\partial s} = \frac{\mathrm{d} f_t(s)}{\mathrm{d} s}$$

The second vector field of importance is the velocity vector field of γ, given by

$$V_{\gamma(s)} := \frac{\mathrm{d}\gamma(t)}{\mathrm{d} t} = \frac{\partial f(s,t)}{\partial t} = \frac{\mathrm{d} f_s(t)}{\mathrm{d} t}$$

In order to perform calculus on the curves of this Riemannian manifold, the Levi-Civita connection ∇ is introduced. Given a curve $\gamma(t)$ and a connection, there exists a covariant derivative, which is denoted by $\frac{\mathrm{D}}{\mathrm{d} t}$.

The Levi-Civita connection satisfies the following compatibility and symmetry conditions, presented in this order:

$$\frac{\mathrm{d}}{\mathrm{d} t}\langle\langle U,W \rangle\rangle = \left\langle\left\langle \frac{\mathrm{D} U}{\mathrm{d} t},W \right\rangle\right\rangle + \left\langle\left\langle U,\frac{\mathrm{D} W}{\mathrm{d} t} \right\rangle\right\rangle \quad (11a)$$

$$\nabla_X Y - \nabla_Y X = [X,Y] \quad (11b)$$

for any vector fields U and W, along the differentiable curve γ, and any vector field $X,Y \in \mathfrak{X}(M)$.

The curvature R of a Riemannian manifold M is a correspondence that associates to every pair $X,Y \in \mathfrak{X}(M)$ a mapping $R(X,Y) : \mathfrak{X}(M) \to \mathfrak{X}(M)$ given by

$$R(X,Y)Z = \nabla_Y \nabla_X Z - \nabla_X \nabla_Y Z + \nabla_{[X,Y]} Z,$$

where $Z \in \mathfrak{X}(M)$. Some of the properties of the curvature that are going to be used are

$$\frac{\mathrm{D}}{\partial t}\frac{\mathrm{D}}{\partial s}X - \frac{\mathrm{D}}{\partial s}\frac{\mathrm{D}}{\partial t}X = R\left(\frac{\partial f}{\partial s}, \frac{\partial f}{\partial t}\right)X,$$

$$\langle R(X,Y)Z, T \rangle = \langle R(Z,Y)X,Y \rangle.$$

5.2 Minimum Acceleration Planning

In this work, the acceleration functional in $SE(3)$ is minimized by following theory developed in (Zefran et al., 1998). The acceleration functional to be minimized by the choice of the path to be followed by the paddle is

$$J = \int_{t_a}^{t_b} \langle \nabla_V \mathbf{V}, \nabla_V \mathbf{V} \rangle \, \mathrm{d} t, \quad (12)$$

where $[t_a,t_b]$ is the time interval over which the trajectory is planned, $\mathbf{V} = (\omega_P, \mathbf{v}_P) \in \mathfrak{se}(3)$ is the velocity of the paddle along a particular path, and ∇ denotes the Levi-Civita affine connection derived from a particular choice of metric on $SE(3)$. This latter object allows one to perform differentiation along curves on any smooth manifold. In particular, in (12), it is used to express the inner product of the acceleration of a particular path with itself, which may also be identified by the squared norm of the acceleration of this path at a particular point. If the metric on $SE(3)$ is taken in the form

$$\mathbf{W} = \begin{bmatrix} \alpha\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \beta\mathbf{I}_3 \end{bmatrix}$$

where $\alpha, \beta > 0$ so that for $\mathbf{T}_1, \mathbf{T}_2 \in \mathfrak{se}(3)$, then $\langle\langle\mathbf{T}_1,\mathbf{T}_2\rangle\rangle = \mathbf{t}_1^\top \mathbf{W}\mathbf{t}_2$ with $\mathbf{t}_1$ and $\mathbf{t}_2$ the 6×1 components of $\mathbf{T}_1$ and $\mathbf{T}_2$, then the Levi-Civita connection can be computed to be

$$\nabla_X Y = \left\{ \frac{\mathrm{d}}{\mathrm{d} t}\omega_y + \frac{1}{2}\omega_x \times \omega_y, \frac{\mathrm{d} v_y}{\mathrm{d} t} + \omega_x \times v_y \right\}$$

where ω_x and ω_y are the angular components and v_x and v_y are the linear components of the rigid body velocities $X \in \mathfrak{se}(3)$ and $Y \in \mathfrak{se}(3)$, respectively.

Equating the first variation of the cost functional (12) to zero provides necessary conditions for the path to be a minimizer of the acceleration functional. This necessary condition is a fourth order boundary problem given by

$$\nabla_V \nabla_V \nabla_V \mathbf{V} + R(\mathbf{V}, \nabla_V \mathbf{V})\mathbf{V} = \mathbf{0}, \quad (13)$$

where R is the curvature tensor associated with the Levi-Civita affine connection (do Carmo, 1992). This condition can be written down in terms of the angular and linear velocity components of the paddle as

$$\omega_P^{(3)} + \omega_P \times \ddot{\omega}_P = \mathbf{0}, \quad (14a)$$

$$\mathbf{p}_P^{(4)} = \mathbf{0}, \quad (14b)$$

where $(\cdot)^{(n)}$ denotes the nth derivative of $(\cdot)$. These ordinary differential equations turn into a well-defined boundary value problem with the addition of the

boundary conditions. For the differential equations (14a) governing the optimal rotational path, the boundary conditions are

$$\mathbf{R}_P(t_a) = \mathbf{R}_P^0 \quad \omega_P(t_a) = \omega_P^0, \tag{15a}$$

$$\mathbf{R}_P(t_b) = \mathbf{R}_P^i, \quad \omega_P(t_b) = \omega_P^i, \tag{15b}$$

where $\mathbf{R}_P^0$ and ω_P^0 are the initial orientation and angular velocity of the paddle, respectively. Similarly for the differential equations (14b) governing the optimal translational path of the paddle, the boundary conditions are

$$\mathbf{p}_P(t_a) = \mathbf{p}_P^0 \quad \mathbf{v}_P(t_a) = \mathbf{v}_P^0, \tag{16a}$$

$$\mathbf{p}_P(t_b) = \mathbf{p}_P^i \quad \mathbf{v}_P(t_b) = \mathbf{v}_P^i, \tag{16b}$$

In practice, the optimal translational motion of the paddle is found by merely solving a small scale linear system of equations obtained by (14b) and (16). As a result, it may be performed very fast. On the other hand, in order to determine the rotary motion of the paddle, a boundary value problem needs to be solved. The boundary value problem is time invariant and non linear, but the forcing function (14a) is hardly complicated. In Section 6, the computational burden of the proposed optimal paddle trajectory planner is deeply analysed.

6 SIMULATIONS

This section shows the results obtained in simulation. Section 6.1 shows an exemplar simulation of the proposed algorithm (see Figure 2). The comparative case studies in Section 6.2 show instead the results of the comparison between the proposed solution to compute the ball post-impact velocity (see Section 4.1.2) and the one used in (Liu et al., 2012). Lastly, in Section 6.3 the optimal planner simulations are focused on the paddle motion to highlight the properties of the planned trajectories.

The model parameters considered to simulate the physical system (1) and (4) are listed in Table 1. A procedure to identify the aerodynamic and rebound parameters is described by (Nonomura et al., 2010). The simulations described below are implemented in the Matlab environment: the *ode45* solver, with the *events* option, is used for the dynamic model; the *lsqcurvefit* function, based on the *Levembert-Marquardt*'s algorithm, is employed to find the aerodynamic forward and backward solutions; the *bvp4c* function is used for the minimum acceleration planner.

Table 1: Dynamic parameters.

r	Ball radius	2e-2 m
r_p	Paddle radius	1.5e-1 m
m	Ball mass	2.7e-3 kg
ρ	Air density ($25°C$)	1.184 kg/m^3
g	Gravity constant	9.81 m/s^2
e_v	Velocity rebound coefficient	6.15e-1
e_ω	Spin rebound coefficient	2.57e3
e_r	Linear rebound coefficient	7.3e-1
a_d	Drag coefficient	5.05e-1
b_d	Drag coefficient	6.5e-2
a_l	Lift coefficient	9.4e-2
b_l	Lift coefficient	-2.6e-2
c_d	Simplified drag coefficient	5.4e-1

6.1 Batting Task Simulation

In this case study the proposed algorithm, depicted in Figure 2, is simulated. Then, supposing to have at disposition the estimated trajectory of the ball from the visual system and the desired final configuration of the ball, it is possible to compute the optimal paddle trajectory to achieve the batting task through the two minimization problems (7) and (8), and the solution of (9) and (14).

Therefore, the visual measurement system is assumed to provide the initial position, linear and angular velocity produced by the opponent's hit, respectively correspondent to $\mathbf{p}_B^0 = \begin{bmatrix} 1.2 & 0.7 & 0.9 \end{bmatrix}$ m, $\mathbf{v}_B^0 = \begin{bmatrix} -3 & 0.2 & 1.5 \end{bmatrix}$ m/s and $\omega_B^- = \begin{bmatrix} 0, 150, 0 \end{bmatrix}$. The impact time is fixed to $t_i = 0.5$ s and the desired final position of the ball on the opposite court is assigned as $\mathbf{p}_B^d = \begin{bmatrix} 1.9 & 0.8 & 0.02 \end{bmatrix}$ m. The desired flight time and post-impact ball spin are assumed to be respectively $t_d = 0.6$ s and $\begin{bmatrix} \omega_{By}^+, \omega_{Bz}^+ \end{bmatrix} = \begin{bmatrix} -100, 0 \end{bmatrix}$ rad/s. The third component of the desired post-impact spin of the ball, ω_B^{+d}, is computed from the equation $\tilde{\mathbf{v}}_B^T \tilde{\omega}_B = 0$. It is remarkable that, since the goal is achieved when the ball hits the table on the opposite court, the final goal time is evaluated when the third component of the position vector of the ball is equal to the radius of the ball.

In this case study, the aerodynamic forward solution of (7) is $\mathbf{p}_B^i = \begin{bmatrix} -0.1394 & 0.7892 & 0.4820 \end{bmatrix}^T$ m and $\mathbf{v}_B^- = \begin{bmatrix} -2.4156 & 0.1570 & -2.9788 \end{bmatrix}^T$ m/s, the aerodynamic backward solution of (8) is $\mathbf{v}_B^+ = \begin{bmatrix} 4.0516 & 0.0214 & 2.0984 \end{bmatrix}$ m/s, while the solution of reset map (9) is

$$\mathbf{R}_p^i = \begin{bmatrix} 0.8614 & 0.0054 & 0.5080 \\ 0 & 0.9999 & -0.0106 \\ -0.5080 & 0.0092 & 0.8613 \end{bmatrix}$$

and $\mathbf{v}_P^i = \begin{bmatrix} 1.4388 & 0.0220 & -0.1131 \end{bmatrix}^T$ m/s. The

Table 2: Ball pre-impact configuration.

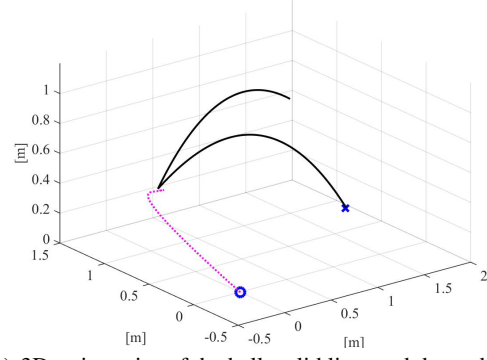|	$\mathbf{v}_B^-$ $[m/s]$	ω_B^- $[rad/s]$
1	$-2.5, 0, 0.1$	$0, 150, 0$
2	$-4.5, 0, 0.3$	$0, -150, 0$
3	$-2.75, -0.8, -0.5$	$0, -50, 150$

optimal trajectory is consequently planned solving (14).

The time histories given by the simulation of this case study are depicted in Figure 3. In particular, the 3D trajectories of both the ball and the paddle are represented in Figure 3(a). The solid line represents the motion of the ball, while the trajectory for the paddle is depicted by a dashed line. The blue cross represents the final desired position of the ball $\mathbf{p}_B^d$, while the blue circle is the initial position of the paddle $\mathbf{p}_P^0$. Assuming that Δt_i is the difference between the planned and the actual impact time, $\Delta \mathbf{p}_B^i$ the Euclidean norm of the difference between the planned impact position of the ball and the actual one, Δt_d the difference between the desired final time and the actual one, $\Delta \mathbf{p}_B^d$ the Euclidean norm of the difference between the desired position of the ball $\mathbf{p}_B^d$ and the actual one. In this case study, the time and position impact and final errors are respectively $\Delta t_i = 4.5e-3$ s, $\Delta \mathbf{p}_B^i = 1.73e-2$ m, $\Delta t_d = 1.9e-3$ s and $\Delta \mathbf{p}_B^d = 9.4e-3$ m. Therefore, the average of the errors is acceptably small for a table tennis robot.
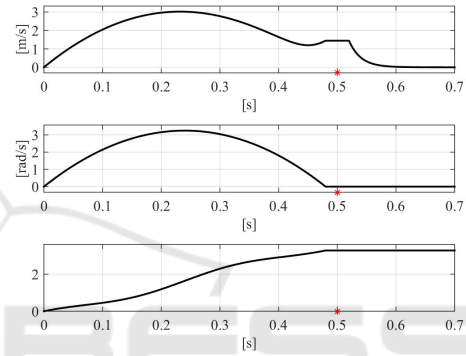
A video can be found in (Serra et al., 2016) which shows the simulation of this case study performed in Matlab, while visualization employs the V-Rep environment. A 21 degree of freedom humanoid robot is used for the simulation. The 7 degree of freedom right arm is equipped with a parallel jaw gripper, which firmly grasps the paddle.

6.2 Comparative Case Studies

The purpose of this subsection is to compare the proposed planning method with the state-of-the-art approach introduced by (Liu et al., 2012), which presents several case studies, with ample implementation details, allowing a fair and critical comparison. Three case studies are considered, respectively: backspin, topspin and sidespin. In order to obtain a fair comparison with the work (Liu et al., 2012), the first step of the proposed algorithm is not taken into account, i.e. the ball pre-impact state is assumed to be known a-priori and then the minimization problem (7) is not solved. As a matter of fact, in each of these cases, the impact position of both the ball and the paddle is assigned to be $\mathbf{p}_P^i = \mathbf{p}_B^i = \begin{bmatrix} -0.15 & 0.70 & 0.25 \end{bmatrix}$ m, while the linear and angular velocity of the ball before the impact for the back-

(a) 3D trajectories of the ball, solid line, and the paddle, dashed line, obtained with the proposed method. The blue circle represents the initial position of the paddle, while the blue cross is the desired final position of the ball.

(b) From the top to the bottom: magnitude of the planned linear and angular velocity of the paddle, evaluation of the acceleration functional J in (12) between the motion plan devised using the Euler angles and the optimal proposed one. The red star represents the impact time t_i.

Figure 3: Batting Task Simulation.

spin (1), topspin (2) and sidespin (3) case studies are shown in Table 2. On the other hand, the impact time is $t_i = 0.1$ s, the desired goal position for the ball is $\mathbf{p}_B^d = \begin{bmatrix} 2.055 & 0.7680 & 0.02 \end{bmatrix}$ m, while the desired final time is $t_d = 0.6$ s. Moreover, the desired spin of the ball after the impact $\begin{bmatrix} \omega_{By}^+, \omega_{Bz}^+ \end{bmatrix}$ is set for the first, second and third case studies to $\begin{bmatrix} -100, 0 \end{bmatrix}$ rad/s, $\begin{bmatrix} 100, 0 \end{bmatrix}$ rad/s and $\begin{bmatrix} 0, -100 \end{bmatrix}$ rad/s respectively, according to the setting of the work described in (Liu et al., 2012).

The results obtained for each of the three case studies are depicted by their time histories in Figures 4, 5 and 6, respectively. In particular, the 3D trajectories of both the ball and the paddle are represented in Figures 4(a), 5(a) and 6(a), respectively. The solid line represents the ball, while the paddle is depicted by a dashed line. The time histories of each component p_{Bx}, p_{By} and p_{Bz} related to the trajectory of the ball are represented in Figures 4(b), 5(b)

Table 3: Comparative case study 1 - Numerical results.

	Δt_i [s]	$\Delta \mathbf{p}_B^i$ [m]	Δt_d [s]	$\Delta \mathbf{p}_B^d$ [m]	t_c [ms]
A	5.2e-3	1.3e-2	1.12e-2	5.47e-2	9e-3
B	5.14e-3	1.29e-2	4.88e-3	6.47e-4	21

Table 4: Comparative case study 2 - Numerical results.

	Δt_i [s]	$\Delta \mathbf{p}_B^i$ [m]	Δt_d [s]	$\Delta \mathbf{p}_B^d$ [m]	t_c [ms]
A	3.86e-3	1.74e-2	4.26e-2	1.3e-1	8e-3
B	4.01e-3	1.81e-2	7.34e-4	3.29e-2	21

Table 5: Comparative case study 3 - Numerical results.

	Δt_i [s]	$\Delta \mathbf{p}_B^i$ [m]	Δt_d [s]	$\Delta \mathbf{p}_B^d$ [m]	t_c [ms]
A	4.88e-3	1.42e-2	2.12e-2	1.03e-1	1e-2
B	4.86e-3	1.41e-2	9.02e-4	1.94e-2	25

and 6(b), respectively, for both of the compared methods: the solid line graphs the solution obtained by exploiting the method described by (Liu et al., 2012), while the dashed curve illustrates the trajectory of the ball obtained using the proposed method. The figures indicate that the proposed method yields an improvement over (Liu et al., 2012) by directing the ball closer to the goal configuration.

The quantitative results for the backspin, topspin and sidespin case studies are shown in Tables 3, 4 and 5, where A represents the results obtained employing the Liu's planner, whereas B the results obtained employing our planner. The first two columns of each of the aforementioned tables refer to the precision of the impact. In both the approaches the results are very similar. The gist of the comparison may be captured by analyzing the last three columns of these tables. The final time error is reduced by about an order of magnitude. The Euclidean norm of the error between the actual final position of the ball and the desired one is smaller by about an order of magnitude, too.
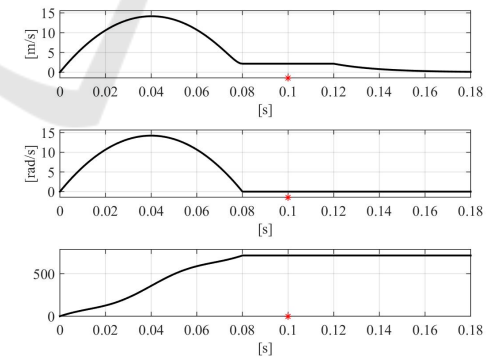
Unfortunately, experiments are not yet available for the proposed approach since the practical set-up is under development. However, the switching from Matlab into C++ language is already accomplished. The computational time t_c required to solve at runtime both the simplified aerodynamic model and the complete one is show in the last column of Tables 3, 4 and 5. The code is running on a computer with specifications Intel Core 2 Quad CPU Q6600 @ 2.4 GHz, Ubuntu 12.04 32-bit operating system, including the Levenberg-Marquardt C++ library (Lourakis, 2004). The same numerical results obtained with Matlab have been retrieved, but the evaluation of the computation burden is more precise, in the sense that it is the one that will appear during the practical experiments. To elaborate, compared to (Liu et al., 2012), the presented method increases the accuracy of the fi-

(a) 3D trajectories of the ball, solid line, and the paddle, dashed line, obtained with the proposed method. The blue circle represents the initial position of the paddle, while the blue cross is the desired final position of the ball.
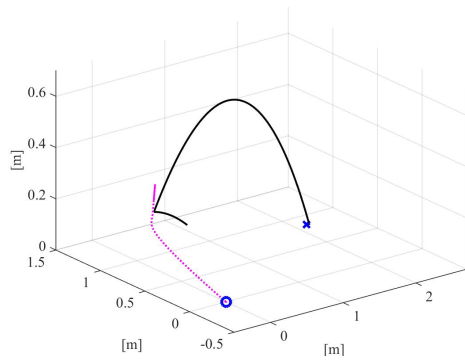

(b) Comparison between ball trajectories: the solution given by analytically solving (6) is depicted through a solid line, the proposed one is instead represented with a dashed line. The blue cross represents the desired final position of the ball.
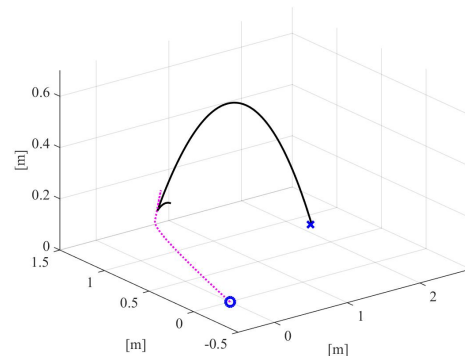

(c) From the top to the bottom: magnitude of the planned linear and angular velocity of the paddle, evaluation of the acceleration functional J in (12) between the motion plan devised using the Euler angles and the optimal proposed one. The red star represents the impact time t_i.
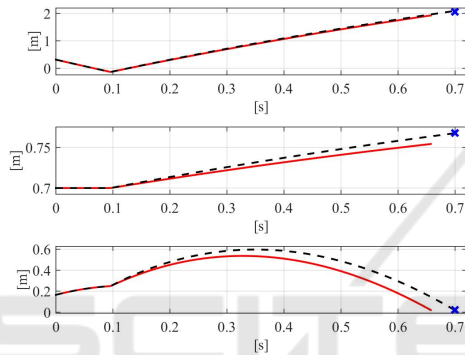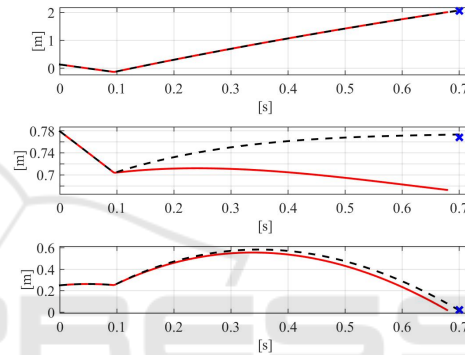
Figure 4: Comparative case study 1.

nal desired position an order of magnitude for topspin and sidespin cases and two orders of magnitude for the backspin case. Furthermore, the C++ implemen-
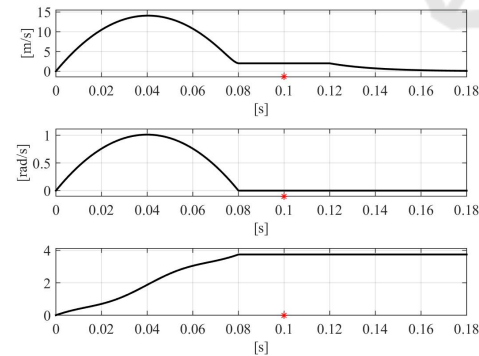
(a) 3D trajectories of the ball, solid line, and the paddle, dashed line, obtained with the proposed method. The blue circle represents the initial position of the paddle, while the blue cross is the desired final position of the ball.
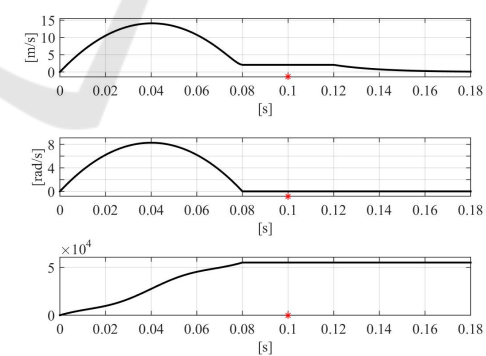

(b) Comparison between ball trajectories: the solution given by analytically solving (6) is depicted through a solid line, the proposed one is instead represented with a dashed line. The blue cross represents the desired final position of the ball.

(c) From the top to the bottom: magnitude of the planned linear and angular velocity of the paddle, evaluation of the acceleration functional J in (12) between the motion plan devised using the Euler angles and the optimal proposed one. The red star represents the impact time t_i.

Figure 5: Comparative case study 2.

(a) 3D trajectories of the ball, solid line, and the paddle, dashed line, obtained with the proposed method. The blue circle represents the initial position of the paddle, while the blue cross is the desired final position of the ball.

(b) Comparison between ball trajectories: the solution given by analytically solving (6) is depicted through a solid line, the proposed one is instead represented with a dashed line. The blue cross represents the desired final position of the ball.

(c) From the top to the bottom: magnitude of the planned linear and angular velocity of the paddle, evaluation of the acceleration functional J in (12) between the motion plan devised using the Euler angles and the optimal proposed one. The red star represents the impact time t_i.

Figure 6: Comparative case study 3.

tation of the proposed nonlinear minimization problem, which considers the full aerodynamic model of the ball, takes about 20 ms to give the desired veloc-

ity of the ball after the impact. This duration is grater than what is shown in (Liu et al., 2012), but it is still acceptable for real-time implementation.

6.3 Optimal Planner Simulations

For each case study, the paddle trajectory is planned over the time interval $[t_a, t_b] = [t_0, t_i - \varepsilon]$, where $\varepsilon = 0.02$ s. The paddle trajectory is supposed to start still, from the origin of the world frame, with initial orientation $\mathbf{R}_P^0 = \mathbf{R}_Y(\pi/2)\mathbf{R}_X(0)$, without loss of generality. According to the proposed algorithm, the position, orientation and linear velocity of the paddle at the impact time are given by the ball reset map solution. The Euclidean norm of the linear and angular velocities of the paddle, planned with the minimum total acceleration, are represented by the top and middle plots in Figures 4(c), 5(c), 6(c) and 3(b), for each simulation.

Once the desired orientation is achieved with zero angular velocity, the angular acceleration is set to zero so that the orientation of the paddle remains the same until the impact occurs. As long as one has the control authority at the torque level, this control strategy, which switches only once, is straightforward to implement. Once the impact has occurred at time t_i, notice that the linear velocity of the paddle is exponentially dissipated by the following term $\exp(-\mu(t - (t_i + \delta)))$, where t is the time, $\mu = 50$ and $\delta = 0.02$, so that the paddle stops.

The optimal trajectory discovered by solving the two-point boundary value problem (14) indeed minimizes the L_2 norm of the total acceleration of the paddle. So as to illustrate this fact, another typical trajectory for the orientation of the paddle is planned. This alternative plan constructs a third-order polynomial function for the Euler angles, ϕ and θ, such that the initial and final orientation and angular velocity constraints are satisfied. Both the angular acceleration that corresponds to this motion plan and the acceleration functional J in (12) are then computed. For each case study, the bottom time histories of Figures 3(b), 4(c), 5(c) and 6(c), depict the value of the difference of the acceleration functional between the motion plan devised using the Euler angles and the optimal motion plan. Notice that the value of this cost functional is positive at $t = t_i$, indicating that the optimal motion plan indeed yields a smaller value of the acceleration functional than a typical plan performed using polynomials on the Euler angles, ϕ and θ.

About the computational burden of the proposed planner, the code has been translated in C++ and evaluated on the same PC as in Section 6.2. The boundary value problem for the optimal paddle trajectory planner takes less than 30ms. To sum up, after the high-speed vision system gives a stable trajectory estimation of the ball coming towards our court, it is possible to compute the desired trajectory for the paddle in 50ms (20ms + 30ms), hitting the ball with a proper velocity to redirect it to the opposite court at the desired position with the imposed spin. Another possibility is that, once the desired impact position, velocity and orientation is determined, one can immediately start controlling the paddle to achieve these via a PD controller, and revert to a trajectory following controller once the optimal trajectory is available and is periodically updated.

7 CONCLUSIONS AND FUTURE WORK

The presented paper proposes an algorithm to plan a robotic batting task. In particular, a table tennis game performed by a robot has been considered. The proposed solution improves the control accuracy while dealing with the real-time constraint. A coordinate-free, smooth, optimal motion plan, that minimizes the acceleration functional of the paddle, is proposed. As future work, the proposed technique is planned to be validated through experimental studies. Different optimal planners which make use of the dynamic model of the robotic manipulator grasping the paddle may also be considered.

ACKNOWLEDGEMENTS

REFERENCES

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

Andersson, R. (1989a). Aggressive trajectory generator for a robot ping-pong player. *IEEE Control Systems Magazine*, 9(2):15–21.

Andersson, R. (1989b). Dynamic sensing in a ping-pong playing robot. *IEEE Transactions on Robotics and Automation*, 5(6):728–739.

Cigliano, P., Lippiello, V., Ruggiero, F., and Siciliano, B. (2015). Robotic ball catching with an eye-in-hand single-camera system. *IEEE Transactions on Control Systems Technology*, 23(5):1657–1671.

do Carmo, M. (1992). *Riemannian Geometry*. Mathematics (Boston, Mass.). Birkhäuser.

Huang, Y., Xu, D., Tan, M., and Su, H. (2013). Adding active learning to LWR for ping-pong playing robot. *IEEE Transactions on Control Systems Technology*, 21(4):1489–1494.

Kuka (2014). The Duel: Timo Boll vs. KUKA Robot. [web page] https://youtu.be/tIIJME8-au8+.

Lippiello, V. and Ruggiero, F. (2012). 3D monocular robotic ball catching with an iterative trajectory estimation refinement. In *IEEE International Conference on Robotics and Automation*, pages 3950–3955, Saint Paul, MN, USA.

Liu, C., Hayakawa, Y., and Nakashima, A. (2012). Racket control and its experiments for robot playing table tennis. In *IEEE International Conference on Robotics and Biomimetics*, pages 241–246, Guangzhou, CN.

Lourakis, M. (Jul. 2004). levmar: Levenberg-Marquardt nonlinear least squares algorithms in C/C++. [web page] http://www.ics.forth.gr/ lourakis/levmar/+. [Accessed on 31 Jan. 2005.].

Lynch, K. and Mason, M. (1999). Dynamic nonprehensile manipulation: Controllability, planning, and experiments. *The International Journal of Robotics Research*, 18(64):64–92.

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

Müller, M., Lupashin, S., and D'Andrea, R. (2011). Quadrocopter ball juggling. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5113–5120, San Francisco, CA, USA.

Mülling, K., Kober, J., Kroemer, O., and Peters, J. (2013). Learning to select and generalize striking movements in robot table tennis. *The International Journal of Robotics Research*, 32(3):263–279.

Mülling, K., Kober, J., and Peters, J. (2010). A biomimetic approach to robot table tennis. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1921–1926, Taipei, TW.

Nakashima, A., Ito, D., and Hayakawa, Y. (2014). An online trajectory planning of struck ball with spin by table tennis robot. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 865–870, Besançon, F.

Nakashima, A., Ogawa, Y., Kobayashi, Y., and Hayakawa, Y. (2010). Modeling of rebound phenomenon of a rigid ball with friction and elastic effects. In *IEEE American Control Conference*, pages 1410–1415, Baltimore, MD, USA.

Nonomura, J., Nakashima, A., and Hayakawa, Y. (2010). Analysis of effects of rebounds and aerodynamics for trajectory of table tennis ball. In *IEEE Society of Instrument and Control Engineers Conference*, pages 1567–1572, Taipei, TW.

Omron (2015). CEATEC 2015: Omron's Ping Pong Robot. [web page] https://youtu.be/6MRxwPHH0Fc+.

Oubbati, F., Richter, M., and Schoner, G. (2013). Autonomous robot hitting task using dynamical system approach. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 4042–4047, Manchester, UK.

Senoo, T., Namiki, A., and Ishikawa, M. (2006). Ball control in high-speed batting motion using hybrid trajectory generator. In *IEEE International Conference on Robotics and Automation*, pages 1762–1767, Orlando, FL, USA.

Serra, D., Satici, A. C., Ruggiero, F., Lippiello, V., and Siciliano, B. (2016). An optimal trajectory planner for a robotic batting task: The table tennis example. [web page] https://youtu.be/GXtBvbUHu5s+.

Silva, R., Melo, F., and Veloso, M. (2015). Towards table tennis with a quadrotor autonomous learning robot and onboard vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 649–655, Hamburg, D.

Wei, D., Guo-Ying, G., Ye, D., Xiangyang, Z., and Han, D. (2015). Ball juggling with an under-actuated flying robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 68–73, Hamburg, D.

Yanlong, H., Bernhard, S., and Jan, P. (2015). Learning optimal striking points for a ping-pong playing robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4587–4592, Hamburg, D.

Zefran, M., Kumar, V., and Croke, C. (1998). On the generation of smooth three-dimensional rigid body motions. *IEEE Transactions on Robotics and Automation*, 14(4):576–589.

Zhang, Y., Xiong, R., Zhao, Y., and Chu, J. (2012). An adaptive trajectory prediction method for ping-pong robots. In *Intelligent Robotics and Applications*, pages 448–459. Springer.