Time–Optimal Paths for a Robotic Batting Task

Diana Serra*, Fabio Ruggiero*, Aykut C. Satici+, Vincenzo Lippiello*, and Bruno Siciliano*

* Department of Electrical Engineering and Information Technology, University of Naples Federico II, Via Claudio 21, 80125, Naples, Italy
+ Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 32 Vassar Street, Cambridge, Massachusetts, USA

Abstract. This paper presents a novel method to optimize the motion of a paddle within a nonprehensile batting task. The proposed approach shows that it is possible to online predict the impact time and the configuration of the paddle, in terms of its linear velocity and orientation, to re-direct a ball towards a desired location, imposing also a desired spin during the free flight. While exploiting the hybrid dynamics of the task during the minimization process, the obtained position and orientation paths are planned by minimizing the acceleration function of the paddle in SE(3). The batting paths are then tracked by a semi-humanoid robot through a closed-loop kinematic inversion. Numerical tests are implemented to compare different metrics to define the optimal impact time.

Keywords: Optimal Trajectory Planning, Robotic Batting Task, Dynamic Nonprehensile Manipulation

1 Introduction

Research in the field of dynamic nonprehensile manipulation is essential to achieve robots with human-like task execution capabilities. Humans can indeed perform several nonprehensile manipulation actions everyday, both with and without hands, such as pushing objects, folding clothes, serving a dish on a tray, flipping a pancake in a pan, or rolling in a wok. A general taxonomy about manipulation is presented in [1], where four main categories emerge: kinematic, static, quasi-static and dynamic manipulation. Kinematic manipulation is often related to slow tasks: motion of the object can be retrieved by movements of the robotic hand and its kinematics. Static manipulation, instead, can be studied using both kinematics and static forces, while quasi-static manipulation makes use of quasi-static forces (such as frictional forces) in the formulation. Within dynamic manipulation, instead, a relevant role is played by forces and accelerations which are used together with kinematics, static and quasi-static forces, to achieve a general description of a manipulation task.

Dynamic nonprehensile manipulation offers some potential advantages, such as [2]: reduction of the task execution time; extension of the workspace of the robot; increased dexterity of the robotic system; cheap and simple design of grippers; minimal deformation of the manipulated objects, and so on. On the other hand, the complexity of such nonprehensile tasks leads to adopt a "*divide et impera*" strategy for motion planning and control. This means to split the complex task in simpler *primitives* of motion,

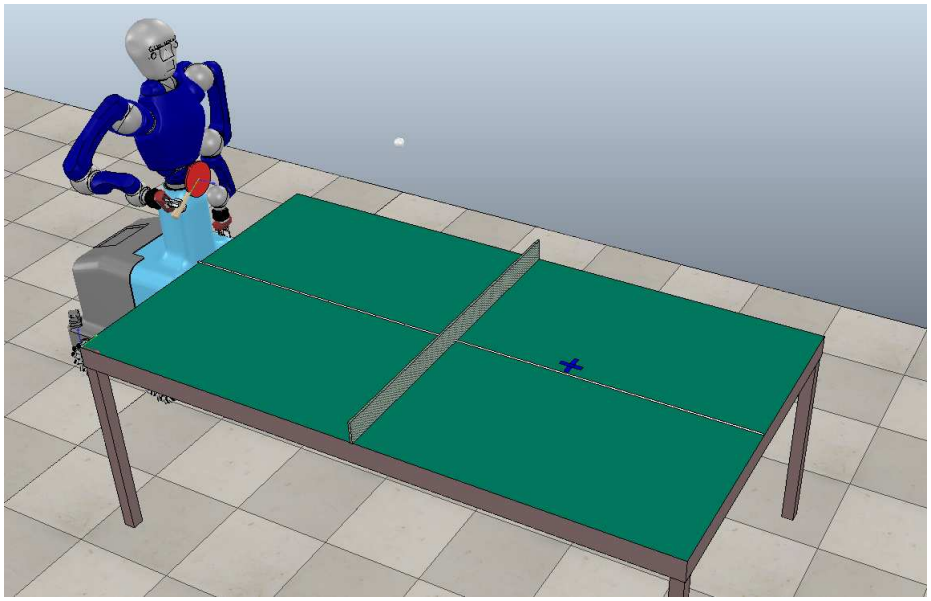


Fig. 1. Semi-humanoid playing table tennis in the V-Rep simulation environment.

and design a motion planner for each of them separately. A supervisory controller is then assumed to identify the primitives that compose the task and switch between them. Batting, throwing, catching, rolling, sliding are typical examples of nonprehensile manipulation primitives [3].

1.1 Overview and Outline of the Paper

In this paper, the focus is on the design of an optimal motion planner for one of the above mentioned nonprehensile manipulation primitives, the *batting* one. The batting primitive is an agile nonprehensile manipulation task that is typically used by athletes in sports like baseball, cricket, or table tennis. It definitely represents an engaging challenge for existing robotic software and hardware resources, because of its inherent unpredictability, velocity, and complexity. A novel approach to derive the optimal path for a semi-humanoid robot to bat a ball with a paddle and direct it to a desired location is here proposed. As displayed in Figure 1, the considered application is the table tennis game. The same application is considered in [4], where an algorithm to select position and velocity of the paddle is designed so as to intercept the upcoming ball. The minimum acceleration path for the paddle accomplishing the batting task is planned considering a predefined impact time. The technique presented in [4] improves the control accuracy in comparison to the state of the art by considering a full aerodynamic model of the ball, and taking into account drag and lift forces; additionally, it requires a computation time comparable to real-time. In detail, the approach consists of two main phases. In the former phase, position, orientation and linear velocity of the paddle

are computed at the predefined impact time, such that the goal of driving back the ball towards the other paddle is satisfied. Then, in the latter phase, an optimization of the path in SE(3) gives the angular and linear trajectory of the paddle up to the predefined impact time. In this work, instead, an extension of the algorithm proposed in [4] is presented. The assumption of having a constant predefined impact time is relaxed, while different metrics are compared to define the optimal impact time. Numerical tests are implemented to evaluate the algorithm.

The paper is organized as follows. Section 2 presents the state of the art about planning methods for the batting motion, typically applied to the robotic table tennis game. Section 3 introduces the hybrid dynamics of the system. Section 4 presents the workflow of the proposed batting algorithm, and describes the method employed for predicting the state of the ball at the impact time, and the following computation of the desired configuration of the paddle. The details about the minimum acceleration planner in SE(3) for the paddle are introduced in Section 5. Section 6 shows the numerical evaluation of the different versions of the algorithm, and its application to a semi-humanoid robot equipped with a paddle as an end-effector. Finally, Section 7 concludes the paper.

2 State of the Art about the Robotic Batting Primitive

One of the first real-time table tennis robot prototype is proposed by Andersson in [5]. It is built on a commercial PUMA 260 robot arm, which is a 5 degrees of freedom (DoF) industrial robot. In [6], the same author employs fifth-order polynomials to generate a trajectory for the paddle intercepting the ball. The trajectory of the arm of the robot is adjusted while the ball is in free flight through a sensor-driven approach. A low-cost ping-pong player prototype is proposed in [7]. The authors propose to detect the location of the ball combining the information about the ball and its shadow on the table. An expert module defines the desired return point for the ball. A high-speed trajectory planner is presented in [8], where the authors propose to split the motion in two phases: a high speed phase and a reactive one, named swing and hitting motion, respectively. The hitting point is estimated before the impact, while the batting task is accomplished by modifying this point through a visual feedback. An approach to keep stability of a biped humanoid robot while playing table tennis is presented in [9]. In that work, an optimal momentum compensation method using lower body joints to cancel the momentum generated by arms is discussed. The authors of [10] propose an algorithm for returning a table tennis ball to a desired position with a desired spin. An approximated hybrid aerodynamics of the ball is exploited to compute the configuration of the paddle at impact time to accomplish the batting task.

In the field of artificial intelligence, learning techniques exploiting data-driven perspectives instead of inverse kinematics and physical models are often applied to the robotic table tennis game, through offline training of the system. The study shown in [11] predicts the ball trajectory using a fuzzy adaptive resonance theory network, and self-learns the behavior for each strike using a reinforcement learning network imitating human learning behavior. The work presented in [12] shows instead some experiments on two humanoid robots playing ping pong. The approach employs an adjustment of the trajectory prediction from an offline training of the model parameters based on a neural

network. In [13], the robot first learns a set of elementary table tennis hitting movements from a human teacher, and then it generalizes those movements in a wider range of situations using a mixture of motor primitives approach. The authors of [14] present an active learning approach, where the initial parameters of the paddle are computed through a locally weighted regression method. In [15], the authors 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. The work in [16] presents an approach for robotic table tennis game consisting in two stages: a first regression phase, in which the joint trajectories are generated to strike the incoming ball, and a second reinforcement learning phase, where the joint trajectories are updated to properly return the ball. The authors of [17] present a probabilistic approach to intercept a table tennis ball in space. A probabilistic representation is employed to find the initial time and the duration of the movement primitive maximizing the likelihood of hitting the ball.

Moreover, aerial robotics researchers have also been interested in planning the motion of an aerial vehicle to implement the batting motion primitive. An algorithm to generate an open loop trajectory guiding a prototype quadcopter to a predicted impact point is proposed in [18] by exploiting a Kalman filter. The authors of [19] implement the hitting primitive on a commercial drone. Finally, a trajectory tracking control strategy for a ball juggling task on a quadrotor, based on the subspace stabilization approach, is presented in [20]. An optimal trajectory generation method is adopted to obtain a dynamically feasible minimum jerk trajectory.

3 Hybrid Dynamic Equations of the System

The main methodology employed in this work consists in an optimization based planning of the robot trajectories using the hybrid dynamic model of the ball together with the motion of the paddle. The analytic model here considered for the hybrid ball and paddle dynamics is derived in [4]. The hybrid dynamics of the ball is made up of the free fall and the rebound equations. Since the mass of the ball is usually smaller than the mass of the paddle, the velocity of the paddle is assumed to be not affected by the collision: therefore, the rebound equations are only related to velocity of the ball. Additionally, a point contact is assumed between the ball and the paddle at impact time. Other assumptions related to the hybrid model of the ball motion are detailed in [21].

Let Σ_W be the world frame, Σ_B be the frame placed at the center of the ball, and Σ_P be the frame placed at the center of the paddle, with the z-axis as the outward normal. Let $\mathbf{p}_B = \begin{bmatrix} p_{Bx} & p_{By} & p_{Bz} \end{bmatrix}^T \in \mathbb{R}^3$, $\mathbf{v}_B = \begin{bmatrix} v_{Bx} & v_{By} & v_{Bz} \end{bmatrix}^T \in \mathbb{R}^3$ and $\omega_B = \begin{bmatrix} \omega_{Bx} & \omega_{By} & \omega_{Bz} \end{bmatrix}^T \in \mathbb{R}^3$ be the position, linear velocity and spin of the ball, respectively, with respect to Σ_W. The spin of the ball is assumed to be constant before and after the rebound. Moreover, let $\mathbf{p}_P = \begin{bmatrix} p_{Px} & p_{Py} & p_{Pz} \end{bmatrix}^T \in \mathbb{R}^3$, $\mathbf{v}_P = \begin{bmatrix} v_{Px} & v_{Py} & v_{Pz} \end{bmatrix}^T \in \mathbb{R}^3$ and $\omega_P = \begin{bmatrix} \omega_{Px} & \omega_{Py} & \omega_{Pz} \end{bmatrix}^T \in \mathbb{R}^3$ be the position, linear and angular velocity of the paddle, respectively, 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. Assuming

Table 1. List of symbols

r	Radius of the ball
r_p	Radius of the paddle
m	Mass of the ball
ρ	Density of the air at $(25°C)$
g	Gravity constant
ε_v	Velocity rebound coefficient
ε_ω	Spin rebound coefficient
ε_r	Linear rebound coefficient
κ_1^d	Drag coefficient
κ_2^d	Drag coefficient
κ_1^l	Lift coefficient
κ_2^l	Lift coefficient
ζ^d	Simplified drag coefficient

that the drag and lift coefficients k_d and k_l, respectively, are represented by

$$k^d = \tfrac{1}{2m}\rho\pi r^2(\kappa_1^d + \kappa_2^d\mu), \quad k^l = \tfrac{1}{m}\rho 4\pi r^3(\kappa_1^l + \kappa_2^l\mu), \tag{1}$$

where

$$\mu = \left(1 + \frac{(v_{Bx}^2 + v_{Bx}^2)\omega_{Bz}^2}{(v_{Bx}\omega_{By} - v_{By}\omega_{Bx})^2}\right)^{-\frac{1}{2}}, \tag{2}$$

while the other symbols are listed in Table 1, the continuous ball and paddle dynamics are given by

$$\dot{\mathbf{p}}_B = \mathbf{v}_B, \tag{3a}$$
$$\dot{\mathbf{v}}_B = -\mathbf{g} - k^d||\mathbf{v}_B||\mathbf{v}_B + k^l\mathbf{S}(\omega_B)\mathbf{v}_B, \tag{3b}$$
$$\dot{\mathbf{p}}_P = \mathbf{v}_P, \tag{3c}$$
$$\dot{\mathbf{R}}_P = \mathbf{R}_P\mathbf{S}(\omega_P), \tag{3d}$$

where $\mathbf{g} = \begin{bmatrix}0 & 0 & g\end{bmatrix}^T$ is the gravity acceleration, $||\cdot||$ is the Euclidean norm and $\mathbf{S}(\cdot) \in \mathbb{R}^{3\times 3}$ is the skew-symmetric matrix operator.

A method to identify the aerodynamic and rebound coefficients for different kind of rebound materials is presented in [21]. Within this framework, the aerodynamics of the ball (3b) models the magnitude of the drag and lift forces, and the respective parameters depend on the spin of the ball. This last is thus included in the formulation since it is relevant in the table tennis game, distinguishing an expert player form an amateur one. In particular, the type of spin is identified by the sign of the components of ω_B, i.e. *topspin*, if $\omega_{By} < 0$, *backspin*, if $\omega_{By} > 0$, and *sidespin*, if $\omega_{Bz} > 0$.

Assuming that the superscripts $-$ and $+$ represent the state before and after the impact time, respectively, the linear and angular velocity of the ball after the impact can

be retrieved from

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

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

where the matrices of rebound parameters are defined as

$$\begin{aligned} \mathbf{A}_{vv} &= \mathrm{diag}(1-\varepsilon_v, 1-\varepsilon_v, \varepsilon_r), \mathbf{A}_{v\omega} = -\varepsilon_v r \mathbf{S}(\mathbf{e}_3), \\ \mathbf{A}_{\omega v} &= \varepsilon_\omega r \mathbf{S}(\mathbf{e}_3), \mathbf{A}_{\omega\omega} = \mathrm{diag}(1-\varepsilon_\omega r^2, 1-\varepsilon_\omega r^2, 1), \end{aligned} \tag{5}$$

and ε_v and ε_w are detailed in Table 1, whereas $\mathbf{e}_i \in \mathbb{R}^3$ is the unit vector along the i^{th}-axis, $i = \{1,2,3\}$.

Notice that the hybrid system (3)-(4) is characterized by the fact that its input, corresponding to the paddle linear and angular velocities, enters the ball dynamics exclusively through the rebound equations (4) since, as stated above, the paddle can modify the ball velocity only at impact time.

4 Time-Optimal Prediction

4.1 Workflow of the Algorithm

In order to generate the optimal motion for the paddle to bat a table tennis ball towards a desired position with a desired spin, the paddle has to intercept the ball with a specific orientation and velocity. The algorithm to realize such batting motion primitive can be roughly divided in three main phases: the prediction of the motion of the ball, the selection of the configuration of the paddle at impact time, and the trajectory planning for the paddle. A graphical representation of the phases generating the optimal path for the paddle is showed in Figure 2.

1. In the first stage, the impact time t^i, the impacting position $\mathbf{p}_B^i = \left[p_{Bx}^i \; p_{By}^i \; p_{Bz}^i\right]^T \in \mathbb{R}^3$ and the pre-impact velocity $\mathbf{v}_B^-$ of the ball are predicted assuming to know the initial position of the ball $\mathbf{p}_B^0 = \left[p_{Bx}^0 \; p_{By}^0 \; p_{Bz}^0\right]^T \in \mathbb{R}^3$ and its linear and angular velocity, respectively, $\mathbf{v}_B^0 = \left[v_{Bx}^0 \; v_{By}^0 \; v_{Bz}^0\right]^T \in \mathbb{R}^3$ and $\omega_B^0 = \left[\omega_{Bx}^0 \; \omega_{By}^0 \; \omega_{Bz}^0\right]^T \in \mathbb{R}^3$. These are produced by the opponent's hit and obtained from the visual measurement system. This step is accomplished by solving forward the model (3a)-(3b). Notice that the main novelty of the paper is introduced in this first stage: the impact time is not a priori defined, as in [4], but it is predicted online. Afterwards, choosing the goal configuration of the ball given by $\mathbf{p}_B^d = \left[p_{Bx}^d \; p_{By}^d \; p_{Bz}^d\right]^T \in \mathbb{R}^3$, $\omega_B^{+d} \in \mathbb{R}^3$, and the desired final time t^d, the post-impact velocity of the ball $\mathbf{v}_B^+ \in \mathbb{R}^3$ is obtained through the backward solution of (3a)-(3b).
2. In the second stage, once the spin and velocity of the ball before and after the impact are computed from the previous stage, the algorithm selects the orientation $\mathbf{R}_P^i$ and the velocity $\mathbf{v}_P^i$ of the paddle at impact time through the solution of the reset map (4).

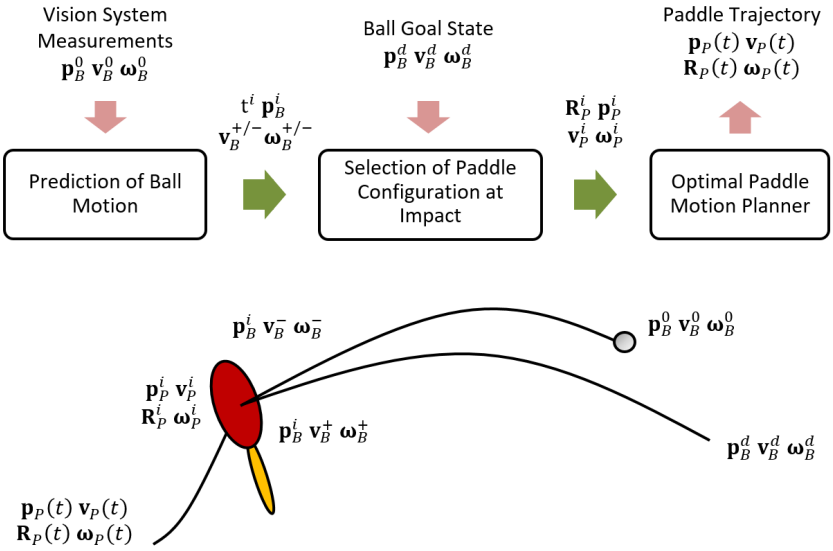


Fig. 2. Graphic representation of the stages of the algorithm.

3. In the third stage, the optimal trajectory planner for the paddle receives as input the desired configuration of the paddle at impact time. A boundary value problem is solved to compute the linear and angular trajectories of the paddle without specifying any representation of the angular coordinates. At this point, the linear and angular path of the paddle, namely $\mathbf{p}_P(t), \mathbf{v}_P(t), \mathbf{R}_P(t), \omega_P(t)$, can be tracked by the end effector of the semi-humanoid robot with a classical second order closed loop kinematic inversion [22].

In the following, the accurate description of the first two stages will be carried out. Section 5 is instead entirely devoted to present the third stage.

4.2 Stage 1: Prediction of the Impacting Time, Position and Velocities of the Ball

In order to predict the impact time and position of the ball, and its linear pre- and post-impact velocity such that it reaches the desired location after the batting action, the equations (3a) and (3b) of the aerodynamic model are employed. However, this model is nonlinear and coupled, thus an analytic solution does not exist. Other approaches such as in [7] and in [18] employ linearized or simplified models with the aim to cut down the elaboration time. For example, the following simplified model of (3a)-(3b) is

employed in [10] since it directly provides an analytic solution

$$\dot{v}_{Bx} = -\zeta^d |v_{Bx}| v_{Bx}, \qquad \dot{p}_{Bx} = v_{Bx}, \tag{6a}$$

$$\dot{v}_{By} = -\zeta^d |v_{By}| v_{By}, \qquad \dot{p}_{By} = v_{By}, \tag{6b}$$

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

where $\zeta^d = \frac{\rho}{2m}\pi r^2 k^d$, with k^d considered as a suitable constant coefficient. The approach here proposed goes instead in a different direction: it exploits a proper numerical solver suitable for real-time processing.

Three different optimization problems are considered to predict the impact time and position of the ball, as well as its velocity before the rebound. The general methodology consists in the solution of nonlinear curve fitting problems.

Prediction with a predefined impact time A first approach to predict the impact position and the pre-impact velocity of the ball is showed in [4]. By assigning the impact time, the impact configuration of the ball is predicted according to its initial position and velocity produced by the opponent's hit. The following minimization problem is then considered

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

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 (3a) and (3b) starting from the optimization variables $\mathbf{p}_B^i, \mathbf{v}_B^-$ at the predefined impact time t^i. In this case, the determination of the impact position and pre-impact velocity of the ball is addressed at run-time, differently from other approaches in the literature, such as [10, 23].

Prediction of a variable impact time In a second possible optimization problem the impact time is predicted as well as the impact position and the pre-impact velocity of the ball. Starting from the initial state of the ball, the following minimization problem is equivalent to (7) but it includes the impact time as a decision variable

$$\min_{t^i, \mathbf{p}_B^i, \mathbf{v}_B^-} \left|\left| \begin{bmatrix} \widetilde{\mathbf{p}}_B^0 \\ \widetilde{\mathbf{v}}_B^0 \end{bmatrix} - \begin{bmatrix} \mathbf{p}_B^0 \\ \mathbf{v}_B^0 \end{bmatrix} \right|\right|^2. \tag{8}$$

In this case, the resulting impact time is optimized in the sense that it best fits the numerical curve.

Prediction of a variable impact time minimizing the motion of the paddle The third possible approach consists in the computation of the impact time minimizing the distance between the initial position of the paddle and its impact position. The following minimization problem is thus considered

$$\min_{t^i, \mathbf{p}_B^i, \mathbf{v}_B^-} ||\widetilde{\mathbf{p}}_B^i - \mathbf{p}_P^0||^2, \tag{9}$$

where $\widetilde{\mathbf{p}}_B^i = \widetilde{\mathbf{p}}_B^i(\mathbf{p}_B^0, \mathbf{v}_B^0)$ is the position of the ball at impact time numerically obtained by forward integrating (3a) and (3b) starting from the optimization variables $\mathbf{p}_B^0, \mathbf{v}_B^0$ at time t^0. In this case, the resulting impact time is optimized in the sense that the paddle travels the shortest distance to intercept the ball. Therefore, the introduction of this metric in the prediction stage allows not only to predict the position of the ball and its pre-impact velocity but even, more interestingly, to optimize the impact time with respect to the length of the path that the paddle should cover.

Another optimization is considered in order to compute the post-impact velocity of the ball $\mathbf{v}_B^+$ such that it reaches the goal $\mathbf{p}_B^d$ at the desired time t^d. The following curve fitting is implemented

$$\min_{\mathbf{v}_B^+} \|\widetilde{\mathbf{p}}_B^d(\mathbf{v}_B^+) - \mathbf{p}_B^d\|^2, \tag{10}$$

where $\mathbf{v}_B^+$ is the decision 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 (3a) and (3b) starting from the ball position $\mathbf{p}_B^i$ at impact time t^i, computed through one of the metrics given above. This solution ensures that the post-impact motion of the ball is such that it reaches the desired location at the time t^d as close as possible. An initial guess for the solution of the minimization problem (10) is analytically calculated from (6) as

$$v_{Bx}^+ = \frac{(p_{Bx}^d - p_{Bx}^i)(e^{\zeta^d |p_{Bx}^d - p_{Bx}^i|} - 1)}{\zeta^d |p_{Bx}^d - p_{Bx}^i| (t^d - t^i)}, \tag{11a}$$

$$v_{By}^+ = \frac{(p_{By}^d - p_{By}^i)(e^{\zeta^d |p_{By}^d - p_{By}^i|} - 1)}{\zeta^d |p_{By}^d - p_{By}^i| (t^d - t^i)}, \tag{11b}$$

$$v_{Bz}^+ = -\frac{g(t^d - t^i)}{2} + \frac{p_{Bz}^d - p_{Bz}^i}{t^d - t^i}. \tag{11c}$$

4.3 Stage 2: Desired Configuration of the Paddle at Impact

Once the impact time t^i, the impact position $\mathbf{p}_B^i$ of the ball and its pre- and post-impact velocities, $\mathbf{v}_B^-$ and $\mathbf{v}_B^+$ respectively, are computed as in Section 4.2, the paddle configuration is derived solving the rebound model of the ball.

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 impact time are respectively computed through

$$\mathbf{v}_P^i = \mathbf{v}_B^- + \mathbf{R}_P^i (\mathbf{I}_3 - \mathbf{A}_{vv})^{-1} (\mathbf{R}_P^{iT} \tilde{\mathbf{v}} - \mathbf{A}_{v\omega} \omega_B^-), \tag{12a}$$

$$\mathbf{R}_P^i = \mathbf{R}_Y(\theta) \mathbf{R}_X(\phi), \tag{12b}$$

where $\mathbf{I}_3 \in \mathbb{R}^{3\times 3}$ 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

$$\tilde{v}_z \cos\phi \sin\theta - \tilde{v}_x \cos\phi \cos\theta = \tilde{\omega}_y, \tag{13a}$$

$$e_c^2 ||\tilde{\mathbf{v}}||^2 \sin^2\phi - 2e_c \mathbf{e}_2 S(\tilde{\mathbf{v}})\tilde{\omega}\sin\phi + (\mathbf{e}_1 + \mathbf{e}_3)||\tilde{\omega}||^2 - e_c \mathbf{e}_2 ||\tilde{\mathbf{v}}||^2 = 0, \tag{13b}$$

where $e_c = \varepsilon_\omega r/\varepsilon_v$, and the other symbols are listed in Table 1. In order to obtain a well-posed solution of the reset map, the ball motions must comply with the Proposition 1 in Section III-A of [10].

5 Minimum Acceleration Path in SE(3)

The third stage of the workflow of the proposed algorithm is described in this section. In detail, the generation of an optimal trajectory in SE(3) for the paddle is addressed. By knowing the initial configuration of the paddle, and after having computed its desired configuration at the impact (see Section 4.3), many different paths and trajectories can be followed to fulfill the requirements. A wiser method would be to found this path such that a certain objective function is optimized. As in [4], the optimization of the acceleration functional of the paddle along the path in SE(3) is pursued. In the following, a brief background about differential geometry is provided. Afterwards, the theory about the minimum acceleration planner for the paddle is reported.

5.1 Brief Background about Differential Geometry

Within this context, trajectories for which it is possible to specify both the initial and final position and velocity are of interest. The motion can be specified in either the joint space, which is a torus, or the task space, which is SE(3). At this stage, to be general, it is assumed that the path is generated on an arbitrary Riemannian manifold $\mathcal{M}$ (see [24] for more details).

Let $\gamma : (a,b) \to \mathcal{M}$ be the path, and $\langle\!\langle \cdot, \cdot \rangle\!\rangle$ be the metric on $\mathcal{M}$. Let $f : (-\varepsilon, \varepsilon) \times (a,b) \to \mathcal{M}$ be a proper variation of γ satisfying

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

Two vector fields are relevant along the path γ. The former is the *variation field* which is defined by

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

The latter is the velocity vector field of γ, given by

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

Hence, the Levi-Civita connection ∇ is introduced to perform calculus on the curves of $\mathcal{M}$. Therefore, given a curve $\gamma(t)$ and a connection, there exists a covariant derivative denoted by $\frac{\mathrm{D}}{\mathrm{d}t}$.

The Levi-Civita connection satisfies the following compatibility and symmetry conditions:

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle\!\langle U,W\rangle\!\rangle = \langle\!\langle \frac{\mathrm{D}U}{\mathrm{d}t},W\rangle\!\rangle + \langle\!\langle U,\frac{\mathrm{D}W}{\mathrm{d}t}\rangle\!\rangle \tag{14a}$$

$$\nabla_X Y - \nabla_Y X = [X,Y] \tag{14b}$$

for any vector fields U and W, along the differentiable curve γ, and any vector field $X,Y \in \mathfrak{X}(\mathcal{M})$, where $\mathfrak{X}(\mathcal{M})$ is the set of all vector fields on $\mathcal{M}$.

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

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

where $Z \in \mathfrak{X}(\mathcal{M})$. In the next subsection, the following properties related to the curvature are employed

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

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

5.2 Optimized Path Planning in SE(3)

Following the theory developed in [25], the following acceleration function along the path of the paddle in SE(3) is minimized within the proposed framework

$$J = \int_{t_a}^{t_b} \langle \nabla_V \mathbf{V}, \nabla_V \mathbf{V}\rangle \,\mathrm{d}t, \tag{15}$$

where $[t_a,t_b]$ is the time interval over which the trajectory is planned, $\mathbf{V} = (\boldsymbol{\omega}_P, \mathbf{v}_P) \in \mathfrak{se}(3)$ 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 the differentiation along curves on any smooth manifold. In particular, the inner product of the acceleration of a particular path with itself is expressed in (15): this may also be identified by the squared norm of the acceleration of this path at a particular point. Choosing the metric on SE(3) as

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

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

$$\nabla_X Y = \left\{ \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\omega}_y + \frac{1}{2}\boldsymbol{\omega}_x \times \boldsymbol{\omega}_y, \frac{\mathrm{d}v_y}{\mathrm{d}t} + \boldsymbol{\omega}_x \times v_y \right\},$$

where $\boldsymbol{\omega}_x$ and $\boldsymbol{\omega}_y$ 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.

In order to have necessary conditions to find a path minimizing the acceleration function, the first variation of such a cost (15) has to be equated to zero. This yields a fourth order boundary problem given by

$$\nabla_V \nabla_V \nabla_V \mathbf{V} + R(\mathbf{V}, \nabla_V \mathbf{V})\,\mathbf{V} = \mathbf{0}, \tag{16}$$

where R is the curvature tensor associated with the Levi-Civita affine connection [24]. Notice that it is possible to write down (16) in terms of the angular and linear velocity components of the paddle as

$$\boldsymbol{\omega}_P^{(3)} + \boldsymbol{\omega}_P \times \ddot{\boldsymbol{\omega}}_P = \mathbf{0}, \tag{17a}$$

$$\mathbf{p}_P^{(4)} = \mathbf{0}, \tag{17b}$$

where $(\cdot)^{(n)}$ denotes the n^{th} derivative of $(\cdot)$. The obtained ordinary differential equations (17) turn into a well-defined boundary value problem with the addition of the boundary conditions. Regarding the rotational path (17a), such boundary conditions are

$$\mathbf{R}_P(t_a) = \mathbf{R}_P^0 \quad \boldsymbol{\omega}_P(t_a) = \boldsymbol{\omega}_P^0, \tag{18a}$$

$$\mathbf{R}_P(t_b) = \mathbf{R}_P^i, \quad \boldsymbol{\omega}_P(t_b) = \boldsymbol{\omega}_P^i, \tag{18b}$$

where $\mathbf{R}_P^0$ and $\boldsymbol{\omega}_P^0$ are the initial orientation and angular velocity of the paddle, respectively. On the other hand, regarding the translational path (17b), the boundary conditions are

$$\mathbf{p}_P(t_a) = \mathbf{p}_P^0 \quad \mathbf{v}_P(t_a) = \mathbf{v}_P^0, \tag{19a}$$

$$\mathbf{p}_P(t_b) = \mathbf{p}_P^i \quad \mathbf{v}_P(t_b) = \mathbf{v}_P^i, \tag{19b}$$

Notice that in the practice, the optimal translational motion of the paddle is found by merely solving a small scale linear system of equations obtained by (17b) and (19): this may be performed very fast from an elaboration time point of view. Nevertheless, in order to determine the rotary motion of the paddle, a boundary value problem needs to be solved. In general, a boundary value problem is nonlinear and time invariant, but the forcing function (17a) is hardly complicated. The computational burden of the proposed planner for an optimal trajectory of the paddle is analyzed within Section 6.3.

6 Simulations

This section presents a discussion about the numerical evaluation of the batting algorithm. The three metrics introduced in Section 4.2 are taken into account. In detail, Section 6.1 shows an exemplar simulation of the proposed technique with a predefined impact time. Section 6.2 describes instead the evaluation of the approach with the prediction of a variable impact time using the last two metrics (8)-(9) in Section 4.2. Finally, Section 6.3 presents some tests focused on the paddle motion to underline the properties of the planned trajectories.

The values of the parameters of the dynamic model considered to simulate the physical system (3) and (4) are listed in Table 2 [4]. The Matlab environment is used for numerical tests. For the hybrid dynamics, the *ode*45 solver, with the *events* option, is employed. The *lsqcurvefit* function, which is based on the Levembert-Marquardt's algorithm, is adopted to solve the nonlinear curve fittings. The boundary value problem, for the minimum acceleration planner, is solved through the *bvp*4*c* function. In this work, the Levembert-Marquardt's algorithm is employed to solve the non-linear least squares problems (7), (8), and (9) since it guarantees small elaboration time, as explained in [26] and [27]. In order to speed up the convergence of the optimization problems, the initial guess for the solution is obtained solving analytically the approximated model (11).

Table 2. Numerical values of the considered parameters of the hybrid dynamic model

r	2e-2 m
r_p	1.5e-1 m
m	2.7e-3 kg
ρ	1.184 kg/m^3
g	9.81 m/s^2

κ_1^d	5.05e-1
κ_2^d	6.5e-2
κ_1^l	9.4e-2
κ_2^l	-2.6e-2

ε_v	6.15e-1
ε_ω	2.57e3
ε_r	7.3e-1
ζ^d	5.4e-1

6.1 Evaluation of the Batting Motion Planner with a Predefined Impact Time

In this case study the proposed algorithm, depicted in Figure 2, is simulated considering a constant predefined impact time. Then, supposing to have at disposition the estimated trajectory of the ball from the visual system and the desired final configuration of the ball, the optimal paddle trajectory is derived through the two minimization problems (7) and (10), and the solution of (12) and (17).

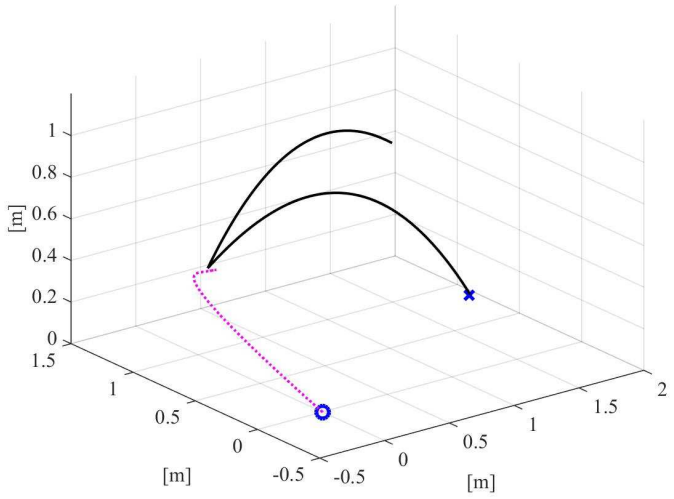
In particular, the initial state of the ball is assumed to be equal to $\mathbf{p}_B^0 = \begin{bmatrix}1.2,\ 0.7,\ 0.9\end{bmatrix}$ m, $\mathbf{v}_B^0 = \begin{bmatrix}-3,\ 0.2,\ 1.5\end{bmatrix}$ m/s and $\omega_B^- = \begin{bmatrix}0, 150, 0\end{bmatrix}$. The impact time is fixed to $t^i = 0.5$ s and the goal position of the ball after the rebound is assigned as $\mathbf{p}_B^d = \begin{bmatrix}1.9,\ 0.8,\ 0.02\end{bmatrix}$ m. The desired final time corresponds to $t^d = 0.6$ s, and the post-impact spin of the ball to $\begin{bmatrix}\omega_{By}^+, \omega_{Bz}^+\end{bmatrix} = \begin{bmatrix}-100, 0\end{bmatrix} rad/s$. The term ω_B^{+d} derives from the expression $\tilde{\mathbf{v}}_B^T \tilde{\omega}_B = 0$. Notice that the actual final time is evaluated when the third component of the position vector of the ball corresponds to the radius of the ball.

With these hypothesis, solving (7) yields $\mathbf{p}_B^i = \begin{bmatrix}-0.1394,\ 0.7892,\ 0.4820\end{bmatrix}$ m and $\mathbf{v}_B^- = \begin{bmatrix}-2.4156,\ 0.1570,\ -2.9788\end{bmatrix}$ m/s, while the solution of the minimization (10) yields $\mathbf{v}_B^+ = \begin{bmatrix}4.0516,\ 0.0214,\ 2.0984\end{bmatrix}$ m/s. Furthermore, the solution of the rebound model (12) yields

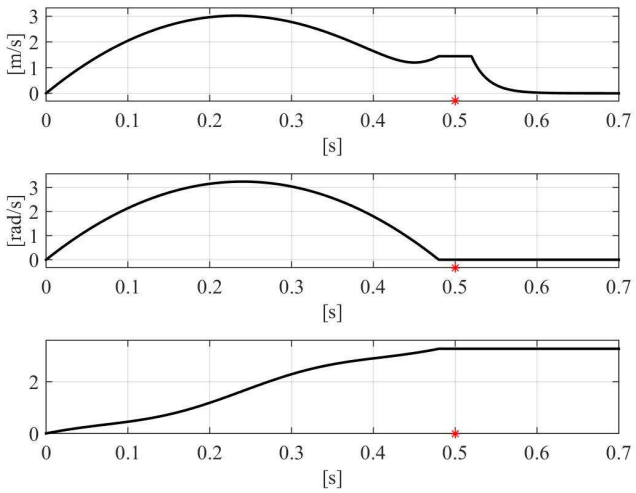
$$\mathbf{R}_p^i = \begin{bmatrix} 0.8614 & 0.0054 & 0.5080 \\ 0 & 0.9999 & -0.0106 \\ -0.5080 & 0.0092 & 0.8613 \end{bmatrix},$$

and $\mathbf{v}_P^i = \begin{bmatrix}1.4388,\ 0.0220,\ -0.1131\end{bmatrix}$ m/s. Afterwards, the minimum acceleration trajectory for the paddle is planned employing (17).

Let Δt_i be the error between the predefined impact time and the one obtained during the simulation through the *ode*45 solver together with the *events* option. Let $\Delta\mathbf{p}_B^i$ be the Euclidean norm of the error between the planned and actual impact position of the ball, Δt_d be the error between the predefined and actual final time t^d, $\Delta\mathbf{p}_B^d$ be the Euclidean norm of the error between the goal position of the ball and the actual one. The first row of Table 3 shows these errors in case of a predefined impact time. Whereas, the resulting plots 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 $\mathbf{p}_P^0$. A more detailed evaluation of the proposed batting algorithm with fixed impact time in case of sidespin, backspin, and topspin can be found in [4].



(a) 3D trajectories of the ball and the paddle (respectively solid and dashed line) obtained from the predefined impact time method. The blue circle represents the initial position of the paddle, while the blue cross identifies the goal position of the ball.



(b) Euclidean norm of the linear (top) and angular (center) velocity paths planned for the paddle, and evaluation of the acceleration functional J in (15) between the motion plan devised using the Euler angles and the optimal proposed one (bottom). The red star represents the impact time t^i.

Fig. 3. Simulation of the batting task with predefined impact time.

Table 3. Comparison of the fixed and optimized impact time prediction methods.

	Δt_i	$\Delta \mathbf{p}_B^i$	Δt_d	$\Delta \mathbf{p}_B^d$
Fixed Impact Time Prediction	4.5e-3 s	1.73e-2 m	1.9e-3 s	9.4e-3 m
Optimal Impact Time Prediction	4e-3 s	1.69e-2 m	1.1e-3 s	1.33e-2 m

Moreover, a video which shows this simulation performed in Matlab in connection with the V-Rep virtual simulation environment can be found in [28]. A 21 DoF semi-humanoid robot is used within this simulation. The 7 DoF right arm is equipped with a parallel jaw gripper firmly grasping the paddle (see Figure 1). The rebound model of the ball with the table is given by [29], and a second order closed loop kinematic inversion is implemented to map the planned Cartesian variables to the joints space of the robot [22].

6.2 Evaluation of the Optimal Impact Time Prediction

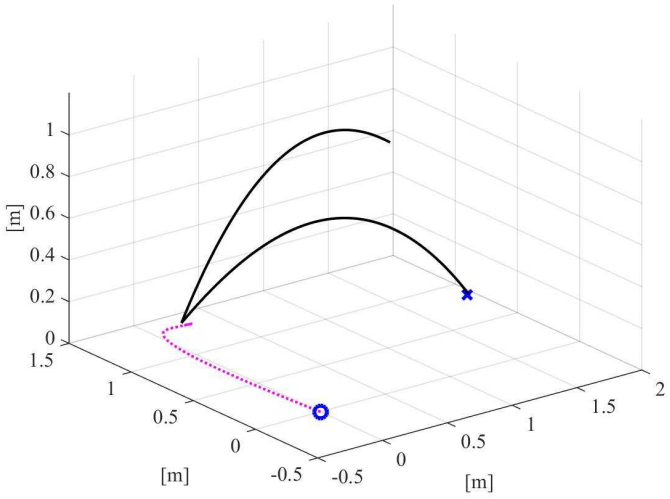
In this case study, the same initial configuration of the ball used in Section 6.1 is considered, and the proposed algorithm is evaluated including the online prediction of the impact time. The results obtained with the two prediction metrics presented in Section 4.2 are here reported.

A first test is done considering the minimization problem (8). In this case, the solution of the prediction phase is t^i=0.5051 s, $\mathbf{p}_B^i = \left[-0.1516,\, 0.79,\, 0.4668\right]$ m, and $\mathbf{v}_B^- = \left[-2.4105,\, 0.1565,\, -3.0174\right]$ m/s. The obtained result is not substantially different from the one got from the simulation considering a pre-defined impact time. For this reason, figures for this case are not included.

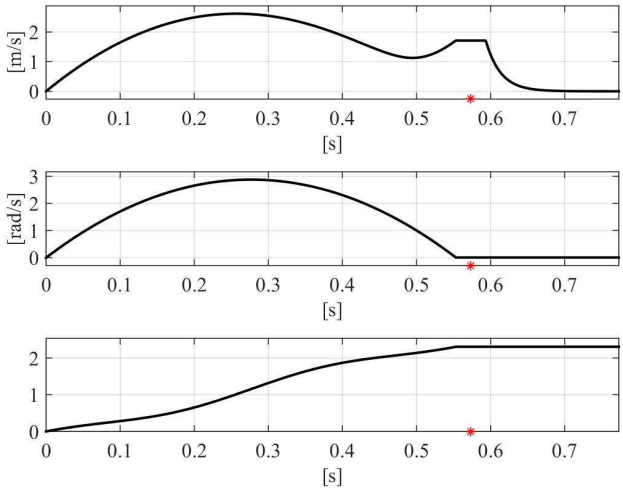
Whereas, another test is done solving (9), (10), (12), and (17). In this case, the solution of (9) is $\mathbf{p}_B^i = \left[-0.3141,\, 0.8,\, 0.243\right]$ m, $\mathbf{v}_B^- = \left[-2.34,\, 0.1498,\, -3.521\right]$ m/s, and $t^i = 0.5735$ s. The predicted impact time guarantees that the paddle follows the minimum length path to intercept the ball. The plots resulting from this test are displayed in Figure 4. The 3D paths of the ball and the paddle are reported in Figure 4(a). The solid and dashed lines represent the motion of ball and paddle, respectively. The blue cross identifies the goal position of the ball, whereas the blue circle is the initial position of the paddle. The second row of Table 3 shows the values of Δt_i, $\Delta \mathbf{p}_B^i$, Δt_d, and $\Delta \mathbf{p}_B^d$ for this numerical test. Now, Δt_i represents the error between the impact time resulting from the minimization problem and the one obtained during the simulation. The results point out that, even if the position error is slightly increased with respect to the constant impact time prediction case, now the impact time and position are planned online, and the ball hits the table at a time closer to the desired one.

6.3 Discussion on the Minimum Acceleration Planner

In this section a discussion on the evaluation of the minimum acceleration planner presented in Section 5 is reported. Without loss of generality, the paddle is supposed to



(a) 3D trajectories of the ball and the paddle (respectively solid and dashed line) obtained considering the optimal impact time method. The initial position of the paddle is represented by the blue circle, while the goal position of the ball is the blue cross.



(b) Euclidean norm of the linear (top) and angular (center) velocity planned for the paddle, and evaluation of the acceleration functional J in (15) between the motion plan devised using the Euler angles and the optimal one (bottom). The red star represents the impact time t^i.

Fig. 4. Simulation of the batting task with optimal impact time.

start at the origin of the world frame with initial orientation $\mathbf{R}_P^0 = \mathbf{R}_Y(\pi/2)\mathbf{R}_X(0)$ and zero velocity. The path is planned for the paddle in the time interval $[t_a, t_b] = [t^0, t^i - \varepsilon]$, with $\varepsilon = 0.02$ s. Following the batting algorithm, the solution of the rebound model presented in Section 4.3 produces the configuration of the paddle at impact time. The Euclidean norm of the linear and angular velocities of the paddle, planned with the minimum total acceleration for the two above mentioned case studies, are respectively represented by the top and middle plots in Figure 3(b) and 4(b).

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. After the collision with the ball, in order to stop the paddle its linear velocity is exponentially decreased with the function $exp(-\mu(t-(t^i+\delta)))$, where $\mu = 50$ and $\delta = 0.02$.

The L_2 norm of the acceleration of the paddle is found through the two-point boundary value problem (17). Moreover, for comparison with these minimum acceleration paths, third-order polynomials for Euler angles, ϕ and θ, are considered. Initial and final orientation and angular velocity constraints are correspondingly imposed. The difference of the acceleration functional between the motion plan devised using the Euler angles and the optimal motion planner is displayed in the bottom plots of Figure 3(b) and 4(b). This difference is positive at impact time, implying that the optimal motion plan undoubtedly generates a smaller acceleration than typical polynomial paths on Euler angles.

Regarding computational efficiency, an analysis is done coding the algorithm in C++ and evaluating it 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 [30]. The analysis reveals that the boundary value problem takes less than 30ms. On account of this, as long as the vision system provides the ball's trajectory estimation, about 50 ms are necessary to calculate the appropriate trajectory for the paddle. If this time constraint is still excessively tight, after that the desired impact time, position, velocity and orientation of the paddle are imposed, the paddle can be rapidly controlled through a classical proportional-derivative controller, and revert to a trajectory following controller at each time the planner generates the optimal path.

7 Conclusions and Future Work

The presented paper proposes a novel algorithm to plan a robotic batting task. In this work, the table tennis is considered as application of the batting primitive. Through the solutions of a two stages curve fitting problem, the prediction of the impact time, and of the impact state of the ball is implemented. Then, the configuration of the paddle such that the goal of driving back the ball towards a desired location is generated. Finally, an optimization of the path in SE(3) gives minimum acceleration angular and linear trajectory of the paddle up to the impact time. Simulations in the Matlab/V-Rep environment validate the approach.

Experiments on the real robotic prototype will follow in a short future. The approach could also be integrated with advanced image processing techniques, with the aim of

increasing the potentiality of the visual measurement system. Possible extensions of this work could be related to the application of the proposed framework to other non-prehensile manipulation tasks, i.e. ball juggling or intermittent pushing. An evolution of the algorithm could be conceived in combination with machine learning techniques.

Acknowledgements

The research leading to these results has been supported by the RoDyMan project, which has received funding from the European Research Council FP7 Ideas under Advanced Grant agreement number 320992.

Bibliography

[1] M. T. Mason and K. M. Lynch. Dynamic manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 152–159, Tokyo, J, 1993.

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

[3] D. Serra. Robot control for nonprehensile dynamic manipulation tasks. In *International Conference on Informatics in Control, Automation and Robotics, Doctoral Consortium*, pages 3–12, Lisbon, PT, 2016.

[4] D. Serra, A.C. Satici, F. Ruggiero, V. Lippiello, and B. Siciliano. An optimal trajectory planner for a robotic batting task: the table tennis example. In *International Conference on Informatics in Control, Automation and Robotics*, pages 90–101, Lisbon, P, 2016.

[5] R. L. Andersson. *A robot ping-pong player: experiment in real-time intelligent control*. MIT press, 1988.

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

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

[8] T. Senoo, A. Namiki, and M. Ishikawa. 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, 2006.

[9] Y. Sun, R. Xiong, Q. Zhu, J. Wu, and J. Chu. Balance motion generation for a humanoid robot playing table tennis. In *IEEE-RAS International Conference on Humanoid Robots*, pages 19–25, Bled, SI, 2011.

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

[11] C. H. Lai and T.I.J. Tsay. Self-learning for a humanoid robotic ping-pong player. *Advanced Robotics*, 25(9-10):1183–1208, 2011.

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

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

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

[15] H. Yanlong, S. Bernhard, and P. Jan. 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, 2015.

[16] Y. Huang, D. Büchler, B. Schölkopf, and J. Peters. Jointly learning trajectory generation and hitting point prediction in robot table tennis. In *IEEE-RAS International Conference on Humanoid Robots*, pages 650–655, Cancun, MX, 2016.

[17] S. Gomez-Gonzalez, G. Neumann, B. Schölkopf, and J. Peters. Using probabilistic movement primitives for striking movements. In *IEEE-RAS International Conference on Humanoid Robots*, pages 502–508, Cancun, MX, 2016.

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

[19] R. Silva, F.S. Melo, and M. Veloso. 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, 2015.

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

[21] J. Nonomura, A. Nakashima, and Y. Hayakawa. 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, 2010.

[22] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

[23] A. Nakashima, D. Ito, and Y. Hayakawa. 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, 2014.

[24] M.P. do Carmo. *Riemannian Geometry*. Mathematics (Boston, Mass.). Birkhäuser, 1992.

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

[26] V. Lippiello and F. Ruggiero. 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, 2012.

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

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

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

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