A Nonlinear Least Squares Approach for Nonprehensile Dual-Hand Robotic Ball Juggling

Diana Serra, Fabio Ruggiero, Vincenzo Lippiello, Bruno Siciliano

Department of Electrical Engineering and Information Technology,
University of Naples Federico II, Via Claudio 21, 80125, Naples, Italy.
{*diana.serra, fabio.ruggiero, vincenzo.lippiello, bruno.siciliano*}*@unina.it*

Abstract: This paper presents a nonlinear least squares approach to deal with dual-hand robotic ball juggling. The task considers the repetitive batting (throwing and catching in a single collision) of a ball between two paddles/hands in a nonprehensile way. In detail, assuming to measure the trajectory of the ball, by solving a sequence of nonlinear minimization problems through a least squares method, the configuration of the paddles at the next impact is computed online to juggle the ball between the hands. Afterwards, an optimal trajectory for the paddles is planned in SE(3). The proposed approach is evaluated on a semi-humanoid robot with 21 degrees of freedom. Numerical tests show the smoothness of the planned trajectories and the precision of the proposed juggling algorithm.

Keywords: Robotic Ball Juggling, Nonlinear Least Squares Estimation, Nonprehensile Manipulation

1. INTRODUCTION

Developments in *nonprehensile* dynamic manipulation are paving the way to endow robots with human-like dexterous capabilities for tasks execution. An object is manipulated in a nonprehensile way when it is not directly caged between the fingertips or the hand's palm. Moreover, the force closure constraint (see Murray and Sastry (1994)) does not hold. The grasp is then performed by exploiting only unilateral constraints, allowing the object to roll, slide, and break the contact with the robot manipulating it. Juggling, hopping, walking, pushing are typical examples of nonprehensile tasks, according to Lynch and Mason (1999). Since such kind of tasks are usually complex, skillful and dexterous, they are often divided in simpler subtasks, named *primitives*, such as rolling, sliding, throwing, catching, pushing, batting and so on. Assuming that a supervisory controller identifies the primitives in the nonprehensile task, a motion planner for each primitive might be conceived, as explained by Serra (2016). Focusing on *batting*, this combines the nonprehensile primitives of catching and throwing in a single collision. A representative example for this task is given by the table tennis game as described by Serra et al. (2016). This paper, instead, deals with the iteration of the batting primitive to create a simple *juggling* task where a ball is repetitively caught and thrown between two hands/paddles, without let it fall. Since juggling actions, in general, require high velocity and precision, the investigation of the related aspects is useful to confer dexterity and powerful manipulation skills to the robotic system . Moreover, the art of juggling provides also an interesting point of view on similar problems: as a matter of fact, Mason (2001) defines running as as kind of self-juggling, creating in this way a parallel between dynamic manipulation and locomotion.

This paper extends what previously presented by Serra et al. (2016), in which only the single batting action is taken into account. Now, the proposed algorithm is improved by considering the iterative motion required to juggle the ball between the paddles/hands. The aim of this work is thus to exploit the single batting primitive to plan an optimal path for the dual-hand ball juggling task, demonstrating how a complex nonprehensile manipulation can be dealt with a bottom-up approach, from the single primitive to the complete task. The paper is organized as follows. Section 2 presents the state of the art about robotic batting and juggling nonprehensile manipulation. Section 3 describes the hybrid dynamics within the ball juggling task. Section 4 presents the proposed dual-hand juggling algorithm. Section 5 shows the application of the proposed algorithm on the semi-humanoid robot, while numerical results and critical discussions endorse the performance of the proposed algorithm. Finally, Section 6 concludes the paper.

2. RELATED WORK

Planning and control of rhythmic tasks, like juggling ones, has been a very active research area over the last years. Buehler et al. (1988); Buhler et al. (1990); Buehler et al. (1994) investigate the stabilization of juggling tasks. In their work, they control the trajectory of a puck through a bar which is actuated around a revolute joint. By means of experimental validations, they propose the mirror algorithm to control the system. Aboaf et al. (1989); Schaal and Atkeson (1993, 1994a,b) investigate robotic juggling tasks putting instead emphasis on learning control aspects. Brogliato and Rio (2000) consider the robot juggling as a complementary-slackness hybrid mechanical system, where the force input mainly consists of a family of dead-beat feedback control laws, introduced via a recursive procedure. Lynch and Black (2001) show how to control a one joint revolute arm to bat a planar disk in a gravity field towards a desired juggling limit cycle. The controller is based on a real-time nonlinear optimization using a model of the discrete dynamics, with the recurrent control as the initial guess for the optimization procedure. Ronsse and Sepulchre (2006); Ronsse et al. (2007) investigate some feedback control strategies to continuously bounce a ball in the air. The influence of the impact acceleration on the robustness of the system due to parameter uncertainties is discussed. Trajectory tracking control for a one degree of

freedom (dof) juggling system is deeply studied by Sanfelice et al. (2007), also with multiple balls. They model mechanical systems with impacts through hybrid dynamics including both a set of differential equations/inclusions and a set of difference equations/inclusions on specific subsets of the state space. The approach is experimentally validated on a system consisting of a nearly smooth vertical shaft with a piston-actuated bouncing ball. In addition, Tian et al. (2013) present juggling experiments to validate a hybrid control algorithm capable of tracking a periodic reference trajectory. On the other hand, Reist and D'Andrea (2009, 2012) propose a bouncing ball robot without using any kind of external sensors (the so-called *blind juggler*). Taking the impact time measurements as feedback, they prove that the closed-loop performance is only marginally improved as compared to open-loop control. Aoyama et al. (2016) inspect the flower-stick juggling task through an analytical technique. By means of the the Poincarè maps, they analyze the stability of the approach exploiting the concept of virtual connecting manipulation. Tabata and Aiyama (2003) tackle the concepts of tossing and catching manipulation to pass an object between a couple of one dof manipulators. The authors provide a kinematic model for the task and an iterative learning control approach. Akbarimajd and Ahmadabadi (2007) propose a planner to reconfigure planar polygonal objects by juggling them between the palms of two hand-like manipulators.

As explained in the Introduction, since the dual-hand juggling task addressed within this paper is intended as an iterated version of the batting task, some literature about such nonprehensile primitive is now provided. A high-speed trajectory planner applied to the batting task is described by Senoo et al. (2006). The authors explicitly include the dynamic model of the robot inside their framework, while they rely upon a 1kHz high-speed vision system. Nevertheless, they do not go into details about the controller computational effort. 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 movements is controlled by nonlinear oscillators. Their activation and deactivation are coordinated by a hierarchical neural dynamic architecture. Hsiao et al. (2014) propose instead an optimal trajectory planner based on the simple minimization of the rebounding velocity of the ball, with the aim of reducing the speed of the paddle. Liu et al. (2012) propose an approach for the batting problem in the context of table tennis games. The method determines the state of the paddle employing the hybrid dynamic model of the ball during the free flight together with the state transition at the impact. In order to compute the control action, an approximated aerodynamic model is considered. Serra et al. (2016) extend the technique proposed by Liu et al. (2012) considering a complete aerodynamic model of the ball, while dealing with the real-time issue, and introducing a optimal motion planner in SE(3) for the trajectory of the paddle so as to correctly strike the ball at the impact time.

3. MODELLING OF THE HYBRID DYNAMICS

The dynamics of the ball is said to be hybrid since it consists of both the differential equations modelling the free flight aerodynamics, and the algebraic equations representing the impact reset map. The former are modelled through Newton's equations of motion, while the latter are a reset of the state, updated according to the impact detection. The considered dual-hand juggler is equipped with two paddles. Since the mass of the paddle is usually bigger than the mass of the ball, only the velocity of the ball is affected by the impact, and not the one of the paddle. In the following, the paddle that in turn is going to catch the ball is referred to as *impacting* paddle, whereas the other one is referred to as *free*. They are indicated through the i and f subscripts, respectively. On the other hand, the superscripts $-$ and $+$ represent the state of the ball before and after the impact time, respectively. As an assumption, a point contact occurs between the ball and the paddle during the collision. Let Σ_W be the fixed world frame, and let Σ_{ip} and Σ_{fp} be the frames placed at the center of the impacting and free paddles, respectively. The z-axis is denoted as the outward normal to the surfaces of the paddles. Let $\mathbf{p},\mathbf{v},\boldsymbol{\omega} \in \mathbb{R}^3$ be position, linear and angular velocities of the ball, respectively. Moreover, let $\mathbf{p}_{ip},\mathbf{v}_{ip},\boldsymbol{\omega}_{ip},\mathbf{p}_{fp},\mathbf{v}_{fp},\boldsymbol{\omega}_{fp} \in \mathbb{R}^3$ be position, linear and angular velocities of the impacting and free paddles, respectively, all expressed in Σ_W. Finally, let $\mathbf{R}_{ip},\mathbf{R}_{fp} \in SO(3)$ be the rotation matrices of Σ_{ip} and Σ_{fp} w.r.t. Σ_W, respectively. The hybrid dynamics of the ball is then

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

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

$$\mathbf{v}^+ = \mathbf{v}_{ip} + \mathbf{R}_{ip}\mathbf{A}_{vv}\mathbf{R}_{ip}^{\mathrm{T}}(\mathbf{v}^- - \mathbf{v}_{ip}) + \mathbf{R}_{ip}\mathbf{A}_{v\omega}\mathbf{R}_{ip}^{\mathrm{T}}\boldsymbol{\omega}^-, \tag{1c}$$

$$\boldsymbol{\omega}^+ = \mathbf{R}_{ip}\mathbf{A}_{\omega v}\mathbf{R}_{ip}^{T}(\mathbf{v}^- - \mathbf{v}_{ip}) + \mathbf{R}_{ip}\mathbf{A}_{\omega\omega}\mathbf{R}_{ip}^{T}\boldsymbol{\omega}^-, \tag{1d}$$

where $\mathbf{S}(\cdot) \in \mathbb{R}^{3\times3}$ is the skey-symmetric matrix operator, $\mathbf{g} = [0\ 0\ g]^T$ is the gravity acceleration vector, and $||\cdot||$ is the Euclidean norm. The functions $k_d(\mathbf{v},\boldsymbol{\omega})$ and $k_l(\mathbf{v},\boldsymbol{\omega})$, and the matrices of rebound coefficients $\mathbf{A}_{vv},\mathbf{A}_{v\omega},\mathbf{A}_{\omega v},\mathbf{A}_{\omega\omega} \in \mathbb{R}^{3\times3}$ are detailed by Liu et al. (2012). Equations (1a) and (1b) represent the ball aerodynamics, while equations (1c) and (1d) are the reset map of the ball's state. Notice that in (1b), the spin of the ball is assumed to be constant during the free flight. The equations of motion for the impacting and free paddles are respectively given by: $\dot{\mathbf{p}}_{ip} = \mathbf{v}_{ip}, \dot{\mathbf{R}}_{ip} = \mathbf{R}_{ip}\mathbf{S}(\boldsymbol{\omega}_{ip})$, and $\dot{\mathbf{p}}_{fp} = \mathbf{v}_{fp}, \dot{\mathbf{R}}_{fp} = \mathbf{R}_{fp}\mathbf{S}(\boldsymbol{\omega}_{fp})$. Notice that the ball is acted only at each impact time by mean of the velocity of the impacting paddle, which enters the ball dynamics through the reset map.

4. ALGORITHM FOR DUAL-HAND BALL JUGGLING

In order to accomplish the desired task, the paddles must repetitively intercept the ball in turn. It is assumed that the algorithm receives as input the measure of the state of the ball (i.e., through a visual system). This aspect is out of the scope of this paper. Hence, the goal of the proposed algorithm is to compute the orientation and the linear velocity of the impacting paddle so as to re-direct the ball towards the free paddle. In order to solve this task, the algorithm has to know a-priori a desired location where the ball has to be re-directed after the collision with the paddle, and the time interval Δt between two consecutive impacts. Such predetermined locations and Δt are input values shaping the trajectory of the ball within the juggling task. They should be thus tuned according to the available robot capabilities (i.e., maximum joint velocities) and the reachable workspace. After each collision, the algorithm swaps in turn the free and impacting paddles. The algorithm consists of two main phases, which are the topic of the following subsections. In the former, the computation of the state for each paddle is addressed by solving a sequence of nonlinear minimization problems through a least squares approach. In the latter, the trajectory optimization for the paddles is described. Fig. 1

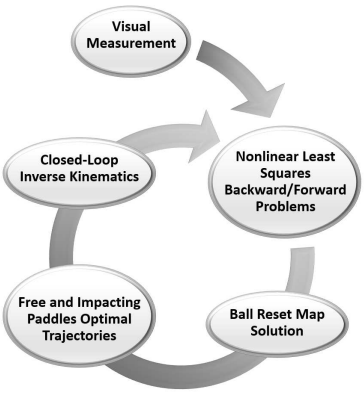


Fig. 1. Scheme of the dual-hand ball juggling algorithm.

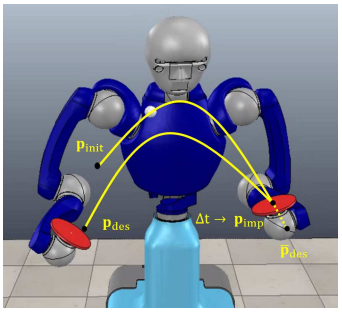


Fig. 2. Snapshot of one iteration of the dual-hand juggling task.

schematically resumes the proposed solution. In particular, after that the visual system provides the configuration of the ball (i.e., the initial state following the previous impact), solving two nonlinear minimization problems determines the states of the ball pre- and post- the impact. This allows to solve the reset map and compute the desired configuration for the impacting paddle. This is the first phase addressed in the next subsection. Afterwards, the optimal trajectory planner computes the minimum acceleration paths for the free and impacting paddles to respectively reconfigure at the initial pose and intercept the ball. Finally, a closed loop inverse kinematics provides the joint motion for the robot to practically accomplish the task. This is the phase addressed in the subsection 4.2.

4.1 Computation of the state of the impacting paddle

In order to predict the state of the ball before the impact and to properly control the paddle, a two-stage nonlinear least squares (NLS) fitting is designed. The three more relevant positions of the ball in the space are depicted in Fig. 2 for a single iteration of the repetitive dual-hand juggling task. $\mathbf{p}_{init} \in \mathbb{R}^3$ is the initial position of the ball soon after the previous impact, eventually given by the visual system; $\tilde{\mathbf{p}}_{des}$ is the desired location of the previous iteration step; $\mathbf{p}_{des} \in \mathbb{R}^3$ is the desired location of the current iteration step where the ball has to be re-directed after the impact; Δt is the predetermined time interval between two impacts. Notice that the location in the space where the collision between the paddle and the ball happens could be, in general, different from the predetermined desired location where the ball has to be re-directed after the impact. It goes without saying that, to obtain a repetitive dual-hand ball juggling task, the chosen position $\mathbf{p}_{des}$ and the interval time Δt have to shape the pattern trajectory of the ball such that the free paddle can intercept the ball within the robot workspace.

By knowing Δt and the velocity $\mathbf{v}^+_{init} \in \mathbb{R}^3$ after the previous collision, given again by the external visual system, it is possible to retrieve the predicted pre-impact states of the ball $(\mathbf{p}_{imp}, \mathbf{v}^-_{imp}) \in \mathbb{R}^3$ through the minimization problem

$$\min_{\mathbf{p}_{imp}, \mathbf{v}^-_{imp}} \left\| \begin{bmatrix} \widetilde{\mathbf{p}}_{init}(\mathbf{p}_{imp}, \mathbf{v}^-_{imp}) - \mathbf{p}_{init} \\ \widetilde{\mathbf{v}}^+_{init}(\mathbf{p}_{imp}, \mathbf{v}^-_{imp}) - \mathbf{v}^+_{init} \end{bmatrix} \right\|^2, \tag{2}$$

where $\widetilde{\mathbf{p}}_{init}$ and $\widetilde{\mathbf{v}}^+_{init}$ are obtained by numerically backward integrating the nonlinear equations (1a) and (1b) starting from the current optimization variables. In rough words, the predicted pre-impact states of the ball are chosen such that, back-integrating the aerodynamic model of the ball by Δt, the value of the obtained initial state is close as much as possible to the measured one. In order to obtain the velocity $\mathbf{v}^+_{imp} \in \mathbb{R}^3$ of the ball after the considered impact, which is necessary to direct the ball towards the predefined desired point $\mathbf{p}_{des}$, the following minimization problem is solved

$$\min_{\mathbf{v}^+_{imp}} \left\| \widetilde{\mathbf{p}}_{des}(\mathbf{v}^+_{imp}) - \mathbf{p}_{des} \right\|^2. \tag{3}$$

Notice that $\widetilde{\mathbf{p}}_{des}(\mathbf{v}^+_{imp})$ is obtained by numerically forward integrating the nonlinear equations (1a) and (1b) starting from $\mathbf{p}_{imp}$ and the current value of the optimization variable. In rough words, the predicted velocity of the ball after the impact is chosen such that, forward integrating the aerodynamic model of the ball by Δt, the value of the obtained final position is close as much as possible to the desired one. Hence, it is now possible to determine the configuration of the impacting paddle. As an assumption, at each iteration, the predicted impact position $\mathbf{p}_{imp}$ for the ball corresponds to the goal position for the impacting paddle. On the other hand, the orientation, the velocity of the impacting paddle are retrieved solving the reset map (1c)-(1d). Assume 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 $\mathbf{B}_{vv} = (\mathbf{I}_3 - \mathbf{A}_{vv})^{-1} \in \mathbb{R}^{3\times3}$, with $\mathbf{I}_3 \in \mathbb{R}^{3\times3}$ the identity matrix. Having solved (2) and (3), the pose and the velocity of the impacting paddle are computed through

$$\mathbf{v}_{ip} = \mathbf{v}^-_{imp} + \mathbf{R}_{ip}\mathbf{B}_{vv}(\mathbf{R}^T_{ip}(\mathbf{v}^+_{imp} - \mathbf{v}^-_{imp}) - \mathbf{A}_{v\omega}\omega^-_{imp}), \tag{4a}$$

$$\mathbf{R}_{ip} = \mathbf{R}_Y(\theta)\mathbf{R}_X(\phi), \tag{4b}$$

assuming that $\omega_{ip} = \mathbf{0}_3$ and $\mathbf{p}_{ip} = \mathbf{p}_{imp}$, where $\mathbf{0}_3 \in \mathbb{R}^3$ is the zero vector, and $\mathbf{R}_k(\cdot) \in SO(3)$ is the elementary rotation matrix with $k = \{X, Y\}$, representing the rotation of an angle around the k-axis. More details on the reset map solution and the matrices $\mathbf{A}_{vv}$, $\mathbf{A}_{v\omega}$ are given by Serra et al. (2016) and Liu et al. (2012). In the first iteration only the impacting paddle is actuated; after that, at each impact, the free paddle is imposed to stop in a rest position $\mathbf{p}_0 \in \mathbb{R}^3$ and orientation $\mathbf{R}_0 \in SO(3)$, respectively, yielding: $\mathbf{p}_{fp} = \mathbf{p}_0$, $\mathbf{v}_{fp} = \mathbf{0}$, $\mathbf{R}_{fp} = \mathbf{R}_0$, $\omega_{fp} = \mathbf{0}$. As soon as the first impact occurs, the initial configurations of the system for the next cycle is available. Impacting and free paddles are swapped in order to restart another iteration of the algorithm.

4.2 Dual-hand motion planner

The considered motion planner for the paddles replicates what previously presented by Serra et al. (2016). The configurations of the two paddles at each impact time, derived in the previous section, are the input of an optimal trajectory planner which online computes the optimal path for each paddle from the current state. The optimal trajectories are generated in $SE(3)$,

minimizing the acceleration functional, expressed on arbitrary manifold, implemented following the work by Zefran et al. (1998). In this way, the resulting trajectories require smooth velocities and accelerations at the end-effectors of the robot. Notice that the planned trajectories in SE(3) are independent from the chosen representation for the robots end-effectors' orientation. Changing the representation for the orientation do not imply a change in the planner, but only a different mapping from SE(3). Finally, the motion of the robot joints has to be computed from the planned Cartesian trajectories for the two paddles through a first order kinematic inversion algorithm.

5. SIMULATIONS

Numerical results validating the presented approach are shown in this section. Three case studies are described in the following. For all of them, the considered robot is a semi-humanoid with 21 dofs. This is equipped with an omnidirectional mobile base, a two dofs torso, a pan and tilt neck, two seven dofs arms, two cameras on the head following the ball, and two paddles firmly attached at its end-effectors. Within the simulations, the radii of ball and paddles are considered 2 cm and 15 cm, respectively, while the considered mass of the ball is 2.7 g. The values for the matrices $\mathbf{A}_{vv}, \mathbf{A}_{v\omega}, \mathbf{A}_{\omega v}, \mathbf{A}_{\omega\omega} \in \mathbb{R}^{3\times 3}$ in (1c) and (1d) are retrieved from Serra et al. (2016). Simulations are implemented in the Matlab environment, in connection with the V-REP virtual platform by Rohmer et al. (2013). The *ode45* solver is used by activating the *events* option, so as to integrate ball and paddles dynamics. The minimization problems (2) and (3) are solved through a least squares approach based on the Levembert-Marquardt's algorithm (*lsqcurvefit* solver). According to Lippiello and Ruggiero (2012), Cigliano et al. (2015) and Serra et al. (2016), this is well suited for real-time computations even with complex and nonlinear equations, such as the ball and paddles dynamics (1). As initial guess for the minimization problems, the initial values for the optimization variables $\mathbf{p}_{imp}$, $\mathbf{v}_{imp}^{-}$, and $\mathbf{v}_{imp}^{+}$ are analytically calculated by solving a simplified free flight model proposed by Liu et al. (2012). The *bvp4c* solver is finally used for the two boundary value problem in the minimum acceleration planner. More details about the implementation of the minimum acceleration planner can be found in Serra et al. (2016). In order to compute the motion of the robot joints from the planned minimum acceleration trajectories for the two paddles, a closed loop inverse kinematic algorithm by Siciliano et al. (2010) is employed.

First case study

In this case study, with reference to Fig. 2, the main assumption is given from the following equality $\mathbf{p}_{des} = \mathbf{p}_{init}$, holding for each iteration. This means that the desired location $\mathbf{p}_{des}$ for the ball after the impact with the paddle has been put equal to the previous impact location $\mathbf{p}_{init}$. The only exception is given by the first iteration since the ball has not performed yet any previous impact. The very first $\mathbf{p}_{des}$ point should be then assigned. Its value together with the initial position and velocity of the ball are detailed in Table 1. The chosen time interval between two consecutive impacts is set to the fixed value of $\Delta t = 0.5$ s. Only for the very first impact, this is set to 0.3 s. The rest positions $\mathbf{p}_0$ and orientations $\mathbf{R}_0$ for the paddles are indicated in Table 1. The 3D trajectories planned for the left and right hands of the semi-humanoid robot and the path of the ball are depicted in Fig. 3, where it is possible to appreciate that the paths planned for the paddles oscillate between the respective rest positions

Table 1. Input parameters for the first case study

Initial position of the ball	$[0.25\ \ 0\ \ 0.4]$ m
Initial velocity of the ball	$[1.65\ \ -0.2\ \ 0.05]$ m/s
First desired point (near right paddle)	$[0\ \ 0\ \ 0]$ m
Rest position right paddle	$[0\ \ 0\ \ -0.1]$ m
Rest position left paddle	$[0.7\ \ 0\ \ -0.1]$ m
Rest orientation for both paddles	$\mathbf{I}_3$

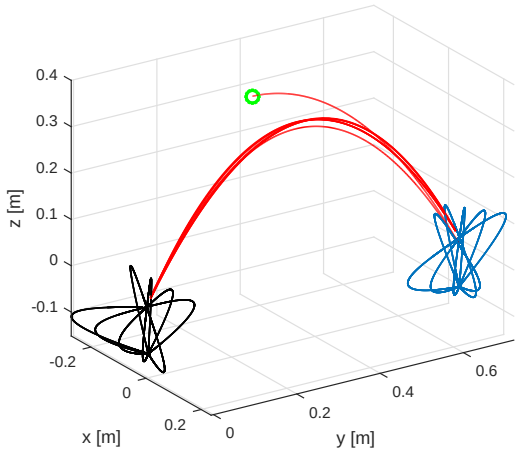


Fig. 3. 3D trajectories of the ball (red line), left (blue line) and right (black line) paddles, for the first case study.

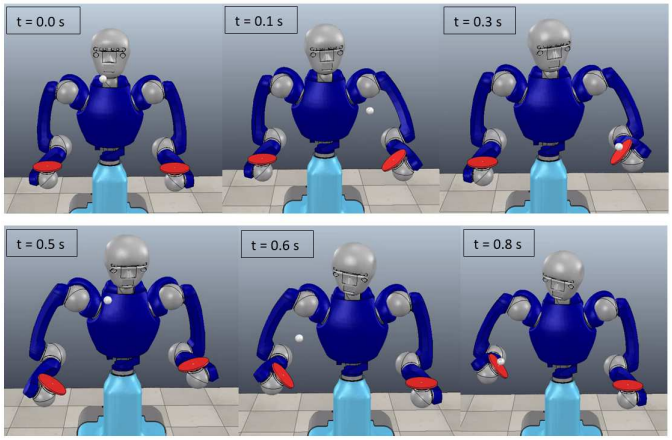


Fig. 4. Time sequence in the V-REP environment of the motion planned for the semi-humanoid robot to accomplish the first two ball juggling iterations. The first two impacts occur at 0.3 s and 0.8 s. See the accompanying video.

$\mathbf{p}_0$ and the impact points obtained by solving the minimization problems previously introduced. The picture shows that the ball follows the juggling pattern. In this picture (and in the next ones related to the 3D paths) the green marker defines the initial position of the ball. Fig. 6(a) shows that the average Euclidean norm of the error between the desired points and the actual ones is about 2 cm. Only for the very first iteration, the figure shows the actual impact point compared to the predicted one. The observed error is mainly justified by the numerical optimization procedure employed in the presented formulation and explained in the previous sections. Fig. 4 presents the time sequence of the semi-humanoid motion, resulting from the kinematic inversion, during the first two juggling iterations.

Table 2. Input parameters of the second case study

Initial position of the ball	$[0.5\ \ -0.2\ \ 0.5]$ m
Initial velocity of the ball	$[1\ \ -0.2\ \ 0.05]$ m/s
1st desired point	$[0\ \ 0\ \ 0]$ m
2nd desired point	$[0\ \ -0.01\ \ 0.06]$ m
3rd desired point	$[0.7\ \ -0.085\ \ 0.1026]$ m
4th desired point	$[0\ \ -0.02\ \ 0.12]$ m
5th desired point	$[0.7\ \ -0.095\ \ 0.1626]$ m
6th desired point	$[0\ \ -0.03\ \ 0.18]$ m
7th desired point	$[0.7\ \ -0.105\ \ 0.2226]$ m

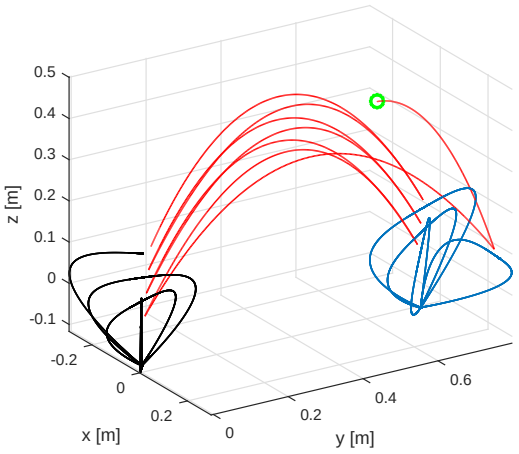


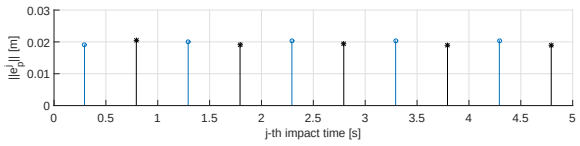
Fig. 5. 3D trajectories of the ball (red line), left (blue line) and right (black line) paddles, for the second case study.

Second case study

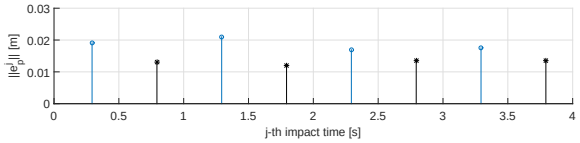
In this case study, the assumption made in the previous one is relaxed, and thus $\mathbf{p}_{des} \neq \mathbf{p}_{init}$. Therefore, seven different desired points are considered and listed in Table 2. In the same table, the initial position and velocity of the ball are detailed, which are different from the previous case study. The chosen time interval between two consecutive impacts is set to the fixed value of $\Delta t = 0.5$ s. Only for the very first impact, this is set to 0.3 s. The rest positions $\mathbf{p}_0$ and orientations $\mathbf{R}_0$ for the paddles are the same as the first case study (see Table 1). The shape of the minimum acceleration paths planned for the paddles of the semi-humanoid and the ball path for this second case study are depicted in Fig. 5. Again, the average Euclidean norm of the error between the desired points and the actual ones is about 2 cm, as plotted in Fig. 6(b).

Third case study

In this case study, the assumption $\mathbf{p}_{des} = \mathbf{p}_{init}$ is re-introduced, while the value of Δt now changes during each iteration. The intervals of time between two impacts, from the first to the seventh impact, are $\Delta t = [0.6\ \ 0.55\ \ 0.5\ \ 0.45\ \ 0.4\ \ 0.35\ \ 0.3]$ s. The initial position and velocity of the ball are equal to the ones of the first case study, as well as the very first desired point. In addition, the rest positions $\mathbf{p}_0$ and orientations $\mathbf{R}_0$ for the paddles are also equal to the first case study (see Table 1). The shape of the minimum acceleration paths planned for the paddles of the semi-humanoid and the ball path for this third case study are depicted in Fig. 7. It is possible to observe that, keeping fixed the desired points as highlighted in the assumption, the time Δt shapes the juggling patter. In particular, by reducing the time interval, the maximum height of the ball



(a) First case study.



(b) Second case study.

Fig. 6. Norm of the error between the desired positions and the actual ones, at each impact time. Blue circles and black stars respectively represent the left and right impact errors.

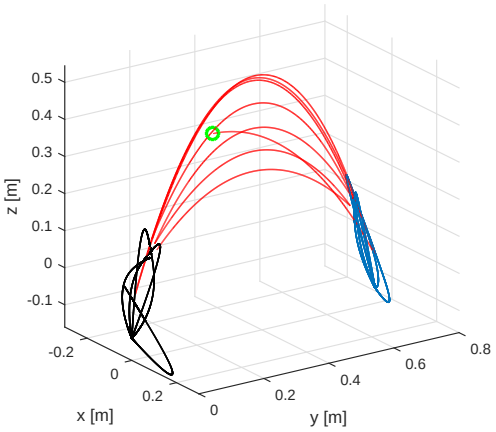


Fig. 7. 3D trajectories of the ball (red line), left (blue line) and right (black line) paddles, for the third case study.

reduces as well. Moreover, it is possible to notice from the same figure that, with a shorter Δt, the paddle impacts the ball before this last reaches the desired position $\mathbf{p}_{des}$, as in general depicted in Fig. 2. Due to space constraints, the norm error plot is not depicted for this case study. In the online video [1], the former and latter case studies are included. The video demonstrates the smoothness of the planned motion for the joints of the robot, and the synchronized motion of the two hands/paddles.

6. CONCLUSION AND FUTURE WORK

The batting primitive has been exploited to engender the ball juggling task. A sequence of nonlinear minimization problems is solved through a least squares approach and an optimal planner in SE(3) is proposed for the two paddles/hands. Simulation case studies have tested the performance of the proposed algorithm. As future work, the time interval Δt between two consecutive impacts will be automatically computed online as a further result of the optimization problems. The approach will be also experimentally validated on the physical robotic prototype available in the lab. Besides, a number of juggling patterns could be implemented exploiting again the batting primitive. The robot is now able to juggle only one ball, but the method may be further extended to add further balls.

[1] https://youtu.be/VtRe1zE_lhM

ACKNOWLEDGEMENTS

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

REFERENCES

Aboaf, E.W., Drucker, S., and Atkeson, C.G. (1989). Task-level robot learning: Juggling a tennis ball more accurately. In *1989 IEEE International Conference on Robotics and Automation*, 1290–1295. Scottsdale, USA.

Akbarimajd, A. and Ahmadabadi, M.N. (2007). Manipulation by juggling of planar polygonal objects using two 3-dof manipulators. In *2007 IEEE/ASME international conference on advanced intelligent mechatronics*, 1–6. Zurich, CH.

Aoyama, T., Takaki, T., Gu, Q., and Ishii, I. (2016). Control scheme of nongrasping manipulation based on virtual connecting constraint. In *2016 IEEE International Conference on Robotics and Automation*, 3819–3824. Stockholm, S.

Brogliato, B. and Rio, A.Z. (2000). On the control of complementary-slackness juggling mechanical systems. *IEEE Transactions on Automatic Control*, 45(2), 235–246.

Buehler, M., Koditschek, D.E., and Kindlmann, P.J. (1994). Planning and control of robotic juggling and catching tasks. *The International Journal of Robotics Research*, 13(2), 101–118.

Buehler, M., Koditschek, D.E., and Kindlmann, P. (1988). A one degree of freedom juggler in a two degree of freedom environment. In *IEEE International Workshop on Intelligent Robots*, 91–97. Tokyo, J.

Buhler, M., Koditschek, D.E., and Kindlmann, P.J. (1990). A family of robot control strategies for intermittent dynamical environments. *IEEE Control Systems Magazine*, 10(2), 16–22.

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.

Hsiao, T., Yang, C.M., Lee, I.H., and Hsiao, C.C. (2014). Design and implementation of a ball-batting robot with optimal batting decision making ability. In *2014 IEEE International Conference on Automation Science and Engineering*, 1026–1031. Taipei, Taiwan.

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

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

Lynch, K.M. and Black, C.K. (2001). Recurrence, controllability, and stabilization of juggling. *IEEE Transactions on Robotics and Automation*, 17(2), 113–124.

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

Mason, M. (2001). *Mechanics of robotic manipulation*. MIT Press, Cambridge, USA.

Murray, R. and Sastry, S.S. (1994). *A mathematical introduction to robotic manipulaiton*. CRC Press, Boca Raton, USA.

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*, 4042–4047. Manchester, UK.

Reist, P. and D'Andrea, R. (2009). Bouncing an unconstrained ball in three dimensions with a blind juggling robot. In *2009 IEEE International Conference on Robotics and Automation*, 1774–1781. Kobe, J.

Reist, P. and D'Andrea, R. (2012). Design and analysis of a blind juggling robot. *IEEE Transactions on Robotics*, 28(6), 1228–1243.

Rohmer, E., Singh, S., and Freese, M. (2013). V-REP: A versatile and scalable robot simulation framework. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1321–1326. Tokyo, J.

Ronsse, R., Lefevre, P., and Sepulchre, R. (2007). Rhythmic feedback control of a blind planar juggler. *IEEE Transactions on Robotics*, 23(4), 790–802.

Ronsse, R. and Sepulchre, R. (2006). Feedback control of impact dynamics: the bouncing ball revisited. In *45th IEEE Conference on Decision and Control*, 4807–4812. San Diego, USA.

Sanfelice, R.G., Teel, A.R., and Sepulchre, R. (2007). A hybrid systems approach to trajectory tracking control for juggling systems. In *46th IEEE Conference on Decision and Control*, 5282–5287. New Orleans, USA.

Schaal, S. and Atkeson, C.G. (1993). Open loop stable control strategies for robot juggling. In *1993 IEEE International Conference on Robotics and Automation*, 913–918. Atlanta, USA.

Schaal, S. and Atkeson, C.G. (1994a). Memory-based robot learning. In *1994 IEEE International Conference on Robotics and Automation*, 2928–2933. San Diego, USA.

Schaal, S. and Atkeson, C.G. (1994b). Robot juggling: implementation of memory-based learning. *IEEE Control Systems*, 14(1), 57–71.

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

Serra, D. (2016). Robot control for nonprehensile dynamic manipulation tasks. In *International Conference on Informatics in Control, Automation and Robotics, Doctoral Consortium*. Lisbon, P.

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

Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2010). *Robotics: Modelling, Planning and Control*. Springer-Verlag, London, UK.

Tabata, T. and Aiyama, Y. (2003). Passing manipulation by 1 degree-of-freedom manipulator-catching manipulation of tossed object without impact. In *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2920–2925. Las Vegas, USA.

Tian, X., Koessler, J.H., and Sanfelice, R.G. (2013). Juggling on a bouncing ball apparatus via hybrid control. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1848–1853. Tokyo, J.

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.