Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 50-1 (2017) 8975–8980

A Predictive Approach to Redundancy Resolution for Robot Manipulators

Marco Faroni * **Manuel Beschi** ** **Lorenzo Molinari Tosatti** ** **Antonio Visioli** *

* *Dipartimento di Ingegneria Meccanica e Industriale, University of Brescia, Brescia, Italy (e-mail: {m.faroni003, antonio.visioli}@unibs.it).*

** *Istituto di Tecnologie Industriali e Automazione, National Research Council, Milan, Italy (e-mail: {manuel.beschi, lorenzo.molinaritosatti}@itia.cnr.it)*

Abstract: In this paper, we propose a new method for the online redundancy resolution of robot manipulators, which implements a predictive strategy to calculate the optimal control action. In this way, it is possible to obtain a more efficient handling of the constraints, which represents one of the main issues in online resolution methods. The predictive model has been obtained by considering every joint as a kth-order integral system, and the predictive equations are derived from a continuous-time formulation. This allows the use of an irregular distribution of the prediction and control time instants and, as a consequence, longer prediction and control horizons can be obtained, without increasing the computational complexity of the algorithm. Finally, joint hard bounds are easily included in a linear-model-predictive-like framework, and the optimal control action is calculated by solving a linear quadratic problem. Simulation results for a 4-degree-of-freedom planar arm show the effectiveness of the method compared to purely local resolution techniques.

Keywords: robot manipulators, redundant robots, continuous model predictive control, inverse kinematics, hard joint limits, optimal redundancy handling.

1. INTRODUCTION

Redundant manipulators have been broadly exploited in industry and research, since they allow greater versatility in the task execution. As a matter of fact, the extra degrees of freedom permit to satisfy (completely or partially) secondary tasks, or to optimise a given objective function such as manipulability maximisation, joint availability or minimum energy consumption.

Several methods for the optimal resolution of the redundancy has been proposed during the years. They are classified mainly in local and global methods (Nakamura, 1990): the former class solves the redundancy using information related to the actual configuration of the robot, whereas the latter aims at obtaining a globally optimal solution, taking into account the whole desired task in the optimisation. Of course, local methods generally lead to poorer global performance (Faroni et al., 2016), but they allow an online resolution thanks to their lighter computational burden. On the contrary, the computational complexity of global methods is too high for real-time control applications and, therefore, they need to be performed offline.

As regards local methods, they are based on the resolution of the inverse kinematics of the robot, in order to obtain joint velocities and/or accelerations. The typical approach is based on the pseudo-inverse of the Jacobian of the manipulator (Siciliano, 1989). One of the main issues in the redundancy handling is the satisfaction of joint position, velocity and acceleration bounds. Pseudo-inverse methods do not explicitly take them into account. Typically, hard position bounds are converted into soft ones by means of a cost function that keeps the joints in the middle of their ranges (Liegeois, 1977). However, the impossibility of handling velocity and acceleration bounds would generate the saturation of joint commands, leading to the deformation of the primary task. In order to consider also these constraints, the local redundancy problem may be considered as a Quadratic Program (QP), with linear equality and inequality constraints (Kanoun et al., 2011). In this way, joint constraints are considered explicitly and an optimal solution is found in case the task is locally feasible.

However, the solution of these method is only locally optimal, and it does not take into account the future evolution of the task, the constraints, and the kinematic states. This means that the iterative local resolution of redundancy might move the robot to a disadvantageous configuration with respect to the constraints. This would cause undesired velocity/acceleration peaks or even the impossibility of satisfying the constraints with bounded control actions. In case no solution that satisfies all the constraints exists, the saturation of the actuators leads to the deformation of the task. Recent methods, for instance, implement task scaling techniques to preserve at least the geometrical task when no local feasible solution exists (Flacco et al., 2015). Anyway, considering that constraints and bounds along the task are usually known a priori, those situations could often be avoided, and this represents one of the main

10.1016/j.ifacol.2017.08.1324

drawbacks of purely local approaches, as pointed out in (Schuetz et al., 2014).
This paper proposes a method to solve the inverse kinematic of a redundant manipulator by means of a model predictive approach.
One great advantage of MPC techniques is the efficient handling of constraints. Moreover, the predictive approach permits to take into account the constraints in the successive time instants as well. For this reason, such approach seems to be a good candidate for overcoming the problems of punctual resolution of the redundancy in a robotic manipulator.
Following this principle, Schuetz et al. (2014) exploited Pontryagin's Minimum Principle to set up a nonlinear optimisation problem over a finite horizon. In (Zube, 2015), by contrast, a typical nonlinear MPC (NMPC) is applied to solve the inverse kinematics of a redundant manipulator. Although the results are certainly satisfactory in terms of kinematic performance, real-time applications for long-enough horizons seem to be still far away.
In this work, we propose a different approach, where every joint of the robot is treated as an kth-order integral system, while the desired trajectory is considered as a nonlinear constraint in the optimisation. As shown in Section 3.2, this constraint can be easily linearised using the prediction of the state variables. In this way it is possible to reduce the optimisation of some typical cost functions (e.g. minimum control effort and joint availability) to a linear quadratic programming problem, which can be solved much more rapidly. In order to obtain control and prediction horizons sufficiently long for the purposes of the paper, control and prediction time instants are not equally distributed along the horizons. As a matter of fact, the time instants are more frequent at the beginning of the horizons and less frequent in the end. This strategy permits to reduce the number of variables involved in the prediction and in the optimisation and, therefore, it dramatically decreases the computational burden of the algorithm.
Another advantage of the proposed method is that first and second (or even third) order resolution levels may be obtained just by using either a single or a double (or triple) integral system respectively, without any modification to the algorithm. Moreover, the intrinsic structure of linear MPC permits to handle position, velocity and acceleration boundaries in an elegant and efficient manner.

2. BACKGROUND

Consider an n-degree-of-freedom manipulator and an m-dimensional task with $m < n$. The associated direct kinematics is given by:

$$x = f(q) \tag{1}$$

where $x \in \mathbb{R}^m$ is the task position vector and $q \in \mathbb{R}^n$ is the joint position vector. Differentiating with respect to time gives:

$$\dot{x} = J\,\dot{q} \tag{2}$$

where $J = \partial f/\partial q$ is the $m \times n$ Jacobian matrix, $\dot{x} \in \mathbb{R}^m$ is the task velocity vector and $\dot{q} \in \mathbb{R}^n$ is the joint velocity vector.
Typical inverse kinematics are based on the solution of the linear system (2) by means of the pseudoinverse of the Jacobian. This can be performed at velocity or acceleration level (De Luca and Oriolo, 1991). However, pseudoinverse methods do not provide an intrinsic way of handling boundaries and constraints. As a matter of fact, their mere application may cause the violation of position, velocity and acceleration boundaries if no strategies are implemented to avoid it. Moreover, the violation of velocity and acceleration bounds would saturate the actuators, with consequent deformation of the primary task.
Alternatively, it is possible to represent the redundancy problem as a quadratic optimisation, as follows:

$$\begin{aligned} \dot{q} &= \operatorname{argmin} \frac{1}{2} e(\dot{q})^T e(\dot{q}) \\ \text{s.t.} \quad & J(q)\dot{q} = \dot{x}\,; \quad \dot{Q}_{min} \le \dot{q} \le \dot{Q}_{max} \end{aligned} \tag{3}$$

where $\dot{Q}_{min}$ and $\dot{Q}_{max}$ represent the kinematic constraints at velocity level.
When the function $e(\dot{q})$ is linear with respect to $\dot{q}$, the minimisation becomes a QP.
The extension of such method to the acceleration level is easily derived and gives:

$$\begin{aligned} \ddot{q} &= \operatorname{argmin} \frac{1}{2} e(\ddot{q})^T e(\ddot{q}) \\ \text{s.t.} \quad & J(q)\ddot{q} = \ddot{x} - \dot{J}\dot{q}\,; \quad \ddot{Q}_{min} \le \ddot{q} \le \ddot{Q}_{max} \end{aligned} \tag{4}$$

where $\ddot{Q}_{min}$ and $\ddot{Q}_{max}$ represent the kinematic constraints at acceleration level.
The kinematic constraints typically include position, velocity and/or acceleration limits, as extensively explained in (Flacco et al., 2012).

3. PROPOSED METHOD

3.1 Formulation of the predictive model

Consider an n-degree-of-freedom manipulator and an m-dimensional task, wit $m < n$. From a control perspective, the aim of redundancy resolution consists in finding the optimal control actions that have to be applied to the joints, taking into account the constraints introduced by the Cartesian task. To this end, every joint is modelled as a kth-order integral system:

$$q_i(s) = \frac{1}{s^k} u_i(s) \qquad \forall\, i = 1 \dots n \tag{5}$$

Equivalently, choosing the state vector as

$$\xi_i := [q_i \;\; \dot{q}_i \dots q_i^{(k-1)}]^T \qquad \forall\, i = 1 \dots n \tag{6}$$

the following state-space formulation $(\tilde{A}, \tilde{B}, \tilde{C})$ results:

$$\dot{\xi}_i(t) = \tilde{A}\xi_i(t) + \tilde{B}u_i(t) \tag{7}$$

$$q_i(t) = \tilde{C}\xi_i(t) \tag{8}$$

where the matrices $\tilde{A} \in \mathbb{R}^{k\times k}$, $\tilde{B} \in \mathbb{R}^{k\times 1}$, and $\tilde{C} \in \mathbb{R}^{1\times k}$ are given by:

$$\tilde{A} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ \vdots & & & \ddots & 1 \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix} \tag{9}$$

$$\tilde{B} = [\; 0 \;\cdots\; 0 \;\; 1 \;]^T \tag{10}$$

$$\tilde{C} = [\; 1 \;\; 0 \;\cdots\; 0 \;] \tag{11}$$

(Note that in the remarkable case of $k = 1$, the matrices reduce to the scalar case, in which $\tilde{A} = 0$, $\tilde{B} = 1$, $\tilde{C} = 1$.)

In this way the control action u_i results to be the kth derivative of the corresponding joint position. Choosing $k = 1, 2$, or 3 the joint will be controlled at velocity, acceleration, or jerk level respectively.
It is now possible to obtain the kinematic model of the whole manipulator from the single-joint models by coupling them in state-space formulation (A, B, C), where:

$$A := \text{blkdiag}\big(\tilde{A}, \dots, \tilde{A}\big) \qquad \in \mathbb{R}^{nk \times nk} \tag{12}$$

$$B := \text{blkdiag}\big(\tilde{B}, \dots, \tilde{B}\big) \qquad \in \mathbb{R}^{nk \times n} \tag{13}$$

$$C := \text{blkdiag}\big(\tilde{C}, \dots, \tilde{C}\big) \qquad \in \mathbb{R}^{n \times nk} \tag{14}$$

where blkdiag$(\cdot)$ generates a block-diagonal matrix from the given matrices. The complete state vector results:

$$\xi = [q_1, \ \dot{q}_1, \dots q_1^{(k-1)}, \dots, q_n, \ \dot{q}_n, \dots q_n^{(k-1)}]^T \tag{15}$$

whereas the output vector is:

$$q = [q_1, \ q_2, \dots q_n]^T \tag{16}$$

Once the state-space model of the whole manipulator has been obtained, the application of classic discrete-time MPC would result straightforward. On the other hand, in this work, a continuous-time approach has been preferred. Such choice allows the irregular distribution of prediction and control time instants along the horizons, as will be explained later.
Therefore, consider a time instant t_0 and the corresponding state $\xi_0 = \xi(t_0)$. The predicted output at a generic time $t = t_0 + \tau$, $\tau > 0$, is given by:

$$q(t_0 + \tau) = C\, \mathrm{e}^{A\tau}\, \xi_0 + C\, \mathrm{e}^{A\tau} \int_0^\tau \mathrm{e}^{-A\gamma}\, B u(\gamma)\, d\gamma \tag{17}$$

Now assume that B is constant and u is a staircase function. Let $\bar{t}_i$ be the time instants, with respect to t_0, in which the steps of the function are applied and let $u(t_0+\bar{t}_i)$ be the input value in the time interval between $\bar{t}_{i-1}$ and $\bar{t}_i$. These intervals are defined, then, as $\Delta\bar{t} = [\bar{t}_1 - t_0, \bar{t}_2 - \bar{t}_1, , \dots, \bar{t}_c - \bar{t}_{c-1}]$, $c > 0$. Note that the lengths of the intervals does not need to be the same.
Thus, (17) becomes:

$$q(t_0 + \tau) = C\, \mathrm{e}^{A\tau}\, \xi_0 + C \sum_{i=1}^{c} \Gamma_i\, u(t_0 + \bar{t}_i) \tag{18}$$

where:

$$\Gamma_i := \begin{cases} \mathrm{e}^{A(\tau - \bar{t}_i)} \int_0^{\Delta\bar{t}_i} \mathrm{e}^{A\gamma}\, d\gamma\, B, & \text{if } \tau \geq \bar{t}_i \\ \int_0^{\tau - \bar{t}_{i-1}} \mathrm{e}^{A\gamma}\, d\gamma\, B, & \text{if } \bar{t}_{i-1} < \tau < \bar{t}_i \\ 0, & \text{if } \tau < \bar{t}_i \wedge \tau < \bar{t}_{i-1} \end{cases} \tag{19}$$

assuming $\bar{t}_0 = 0$.
Differentiating (18) with respect to τ gives:

$$\dot{q}(t_0 + \tau) = CA\, \mathrm{e}^{A\tau}\, \xi_0 + C \sum_{i=1}^{c} \dot{\Gamma}_i\, u(t_0 + \bar{t}_i) \tag{20}$$

where:

$$\dot{\Gamma}_i = \begin{cases} A\, \mathrm{e}^{A(\tau - \bar{t}_i)} \int_0^{\Delta\bar{t}_i} \mathrm{e}^{A\gamma}\, d\gamma\, B, & \text{if } \tau \geq \bar{t}_i \\ \mathrm{e}^{A(\tau - \bar{t}_{i-1})}\, B, & \text{if } \bar{t}_{i-1} < \tau < \bar{t}_i \\ 0, & \text{if } \tau < \bar{t}_i \wedge \tau < \bar{t}_{i-1} \end{cases} \tag{21}$$

Repeating for a dth-order derivative, the following equation results:

$$q^{(d)}(t_0 + \tau) = CA^d\, \mathrm{e}^{A\tau}\, \xi_0 + C \sum_{i=1}^{c} \Gamma_i^{(d)}\, u(t_0 + \bar{t}_i) \tag{22}$$

where $d = 1, \dots, k-1$, and

$$\Gamma_i^{(d)} = \begin{cases} A^d\, \Gamma, & \text{if } \tau \geq \bar{t}_i \\ A^{d-1}\, \mathrm{e}^{A(\tau - \bar{t}_{i-1})}\, B, & \text{if } \bar{t}_{i-1} < \tau < \bar{t}_i \\ 0, & \text{if } \tau < \bar{t}_i \wedge \tau < \bar{t}_{i-1} \end{cases} \tag{23}$$

assuming $A^0 = I$ in case $A = 0$.
Having defined the control time instants $\bar{t}$, a series of constant matrices $\Phi_{d,\tau}$ can be defined for a given prediction time τ, such that:

$$\Phi_{d,\tau} := \Big[C\, \Gamma_1^{(d)},\ C\, \Gamma_2^{(d)}, \dots, C\, \Gamma_c^{(d)}\Big] \tag{24}$$

for $d = 0, \dots, k-1$, having assumed $\Gamma_i^{(d)} = \Gamma_i$ if $d = 0$.
Substituting (24) in (18) and (22) gives respectively:

$$q(t_0 + \tau) = C\, \mathrm{e}^{A\tau}\, \xi_0 + \Phi_{0,\tau}\, u_{(\bar{t})} \tag{25}$$

$$q^{(d)}(t_0 + \tau) = CA^d\, \mathrm{e}^{A\tau}\, \xi_0 + \Phi_{d,\tau}\, u_{(\bar{t})} \tag{26}$$

where $u_{(\bar{t})} := [u(t_0 + \bar{t}_1), \dots, u(t_0 + \bar{t}_c)]^T$.
Extending this reasoning to a generic number of prediction times $[\tau_1, \dots, \tau_p]$, $p > 0$, leads to the following systems of equations:

$$\begin{cases} q(t_0 + \tau_1) = C\, \mathrm{e}^{A\tau_1}\, \xi_0 + \Phi_{0,\tau_1}\, u_{(\bar{t})} \\ \qquad\qquad \vdots \\ q(t_0 + \tau_p) = C\, \mathrm{e}^{A\tau_p}\, \xi_0 + \Phi_{0,\tau_p}\, u_{(\bar{t})} \end{cases} \tag{27}$$

$$\begin{cases} q^{(d)}(t_0 + \tau_1) = CA^d\, \mathrm{e}^{A\tau_1}\, \xi_0 + \Phi_{d,\tau_1}\, u_{(\bar{t})} \\ \qquad\qquad \vdots \\ q^{(d)}(t_0 + \tau_p) = CA^d\, \mathrm{e}^{A\tau_p}\, \xi_0 + \Phi_{d,\tau_p}\, u_{(\bar{t})} \end{cases} \tag{28}$$

As for the control time instants, the prediction instants $[\tau_1, \dots, \tau_p]$ are not necessarily equally spaced.
The systems (27) and (28) can be rewritten in a single matrix form as:

$$\hat{q}^{(d)} = L_d\, \xi_0 + F_d u_{(\bar{t})} \qquad d = 0, \dots, k-1 \tag{29}$$

where $\hat{q} \in \mathbb{R}^{np \times 1}$ is the predicted output vector at times $(t_0 + \tau_1, \dots, t_0 + \tau_p)$, $u_{(\bar{t})} \in \mathbb{R}^{nc \times 1}$ is the vector of the control inputs, and:

$$L_d = \begin{bmatrix} CA^d\, \mathrm{e}^{A\tau_1} \\ \vdots \\ CA^d\, \mathrm{e}^{A\tau_p} \end{bmatrix} \in \mathbb{R}^{np \times nk}, \quad F_d = \begin{bmatrix} \Phi_{d,\tau_1} \\ \vdots \\ \Phi_{d,\tau_p} \end{bmatrix} \in \mathbb{R}^{np \times nc}.$$

The formulation presented in (29) is convenient since the output and its derivatives have clear physical meaning. In this way, the series of matrices L_d and F_d permits to easily express joint positions, velocities or accelerations (depending on the order k of the model) with respect to the control action. For instance, choosing $k = 2$, the joint positions and velocities are given by:

$$\hat{q} = L_0\, \xi_0 + F_0 u_{(\bar{t})} \tag{30}$$

$$\dot{\hat{q}} = L_1\, \xi_0 + F_1 u_{(\bar{t})} \tag{31}$$

whereas the vector $u_{(\bar{t})}$ represents the future accelerations at time instants $(t_0 + \bar{t}_1, \dots, t_0 + \bar{t}_c)$.

In classic MPC, prediction and control instants are equally distributed along the respective horizons and correspond to the sampling times. However, robotic systems generally have sampling periods of the order of few milliseconds, not allowing long-enough horizons and low computational burdens at the same time. Although the derivation of the

predictive equations in continuous time might seems less intuitive, their great advantage consists in the possibility of using a free distribution of prediction and control instants along their horizons, disregarding the real sample time of the system. As a matter of fact, it is not necessary for the time instants $\bar{t}_i$ and τ_i to be neither equally distributed nor coincident with a sample instant.
Adopting an irregular partition of the predictive and control horizons, thus, it is possible to obtain much longer horizons, without increasing the computational burden of the algorithm.

3.2 Linearisation of the task constraint

Up to this point, a predictive model of the robot in the joint space has been obtained, with longer horizons thanks to the continuous-time formulation of the model. On the other hand, nothing has been said about the satisfaction of the Cartesian task.
In order to perform the task, the robot has to satisfy (2) for all time instants. Imposing this constraint to every control instant $\bar{t}_i$ of the control horizon gives:

$$J^*\big(q(t_0+\bar{t}_i)\big)\,\dot{q}(t_0+\bar{t}_i) = \dot{x}(t_0+\bar{t}_i) \quad \forall\, i = 1,\dots,c \tag{32}$$

where:

$$J^*\big(q(t_0+\bar{t}_i)\big) = \text{blkdiag}\Big(J\big(q(t_0+\bar{t}_1)\big),\dots,J\big(q(t_0+\bar{t}_c)\big)\Big).$$

This constraint is nonlinear with respect to the variables $q(t_0+\bar{t}_i)$, due to the Jacobian.
However, using (30) it is possible to calculate the prediction of the joint position. Therefore, approximating $q(t_0+\bar{t}_i)$ with its prediction at the previous cycle, (32) becomes linear and depends only on the joint velocities $\dot{q}(t_0+\bar{t}_i)$.
Moreover, considering that the relationship between the future joint velocities and the control action is given by (31), the task constraint becomes linear and dependent on the sole control action $u_{(\bar{t})}$:

$$J^*(\hat{q})F_{1c}\,u_{(\bar{t})} = \dot{x}(t_0+\bar{t}_i) - J^*(\hat{q})L_{1c}\,\xi_0 \quad \forall\, i = 1,\dots,c \tag{33}$$

where F_{1c} and L_{1c} are the velocity prediction matrices as defined in (29), but related only to the control time instants. In other words, they correspond to the first nc rows of the matrices F_1 and L_1 used in (31).
(Note that, in case of first-order model, the joint velocity is given simply by the control action $u_{(\bar{t})}$. So, in that case, the matrix F_{1c} would result equal to the identity matrix I_c, while the matrix L_{1c} would be null.)
Obviously, the constraint must be updated at each iteration with the new prediction $\hat{q}$ in the Jacobians, the new initial state ξ_0, and the Cartesian velocities $\dot{x}$ related to the corresponding time instants.

3.3 Handling of constraints

The formulation of the predictive model (29) allows a straightforward inclusion of constraints on linear combinations of the state variables. As a matter of fact, the use of (29) as prediction of the kinematic variables permits to express such constraints with respect to the control variables in a easy and intuitive manner.

In the resolution of the inverse kinematics, the main constraints are represented by the mechanical limits of the joints and the boundaries of the kinematic variables, such as velocity and acceleration. Therefore, chosen the order k of the kinematic model, these constraints are given by:

$$\begin{aligned} q_{min} &\le q &&\le q_{max} \\ &\vdots \\ q^{(k)}_{min} &\le q^{(k)} &&\le q^{(k)}_{max} \end{aligned} \tag{34}$$

By using (29) it is possible to impose these boundaries with respect to the control actions $u_{(\bar{t})}$ as:

$$\begin{aligned} q_{min} &\le L_0\xi_0 + F_0 u_{(\bar{t})} &&\le q_{max} \\ &\vdots \\ q^{(k)}_{min} &\le u_{(\bar{t})} &&\le q^{(k)}_{max} \end{aligned} \tag{35}$$

which can be easily rewritten in the typical form of linear inequality constraints with respect to $u_{(\bar{t})}$ as:

$$\begin{aligned} F_d\,u_{(\bar{t})} &\le q^{(d)}_{max} - L_d\xi_0 && \forall\, d = 0,\dots,k-1 \\ -F_d\,u_{(\bar{t})} &\le -q^{(d)}_{min} + L_d\xi_0 && \forall\, d = 0,\dots,k-1 \\ u_{(\bar{t})} &\le q^{(k)}_{max} \\ -u_{(\bar{t})} &\le -q^{(k)}_{min} \end{aligned} \tag{36}$$

3.4 Optimal control action

In classic linear MPC, the control action to apply to the system is obtained by minimising a cost function in the following form:

$$\eta(u) = \int_0^{\tau_p} e^2(t)dt + \lambda \int_0^{\bar{t}_c} u^2(t)dt \tag{37}$$

where e is a linear function of the output and, due to the linearity of the model, of the control variable as well. This cost function may be approximated as follows:

$$\eta(u) \approx e(\tau)^T\,\overline{\Delta\tau}\,e(\tau) + \lambda u^T_{(\bar{t})}\,\overline{\Delta t}\,u_{(\bar{t})} \tag{38}$$

where $\overline{\Delta\tau} = \text{diag}(\tau_1, \tau_2 - \tau_1, \dots, \tau_p - \tau_{p-1})$, $\overline{\Delta t} = \text{diag}(\Delta t_1,\dots,\Delta t_c)$, $e(\tau)$ is given by the function e evaluated at time instants $(t_0+\tau_1,\dots,t_0+\tau_p)$, and $\lambda > 0$.
Typically, e is given by the difference between the output variable and a reference signal. In this way, the optimisation of the cost function leads to a trade-off between the minimisation of the control error (first term in (38)) and the control effort (second term in (38)). Moreover, if the control variable u has to satisfy only linear constraints, the optimisation of (38) can be carried out by means of typical QP procedures. Applying the same approach to the inverse kinematics, it is possible to minimise a generic quadratic cost function with respect to the control variable u or to the state variables ξ (reminding the linear relationship between them, given by (29)).
The Cartesian task is satisfied by imposing the linearised constraint (33). Moreover, (33) requires the task constraint to be satisfied at the future control times as well, and this permits to obtain a prediction of the states which takes into account the task constraint within the control horizon. Finally, the inequality constraints (36) ensure that the kinematic variables remain between their boundaries within the control horizon. In this way, a QP with linear constraints results, that intrinsically takes into account the Cartesian task and the limits of the kinematic variables in an efficient and elegant manner.

3.5 Typical quadratic cost function in redundancy resolution

Typical quadratic functions used in the redundancy resolution are:

- *Minimum control effort*
  In this case only the second term appears in the cost function (37), then $e(t) = 0$ and
  $$\eta(u) = \int_0^{\tau_p} u^2(t)dt \approx u_{(\bar{t})}^T \overline{\Delta t}\, u_{(\bar{t})}. \qquad (39)$$
  Generally, if the order of the system is greater than 1, it is convenient to include in the cost function the minimisation of all the kinematic variables. The resulting cost function will be therefore:
  $$\eta(u) = \int_0^{\tau_p} \sum_{d=1}^{k} \left(q^{(k)}(t)\right)^2 dt \approx$$
  $$\sum_{d=1}^{k-1} \left((L_d\xi_0 + F_d u)^T \overline{\Delta\tau}\, (L_d\xi_0 + F_d u) \right) + u_{(\bar{t})}^T \overline{\Delta t}\, u_{(\bar{t})} \qquad (40)$$
- *Joint-range availability*
  The distance between the joint positions and a given reference value $\bar{q}$ (usually the middle value of the joint range) has to be minimised, thus the first term e in (37) and (38) becomes:
  $$e(t) = W(q(t) - \bar{q}) \approx W(L_0\xi_0 + F_0 u_{(\bar{t})} - \bar{q}) \qquad (41)$$
  where W is an appropriate weighting matrix.

4. SIMULATION RESULTS

As an illustrative example, consider a 4-degree-of-freedom planar arm with unitary length axis. As shown in Figure 1, the base of the robot corresponds to the origin of the Cartesian plane and the task consists of a circle of centre $C(0, 2)$ and radius $r = 1$ m. The end-effector has to execute the circle, starting from and returning to the point $(1, 2)$ in a total execution time equal to 3 s. The scalar Cartesian velocity of the end-effector is requested to follow a trapezoidal timing law. In this example, we choose to minimise the cost function (41). In this way, the control action will be calculated in order to attract the joint positions in the middle of their ranges. At the same time, the control actions ensure that the joint variables respect boundaries and constraints along the control horizon. Thus, at each cycle, the optimal solution corresponding to the first time instant is applied to the system, following the receding horizon principle. This simulation aims at demonstrating that, using the proposed method, the resulting kinematic profiles give better global results in terms of satisfaction of the boundaries and minimisation of the control action. Similar conclusions can be obtained by minimising other performance indices (e.g. (39)). The results will be omitted for the sake of brevity.
The method has been tested at both first and second kinematic levels, that is, k equal to 1 and 2, with the tuning parameters shown in Table 1. The number and the distribution of the prediction and control instants have been chosen empirically. As a general approach, the first time instants are chosen at a distance equal to the sampling time of the system, while the following ones are more and more sparse.

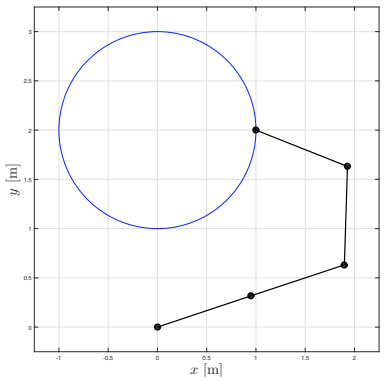

Fig. 1. Desired task and initial configuration of the robot.

Table 1. Tuning parameters used in simulation.

T [ms]	1
N_p	40
N_c	10
τ [ms]	$T,\ 2T, ..., 5T,\ 10T, ..., 30T,\ 40T, ..., 80T,\ 105T, ..., 705T$
$\bar{t}$ [ms]	$0,\ T,\ 2T, ..., 5T,\ 10T, ..., 25T$
λ	$5 \cdot 10^{-3}$

Table 2. Position, velocity and accelerations boundaries for the simulated robot.

Joint	q_{min} [rad]	q_{max} [rad]	$\dot{q}_{min}$ [rad/s]	$\dot{q}_{max}$ [rad/s]	$\ddot{q}_{min}$ [rad/s^2]	$\ddot{q}_{max}$ [rad/s^2]
1	0	π	-4	4	-100	100
2	$-\pi/2$	$\pi/2$	-4	4	-100	100
3	$-\pi/2$	$\pi/2$	-4	4	-100	100
4	$-\pi/2$	$\pi/2$	-4	4	-100	100

The proposed method has been compared against the local quadratic optimisations (3) and (4), setting the joint range limits as reported in Table 2. The resulting joint kinematic states are shown in Figure 2 for the velocity level resolution, and in Figures 3 and 4 for the acceleration level.
Although the QP solver considers the bounds on position, velocity and acceleration of the joints, the local optimisation causes abrupt changes in the control variable, since it is not able to foresee the activation of the constraints (Figures 2 and 4, magenta lines). This situation often leads to the impossibility of satisfying both the primary task and the joint bounds, and task scaling techniques would be needed to preserve at least the geometric path. The proposed method, on the contrary, considers the presence of the boundaries along the whole control horizon. In this way, the algorithm calculates a smoother control action that tends to avoid its saturation. As a consequence, if the task is feasible for the robot capabilities, it is more likely to find a solution that does not cause task deformation.

Of course the better performance of the proposed method implies a heavier computational burden. Nonetheless, the calculation of the control action is still a QP problem and this permits to obtain a much lighter optimisation problem if compared to typical NMPC schemes, as (Zube, 2015). The optimisation has been performed in Matlab by using the QP solver described in (Ferreau et al., 2014). The optimisation takes about 0.45 ms on average (using a 2.5 GHz Intel Core i5-2520M processor). During the

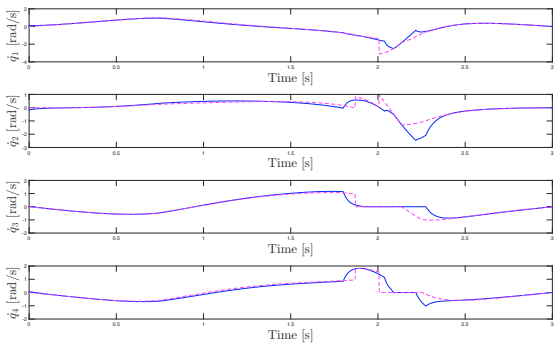

Fig. 2. Comparison between the joint velocities given by the local-QP (dashed magenta) and the proposed (solid blue) methods in the constrained case ($k = 1$).

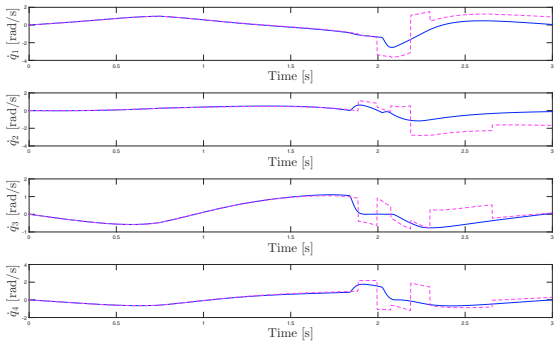

Fig. 3. Comparison between the joint velocities given by the local-QP (dashed magenta) and the proposed (solid blue) methods in the constrained case ($k = 2$).

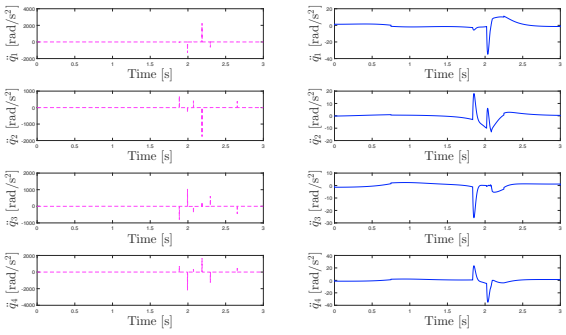

Fig. 4. Comparison between the joint accelerations given by the local-QP (dashed magenta) and the proposed (solid blue) methods in the constrained case ($k = 2$). Note the different axis scales.

first cycle the optimisation takes about 3.8 ms, since the solver has to initialise the problem. Existing algorithms using NMPC are, at least, one order of magnitude slower, even on a dedicated hardware and with a C/Fortran implementation.

5. CONCLUSIONS

This paper presented a new method for the online redundancy handling of robot manipulators. It has been shown that the use of a model-based predictive strategy permits to take into account the future evolution of the kinematic variables of the robot. In this way, the control actions are optimised with respect to a desired cost function (e.g. joint range availability) and to the presence of joint boundaries, along a certain time horizon. Considering the presence of the boundaries in the future permits to take action before their activation and this leads to smoother trends of the kinematic states. Moreover, if the desired task is feasible for the manipulator, the algorithm is more likely to find a solution without deforming the original trajectory. The irregular distribution of control and prediction instants permits to have longer horizons without increasing the computational burden. Finally, the reduction of the Cartesian task to a linear constraint allows the use of linear MPC techniques, ensuring a computational complexity lower than the classic NMPC strategies used in the literature to the same purpose.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Union H2020 program under grant agreement n. 637095 (FourByThree) and ECSEL-2016-1 under grant agreement n. 737453 (I-MECH).

REFERENCES

De Luca, A. and Oriolo, G. (1991). Issues in acceleration resolution of robot redundancy. In *Proceedings IFAC Symposium on Robot Control*, 93–98. Vienna (Austria).

Faroni, M., Beschi, M., Visioli, A., and Molinari Tosatti, L. (2016). A global approach to manipulability optimisation for a dual-arm manipulator. In *Proceedings IEEE International Conference on Emerging Technologies and Factory Automation*. Berlin (Germany).

Ferreau, H., Kirches, C., Potschka, A., Bock, H., and Diehl, M. (2014). qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6, 327–363.

Flacco, F., De Luca, A., and Khatib, O. (2012). Motion control of redundant robots under joints constraints: Saturation in the null space. In *Proceedings IEEE International Conference on Robotics and Automation*, 285–292. St. Paul (USA).

Flacco, F., De Luca, A., and Khatib, O. (2015). Control of redundant robots under hard joint constraints: Saturation in the null space. *IEEE Transaction on Robotics*, 31, 637–654.

Kanoun, O., Lamiraux, F., and Wieber, P.B. (2011). Kinematic control of redundant manipulators: generalizing the task priority framework to inequality tasks. *IEEE Transactions on Robotics*, 27, 785–792.

Liegeois, A. (1977). Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems Man and Cybernetics*, 7, 868–871.

Nakamura, Y. (1990). *Advanced Robotics: Redundancy and Optimization*. Addison-Wesley, MA, USA.

Schuetz, C., Buschmann, T., Baur, J., Pfaff, J., and Ulbrich, H. (2014). Predictive online inverse kinematics for redundant manipulators. In *Proceedings IEEE International Conference on Robotics and Automation*, 5056–5061. Hong Kong (China).

Siciliano, B. (1989). Kinematic control of redundant robot manipulators: A tutorial. *Journal of Intelligent and Robotic Systems*, 3, 201–212.

Zube, A. (2015). Cartesian nonlinear model predictive control of redundant manipulators considering obstacles. In *Proceedings IEEE International Conference on Industrial Technology*, 137–142. Seville (Spain).