The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Prioritized Closed-Loop Inverse Kinematic Algorithms for Redundant Robotic Systems with Velocity Saturations

Gianluca Antonelli, Giovanni Indiveri and Stefano Chiaverini

Abstract— **Standard kinematics prioritized task based motion control solutions do not take into account the physical limitations in terms of maximum actuator speed of robots. In this paper, a prioritized task based kinematics control solution is presented that, under given conditions on the kind of concurrent tasks to be pursued, guarantees task error stability and convergence. Moreover the joint velocities are guaranteed to be bounded by a desired threshold. As for other a null-space projection techniques known in the literature, joint speed commands are computed in such a way that lower priority tasks do not interfere with higher priority ones in the assumption that joint speeds can be arbitrarily large: in addition, if joint speeds are to be bounded by a desired value, joint velocity commands are limited by dynamically chosen values depending on the task priority. As a result, joint velocities are always bounded such that, if necessary, higher priority tasks are executed first.**

I. Introduction

Kinematic task redundancy occurs in a robotic system when the available Degrees of Freedom (DOFs) exceed the minimum number required to perform a given task. Redundancy can be exploited to perform the task while optimizing some criteria or eventually to perform more tasks concurrently. Several control law architectures have been proposed in the literature to exploit redundancy for the execution of multiple tasks such as, e.g., [1], [2], [3] that share the main approach in the definition of a primary task which is fulfilled with higher priority with respect to a secondary task. Extensions of the algorithm proposed in [2] to a large number of tasks is given in [4]. Within the same framework, the work presented in [5] investigates the use of a proper weighted pseudo-inverse. In [6] the null-space projector is used together with a projection based on the transpose of the Jacobian and the stability analysis is presented for the two-task case.

A common feature to many of these approaches consists in defining a task-priority such that lower priority tasks can be filtered by proper null-space projectors of higher priority ones before being commanded. The spirit of these solutions is to assure the compatibility between tasks by projecting lower priority ones in the null-space of higher priority ones. Yet only recently a rigorous Lyapunov based stability and convergence analysis was provided in [7], [8] where tasks are classified according their Jacobian matrices and conditions for Lyapunov stability of the tasks error are described. All previous results, including the ones in [7], [8], are based on the assumption that the commanded joint velocities can be actually implemented on the robot. In practice, due to the physical limitations of actuators, the maximum velocity that a given actuator can provide is upper bounded. In this paper, building on the results in [9], the solution proposed in [7], [8] is extended to include upper bounds on actuator velocities. As a result, a general solution for assuring Lyapunov stability of the tasks errors in regulation task-prioritized problems is presented.

Gianluca Antonelli and Stefano Chiaverini with the Dipartimento di Automazione, Elettromagnetismo, Ingegneria dell'Informazione e Matematica Industriale, Università degli Studi di Cassino, Via G. Di Biasio 43, 03043, Cassino (FR), Italy, {antonelli, chiaverini}@unicas.it, http://webuser.unicas.it/lai/robotica

Giovanni Indiveri is with the Dipartimento Ingegneria Innovazione, Università del Salento, Via Monteroni, 73100, Lecce, Italy, giovanni.indiveri@unile.it, http://cor.unile.it

II. Mathematical Background

By defining as $\boldsymbol{\sigma} \in \mathbb{R}^m$ the task variable to be controlled and as $\boldsymbol{q} \in \mathbb{R}^n$ the vector of the system configuration, it is:

$$\boldsymbol{\sigma} = \boldsymbol{f}(\boldsymbol{q}) \tag{1}$$

with the corresponding differential relationship:

$$\dot{\boldsymbol{\sigma}} = \frac{\partial \boldsymbol{f}(\boldsymbol{q})}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q}) \dot{\boldsymbol{q}}\,, \tag{2}$$

where $\boldsymbol{J}(\boldsymbol{q}) \in \mathbb{R}^{m \times n}$ is the configuration-dependent task Jacobian matrix and $\dot{\boldsymbol{q}} \in \mathbb{R}^n$ is the system velocity. Notice that n depends on the specific robotic system considered, in case of an industrial manipulator n is generally $n = 6$ and $\boldsymbol{q}$ is the vector of joint positions. For a differential mobile robot $n = 3$, and the term *system configuration* simply refers to the robot position/orientation. For a multi-robot system n is related to the number of robots, in case of a full actuated underwater vehicle $n = 6$, finally, an anthropomorphic robots can reach very large value of n.

Motion references $\boldsymbol{q}_{\text{des}}(t)$ for the robotic system starting from desired values $\boldsymbol{\sigma}_{\text{des}}(t)$ of the task function are usually generated by inverting the (locally linear) mapping in eq. (2) [10]. A typical requirement is to pursue minimum-norm velocity: when $n \geq m$ and $\boldsymbol{J}(\boldsymbol{q})$ has full rank, this leads to the least-squares solution (dependencies in the Jacobian are dropped out to increase readability):

$$\dot{\boldsymbol{q}}_{\text{des}} = \boldsymbol{J}^{\dagger} \dot{\boldsymbol{\sigma}}_{\text{des}} = \boldsymbol{J}^{\mathrm{T}} \left(\boldsymbol{J} \boldsymbol{J}^{\mathrm{T}} \right)^{-1} \dot{\boldsymbol{\sigma}}_{\text{des}}\,. \tag{3}$$

In order to avoid the well known problem of numerical drift, a Closed Loop Inverse Kinematics (CLIK) version of

the algorithm is usually implemented [11], namely,

$$\dot{\boldsymbol{q}}_{\text{des}} = \boldsymbol{J}^{\dagger}\left(\dot{\boldsymbol{\sigma}}_{\text{des}} + \boldsymbol{\Lambda}\widetilde{\boldsymbol{\sigma}}\right), \tag{4}$$

where $\boldsymbol{\Lambda}$ is a suitable positive-definite matrix of gains and $\widetilde{\boldsymbol{\sigma}}$ is the task error defined as

$$\widetilde{\boldsymbol{\sigma}} = \boldsymbol{\sigma}_{\text{des}} - \boldsymbol{\sigma}.$$

In case of system redundancy, i.e., if $n > m$, the classic general solution contains a null projector operator [1]:

$$\dot{\boldsymbol{q}}_{\text{des}} = \boldsymbol{J}^{\dagger}\left(\dot{\boldsymbol{\sigma}}_{\text{des}} + \boldsymbol{\Lambda}\widetilde{\boldsymbol{\sigma}}\right) + \left(\boldsymbol{I}_n - \boldsymbol{J}^{\dagger}\boldsymbol{J}\right)\dot{\boldsymbol{q}}_{\text{null}}, \tag{5}$$

where $\boldsymbol{I}_n$ is the $(n \times n)$ Identity matrix and the vector $\dot{\boldsymbol{q}}_{\text{null}} \in \mathbb{R}^n$ is an arbitrary system velocity vector. It can be recognized that the operator $\left(\boldsymbol{I}_n - \boldsymbol{J}^{\dagger}\boldsymbol{J}\right)$ projects a generic velocity vector in the null space of the Jacobian matrix. This corresponds to generate a motion of the robotic system that does not affect that of the given task; this is usually named as *internal* motion inheriting its meaning from the original application of these techniques where the primary task was the end-effector position of a manipulator.

For highly redundant systems, multiple tasks can be arranged in priority in order to try to fulfill most of them, hopefully all of them, simultaneously. Let us consider, for sake of simplicity, 3 regulation tasks (i.e. $\dot{\boldsymbol{\sigma}}_{\text{des}} = \boldsymbol{0}$ for each task), that will be denoted with the subscript a, b and c, respectively:

$$\begin{aligned} \boldsymbol{\sigma}_{\text{a}} &= \boldsymbol{f}_{\text{a}}(\boldsymbol{q}) \\ \boldsymbol{\sigma}_{\text{b}} &= \boldsymbol{f}_{\text{b}}(\boldsymbol{q}) \\ \boldsymbol{\sigma}_{\text{c}} &= \boldsymbol{f}_{\text{c}}(\boldsymbol{q}) \end{aligned}$$

where $\boldsymbol{\sigma}_{\text{a}} \in \mathbb{R}^{m_a}$, $\boldsymbol{\sigma}_{\text{b}} \in \mathbb{R}^{m_b}$ and $\boldsymbol{\sigma}_{\text{c}} \in \mathbb{R}^{m_c}$. For each of the tasks a corresponding Jacobian matrix can be defined, in detail $\boldsymbol{J}_{\text{a}} \in \mathbb{R}^{m_a \times n}$, $\boldsymbol{J}_{\text{b}} \in \mathbb{R}^{m_b \times n}$ and $\boldsymbol{J}_{\text{c}} \in \mathbb{R}^{m_c \times n}$. Let us further define the corresponding null space projectors as

$$\begin{aligned} \boldsymbol{N}_{\text{a}} &= \left(\boldsymbol{I}_n - \boldsymbol{J}_{\text{a}}^{\dagger}\boldsymbol{J}_{\text{a}}\right) \\ \boldsymbol{N}_{\text{b}} &= \left(\boldsymbol{I}_n - \boldsymbol{J}_{\text{b}}^{\dagger}\boldsymbol{J}_{\text{b}}\right). \end{aligned}$$

A generalization of the singularity-robust task priority inverse kinematic solution proposed in [12] and [13] leads to the following equation:

$$\dot{\boldsymbol{q}}_{\text{des}} = \boldsymbol{J}_{\text{a}}^{\dagger}\boldsymbol{\Lambda}_{\text{a}}\widetilde{\boldsymbol{\sigma}}_{\text{a}} + \boldsymbol{N}_{\text{a}}\boldsymbol{J}_{\text{b}}^{\dagger}\boldsymbol{\Lambda}_{\text{b}}\widetilde{\boldsymbol{\sigma}}_{\text{b}} + \boldsymbol{N}_{\text{ab}}\boldsymbol{J}_{\text{c}}^{\dagger}\boldsymbol{\Lambda}_{\text{c}}\widetilde{\boldsymbol{\sigma}}_{\text{c}} \tag{6}$$

where

$$\boldsymbol{J}_{\text{ab}} = \begin{bmatrix} \boldsymbol{J}_{\text{a}} \\ \boldsymbol{J}_{\text{b}} \end{bmatrix}, \quad \boldsymbol{N}_{\text{ab}} = \left(\boldsymbol{I}_n - \boldsymbol{J}_{\text{ab}}^{\dagger}\boldsymbol{J}_{\text{ab}}\right) \tag{7}$$

the approach in eq. (6) will be denoted as *augmented projections method*.

In what follows, the assumption that the Jacobians are full rank will be made.

A. Proposed modified inverse kinematics

Eq. (6) can be written as:

$$\dot{\boldsymbol{q}}_{\text{des}} = \dot{\boldsymbol{q}}_{\text{a}} + \dot{\boldsymbol{q}}_{\text{b}} + \dot{\boldsymbol{q}}_{\text{c}} \tag{8}$$

where

$$\begin{aligned} \dot{\boldsymbol{q}}_{\text{a}} &= \boldsymbol{J}_{\text{a}}^{\dagger}\boldsymbol{\Lambda}_{\text{a}}\widetilde{\boldsymbol{\sigma}}_{\text{a}} \\ \dot{\boldsymbol{q}}_{\text{b}} &= \boldsymbol{N}_{\text{a}}\boldsymbol{J}_{\text{b}}^{\dagger}\boldsymbol{\Lambda}_{\text{b}}\widetilde{\boldsymbol{\sigma}}_{\text{b}} \\ \dot{\boldsymbol{q}}_{\text{c}} &= \boldsymbol{N}_{\text{ab}}\boldsymbol{J}_{\text{c}}^{\dagger}\boldsymbol{\Lambda}_{\text{c}}\widetilde{\boldsymbol{\sigma}}_{\text{c}}. \end{aligned}$$

In the presence of actuator saturations, commanding the joint velocities given by eq. (8) may result in a very poor performance: consider the case, by example, where the highest priority task command $\dot{\boldsymbol{q}}_{\text{a}}$ is compatible with the joint velocity thresholds, but $\dot{\boldsymbol{q}}_{\text{des}}$ is not. Assume that when commanding $\dot{\boldsymbol{q}}_{\text{des}}$ some joint velocities saturate and other do not. In this case the command $\dot{\boldsymbol{q}}_{\text{des}}$ would not guarantee the fulfillment on any task (not even the first) in spite of the null space projection and in spite of $\dot{\boldsymbol{q}}_{\text{a}}$ alone being feasible w.r.t. to the joint velocity saturation thresholds. To overcome this limit, the solution in eq. (6) is modified in the following

$$\dot{\boldsymbol{q}}_{\text{des}} = \alpha_{\text{a}}\dot{\boldsymbol{q}}_{\text{a}} + \alpha_{\text{b}}\dot{\boldsymbol{q}}_{\text{b}} + \alpha_{\text{c}}\dot{\boldsymbol{q}}_{\text{c}} \tag{9}$$

where the scaling factors $\alpha_{\text{a}}, \alpha_{\text{b}}$ and α_{c} belong to $[0, 1]$ and are defined as described in the following. Indicating with $\underline{c}_{\text{a}j}$ and $\overline{c}_{\text{a}j}$ the lower and upper velocity bounds of joint j for the highest priority task a, consider the set of candidate scaling factors:

$$\Omega_{\text{a}} := \left\{\beta_{\text{a}} \in (0, 1] : \beta_{\text{a}}\, \dot{q}_{\text{a}j} \in [\,\underline{c}_{\text{a}j}, \overline{c}_{\text{a}j}\,] \;\forall\, j\right\}. \tag{10}$$

Then α_{a} is computed as:

$$\alpha_{\text{a}} = \begin{cases} \max \Omega_{\text{a}} & \text{if } \Omega_{\text{a}} \text{ is not empty} \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

so that the task a velocity command is eventually scaled down or totally canceled to match the thresholds $\underline{c}_{\text{a}j}$ and $\overline{c}_{\text{a}j}\ \forall\, j$. Consequently, the thresholds for the priority 2 task b are computed as

$$\underline{c}_{\text{b}j} = \underline{c}_{\text{a}j} - \alpha_{\text{a}}\, \dot{q}_{\text{a}j} \tag{12}$$

$$\overline{c}_{\text{b}j} = \overline{c}_{\text{a}j} - \alpha_{\text{a}}\, \dot{q}_{\text{a}j} \tag{13}$$

and the scaling factor α_{b} is defined as:

$$\alpha_{\text{b}} = \begin{cases} \max \Omega_{\text{b}} & \text{if } \Omega_{\text{b}} \text{ is not empty} \\ 0 & \text{otherwise,} \end{cases} \tag{14}$$

where

$$\Omega_{\text{b}} := \left\{\beta_{\text{b}} \in (0, 1] : \beta_{\text{b}}\, \dot{q}_{\text{b}j} \in [\,\underline{c}_{\text{b}j}, \overline{c}_{\text{b}j}\,] \;\forall\, j\right\}. \tag{15}$$

Similarly, the thresholds $\underline{c}_{\text{c}j}$ and $\overline{c}_{\text{c}j}\ \forall\ j$ and the scaling factor α_{c} are computed through equations corresponding to (12) - (15) by replacing the subscript b with c and a with b.

B. Definitions

Applying basic geometric similarities, some definitions concerning the relationships between two tasks will also be given in this Section.

Given two generic tasks, denoted with the lower scripts x and y, they will be defined as *orthogonal* if:

$$\boldsymbol{J}_{\mathrm{x}}\boldsymbol{J}_{\mathrm{y}}^{\dagger} = \boldsymbol{O}_{m_x \times m_y} \tag{16}$$

where $\boldsymbol{O}_{m_x \times m_y}$ is the $(m_x \times m_y)$ null matrix. The two tasks will be defined as *dependent* if

$$\rho(\boldsymbol{J}_{\mathrm{x}}^{\dagger}) + \rho(\boldsymbol{J}_{\mathrm{y}}^{\dagger}) > \rho\left([\, \boldsymbol{J}_{\mathrm{x}}^{\dagger} \quad \boldsymbol{J}_{\mathrm{y}}^{\dagger} \,]\right). \tag{17}$$

where $\rho(\cdot)$ denotes the rank of the matrix. Finally, they will be defined as *independent* if

$$\rho(\boldsymbol{J}_{\mathrm{x}}^{\dagger}) + \rho(\boldsymbol{J}_{\mathrm{y}}^{\dagger}) = \rho\left([\, \boldsymbol{J}_{\mathrm{x}}^{\dagger} \quad \boldsymbol{J}_{\mathrm{y}}^{\dagger} \,]\right) \tag{18}$$

and they are not orthogonal.

It is worth noticing that the three conditions of orthogonality, dependency and independency given may be verified by resorting to the transpose of the corresponding Jacobians instead of the pseudoinverse, in fact, they share the same span. Thus, the independency condition becomes:

$$\rho(\boldsymbol{J}_{\mathrm{x}}^{\mathrm{T}}) + \rho(\boldsymbol{J}_{\mathrm{y}}^{\mathrm{T}}) = \rho\left([\, \boldsymbol{J}_{\mathrm{x}}^{\mathrm{T}} \quad \boldsymbol{J}_{\mathrm{y}}^{\mathrm{T}} \,]\right). \tag{19}$$

In the following section it will be demonstrated that these definitions, and condition in eq. (19), play an important role in the eventual convergence of the task errors.

III. Stability Analysis

Let us define a candidate Lyapunov function as

$$V = \frac{1}{2}\tilde{\boldsymbol{\sigma}}_{\mathrm{a}}^{\mathrm{T}}\tilde{\boldsymbol{\sigma}}_{\mathrm{a}} \tag{20}$$

whose time derivative is

$$\dot{V} = \tilde{\boldsymbol{\sigma}}_{\mathrm{a}}^{\mathrm{T}}\dot{\tilde{\boldsymbol{\sigma}}}_{\mathrm{a}} = \tilde{\boldsymbol{\sigma}}_{\mathrm{a}}^{\mathrm{T}}\left(\dot{\boldsymbol{\sigma}}_{\mathrm{a,des}} - \dot{\boldsymbol{\sigma}}_{\mathrm{a}}\right) \tag{21}$$

that for the regulation problem at hand yields

$$\dot{V} = -\tilde{\boldsymbol{\sigma}}^{\mathrm{T}}\boldsymbol{J}_{\mathrm{a}}\dot{\boldsymbol{q}} = -\alpha_{\mathrm{a}}\tilde{\boldsymbol{\sigma}}^{\mathrm{T}}\boldsymbol{J}_{\mathrm{a}}\dot{\boldsymbol{q}}_{\mathrm{a}} \tag{22}$$

notice, in fact, that all the remaining terms are null since, even if scaled, are still projected onto the null space of the primary task a.

Noticing that if $\alpha_{\mathrm{a}} > 0$ at all times and $\boldsymbol{\Lambda}_{\mathrm{a}}$ is positive definite, also $\boldsymbol{\Lambda}_{\mathrm{a}}\,\alpha_{\mathrm{a}}$ will be positive definite at all times, the time derivative of V given in eq. (22) will result in

$$\dot{V} = -\alpha_{\mathrm{a}}\tilde{\boldsymbol{\sigma}}_{\mathrm{a}}\boldsymbol{\Lambda}_{\mathrm{a}}\tilde{\boldsymbol{\sigma}}_{\mathrm{a}} < 0 \tag{23}$$

and it is possible to say that

$$\exists\, t_{\mathrm{a}} : \qquad \forall\, t > t_{\mathrm{a}} \quad \Rightarrow \quad \|\dot{\boldsymbol{q}}_{\mathrm{a}}\| < \varepsilon_{\mathrm{a}} \tag{24}$$

that comes directly from the observation that the task error is decreasing to zero and the Jacobian pseudoinverse is bounded.

Let us define $\tilde{\boldsymbol{\sigma}} \in \mathbb{R}^{m_a+m_b}$ as

$$\tilde{\boldsymbol{\sigma}} = \begin{bmatrix} \tilde{\boldsymbol{\sigma}}_{\mathrm{a}} \\ \tilde{\boldsymbol{\sigma}}_{\mathrm{b}} \end{bmatrix}, \tag{25}$$

that is the stacked vector of first two tasks' errors. A possible Lyapunov function candidate is given by

$$V = \frac{1}{2}\tilde{\boldsymbol{\sigma}}^{\mathrm{T}}\tilde{\boldsymbol{\sigma}} \tag{26}$$

whose time derivative is

$$\dot{V} = -\tilde{\boldsymbol{\sigma}}^{\mathrm{T}}\begin{bmatrix} \boldsymbol{J}_{\mathrm{a}} \\ \boldsymbol{J}_{\mathrm{b}} \end{bmatrix}\dot{\boldsymbol{q}} \tag{27}$$

i.e.,

$$\dot{V} = -\tilde{\boldsymbol{\sigma}}^{\mathrm{T}}\begin{bmatrix} \boldsymbol{J}_{\mathrm{a}} \\ \boldsymbol{J}_{\mathrm{b}} \end{bmatrix}\left(\alpha_{\mathrm{a}}\dot{\boldsymbol{q}}_{\mathrm{a}} + \alpha_{\mathrm{b}}\dot{\boldsymbol{q}}_{\mathrm{b}}\right) \tag{28}$$

showing that the term in $\alpha_{\mathrm{c}}\dot{\boldsymbol{q}}_{\mathrm{c}}$, being projected onto the null space of the higher priority terms, does not affect the dynamics of the tasks a and b. The time derivative of V can be rewritten as

$$\dot{V} \;=\; -\tilde{\boldsymbol{\sigma}}^{\mathrm{T}}\begin{bmatrix} \alpha_{\mathrm{a}}\boldsymbol{\Lambda}_{\mathrm{a}} & \boldsymbol{O}_{m_a,m_b} \\ \alpha_{\mathrm{a}}\boldsymbol{J}_{\mathrm{b}}\boldsymbol{J}_{\mathrm{a}}^{\dagger}\boldsymbol{\Lambda}_{\mathrm{a}} & \alpha_{\mathrm{b}}\boldsymbol{J}_{\mathrm{b}}\boldsymbol{N}_{\mathrm{a}}\boldsymbol{J}_{\mathrm{b}}^{\dagger}\boldsymbol{\Lambda}_{\mathrm{b}} \end{bmatrix}\tilde{\boldsymbol{\sigma}}.$$

It is worth noticing that it is possible to consider the scaling factors α_* as part of the gains: consequently the task b is characterized by a gain

$$\boldsymbol{\Lambda}'_{\mathrm{b}} = \alpha_{\mathrm{b}}\boldsymbol{\Lambda}_{\mathrm{b}} \tag{29}$$

According to the results presented in [7], [8] the above reported $\dot{V}$ function is negative definite with a proper choice of the gains and if the tasks are independent. However, due to the velocity saturation, this is not guaranteed anymore $\forall t$. It is worth noticing that $\forall t > t_{\mathrm{a}}$ the velocity associated with task a is smaller than a given ε_{a}; this implies that the thresholds $\underline{c}_{\mathrm{b}_j}$ and $\overline{c}_{\mathrm{b}_j}$ of the task b are approaching the corresponding thresholds of task a. Assuming that $\alpha_{\mathrm{b}} > 0$ $\forall t > t^*_{\mathrm{b}}$, then $\dot{V} < 0$ and the tasks errors of the first two tasks are guaranteed to converge to zero.

A consequence is that the following arise

$$\exists t_{\mathrm{b}} > t_{\mathrm{a}} : \qquad \forall t > t_{\mathrm{b}} \quad \Rightarrow \quad \left\|\dot{\boldsymbol{q}}_{\mathrm{b}}\right\| < \varepsilon_{\mathrm{b}} \tag{30}$$

It is now possible to consider the task c in the stability analysis by defining $\tilde{\boldsymbol{\sigma}} \in \mathbb{R}^{m_a+m_b+m_c}$ as

$$\tilde{\boldsymbol{\sigma}} = [\, \tilde{\boldsymbol{\sigma}}_{\mathrm{a}}^{\mathrm{T}} \quad \tilde{\boldsymbol{\sigma}}_{\mathrm{b}}^{\mathrm{T}} \quad \tilde{\boldsymbol{\sigma}}_{\mathrm{c}}^{\mathrm{T}} \,]^{\mathrm{T}}, \tag{31}$$

A possible Lyapunov function candidate is given by

$$V = \frac{1}{2}\tilde{\boldsymbol{\sigma}}^{\mathrm{T}}\tilde{\boldsymbol{\sigma}} \tag{32}$$

whose time derivative is

$$\dot{V} = -\tilde{\boldsymbol{\sigma}}^{\mathrm{T}}\, M\, \tilde{\boldsymbol{\sigma}} \tag{33}$$

$$M = \begin{bmatrix} \alpha_{\mathrm{a}}\boldsymbol{\Lambda}_{\mathrm{a}} & \boldsymbol{O}_{m_a,m_b} & \boldsymbol{O}_{m_a,m_c} \\ \alpha_{\mathrm{a}}\boldsymbol{J}_{\mathrm{b}}\boldsymbol{J}_{\mathrm{a}}^{\dagger}\boldsymbol{\Lambda}_{\mathrm{a}} & \alpha_{\mathrm{b}}\boldsymbol{J}_{\mathrm{b}}\boldsymbol{N}_{\mathrm{a}}\boldsymbol{J}^{\dagger}{}_{\mathrm{b}}\boldsymbol{\Lambda}_{\mathrm{b}} & \boldsymbol{O}_{m_b,m_c} \\ \alpha_{\mathrm{a}}\boldsymbol{J}_{\mathrm{c}}\boldsymbol{J}_{\mathrm{a}}^{\dagger}\boldsymbol{\Lambda}_{\mathrm{a}} & \alpha_{\mathrm{b}}\boldsymbol{J}_{\mathrm{c}}\boldsymbol{N}_{\mathrm{a}}\boldsymbol{J}_{\mathrm{b}}^{\dagger}\boldsymbol{\Lambda}_{\mathrm{b}} & \alpha_{\mathrm{c}}\boldsymbol{J}_{\mathrm{c}}\boldsymbol{N}_{\mathrm{ab}}\boldsymbol{J}_{\mathrm{c}}^{\dagger}\boldsymbol{\Lambda}_{\mathrm{c}} \end{bmatrix}$$

By repeating the same line of reasoning applied to the analysis of stability and convergence of the first two tasks only, it can be concluded that under the hypothesis that $\alpha_{\mathrm{c}} > 0 \; \forall t > t_{\mathrm{c}}^{*}$, there will exist a time instant $t_{\mathrm{c}} > t_{\mathrm{b}} > t_{\mathrm{a}}$ such that $\forall t > t_{\mathrm{c}}$ the time derivative of the Lyapunov function V of the three tasks (eq. (32)) is negative definite provided that the gain conditions derived in [7], [8] are also satisfied.

IV. Numerical Case Study

To verify the effectiveness of the smart inverse kinematics saturation proposed a numerical case study has been developed. A 6-DOFs planar manipulator with revolute joint positions $\boldsymbol{q} \in \mathbb{R}^6$ is considered. All the manipulator's links are assumed to have length $l_i = 1\,\mathrm{m}$.

The case study considers three independent tasks arranged in priority according to the following table:

priority	task's description	task's dim.
a	end-effector position	2
b	end-effector orientation	1
c	robot intermediate position	2

this might represent, e.g., a simplified version of a macro-micro manipulator.

The first-task analytical function $\boldsymbol{\sigma}_{\mathrm{a}} \in \mathbb{R}^2$ and its Jacobian $\boldsymbol{J}_{\mathrm{a}} \in \mathbb{R}^{2\times 6}$ are given by:

$$\boldsymbol{\sigma}_{\mathrm{a}} = \begin{bmatrix} \displaystyle\sum_{i=1}^{6} l_i \cos\left(\sum_{j=1}^{i} q_j\right) \\ \displaystyle\sum_{i=1}^{6} l_i \sin\left(\sum_{j=1}^{i} q_j\right) \end{bmatrix}$$

and

$$\boldsymbol{J}_{\mathrm{a}} = \begin{bmatrix} \cdots & -\displaystyle\sum_{i=k}^{6} l_i \sin\left(\sum_{j=1}^{i} q_j\right) & \cdots \\ \cdots & \underbrace{\displaystyle\sum_{i=k}^{6} l_i \cos\left(\sum_{j=1}^{i} q_j\right)}_{k \text{ column}} & \cdots \end{bmatrix}$$

where k is a generic column of the matrix ranging from 1 to 6. The rank of $\boldsymbol{J}_{\mathrm{a}}$ is always $\rho(\boldsymbol{J}_{\mathrm{a}}) = 2$ except when the manipulator reaches a singular configuration given by an alignment of all the 6 joint positions. The possible occurrence of kinematic singularities will be handled by resorting to the damped least squares technique.

The second-task analytical function $\boldsymbol{\sigma}_{\mathrm{b}} \in \mathbb{R}$ and its Jacobian $\boldsymbol{J}_{\mathrm{b}} \in \mathbb{R}^{1\times 6}$ are given by:

$$\boldsymbol{\sigma}_{\mathrm{b}} = \sum_{i=1}^{n} q_i$$

and

$$\boldsymbol{J}_{\mathrm{b}} = [\,\cdots \quad 1 \quad \cdots\,]$$

whose rank is always full: $\rho(\boldsymbol{J}_{\mathrm{b}}) = 1$.

The third-task analytical function $\boldsymbol{\sigma}_{\mathrm{c}} \in \mathbb{R}^2$ is given by

$$\boldsymbol{\sigma}_{\mathrm{c}} = \begin{bmatrix} \displaystyle\sum_{i=1}^{2} l_i \cos\left(\sum_{j=1}^{i} q_j\right) \\ \displaystyle\sum_{i=1}^{2} l_i \sin\left(\sum_{j=1}^{i} q_j\right) \end{bmatrix}$$

its Jacobian $\boldsymbol{J}_{\mathrm{c}} \in \mathbb{R}^{2\times 6}$ can be easily derived and it is omitted for seek of brevity.

According to the stability analysis performed in [7], [8], with a proper choice of the gains, the convergence to zero of the tasks errors is guaranteed for equation (6), moreover, according to the results presented in this paper, even the *saturated* version of eq. (6), i.e., eq. (9), guarantees convergence to zero of the tasks errors.

For the three tasks the following desired values are commanded:

$$\begin{cases} \boldsymbol{\sigma}_{\mathrm{a,des}} = [\,3 \quad 2\,]^{\mathrm{T}} \;\; [\mathrm{m}] \\ \boldsymbol{\sigma}_{\mathrm{b,des}} = \pi/6 \;\; [\mathrm{rad}] \\ \boldsymbol{\sigma}_{\mathrm{c,des}} = [\,1 \quad 1\,]^{\mathrm{T}} \;\; [\mathrm{m}] \end{cases} \tag{34}$$

with gains:

$$\begin{aligned} \lambda_{\mathrm{a}} &= 50 \\ \lambda_{\mathrm{b}} &= 200 \\ \lambda_{\mathrm{c}} &= 100 \end{aligned} \tag{35}$$

Three different simulations have been presented discretizing the inverse kinematics algorithm with a sampling time $T = 1\,\mathrm{ms}$, namely without any kind of saturation, with an *uncontrolled* saturation at the joints and finally with the smart saturation proposed in this paper.

Figure 1 reports the time history of norm of the tasks' errors by resorting to the eq. (6) and without any velocity saturation. It can be observed that the primary task error converges monotonically to zero, coherently with the theoretical analysis, the second and third task errors do not have a negative time derivative during all of the transient, but they finally converge to zero as well.

The joint velocities are not saturated and thus they reach very large values in the initial transient. Figure 2 shows a zoom in the first time instants of the 6 velocity components.

Let us now apply a saturation of

$$\dot{q}_{\max} = 10 \quad \frac{\mathrm{rad}}{\mathrm{s}}$$

on the scalar components of the joint velocity vector; this corresponds to a value of $\underline{c}_{\mathrm{a}j} = -10\,\mathrm{rad/s}$ and $\overline{c}_{\mathrm{a}j} = 10\,\mathrm{rad/s}$. Figure 3 and 4 report the errors and the velocities,

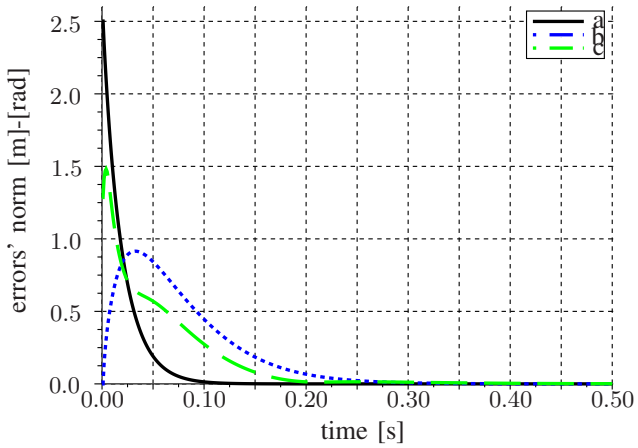

Fig. 1. Norm of the tasks errors by applying eq. (6) without any velocity saturation.

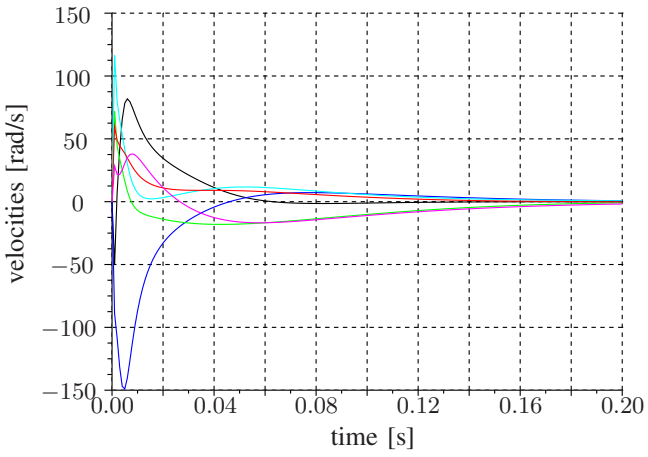

Fig. 2. Joint velocities by applying eq. (6) without any velocity saturation.

respectively. It can be noticed that there isn't any analytical guarantee that the task errors converge to zero, in fact, even the primary task, during the initial transient, experiences a positive time derivative meaning that its priority is not satisfied.

Figure 5 reports the norm of the tasks errors for the saturated inverse kinematics algorithm proposed in this paper, i.e., eq. (9). It is worth noticing that all the tasks errors converge to zero, the primary with a monotonically decreasing shape, the second and third task need to *wait* that the respective scaling factors α_* increase their values and finally converge to zero. The time history of the scaling factors α_* is given in Figure 6. The comparison of figures 3 and 5 reveals that the proposed control method guarantees (as expected) a faster execution of the highest priority task with respect to what would occur without a proper handling of the joint speed saturations.

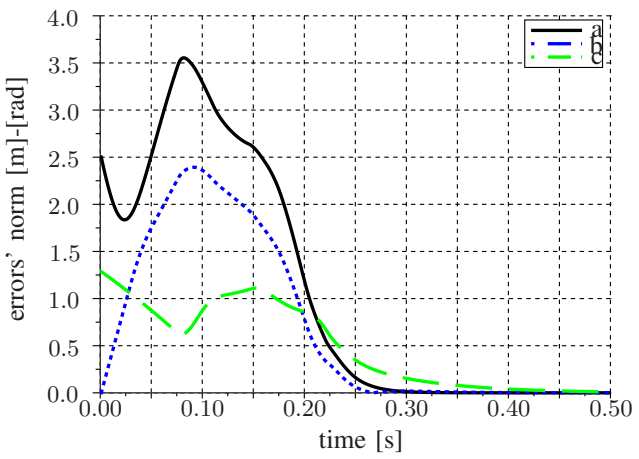

Fig. 3. Norm of the tasks errors by applying eq. (6) with a velocity saturation on the single joints components.

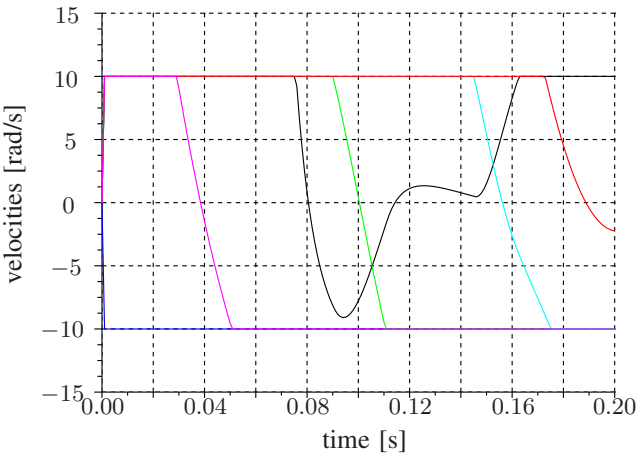

Fig. 4. Joint velocities by applying eq. (6) with a velocity saturation on the single joints components.

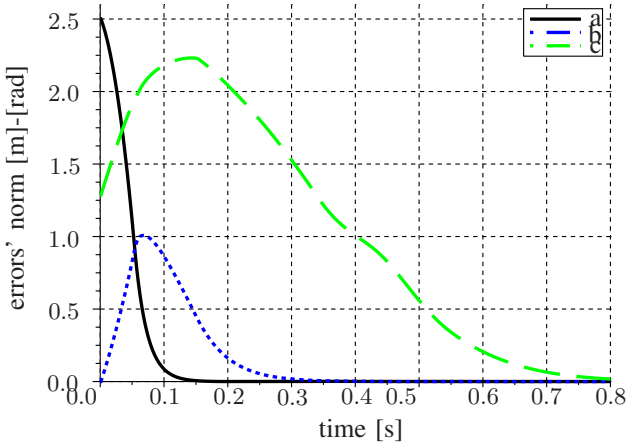

Fig. 5. Norm of the tasks errors by applying eq. (9) with the velocity saturation proposed in this paper.

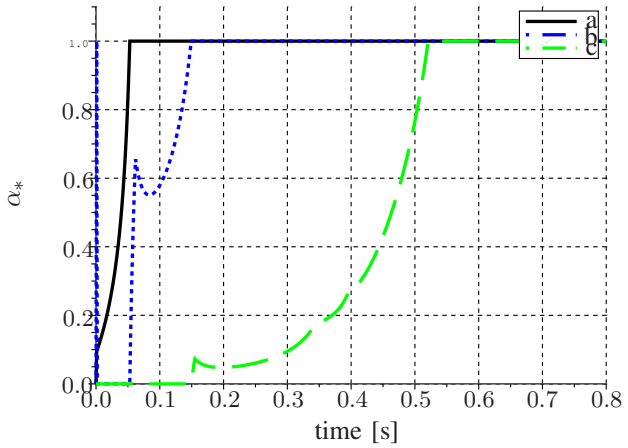

Fig. 6. Scaling factors.

Figure 7 reports the joint velocities by applying eq. (9) with the velocity saturation proposed in this paper, it can be noticed that the velocities bounds are satisfied.

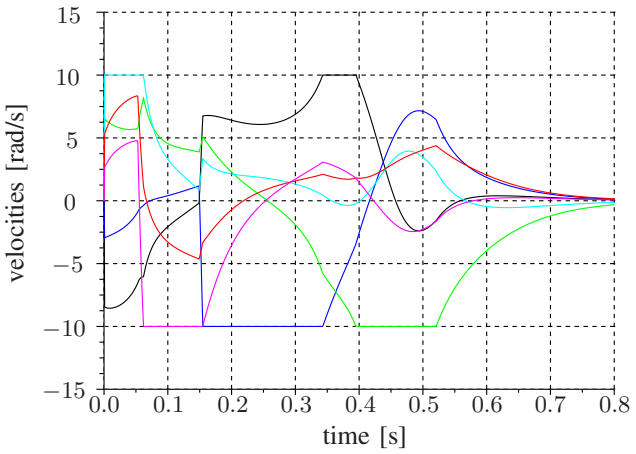

Fig. 7. Joint velocities by applying eq. (9) with the velocity saturation proposed in this paper.

V. Conclusions

A novel inverse kinematics control strategy is presented to command prioritized tasks in the presence of joint velocity saturations. Standard solutions based on similar null-space projection methods without taking care of velocity saturations may exhibit poor performance if the commanded joint speeds exceed the feasible bounds. A common work around of this issue consists in scaling the commanded speed so that the infinity norm of $\dot{\boldsymbol{q}}_{\text{des}}$ is compatible with the maximum joint speed: the drawback of this solution is that eventually the execution of higher priority tasks is slowed down due to the presence of lower priority ones that demand extra joint speed. In short, lower priority tasks may end up "disturbing" (i.e. slowing down) higher priority ones. The proposed solution, instead, guarantees that if the total commanded speed exceeds the maximum possible upper bound, only the lower priority tasks are slowed down to a value allowing the higher priority ones to be accomplished first.

Although the effectiveness of the proposed solution was demonstrated for 3 tasks only (for the sake of simplicity), the extension to N tasks is straightforward. Moreover it can be easily shown that the use of the transpose instead of pseudoinverse gives similar results.

Building on previous results, it was shown that under suitable hypothesis the proposed solution guarantees stability and convergence of the task errors to zero.

References

[1] A. Liégeois, "Automatic supervisory control of the configuration and behavior of muldibody mechanisms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7, pp. 868–871, 1977.

[2] A. Maciejewski and C. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *The International Journal of Robotics Research*, vol. 4, no. 3, pp. 109–117, 1985.

[3] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *The International Journal Robotics Research*, vol. 6, no. 2, pp. 3–15, 1987.

[4] B. Siciliano and J.-J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *Proceedings 5th International Conference on Advanced Robotics*, Pisa, I, 1991, pp. 1211–1216.

[5] J. Park, Y. Choi, W. Chung, and Y. Youm, "Multiple tasks kinematics using weighted pseudo-inverse for kinematically redundant manipulators," in *Proceedings 2001 IEEE International Conference on Robotics and Automation*, Seoul, KR, May 2001, pp. 4041–4047.

[6] P. Chiacchio, S. Chiaverini, L. Sciavicco, and B. Siciliano, "Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy," *The International Journal Robotics Research*, vol. 10, no. 4, pp. 410–425, 1991.

[7] G. Antonelli, "Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems," in *Proceedings 2008 IEEE International Conference on Robotics and Automation*, Pasadena, CA, May 2008, pp. 1993–1998.

[8] ——, "Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems," *accepted for publication IEEE Transactions on Robotics*, 2009.

[9] P. G. Ploeger, G. Indiveri, and J. Paulus, "Task based kinematical robot control in the presence of velocity saturation and its application to trajectory tracking for an omni-wheeled mobile robot," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Rome, I, Apr. 2007.

[10] B. Siciliano, "Kinematic control of redundant robot manipulators: A tutorial," *Journal of Intelligent Robotic Systems*, vol. 3, pp. 201–212, 1990.

[11] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 3, pp. 398–410, 1997.

[12] N. Mansard and F. Chaumette, "Tasks sequencing for visual servoing," in *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, J, Sept. 2004.

[13] ——, "Task sequencing for high-level sensor-based control," *IEEE Transactions on Robotics and Automation*, vol. 23, no. 1, pp. 60–72, 2007.