

Priority Oriented Adaptive Control of Kinematically Redundant Manipulators

Hamid Sadeghian, Mehdi Keshmiri, Luigi Villani and Bruno Siciliano

Abstract—In this paper an adaptive multi-priority nonlinear control algorithm for a redundant manipulator system is developed based on the Lyapunov like approach. The method considers the parametric uncertainties in the system and defines a proper filtered error signal to achieve asymptotic stability and convergence in tracking error, both for the main task and sub-tasks according to the allocated priority. The performance of the proposed method is studied by some numerical simulations.

I. INTRODUCTION

KINEMATICALLY redundant robots have more degrees of freedom than those necessary to achieve a desired task. The redundant degrees of freedom can be conveniently used to perform some additional tasks besides the main task. These additional tasks can be a defined performance objective or any secondary task as somehow similar to the main task. For example a given Cartesian position for a point on the body of robot different from its end-effector can be considered as the additional task.

There are plenty of papers that deal with how to use redundancy effectively to optimize some performance objective besides the main task control. This optimization is usually performed in the null-space of the main task to ensure its perfect tracking. In order to solve the conflict between tasks in a case where several objective functions are going to be satisfied simultaneously, the so-called task priority strategy developed in [1,2] is adopted. The formulation has later been extended to a general framework for managing multiple tasks by Siciliano and Slotine [3]. Their formulation uses first-order differential kinematic equation and solves redundancy in the Least-Squares (LS) sense, based on the assigned priority by resorting to pseudo-inverse solution. Because of using the pseudo-inverse of the *projected Jacobians* –the Jacobians of the lower-priority tasks that are projected into the null-space of the higher-

priority tasks– the formulation may suffer from high velocity norms during transition into and out of algorithmic singularities.

Usually a singularity-robust pseudo-inverse that allows limiting joint velocities at the expense of small tracking error in lower priority tasks is the first remedy to cope with this problem. Efficient damping techniques have been suggested by Nakamura and Hanafusa [4] and Wampler [5] and also by Nenchev and Sotirov [6] for the case of multiple priorities.

Chiaverini [7] proposed the singularity-robust task-priority resolution without using the projected Jacobian. This formulation always involves tracking errors in the additional tasks but singularities do not occur as long as the Jacobian of each additional task is full rank. The stability of this formulation has been shown in [8].

In contrast to velocity-based control approach, The acceleration-based control approach computes the desired joint accelerations for a given tasks [9,10,11,12]. Synthesis of joint accelerations in a redundant robot usually requires a more involved analysis. However, for second-order systems such as robotic systems this formulation is the most natural one that offers improved tracking ability due to the explicit incorporation of the acceleration information.

Adaptive nonlinear tracking control of kinematically redundant manipulators has been presented in [13] by Zengeroglu et al and later in [14] by Tatlicioglu et al. They used a Lyapunov approach to design a controller that achieves asymptotic tracking in the task space with systematic integration of the sub-task control into the stability analysis.

In this paper a nonlinear controller for the case of multiple tasks control based on the allocated priority is designed. Usually in a multi-priority control algorithm the joint space trajectories are obtained based on the inverse kinematics formulation, considering the priority orders. This joint space trajectory is then tracked by an inner control loop. A resolved acceleration or an inverse dynamics controller is typically adopted to track these joint space trajectories [15]. The resolution which is performed at velocity or acceleration level is usually based on the kinematics of the manipulator and thus is model dependent. It is clear that in the case of parametric uncertainty, the joint space trajectories obtained are no longer accurate. In addition to this deficiency, the inverse dynamics control method which is usually used for the joint space trajectory tracking is also model dependent. This, itself, doubles the system errors.

B. Siciliano and L. Villani (emails: {bruno.siciliano,luigi.villani}@unina.it) are with PRISMA Lab, Dipartimento di Informatica e Sistemistica, Università di Napoli Federico II, Italy.

H. Sadeghian is a Ph.D. candidate at Isfahan University of Technology, and currently a visiting scholar at PRISMA Lab, Dipartimento di Informatica e Sistemistica, Università di Napoli Federico II, Italy. (e-mail: h.sadeghian@me.iut.ac.ir).

M. Keshmiri is with the Mechanical Engineering Department, Isfahan University of Technology, Isfahan, Iran (e-mail: mehdik@cc.iut.ac.ir)

This work was partially funded by the Italian Ministry of University and Research within the PRIN 2009 20094WTJ29_003 ROCOCO' project and by the European Community, within the FP7 ICT-287513 SAPHARI project.

Since the controller proposed in this paper takes explicitly into account the robot's dynamic and is based on Lyapunov analysis, it has more flexibility and can be easily redesigned as an adaptive controller to compensate for dynamic parametric uncertainties.

II. ADAPTIVE CONTROL IN THE CARTESIAN SPACE

The dynamic model for an n -link robot manipulator can be expressed by

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \mathbf{F}_f\dot{\mathbf{q}} = \boldsymbol{\tau}, \quad (1)$$

where $\mathbf{q} \in \mathbb{R}^n$ is the vector of joint variable, $\mathbf{M}(\mathbf{q})$ is the inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ contains Coriolis, centrifugal terms, $\mathbf{g}(\mathbf{q})$ is the vector of gravitational torques and $\mathbf{F}_f$ is the constant, diagonal, friction matrix.

Equation (1) possesses some interesting properties for control purposes.

Property 1:

The inertia matrix $\mathbf{M}(\mathbf{q})$ is symmetric, positive definite and there exist positive scalars m_1 and m_2 that satisfy the following inequality

$$m_1 \|\mathbf{x}\|^2 \leq \mathbf{x}^T \mathbf{M}(\mathbf{q}) \mathbf{x} \leq m_2 \|\mathbf{x}\|^2, \quad \forall \mathbf{x} \in \mathbb{R}^n. \quad (2)$$

Property 2:

The centrifugal and Coriolis force matrix $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$, can be chosen in such a way that $(\dot{\mathbf{M}} - 2\mathbf{C})$ is skew symmetric, i.e. for any nonzero vector, $\boldsymbol{\xi} \in \mathbb{R}^n$, $\boldsymbol{\xi}^T (\dot{\mathbf{M}} - 2\mathbf{C}) \boldsymbol{\xi} = 0$.

Property 3:

The left hand side of the robot dynamics (1), can be linearly parameterized as follows

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \mathbf{F}_f\dot{\mathbf{q}} = \mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\mathbf{a}, \quad (3)$$

where $\mathbf{a}$ contains constant parameters and the regression matrix $\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ is bounded for bounded argument signals.

Adaptive control in the Cartesian space is similar to that in joint space one and originally was proposed in [16]. In detail, for a non-redundant manipulator, the reference trajectory is introduced by

$$\dot{\mathbf{q}}_r = \mathbf{J}^{-1}[\dot{\mathbf{x}}_d + \mathbf{A}(\mathbf{x}_d - \mathbf{x})], \quad (4)$$

where $\mathbf{J}(\mathbf{q})$ is the Jacobean matrix and $\mathbf{A}$ is a constant positive definite matrix. By defining

$$\mathbf{s} = \dot{\mathbf{q}}_r - \dot{\mathbf{q}} = \mathbf{J}^{-1}[(\dot{\mathbf{x}}_d - \dot{\mathbf{x}}) + \mathbf{A}(\mathbf{x}_d - \mathbf{x})], \quad (5)$$

and the regression matrix $\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}_r)$ and parameters vector $\mathbf{a}$ as

$$\mathbf{Y}\mathbf{a} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_r + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_r + \mathbf{g}(\mathbf{q}) + \mathbf{F}_f\dot{\mathbf{q}}, \quad (6)$$

one can come up with the following control and adaptation laws

$$\begin{aligned} \boldsymbol{\tau} &= \mathbf{Y}\hat{\mathbf{a}} + \mathbf{K}\mathbf{s} \\ \dot{\hat{\mathbf{a}}} &= \boldsymbol{\Gamma}\mathbf{Y}^T\mathbf{s}. \end{aligned} \quad (7)$$

where $\mathbf{K}$ and $\boldsymbol{\Gamma}$ are symmetric positive definite matrices. More details about the above adaptive algorithm can be found in [16,17,18].

III. MULTI-PRIORITY CONTROL; ERROR FORMULATION

Multi-priority inverse kinematics is a well-established framework to manage the tasks in a kinematically redundant robotic system. Assume that the robot mission is composed of two prioritized tasks. The first-priority task (main task) is specified by the first manipulation variable $\mathbf{x}_1 \in \mathbb{R}^{m_1}$ and the second-priority task (sub-task) by $\mathbf{x}_2 \in \mathbb{R}^{m_2}$. The kinematic relationships between the joint vector $\mathbf{q} \in \mathbb{R}^n$ and the vectors of task variables are expressed by

$$\dot{\mathbf{x}}_i = \mathbf{J}_i(\mathbf{q})\dot{\mathbf{q}}, \quad (i=1,2) \quad (8)$$

where $\mathbf{J}_i(\mathbf{q}) \in \mathbb{R}^{m_i \times n}$ is the Jacobian matrix of the i -th task. The inverse kinematics solution considering the priority of the main task over the sub-task is given by

$$\dot{\mathbf{q}} = \mathbf{J}_1^+ \dot{\mathbf{x}}_{1c} + \mathbf{N}_1 \bar{\mathbf{J}}_2^+ (\dot{\mathbf{x}}_{2c} - \mathbf{J}_2 \mathbf{J}_1^+ (\mathbf{q}) \dot{\mathbf{x}}_{1c}) + \mathbf{N}_2 \boldsymbol{\eta}, \quad (9)$$

where $(\cdot)^+$ is the pseudo-inverse of the related matrix. $\bar{\mathbf{J}}_2 = \mathbf{J}_2 \mathbf{N}_1$ is the projected Jacobian, which gives the available range for the sub-task to be executed without affecting the main task, $\mathbf{N}_1 = (\mathbf{I} - \mathbf{J}_1^+ \mathbf{J}_1)$, $\mathbf{N}_2 = \mathbf{N}_1 (\mathbf{I} - \bar{\mathbf{J}}_2^+ \bar{\mathbf{J}}_2)$ and $\boldsymbol{\eta}$ is an arbitrary vector [2].

In (9) $\dot{\mathbf{x}}_{1c}$ and $\dot{\mathbf{x}}_{2c}$ are the command velocities for the main task and sub-task respectively. They can be given as $\dot{\mathbf{x}}_{ic} = \dot{\mathbf{x}}_{id} + \mathbf{A}_i(\mathbf{x}_{id} - \mathbf{x}_i)$, where $\mathbf{A}_i$ are appropriate positive definite matrix gains and index d stands for the desired trajectory. Also notice that identity $\mathbf{N}_1(\bar{\mathbf{J}}_2^+) = \bar{\mathbf{J}}_2^+$, can be used in (9) for further reduction.

A recursive extension of (9) for k prioritized tasks was proposed in [3] and is given by

$$\begin{aligned} \dot{\mathbf{q}}_c^{(i)} &= \dot{\mathbf{q}}_c^{(i-1)} + \mathbf{N}_{i-1} \bar{\mathbf{J}}_i^+ (\dot{\mathbf{x}}_{ic} - \mathbf{J}_i \dot{\mathbf{q}}_c^{(i-1)}) + \mathbf{N}_i \boldsymbol{\eta}, \quad i=1, \dots, k \\ \bar{\mathbf{J}}_i &= \mathbf{J}_i \mathbf{N}_{i-1} \\ \mathbf{N}_i &= \mathbf{I} - \bar{\mathbf{J}}_i^+ \bar{\mathbf{J}}_i, \\ \mathbf{N}_0 &= \mathbf{I}, \quad \dot{\mathbf{q}}_c^{(0)} = \mathbf{0}, \end{aligned} \quad (10)$$

By the above resolution, the main task is performed completely. Indeed by multiplying both sides of (9) by $\mathbf{J}_1$, using (8) and defining $\tilde{\mathbf{x}}_1(t) = \mathbf{x}_{1d} - \mathbf{x}_1$, the closed-loop error dynamics for the main task is obtained as

$$\dot{\mathbf{x}}_1 = \dot{\mathbf{x}}_{1c} \Rightarrow \dot{\tilde{\mathbf{x}}}_1 + \mathbf{A}_1 \tilde{\mathbf{x}}_1 = \mathbf{0}. \quad (11)$$

Multiplying both sides of (9) and generally (10) by $\mathbf{J}_i$ and further by $\bar{\mathbf{J}}_i \bar{\mathbf{J}}_i^+$ and using its idempotency property, the closed-loop error dynamics for the sub-task i is given by

$$\bar{\mathbf{J}}_i \bar{\mathbf{J}}_i^+ (\dot{\tilde{\mathbf{x}}}_i + \mathbf{A}_i \tilde{\mathbf{x}}_i) = \mathbf{0}, \quad \tilde{\mathbf{x}}_i(t) = \mathbf{x}_{id} - \mathbf{x}_i, \quad i=2, \dots, k. \quad (12)$$

Thus it is clear from (12), that in the case of independent tasks, $\bar{\mathbf{J}}_i \bar{\mathbf{J}}_i^+ = \mathbf{I}$ and the main task and sub-tasks i are performed completely. However, in the case of dependent

tasks instead, the solution (12) ensures minimization of $\|\dot{\tilde{\mathbf{x}}}_i + \mathbf{A}_i \tilde{\mathbf{x}}_i\|$ subject to all higher priority constraints.

In multi-priority algorithm, if $\mathbf{J}_i$ is singular (task singularity), the i -th task cannot be satisfied, regardless of all the other tasks. If $\bar{\mathbf{J}}_i$ is singular, without $\mathbf{J}_i$ being singular (algorithmic singularity), the i -th task cannot be satisfied given the previous $i-1$ tasks. This will happen when tasks are dependent. Namely, two generic tasks are dependent when

$$\rho(\mathbf{J}_1) + \rho(\mathbf{J}_2) > \rho\left(\begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix}\right), \quad (13)$$

where $\rho(\cdot)$ denotes rank of a matrix. For more details about task independency conditions and algorithmic singularities see [7, 8, 19].

It is obvious that singularities may be introduced by the lower-priority tasks. Near the singularity, a damped least square inverse can be adopted for the projected Jacobian. For a singularity-robust task-priority handling, Chiaverini [7] proposed the following formulation for the case with two tasks

$$\dot{\mathbf{q}} = \mathbf{J}_1^\dagger \dot{\mathbf{x}}_{1c} + \mathbf{N}_1 \mathbf{J}_2^\dagger \dot{\mathbf{x}}_{2c}, \quad (14)$$

Comparing this with (9), algorithmic singularities are absent in the cost of a greater tracking error for the sub-task [8,20].

IV. ADAPTIVE MULTI-PRIORITY CONTROL

In this section, initially we design a nonlinear multi-priority control. Then we illustrate how this controller can be used as an adaptive controller for the case of parameter uncertainties.

The goal is to find an adaptive control algorithm which ensures asymptotically stable closed-loop error dynamics for the main task as well as all sub-tasks. To this end and based on (9) a filtered error signal $\mathbf{s}$ is defined as follows

$$\mathbf{s} = \dot{\mathbf{q}}_r - \dot{\mathbf{q}}, \quad (15)$$

where

$$\dot{\mathbf{q}}_r = \mathbf{J}_1^\dagger \dot{\mathbf{x}}_{1c} + \mathbf{N}_1 \bar{\mathbf{J}}_2^\dagger (\dot{\mathbf{x}}_{2c} - \mathbf{J}_2 \mathbf{J}_1^\dagger (\mathbf{q}) \dot{\mathbf{x}}_{1c}). \quad (16)$$

Note that based on this definition, the main task error dynamics is obtained as

$$\mathbf{J}_1 \mathbf{s} = \dot{\mathbf{x}}_{1c} - \mathbf{J}_1 \dot{\mathbf{q}} = \dot{\tilde{\mathbf{x}}}_1 + \mathbf{A}_1 \tilde{\mathbf{x}}_1. \quad (17)$$

Multiplying both sides of (16) by $\mathbf{J}_2$ gives

$$\mathbf{J}_2 \mathbf{s} = \mathbf{J}_2 \mathbf{J}_1^\dagger \dot{\mathbf{x}}_{1c} + \bar{\mathbf{J}}_2 \bar{\mathbf{J}}_2^\dagger (\dot{\mathbf{x}}_{2c} - \mathbf{J}_2 \mathbf{J}_1^\dagger (\mathbf{q}) \dot{\mathbf{x}}_{1c}) - \mathbf{J}_2 \dot{\mathbf{q}}. \quad (18)$$

Further, by multiplication of both sides by $\bar{\mathbf{J}}_2 \bar{\mathbf{J}}_2^\dagger$, the error dynamics for the sub-task is given by

$$\bar{\mathbf{J}}_2 \bar{\mathbf{J}}_2^\dagger \mathbf{J}_2 \mathbf{s} = \bar{\mathbf{J}}_2 \bar{\mathbf{J}}_2^\dagger (\dot{\tilde{\mathbf{x}}}_2 + \mathbf{A}_2 \tilde{\mathbf{x}}_2). \quad (19)$$

Generally in the case of k sub-tasks, $\dot{\mathbf{q}}_r$ used in (16) can be obtained from the recursive scheme (10) and the error dynamics for sub-task i is obtained as

$$\bar{\mathbf{J}}_i \bar{\mathbf{J}}_i^\dagger \mathbf{J}_i \mathbf{s} = \bar{\mathbf{J}}_i \bar{\mathbf{J}}_i^\dagger (\dot{\tilde{\mathbf{x}}}_i + \mathbf{A}_i \tilde{\mathbf{x}}_i), \quad i = 2, \dots, k. \quad (20)$$

The attractive property of definition (15) is that it contains the entire sub-task errors. In other word when $\mathbf{s}$ goes to zero all the projected sub-task errors defined based on (12) converge to zero as it can be seen from (20).

Based on the above error analysis, the following controller is introduced.

Theorem 1: Consider the robot manipulator dynamics presented in (1) with the following control law

$$\boldsymbol{\tau} = \mathbf{Y}\mathbf{a} + \mathbf{K}\mathbf{s} + k_1 \mathbf{J}^T \tilde{\mathbf{x}}_1, \quad (21)$$

where k_1 is a positive gain, $\mathbf{K}$ is a symmetric positive definite matrix and $\mathbf{Y}\mathbf{a}$ based on property (3) is given by

$$\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}_r) \mathbf{a} = \mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}}_r + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \dot{\mathbf{q}}_r + \mathbf{g}(\mathbf{q}) + \mathbf{F}_f \dot{\mathbf{q}}. \quad (22)$$

Tracking errors of the main task and all sub-tasks based on the error definition (12), asymptotically converge to zero.

Proof: see the proof of theorem 2.

Theorem 2. In the case of parametric uncertainties the control law and adaptation law given by

$$\begin{aligned} \boldsymbol{\tau} &= \mathbf{Y}\hat{\mathbf{a}} + \mathbf{K}\mathbf{s} + k_1 \mathbf{J}_1^T \tilde{\mathbf{x}}_1, \\ \dot{\hat{\mathbf{a}}} &= \boldsymbol{\Gamma} \mathbf{Y}^T \mathbf{s}, \end{aligned} \quad (23)$$

where $\mathbf{K}$ and $\boldsymbol{\Gamma}$ are symmetric positive definite matrices, and

$$\mathbf{Y}\hat{\mathbf{a}} = \hat{\mathbf{M}}(\mathbf{q}) \ddot{\mathbf{q}}_r + \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}) \dot{\mathbf{q}}_r + \hat{\mathbf{g}}(\mathbf{q}) + \hat{\mathbf{F}}_f \dot{\mathbf{q}}, \quad (24)$$

guarantee asymptotic minimization of the main task and all the sub-task tracking errors in the sense of (12).

Proof: The manipulator open loop dynamics (1) can be rewritten in a useful form, based on (15) and (22) as

$$\mathbf{Y}\mathbf{a} - \mathbf{M}\dot{\mathbf{s}} - \mathbf{C}\mathbf{s} = \boldsymbol{\tau}. \quad (25)$$

Upon substitution of the proposed controller into (25), the closed-loop system equations can be obtained as

$$\begin{aligned} \mathbf{M}\dot{\mathbf{s}} &= \mathbf{Y}\hat{\mathbf{a}} - (\mathbf{C} + \mathbf{K})\mathbf{s} - k_1 \mathbf{J}_1^T \tilde{\mathbf{x}}_1, \\ \dot{\hat{\mathbf{a}}} &= \boldsymbol{\Gamma} \mathbf{Y}^T \mathbf{s}, \end{aligned} \quad (26)$$

where $\tilde{\mathbf{a}} = \mathbf{a} - \hat{\mathbf{a}}$. Equations (26) together with (17) and (20) constitute the closed-loop dynamic of the system. A non-negative function can be defined in terms of variables $\mathbf{x} = [\tilde{\mathbf{x}}_1^T, \mathbf{s}^T, \tilde{\mathbf{a}}^T]^T$ as

$$V(\mathbf{x}, t) = \frac{k_1}{2} \tilde{\mathbf{x}}_1^T \tilde{\mathbf{x}}_1 + \frac{1}{2} \mathbf{s}^T \mathbf{M} \mathbf{s} + \frac{1}{2} \tilde{\mathbf{a}}^T \boldsymbol{\Gamma}^{-1} \tilde{\mathbf{a}}, \quad (27)$$

where $\boldsymbol{\Gamma}$ is a constant positive definite diagonal matrix. Note that $\mathbf{M}(\mathbf{q})$ is a time-varying positive definite matrix.

Using property (2) it is clear that

$$\begin{aligned} V_0(\mathbf{x}) &\leq V(\mathbf{x}, t) \leq V_1(\mathbf{x}), \\ V_0(\mathbf{x}) &= \frac{k_1}{2} \tilde{\mathbf{x}}_1^T \tilde{\mathbf{x}}_1 + \frac{m_1}{2} \mathbf{s}^T \mathbf{s} + \frac{\gamma_1}{2} \tilde{\mathbf{a}}^T \tilde{\mathbf{a}}, \\ V_1(\mathbf{x}) &= \frac{k_1}{2} \tilde{\mathbf{x}}_1^T \tilde{\mathbf{x}}_1 + \frac{m_2}{2} \mathbf{s}^T \mathbf{s} + \frac{\gamma_2}{2} \tilde{\mathbf{a}}^T \tilde{\mathbf{a}}, \end{aligned} \quad (28)$$

where m_1, m_2 and γ_1, γ_2 denote minimum and maximum eigenvalues of $\mathbf{M}$ and $\boldsymbol{\Gamma}^{-1}$ respectively. Also note that $V(0, t) = 0$. Hence V satisfies the positive definiteness and descent conditions.

It is worth remarking that the nonnegative function given by (27) is a Lyapunov-like function candidate, because the vector variable $\mathbf{x}$ is not a complete set of state variables and there exist other variables $\tilde{\mathbf{x}}_i$ that are closely related to $\mathbf{s}$ by equation (20) and not considered in (27).

Time derivative of (27) along the system trajectories is given by

$$\begin{aligned} \dot{V} = & k_1 \tilde{\mathbf{x}}_1^T (J_1 \mathbf{s} - A_1 \tilde{\mathbf{x}}_1) + \mathbf{s}^T (\mathbf{Y} \tilde{\mathbf{a}} - (\mathbf{C} + \mathbf{K}) \mathbf{s} - k_1 J_1^T \tilde{\mathbf{x}}_1) \\ & + \frac{1}{2} \mathbf{s}^T \dot{\mathbf{M}} \mathbf{s} + \tilde{\mathbf{a}}^T \Gamma^{-1} \dot{\tilde{\mathbf{a}}}. \end{aligned} \quad (29)$$

Substituting the adaptation law (23) and assuming constant system parameters vector $\mathbf{a}$ yields

$$\dot{V} = -\tilde{\mathbf{x}}_1^T k_1 A_1 \tilde{\mathbf{x}}_1 - \mathbf{s}^T \mathbf{K} \mathbf{s} \leq 0, \quad (30)$$

where the skew-symmetry of $(\dot{\mathbf{M}} - 2\mathbf{C})$ has been taken into account.

Equations (27) and (30) indicate that $\tilde{\mathbf{x}}_1, \mathbf{s}$ and $\tilde{\mathbf{a}}$ are all bounded. The Barbalat's lemma can be used to check the uniform continuity of $\dot{V}$. Time derivative of $\dot{V}$ is

$$\ddot{V} = -2\tilde{\mathbf{x}}_1^T k_1 A_1 \dot{\tilde{\mathbf{x}}}_1 - 2\mathbf{s}^T \mathbf{K} \dot{\mathbf{s}}. \quad (31)$$

From the dynamics of the closed-loop system (as we will show in the following) $\dot{\mathbf{s}}, \dot{\tilde{\mathbf{x}}}_1$ and thus $\ddot{V}$ remains bounded. The application of Barbalat's lemma then indicates that $\dot{V} \rightarrow 0$ and consequently $\tilde{\mathbf{x}}_1, \mathbf{s} \rightarrow 0$ as $t \rightarrow \infty$.

Now we only have to show that $\dot{\mathbf{s}}, \dot{\tilde{\mathbf{x}}}_1$ remain bounded. From (17) it can be easily inferred that $\dot{\tilde{\mathbf{x}}}_1$ is bounded because of boundedness of $\mathbf{s}$ and $\tilde{\mathbf{x}}_1$. Furthermore since (16) resolve the redundancy of the system through first-order multi-priority inverse kinematics, the internal motion does not exist and thus $\dot{\mathbf{q}}_r$ is bounded. This in turn, implies that $\dot{\mathbf{q}}$ which is given by $\dot{\mathbf{q}} = \dot{\mathbf{q}}_r - \mathbf{s}$ and also $\dot{\mathbf{q}}_r$ are bounded. Notice that we implicitly assume that $\mathbf{x}_{id}(t)$ and its derivatives are bounded and the Jacobians are also bounded for all possible $\mathbf{q}(t)$. Finally by the closed-loop dynamic (26) $\dot{\mathbf{s}}$ is bounded. Therefore all the projected sub-task errors defined based on (12) converge to zero as it can be seen from (20).

Yet the problem that may seem to exist is the self motion of redundant manipulator and boundedness of $\mathbf{q}(t)$ especially in the case that we don't have enough sub-tasks. However, as pointed out above since redundancy is resolved at velocity level, internal instability does not occur. Nevertheless, let us assume the case that we don't have any sub-tasks. The error signal $\mathbf{s}$ is given by (15) and

$$\dot{\mathbf{q}}_r = J_1^+ \dot{\mathbf{x}}_{ic} + N_1 \dot{\mathbf{q}}_n, \quad (32)$$

where $\dot{\mathbf{q}}_n$ is null-space velocity. Substituting (32) in (15) and multiplying both sides by N_1 , the null-space error dynamics is obtained as

$$N_1 \mathbf{s} = N_1 (\dot{\mathbf{q}}_n - \dot{\mathbf{q}}) \quad (33)$$

For the case where $\dot{\mathbf{q}}_n = 0$ it can be seen that null-space

velocity converges to zero as $\mathbf{s} \rightarrow 0$.

V. NUMERICAL SIMULATION

Performance of the above control algorithm has been verified on a 4 DOF serial planar manipulator. The numerical values of the manipulator parameters are as follows

$$\begin{aligned} l_1 = 0.5, l_2 = 0.4, l_3 = 0.35, l_4 = 0.2 \text{ (m)}, \\ m_1 = 1.4, m_2 = 1.3, m_3 = 1.2, m_4 = 1.1 \text{ (kg)}, \\ I_1 = 0.5, I_2 = 0.45, I_3 = 0.4, I_4 = 0.3 \text{ (kg.m}^2\text{)}, \\ c_1 = 1.7, c_2 = 1.6, c_3 = 1.5, c_4 = 1.4 \text{ (Nm.sec)}. \end{aligned} \quad (34)$$

The mass centre of each link, r_i , is assumed to be in the middle of each one. The desired trajectory for the main task is selected as follows

$$\mathbf{x}_d(t) = [\sin(0.25t) + 0.2, \cos(0.25t) + 0.15]^T \quad (35)$$

The initial condition used in all simulations is $\mathbf{q}_0 = [0.8727, 1.7, -1.6, 0.85]^T$.

This system with the above operational task creates a redundant system with two degrees of redundancy. For our purpose of multi-priority control two sub-tasks are defined with allocated priority. The orientation of the last link and angular position of the first link are chosen as the second and third priority sub-tasks respectively. The desired orientation for the first sub-task is considered in such a way that last link to be perpendicular to the main task trajectory. The desired value for the third task is assumed to be a constant joint position, $x_{3d} = 35^\circ$. The arm configuration during the operation and the desired and actual trajectories are depicted in Fig 1.

As the first case study we assume no uncertainty in the system and the nonlinear controller (21) is applied to the system. The results are provided in Fig 2.

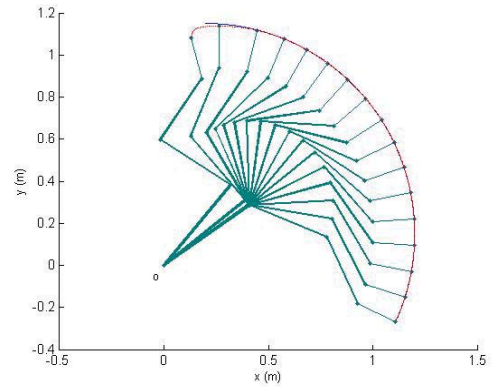


Fig 1. Four DOF planar robot arm configuration

One can obtain similar results if the feedback linearization controller is used. These results are not presented here for the sake of shortness. As it can be seen, for this case where tasks are independent, both main task and sub-tasks are

performed successfully.

In the second case study, the capability of adaptive controller (23) is studied. Here we assume uncertainty in dynamic parameters (i.e. mass and inertia of the links, position of mass center of each link and friction coefficients). The system parameters vector $\mathbf{a}$ includes 18 parameters and is given by

$$\mathbf{a} = [m_1 r_1^2, I_1, I_2, I_3, I_4, m_2, m_2 r_2, m_2 r_2^2, m_3, m_3 r_3, m_3 r_3^2, m_4, m_4 r_4, m_4 r_4^2, c_1, c_2, c_3, c_4]^T \quad (36)$$

The controller parameters including $\mathbf{\Gamma}$ for the adaptive control case, are chosen as follows

$$\begin{aligned} \mathbf{\Gamma} &= \text{diag}(0.1, 0.2, 0.2, 0.2, 0.2, 0.2, 0.2, 0.1, \\ &\quad 0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 30, 30, 20, 10), \\ \mathbf{K} &= 8\mathbf{I}, \\ \mathbf{A}_1 &= 5\mathbf{I}, \mathbf{A}_2 = 4\mathbf{I}, \mathbf{A}_3 = 4\mathbf{I}. \end{aligned} \quad (37)$$

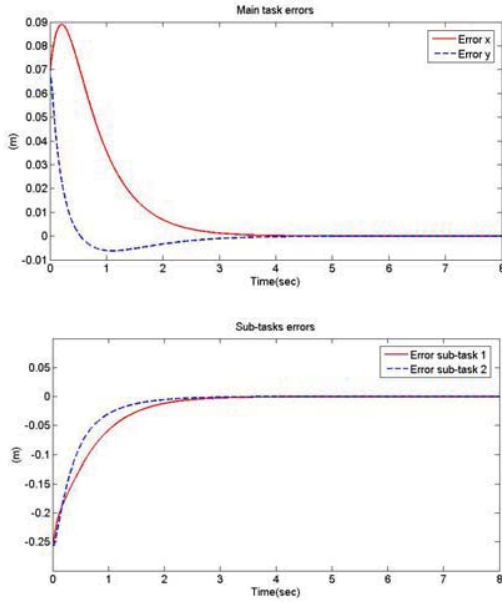


Fig 2. Main task error and sub-tasks errors

To show the performance of adaptive controller (23) the simulation first is performed by the controller (21) with 50% uncertainty in system parameters vector (i.e. $\hat{\mathbf{a}}(0) = 0.5\mathbf{a}$). The corresponding tracking errors for the main task and sub-tasks are plotted in Fig 3. As we can see this controller is somehow robust encountering with parametric uncertainty. In the next simulation adaptive controller (23) is used with the same controller gains as the previous analysis, starting with 50% a priori information about the system parameters. The result is illustrated in Fig 4.

The estimated parameters vector $\mathbf{a}$ is shown in Fig 5 for the case of adaptive controller.

Notice that the nonlinear controller also works in the case of dependent tasks. The only problem that exists is the transition into and out of singularities.

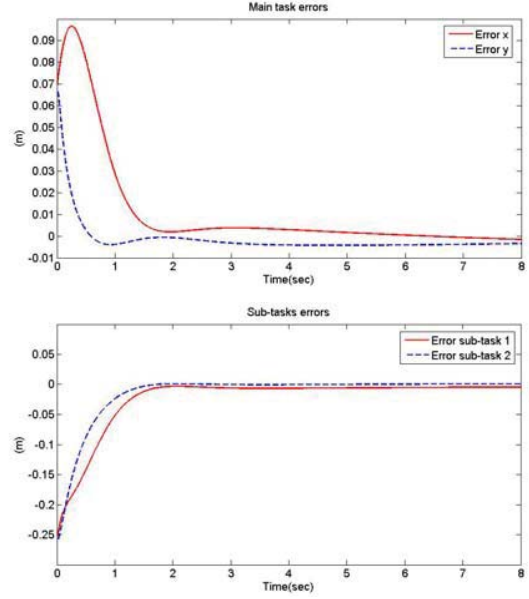


Fig 3. Main task error and sub-tasks errors, using controller (21) with 50% uncertainty

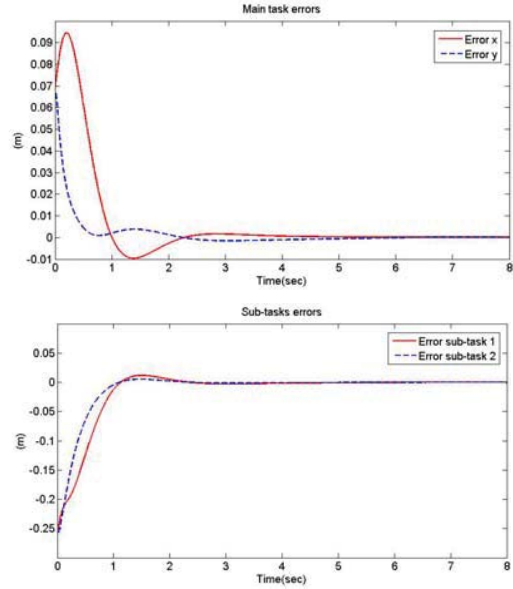


Fig 4. Main task error and sub-tasks errors under adaptive control

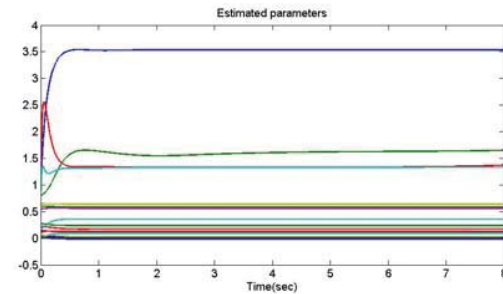


Fig 5. Parameters estimation under adaptive control

Since the redundancy is resolved at velocity level, damped pseudo-inverse $\bar{J}_i^{\dagger\lambda}$ can be easily used instead of $\bar{J}_i^{\dagger}$ in the reference joint trajectory $\dot{q}_r$. To see the performance of adaptive controller near singularity we continue the above simulation for 3 more seconds using damped pseudo-inverse with $\lambda = 0.1$ for the second sub-task. The result is shown in Fig 6 and Fig 7 and also in accompanying video. As we can see second sub-task is sacrificed in order to perform high priority tasks, the same that happens also by feedback linearization controller.

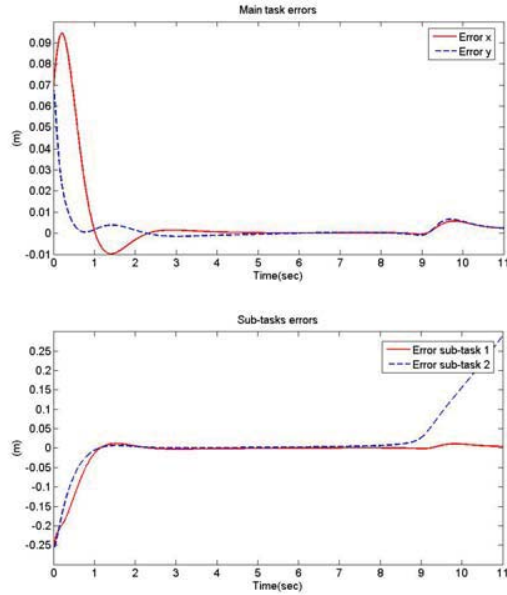


Fig 6. Main task error and sub-tasks errors under adaptive control and using damped pseudo-inverse in controller

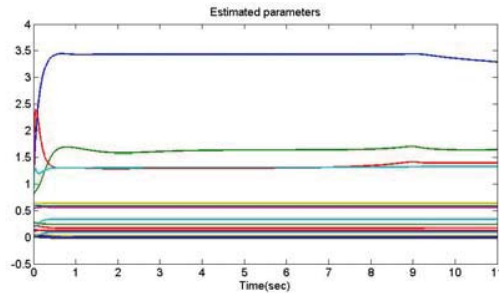


Fig 7. Parameters estimation under adaptive control and using damped pseudo-inverse in controller

VI. CONCLUSION

A nonlinear control algorithm for multi-priority task control has been presented in this paper. The flexibility provided by this approach allows a simple extension to an adaptive control algorithm by proper error definition. The provided simulations show that this controller can be effectively used instead of model based multi-priority

control algorithm.

REFERENCES

- [1] Y. Nakamura, H. Hanafusa and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *International Journal of Robotics Research*, 6(2), 3–15, 1987.
- [2] Y. Nakamura, *Advanced Robotics Redundancy and Optimization*, Addison-Wesley, 1991.
- [3] B. Siciliano and J.J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," *5th International Conference on Advanced Robotics*, 1991.
- [4] Y. Nakamura and H. Hanafusa, "Inverse kinematic solutions with singularity robustness for robot manipulator control," *ASME Journal of Dynamic Systems, Measurement and Control*, 108, 163–171, 1986.
- [5] C. W. Wampler II, "Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods," *IEEE Transactions on Systems, Man and Cybernetics*, 16, 93–101, 1986.
- [6] D. Nenchev and Z. M. Sotirov, "Dynamic task-priority allocation for kinematically redundant robotic mechanisms," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.
- [7] S. Chiaverini, "Singularity-robust task priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Transactions on Robotics and Automation*, 13, 398–410, 1997.
- [8] G. Antonelli, "Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems," *IEEE International Conference on Robotics and Automation*, 2008.
- [9] P. Hsu, J. Hauser and S. Sastry, "Dynamic control of redundant manipulators," *Journal of Robotic Systems*, 6, 133–148, 1989.
- [10] A. De Luca, G. Oriolo and B. Siciliano, "Robot redundancy resolution at the acceleration level," *Laboratory Robotics and Automation*, 4, 97–106, 1992.
- [11] J. Nakanishi, R. Cory, M.J. Peters and S. Schaal, "Operational space control: a theoretical and empirical comparison," *International Journal of Robotics Research*, 27, 737–757, 2008.
- [12] H. Sadeghian, L. Villani, M. Keshmiri and B. Siciliano, "Multi-priority control in redundant robotic systems," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.
- [13] E. Zengeroglu, D. Dawson, I. Walker and A. Behal, "Nonlinear tracking control of kinematically redundant robot manipulators," *Proceeding of American Control Conference*, 2513–2517, 2000.
- [14] E. Tatlicioglu, M.L. McIntyre, D.M. Dawson and I.D. Walker, "Adaptive nonlinear tracking control of kinematically redundant robot manipulators," *International Journal of Robotics and Automation*, 23(2), 98–105, 2008.
- [15] F. Caccavale, C. Natale, B. Siciliano and L. Villani, "Resolved acceleration control of robot manipulators: A critical review with experiments," *Robotica*, 16, 565–573, 1998.
- [16] J. Slotine and W. Li, "On the adaptive control of robot manipulators," *International Journal of Robotics Research*, 6(3), 49–59, 1987.
- [17] J.J. Slotine and W. Li, *Applied Nonlinear Control*, Prentice-Hall, 1991.
- [18] M. W. Spong, R. Ortega and R. Kelly, "Comments on adaptive control: a case study," *IEEE Transactions on Automatic Control*, 33, 761–762, 1990.
- [19] J. Park, Y. Choi, W. K. Chung and Y. Youm, "Multiple tasks kinematics using weighted pseudo-inverse for kinematically redundant manipulators," *IEEE International Conference on Robotics and Automation*, 2001.
- [20] G. Antonelli, F. Arrichiello and S. Chiaverini, "The null-space based behavioral control for autonomous robotic systems," *Intelligent Service Robotics*, 1, 27–39, 2008.