Singularity-Robust Task-Priority Redundancy Resolution for Real-Time Kinematic Control of Robot Manipulators

Stefano Chiaverini

***Abstract*—Practical application of the task-priority redundancy resolution technique must deal with the occurrence of kinematic and algorithmic singularities. The aim of this paper is twofold. First, the application of existing singularity-robust methods to the case of kinematically redundant arms is studied. Then, a new task-priority redundancy resolution technique is developed that overcomes the effects of algorithmic singularities. Computational aspects of the solutions are also considered in view of real-time implementation of a kinematic control algorithm. The method is applied to a seven-degree-of-freedom manipulator in numerical case studies to demonstrate its effectiveness.**

I. Introduction

KINEMATIC redundancy occurs when a manipulator possesses more degrees of freedom than those required to execute a given task. Thus, strictly speaking, do not exist redundant manipulators but rather tasks exist that make a manipulator redundant. Since it is widely recognized that a general task consists of following an end-effector motion trajectory requiring six degrees-of-freedom, a seven-joint robot arm is usually brought as an example of an inherently redundant manipulator. However, a number of tasks might be devised in which an arm with fewer joints, e.g., a conventional six-joint industrial manipulator, becomes itself redundant.

For a redundant manipulator the inverse kinematics problem admits an infinite number of solutions and a criterion to select one of them is needed. In the remainder we will consider the problem of redundancy resolution at the inverse differential kinematics level. In this framework, the simplest method is based on the use of the pseudoinverse of the Jacobian matrix [21]; this guarantees optimal reconstruction of the desired end-effector velocity—in a least-squares sense—with the minimum-norm joint velocity. One drawback of this method is that, despite of local minimization of joint velocities, singularity avoidance cannot be guaranteed [2].

More conveniently, redundancy can be exploited to meet additional constraints on the solution of the inverse kinematics problem besides the typical end-effector task.

A first possibility is to add a term for local optimization of a scalar criterion to the basic pseudoinverse joint velocity solution; accordingly, a term proportional to the gradient of the criterion is projected onto the null space of the Jacobian matrix so that the end-effector task is not affected [12]. Several criteria have been proposed, e.g., to achieve maximum joint range availability, obstacle avoidance, minimization of joint torques, maximization of different dexterity measures.

Another approach, the so-called task-space augmentation, introduces a constraint task to be fulfilled along with the end-effector task; then, an augmented Jacobian matrix is set-up whose inverse gives the sought joint velocity solution [8], [19]. The augmented Jacobian matrix, however, happens to be singular even in configurations at which the end-effector Jacobian is full-rank. Conceptually similar is the extended Jacobian technique, for which the occurrence of algorithmic singularities besides kinematic singularities was pointed out [1]. In both techniques, the additional singularities arise when the constraint task conflicts with the end-effector task. To overcome this type of singularities, either the constraint function must be keenly specified case-by-case (e.g., [3]) or singularity-robust techniques must be adopted to invert the augmented or the extended Jacobian matrix (e.g., [9]).

Conflicts between the end-effector task and the constraint task are handled in the framework of the task-priority strategy by suitably assigning an order of priority to the given tasks and then satisfying the lower priority task only in the null space of the higher priority task [13], [18]. In the typical case, the end-effector task is considered as the primary task although examples might be given in which it becomes the secondary task. It can be recognized that the problem of algorithmic singularities still remains but, differently from the task-space augmentation approach, correct primary-task solutions are expected as long as the sole primary-task Jacobian matrix is full-rank. On the other hand, out of the algorithmic singularities the task-priority strategy gives the same solution as the task-space augmentation approach; this implies that close to an algorithmic singularity the solution becomes ill-conditioned and large joint velocities may result. This problem has partly been solved by looking at the efficient rank of the matrix involved and treating it as singular depending on a suitable threshold [13]; the obtained joint velocities are thus limited, but continuity of the null-space component must be ensured.

In this paper we will focus our attention on the handling of singular configurations in the task-priority strategy. A first problem is related to the use of the pseudoinverse of the end-effector Jacobian matrix that may give infeasible joint

Manuscript received May 6, 1996. This work was supported by the Ministero dell'Università e della Ricerca Scientifica e Tecnologica under 40% and 60% funds. This paper was recommended for publication by Associate Editor V. Hayward and Editor S. Salcudean upon evaluation of the reviewers' comments.

The author is with the Dipartimento di Informatica e Sistemistica, Università degli Studi di Napoli "Federico II", 80125 Naples, Italy (e-mail: chiaverini@disna.dis.unina.it).

Publisher Item Identifier S 1042-296X(97)03824-X.

1042–296X/97$10.00 © 1997 IEEE

velocities in the neighborhood of kinematic singularities [4]. The adoption of existing singularity-robust methods to the case of kinematically redundant arms is thus studied in terms of the error obtained when using the joint velocity solution to reconstruct the commanded end-effector velocity. Then, the problem of algorithmic singularities is considered and a new task-priority redundancy resolution technique is developed [6] that avoids the inversion of the matrix which is ill-conditioned when close to the algorithmic singularity. Computational aspects of the solutions are also discussed in view of real-time implementation of a kinematic control algorithm. A seven-degree-of-freedom manipulator is considered in numerical case studies to demonstrate practical application of the method and its effectiveness.

II. Background

For a given manipulator, the mapping

$$\dot{\mathbf{x}}_E = \mathbf{J}_E(\mathbf{q})\dot{\mathbf{q}} \tag{1}$$

relates the n-dimensional joint velocity vector $\dot{\mathbf{q}}$ to the m-dimensional end-effector task velocity vector $\dot{\mathbf{x}}_E$ through the $(m \times n)$ end-effector Jacobian matrix $\mathbf{J}_E$.

In all practical cases it is $n \geq m$. If it is $n > m$, then the manipulator is kinematically redundant with respect to the given task and $(n-m)$ degrees-of-freedom are available for solving redundancy. If for some configuration $\hat{\mathbf{q}}$ the end-effector Jacobian matrix has rank r less than m the manipulator is said to be at a kinematic singularity.

To better understand the concepts of kinematic redundancy and singularity, consider the singular value decomposition of the matrix $\mathbf{J}_E$; this can be written in the well-known form

$$\mathbf{J}_E = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \tag{2}$$

where $\mathbf{U}$ is the $(m \times m)$ orthonormal matrix of the output singular vectors $\mathbf{u}_i$, $\mathbf{V}$ is the $(n \times n)$ orthonormal matrix of the input singular vectors $\mathbf{v}_i$, and $\boldsymbol{\Sigma} = (\mathbf{S} \;\; \mathbf{O})$ is the $(m \times n)$ matrix whose $(m \times m)$ diagonal submatrix $\mathbf{S}$ contains the singular values σ_i of the matrix $\mathbf{J}_E$.

If the Jacobian matrix is full-rank, i.e., $r = m$, all the singular values are nonnull. In this case, the range space of $\mathbf{J}_E$ is the entire $\mathbb{R}^m$, and an $(n-m)$-dimensional null space of $\mathbf{J}_E$ exists which is spanned by the $(n-m)$ last input singular vectors. The joint velocity vectors $\dot{\mathbf{q}}$ that yield null end-effector task velocity $\dot{\mathbf{x}}_E$ are shortly termed as null-space velocities. On the other hand, the set of end-effector task velocities $\dot{\mathbf{x}}_E$ that can be obtained as a result of all possible joint velocities constitute the space of feasible velocities for the manipulator's end effector.

At a kinematic singularity, instead, it is $r < m$; thus, the last $(m-r)$ singular values are zero. In this case, the range space of $\mathbf{J}_E$ is an r-dimensional subspace of $\mathbb{R}^m$ spanned by the r first output singular vectors, and an $(n-r)$-dimensional null space of $\mathbf{J}_E$ exists which is spanned by the $(n-r)$ last input singular vectors. Therefore, one effect of a kinematic singularity is to reduce the feasible velocity capabilities of the end effector; another effect is to introduce additional possibilities of null-space velocities.

From a different point of view, when a kinematic singularity is approached, the rth singular value tends to zero and the end-effector velocity produced by a fixed joint velocity along $\mathbf{v}_r$ is decreased proportionally to σ_r. At the singular configuration, the joint-space velocities along $\mathbf{v}_r$ fall in the null-space velocities and the end-effector velocities along $\mathbf{u}_r$ become infeasible to the manipulator.

The inverse solution to the differential kinematics mapping (1) dates back to the pioneering work of Whitney on resolved-rate control [21]. In this framework, the joint velocity vector corresponding to a given end-effector task velocity vector is found through the mapping

$$\dot{\mathbf{q}} = \mathbf{J}_E^{\dagger}(\mathbf{q})\dot{\mathbf{x}}_E \tag{3}$$

where $\mathbf{J}_E^{\dagger}$ is the $(n \times m)$ Moore-Penrose pseudoinverse of the end-effector Jacobian matrix. The obtained $\dot{\mathbf{q}}$ enjoys the minimum-norm property; however, even though the joint velocities are instantaneously minimized, there is no guarantee at all that kinematic singularities are avoided [2].

Non-minimum-norm solutions to (1) based on the Jacobian pseudoinverse can be written in the general form [12]

$$\dot{\mathbf{q}}_{\mathrm{PI}} = \mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E + (\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E)\dot{\mathbf{q}}_0 \tag{4}$$

where $\mathbf{I}$ denotes the $(n \times n)$ identity matrix and $\dot{\mathbf{q}}_0$ is an n-dimensional arbitrary joint velocity vector. In (4) a homogeneous term which is obtained by filtering the null-space velocity components of $\dot{\mathbf{q}}_0$ through the projection operator $(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E)$ is added to the minimum-norm solution (3).

By observing that null-space velocities produce a change in the configuration of the manipulator without affecting its velocity at the end effector, they can be exploited to achieve additional goals—like obstacle or singularity avoidance—besides the desired end-effector task.

A first possibility is to choose the vector $\dot{\mathbf{q}}_0$ as the gradient of a scalar objective function $H(\mathbf{q})$ that suitably describes the above-mentioned constraint task [12], i.e.,

$$\dot{\mathbf{q}}_0 = -\alpha \nabla H(\mathbf{q}) \tag{5}$$

in which α is a scalar gain factor; with this choice, local minima of the selected function are obtained. Easy-to-build optimization criteria are quadratic-type objective functions that also result in tractable expressions for (5).

Another possibility is to compute $\dot{\mathbf{q}}_0$ so as to achieve an $(n-m)$-dimensional constraint task velocity vector $\dot{\mathbf{x}}_C$ associated to a suitable constraint task function $\mathbf{x}_C(\mathbf{q})$ [13], [18]; remarkably, the projection of $\dot{\mathbf{q}}_0$ onto the null space of $\mathbf{J}_E$ ensures lower priority of the constraint task with respect to the end-effector task [10]. In this case, the mapping

$$\dot{\mathbf{x}}_C = \mathbf{J}_C(\mathbf{q})\dot{\mathbf{q}} \tag{6}$$

where $\mathbf{J}_C = \partial \mathbf{x}_C / \partial \mathbf{q}$ is the $((n-m) \times n)$ constraint task Jacobian matrix, must be considered in addition to the mapping (1) when selecting $\dot{\mathbf{q}}_0$ to be used in (4).

A reasonable choice is to compute $\dot{\mathbf{q}}_0$ such that the constraint task reconstruction error $\dot{\mathbf{x}}_C - \mathbf{J}_C\dot{\mathbf{q}}$ is minimized; this gives [18]

$$\dot{\mathbf{q}}_0 = \big(\mathbf{J}_C\big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\big)^{\dagger}\big(\dot{\mathbf{x}}_C - \mathbf{J}_C\mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E\big). \tag{7}$$

By observing that the null-space projection operator is both Hermitian and idempotent, solution (4), (7) can be simplified to [13]

$$\dot{\mathbf{q}} = \mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E + \big(\mathbf{J}_C\big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\big)^{\dagger}\big(\dot{\mathbf{x}}_C - \mathbf{J}_C\mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E\big). \tag{8}$$

III. Kinematic Singularities

To analyze solution (4) consider its expression in terms of the singular value decomposition, which in view of (2) can be arranged in the form

$$\dot{\mathbf{q}}_{\mathrm{PI}} = \sum_{i=1}^{r}\frac{\big(\mathbf{u}_i^T\dot{\mathbf{x}}_E\big)}{\sigma_i}\mathbf{v}_i + \sum_{i=r+1}^{n}\big(\mathbf{v}_i^T\dot{\mathbf{q}}_0\big)\mathbf{v}_i. \tag{9}$$

It can be recognized that in the neighborhood of a kinematic singularity σ_r tends to zero and the solution becomes ill-conditioned. Furthermore, joint velocity discontinuities are experienced in the transition to and from the singular configuration [4]. On the other hand, solution (4) performs very accurately since the reconstruction error vector

$$\dot{\mathbf{x}}_E - \mathbf{J}_E\dot{\mathbf{q}}_{\mathrm{PI}} = \sum_{i=r+1}^{m}\big(\mathbf{u}_i^T\dot{\mathbf{x}}_E\big)\mathbf{u}_i \tag{10}$$

is purely due to loss of mobility caused by the kinematic singularity and is aligned with the infeasible velocity.

A different solution to (1) which is instead robust to the occurrence of kinematic singularities is based on the use of the damped least-squares inverse of the end-effector Jacobian matrix [17], [20] given by

$$\mathbf{J}_E^{*} = \mathbf{J}_E^T\big(\mathbf{J}_E\mathbf{J}_E^T + \lambda^2\mathbf{I}\big)^{-1} = \sum_{i=1}^{r}\frac{\sigma_i}{\sigma_i^2+\lambda^2}\mathbf{v}_i\mathbf{u}_i^T \tag{11}$$

in which $\lambda \in \mathbb{R}$ is the damping factor. By replacing $\mathbf{J}_E^{\dagger}$ with $\mathbf{J}_E^{*}$ in (4), we obtain

$$\dot{\mathbf{q}}_{\mathrm{DLS}} = \mathbf{J}_E^{*}\dot{\mathbf{x}}_E + (\mathbf{I} - \mathbf{J}_E^{*}\mathbf{J}_E)\dot{\mathbf{q}}_0 \tag{12}$$

that can be rewritten as

$$\dot{\mathbf{q}}_{\mathrm{DLS}} = \sum_{i=1}^{r}\frac{\sigma_i\big(\mathbf{u}_i^T\dot{\mathbf{x}}_E\big) + \lambda^2\big(\mathbf{v}_i^T\dot{\mathbf{q}}_0\big)}{\sigma_i^2+\lambda^2}\mathbf{v}_i + \sum_{i=r+1}^{n}\big(\mathbf{v}_i^T\dot{\mathbf{q}}_0\big)\mathbf{v}_i. \tag{13}$$

As can be seen from (13), a nonnull damping factor in (11) and (12) ensures continuity and good conditioning of the solution in the whole manipulator's workspace. However, this is obtained at the expense of an increased reconstruction error due to the damping; in fact, it is

$$\dot{\mathbf{x}}_E - \mathbf{J}_E\dot{\mathbf{q}}_{\mathrm{DLS}} = \sum_{i=1}^{r}\lambda^2\frac{\big(\mathbf{u}_i^T\dot{\mathbf{x}}_E\big) - \sigma_i\big(\mathbf{v}_i^T\dot{\mathbf{q}}_0\big)}{\sigma_i^2+\lambda^2}\mathbf{u}_i + \sum_{i=r+1}^{m}\big(\mathbf{u}_i^T\dot{\mathbf{x}}_E\big)\mathbf{u}_i \tag{14}$$

where additional terms are present with respect to (10) affecting the feasible velocity components. The contribution of these terms can be reduced by both lowering the damping factor far from kinematic singularities and avoiding unnecessary damping along the feasible velocity components. To this purpose the damped least-squares solution has been refined to include a variable damping factor [17] and to provide numerical filtering of the velocity components [14].

A singular region can be conveniently introduced in the neighborhood of a kinematic singularity so that outside the region the Jacobian matrix is full-rank and a null damping factor can be used, while inside the region the damping factor is increased in proportion to closeness to the singularity. The following choice for the damping factor can be adopted [7]

$$\lambda^2 = \begin{cases} 0 & \sigma_m \geq \varepsilon \\ \big(1 - \big(\frac{\sigma_m}{\varepsilon}\big)^2\big)\lambda_{\max}^2 & \sigma_m < \varepsilon \end{cases} \tag{15}$$

which ensures continuity and good shaping of the solution. In (15), ε defines the size of the singular region, and $\lambda_{\max}$ sets the maximum value of the damping factor, which is used at the singularity.

In the case of a single kinematic singularity (i.e., $\sigma_{m-1} > 0$, $\sigma_m \geq 0$) the damped least-squares inverse with numerical filtering $\mathbf{J}_E^{\circ}$ is given by

$$\begin{aligned}\mathbf{J}_E^{\circ} &= \mathbf{J}_E^T\big(\mathbf{J}_E\mathbf{J}_E^T + \lambda^2\mathbf{u}_m\mathbf{u}_m^T\big)^{-1} \\ &= \sum_{i=1}^{m-1}\frac{1}{\sigma_i}\mathbf{v}_i\mathbf{u}_i^T + \frac{\sigma_m}{\sigma_m^2+\lambda^2}\mathbf{v}_m\mathbf{u}_m^T.\end{aligned} \tag{16}$$

By replacing $\mathbf{J}_E^{\dagger}$ with $\mathbf{J}_E^{\circ}$ in (4), we obtain

$$\dot{\mathbf{q}}_{\mathrm{NF}} = \mathbf{J}_E^{\circ}\dot{\mathbf{x}}_E + \big(\mathbf{I} - \mathbf{J}_E^{\circ}\mathbf{J}_E\big)\dot{\mathbf{q}}_0 \tag{17}$$

that can be rewritten as

$$\begin{aligned}\dot{\mathbf{q}}_{\mathrm{NF}} = &\sum_{i=1}^{m-1}\frac{\big(\mathbf{u}_i^T\dot{\mathbf{x}}_E\big)}{\sigma_i}\mathbf{v}_i + \frac{\sigma_m\big(\mathbf{u}_m^T\dot{\mathbf{x}}_E\big) + \lambda^2\big(\mathbf{v}_m^T\dot{\mathbf{q}}_0\big)}{\sigma_m^2+\lambda^2}\mathbf{v}_m \\ &+ \sum_{i=m+1}^{n}\big(\mathbf{v}_i^T\dot{\mathbf{q}}_0\big)\mathbf{v}_i.\end{aligned} \tag{18}$$

Further, it can be easily obtained that

$$\dot{\mathbf{x}}_E - \mathbf{J}_E\dot{\mathbf{q}}_{\mathrm{NF}} = \lambda^2\frac{\big(\mathbf{u}_m^T\dot{\mathbf{x}}_E\big) - \sigma_m\big(\mathbf{v}_m^T\dot{\mathbf{q}}_0\big)}{\sigma_m^2+\lambda^2}\mathbf{u}_m. \tag{19}$$

By comparison of (19) with (10), it can be recognized that the damped least-squares solution with numerical filtering (16), (17) is as much accurate as the pseudoinverse solution (4) not only when low damping is used but also very close to the kinematic singularity. In the other cases, a reconstruction error is present affecting the sole end-effector velocity component that is close to becoming infeasible. On the other hand, (18) shows that, similarly to (13), continuity and good conditioning of the solution are maintained in the transition to and from the singular configuration with lower errors than those found from (14).

According to the above, the damped least-squares solution with numerical filtering represents a good compromise between the accurate tracking performance of the pseudoinverse

solution and the capability of providing feasible joint velocities of the damped least-squares solution. However, care must be taken to handle the case of multiple kinematic singularities so as to avoid ill-conditioning of solution (16), (17); to this purpose, low isotropic damping β, with $\beta^2 \ll \lambda^2$, can be added in (16) [14] yielding

$$\begin{aligned}\mathbf{J}_E^{\diamond} &= \mathbf{J}_E^T\big(\mathbf{J}_E\mathbf{J}_E^T + \beta^2\mathbf{I} + \lambda^2\mathbf{u}_m\mathbf{u}_m^T\big)^{-1} \\ &= \sum_{i=1}^{m-1}\frac{\sigma_i}{\sigma_i^2+\beta^2}\mathbf{v}_i\mathbf{u}_i^T + \frac{\sigma_m}{\sigma_m^2+\beta^2+\lambda^2}\mathbf{v}_m\mathbf{u}_m^T. \quad (20)\end{aligned}$$

IV. Algorithmic Singularities

For the task-priority strategy leading to solution (8) the problem remains of algorithmic singularities. These are configurations at which the matrix $\tilde{\mathbf{J}}_C = \mathbf{J}_C(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E)$ loses rank with full-rank $\mathbf{J}_C$ and $\mathbf{J}_E$.

An algorithmic singularity occurs whenever

$$\mathcal{N}(\mathbf{J}_C) \cap \mathcal{N}(\mathbf{J}_E) \neq \{\mathbf{0}\} \quad (21)$$

since, in this case, the subspaces $\mathcal{N}(\mathbf{J}_C)$ and $\mathcal{N}(\mathbf{J}_E)$ are linearly dependent and the product $\mathbf{J}_C(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E)$ loses rank. It can be recognized that at an algorithmic singularity also

$$\mathcal{N}^{\perp}(\mathbf{J}_C) \cap \mathcal{N}^{\perp}(\mathbf{J}_E) \neq \{\mathbf{0}\} \quad (22)$$

which is equivalent to the condition

$$\mathcal{R}\big(\mathbf{J}_C^T\big) \cap \mathcal{R}\big(\mathbf{J}_E^T\big) \neq \{\mathbf{0}\}. \quad (23)$$

When

$$\mathcal{N}^{\perp}(\mathbf{J}_C) \subset \mathcal{N}^{\perp}(\mathbf{J}_E) \quad (24)$$

the achievement of the secondary task always affects the primary task and thus the two tasks are not compatible. In this case $\tilde{\mathbf{J}}_C = \mathbf{O}$.

Outside algorithmic singularities the subspaces $\mathcal{N}(\mathbf{J}_C)$ and $\mathcal{N}(\mathbf{J}_E)$ are linearly independent and

$$\mathcal{N}^{\perp}(\mathbf{J}_C) \cap \mathcal{N}^{\perp}(\mathbf{J}_E) = \{\mathbf{0}\}. \quad (25)$$

When

$$\mathcal{N}^{\perp}(\mathbf{J}_C) \equiv \mathcal{N}(\mathbf{J}_E) \quad (26)$$

the subspaces $\mathcal{N}^{\perp}(\mathbf{J}_C)$ and $\mathcal{N}^{\perp}(\mathbf{J}_E)$ are orthogonal; thus, joint velocities aimed at achieving the secondary task do not affect the primary task and the two tasks are compatible.

In contrast to the task-space augmentation approach, in which inversion of the augmented Jacobian matrix $\mathbf{J} = (\mathbf{J}_E^T\ \mathbf{J}_C^T)^T$ at algorithmic singularities gives reconstruction errors on both the end-effector and the constraint task velocities, in the task-priority strategy the components of the joint velocity required by the secondary task that cause interference with the primary task are removed through projection by the matrix $\tilde{\mathbf{J}}_C^{\dagger}$; thus, correct primary-task solutions are expected as long as the sole primary-task Jacobian is full-rank. However, similarly to the case of kinematic singularities, close to an algorithmic singularity ill-conditioned and discontinuous joint velocity solutions are experienced due to the null-space velocity (7).

To better understand algorithmic singularities let us resort again to the singular value decomposition method. In this case, it is interesting to study the matrix $\tilde{\mathbf{J}}_C$ in terms of the properties of $\mathbf{J}_E$ and $\mathbf{J}_C$, rather than looking for a decomposition of $\tilde{\mathbf{J}}_C$ as a whole. If

$$\mathbf{J}_C = \sum_{i=1}^{n-m}\sigma_{C,i}\mathbf{u}_{C,i}\mathbf{v}_{C,i}^T \quad (27)$$

is the singular value decomposition of the constraint task Jacobian matrix, by recalling (2) we can decompose $\tilde{\mathbf{J}}_C$ in the form

$$\begin{aligned}\tilde{\mathbf{J}}_C = \mathbf{ABC} &= (\mathbf{u}_{C,1} \quad \cdots \quad \mathbf{u}_{C,n-m}) \\ &\times \begin{pmatrix} \sigma_{C,1}\mathbf{v}_{C,1}^T\mathbf{v}_{r+1} & \cdots & \sigma_{C,1}\mathbf{v}_{C,1}^T\mathbf{v}_n \\ \vdots & \ddots & \vdots \\ \sigma_{C,n-m}\mathbf{v}_{C,n-m}^T\mathbf{v}_{r+1} & \cdots & \sigma_{C,n-m}\mathbf{v}_{C,n-m}^T\mathbf{v}_n \end{pmatrix} \\ &\times \begin{pmatrix} \mathbf{v}_{r+1}^T \\ \vdots \\ \mathbf{v}_n^T \end{pmatrix}. \quad (28)\end{aligned}$$

Since in (28) the matrix $\mathbf{A}$ is orthonormal and the matrix $\mathbf{C}$ is full row rank with orthonormal rows, a loss of rank of $\tilde{\mathbf{J}}_C$ is due to a loss of rank of the $((n-m)\times(n-r))$ matrix $\mathbf{B}$. We can study linear dependency of the rows of $\mathbf{B}$, being $n-m \leq n-r$. It can be recognized that, in view of the orthonormality of the $\mathbf{v}_{C,i}$'s, each row of $\mathbf{B}$ cannot be obtained as a linear combination of the others. Hence, a loss of rank of $\tilde{\mathbf{J}}_C$ is due to a row of $\mathbf{B}$ becoming null. This happens, besides at the singularities of $\mathbf{J}_C$ where $\sigma_{C,n-m} = 0$, when for some i it is $\mathbf{v}_{C,i}^T\mathbf{v}_j = 0\ \forall j \in [r+1, n]$, that corresponds to (22).

When an algorithmic singularity is approached, the above scalar products tend to zero leading to infeasible velocities along $\mathbf{u}_{C,i}$. Therefore, from a practical point of view, solution (8), that requires inversion of $\tilde{\mathbf{J}}_C$, can be used only far from both artificial singularities and singularities of $\mathbf{J}_C$ unless modifications are introduced to handle this case, e.g., [16]. However, while for a given constraint task the singularities of $\mathbf{J}_C$ can be determined in advance, algorithmic singularities are difficult to predict since they depend on how the end-effector task conflicts with the constraint task along the trajectory being executed. For this reason it would be preferable to dispose of some mapping in which the singularities of $\mathbf{J}_C$ are decoupled from the cause of algorithmic singularities.

Let then reconsider the problem of finding the null-space velocity in (4) so as to satisfy the (secondary) constraint task. By replacing the general solution (4) into the mapping (6), we obtain the equation

$$\dot{\mathbf{x}}_C = \mathbf{J}_C\mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E + \mathbf{J}_C\big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\dot{\mathbf{q}}_0. \quad (29)$$

Instead of solving (29) with respect to $\dot{\mathbf{q}}_0$—that leads again to solution (7). We observe that the equality between left- and right-hand side (LHS and RHS, respectively) of (29) can be achieved only for the components of $\dot{\mathbf{x}}_C$ belonging to $\mathcal{R}(\mathbf{J}_C)$ [6]. This means that we should rather look for a solution to

$$\mathbf{J}_C\mathbf{J}_C^{\dagger}\dot{\mathbf{x}}_C = \mathbf{J}_C\mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E + \mathbf{J}_C\big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\dot{\mathbf{q}}_0. \quad (30)$$

At this point, we can further observe that (30) is implied by the equation

$$\mathbf{J}_C^{\dagger}\dot{\mathbf{x}}_C = \mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E + \big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\dot{\mathbf{q}}_0 \tag{31}$$

that can be solved for $\dot{\mathbf{q}}_0$ giving

$$\dot{\mathbf{q}}_0 = \big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)^{\dagger}\big(\mathbf{J}_C^{\dagger}\dot{\mathbf{x}}_C - \mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E\big). \tag{32}$$

By recalling that $(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E)^{\dagger} = (\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E)$ and $\mathbf{J}_E^{\dagger} = \mathbf{J}_E^{\dagger}\mathbf{J}_E\mathbf{J}_E^{\dagger}$, we can reduce solution (32) to the very simple form

$$\dot{\mathbf{q}}_0 = \big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\mathbf{J}_C^{\dagger}\dot{\mathbf{x}}_C. \tag{33}$$

Also exploiting the idempotence of $(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E)$, we finally obtain

$$\dot{\mathbf{q}} = \mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E + \big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\mathbf{J}_C^{\dagger}\dot{\mathbf{x}}_C. \tag{34}$$

An intuitive justification of (34) can be given as follows: The pseudoinverses $\mathbf{J}_E^{\dagger}$ and $\mathbf{J}_C^{\dagger}$ are used to solve separately for the joint velocities associated with the respective task velocities; the joint velocity associated with the constraint task is then projected onto the null space of $\mathbf{J}_E$ to remove the components that would interfere with the end-effector task and finally added to the joint velocity associated with the end-effector task.

It is worth noting that solution (34) is equivalent to solution (8) when the constraint task is compatible with the end-effector task, being in this case

$$\mathbf{J}_C\mathbf{J}_E^{\dagger}\dot{\mathbf{x}}_E = \mathbf{0} \quad \forall\dot{\mathbf{x}}_E \tag{35}$$

$$\tilde{\mathbf{J}}_C^{\dagger} = \big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\mathbf{J}_C^{\dagger}. \tag{36}$$

Also, solution (34) is equivalent to solution (8) when the constraint task is not compatible with the end-effector task. In this case, indeed, it is

$$\tilde{\mathbf{J}}_C = \big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\mathbf{J}_C^{\dagger} = \mathbf{O} \tag{37}$$

and both solutions reduce to (3).

By comparison of solution (34) with solution (8), it can be recognized that the matrix $(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E)\mathbf{J}_C^{\dagger}$ plays in (34) a similar role to the matrix $\tilde{\mathbf{J}}_C^{\dagger}$ in (8); the two matrices, indeed, determine the null-space velocity aimed at achieving the secondary task. When an algorithmic singularity is approached, however, solution (8) increases the null-space velocity to preserve accurate tracking of the constraint task close to become infeasible, while solution (34) progressively reduces the null-space velocity associated to the near-to-be-infeasible component of the secondary task. This can be easily recognized by looking at the expression

$$\big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\mathbf{J}_C^{\dagger} = \sum_{i=1}^{n-m}\sum_{j=r+1}^{n}\frac{\big(\mathbf{v}_{C,i}^T\mathbf{v}_j\big)}{\sigma_{C,i}}\mathbf{v}_j\mathbf{u}_{C,i}^T. \tag{38}$$

In detail, (38) shows that in the neighborhood of an algorithmic singularity, i.e., when for some i it is $\mathbf{v}_{C,i}^T\mathbf{v}_j \to 0\ \forall j \in [r+1, n]$—the matrix $(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E)\mathbf{J}_C^{\dagger}$ progressively reduces the null-space velocity component associated to the component $\mathbf{u}_{C,i}$ of the secondary task.

A nice property of solution (34) is that, as desired, algorithmic singularities are decoupled from the singularities of $\mathbf{J}_C$. Thus, in the case in which the matrix $\mathbf{J}_C$ experiences singular configurations, we can use its damped least-squares inverse in lieu of the pseudoinverse in (34) leading to

$$\big(\mathbf{I} - \mathbf{J}_E^{\dagger}\mathbf{J}_E\big)\mathbf{J}_C^{*} = \sum_{i=1}^{n-m}\sum_{j=r+1}^{n}\frac{\sigma_{C,i}\big(\mathbf{v}_{C,i}^T\mathbf{v}_j\big)}{\sigma_{C,i}^2 + \lambda^2}\mathbf{v}_j\mathbf{u}_{C,i}^T. \tag{39}$$

V. Implementation Aspects

In view of the adoption of the above algorithms for real-time kinematic control of redundant manipulators, numerical implementation aspects become of primary concern. For the type of solutions considered, the key issue is to efficiently handle the matrix inversions required by the different inverse differential kinematics laws.

As for the task priority strategy, it has been seen that solution (8) avoids the projection of $\dot{\mathbf{q}}_0$ in (7) onto the null space of the end-effector Jacobian. However, efficient computation is made difficult by the need of explicitly evaluating the matrix $\tilde{\mathbf{J}}_C$ that contains the end-effector Jacobian pseudoinverse and requires matrix-by-matrix products; in addition, this matrix cannot always be assumed to be full-rank making its inversion computationally demanding.

On the other hand, the proposed solution (34) shows the very simple structure

$$\dot{\mathbf{q}} = \mathbf{J}_E^{\diamond}\dot{\mathbf{x}}_E + \big(\mathbf{I} - \mathbf{J}_E^{\diamond}\mathbf{J}_E\big)\dot{\mathbf{q}}_0 \tag{40}$$

in which the superscript $^{\diamond}$ denotes either of the three above inverses—namely the pseudoinverse, the damped least-squares inverse, and the damped least-squares inverse with numerical filtering—and $\dot{\mathbf{q}}_0$ reduces to the vector $\mathbf{J}_C^{\diamond}\dot{\mathbf{x}}_C$. In this case, it is possible to group the terms involving the inverse as follows

$$\dot{\mathbf{q}} = \mathbf{J}_E^{\diamond}(\dot{\mathbf{x}}_E - \mathbf{J}_E\dot{\mathbf{q}}_0) + \dot{\mathbf{q}}_0. \tag{41}$$

A major computational advantage is gained if full matrix inversion algorithms are avoided by resorting to matrix factorization techniques [11]. For the solutions in the form (40) this can be achieved by first solving the equation

$$\mathbf{H}\mathbf{w} = \dot{\mathbf{x}}_E - \mathbf{J}_E\dot{\mathbf{q}}_0, \tag{42}$$

for $\mathbf{w}$, and then computing the joint velocity as

$$\dot{\mathbf{q}} = \mathbf{J}_E^T\mathbf{w} + \dot{\mathbf{q}}_0. \tag{43}$$

In (42), $\mathbf{H}$ is the $(m \times m)$ symmetric matrix given by $\mathbf{J}_E\mathbf{J}_E^T$, $\mathbf{J}_E\mathbf{J}_E^T + \lambda^2\mathbf{I}$, $\mathbf{J}_E\mathbf{J}_E^T + \beta^2\mathbf{I} + \lambda^2\mathbf{u}_m\mathbf{u}_m^T$, in the cases of solution (4), (12), (17), respectively. It is worth noting that in the case of the two damped least-squares solutions, the matrix $\mathbf{H}$ is positive definite thus allowing the use of very efficient factorization methods such as the Cholesky decomposition.

To get an idea of the computational burden required by the above solutions, the number of floating point operations involved must be considered. We have thus compared the flop count resulting from efficient implementation of the classic solution (8) with that involved by the proposed solution (34) in the typical case $n = 7$, $m = 6$ when damped least-squares inverses are used. It can be recognized that for given

$\dot{\mathbf{x}}_E$, $\dot{\mathbf{x}}_C$, $\mathbf{J}_E$, $\mathbf{J}_C$, and damping factors, solution (8) requires 796 flops while the flop count of solution (34) is 632; these data must also be compared with the computational load of the minimum-norm solution (3) which amounts to 519 flops. Therefore, the extra load required by the proposed solution (112 flops) is less than half the one required by the classic solution (8) (277 flops).

It can be seen that in the three solutions most of the computation is devoted to the inversion of the end-effector Jacobian; further, the difference between the number of flops required by solution (8) in comparison to solution (34) is mostly due to calculation of the matrix $\tilde{\mathbf{J}}_C$. In the case considered this difference is limited because $n - m = 1$ and the above matrix is a row vector of dimension n. However, if $n - m$ is increased, the difference in the flop requirement of the two task-priority solutions becomes more significant. For example, in the case $n = 8$, $m = 6$ it can be found that for given $\dot{\mathbf{x}}_E$, $\dot{\mathbf{x}}_C$, $\mathbf{J}_E$, $\mathbf{J}_C$, and damping factors, solution (8) requires 1212 flops, the flop count of solution (34) is 760, and the computational load of solution (3) amounts to 572 flops. Remarkably, the extra load required by the proposed solution (188 flops) is still small in comparison with the requirements of the minimum-norm solution (3); on the other hand, the extra load required by the classic solution (8) (640 flops) is more than the whole computation of solution (3).

The implementation of the damping least-squares solution with varying damping factor (15) and/or numerical filtering (16) requires on-line evaluation of the smallest singular value σ_m and the associated singular vector $\mathbf{u}_m$. Since the full singular value decomposition is computationally demanding, an algorithm that takes advantage of the nature of robotic matrix calculations has been proposed in [15]. In most applications, however, an estimate of the sole singular value and vector of interest proves to be effective in terms of both numerical efficiency and proper behavior of the solution. The sought estimate can be obtained via the recursive algorithm described in [14] which has later been extended in [5] to handle the crossing of the two smallest singular values; the extra load required by those algorithms is about 100 flops for a six-dimensional task space.

VI. Case Studies

Numerical case studies are developed in this section to analyze the performance of the proposed technique (34) in comparison to the previous inverse kinematics (8). To handle the occurrence of singularities, the inverses required by the single solutions have been replaced by either an inverse—denoted by the superscript $^{\#}$—based on truncated singular value decomposition of the relevant matrix or the damped least-squares inverse. Further, in view of the implementation for kinematic control of redundant manipulators, also closed-loop versions [3] of the above three inverse kinematics algorithms have been considered.

In the following, we will thus consider the three inverse kinematics algorithms

$$\dot{\mathbf{q}}_{\mathrm{TPI}} = \mathbf{J}_E^{\#}\mathbf{w}_E + \left(\mathbf{J}_C\left(\mathbf{I} - \mathbf{J}_E^{\#}\mathbf{J}_E\right)\right)^{\#}\left(\mathbf{w}_C - \mathbf{J}_C\mathbf{J}_E^{\#}\mathbf{w}_E\right) \tag{44}$$

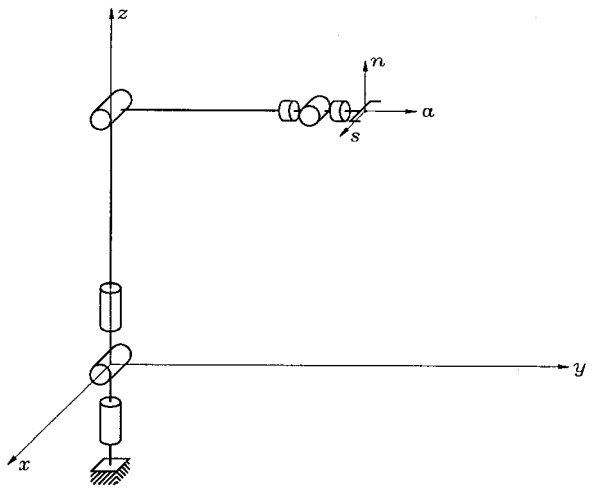

Fig. 1. Kinematic structure of the seven-joint manipulator with human-arm-like geometry.

$$\dot{\mathbf{q}}_{\mathrm{DLS}} = \mathbf{J}_E^{*}\mathbf{w}_E + (\mathbf{J}_C(\mathbf{I} - \mathbf{J}_E^{*}\mathbf{J}_E))^{*}(\mathbf{w}_C - \mathbf{J}_C\mathbf{J}_E^{*}\mathbf{w}_E) \tag{45}$$

$$\dot{\mathbf{q}}_{\mathrm{NEW}} = \mathbf{J}_E^{*}\mathbf{w}_E + (\mathbf{I} - \mathbf{J}_E^{*}\mathbf{J}_E)\mathbf{J}_C^{*}\mathbf{w}_C \tag{46}$$

with

$$\mathbf{w}_E = \dot{\mathbf{x}}_{E,d} + \mathbf{K}_E\mathbf{e}_E \tag{47}$$

$$\mathbf{w}_C = \dot{\mathbf{x}}_{C,d} + \mathbf{K}_C\mathbf{e}_C \tag{48}$$

where $\mathbf{x}_{E,d}$, $\mathbf{x}_{C,d}$ represent the desired trajectories for the end-effector task and the constraint task, respectively, $\mathbf{K}_E$ and $\mathbf{K}_C$ are suitable gain matrices, and $\mathbf{e}_E$, $\mathbf{e}_C$ are the algorithm's errors relative to the end-effector task and the constraint task respectively. Remarkably, the corresponding open-loop inverse kinematics algorithms are recovered by setting null gain matrices.

We have considered a seven-degree-of-freedom manipulator with an human-arm-like geometry, i.e., constituted by an arm and a forearm connecting a spherical shoulder to a spherical wrist through an elbow joint; a distal link is also present attaching the end effector to the wrist. The kinematic structure of the manipulator has zero offsets and the dimensions of the three links are 0.5, 0.4, 0.1 [m] proceeding from the base to the end point, respectively. The manipulator is depicted in Fig. 1 at the joint configuration $(0\ 0\ 0\ -\pi/2\ 0\ 0\ 0)^T$ [rad]; the base frame and the end-effector frame are also displayed.

A. Case A

The primary task is to execute a given motion of the end effector. The error $\mathbf{e}_E$ has a position and orientation component, and is given by

$$\mathbf{e}_E = \begin{pmatrix} \mathbf{p}_d - \mathbf{p} \\ \frac{1}{2}(\mathbf{n} \times \mathbf{n}_d + \mathbf{s} \times \mathbf{s}_d + \mathbf{a} \times \mathbf{a}_d) \end{pmatrix}$$

where $\mathbf{p}_d$, $\mathbf{p}$ are the (3×1) vectors of the desired and actual position of the end effector, and $\mathbf{R}_d = (\mathbf{n}_d \quad \mathbf{s}_d \quad \mathbf{a}_d)$, $\mathbf{R} = (\mathbf{n} \quad \mathbf{s} \quad \mathbf{a})$ are the (3×3) rotation matrices expressing the desired and actual orientation of the end effector, respectively.

The initial and final location of the end effector are assigned and they are interpolated using a fifth-order polynomial time

law such that null initial and final velocities and accelerations are obtained. This results in a straight-line path for the end-effector position while the end-effector orientation describes a rotation around a fixed axis. The corresponding time laws of the translational and angular velocities are also computed by the symbolic time derivative of the fifth-order polynomial time law to set the reference trajectory $\dot{\mathbf{x}}_{E,d}(t)$ for the end-effector task velocity vector. The initial configuration of the arm is

$$\mathbf{q}(0) = (0 \quad 0 \quad 0 \quad -\tfrac{\pi}{2} \quad 0 \quad \tfrac{\pi}{4} \quad 0)^T \text{ [rad]}$$

corresponding to the initial location

$$\mathbf{p}_i = \begin{pmatrix} 0 \\ 0.4707 \\ 0.5707 \end{pmatrix} \text{ [m]} \quad \mathbf{R}_i = \begin{pmatrix} 0 & 1 & 0 \\ -\sqrt{2}/2 & 0 & \sqrt{2}/2 \\ \sqrt{2}/2 & 0 & \sqrt{2}/2 \end{pmatrix}.$$

For clarity of presentation, we have chosen an end-effector motion that develops in the (yz)-plane. The final desired end-point location is

$$\mathbf{p}_f = \begin{pmatrix} 0 \\ 0.4707 \\ 0.4293 \end{pmatrix} \text{ [m]} \quad \mathbf{R}_f = \begin{pmatrix} 0 & 1 & 0 \\ \sqrt{2}/2 & 0 & \sqrt{2}/2 \\ \sqrt{2}/2 & 0 & -\sqrt{2}/2 \end{pmatrix}$$

that is a downward displacement of 0.1414 m along with a $\pi/2$ rad clockwise rotation around the x-axis is commanded to the end effector.

Since in our case it is $n = 7$ and $m = 6$, a one-dimensional secondary task can be assigned; we have required the angle of the fifth joint of the manipulator to follow a fifth-order polynomial time law $x_{C,d}(t)$ with null initial and final velocities and accelerations. We have also computed the corresponding time law of the constraint-task velocity to set the reference trajectory $\dot{x}_{C,d}(t)$. As can be recognized from $\mathbf{q}(0)$ the initial angle of joint 5 is 0 rad; let the final desired value be $\pi/4$ rad. The error $\mathbf{e}_C$ is a scalar and is given by

$$e_C = x_{C,d} - q_5.$$

The duration of both the end-effector and the constraint task is 1 s.

It can be recognized that the assigned task causes the occurrence of both an algorithmic singularity (because at the beginning of the motion movement of joint 5 as required by the constraint task conflicts with the desired rotation of the end effector around the x-axis) and a kinematic singularity (because the assigned end-effector motion requires joint 6 to cross the zero angle, causing occurrence of the wrist singularity).

For the sake of argument, we have chosen to assume as a measure of closeness to a singularity the smallest singular value being of the order of 0.01. This means that a 0.1 m/s end-effector velocity commanded along a close-to-degeneracy direction would give a 10 rad/sec joint-velocity norm. Accordingly, we have set damping factors and truncation threshold as 0.01 where required.

A discrete time implementation of the algorithms has been developed with a sampling time of 1 ms. At each sampling instant, the reference trajectories for the position and velocity task variables, the direct kinematics, and the inverse velocity transformation are evaluated; Euler integration of the obtained joint velocities is then used to update the joint positions.

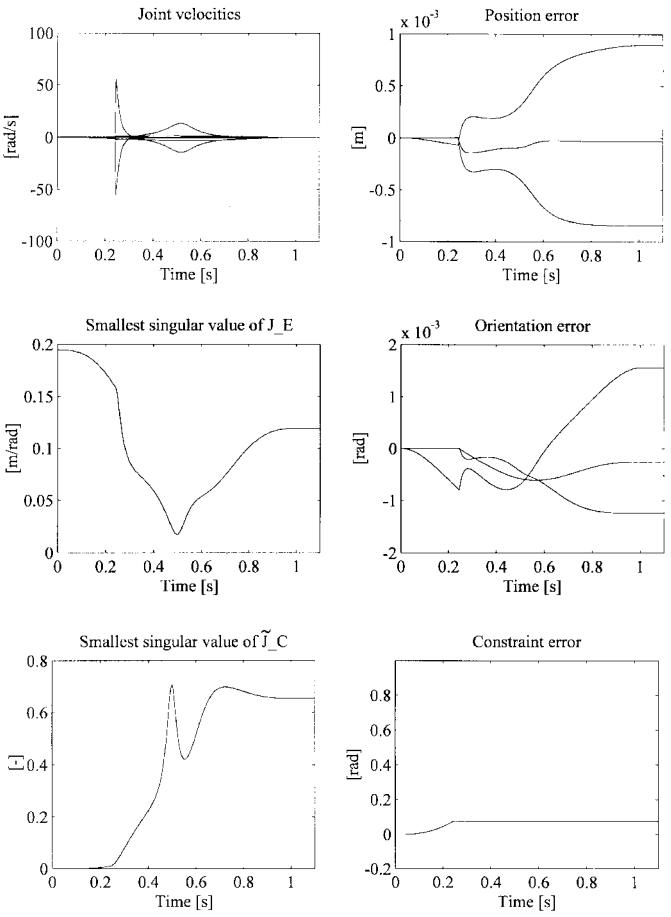

Fig. 2. Case A: Simulation results of the open-loop inverse kinematics algorithm (44).

A first set of simulations has been run to compare the performance of the open-loop versions of the algorithms (44)–(46) in the execution of the above task.

Fig. 2 reports the results obtained with the inverse differential kinematics (44), i.e., the task priority redundancy resolution using matrix inversion based on truncated singular value decomposition. It can be recognized that at the exit of the algorithmic singularity the position error experiences a sudden increase, while the constraint error stops growing; this behavior is contrary to what one would expect from the task-priority strategy, since the error on the primary task is increased in the attempt to fulfill the secondary task. The effect is due to the high null-space velocities, caused by ill-conditioned $\tilde{\mathbf{J}}_C$, that give finite end-effector displacements when numerically integrated over a finite time interval. One additional drawback of solution (44) is that it generates discontinuous joint velocities due to the threshold in the truncated singular value decomposition.

Fig. 3(a) reports the results obtained with the inverse differential kinematics (45), i.e., the task priority redundancy resolution using matrix inversion based on damped least squares. It can be recognized that large position and orientation errors are obtained with errors also in the constraint task; on the other hand, the joint velocities are much lower than the previous case and show continuous time histories. The

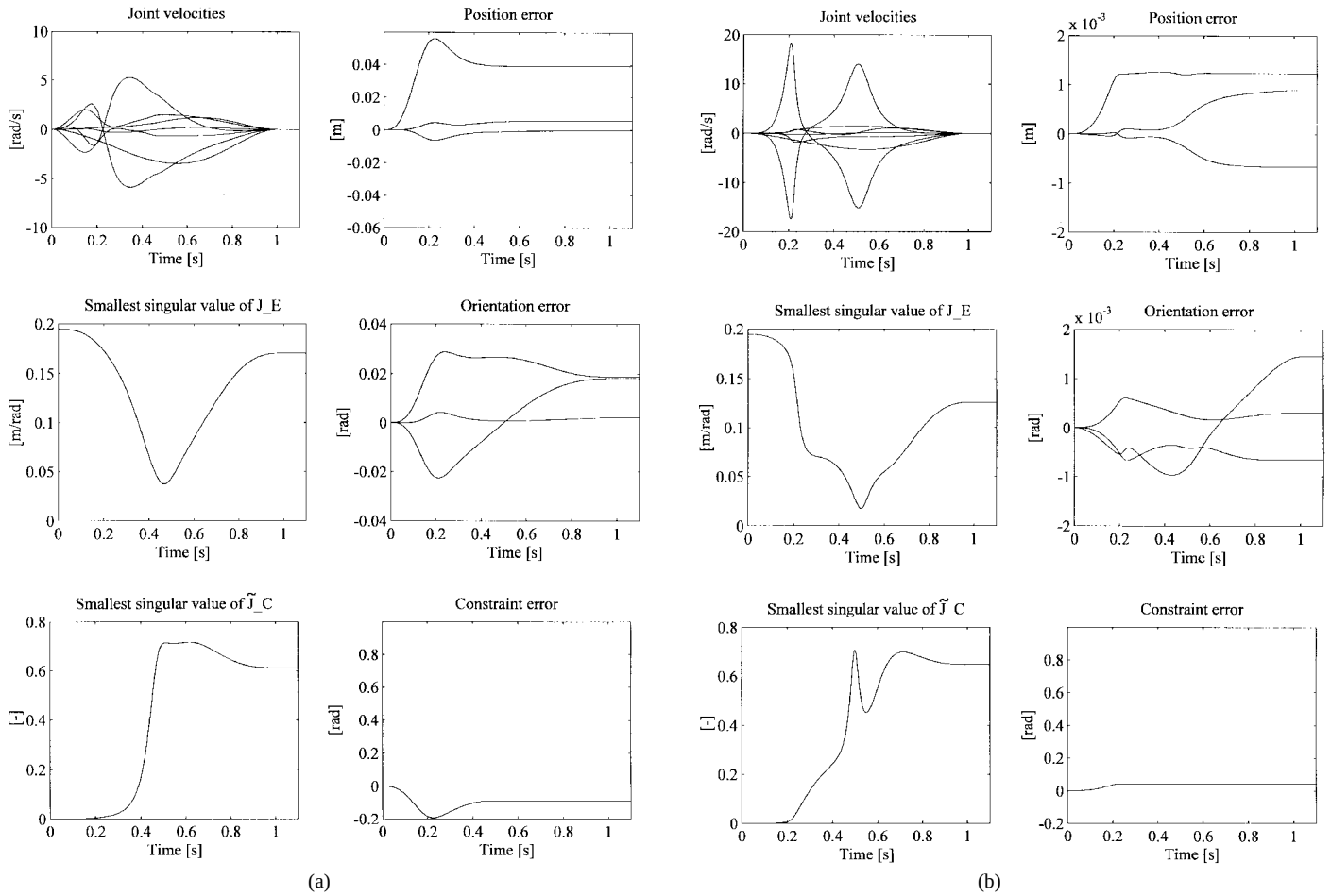

Fig. 3. (a) Case A: Simulation results of the open-loop inverse kinematics algorithm (45). (b) Case A: Simulation results of the open-loop inverse kinematics algorithm (45). $\lambda = 10^{-3}$ in $\mathbf{J}_E^*$.

large errors on the primary task are due to the use of $\mathbf{J}_E^*$; as shown by (14), indeed, solutions of the type (12) experience reconstruction errors affecting the primary-task velocity. As a matter of fact, the results obtained by using (45) with the same parameters as above but $\lambda = 10^{-3}$ in the sole $\mathbf{J}_E^*$ [see Fig. 3(b)] show improved accuracy in tracking the primary task at the expense of increased joint velocities.

Fig. 4(a) reports the results obtained with the proposed inverse differential kinematics (46). It can be recognized that an accurate tracking of the primary task is obtained [compare with the time histories of the position and orientation errors in Figs. 2 and 3(a)] at the expense of a large constraint-task error (joint 5 does not move almost at all). Remarkably, the joint velocities remain low with continuous time histories. Since solution (46) is in the same form as (12), we have also investigated dependence from the damping factor in $\mathbf{J}_E^*$ of the primary-task tracking performance. The results obtained with the same parameters as above but $\lambda = 10^{-3}$ in the sole $\mathbf{J}_E^*$ are reported in Fig. 4(b). It can be recognized that solution (46) shows less sensitivity than solution (45) to changes of the damping factor in $\mathbf{J}_E^*$; since the two solutions differ in the handling of the secondary task, we can conclude that in the case of solution (45) the primary-task errors were essentially due to the second term on the RHS. On the other hand, the contribution of the second term on the RHS of (46) to the arm motion is minimal being the secondary task scarcely compatible with the primary task.

A second set of simulations has been run to compare the performance of the closed-loop versions of the algorithms (44)–(46) in the execution of the same task as above.

Fig. 5 reports the results obtained with the inverse differential kinematics (44), i.e., the task priority redundancy resolution using matrix inversion based on truncated singular value decomposition, with $\mathbf{K}_E = \mathrm{diag}\,\{1000, 1000, 1000, 1000, 1000, 1000\}$ and $\mathbf{K}_C = 2$. Notice that the low gain on the constraint error has been required to ensure stability of the inverse kinematic algorithm. It can be recognized that with respect to the corresponding open-loop case, a substantial reduction of the end-effector task tracking error is obtained with null tracking errors at steady state also for the constraint task. However, high peaks and discontinuities of the joint velocity histories are still present due to the threshold in the truncated singular value decomposition.

It was not possible to run the closed-loop inverse differential kinematics (45), i.e., the task priority redundancy resolution using matrix inversion based on damped least squares, with significant values of the feedback gains in (47), (48). This is due to the large errors given by the inverse velocity

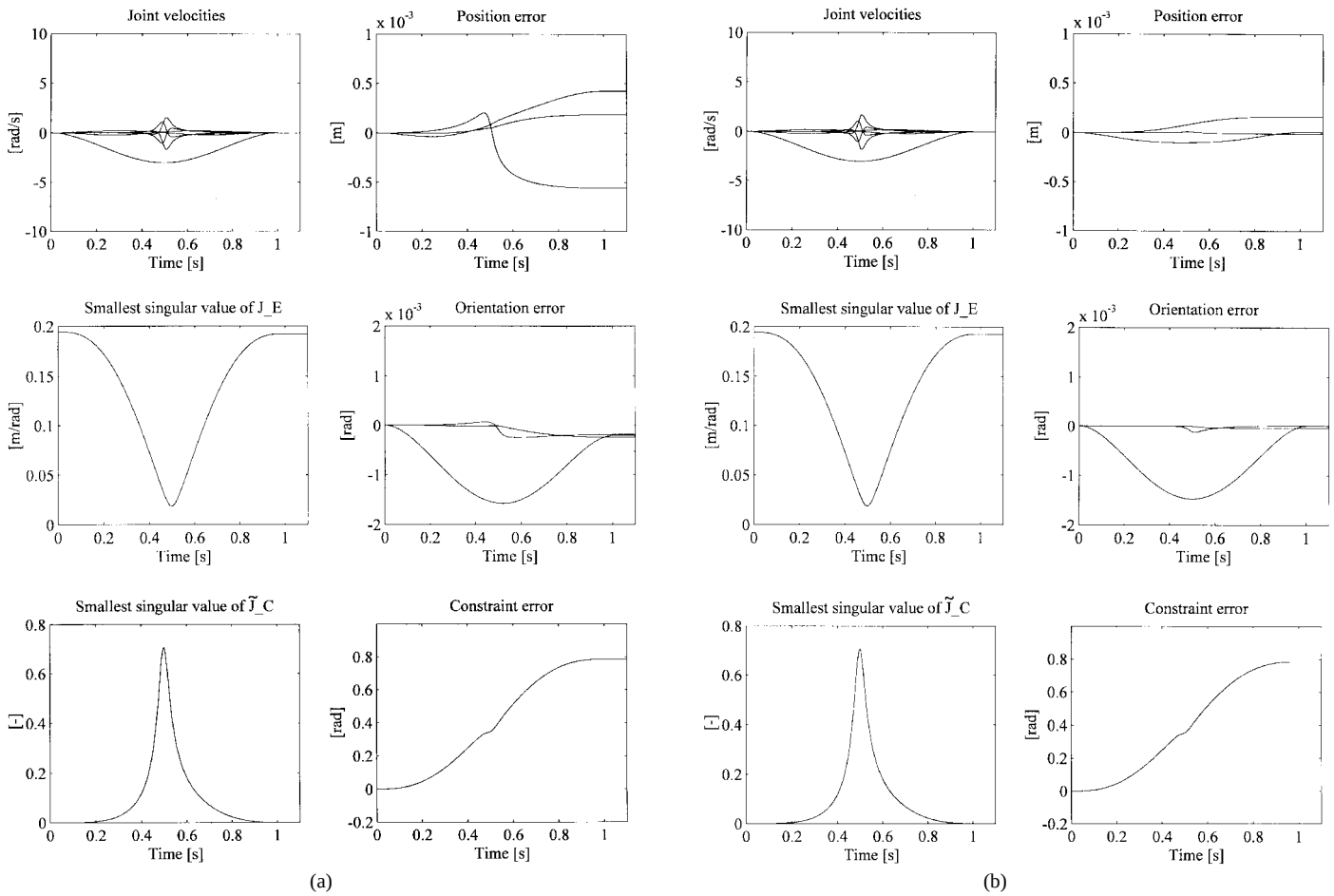

Fig. 4. (a) Case A: Simulation results of the open-loop inverse kinematics algorithm (46). (b) Case A: Simulation results of the open-loop inverse kinematics algorithm (46). $\lambda = 10^{-3}$ in $\mathbf{J}_E^*$.

transformation (45) with $\lambda = 0.01$ [cf. Fig. 3(a)] that build up when fed back through (47) and (48).

Fig. 6 reports the results obtained with the proposed inverse differential kinematics (46) with $\mathbf{K}_E = \text{diag}\{1000, 1000, 1000, 1000, 1000, 1000\}$ and $\mathbf{K}_C = 2000$; with respect to (44), an higher gain on the constraint error is made possible because of the different filtering of the constraint task term. It can be recognized that the closed-loop version of the algorithm achieves the expected improvement over the open-loop case in terms of both the position/orientation errors and the constraint error. The tracking errors are reduced during the arm motion and are null at steady state; this is obtained, however, at the expense of joint velocities larger than in the open-loop case.

B. Case B

The primary task is to execute a given displacement of the end-effector. Thus, the error $\mathbf{e}_E$ has only a position component and is given by

$$\mathbf{e}_E = \mathbf{p}_d - \mathbf{p}.$$

The initial and final position of the end effector are assigned and they are interpolated using a fifth-order polynomial time law such that null initial and final velocities and accelerations are obtained; this results in a straight-line path for the end-effector position. The corresponding time law of the translational velocity is also computed to set the reference trajectory $\dot{\mathbf{x}}_{E,d}(t)$ for the end-effector task velocity vector. The initial configuration of the arm is

$$\mathbf{q}(0) = (0 \quad \tfrac{\pi}{3} \quad 0 \quad -\tfrac{2\pi}{3} \quad 0 \quad 0 \quad 0)^T \text{ [rad]}$$

corresponding to the initial position

$$\mathbf{p}_i = \begin{pmatrix} 0 \\ 0 \\ 0.5 \end{pmatrix} \text{ [m]},$$

For clarity of presentation, we have chosen a task that develops in the (yz)-plane. The final desired end-point position is

$$\mathbf{p}_f = \begin{pmatrix} 0 \\ 0 \\ 0.9 \end{pmatrix} \text{ [m]}$$

that is an upward displacement of 0.4 m is commanded to the end effector.

Since in our case it is $n = 7$ and $m = 3$, a four-dimensional secondary task might be specified. To investigate the assignment of less constraints than the available degree of redundancy, we have chosen to prescribe a desired orientation of the end effector, that results in a three-dimensional secondary task. As can be recognized from $\mathbf{q}(0)$, the initial

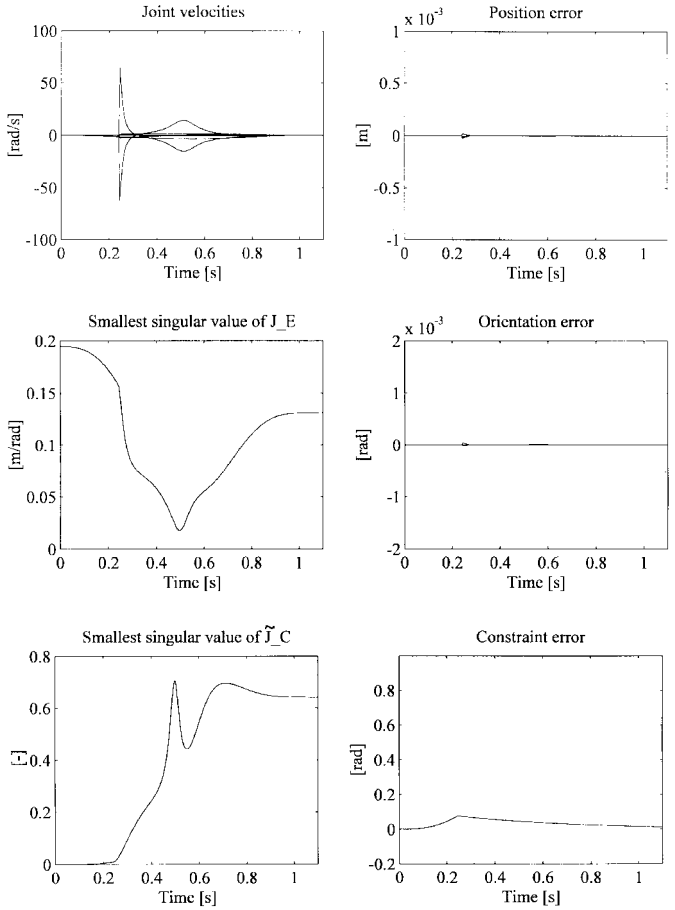

Fig. 5. Case A: Simulation results of the closed-loop inverse kinematics algorithm (44).

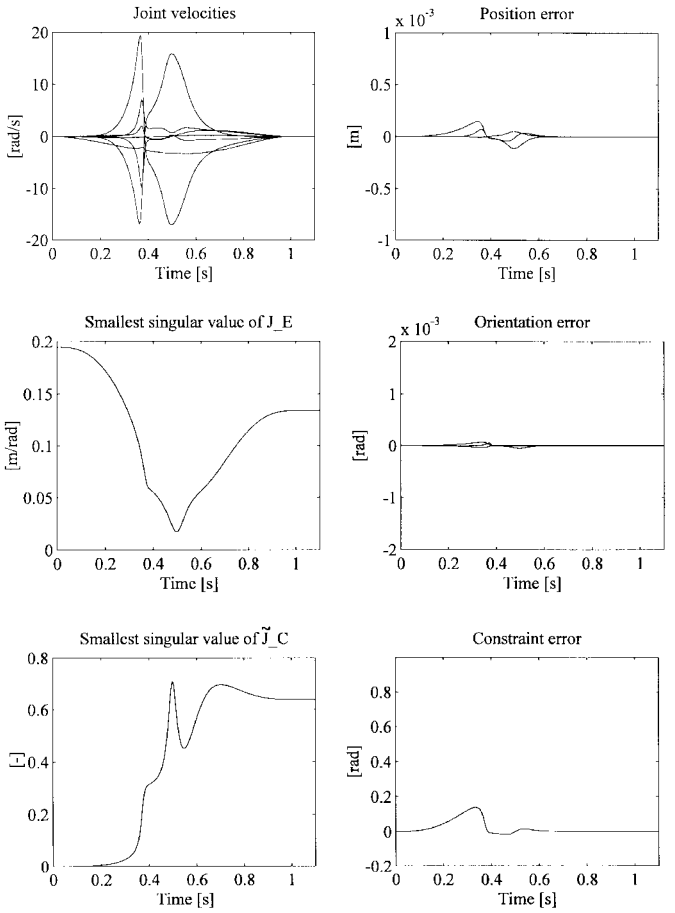

Fig. 6. Case A: Simulation results of the closed-loop inverse kinematics algorithm (46).

orientation of the end effector is expressed by the rotation matrix

$$\mathbf{R}_i = \begin{pmatrix} 0 & 1 & 0 \\ -0.5 & 0 & \sqrt{3}/2 \\ \sqrt{3}/2 & 0 & 0.5 \end{pmatrix}.$$

Let the final desired end-effector orientation be

$$\mathbf{R}_f = \begin{pmatrix} 0 & 1 & 0 \\ 0.5 & 0 & \sqrt{3}/2 \\ \sqrt{3}/2 & 0 & -0.5 \end{pmatrix}.$$

The initial and final orientation of the end effector are interpolated through the equivalent axis/angle method by using a 5th-order polynomial time law such that null initial and final velocities and accelerations are obtained; this results in a $\pi/3$ clockwise rotation around the x-axis. The corresponding time law of the angular velocity is also computed to set the reference trajectory $\dot{\mathbf{x}}_{C,d}(t)$ for the constraint task velocity vector. Finally, the error $\mathbf{e}_C$ is given by

$$\mathbf{e}_C = \frac{1}{2}(\mathbf{n} \times \mathbf{n}_d + \mathbf{s} \times \mathbf{s}_d + \mathbf{a} \times \mathbf{a}_d).$$

The duration of both the end-effector and the constraint task is 2 s.

It can be recognized that the assigned task causes the occurrence of an algorithmic singularity because at the end of the motion the end-effector position cannot be tracked along with the assigned orientation.

As in Case A, we have set damping factors and truncation threshold as 0.01 where required and a discrete time implementation of the algorithms has been developed with a sampling time of 1 ms.

A first set of simulations has been run to compare the performance of the open-loop versions of the algorithms (44)–(46) in the execution of the task.

Fig. 7 reports the results obtained with the inverse differential kinematics (44), i.e., the task priority redundancy resolution using matrix inversion based on truncated singular value decomposition. It can be recognized that in entering the algorithmic singularity high joint velocities are obtained due to ill-conditioned $\tilde{\mathbf{J}}_C$, and the position and orientation error experience a considerable increase.

Fig. 8 reports the results obtained with the inverse differential kinematics (45), i.e., the task priority redundancy resolution using matrix inversion based on damped least squares. It can be recognized that very large position and orientation errors are present. Surprisingly, high joint velocities are experienced too; this happens because the two singularities (namely, kinematic and algorithmic) are entered at the same time, and there the single damped least-squares inverses give maximum velocity transformation ratios.

Fig. 9 reports the results obtained with the proposed inverse differential kinematics (46). It can be recognized that an accurate tracking of the end-effector position is obtained

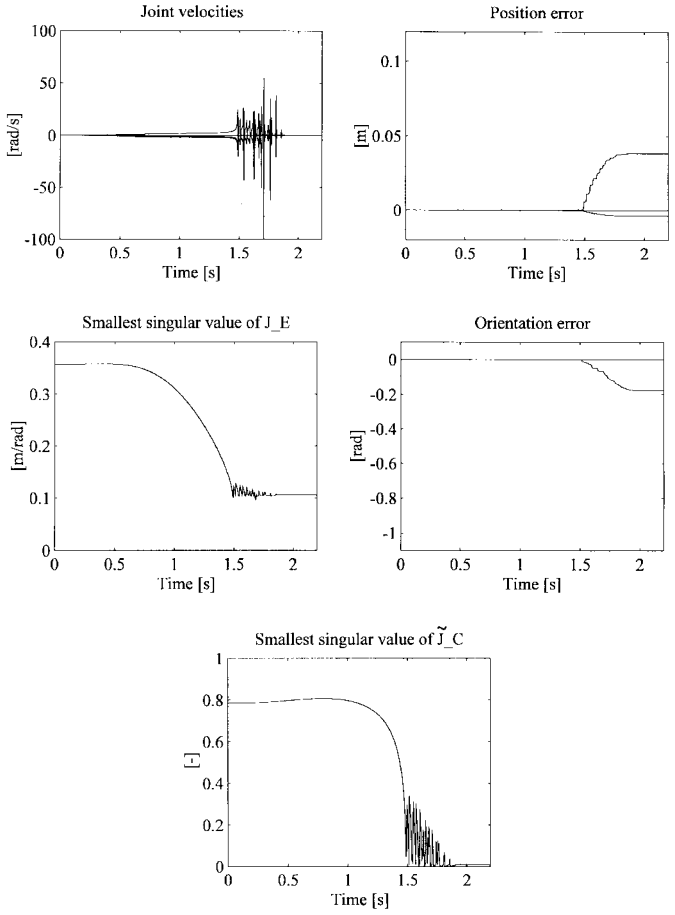

Fig. 7. Case B: Simulation results of the open-loop inverse kinematics algorithm (44).

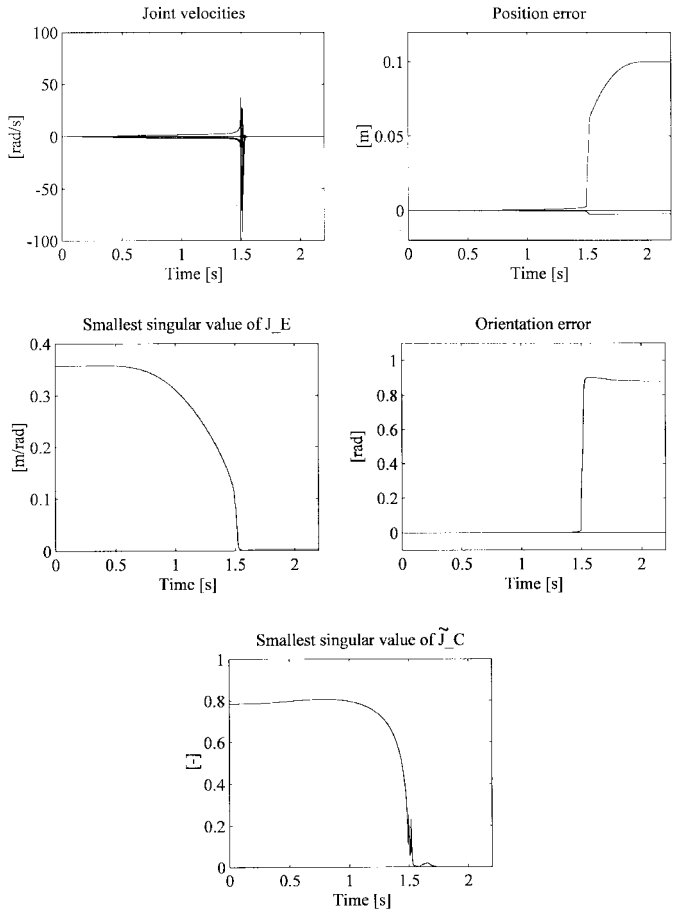

Fig. 8. Case B: Simulation results of the open-loop inverse kinematics algorithm (45).

(compare with the time histories of the position error in Figs. 7 and 8) at the expense of a large orientation error (the orientation is practically kept constant). Remarkably, the joint velocities remain low with continuous time histories.

A second set of simulations has been run to compare the performance of the closed-loop versions of the algorithms (44)–(46) in the execution of the same task as above.

It was not possible to run both the inverse differential kinematics (44), i.e., the task priority redundancy resolution using matrix inversion based on truncated singular value decomposition, and the closed-loop inverse differential kinematics (45), i.e., the task priority redundancy resolution using matrix inversion based on damped least squares, with significant values of the feedback gains in (47) and (48). This is due to the large errors given by the inverse velocity transformation (44) and (45) (cf. Figs. 7 and 8) that build up when fed back through (47) and (48).

Fig. 10 reports the results obtained with the proposed inverse differential kinematics (46) with $\mathbf{K}_E = \text{diag}\ \{1000, 1000, 1000\}$ and $\mathbf{K}_C = \text{diag}\ \{2000, 2000, 2000\}$. It can be recognized that the closed-loop version of the algorithm improves over the open-loop case in terms of both the position error and the orientation error. The tracking errors, however, are reduced during the arm motion but are not null at steady state. The steady-state error on the end-effector orientation is due to the algorithmic singularity and cannot be avoided. On the other hand, the steady-state error on the end-effector position is due to the use of $\mathbf{J}_E^*$ in the null-space filtering in (46) and depends on λ.

VII. Conclusion

Real-time implementation of kinematic control schemes for robot manipulators demands robustness to the occurrence of singular configurations. In the case of redundant manipulators, singular configurations are either those at which the end effector loses mobility (i.e., kinematic singularities) or those at which the end-effector task and the constraint task conflict despite the redundant degrees of freedom (i.e., algorithmic singularities).

This paper has addressed the handling of singular configurations in the framework of the task-priority redundancy resolution technique.

As for kinematic singularities, we have studied the reconstruction errors on the commanded end-effector velocity obtained in the neighborhood of the singularity when null-space projection terms are present. It has been recognized that the damped least-squares solution with numerical filtering represents a good compromise between the accurate tracking performance of the pseudoinverse solution and the capability of providing feasible joint velocities of the damped least-

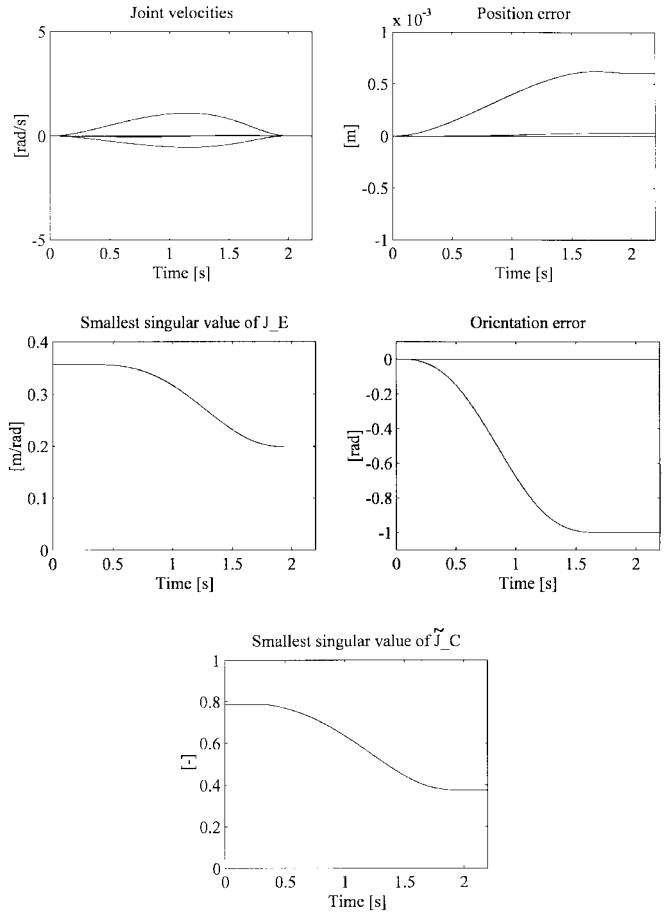

Fig. 9. Case B: Simulation results of the open-loop inverse kinematics algorithm (46).

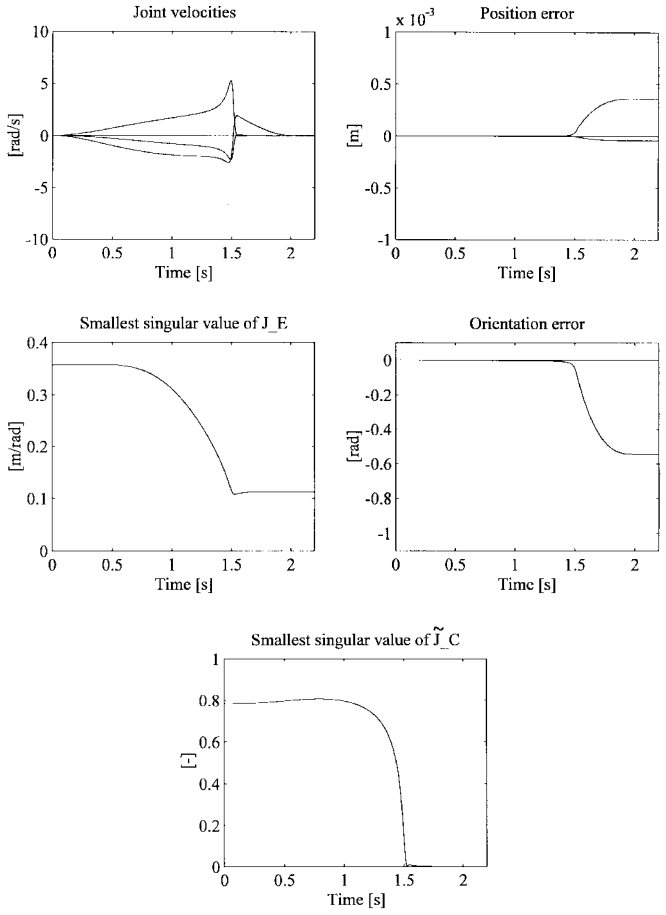

Fig. 10. Case B: Simulation results of the closed-loop inverse kinematics algorithm (46).

squares solution; care must be taken to handle the case of multiple kinematic singularities.

As for algorithmic singularities, a new task-priority redundancy resolution technique has been developed which avoids the inversion of the matrix which is ill-conditioned in the neighborhood of the singularity; this gives continuous joint velocity solutions and accurate tracking of the end-effector task. The proposed inverse kinematics algorithm is also shown to be advantageous from a computational point of view with respect to the classic task-priority redundancy resolution inverse kinematics.

Two case studies have been worked out to compare the performance of different inverse kinematics algorithms applied to a seven-degree-of-freedom manipulator. The first case study considers a six-dimensional end-effector task along with a one-dimensional constraint task. The second case study investigates the assignment of less constraints than the available degree of redundancy; in detail, the primary task is the three-dimensional end-effector position and the secondary task is the three-dimensional end-effector orientation.

It has been recognized that the use of a damped least-squares inverse of $\tilde{\mathbf{J}}_C$ might be problematic when $\mathbf{J}_E^*$ is used in lieu of $\mathbf{J}_E^\dagger$, i.e., when both kinematic and algorithmic singularities are expected; this becomes critical in closed-loop inverse kinematics algorithms. On the other hand, the proposed solution—which has low tracking accuracy on the secondary task but good robustness to singularities—well fits to the arrangement in a closed-loop form besides giving a computationally attractive task-priority inverse velocity transformation.

REFERENCES

[1] J. Baillieul, "Kinematic programming alternatives for redundant manipulators," in *Proc. 1985 IEEE Int. Conf. Robot. Automat.*, St. Louis, MI, Mar. 1985, pp. 722–728.

[2] J. Baillieul, J. M. Hollerbach, and R. W. Brockett, "Programming and control of kinematically redundant manipulators," in *Proc. 23rd IEEE Conf. Decision Contr.*, Las Vegas, NV, Dec. 1984, pp. 768–774.

[3] 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," *Int. J. Robot. Res.*, vol. 10, no. 4, pp. 410–425, 1991.

[4] S. Chiaverini, "Inverse differential kinematics of robotic manipulators at singular and near-singular configurations," in *1992 IEEE Int. Conf. Robot. Automat.—Tutorial on Redundancy: Performance Indices, Singularities Avoidance, and Algorithmic Implementations*, Nice, France, May 1992, pp. 2-1–9.

[5] ———, "Estimate of the two smallest singular values of the Jacobian matrix: Application to damped least-squares inverse kinematics," *J. Robot. Syst.*, vol. 10, no. 8, pp. 991–1008, 1993.

[6] ———, "Task-priority redundancy resolution with robustness to algorithmic singularities," in *Postprint 4th IFAC Symp. Robot Contr. (SY.RO.CO.'94)*, Capri, Italy, vol. I, Sept. 1994, pp. 393–399.

[7] S. Chiaverini, O. Egeland, and R. K. Kanestrøm, "Achieving user-defined accuracy with damped least-squares inverse kinematics," in *Proc. 5th Int. Conf. Advanced Robot. (ICAR'91)*, Pisa, Italy, vol. I, June 1991, pp. 672–677.

[8] O. Egeland, "Task-space tracking with redundant manipulators," *IEEE J. Robot. Automat.*, vol. RA-3, pp. 471–475, 1987.
[9] O. Egeland, J. R. Sagli, I. Spangelo, and S. Chiaverini, "A damped least-squares solution to redundancy resolution," in *Proc. 1991 IEEE Int. Conf. Robot. Automat.*, Sacramento, CA, Apr. 1991, pp. 945–950.
[10] H. Hanafusa, T. Yoshikawa, and Y. Nakamura, "Analysis and control of articulated robot arms with redundancy," in *Preprint IFAC 8th Triennal World Congress*, Kyoto, Japan, Aug. 1981, vol. 14, pp. 78–83.
[11] C. A. Klein and C.-H. Huang, "Review of pseudoinverse control for use with kinematically redundant manipulators," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, pp. 245–250, 1983.
[12] A. Liégeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-7, pp. 868–871, 1977.
[13] A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *The Int. J. Robot. Res.*, vol. 4, no. 3, pp. 109–117, 1985.
[14] ———, "Numerical filtering for the operation of robotic manipulators through kinematically singular configurations," *J. Robot. Syst.*, vol. 5, no. 6, pp. 527–552, 1988.
[15] ———, "The singular value decomposition: Computation and applications to robotics," *The Int. J. Robot. Res.*, vol. 8, no. 6, pp. 63–79, 1989.
[16] A. A. Maciejewski and J. M. Reagin, "A parallel algorithm and architecture for the control of kinematically redundant manipulators," *IEEE Trans. Robot. Automat.*, vol. 10, pp. 405–414, 1994.
[17] Y. Nakamura and H. Hanafusa, "Inverse kinematic solutions with singularity robustness for robot manipulator control," *Trans. ASME J. Dynamic Syst., Measure., Contr.*, vol. 108, pp. 163–171, 1986.
[18] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *The Int. J. Robot. Res.*, vol. 6, no. 2, pp. 3–15, 1987.
[19] L. Sciavicco and B. Siciliano, "A solution algorithm to the inverse kinematic problem for redundant manipulators," *IEEE J. Robot. Automat.*, vol. 4, pp. 403–410, 1988.
[20] C. W. Wampler II, "Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-16, pp. 93–101, 1986.
[21] D. E . Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man, Machine Syst.*, vol. MMS-10, pp. 47–53, 1969.

Stefano Chiaverini received the Laurea and the Research Doctorate degrees in electronic engineering from the University of Naples, Italy, in 1986 and 1990, respectively.

He became a Research Associate in 1990 and an Assistant Professor in 1993 at the Department of Computer and Systems Engineering of the University of Naples. From January to June 1989 he was a Visiting Scientist at the Robotics Laboratory of the German Aerospace Research Establishment (DLR) in Oberpfaffenhofen, Germany. His research interests include manipulator inverse kinematics techniques, redundant manipulator control, cooperative robot systems, and force/position control of manipulators.