Robotica (1990) volume 8, pp 231–243

A closed-loop inverse kinematic scheme for on-line joint-based robot control*

Bruno Siciliano

Dipartimento di Informatica e Sistemistica, Università degli Study "Federico II" di Napoli, Via Claudio 21, 80125 Napoli (Italy)

(Received in final form: November 14, 1989)

SUMMARY

A computationally fast inverse kinematic scheme is derived which solves robot's end-effector (EE) trajectories in terms of joint trajectories. The inverse kinematic problem (IKP) is cast as a control problem for a simple dynamic system. The resulting closed-loop algorithms are shown to guarantee satisfactory tracking performance. Differently from previous first-order schemes which only solve for joint positions and velocities, we propose here new *second order tracking schemes* which allow the on-line generation of joint position + velocity + acceleration (PVA) reference trajectories for any computed torque-like controller in sensor-based robot applications. The algorithms do explicitly solve the IKP for both EE position and orientation. Simulation results for a *six*-degree-of-freedom PUMA-like geometry demonstrate the effectiveness of the scheme, even near singularities.

KEYWORDS: Kinematic scheme; Closed-loop algorithms; On-line control; Robots.

I. INTRODUCTION

Robot control actions are naturally executed in the joint space, whilst robot motion trajectories are better specified in the task space. Usually, the location of the robot's end-effector (EE) is commanded to vary as a function of time along a given path. Three Cartesian coordinates describe the EE position and three angular coordinates (e.g. Euler angles) characterise the EE orientation in a *six*-dimensional task-space.[1] Let then the direct kinematic equation of a manipulator with arbitrary structure be known; the problem we want to solve in this paper can be formulated as follows:

> Assigned an EE trajectory (position + orientation), find a (PVA) joint position + velocity + acceleration joint trajectory which is a solution to the direct kinematic equation, i.e. it reproduces the desired motion at the EE.

Most approaches proposed in the literature are aimed at solving the inverse kinematic problem (IKP) for a given constant EE location. It has been well-known for a long time that analytical inverse kinematic solutions exist only for special manipulator geometries,[2,3] such as the so-called wrist-partitioned type of manipulator.

For all those structures which are not solvable in closed-form, a number of numerical techniques have been proposed most of which are based on the computation of the manipulator's Jacobian. An iterative method based on a nonlinear optimisation algorithm which uses a modified Newton–Raphson method has been proposed by Goldenberg *et al.*[4] Angeles[5] derived a numerical method from the formulation of invariants in the rotational part of the so-called closure equations. Another technique by Lenarčič[6] solves the IKP by using the conjugate gradient method. A conceptually different approach by Tsai and Morgan[7] is based on the use of continuation methods. The resulting computationally lengthy technique has recently been simplified by Manseur and Doty[8] when applied to the so-defined orthogonal manipulators.

On the other hand, solving the IKP along a trajectory is of crucial importance in order to provide the robot controlled in the joint space with the reference joint trajectories to be tracked. Here, computation time becomes a primary concern for those on-line sensor-driven tasks when computing the inverse kinematics at the same rate as the joint servo rate becomes a must.[3] Since the pioneering resolved motion rate technique,[9] computationally efficient velocity and acceleration inverse kinematic solutions have been derived for wrist-partitioned geometries by Featherstone[10] and Hollerbach and Sahar,[11] respectively. All these techniques are inherently *open-loop* computational methods which suffer from problems with long-term drift and initial EE location errors.

A rather different approach to the solution of the IKP is obtained by constructing a simple *closed-loop* dynamic system, whose input is the desired EE trajectory and whose outputs are the joint trajectories which give the desired motion at the EE. The original idea was independently proposed by Balestrino *et al.*[12] and Wolovich and Elliott[13] for solving only the position component of the EE trajectory. A closed-loop scheme based on the computation of the Jacobian *transpose* was devised which generates the joint displacements and velocities while guaranteeing a null positional error and a norm-bounded tracking error.

The non-trivial extension to account for the orientation component of the EE trajectory has been described

* This paper is based on work supported by the *Ministero dell 'Università e della Ricerca Scientifica e Technologica Pubblica Istruzione* under MPI 40% and 60% funds.

in Siciliano[14] and Balestrino *et al.*,[15] and the investigation of a special non-solvable structure has been presented by Sciavicco and Siciliano.[16] The application to the case of robots with redundancy has been discussed by Sciavicco and Siciliano[17,19,20] and Sciavicco *et al.*[18] The issue of kinematic singularity robustness of the scheme has recently been addressed by Chiacchio and Siciliano.[21] If the EE location is assumed to be constant, closed-loop schemes based on the same concept of Jacobian transpose have been designed in Asada and Slotine[22] for general positional tasks, and in Das *et al.*[23] for redundant manipulators.

The convergence of the above schemes is ensured by Lyapunov stability theory which leads to establishing estimates of the region of attractiveness of the solutions. Also, the algorithms are remarkably based on the sole computation of direct kinematic functions, and therefore they avoid the typical numerical instabilities associated with any matrix inversion-based technique. Alternatively, closed-loop convergent schemes solving for EE position + orientation based on the computation of the Jacobian *inverse* has been suggested by Balestrino *et al.*[15] and Tsai and Orin;[24] they can be considered as the natural closed-loop extension of the resolved motion rate technique.[9] Similar is also the scheme proposed by Wolovich and Flueckiger[25] which ensures an exponentially decaying error only, and not perfect tracking as in the above two schemes.[15,24]

All the above schemes can be termed *first-order* schemes, since they solve for joint displacements and velocities. For joint space control purposes, however, it would be nice to generate joint accelerations as well, designing then a *second-order* inverse kinematic scheme. If the EE location is constant, a second-order positional scheme based on the Jacobian transpose has been proposed by Slotine and Yoerger[26] for the general case of redundant manipulators. Conversely, a so-called joint space command generator has been derived by Vaccaro and Hill[27] which is based on the Jacobian inverse, but it only solves for EE position; the same idea has also been developed for robot Cartesian control.[28]

For the most general case of an EE trajectory, a second-order scheme logically derived from resolved-acceleration control[29] has lately been obtained by Sciavicco and Siciliano[30] which requires the computation of the Jacobian inverse. More recently, Siciliano[31] and Novaković[32] have independently established a new second-order scheme based on the use of the Jacobian transpose which makes use of the sliding mode theory.[33] Their scheme can be utilised, however, only for solving EE position.

In this paper we propose a new *second-order tracking scheme* which solves EE trajectories (position + orientation) in terms of joint PVA trajectories by means of a fast convergent algorithm which only requires the computation of the direct kinematic function and of the Jacobian of the manipulator. The definition of an orientation error which is consistent with the adoption of EE angular velocities as task space variables directly follows from the first-order scheme recently proposed by Chiacchio and Siciliano.[34]

A PUMA-like manipulator is chosen to develop a numerical example. Being this geometry wrist-partitioned, the EE position IKP is solved separately from the EE orientation IKP. The limited number of computations required allows for a 1 kHz solution rate on condition that a high-speed floating-point arithmetic processor is adopted. The solutions obtained with the Jacobian transpose technique are then compared with those obtained with the more computationally demanding Jacobian inverse technique in order to illustrate the power of the proposed algorithm. The tracking errors will be shown to be less than 0.45 mm for position with an average EE velocity of 1 m/s, and less than 0.35° for orientation with an average EE velocity of 180°/s. Also, the convergence of the Jacobian transpose algorithm is tested in the neighbourhood of a double singular configuration.

II. THE INVERSE KINEMATIC PROBLEM (IKP)

It is well-known[3] that the manipulator's EE location can be described as a function of time t by a position vector $\mathbf{p}(t)$ and an orientation matrix $R(t) = (\mathbf{n}(t)\ \mathbf{s}(t)\ \mathbf{a}(t))$; $\mathbf{p}$ is the vector pointing from a reference base frame to an EE frame, and $\mathbf{n}$, $\mathbf{s}$, and $\mathbf{a}$ are the normal, slide, and approach unit vectors of the EE frame expressed in the base frame coordinates. The direct kinematic equation of the manipulator defines the transformation from the n-dimensional vector of joint displacements $\mathbf{q}$ to the vectors of EE location $(\mathbf{p}, R)$ as

$$\mathbf{p}(t) = \mathbf{p}(\mathbf{q}(t)) \quad R(t) = R(\mathbf{q}(t)). \tag{1}$$

Given a desired EE location $(\mathbf{p}_d, R_d)$, the IKP can be stated as that to find a solution $\mathbf{q}_d$ to (1).

In manipulator kinematics it is of interest also the mapping from the vector of joint velocities $\dot{\mathbf{q}}$ to the vector of EE velocities $\mathbf{v}$, i.e.

$$\mathbf{v}(t) = \begin{pmatrix} \dot{\mathbf{p}}(t) \\ \omega(t) \end{pmatrix} = J(\mathbf{q})\dot{\mathbf{q}}(t) \tag{2}$$

through the $(6 \times n)$ Jacobian matrix. In (2), $\dot{\mathbf{p}}$ is the vector of EE linear velocities obtained as the time derivative of $\mathbf{p}$ in (1), and ω is the vector of EE angular velocities. As a consequence, the matrix $J(\mathbf{q})$ can be thought as partitioned into two $(3 \times n)$ matrices, i.e.

$$J(\mathbf{q}) = \begin{pmatrix} J_p(\mathbf{q}) \\ J_o(q) \end{pmatrix}. \tag{3}$$

Notice that an appropriate orientation counterpart for $\mathbf{p}$ which represents $\int \boldsymbol{\omega}\, dt$ cannot be defined.[3]

Assigned a desired EE velocity vector $\mathbf{v}_d$, equation (2) can be solved for the joint velocity vector $\dot{\mathbf{q}}_d$ according to the so-called resolved rate technique[9] as

$$\dot{\mathbf{q}}_d(t) = J^{-1}(\mathbf{q}_d)\mathbf{v}_d(t) \tag{4}$$

which, once $\mathbf{q}_d(0)$ is known, can be integrated over time to provide $\mathbf{q}_d(t)$. In (4) it is assumed that an inverse to J does exist for all $\mathbf{q}_d$'s; a pseudo-inverse must be used if the Jacobian degenerates or if the manipulator is redundant.[35]

Differentiating both sides of (2) with respect to time yields the mapping from the vector of joint accelerations $\ddot{\mathbf{q}}$ to the vector of EE accelerations $\dot{\mathbf{v}}$, i.e.

$$\dot{\mathbf{v}}(t) = \dot{J}(\mathbf{q})\dot{\mathbf{q}}(t) + J(\mathbf{q})\ddot{\mathbf{q}}(t) \quad (5)$$

where $\dot{J} = \partial J / \partial t$. Solving (5) for the joint accelerations $\ddot{\mathbf{q}}_d$, similarly to (4), according to the so-called resolved acceleration technique[29] gives

$$\ddot{\mathbf{q}}_d(t) = J^{-1}(\mathbf{q}_d)(\dot{\mathbf{v}}_d(t) - \dot{J}(\mathbf{q}_d)\dot{\mathbf{q}}_d(t)) \quad (6)$$

which, once $\mathbf{q}_d(0)$ and $\dot{\mathbf{q}}_d(0)$ are known, can be integrated over time to provide $\mathbf{q}_d(t)$ and $\dot{\mathbf{q}}_d(t)$.

Having obtained the desired PVA joint trajectories, one can design a computed torque-like controller.[22] If the robot operates in a sensor-based fashion, however, it is crucial to generate the joint trajectories on-line at a minimum of several hundred Hertz, ten to twenty times the robot structural resonant frequency,[3] such that satisfactory trajectory tracking is obtained.

III. CLOSED-LOOP FORMULATION OF THE IKP

A conceptually different approach to the solution of the IKP which is independent of the particular robot geometry is illustrated in the following. The idea is to reformulate the IKP as a tracking problem for a simple *closed-loop* dynamic system, whose input is the desired EE trajectory and whose outputs are the joint trajectories. This approach opposes the computational method based on (6) which is an *open-loop* style method, thus overcoming drawbacks like long-term drift and initial EE location errors.

Several *first-order* schemes based on this idea which solve for joint displacements and velocities have been proposed in the literature. They can be distinguished into those based on the computation of the Jacobian *transpose*[12–23,34] and those based on the computation of the Jacobian *inverse*.[24,25,27,28] Only some of the above schemes, however, do explicitly account for the EE orientation[14–16,24,25,34] which is not a trivial problem, as discussed in the previous section. The schemes based on the inverse give better results than those based on the transpose, but they are more computationally demanding and may fail in the neighbourhood of kinematic singularities. It is to be mentioned, also, that the schemes based on the Jacobian transpose can be suitably extended to redundant manipulators without noticeable computational efforts,[17–20,23] whereas a pseudoinverse is usually required for the others.[24,25,27,28] The reader is referred to the wide list of references provided at the end of the paper for more details concerning first-order schemes.

The goal of this section is to derive a new *closed-loop second-order tracking scheme* which solves the IKP for EE position + orientation trajectories. Partial solutions based either on the Jacobian transpose[26,31,32] or on the Jacobian inverse[27,28,30] have been proposed, but none of them accounts for the EE orientation.

Let then $\mathbf{e}(t)$ denote a *six*-dimensional error vector between the desired EE location $(\mathbf{p}_d, R_d)$ and the actual EE location $(\mathbf{r}, R)$ which can be computed from the current joint configuration vector $\mathbf{q}$ via (1). Notice that $\mathbf{q}$ is not to be interpreted as sensed at the robot joint actuators, but it is just algorithmically computed. The error $\mathbf{e}$ is though of as partitioned into

$$\mathbf{e}(t) = \begin{pmatrix} \mathbf{e}_p(t) \\ \mathbf{e}_o(t) \end{pmatrix} \quad (7)$$

where $\mathbf{e}_p$ denotes an EE position error and $\mathbf{e}_o$ an EE orientation error.

The definition of $\mathbf{e}_p$ is straightforward, i.e.

$$\mathbf{e}_p(t) = \mathbf{p}_d(t) - \mathbf{p}(t) \quad (8)$$

whereas the definition of $\mathbf{e}_o$ is the same as originally proposed by Luh *et al.*,[29], i.e.

$$\mathbf{e}_o(t) = \tfrac{1}{2}(\mathbf{n}(t) \times \mathbf{n}_d(t) + \mathbf{s}(t) \times \mathbf{s}_d(t) + \mathbf{a}(t) \times \mathbf{a}_d(t)) \quad (9)$$

where $(\mathbf{n}, \mathbf{s}, \mathbf{a})$ and $(\mathbf{n}_d, \mathbf{s}_d, \mathbf{a}_d)$ denote the actual and the desired unit vector triples of the EE frame, respectively.

It should be remarked that the desired EE orientation is usually described by a minimal number of coordinates, typically three Euler angles (ψ, θ, φ). The desired unit vector triple of the EE frame $(\mathbf{n}_d, \mathbf{s}_d, \mathbf{a}_d)$ can then be computed through the rotation matrix associated with the Euler angles representation.[1]

Additionally, let $\dot{\mathbf{e}}(t)$ denote the error vector between the desired EE velocity vector $\mathbf{v}_d$ and the actual EE velocity vector $\mathbf{v}$ which can be computed from the current joint velocity vector $\dot{\mathbf{q}}$ via (2). The partition of $\dot{\mathbf{e}}$ follows accordingly to (7), i.e.

$$\dot{\mathbf{e}}(t) = \begin{pmatrix} \dot{\mathbf{e}}_p(t) \\ \dot{\mathbf{e}}_o(t) \end{pmatrix}. \quad (10)$$

It is easy to see that

$$\dot{\mathbf{e}}_p(t) = \dot{\mathbf{p}}_d(t) - \dot{\mathbf{p}}(t). \quad (11)$$

As regards the orientation error, it can be shown that†

$$\dot{\mathbf{e}}_o(t) = \omega_d(t) - \omega(t). \quad (12)$$

The second-order closed-loop scheme of Figure 1 can be devised. If the control law $\ddot{\mathbf{q}}(t)$ is chosen so that the system is guaranteed to be stable, i.e. $\mathbf{e}(t)$ asymptotically tends to zero, it can be concluded that the system performs a second-order kinematic inversion; namely, given $(\mathbf{p}_d(t), R_d(t))$, $\mathbf{v}_d(t)$, and $\dot{\mathbf{v}}_d(t)$ if needed, the scheme generates $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$, $\ddot{\mathbf{q}}(t)$ which are "infinitely close" to $\mathbf{q}_d(t)$, $\dot{\mathbf{q}}_d(t)$, $\ddot{\mathbf{q}}_d(t)$, respectively.

III.A The Jacobian inverse scheme

A first result can now be established which follows from Sciavicco and Siciliano,[30] but it is extended to include EE orientation; the explicit time dependence will often be suppressed for notation compactness.

Proposition 1. If the matrix $J(q)$ has full rank for all Joint configurations $\mathbf{q}$'s, the control law

$$\ddot{\mathbf{q}} = J^{-1}(\mathbf{q})(\dot{\mathbf{v}}_d(t) - \dot{J}(\mathbf{q})\dot{\mathbf{q}} + K_p\mathbf{e} + K_v\dot{\mathbf{e}}) \quad (13)$$

† In fact, even if $\int \omega \, dt$ is not defined, Luh *et al.*[29] argued that, for small errors, $\dot{\mathbf{e}}_o$ is the true time derivative of $\mathbf{e}_o$. Later, this has been demonstrated by Yuan[36] through the use of quaternions.

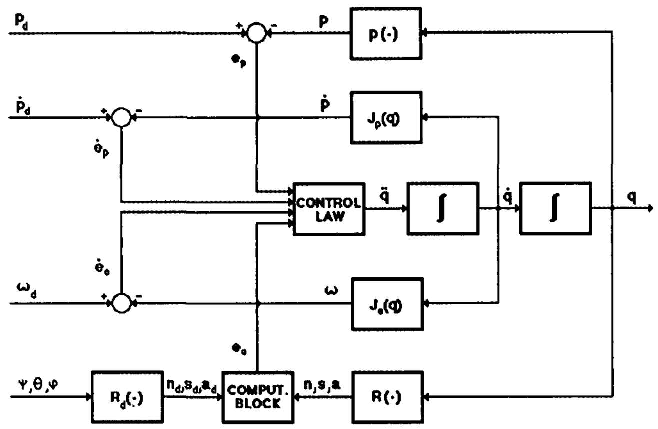


Fig. 1. The closed-loop IKP tracking solution scheme.

with K_p and K_v positive definite diagonal matrices such that the matrix

$$M := \begin{pmatrix} 0 & I \\ -K_p & -K_v \end{pmatrix} \tag{14}$$

is a Hurwitz matrix, ensures that $\mathbf{e}(t) \to \mathbf{0}$ as $t \to \infty$, if $\mathbf{e}(0) \neq \mathbf{0}$, and $\mathbf{e}(t) = \mathbf{0}$ along the EE trajectory if $\mathbf{e}(0) = \mathbf{0}$.

Proof: Differentiating (10) with respect to time and accounting for (5) yields

$$\ddot{\mathbf{e}} = \dot{\mathbf{v}}_d - \dot{J}(\mathbf{q})\dot{\mathbf{q}} - J(\mathbf{q})\ddot{\mathbf{q}}. \tag{15}$$

Direct substitution of (13) in (15) gives

$$\ddot{\mathbf{e}} + K_v\dot{\mathbf{e}} + K_p\mathbf{e} = \mathbf{0}. \tag{16}$$

In force of (14), the convergence of the error $\mathbf{e}$ to zero can be suitably "shaped". Convergence of the position error $\mathbf{e}_p$ needs not to be further explained. Convergence of the orientation error $\mathbf{e}_o$ is ensured except for the singular case when the actual EE orientation differs from the desired EE orientation by an Euler rotation of 180°. This corresponds to having $(\mathbf{n}, \mathbf{s}, \mathbf{a}) = (-\mathbf{n}_d, -\mathbf{s}_d, -\mathbf{a}_d)$, with $\mathbf{e}_o = \mathbf{0}$. Therefore, as long as this case does not occur at $t = 0$, equation (16) guarantees that the orientation error will asymptotically tend to zero. *End of Proof.*

III.B The Jacobian transpose scheme

The scheme that we derive in the following actually originates from the work independently developed by Siciliano[31] and Novaković.[32] The convergence of the scheme is proved by means of Lyapunov stability theory.

Proposition 2. The control law

$$\ddot{\mathbf{q}} = \left(1 + \frac{\mathbf{s}^T K^T \mathbf{z}}{\mathbf{s}^T K^T J(\mathbf{q}) J^T(\mathbf{q}) K \mathbf{s}}\right) J^T(\mathbf{q}) K \mathbf{s} \tag{17}$$

where K is a positive definite diagonal matrix,

$$\mathbf{s} = \dot{\mathbf{e}} + \Lambda\mathbf{e} \tag{18}$$

with Λ being a positive definite diagonal matrix, and

$$\mathbf{z} = \dot{\mathbf{v}}_d - \dot{J}(\mathbf{q})\dot{\mathbf{q}} + \Lambda\dot{\mathbf{e}}, \tag{19}$$

ensures that $\mathbf{s}(t) \to \mathbf{0}$ as $t \to \infty$, if $\mathbf{s}(0) \neq \mathbf{0}$, and $\mathbf{s}(t) = \mathbf{0}$ along the EE trajectory if $\mathbf{s}(0) = \mathbf{0}$. Also, the convergence of $\mathbf{s}$ to zero ultimately implies the convergence of $\mathbf{e}$ to zero, by virtue of (18).

Proof: Define the positive Lyapunov function candidate of the error *sliding* vector $\mathbf{s}$ in (18) as

$$v = \tfrac{1}{2}\mathbf{s}^T K \mathbf{s}. \tag{20}$$

Its time derivative along the trajectories of the system (15) results in

$$\dot{v} = \mathbf{s}^T K^T \mathbf{z} - \mathbf{s}^T K^T J(\mathbf{q})\ddot{\mathbf{q}} \tag{21}$$

with $\mathbf{z}$ defined in (19). Direct substitution of (17) in (21) gives

$$\dot{v} = -\mathbf{s}^T K^T J(\mathbf{q}) J^T(\mathbf{q}) K \mathbf{s} \leq 0 \tag{22}$$

which in turn implies that $\mathbf{e}(t) \to \mathbf{0}$. In detail, since v is lower bounded by zero, in force of the positive definiteness of K in (20), $\mathbf{s}$ is bounded. This implies that $\mathbf{e}$ and $\dot{\mathbf{e}}$ in (18) are bounded, being Λ positive definite too. If $J(q)J^T(q)$ is guaranteed to be uniformly positive definite, equation (22) ensures that $\mathbf{s}(t) \to \mathbf{0}$, and therefore $\mathbf{e}(t) \to \mathbf{0}$.

A crucial point then remains the positive definiteness of $J(\mathbf{q})J^T(\mathbf{q})$, which is not guaranteed when the manipulator is in a singular configuration and J is not a full-rank matrix. From (22) it can be seen that the condition $\dot{v} = 0$ implies $\mathbf{e} = \mathbf{0}$, except when the vector $K\mathbf{s}$ belongs to the null space of the matrix $J^T(\mathbf{q})$, where the algorithm may in principle get "stuck". One can easily show, however, that such equilibrium point is unstable, and the time evolution of the desired EE trajectory will contribute to decrease $\dot{v}$ again.[26]

Similar remarks for the convergence of the orientation error as for the Jacobian inverse scheme of Proposition 1 are in order also in this case. *End of Proof.*

Notice that the control law (17) slightly differs from the analogous solution established by Novaković.[32] There, the resulting equation for the stability is of the kind $\dot{v} = -\alpha v$ with $\alpha > 0$ which allows for prescribing a desired solution settling time. This might be advantageous if $\mathbf{s}(0) \neq \mathbf{0}$.

It is worth remarking here that, from the conceptual viewpoint, the schemes just presented can be respectively regarded as closed-loop versions of the well-known Newton type method and steepest descent method, but here the convergence is established beforehand though.

A nice feature of the Jacobian transpose scheme over the Jacobian inverse scheme is that its computational burden is reduced. Also, it avoids the numerical instabilities associated with matrix inversion near kinematic singularities. One drawback of the scheme in Proposition 2, however, is that it introduces, in the neighbourhood of $\mathbf{s} = \mathbf{0}$, an equivalent gain which tends to ∞. The analysis of the second term generated on the right hand side of (17), indeed, reveals that this is given by the ratio of two quantities that go to zero, as $\mathbf{s} \to \mathbf{0}$, with the same order *two*. This generates a "chattering" behaviour in the time evolution of the joint accelerations, which will be visible in the numerical results reported in Section IV.

Several remedies are possible against chattering, as suggested by Novaković[32] for instance. The most attractive from the computational viewpoint is to simplify the control law (17) into[31]

$$\ddot{\mathbf{q}} = J^T(\mathbf{q})K\mathbf{s} \tag{23}$$

which implies

$$\dot{v} = \mathbf{s}^T K^T \mathbf{z} - \mathbf{s}^T K^T J(\mathbf{q}) J^T(\mathbf{q}) K\mathbf{s}. \tag{24}$$

If $\mathbf{z}$ is assumed to be norm-bounded, one should choose K large enough to ensure that $\dot{v} \le 0$. In practice, however, when $\|J^T K\mathbf{s}\|$ becomes less than a significantly small positive number, it is $\dot{v} > 0$. Thus, it can be concluded that the control law (23) no longer guarantees asymptotic stability, but only ultimate boundedness of the tracking error.

Nonetheless, it might be argued that the assumption on the norm-boundedness of $\mathbf{z}$ cannot be automatically guaranteed *a priori*. More precisely, one should assume that the three terms on the right-hand side of (19) be all norm-bounded. For the first term, it is natural to assume that $\dot{\mathbf{v}}_d$ is norm-bounded. The second term $\dot{J}(\mathbf{q})\dot{\mathbf{q}}$ can be put in the form[37] $H_1(\mathbf{q})[\dot{\mathbf{q}}\dot{\mathbf{q}}] + H_2(\mathbf{q})[\dot{\mathbf{q}}^2]$, with $[\dot{\mathbf{q}}\dot{\mathbf{q}}] = (\dot{q}_1\dot{q}_2\,\dot{q}_1\dot{q}_3 \cdots \dot{q}_{n-1}\dot{q}_n)^T$, $[\dot{\mathbf{q}}^2] = (\dot{q}_1^2\,\dot{q}_2^2 \cdots \dot{q}_n^2)^T$, and H_1, H_2 matrices of appropriate dimensions. The third term $\Lambda\dot{\mathbf{e}}$, finally, can be written as $\Lambda(\mathbf{v}_d - J(\mathbf{q})\dot{\mathbf{q}})$. At this point, it can be recognised that $H_1(\mathbf{q})$, $H_2(\mathbf{q})$, Λ, $J(\mathbf{q})$, and $\mathbf{v}_d$ can be all assumed to be norm-bounded. Therefore, the problem can be reduced to the norm-boundedness of $\dot{\mathbf{q}}$. In practical implementation of the algorithm, however, when the scheme is working well, i.e. $\mathbf{s}$ is small, $\mathbf{e}$ and $\dot{\mathbf{e}}$ are also small and then $\dot{\mathbf{q}}$ is expected to be bounded. A more rigorous proof cannot be given, but numerical results of Section IV will bear out this issue.

Incidentally, it should be remarked that at steady-state, i.e. when $\mathbf{v}_d = \dot{\mathbf{v}}_d = \mathbf{0}$, the control law (23) ensures asymptotic stability. To see this, define the positive definite Lyapunov function[26,31]

$$v = \tfrac{1}{2}(\mathbf{e}^T K \Lambda \mathbf{e} + \dot{\mathbf{q}}^T \dot{\mathbf{q}}). \tag{25}$$

Its time derivative along the trajectories of the system (11) and (12) with $\dot{\mathbf{p}}_d = \boldsymbol{\omega}_d = \mathbf{0}$, under the control (23), results in

$$\dot{v} = -\dot{\mathbf{e}}^T K\mathbf{s} + \dot{\mathbf{e}}^T K \Lambda \mathbf{e} \tag{26}$$

which, by virtue of (18), becomes

$$\dot{v} = -\dot{\mathbf{e}}^T K \dot{\mathbf{e}} \le 0 \tag{27}$$

implying that $\dot{\mathbf{e}} \to \mathbf{0}$, and then $\dot{\mathbf{q}} \to \mathbf{0}$ and $\mathbf{e} \to \mathbf{0}$.

Furthermore, an appealing feature of the solution (23) lies in the following intuitive physical interpretation.[26] It is well known that the relationship between the vector $\boldsymbol{\tau}$ of generalized joint forces and the corresponding vector $\boldsymbol{\gamma}$ of generalized EE forces is given by[3]

$$\boldsymbol{\tau} = J^T(\mathbf{q})\boldsymbol{\gamma} \tag{28}$$

which can be obtained by applying the principle of virtual work and accounting for the "dual" mapping (2).

As a consequence, the control law (23), with $\mathbf{s}$ as in (18), is analogous to applying an *elastic force* $K\Lambda\mathbf{e}$ and a *damping force* $K\dot{\mathbf{e}}$ at the EE of an *ideal* manipulator with the same kinematic structure as the manipulator of interest, but having a unitary inertia matrix and operating in the absence of gravity. This in turn corresponds to apply an *impedance control* scheme[38] to the above ideal manipulator with simple dynamics.

In force of this analogy, it can be recognised, for instance, that in the case when the vector $K\mathbf{s}$ is in the null space of the matrix J^T discussed above, this is equivalent to applying EE forces in the direction along which the manipulator cannot move.

We would emphasise again, however, that the purpose of the above presented schemes is *only* to numerically solve for the IKP and not to design a robot task space control. In other words, the joint variables $\mathbf{q}$ and $\dot{\mathbf{q}}$ to feed back in the scheme of Figure 1 are *not* those sensed by the robot but just those computed by the algorithm.

Another remark is in order concerning the practical implementation of the algorithm. The solution (23) suggests that the tracking error can be made arbitrarily small by choosing K large enough. It should be pointed out, however, that the implementation of the discrete-time solution algorithm, through the sampling rate, limits the maximum values allowable for K. In order to establish an optimum for that value, a discrete-time stability proof should be undertaken, as done for instance in Das *et al.*,[23] but this goes beyond the scopes of the present work.

IV. CASE STUDY

The schemes presented in the previous section have been applied to solve the IKP along given EE position + orientation trajectories for a *six*-degree-of-freedom PUMA-like manipulator with zero offsets. Being this geometry wrist-partitioned, it is possible to solve the IKP into two stages; first for EE position through the first three joint variables, then for EE orientation through the last three joint variables.

A numerical example has been worked out. The desired EE task is assigned in terms of a position trajectory $\mathbf{p}_d(t)$ and an orientation trajectory that is assumed to represent three Roll-Pitch-Yaw angles from which $R_d(t)$ is generated.

The initial configuration of the manipulator is chosen as locating the EE in the desired position and orientation, i.e. $\mathbf{e}(0) = \mathbf{0}$. The trajectories are straight lines of 0.5 m for EE position and 90° in the space of RPY angles, to be executed in a very fast time of 0.5 s; standard trapezoidal velocity profiles are imposed.

The proposed inverse kinematic schemes have then been applied at the same solution rate in order that comparisons be significant. The solution rate is then set up accordingly to the simplified Jacobian transpose scheme which requires the least number of computations. As a matter of fact, the higher the solution rate the better the tracking performance expected for that scheme. It is estimated that, if a high-speed floating-point arithmetic processor is available, a solution rate of 1 kHz is sufficient to perform all the computations required by the solution (23).

The numerical integration method used for computing joint displacements $\mathbf{q}$ and joint velocities $\dot{\mathbf{q}}$ from joint accelerations $\ddot{\mathbf{q}}$ is based on the Simpson rule of integration. More sophisticated integration methods have been tested, but they have been seen not to provide any consistent improvement in tracking performance.

Initially, the Jacobian inverse scheme based on the solution (13) is applied with $K_p = K_v = 0$ in an open-loop fashion. The results are illustrated in Figure 2. It is interesting to note that the position and orientation errors after $t = 0.5$ s linearly increase, since no feedback correction is active.

The feedback matrices $K_p = \mathrm{diag}(250000\ 250000\ 250000\ 62500\ 62500\ 62500)$ and $K_v = \mathrm{diag}(1000\ 1000\ 1000\ 500\ 500\ 500)$ are then introduced such that a double pole at -500 for the position error dynamics and a double pole at -250 for the orientation error dynamics in (16) are obtained. The tracking errors are reported in Figure 3 and the resulting joint accelerations in Figure 4. It is clear that the action of the feedback terms considerably improves the tracking performance and guarantees null steady-state errors. The peaks in the time evolution of the errors are not surprising, since trapezoidal velocity profiles having discontinuous accelerations have been imposed.

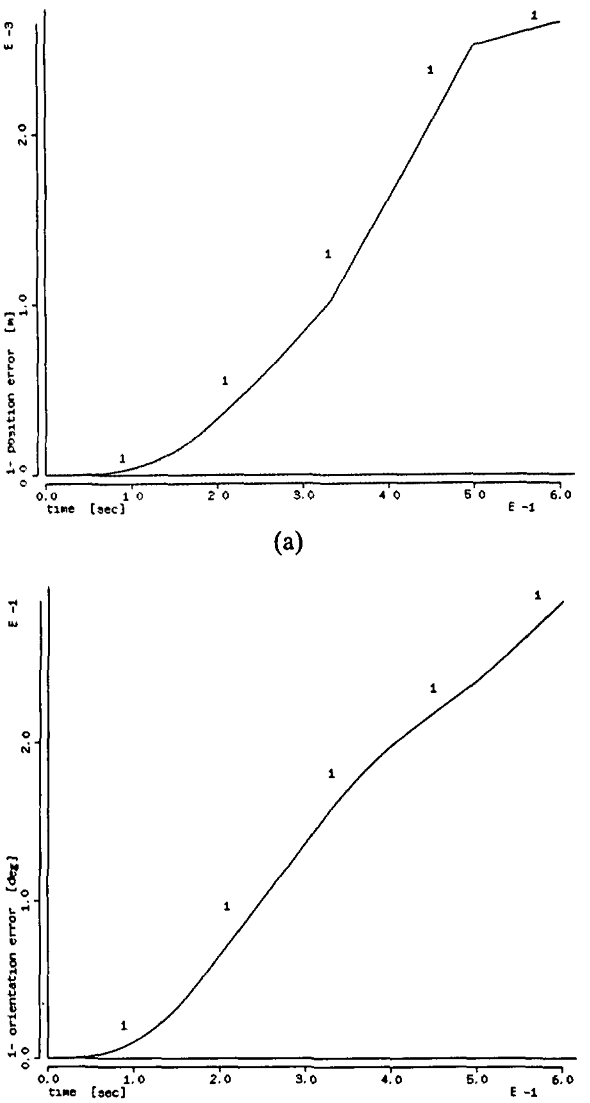


Fig. 2. Tracking errors with the open-loop Jacobian inverse scheme: a) position error, b) orientation error.

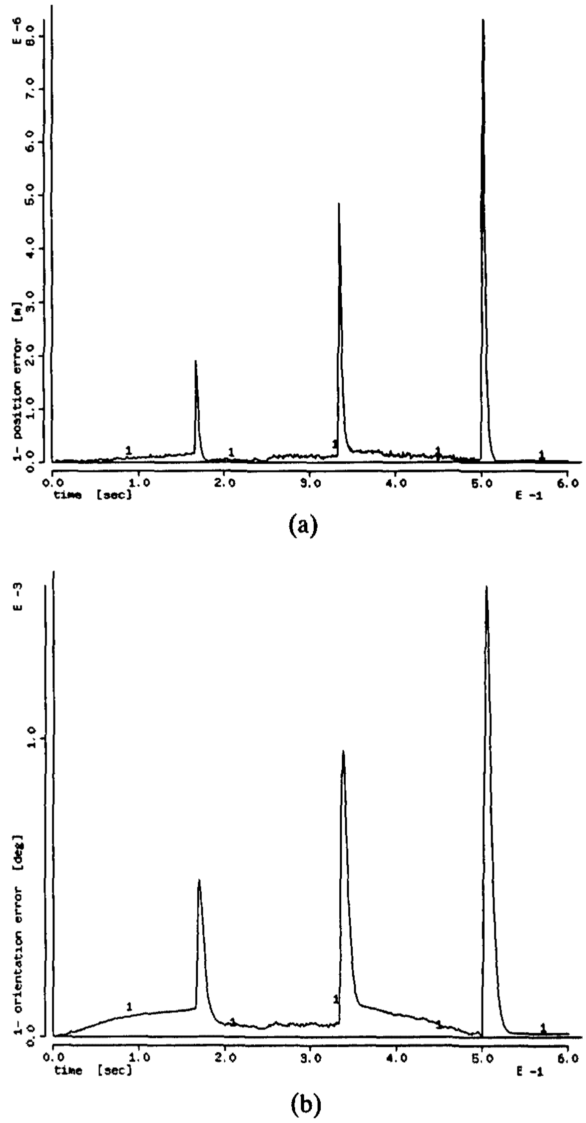


Fig. 3. Tracking errors with the closed-loop Jacobian inverse scheme: a) position error, b) orientation error.

The Jacobian transpose scheme based on the solution (17) is implemented next. In order to make a "fair" comparison, $K = \mathrm{diag}(1000\ 1000\ 1000\ 500\ 500\ 500)$ and $\Lambda = \mathrm{diag}(250\ 250\ 250\ 125\ 125\ 125)$ have been chosen so that equivalent proportional + derivative feedback actions are obtained as for the above scheme. The tracking errors are depicted in Figure 5 and the resulting joint accelerations in Figure 6. It can be seen that the tracking performance is degraded by an order of magnitude, but it is still excellent. As anticipated in theory, however, the joint accelerations chatter about some "mean-value" trajectories which actually resemble the trajectories obtained in Figure 4. However, the joint velocity and position trajectories, not displayed here, are smooth in force of the filtering nature of the integrators in cascade to the joint accelerations.

Finally, the computationally advantageous Jacobian transpose scheme based on the solution (23) is adopted with the same feedback matrices as above. As expected

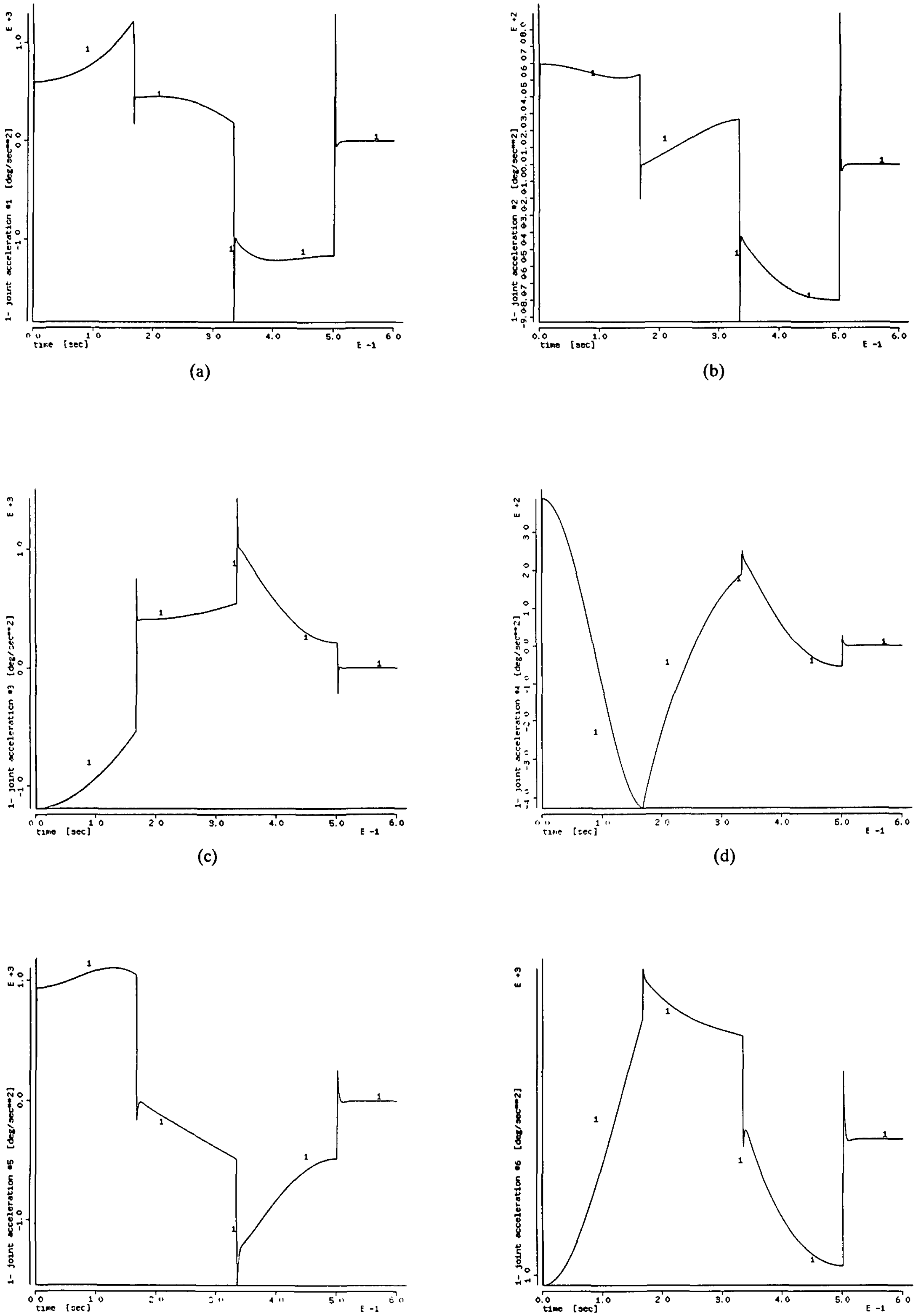


Fig. 4. Joint acceleration trajectories with the closed-loop Jacobian inverse scheme.

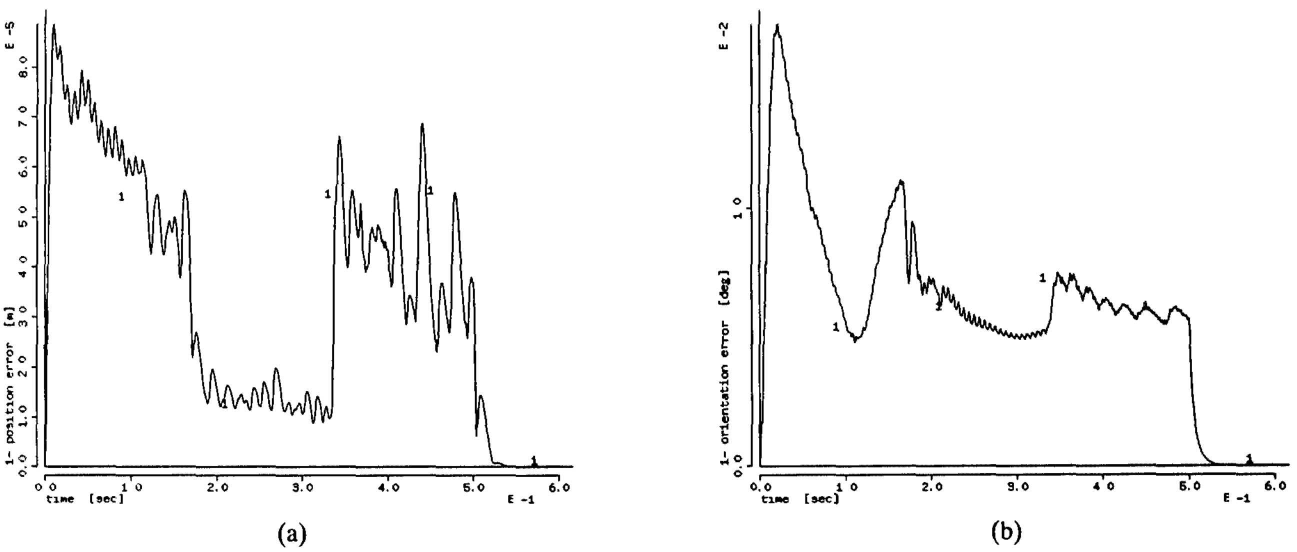


Fig. 5. Tracking errors with the closed-loop Jacobian transpose scheme: a) position error, b) orientation error.

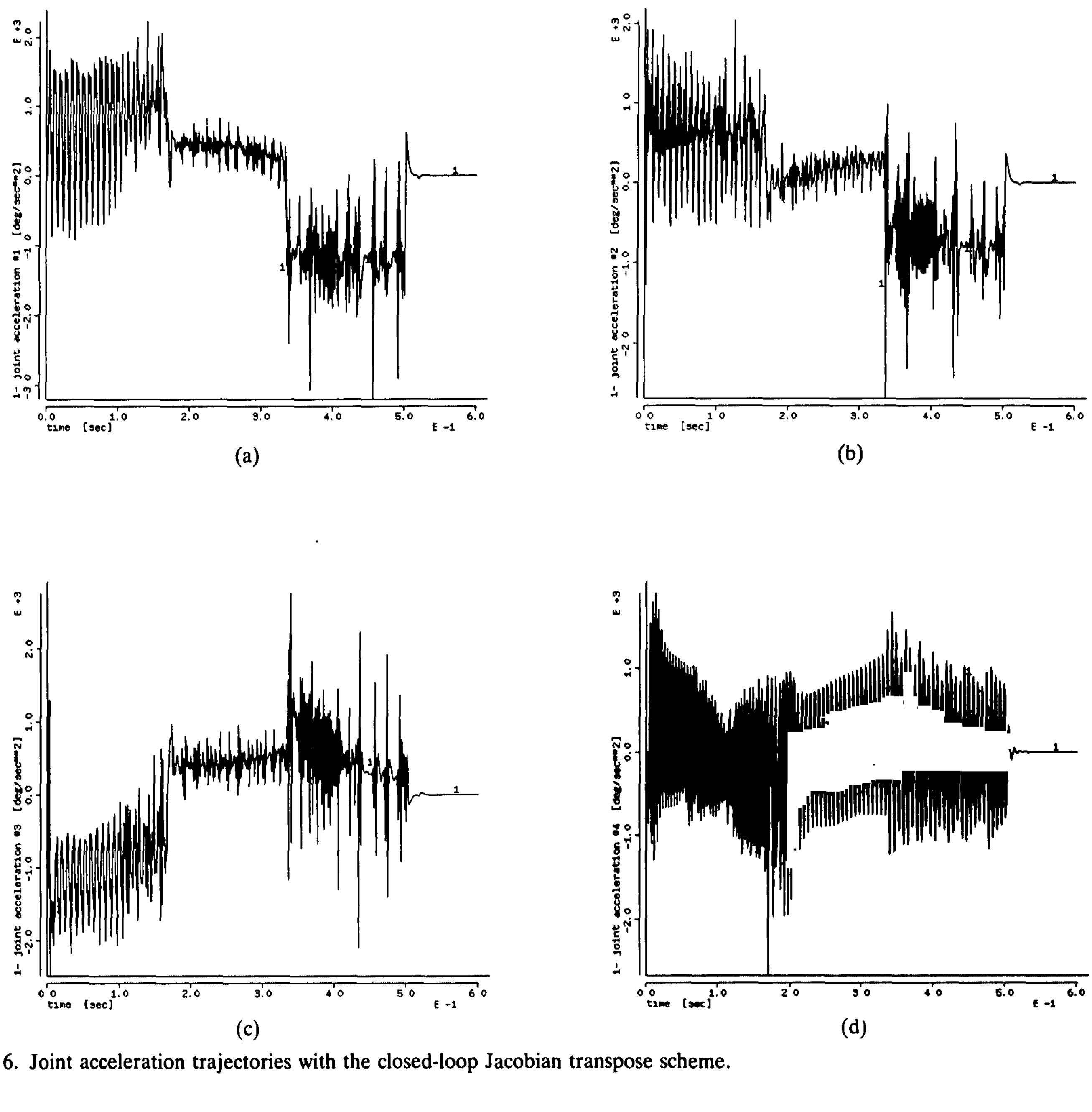
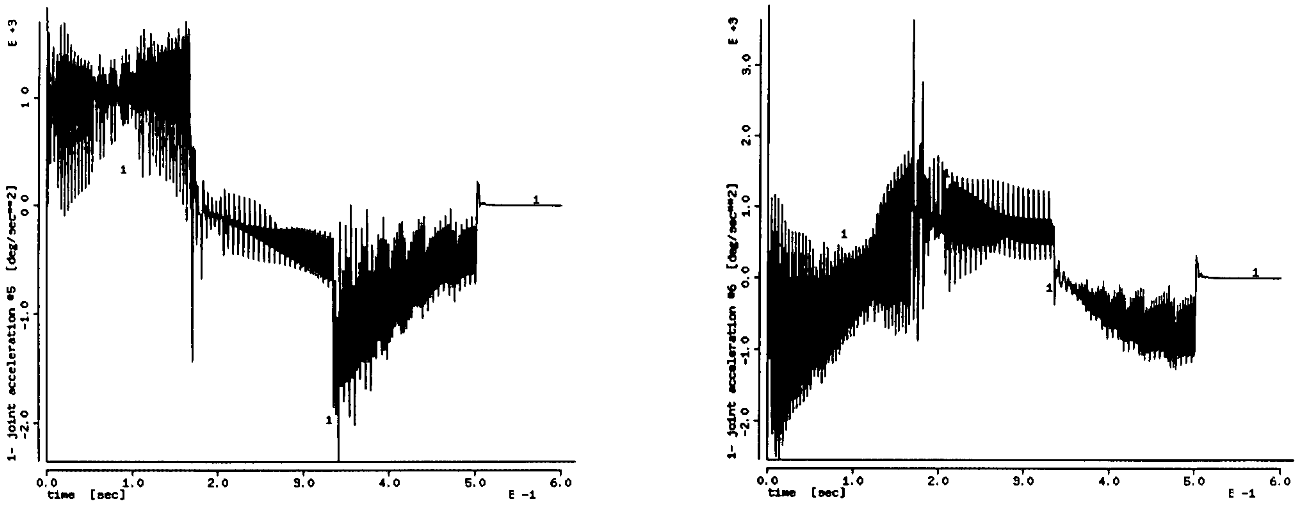
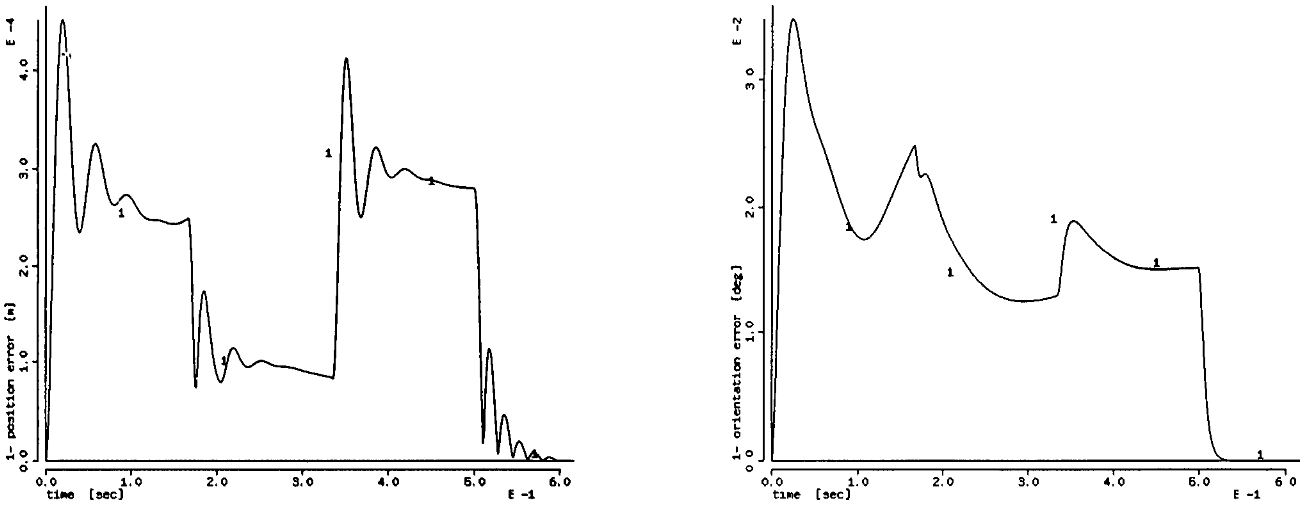


Fig. 6. Joint acceleration trajectories with the closed-loop Jacobian transpose scheme.



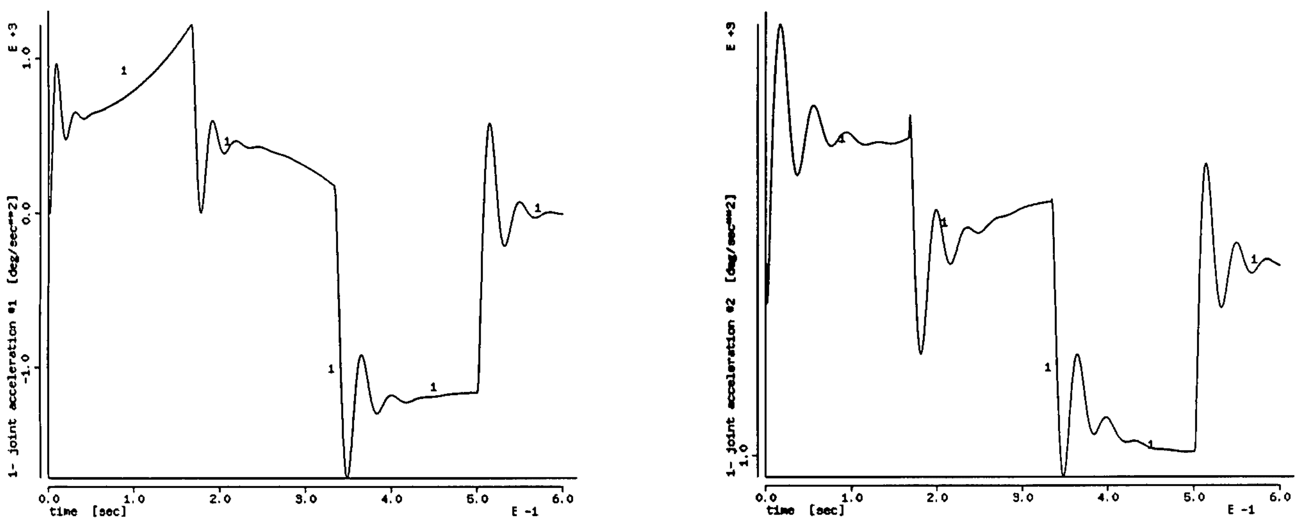
(e) (f)

Fig. 6 (*continued*)



(a) (b)

Fig. 7. Tracking errors with the simplified closed-loop Jacobian transpose scheme: a) position error, b) orientation error.



(a) (b)

Fig. 8. Joint acceleration trajectories with the simplified closed-loop Jacobian transpose scheme.

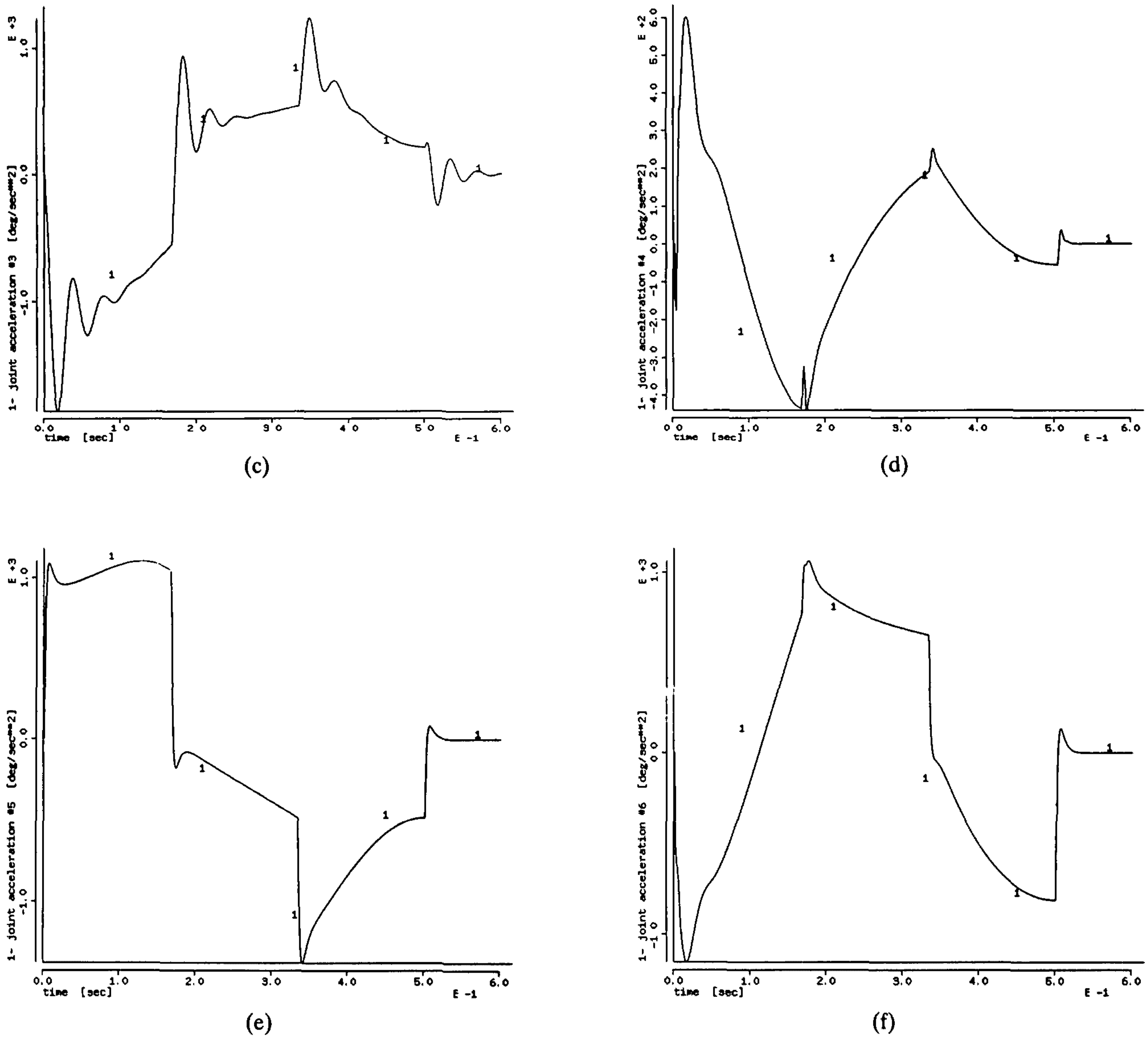


Fig. 8 (*continued*)

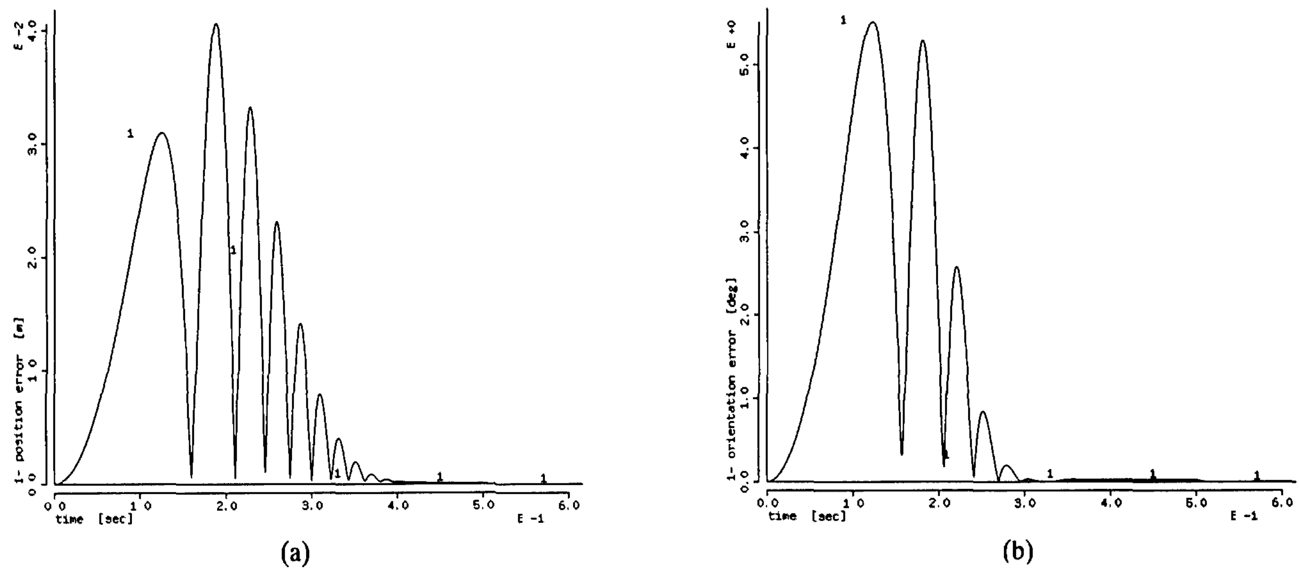


Fig. 9. Tracking errors with the simplified closed-loop Jacobian transpose scheme (the initial configuration has a double singularity): a) position error, b) orientation error.

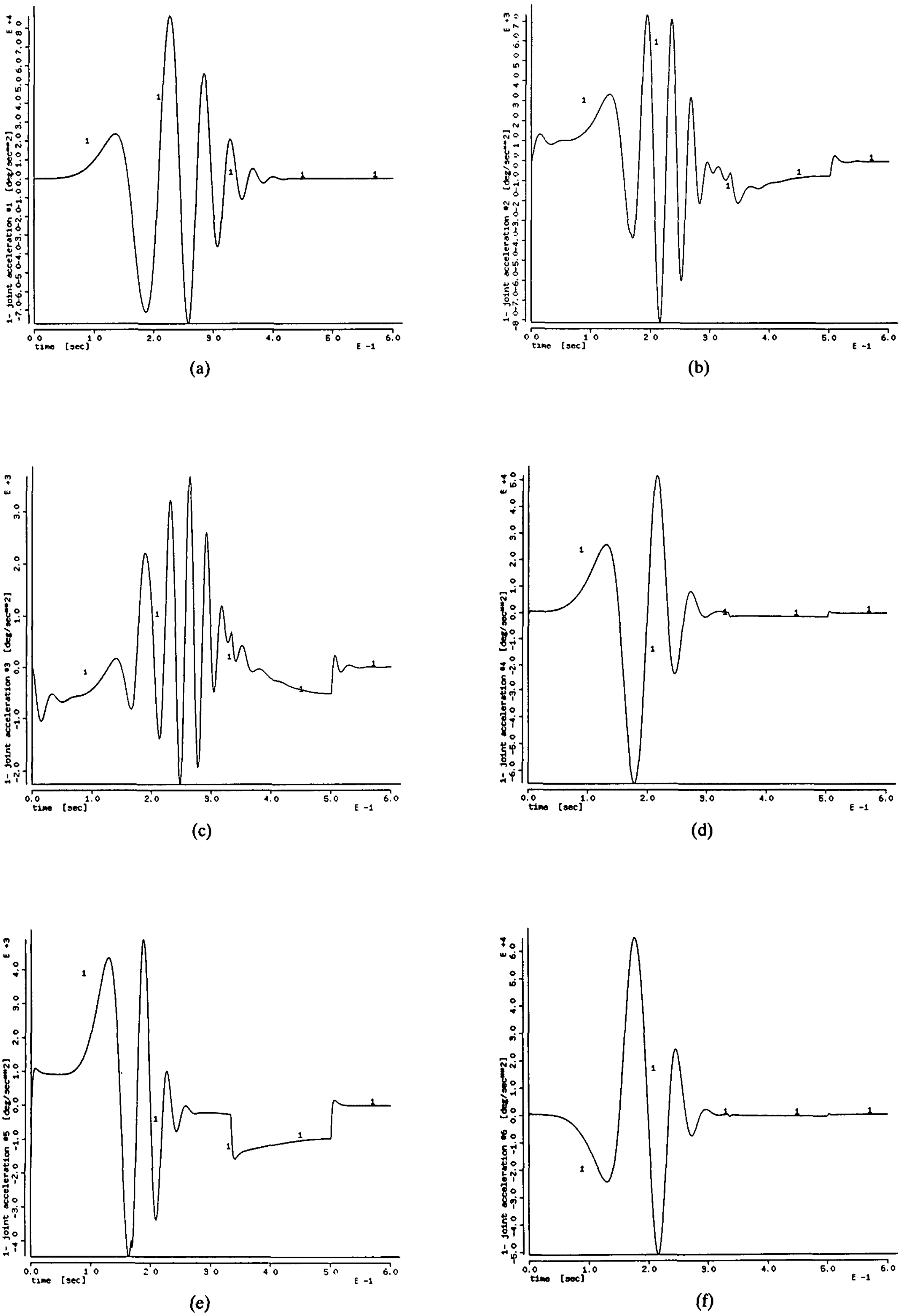


Fig. 10. Joint acceleration trajectories with the simplified closed-loop Jacobian transpose scheme (the initial configuration has a double singularity).

in theory, the tracking performance in Figure 7 is not as good as that in Figure 3, but it is still satisfactory; less than 0.45 mm with an average velocity of 1 m/s and less than 0.35°/s with an average velocity of 180°/s certainly are below the typical accuracy requirements for most current industrial robots. The joint acceleration trajectories in Figure 8 are seen to be "very close" to the trajectories in Figure 4 which can be considered as the "true" ones.

In order to test the convergence of the scheme in the neighbourhood of singular configurations, another example has been developed for the Jacobian transpose scheme (23) with unchanged feedback matrices and zero initial conditions, i.e. $\mathbf{e}(0) = \mathbf{0}$.

The initial joint configuration possesses a double singularity, namely a shoulder singularity—the wrist is located along the shoulder axis—and a wrist singularity—$q_5 = 0$, aligning the other two wrist axes.[39] It is worth recalling here that such singularities are the most troublesome ones since they may be encountered anywhere through the manipulator workspace.

The trajectories have the same path length and duration as above; the EE position trajectory is chosen as having a non-zero component orthogonal to the plane of the structure, and similarly the EE orientation trajectory is chosen as having non-zero roll and yaw components. It follows that the tracking error will have a non-zero component along the null space of the Jacobian transpose at the initial joint configuration. As anticipated in theory, this is undoubtedly a critical test for the algorithm.

From the results illustrated in Figure 9 it can be noticed that both tracking errors initially increase, showing the effort to leave the singularity, but afterwards they decrease. This behaviour is reflected by the initial higher values of joint accelerations in Fig. 10.

V. CONCLUSIONS

Closed-loop second-order tracking schemes for transforming given EE position + orientation trajectories into joint PVA trajectories have been presented in this paper. A first scheme based on the computation of the manipulator's Jacobian inverse has been derived from the concept of resolved acceleration control.

To the purpose of avoiding matrix inversion which is usually critical point in any numerical method, a new scheme has been proposed which is based on the computation of the Jacobian transpose. Asymptotic stability of the tracking error is guaranteed, so as in the Jacobian inverse scheme, but at the expenses of chattering accelerations. A computationally advantageous simplification of the scheme has been proposed which guarantees ultimate boundedness of the tracking error and asymptotic stability of the steady-state error.

A case study has been worked out for the a PUMA-like manipulator, and numerical results have clearly demonstrated that the performance achieved with the computationally fast Jacobian transpose algorithm favours the use thereof in solving the IKP for any industrial robot of arbitrary kinematic structure. Convergence around singular positions has been successfully tested.

An open research issue remains the application of the proposed scheme to kinematically redundant structures, when the number of joint variables exceeds the number of task space variables. In fact, although the extension of the scheme appears to be quite straightforward, the *null space* of the Jacobian matrix in a redundant manipulator allows for joint velocities unobservable at the output, which may generate internal motions of the structure uncontrollable though. This point, along with other related topics such as inclusion of constraints and task-priority strategies[40] for these second-order schemes, will constitute the subject of further investigation.

Acknowledgements
We would like to thank Professor Lorenzo Sciavicco for helpful discussions relating to this work.

References

1. M. Vukobratović and M. Kirćanski, *Scientific Fundamentals of Robotics 3: Kinematics and Trajectory Synthesis of Manipulation Robots* (Springer–Verlag, Berlin, Heidelberg, FRG, 1986).
2. D.L. Pieper, "The Kinematics of Manipulators Under Computer Control" *Ph.D. dissertation,* Stanford University (1968).
3. R.P. Paul, *Robot Manipulators: Mathematics, Programming, and Control* (Cambridge, MA, MIT Press, 1981).
4. A.A. Goldenberg, B. Benhabib and R.G. Fenton, "A complete generalized solution to the inverse kinematics of robots" *IEEE J Robotics and Automation* **RA-1,** No. 1, 14–20 (1985).
5. J. Angeles, "On the Numerical solution of the inverse kinematic problem" *Intern. J. Robotics Resarch* **4,** No. 2, 21–37 (1985).
6. J. Lenarčič, "An efficient numerical approach for calculating the inverse kinematics for robot manipulators" *Robotica* **3,** 21–26 (1985).
7. L.-W. Tsai and A.P. Morgan, "Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods" *ASME J. Mechanism, Transmission, and Automation in Design* **107,** No. 2, 189–200 (1985).
8. R. Manseur and K.L. Doty, "A fast algorithm for inverse kinematic analysis of robot manipulators" *Intern. J. Robotics Research* **7,** No. 3, 52–63 (1988).
9. D.E. Whitney, "Resolved motion rate control of manipulators and human prostheses" *IEEE Trans. Man-Machine Systems* **MMS-10,** No. 2, 47–53 (1969).
10. R. Featherstone, "Position and velocity transformations between robot end-effector coordinates and joint angles" *Intern. J. Robotics Research* **2,** No. 2, 35–45 (1983).
11. J.M. Hollerbach and G. Sahar, "Wrist-partitioned, inverse kinematic acceleration and manipulator dynamics" *Inter. J. Robotics Research* **2,** No. 4, 61–76 (1983).
12. A Balestrino, G. De Maria and L. Sciavicco, "Robust control of robotic manipulators" *Preprints of the 9th IFAC World Congress* **6,** Budapest, Hungary, 80–85 (July, 1984).
14. B. Siciliano, "Solution Algorithms to the Inverse Kinematic Problem for Manipulation Robots" (in Italian) *Ph.D. dissertation,* University of Naples (1986).
15. A. Balestrino, G. De Maria, L. Sciavicco and B. Siciliano, "An algorithmic approach to coordinate transformation for robotic manipulators" *Advanced Robotics* **2,** No. 4, 327–344 (1987).
16. L. Sciavicco and B. Siciliano, "Coordinate transformation:

A solution algorithm for one class of robots" *IEEE Trans. Systems, Man, and Cybernetics* **SMC-16,** No. 4, 550–559 (1986).

17. L. Sciavicco and B. Siciliano, "An inverse kinematic solution algorithm for dexterous redundant manipulators" *Proceedings of the 3rd International Conference on Advanced Robotics,* Versailles, France, 247–256 (Oct., 1987).
18. L. Sciavicco, B. Siciliano and P. Chiacchio, "On the use of redundancy in robot kinematic control" *Proceedings of the 1988 American Control Conference,* Atlanta, GA, 1370–1375 (June, 1988).
19. L. Sciavicco and B. Siciliano, "A solution algorithm to the inverse kinematic problem for redundant manipulators" *IEEE J. Robotics and Automation* **RA-4,** No. 4, 303–310 (1988).
20. L. Sciavicco and B. Siciliano, "On the solution of inverse kinematics of redundant manipulators" *Preprints of the NATO Advanced Research Workshop: Robots with Redundancy,* Salò, Italy (June/July, 1988) to be published by Springer–Verlag.
21. P. Chiacchio and B. Siciliano, "Achieving singularity robustness: An inverse kinematic solution algorithm for robot control" *IEE Control Engineering Series 36 Robot Control: Theory and Application* **16,** 149–156 (Peter Peregrinus Ltd., London, UK, 1988).
22. H. Asada and J.-J.E. Slotine, *Robot Analysis and Control* (Wiley–Interscience, New York, NY, 1986).
23. H. Das, J.-J.E. Slotine and T.B. Sheridan, "Inverse kinematic algorithms for redundant systems" *Proceedings of the 1988 IEEE International Conference on Robotics and Automation,* Philadelphia, PA, 43–48 (Apr., 1988).
24. Y.T. Tsai and D.E. Orin, "A strictly convergent real-time solution for inverse kinematics of robot manipulators" *J. Robotic Systems* **4,** No. 4, 477–501 (1987).
25. W.A. Wolovich and K.F. Flueckiger, "Inverse kinematic-based control" *Proceedings of the Workshop on Space Telerobotics,* Pasadena, CA, 165–175 (Jan., 1987).
26. J.-J.E. Slotine and D.R. Yoerger, "A rule-based inverse kinematics algorithm for redundant manipulators" *Intern. J. Robotics and Automation* **2,** No. 2, 86–89 (1987).
27. R.J. Vaccaro and S.D. Hill, "A joint-space command generator for Cartesian control of robotic manipulators" *IEEE J. Robotics and Automation* **RA-4,** No. 1, 70–76 (1988).
28. S.D. and R.J. Vaccaro, "Cartesian control of robotic manipulators with joint compliance" *Robotica* **5,** 207–215 (1987).
29. J.Y.S. Luh, M.W. Walker and R.P.C. Paul, "Resolved-acceleration control of mechanical manipulators" *IEEE Trans. Automatic Control* **AC-25,** No. 3, 468–474 (1980).
30. L. Sciavicco and B. Siciliano, "A computational technique for solving robot end-effector trajectories into joint trajectories" *Proceedings of the 1988 American Control Conference,* Atlanta, GA, 535–536 (June, 1988).
31. B. Siciliano, "Closed-loop computational schemes of robot inverse kinematics" *Proceedings of the International Meeting: Advances in Robot Kinematics,* Ljubljana, Yugoslavia, 113–121 (Sept., 1988).
32. Z.R. Novaković, "A solution of the inverse kinematics problem using the sliding mode" *Report No. 5153*, Institut Jožef Stefan, Ljubljana, Yugoslavia (Mar., 1988).
33. J.-J.E. Slotine and S.S. Sastry, "Tracking control of nonlinear systems using sliding surfaces with application to robot manipulators" *Intern. J. Control* **38,** No. 2, 465–492 (1983).
34. P. Chiacchio and B. Siciliano, "A closed-loop Jacobian transpose scheme for solving the inverse kinematics of nonredundant and redundant writs" *J. Robotic Systems* **6,** No. 5, 601–630 (1989).
35. P. Coiffet, *Robot Technology Series 1: Modelling and Control* (Kogan Page, London, UK, 1983).
36. J.S.-C. Yuan, "Closed-loop manipulator control using quaternion feedback," *IEEE J. Robotics and Automation* **RA-4,** No. 4, 434–440 (1988).
37. O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation" *IEEE J. Robotics and Automation* **RA-3,** No. 1, 43–53 (1987).
38. N. Hogan, "Impedance control: An approach to manipulation: Part II—Implementation" *ASME J. Dynamic Systems, Measurement, and Control* **107,** No. 1, 8–16 (1985).
39. F.L. Litvin, Z. Yi, V. Parenti Castelli and C. Innocenti, "Singularities, configurations, and displacement functions for manipulators," *Intern. J. Robotics Research* **5,** No. 2, 52–65 (1986).
40. Y. Nakamura, H. Hanafusa and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *Intern. J. Robotics Research* **6,** No. 2, 3–15 (1987).