A Solution Algorithm to the Inverse Kinematic Problem for Redundant Manipulators

LORENZO SCIAVICCO AND BRUNO SICILIANO

Abstract—**Redundancy represents one key towards design and synthesis of more versatile manipulators. Obstacle avoidance and limited joint range constitute two kinds of constraints which can be potentially met by a kinematically redundant manipulator. The natural scenario is the inverse kinematic problem which is certainly a crucial point for robotic manipulator analysis and control.**

Based on a recently proposed algorithmic solution technique, the inverse kinematic problem for redundant manipulators is solved in this paper. The kinematics of the manipulator is appropriately augmented in order to include the above mentioned constraints; the result is an efficient, fast, closed-loop algorithm which only makes use of the direct kinematics of the manipulator. Extensive simulation results illustrate the tracking performance for a given trajectory in the Cartesian space, while guaranteeing a collision-free trajectory and/or not violating a mechanical joint limit.

Introduction

A MANIPULATOR is termed kinematically redundant if the number of degrees of freedom (DOF's) is higher than the number of task space coordinates. The most general location of an object in the Cartesian space is completely specified by six coordinates, three for position and three for orientation. In that case, a manipulator is considered redundant if it has more than six DOF's. The space of redundant solutions can be conveniently exploited to obtain a more versatile manipulator in terms of its kinematical configuration and its interaction with the environment. In particular, redundancy can be used to meet constraints on joint range availability and/or to obtain trajectories in the joint space which are collision-free in presence of obstacles along the motion. Ultimately another challenging use of redundancy is in keeping the manipulator in a configuration which makes it as dexterous as possible, that is, also avoiding kinematic singularities.

The crucial point for robotic manipulator analysis, and then control synthesis, is the capability of transforming the task space coordinates into the joint space coordinates, that is, solving the inverse kinematic problem. The direct kinematic problem allows one to specify in a unique straightforward manner [1] the relationship between the $(n \times 1)$ joint vector q and the $(m \times 1)$ Cartesian vector x as

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

Manuscript received February 17, 1987; revised October 28, 1987. Part of the material in this paper was presented at the 1987 IEEE International Conference on Robotics and Automation, Raleigh, NC, March 31–April 3, 1987. This work was supported by the Ministero della Pubblica Istruzione.

The authors are with the Dipartimento di Informatica e Sistemistica, Università di Napoli, 80125 Napoli, Italy.

IEEE Log Number 8820159.

where f is a continuous nonlinear function, whose structure and parameters are known; it associates to each q a unique x, while the inverse mapping

$$q = f^{-1}(x) \tag{2}$$

may have many q's associated to each x and, because of complexity of (1), is hard to express analytically.

The most direct approach for solving the inverse kinematic problem is certainly to obtain a closed-form solution to (2) [2]. This does not apply to all manipulators; a sufficient condition for obtaining a closed-form solution was established by Pieper [23].

In order to overcome the drawbacks encountered in solving (1) in terms of q, an alternative technique is based on the relationship between joint velocities $\dot{q}$ and Cartesian velocities $\dot{x}$

$$\dot{x} = J(q)\dot{q} \tag{3}$$

where $J(q)$ is the Jacobian matrix $\partial f/\partial q$. The above relation can be inverted to provide the so-called Jacobian control method [3] for redundant manipulators

$$\dot{q} = J^{\dagger}(q)\dot{x} \tag{4}$$

which yields locally a minimum norm joint velocity vector in the space of least square solutions to the left-hand side of (3). The superscript † denotes the Moore–Penrose pseudoinverse given by $J^{\dagger} = J^{T}(JJ^{T})^{-1}$; this particular pseudoinverse can be adopted since $m < n$, according to (1). The solution to (3) can be also modified by the addition of a second term to (4) as

$$\dot{q} = J^{\dagger}\dot{x} + (I - J^{\dagger}J)z. \tag{5}$$

The projection operator $(I - J^{\dagger}J)$ selects the components of z which are in the space of homogeneous solutions to (3), and therefore z can be used for optimization purposes. Joint range availability is optimized in [4], and further related extensions are given in [5]. Obstacle avoidance is achieved in [6], [24], [25]. Conceptually similar are the solutions based on the use of generalized inverses proposed for minimization of actuator energy consumption [7], and obstacle avoidance [8]. A rather different approach is then proposed in [9], [10]; it is essentially an iterative technique which is based on a constrained nonlinear optimization algorithm using a modified Newton–Raphson method.

The goal of this paper is to extend the inverse kinematic solution algorithm proposed in [11], [12] to the case of redundant manipulators. It is shown how constraints on the

0882-4967/88/0800-0403$01.00

joint variables and/or constraints due to obstacles in the workspace can be systematically incorporated in the solution algorithm. The properly extended kinematics is to be evaluated [13], [28], [29]. This ensures that all the advantages of the solution algorithm are retained, such as continuity of the solution, drastic reduction of computation time, and generation of joint velocities at no additional cost. A case study for a simple planar four DOF's manipulator shows the performance of the proposed solution.

The General Solution Algorithm

The general algorithmic solution technique presented in [11], [12] is extended in the following to the case of unconstrained redundant manipulators.

The inverse kinematic problem is conceived as a dynamical one in order to obtain a general solution algorithm which requires only the computation of direct kinematics (1). Let $\hat{q}(t)$ $(n \times 1)$ be a solution to (1) relative to a given Cartesian trajectory $\hat{x}(t)$ $(m \times 1)$. The following error vector $e(t)$ can be defined between the desired trajectory $\hat{x}(t)$ and the actual trajectory $x(t)$ obtained from the algorithm state variables $q(t)$ as

$$e(t) = \hat{x}(t) - x(t). \tag{6}$$

In order to ensure the convergence of $q(t)$ to $\hat{q}(t)$, error dynamics is involved, i.e., via (3) (dropping the time dependence)

$$\dot{e} = \dot{\hat{x}} - J(q)\dot{q}. \tag{7}$$

With the choice

$$\dot{q} = \gamma J^T(q)e \quad \gamma = \alpha + (e^T\dot{\hat{x}})(e^T JJ^T e)^{-1}, \qquad \alpha > 0 \tag{8}$$

the closed-loop system of Fig. 1 ensures that $e \to 0$, and then $q \to \hat{q}$. This issue can be recognized by considering the error Lyapunov function $v = 0.5\, e^T e$ and verifying that its time derivative is negative-definite by virtue of (8) [11], [12], [28]. In the following it seems appropriate to give some remarks on the solution algorithm based on (8).

Remark 1: Let rank $(J) = m$. Suppose that $e(0) = 0$; (8) always guarantees a null tracking error, introducing, in the neighborhood of $e = 0$, an equivalent gain which tends to ∞. In reality, according to (8), $\dot{q}$ is formed by a first term which goes to zero as $e \to 0$, plus a second term which gives a finite contribution since it comes from the ratio of two quantities that, when $e \to 0$, go to zero with the same order two. Therefore, from a computational viewpoint, it seems more convenient to drop the second term in (8) and consider the purely proportional control law

$$\dot{q} = \alpha J^T e. \tag{9}$$

With this choice $\dot{v}$ becomes negative-definite only outside a region in the error space which contains $e = 0$, that is attractive for all trajectories. Obviously, the maximum tracking error will depend directly on $\|\dot{\hat{x}}\|$, and inversely on α; it must be underscored, however, that the steady-state ($\dot{\hat{x}} = 0$) error is identically zero [11], [12], [26], [28].

An alternative inverse kinematic solution algorithm can be conceived by considering

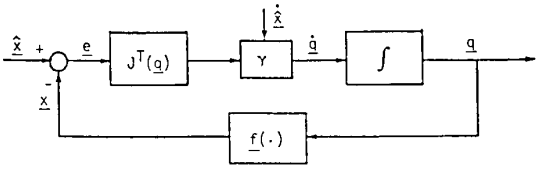


Fig. 1. The general closed-loop inverse kinematics scheme.

$$\dot{q} = J^{\dagger}(Me + \dot{\hat{x}}) \tag{10}$$

where M is a positive-definite matrix whose eigenvalues affect the position error convergence rate to zero [19]. The control law (10) recalls the pseudoinverse Jacobian control (4). If $\dot{x}$ in (4) is intended as a net velocity to bring the arm to a desired position which does involve measured error, (10) is similar to (4). On the other hand, in a trajectory tracking task, the solution (10) seems more robust than the pure pseudoinverse Jacobian control (4), since it is obtained via a computational scheme which is inherently closed-loop.

Among the three solutions derived above, (8)–(10), the proportional one (9) seems the most attractive from the computational point of view. Indeed, only direct kinematics (f, J) is to be computed, which drastically reduces the computational burden. In addition, the closed-loop system of Fig. 1, based on the solution (8) with $\gamma = \alpha$, will produce joint velocities $\dot{q}(t)$ at no additional cost, and with a slight modification it can also generate joint accelerations $\ddot{q}(t)$ [12]. This point is very advantageous for robot control in the joint space [16]–[18].

A more accurate analysis of the solution (9) reveals that the tracking error can be made arbitrarily small by choosing α large enough [13], [19]–[22], [28]. It must be noticed, however, that the implementation of the discrete time solution algorithm does not allow the choice of an arbitrarily large value for α. In order to establish an opportune value of the gain α that is related to the attractiveness region of the error state space, the algorithm should be formulated in the discrete time domain. Either the equivalent discrete Lyapunov method or the contraction mapping theorem could be used to investigate the attractiveness of the solution. A study towards this goal [15] has shown that, even though an estimate of the attractiveness region is not straightforward to derive, an adequate choice for the gain α is related to $1/T$, where T is the sampling period. In the same analysis [15] it has been shown that, in order to further decrease the tracking error, it is more appropriate to compute an error defined as $\epsilon_k = \hat{x}_{k+1} - x_k$. The implementation of the algorithm on a single dedicated microprocessor system with a floating-point unit has been realized in the laboratory and is described also in [15].

Furthermore, an appealing feature of the solution (9) lies in the possibility of achieving the following physical interpretation [12]. A fundamental relationship between the $(n \times 1)$ vector of joint torques τ and the Cartesian $(m \times 1)$ force vector f applied at the end effector is [14], [26]

$$\tau = J^T f. \tag{11}$$

It can be recognized that applying the above inverse kinematic

solution algorithm along a trajectory assigned at the end effector is equivalent to regard the vector $\alpha \boldsymbol{e}$ as the elastic force vector which has to be applied at the end effector of an ideal kinematic structure, with null mass and unit viscous damping coefficient, in order to track the desired trajectory.

Remark 2: Let now rank $(J) < m$. It could be argued that $J^T \boldsymbol{e} = \boldsymbol{0}$ when $\boldsymbol{e}$ belongs to the null space of J^T. In the light of the above equivalence, however, the null space of J^T is in those directions along which a Cartesian force applied at the end effector is completely neutralized by the mechanical constraints of the manipulator. From the implementation viewpoint, the occurrence of such situation can be detected by having $\|\dot{\boldsymbol{q}}\| = 0$ and $\|\boldsymbol{e}\| \neq 0$. Then, assuming that the target trajectory is planned consistently with the actual kinematic structure, it can be stated that in all cases, $J^T \boldsymbol{e} \neq \boldsymbol{0}$ when $\boldsymbol{e} \neq \boldsymbol{0}$.

To be more specific, consider the two DOF planar arm of Fig. 2 as a simple example. It is easy to recognize that a singularity occurs at $\theta_2 = 0$ and the null space of J^T is along the direction of the two aligned links. It can be found that if the desired velocity vector $\dot{\boldsymbol{x}}$ is orthogonal to the null space of J^T, as in Fig. 2(a), the endpoint can cross the singularity without any problem ($J^T \boldsymbol{e} \neq \boldsymbol{0}$). On the other hand, if $\dot{\boldsymbol{x}}$ does have a component along the null space of J^T, as in Fig. 2(b), high joint velocities are mechanically expected to track that trajectory. This issue is very important for correct trajectory planning, as the convergence of the algorithm is naturally degraded in case of high joint velocities. A more articulated example of a trajectory crossing a singularity can be found in [21].

Last but not least, it is to be mentioned that the solution algorithm derived here is completely general and manipulator-independent. An even shorter number of computations are required and better performance is achievable, however, if the algorithm is customized to each particular kinematic structure, as regards the last three axes of motion at the end effector: spherical wrist [13], [19], [20], [28], two-by-two intersecting axes [13], [20]–[22], [28], and nonconverging axes [13], [20], [28].

Inclusion of Constraints

As previously anticipated, redundancy can be conveniently exploited to solve the inverse kinematic problem with obstacle avoidance and/or limited joint range. The occurrence of either or both of the above situations sets some constraints which can be systematically incorporated in the solution algorithm outlined in the previous section, on the condition that the task space vector in (1) is properly enlarged.

A set of task space variables that describe the configuration of the manipulator with respect to the obstacle in the workspace and/or to the limits on the joint variables can be defined. The only requirement is to express those variables in terms of the joint variables so as to obtain the augmented direct kinematics [13], [28], [29]

$$y = f'(q) \tag{12}$$

where y is an $((m + v) \times 1)$ vector, with $0 \leq v \leq n - m$, which is completely specified in the task space. Then a solution to (12) can be formally obtained with a choice of the type (9) which proves very powerful, especially for on-line control purposes.

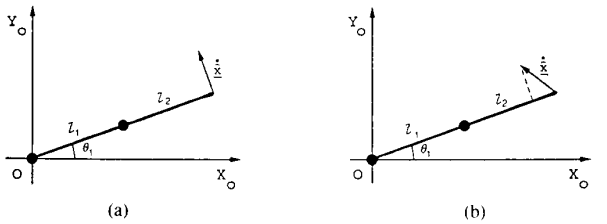


Fig. 2. A two DOF planar arm at a singularity. (a) The endpoint trajectory is executable. (b) The component of the endpoint trajectory on the links direction may cause high joint velocities.

For the sake of clarity, the two cases of obstacle avoidance and limited joint range are treated separately in the following. Nonetheless, they both lead to extend the task space vector as in (12), and then it will be possible to solve the inverse kinematic problem under both constraints, as will be demonstrated later.

Obstacle Avoidance

One of the potential advantages of a kinematically redundant manipulator is the use of the extra (redundant) DOF's to maneuver in a complex workspace and avoid contact with obstacles. The ability of the human arm to work in such environments provides a good model of this ability.

Assume that a manipulator is tracking a desired collision-free end effector trajectory in the task space. One or more links along its kinematical structure, however, may happen to be too close to an obstacle in the workspace, and a collision is expected. Since the inverse kinematic algorithm (9) provides joint configurations which are adjacent to each other as the manipulator proceeds along the trajectory, one or more constraints need to be introduced in order to avoid the collision with the obstacle. The idea that follows is similar in some regard to the approach taken in [8] and [24].

In the following it is supposed that obstacles are modeled as convex volumes in the three-dimensional (3D) space; this formalism seems effective because the actual obstructions can always be enveloped in a convex volume (the simplest is a sphere), while it is easy to compute the distance from such volumes to the links of the manipulator. For a planar mechanism one might simply adopt discs to model obstacles, as was done, for instance, in [25].

It can be assumed that a link has avoided a convex obstacle if its minimum distance form the obstacle is greater than a preplanned threshold distance. If all links satisfy this condition there is no reason for modifying the current solution along the trajectory, and the solution algorithm (9) will select one of the ∞^{n-m} possible configurations, depending on the initial joint configuration.

On the other hand, if the distance between one of the links and an obstacle becomes less than the threshold, the current solution is to be modified. It is understood that at most $n - m$ constraints of such type can be activated. One might also think of setting several values of thresholds in correspondence to each link of the manipulator. This can be done accounting for

the type of sensors used to detect the above distances, such as proximity sensors, video camera, etc.

In order to illustrate the technique proposed here, assume first that a single link is involved in a possible collision with a single obstacle. It is understood that such a pair varies as the manipulator's end effector tracks the desired trajectory of motion. Let then $\hat{d}_0$ be the threshold distance and $\boldsymbol{c}$ denote the position vector of the point of interest on the obstacle. A point at minimum distance from the link to the obstacle can be determined [24]; let $\boldsymbol{p}_0$ indicate the position vector of this so-called obstacle avoidance point. Both vectors $\boldsymbol{c}$ and $\boldsymbol{p}_0$ are defined with respect to the same base frame in which the direct kinematics of the manipulator is expressed; see Fig. 3 for a planar example. Another important remark is in order; the position of the minimum distance point moves, as the manipulator moves about the obstacle, i.e., $\boldsymbol{p}_0$ has to be dynamically recomputed along the trajectory. If the distance $\|\boldsymbol{d}_0\|$ between the two points, where $\boldsymbol{d}_0 = \boldsymbol{p}_0 - \boldsymbol{c}$, becomes less than the threshold distance $\hat{d}_0$, there is a danger of collision, and the joint velocities, which represent the control inputs to the system of Fig. 1, need to be modified accordingly to the new constraint activated. This can be accomplished as follows. In analogy to the error definition between reference and actual task variables (6), define the error

$$e_0 = 0.5(\hat{d}_0^2 - \boldsymbol{d}_0^T \boldsymbol{d}_0). \tag{13}$$

Differentiating (13) with respect to time gives

$$\dot{e}_0 = \hat{d}_0 \dot{\hat{d}}_0 - \boldsymbol{d}_0^T \dot{\boldsymbol{c}} - \boldsymbol{j}_{d0}^T \dot{\boldsymbol{q}} \tag{14}$$

with

$$\boldsymbol{j}_{d0}^T = \boldsymbol{d}_0^T J_{p0} \tag{15}$$

where J_{p0} is the Jacobian matrix $\partial \boldsymbol{p}_0 / \partial \boldsymbol{q}$ of the obstacle avoidance point [24]. If the error dynamics (7) is opportunely extended by (14), a control $\dot{\boldsymbol{q}}$ can be obtained in the same formal way as in (9). To this purpose, $\boldsymbol{e}$ will indicate the extended error vector ($(m + 1) \times 1$) in the Cartesian space, whose last component is e_0 defined in (13), and J will denote the extended Jacobian matrix ($(m + 1) \times n$), whose last row is $\boldsymbol{j}_{d0}^T$ defined in (15). In this way, the motion of those DOF's which determine the position of the obstacle point $\boldsymbol{p}_0$ is broken, preventing the link of interest from approaching the obstacle. As a matter of fact, a link which is a candidate for a collision is forced to move tangentially around the imaginary sphere with center at $\boldsymbol{c}$ and of radius $\hat{d}_0$. It should be noted that in (14) both the cases of moving obstacle ($\dot{\boldsymbol{c}} \neq \boldsymbol{0}$) and varying threshold distance ($\dot{\hat{d}}_0 \neq 0$) have been considered. In what follows it is assumed that $\dot{\boldsymbol{c}} = \boldsymbol{0}$ and $\dot{\hat{d}}_0 = 0$, without loss of generality.

At this extent, it seems quite straightforward to extend this technique to other pairs of points destined for a collision (an obstacle and an obstacle avoidance point along the structure of the manipulator, respectively) to eventually cover the $n - m$ redundant DOF's. Discussion of when an obstacle constraint can be released will be provided after the presentation of the overall solution.

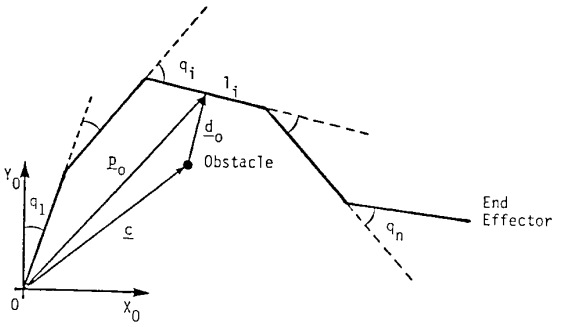


Fig. 3. Geometry of a planar manipulator showing the point nearest to the obstacle.

Limited Joint Range

Conceptually similar is the activation of a mechanical constraint on a joint variable for the inverse kinematic scheme of Fig. 1. The idea that follows was partially inspired by [7]. Assume that a joint variable q_i is kinematically constrained between two constant extremal values $q_{i\,\min}$ and $q_{i\,\max}$

$$q_{i\,\min} \leq q_i \leq q_{i\,\max}. \tag{16}$$

If the joint variable approaches either of the limits while the manipulator's end effector is tracking the preplanned trajectory, the solution given in (9) is to be modified. As done above for obstacle avoidance, a threshold distance $\hat{d}_q$ can be defined with the intent that if the distance of the current q_i from either of the two limits becomes less than $\hat{d}_q$, the control (9) needs to be modified. To this purpose define the error

$$e_q = \hat{d}_q - d_q \tag{17}$$

where either $d_q = q_i - q_{i\,\min}$ or $d_q = q_{i\,\max} - q_i$, depending on which limit is involved; note that d_q has to be dynamically recomputed along the trajectory. Differentiating (17) with respect to time gives

$$\dot{e}_q = \dot{\hat{d}}_q - \boldsymbol{u}_i^T \dot{\boldsymbol{q}} \tag{18}$$

where

$$\boldsymbol{u}_i^T = (0\ 0\ \cdots\ \underset{(i)}{\pm 1}\ \cdots\ 0). \tag{19}$$

The sign + applies to $q_{i\,\min}$ and the sign − applies to $q_{i\,\max}$. Therefore, it seems natural to extend the error dynamics (7) by (18), quite in the same way as it has been done for obstacle avoidance. The extended error vector will include an additive component due to e_q in (17), and, correspondingly, the extended Jacobian matrix will contain a row given by $\boldsymbol{u}_i^T$ in (19). In this way the motion of that DOF which was approaching the limit is opportunely broken and kept on the threshold. As above, if the threshold distance is chosen constant, $\dot{\hat{d}}_q = 0$.

Finally, as in the above case, the technique can be extended to account for other mechanical constraints of this type, to eventually cover the $n - m$ redundant DOF's. The possibility of releasing a joint limit constraint and the occurrence of both types of constraints will be discussed in the following subsection.

The Overall Solution

On the basis of the results of the two previous subsections, the overall solution algorithm to the inverse kinematic problem for constrained redundant manipulators is established in the following. To this purpose, the task space vector $\boldsymbol{x}$ is enlarged into the $((m + v) \times 1)$ vector $\boldsymbol{y}$, as anticipated in (12)

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{x}_0 \\ \boldsymbol{x}_q \end{bmatrix} \tag{20}$$

where $\boldsymbol{x}_0$ is the $(k \times 1)$ vector whose components are the quantities of the type $\boldsymbol{d}_0^T \boldsymbol{d}_0$ defined in (13) for each active obstacle constraint, and $\boldsymbol{x}_q$ is the $(r \times 1)$ vector whose components are the quantities of the type d_q defined in (17) for each active joint limit constraint. Correspondingly, the task space reference vector becomes

$$\hat{\boldsymbol{y}} = \begin{bmatrix} \hat{\boldsymbol{x}} \\ \hat{\boldsymbol{x}}_0 \\ \hat{\boldsymbol{x}}_q \end{bmatrix} \tag{21}$$

with obvious meaning of the vectors $\hat{\boldsymbol{x}}_0$ $(k \times 1)$ and $\hat{\boldsymbol{x}}_q$ $(r \times 1)$. It must be remarked that

$$v = k + r \leq n - m \tag{22}$$

so as to activate at most $n - m$ constraints, on obstacle avoidance (in number of k) [24] and joint range availability (in number of r) [4]. The control is then derived in the same formal way as in (9), i.e.,

$$\dot{\boldsymbol{q}} = \alpha J_e^T \boldsymbol{e}_y \tag{23}$$

where

$$J_e = \begin{bmatrix} J \\ J_{d0} \\ U \end{bmatrix} \tag{24}$$

is the extended Jacobian matrix which includes, besides the end-effector Jacobian matrix J, the Jacobian matrix J_{d0} $(k \times n)$ whose rows $\boldsymbol{j}_{d0}^T$ are defined in (15), and the matrix U $(r \times n)$ whose rows are defined in (19), and

$$\boldsymbol{e}_y = \begin{bmatrix} \boldsymbol{e} \\ \boldsymbol{e}_0 \\ \boldsymbol{e}_q \end{bmatrix} \tag{25}$$

is the extended error vector in the task space which includes, besides the end-effector error vector $\boldsymbol{e}$, the error vector $\boldsymbol{e}_0$ $(k \times 1)$ whose components are defined in (13), and the error vector $\boldsymbol{e}_q$ $(r \times 1)$ whose components are defined in (17). Finally, in (23) $\alpha > 0$ is a feedback gain which determines the convergence rate of $\boldsymbol{e}_y$.

Under the control (23), the end-effector trajectory $\hat{\boldsymbol{x}}(t)$ is still tracked as in (9), and the redundant DOF's are conveniently used to avoid collision with the obstacles present in the environment and/or to avoid the generation of joint trajectories which are not kinematically feasible.

Although (23) is the basis of the inverse kinematic solution proposed here, proper decision making by a higher control level is equally important, if not crucial, to the successful operation of the algorithm. This higher control level should be in charge of trajectory planning and activation + deactivation of contraints. To this purpose the following considerations seem appropriate.

The activation of a constraint is simply performed by detecting that one of the threshold distances (obstacle or joint limit) is violated. Once the threshold has been activated, the solution algorithm gives rise to a transient. Thus in the case the threshold is not wide enough, an obstacle could be hit. Since it can be recognized that the tracking error depends directly upon the velocity of the link destined for a possible collision with the obstacle, a practical criterion for setting the threshold width could be to make it directly proportional to the obstacle avoidance point velocity along the candidate link. In a similar manner a criterion could be derived for joint limit.

The introduction of constraints naturally sets restrictions on joint angles, which contribute to a reduction in the manipulator reachability workspace. A conflict with the desired trajectory $\hat{\boldsymbol{x}}(t)$, originally planned for the end effector, may then arise. In the limit, one may wind up in a "deadlocked" situation where further movement is not possible and the system is overconstrained, for instance, in a complex environment with multiple obstacles. The occurrence of such situation can be recognized, at the algorithmic level, by the fact that the extended error vector $\boldsymbol{e}_y$ in (25) enters the null space of the extended Jacobian J_e. This is equivalent to having $\|\dot{\boldsymbol{q}}\| = 0$ and $\|\boldsymbol{e}_y\| \neq 0$. In this case, the inverse kinematic algorithm must be aborted. This statement can be justified on the basis of the previous equivalence outlined for the unconstrained case.

It should be emphasized, however, that such situation may have not been caused by a complex environment, but rather determined by the fact that no constraint had been released at all. Consequently, the option of inactivating a constraint should be introduced, in the sense that a constraint could be released when the end-effector trajectory "naturally" drives the structure away from the constraint itself. This feature can be performed at algorithmic level as follows. Remember that the solution algorithm based on (23) guarantees only a bounded, but not null, tracking error. Therefore, the criterion for deactivating a constraint can be set up in the same way as it is done for activating a constraint, based on the evaluation of proper distance errors. Thus it is assumed that a constraint is deactivated if the corresponding distance error becomes negative.

It might be argued that the convergence of the solution algorithm as one or more constraints are either activated or deactivated does not strictly follow from the convergence properties of the configurations of the system, before and after either the enlargement or the reduction of the error space. This statement is theoretically correct, but from the engineering point of view, several simulation results have shown that the algorithm gives good performance for cases of practical interest, thus encouraging the adoption of the technique proposed here.

Finally, if the manipulator operates in a congested environment it may violently bounce among various thresholds activated in correspondence of several obstacles. In this case,

it is rather difficult to establish a general operating strategy; it would seem more appropriate to analyze the single case in order to evaluate the possibility of weighting the threshold widths according to the obstacles' location and the motion of the obstacle avoidance points along the candidate links for a collision.

A Case Study

In order to test the performance of the proposed inverse kinematic solution algorithm for constrained redundant manipulators, a simple case study is developed in the following. A planar four DOF manipulator of the type of Fig. 3 is considered; for readers' convenience the extended direct kinematic functions are reported in the Appendix. The link lengths have been chosen as $l_1 = l_2 = l_3 = l_4 = 0.3$ m. A straight-line reference trajectory is commanded from $\hat{\boldsymbol{p}}(0) = (0.5598 \ -0.15)^T$ [m] to $\hat{\boldsymbol{p}}(T) = (0.8 \ -0.2)^T$ [m] with $T = 3$ s, where $\hat{\boldsymbol{p}}$ denotes the desired endpoint position vector; the velocity profile is the typical trapezoidal one (acceleration + cruise + deceleration) with $\|\dot{\hat{\boldsymbol{p}}}\|_{\max} = 0.12$ m/s. The manipulator is assumed to start from a joint configuration which places the endpoint $\boldsymbol{p}(0)$ at $\hat{\boldsymbol{p}}(0)$. Notice that the resulting joint trajectories will depend upon the initial joint configuration, since the algorithm progresses with continuity along the given trajectory. The initial conditions on the joint variables are $\boldsymbol{q}(0) = (180 \ -30 \ -90 \ -30)^T$ [°].

Two constraints of the type previously described have been introduced. An obstacle has been located at $\boldsymbol{c} = (0.7 \ -0.28)^T$ [m], and a mechanical limit on the third joint is $q_3 \leq -80°$.

Four different sets of simulations have been carried out. Figs. 4–6 illustrate the results. A sampling time of 1 ms has been chosen; this is seen to be sufficient to evaluate the direct kinematic functions reported in the Appendix, using a single dedicated microprocessor system with a floating-point multiply unit [15]. The endpoint tracking error has been computed as the norm of the endpoint position error vector $\boldsymbol{e} = \hat{\boldsymbol{p}} - \boldsymbol{p}$.

In the first set of results the unconstrained control (9) has been applied with $\alpha = 1000$; to this end in [15] it is shown how discretizing the algorithm leads to the optimal choice of the feedback gain as the inverse of the sampling time. It is seen that the manipulator would violate both constraints while it tracks the endpoint reference trajectory. As a matter of fact, the endpoint tracking error (Fig. 4) is small, but the fourth link is involved in a collision with the obstacle (Fig. 5), and the third joint variable exceeds $-80°$ (Fig. 6).

In the second set of results, only the obstacle constraint has been activated as soon as $e_0 = \hat{d}_4^2 - \boldsymbol{d}_4^T\boldsymbol{d}_4$ becomes greater than zero; the threshold distance $\hat{d}_4$ has been set to 0.03 m. It is seen that the endpoint tracking performance remains satisfactory (Fig. 4) and the obstacle is avoided (Fig. 5) in the sense that $\|\boldsymbol{d}_4\|$, after going below $\hat{d}_4$, tends asymptotically to this value, as proven in theory. The constraint on the third joint variable, however, is still violated (Fig. 6).

In the third set of results only the joint limit constraint has been activated as soon as $e_q = \hat{d}_{q3} - d_{q3}$ becomes greater than zero; the threshold distance $\hat{d}_{q3}$ has been chosen as 2°. Similarly to the previous case, the endpoint tracking is maintained (Fig. 4) and the third joint variable is braked against passing the limit (Fig. 6) in the sense that q_3, after going above $-82°$, tends asymptotically to this value, as anticipated in theory.

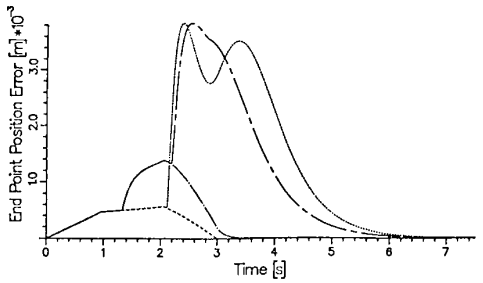


Fig. 4. Endpoint position tracking errors. Dashed line—no constraints; dotted line—obstacle contraint; chain-dotted line—joint constraint; chain-dashed line—both constraints.

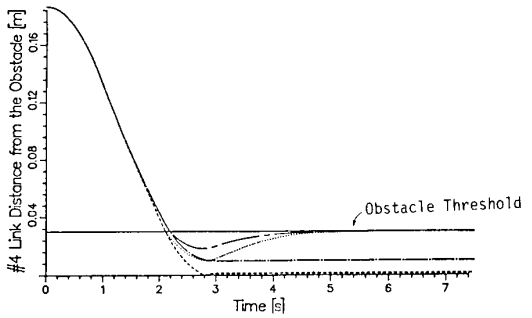


Fig. 5. #4 link distances from the obstacle. Dashed line—no constraints; dotted line—obstacle constraint; chain-dotted line—joint constraint; chain-dashed line—both constraints.

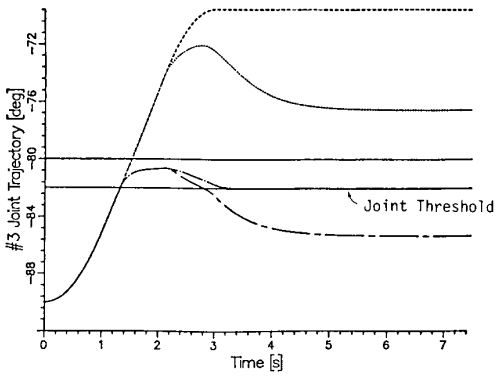


Fig. 6. #3 joint trajectories. Dashed line—no constraints; dotted line—obstacle constraint; chain-dotted line—joint constraints; chain-dashed line—both constraints.

Finally, both constraints have been activated at proper instants of time. In this particular case it can be recognized (Fig. 6) that the joint limit constraint has been released, as the endpoint trajectory naturally drives q_3 away from the limit, due to the earlier activation of the obstacle constraint. It is seen that the endpoint tracking is maintained (Fig. 4) and the obstacle threshold is asymptotically met (Fig. 5).

From an accurate analysis of Fig. 4, it migth be argued that the endpoint tracking error becomes worse when one or two constraints are activated. This is a direct consequence of the fact that the dimension of the error space is augmented and the

convergence characteristics are modified, as was discussed in the preceding section. On the other hand, it has to be pointed out that, via the application of (5) though more computational demanding, the arm could simultaneously move away from the obstacle and meet the endpoint trajectory command.

Conclusions

This paper has presented a solution to the inverse kinematic problem for constrained redundant manipulators. The resulting algorithm is based on a dynamic reformulation of the problem leading to a closed-loop scheme whose stability is assured by choosing a control which only involves the computation of the direct kinematics of the manipulator. The only requirement is to systematically enlarge the direct kinematics in order to include the constraints, such as obstacle avoidance and limited joint range which can be potentially met by a kinematically redundant manipulator. It has been proven, and shown by an example, that the constraints, once active, can be successfully met. The endpoint of the manipulator keeps tracking the desired trajectory and the resulting joint trajectories avoid the collision with an obstacle and/or avoid violation of a mechanical limit on a joint variable. Some issues regarding the activation and the deactivation of constraints have been finally addressed. The possibility of including constraints on the manipulator's dexterity is currently being investigated [27]. Future research efforts will be most likely dedicated to investigation of the application of the technique presented here to particular redundant geometries, such as maintainance robots operating in plasma vessels.

Appendix

The direct kinematics of a planar four DOF manipulator is reported in the following. Given the four joint angles θ_i, $i = 1$–4 and the lengths of the four links l_i, $i = 1$–4, the coordinates of the endpoint vector $\boldsymbol{p}$ result

$$\begin{aligned} p_x &= l_1 s_1 + l_2 s_{12} + l_3 s_{123} + l_4 s_{1234} \\ p_y &= l_1 c_1 + l_2 c_{12} + l_3 c_{123} + l_4 c_{1234} \end{aligned} \tag{A1}$$

where $s_i = \sin\theta_i$ and $c_i = \cos\theta_i$ and similarly for $\theta_1 + \theta_2 + \cdots$. The Jacobian matrix J_p is easily found as

$$J_p = \begin{bmatrix} j_{p11} & j_{p12} & j_{p13} & j_{p14} \\ j_{p21} & j_{p22} & j_{p23} & j_{p24} \end{bmatrix} \tag{A2a}$$

$$\begin{aligned} j_{p14} &= l_4 c_{1234} & j_{p24} &= -l_4 s_{1234} \\ j_{p13} &= j_{p14} + l_3 c_{123} & j_{p23} &= j_{p24} - l_3 s_{123} \\ j_{p12} &= j_{p13} + l_2 c_{12} & j_{p22} &= j_{p23} - l_2 s_{12} \\ j_{p11} &= p_y & j_{p21} &= -p_x. \end{aligned} \tag{A2b}$$

The minimum distance vector from the link i to an obstacle located at $\boldsymbol{c} = (c_x\ c_y)^T$ is simply given by

$$\boldsymbol{d}_i = \min\ [(\boldsymbol{p}_i - \boldsymbol{c}),\ (\boldsymbol{p}_{i+1} - \boldsymbol{c})] \tag{A3}$$

if the perpendicular line from the obstacle location to the line which contains the link does not intersect with the link itself. The vectors $\boldsymbol{p}_i$ in (A3) in this case coincide with the position vectors to the joint locations, $i = 1$–4; they are intrinsically computed in (A1) with $\boldsymbol{p}_4 = \boldsymbol{p}$.

On the other hand, if the link i intersects with the perpendicular line from the obstacle, the minimum distance vectors result

$$\boldsymbol{d}_1 = (d_{1x}\ d_{1y})^T \tag{A4a}$$

$$\begin{aligned} d_{1x} &= c_1(s_1 c_y - c_1 c_x) - c_x \\ d_{1y} &= -s_1(s_1 c_y - c_1 c_x) - c_y \end{aligned} \tag{A4b}$$

$$\boldsymbol{d}_2 = (d_{2x}\ d_{2y})^T \tag{A5a}$$

$$\begin{aligned} d_{2x} &= c_{12}(-l_1 s_2 + s_{12} c_y - c_{12} c_x) - c_x \\ d_{2y} &= -s_{12}(-l_1 s_2 + s_{12} c_y - c_{12} c_x) - c_y \end{aligned} \tag{A5b}$$

$$\boldsymbol{d}_3 = (d_{3x}\ d_{3y})^T \tag{A6a}$$

$$\begin{aligned} d_{3x} &= c_{123}(-l_1 s_{23} - l_2 s_3 + s_{123} c_y - c_{123} c_x) - c_x \\ d_{3y} &= -s_{123}(-l_1 s_{23} - l_2 s_3 + s_{123} c_y - c_{123} c_x) - c_y \end{aligned} \tag{A6b}$$

$$\boldsymbol{d}_4 = (d_{4x}\ d_{4y})^T \tag{A7a}$$

$$\begin{aligned} d_{4x} &= c_{1234}(-l_1 s_{1234} - l_2 s_{34} - l_3 s_4 \\ &\quad + s_{1234} c_y - c_{1234} c_x) - c_x \\ d_{4y} &= -s_{1234}(-l_1 s_{1234} - l_2 s_{34} - l_3 s_4 \\ &\quad + s_{1234} c_y - c_{1234} x_x) - c_y \end{aligned} \tag{A7b}$$

respectively, for the four links.

The Jacobian matrices $\boldsymbol{j}_{di}^T$ defined in (15) are directly computable if the distance vectors $\boldsymbol{d}_i$ are given by (A3), since their elements are already derived in J_p. In the case of $\boldsymbol{d}_i$ given by (A4)–(A7), instead, it results in

$$\boldsymbol{j}_{d1}^T = (j_{d11}\ 0\ 0\ 0) \tag{A8a}$$

$$j_{d11} = s_1 c_1(c_y^2 - c_x^2) + c_x c_y(s_1^2 - c_1^2) \tag{A8b}$$

$$\boldsymbol{j}_{d2}^T = (j_{d21}\ j_{d22}\ 0\ 0) \tag{A9a}$$

$$\begin{aligned} j_{d21} &= s_{12} c_{12}(c_y^2 - c_x^2) - l_1 s_2(c_{12} c_y + s_{12} c_x) \\ &\quad + c_x c_y(s_{12}^2 - c_{12}^2) \\ j_{d22} &= j_{d21} + l_1 c_2(l_1 s_2 + c_{12} c_x - s_{12} c_y) \end{aligned} \tag{A9b}$$

$$\boldsymbol{j}_{d3}^T = (j_{d31}\ j_{d32}\ j_{d33}\ 0) \tag{A10a}$$

$$\begin{aligned} j_{d31} &= s_{123} c_{123}(c_y^2 - c_x^2) + c_x c_y(s_{123}^2 - c_{123}^2) \\ &\quad - (c_{123} c_y + s_{123} c_x)(l_1 s_{23} + l_2 s_3) \\ j_{d32} &= j_{d31} + l_1 c_{23}(s_{23} + l_2 s_3 + c_{123} c_x - s_{123} c_y) \\ j_{d33} &= j_{d32} + l_2 c_3(s_3 + l_1 s_{23} + c_{123} c_x - s_{123} c_y) \end{aligned} \tag{A10b}$$

$$\boldsymbol{j}_{d4}^T = (j_{d41}\ j_{d42}\ j_{d43}\ j_{d44}) \tag{A11a}$$

$$\begin{aligned} j_{d41} &= s_{1234} c_{1234}(c_y^2 - c_x^2) + c_x c_y(s_{1234}^2 - c_{1234}^2) \\ &\quad - (c_{1234} c_y + s_{1234} c_x)(l_1 s_{234} + l_2 s_{34} + l_3 s_4) \\ j_{d42} &= j_{d41} + l_1 c_{234}(s_{234} + l_2 s_{34} + l_3 s_4 + c_{1234} c_x - s_{1234} c_y) \\ j_{d43} &= j_{d42} + l_2 c_{34}(s_{34} + l_1 s_{234} + l_3 s_4 + c_{1234} c_x - s_{1234} c_y) \end{aligned}$$

$$j_{d44} = j_{d43} + l_3 c_4 (s_4 + l_1 s_{234} + l_2 s_{34} + c_{1234} c_x - s_{1234} c_y) \tag{A11b}$$

respectively, for the four links.

Acknowledgment

The authors would like to thank the anonymous reviewers for helpful comments on an earlier version of this paper.

References

[1] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *ASME J. Appl. Mech.*, vol. 22, pp. 215-221, June 1955.

[2] R. P. Paul, B. Shimano, and G. E. Mayer, "Kinematic control equations for simple manipulators," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-11, no. 6, pp. 449-455, June 1981.

[3] D. E. Whitney, "The mathematics of coordinated control of prosthetic arms and manipulators," *Trans. ASME J. Dyn. Syst., Meas., Contr.*, vol. 94, pp. 303-309, Dec. 1972.

[4] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-7, no. 12, pp. 868-871, Dec. 1977.

[5] 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, no. 3, pp. 245-250, Mar./Apr. 1983.

[6] C. A. Klein, "Use of redundancy in the design of robotic systems," in *Proc. 2nd Int. Symp. on Robotics Research* (Kyoto, Japan, Aug. 1984).

[7] M. Vukobratovic and M. Kircanski, "A dynamic approach to nominal trajectory synthesis for redundant manipulators," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-14, no. 4, pp. 580-586, July/Aug. 1984.

[8] M. Kircanski and M. Vukobratovic, "Trajectory planning for redundant manipulators in the presence of obstacles," in *Proc. 5th CISM-IFToMM Ro.Man.Syst.* (Udine, Italy, June 1984).

[9] B. Benhabib, A. A. Goldenberg, and R. G. Fenton, "A solution to the inverse kinematics of redundant manipulators," in *Proc. 1985 American Control Conf.* (Boston, MA, June 1985).

[10] A. A. Goldenberg, B. Benhabib, and R. G. Fenton, "A complete generalized solution to the inverse kinematics of robots," *IEEE J. Robotics Automat.*, vol. RA-1, no. 1, pp. 14-20, Mar. 1985.

[11] A. Balestrino, G. De Maria, and L. Sciavicco, "Robust control of robotic manipulators," in *Proc. 9th IFAC World Congress* (Budapest, Hungary, July 1984).

[12] W. A. Wolovich and H. Elliott, "A computational technique for inverse kinematics," in *Proc. 23rd IEEE Conf. on Decision and Control* (Las Vegas, NV, Dec. 1984).

[13] L. Sciavicco and B. Siciliano, "Solving the inverse kinematic problem for robotic manipulators," in *Proc. 6th CISM-IFToMM Ro. Man. Syst.* (Cracow, Poland, Sept. 1986).

[14] R. P. Paul, *Robot Manipulators: Mathematics, Programming, and Control*. Cambridge, MA: MIT Press, 1981.

[15] G. De Maria and R. Marino, "A discrete algorithm for solving the inverse kinematic problem for robotic manipulators," in *Proc. 2nd Int. Conf. on Advanced Robotics* (Tokyo, Japan, Sept. 1985).

[16] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, "Resolved acceleration control of mechanical manipulators," *IEEE Trans. Automat. Contr.*, vol. AC-25, no. 3, pp. 468-474, June 1980.

[17] A. Balestrino, G. De Maria, and L. Sciavicco, "An adaptive model following control for robotic manipulators," *Trans. ASME J. Dyn. Syst., Meas., Contr.*, vol. 105, no. 3, pp. 143-151, Sept. 1983.

[18] J. J. E. Slotine, "The robust control of robot manipulators," *Int. J. Robotics Res.*, vol. 4, no. 2, pp. 49-64, Summer 1985.

[19] A. Balestrino, G. De Maria, L. Sciavicco, and B. Siciliano, "An algorithmic approach to coordinate transformation for robotic manipulators," *Adv. Robotics*, vol. 2, 1988.

[20] G. De Maria, L. Sciavicco, and B. Siciliano, "A general solution algorithm to coordinate transformation for robotic manipulators," in *Proc. 2nd Int. Conf. on Advanced Robotics* (Tokyo, Japan, Sept. 1985).

[21] L. Sciavicco and B. Siciliano, "Coordinate transformation: A solution algorithm for one class of robots," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-16, no. 4, pp. 550-559, July/Aug. 1986.

[22] L. Sciavicco and B. Siciliano, "An inverse kinematic solution algorithm for robots with two-by-two intersecting axes at the end effector," in *Proc. 1986 IEEE Int. Conf. on Robotics and Automation* (San Francisco, CA, Apr. 1986).

[23] D. Pieper, "The kinematics of manipulators under computer control," Ph.D. dissertation, Stanford University, Stanford, CA, 1968.

[24] A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *Int. J. Robotics Res.*, vol. 4, no. 3, pp. 109-117, Fall 1985.

[25] J. Baillieul, "Avoiding obstacles and resolving kinematic redundancy," in *Proc. 1986 Int. Conf. on Robotics and Automation* (San Francisco, CA, Apr. 1986).

[26] H. Asada and J. J. E. Slotine, *Robot Analysis and Control*. New York, NY: Wiley, 1986.

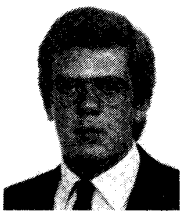
[27] L. Sciavicco and B. Siciliano, "An inverse kinematic solution algorithm for dexterous redundant manipulators," in *Proc. 3rd Int. Conf. on Advanced Robotics* (Versailles, France, Oct. 1987).

[28] B. Siciliano, "Solution algorithms to the inverse kinematic problem for manipulation robots" (in Italian), Ph.D. dissertation, University of Naples, Naples, Italy, Dec. 1986.

[29] L. Sciavicco and B. Siciliano, "A dynamic solution to the inverse kinematic problem for redundant manipulators," in *Proc. 1987 IEEE Int. Conf. on Robotics and Automation* (Raleigh, NC, Mar./Apr. 1987).

Lorenzo Sciavicco was born in Rome, Italy, on December 8, 1938. He received the Doctoral degree in electronic engineering from the Università di Roma in 1963.

Since 1970 he has been with the Università di Napoli, Naples, Italy where he is presently Professor in the area of Automatic Control at the Dipartimento di Informatica e Sistemistica. His fields of interest are in automatic control theory and applications; in particular, stability of nonlinear control systems and robotics.



Bruno Siciliano was born in Naples, Italy, on October 27, 1959. He received the Laurea and the Ph.D. degrees in electronic engineering from the Università di Napoli, Naples, in 1982 and 1987, respectively.

Since 1982 he has been a Research Assistant at the Dipartimento di Informatica e Sistemistica, Università di Napoli. His research interests include robust control of nonlinear systems, kinematics dynamics and control of robotic manipulators, and control of lightweight flexible arms. From Fall 1985 through Spring 1986 he was a Visiting Scholar at the G. W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, under a NATO Research Fellowship. In December 1986, he received the Jean Vertut National Award for the most original paper published in robotics.