Automatica, Vol. 27, No. 3, pp. 535–539, 1991
Printed in Great Britain.

0005-1098/91 $3.00 + 0.00
Pergamon Press plc

Brief Paper

A Sensitivity Approach to Optimal Spline Robot Trajectories*

A. DE LUCA,†‡ L. LANARI† and G. ORIOLO†

Key Words—Robots; splines; optimization; nonlinear programming; sensitivity analysis; trajectory planning.

Abstract—A robot trajectory planning problem is considered. Using smooth interpolating cubic splines as joint space trajectories, the path is parameterized in terms of time intervals between knots. A minimum time optimization problem is formulated under maximum torque and velocity constraints, and is solved by means of a first order derivative-type algorithm for semi-infinite nonlinear programming. Feasible directions in the parameter space are generated using sensitivity coefficients of the active constraints. Numerical simulations are reported for a two-link Scara robot. The proposed approach can be used for optimizing more general objective functions under different types of constraints.

Introduction

OPTIMAL TRAJECTORY planning may considerably improve robot performance in industrial applications, particularly when productivity rate or energy consumption are of primary concern. In order to provide true optimal motion under actuator limitations, the full nonlinear manipulator dynamics has to be explicitly considered in the trajectory planning phase. The interactions between geometric, kinematic and dynamic issues substantially increase problem complexity with respect to purely kinematic approaches.

Specific classes of optimal robot motion planning problems have been recently solved, with minimum time as objective and torque limits as constraints. When the task is a point-to-point motion, a number of numerical approaches are available to minimize the traveling time, e.g. a modified gradient-type algorithm (Weinreb and Bryson, 1985), the multiple-shooting technique for nonlinear TPBVP (Geering *et al.*, 1986), and a dynamic programming scheme (Sahar and Hollerbach, 1986). All the above solution methods are computationally intensive. Moreover, the introduction of state constraints—like joint limits or maximum velocity bounds—brings in additional complexity. In any case, the main limitation of point-to-point motion planning is the unpredictability of the obtained path, which can be dangerous in presence of obstacles.

Alternatively, the robot may be required to follow a safe prespecified geometric path joining the initial to the final point, either in joint or in cartesian space. Assuming that a continuous parameterization can be given for the whole path, Bobrow *et al.* (1985) and Shin and McKay (1985) have derived an efficient solution algorithm for the minimum time problem, directly working in the parameter phase-plane. It should be emphasized that the efficiency of their algorithm strongly relies on the particular form of the cost criterion. Also, the *a priori* specification of an overall geometric path and its continuous parameterization are requirements which may be too restrictive or cumbersome for real applications.

Most commonly, the *task* planner provides the *trajectory* planner with a robotic task description which is intermediate between the above two. Especially in complex environments, the typical output of the task planner is a sequence of cartesian poses (i.e. positions and orientations) for the end-effector, which have to be interpolated. In principle, intermediate poses are not restricted, but a safe overall path can be guaranteed by increasing the number of specified poses in proximity of obstacles. The problem faced here is how the trajectory planner can perform this interpolation in an optimal way. The class of interpolating functions should be chosen so to give nice smoothness properties, thus avoiding excitation of the mechanical structure, together with a low curvature profile.

The most appealing class of functions for generating robotic paths that satisfy the above specifications are spline functions, which are piecewise cubic polynomials smoothly interpolating a sequence of knots (de Boor, 1978). Splines with continuous second derivative (C^2-splines) have been widely used in robotic applications, e.g. to obtain minimum time trajectories under purely kinematic constraints (Lin *et al.*, 1983). The minimization algorithm was the Nelder–Mead flexible polyhedron search; being based only on function evaluations, it has slow convergence and may stop in false constrained minima. Bobrow (1988) used C^1-splines for approximating point-to-point minimum time paths in presence of obstacles.

In this paper, a new method is presented for planning smooth optimal robot trajectories interpolating a given sequence of points. The trajectory is a C^2-spline passing through n knots, with boundary conditions on initial and final velocity. The $n - 1$ time intervals between the knots uniquely parameterize the path. A minimum time problem will be considered here, with both maximum torque and velocity limits. It turns out to be an optimization problem with infinite-dimensional constraints, which is solved via an efficient algorithm proposed by Gonzaga *et al.* (1980). This requires computation of the gradients of the active constraints, namely the sensitivity functions of the constraints with respect to variations of the design parameters. As an intermediate step, it is necessary to compute the sensitivity of the spline functions, which has its own interest and may be relevant also for other applications.

It must be stressed that the approach proposed here is conceptually different from the most common one, that would require *first* to build a spline interpolating the knots, and *then* find the optimal time history on this assigned path, using the algorithm of Bobrow *et al.* (1985). In fact, the

* Received 24 February 1989; revised 10 July 1990; received in final form 25 August 1990. The original version of this paper was presented at the IFAC Symposium on Robot Control (SYROCO '88) which was held in Karlsruhe, Germany during October 1988. The published proceedings of this IFAC Meeting may be ordered from: Pergamon Press plc, Headington Hill Hall, Oxford OX3 0BW, U.K. This paper was recommended for publication in revised form by Associate Editor R. V. Patel under the direction of Editor H. Kwakernaak.

† Dipartimento di Informatica e Sistemistica, Università degli Studi di Roma "La Sapienza", Via Eudossiana 18, 00184 Rome, Italy.

‡ Author to whom all correspondence should be addressed.

following aspects characterize the present method:

- The discrete parameterization allows one to handle any cost function of interest (e.g. energy consumption), while well-established *exact* solutions exist only in the minimum time case;
- Since the interpolating spline is a function of time intervals between knots, the resulting cartesian path changes throughout the optimization process, possibly yielding a shorter final traveling time than the one achievable on the "first-guess" path;
- Velocity bounds are explicitly considered; moreover, smoothness up to the second derivative is guaranteed, avoiding torque discontinuities.

After restating the essential steps for the generation of an interpolating spline trajectory, the minimum time problem is formulated as a semi-infinite nonlinear programming problem and solved by the algorithm of Gonzaga *et al.* (1980). To this goal, the sensitivity analysis of spline and torque functions with respect to time intervals will be derived. Numerical results obtained for two planar robot arms are presented. In the conclusions, some possible extensions of the proposed approach are outlined.

Spline trajectory generation

For a robot with N joints, let the task be assigned by a sequence of n cartesian poses $P_1, P_2, \ldots, P_n$ to be assumed by the end-effector at unspecified time instants $t_1, t_2, \ldots, t_n$. Initial and final cartesian velocities are given. Using inverse kinematics, these data are transformed into joint configurations $q_{j1}, q_{j2}, \ldots, q_{jn}$, and into initial and final joint velocities v_{j1} and v_{jn}, with $j = 1, \ldots, N$. Let $h_i = t_{i+1} - t_i$, $i = 1, \ldots, n-1$, be the time intervals between knots, and $\mathbf{h} = [h_1, \ldots, h_{n-1}]^T$. For a generic joint j, a trajectory is obtained by interpolation using a C^2-spline $Q_j(t)$—a piecewise cubic polynomial. Denote by ω_{ji} the value of the jth joint acceleration at the ith knot. Each of the $n-1$ cubics $Q_{ji}(t)$ constituting the spline can be written in terms of the ω_{ji} (Lin *et al.*, 1983) as

$$Q_{ji}(t) = \frac{(t_{i+1}-t)^3}{6h_i}\omega_{ji} + \frac{(t-t_i)^3}{6h_i}\omega_{j,i+1} + \left[\frac{q_{j,i+1}}{h_i} - \frac{h_i\omega_{j,i+1}}{6}\right](t-t_i) + \left[\frac{q_{ji}}{h_i} - \frac{h_i\omega_{ji}}{6}\right](t_{i+1}-t) \quad (1)$$

where $t \in [t_i, t_{i+1}]$. Spline velocity $\dot{Q}_j(t)$ and acceleration $\ddot{Q}_j(t)$ are piecewise quadratic and linear functions, respectively. The continuity requirement for the acceleration at the internal knots is automatically satisfied, since $\ddot{Q}_{j,i-1}(t_i) = \ddot{Q}_{j,i}(t_i) = \omega_{ji}$. Given a timing for the trajectory (i.e. a vector $\mathbf{h}$), the interpolating spline is completely specified by (1) once knots accelerations ω_{ji} are computed. Denoting by Ω_j the vector of knots accelerations for the jth joint, and imposing the continuity for velocities at the internal knots as well as boundary conditions, for each joint a tridiagonal linear system is derived in the form:

$$\mathbf{A}\Omega_j = \mathbf{b}_j, \quad j = 1, \ldots, N, \quad (2)$$

where

$$\mathbf{A} = \begin{bmatrix} 2h_1 & h_1 & & & \\ h_1 & 2(h_1+h_2) & & & \\ & & \ddots & & \\ & & & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ & & & h_{n-1} & 2h_{n-1} \end{bmatrix},$$

$$\mathbf{b}_j = \begin{bmatrix} 6\left(\frac{q_{j2}-q_{j1}}{h_1} - v_{j1}\right) \\ 6\left(\frac{q_{j3}-q_{j2}}{h_2} - \frac{q_{j2}-q_{j1}}{h_1}\right) \\ \vdots \\ 6\left(\frac{q_{jn}-q_{j,n-1}}{h_{n-1}} - \frac{q_{j,n-1}-q_{j,n-2}}{h_{n-2}}\right) \\ 6\left(v_{jn} - \frac{q_{jn}-q_{j,n-1}}{h_{n-1}}\right) \end{bmatrix}.$$

For $\mathbf{h} > \mathbf{0}$, matrix $\mathbf{A}$ is diagonally dominant and the solution of (2) is unique and continuous in the elements of $\mathbf{h}$. Note that $\mathbf{A}$ is the same for all joints. Solution Ω_j of (2) is efficiently obtained without the need of pivoting by the following Gauss-type algorithm (Stoer and Bulirsch, 1980):

$$\omega_{jn} = \bar{\omega}_{jn}, \quad \omega_{ji} = \bar{\omega}_{ji} - K_i\omega_{j,i+1}, \quad i = n-1, \ldots, 1, \quad (3)$$

with

$$\begin{aligned} &\bar{\omega}_{j1} = \frac{b_{j1}}{a_{11}}, \quad \bar{\omega}_{ji} = \frac{b_{ji} - \bar{\omega}_{j,i-1}a_{i,i-1}}{a_{ii} - K_{i-1}a_{i,i-1}}, \quad i = 2, \ldots, n, \\ &K_1 = \frac{a_{12}}{a_{11}}, \quad K_i = \frac{a_{i,i+1}}{a_{ii} - K_{i-1}a_{i,i-1}}, \quad i = 2, \ldots, n-1. \end{aligned} \quad (4)$$

where a_{hk} are the elements of $\mathbf{A}$, and b_{jk} of $\mathbf{b}_j$.

For any choice of parameter vector $\mathbf{h}$, a unique spline $Q_j(t, \mathbf{h})$ is built for each joint, forming a vector function $\mathbf{Q}(t, \mathbf{h})$. It should be noted that changing one or more of the parameters h_i will modify the whole interpolating spline. This is true unless a uniform scaling is performed on *all* the components of $\mathbf{h}$ (Hollerbach, 1984).

The optimization problem

Each spline trajectory $Q_j(t, \mathbf{h})$ is parameterized in terms of vector $\mathbf{h}$. The optimal value of this vector can be determined according to a specified criterion and subject to proper constraints. If a minimum time problem is considered, typical constraints are symmetric velocity and torque limits for each joint, V_j and U_j respectively. Some peculiar features of the problem are exploited to obtain a compact formulation.

Velocity constraints. Since spline velocity is a piecewise quadratic function, its maximum value in any subinterval is attained either at the knot instants t_i, t_{i+1}, or at one intermediate instant $t_{ji}^* \in [t_i, t_{i+1}]$ where $\ddot{Q}_j(t_{ji}^*, \mathbf{h}) = 0$. This zero-acceleration instant exists iff $\omega_{ji}\omega_{j,i+1} < 0$ holds, in which case $t_{ji}^* = t_i + h_i\omega_{ji}/(\omega_{ji} - \omega_{j,i+1})$ and

$$\dot{Q}_{ji}(t_{ji}^*) = -\frac{h_i}{2(\omega_{j,i+1} - \omega_{ji})}\omega_{ji}\omega_{j,i+1} + \frac{q_{j,i+1} - q_{ji}}{h_i} - \frac{h_i}{6}(\omega_{j,i+1} - \omega_{ji}). \quad (5)$$

Torque constraints. The maximum torque value can be attained anywhere along the spline trajectory; this constraint should be checked in all path points using robot dynamics

$$\mathbf{u}(t) = \mathbf{M}(\mathbf{q}(t))\ddot{\mathbf{q}}(t) + \mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) + \mathbf{e}(\mathbf{q}(t)), \quad (6)$$

where $\mathbf{q} \in \mathbb{R}^N$ are the joint coordinates, $\mathbf{u}$ are the actuator torques, $\mathbf{M}$ is the inertia matrix, $\mathbf{c}$ are the Coriolis and centrifugal forces, and $\mathbf{e}$ is the gravity term. The torque $u_j(t, \mathbf{h})$ arising at joint j during motion along a spline trajectory is obtained by setting in (6) $\mathbf{q}(t) = \mathbf{Q}(t, \mathbf{h})$, together with its time derivatives.

Time interval constraints. Since parameterized splines are defined only for $\mathbf{h} > \mathbf{0}$, this constraint should be included in the optimization problem. The strict positivity requirement can be replaced by a lower bound derived from velocity limits. In fact, since $|q_{j,i+1} - q_{ji}|/h_i \le V_j$ must hold on any subinterval, $\mathbf{h}$ should satisfy the constraint

$$h_i \ge w_i > 0 \quad \text{where} \quad w_i = \max_{j=1,\ldots,N}\left\{\frac{|q_{j,i+1} - q_{ji}|}{V_j}\right\}, \quad i = 1, \ldots, n-1.$$

As a result, the following optimization problem is formulated:

$$\begin{aligned} &\min \sum_{i=1}^{n-1} h_i \\ &\text{s.t. } |\dot{Q}_j(t_i, \mathbf{h})| \le V_j \quad (i = 1, \ldots, n), \\ &\quad |\dot{Q}_j(t_{ji}^*, \mathbf{h})| \le V_j \quad (i = 1, \ldots, n-1), \\ &\quad \max_{t \in [t_1, t_n]} \{|u_j(t, \mathbf{h})|\} \le U_j, \quad j = 1, \ldots, N, \\ &\quad \mathbf{w} - \mathbf{h} \le \mathbf{0}, \end{aligned}$$

This problem is an instance of the semi-infinite nonlinear programming class

$$\min f(\mathbf{h})$$
$$\text{s.t. } g^l(\mathbf{h}) \le 0, \qquad l = 1, \ldots, p,$$
$$\max_{\tau \in T} \phi^j(\tau, \mathbf{h}) \le 0, \quad j = 1, \ldots, r,$$

where $\mathbf{h} \in \mathbb{R}^{n-1}$ is the vector of design parameters, g^l are conventional constraints (velocity constraints plus lower bounds on $\mathbf{h}$), while functional (infinite-dimensional) constraints are represented through the ϕ^j's (torque constraints). The domain T over which the functional constraints have to be satisfied is $[t_1, t_n]$. In the minimum time problem, $f(\mathbf{h})$ and $g^l(\mathbf{h})$ are continuously differentiable, $\phi^j(\tau, \mathbf{h})$ are continuous in both arguments, while gradients $\nabla_{\mathbf{h}}\phi^j(\tau, \mathbf{h})$ are continuous w.r.t. $\mathbf{h}$.

For this class of problems, Gonzaga *et al.* (1980) have developed an efficient solution algorithm which is a combined phase I–phase II method of feasible directions. The method does not require an admissible starting point, and recovers feasibility in a finite number of iterations, already considering the objective function at this stage. At each iteration, a low-dimensional quadratic programming (QP) subproblem is solved to generate a search direction $\mathbf{d}$ in the space of parameters $\mathbf{h}$. Directional derivatives of the objective function and of the ε-active (i.e. active or almost active) constraints are needed in this QP. The algorithm uses a proper discretization of the functional constraints, replacing the continuous domain T with a set T_s of *mesh instants*. This discretization must be tailored to the problem at hand. Since T is itself a function of the current $\mathbf{h}$, an adaptive strategy is devised to discretize T into T_s. At iteration q, the following set of mesh points is used:

$$T_s^{[q]} = \{t_{im}^{[q]} \mid t_{im}^{[q]} = t_i^{[q]} + \alpha_m h_i; \quad m = 0, 1, \ldots, s; i = 1, \ldots, n-1\} \tag{7}$$

where $\alpha_m = m/s \in [0, 1]$. In this way, each subinterval $[t_i, t_{i+1}]$ is uniformly sampled and the knots are always included in the discretization.

Sensitivity analysis

For the solution of the minimum time problem, the following directional derivatives are needed by the Gonzaga *et al.* (1980) algorithm:

$$\langle \nabla f(\mathbf{h}), \mathbf{d} \rangle = \sum_{i=1}^{n-1} d_i,$$
$$\langle \nabla g^l(\mathbf{h}), \mathbf{d} \rangle = \pm \sum_{k=1}^{n-1} \frac{\partial \dot{Q}_l(\tau, \mathbf{h})}{\partial h_k} d_k, \quad \text{with} \quad \tau = t_i \text{ or } t_{ji}^*,$$
$$\langle \nabla_{\mathbf{h}}\phi^j(\tau, \mathbf{h}), \mathbf{d} \rangle = \pm \sum_{k=1}^{n-1} \frac{\partial u_j(\tau, \mathbf{h})}{\partial h_k} d_k, \quad \text{with} \quad \tau = t_{im}. \tag{8}$$

In (8), the evaluation of spline (namely $Q_j, \dot{Q}_j, \ddot{Q}_j$) sensitivity with respect to changes of the generic time interval h_k is required. As a first step, the sensitivity of the solution to (2) (i.e. of knots accelerations ω_{ji}) w.r.t. variations of h_k has to be derived. Since

$$\mathbf{A}(\mathbf{h}) \frac{\partial \Omega_j}{\partial h_k} = \frac{\partial \mathbf{b}_j(\mathbf{h})}{\partial h_k} - \frac{\partial \mathbf{A}(\mathbf{h})}{\partial h_k} \Omega_j \triangleq \bar{\mathbf{b}}_j^{(k)},$$

the sensitivity $\partial \Omega_j / \partial h_k$ is the solution of a linear system having the same tridiagonal coefficient matrix as in (2) with constant $\bar{\mathbf{b}}_j^{(k)}$ in place of $\mathbf{b}_j$. Thus, it can be found using again the recursive algorithm (3). Letting $\omega_{ji}^{(k)} \triangleq \partial \omega_{ji} / \partial h_k$, these are obtained as

$$\omega_{jn}^{(k)} = \bar{\omega}_{jn}^{(k)}, \quad \omega_{ji}^{(k)} = \bar{\omega}_{ji}^{(k)} - K_i \omega_{j,i+1}^{(k)}, \quad i = n-1, \ldots, 1,$$

with

$$\bar{\omega}_{j1}^{(k)} = \frac{\bar{b}_{j1}^{(k)}}{a_{11}}, \quad \bar{\omega}_{ji}^{(k)} = \frac{\bar{b}_{ji}^{(k)} - \bar{\omega}_{j,i-1}^{(k)} a_{i,i-1}}{a_{ii} - K_{i-1} a_{i,i-1}}, \quad i = 2, \ldots, n.$$

The elements of vector $\bar{\mathbf{b}}_j^{(k)}$ are

$$\bar{b}_{ji}^{(k)} = 0, \quad i = 1, \ldots, k-1, k+2, \ldots, n,$$
$$\bar{b}_{jk}^{(k)} = -2\omega_{jk} - \omega_{j,k+1} - \frac{6(q_{j,k+1} - q_{jk})}{h_k^2},$$
$$\bar{b}_{j,k+1}^{(k)} = -\omega_{jk} - 2\omega_{j,k+1} + \frac{6(q_{j,k+1} - q_{jk})}{h_k^2}.$$

A more explicit treatment of the above expressions is pursued in (De Luca *et al.*, 1988). The above analysis is used to compute the sensitivity of the jth ($j = 1, \ldots, N$) spline (1) at a generic mesh point t_{im} ($i = 1, \ldots, n-1; m = 0, \ldots, s$) which is

$$\begin{aligned}\frac{\partial Q_{ji}(t_{im})}{\partial h_k} = \frac{h_i^2}{6} &[(1-\alpha_m)^3 \omega_{ji}^{(k)} + \alpha_m^3 \omega_{j,i+1}^{(k)} \\ &- \alpha_m \omega_{j,i+1}^{(k)} - (1-\alpha_m)\omega_{ji}^{(k)}] \\ &+ \delta_{ik} \frac{h_i}{3} [(1-\alpha_m)^3 \omega_{ji} + \alpha_m^3 \omega_{j,i+1} \\ &- \alpha_m \omega_{j,i+1} - (1-\alpha_m)\omega_{ji}]\end{aligned}$$

for $k = 1, \ldots, n-1$, δ_{ik} being the Kronecker delta. Similarly, the velocity sensitivity at t_{im} is

$$\begin{aligned}\frac{\partial \dot{Q}_{ji}(t_{im})}{\partial h_k} = &\frac{h_i}{2} [\alpha_m^2 \omega_{j,i+1}^{(k)} - (1-\alpha_m)^2 \omega_{ji}^{(k)}] \\ &- \frac{h_i}{6} (\omega_{j,i+1}^{(k)} - \omega_{ji}^{(k)}) \\ &+ \delta_{ik} \left[\frac{\alpha_m^2 \omega_{j,i+1} - (1-\alpha_m)^2 \omega_{ji}}{2} \right. \\ &\left. - \frac{q_{j,i+1} - q_{ji}}{h_i^2} - \frac{\omega_{j,i+1} - \omega_{ji}}{6} \right],\end{aligned}$$

while at the zero-acceleration instants one has

$$\begin{aligned}\frac{\partial \dot{Q}_{ji}(t_{ji}^*)}{\partial h_k} = &- \frac{h_i}{6} (\omega_{j,i+1}^{(k)} - \omega_{ji}^{(k)}) \\ &- \frac{h_i(\omega_{j,i+1}^2 \omega_{ji}^{(k)} - \omega_{ji}^2 \omega_{j,i+1}^{(k)})}{2(\omega_{j,i+1} - \omega_{ji})^2} \\ &- \delta_{ik} \left[\frac{\omega_{j,i+1} - \omega_{ji}}{6} + \frac{q_{j,i+1} - q_{ji}}{h_i^2} \right. \\ &\left. + \frac{\omega_{ji}\omega_{j,i+1}}{2(\omega_{j,i+1} - \omega_{ji})} \right].\end{aligned}$$

Finally, the sensitivity of spline acceleration is simply

$$\frac{\partial \ddot{Q}_{ji}(t_{im})}{\partial h_k} = \alpha_m \omega_{j,i+1}^{(k)} + (1-\alpha_m)\omega_{ji}^{(k)}.$$

These quantities enter directly into the sensitivity of the functional constraints:

$$\begin{aligned}\frac{\partial u_j}{\partial h_k} = \sum_{l=1}^{N} \Bigg(&\sum_{r=1}^{N} \frac{\partial m_{jl}}{\partial Q_r} \frac{\partial Q_r}{\partial h_k} \ddot{Q}_l + m_{jl} \frac{\partial \ddot{Q}_l}{\partial h_k} \\ &+ \frac{\partial c_j}{\partial Q_l} \frac{\partial Q_l}{\partial h_k} + \frac{\partial c_j}{\partial \dot{Q}_l} \frac{\partial \dot{Q}_l}{\partial h_k} + \frac{\partial e_j}{\partial Q_l} \frac{\partial Q_l}{\partial h_k} \Bigg), \end{aligned} \tag{9}$$

where m_{jl}, c_j and e_j are the elements of $\mathbf{M}$, $\mathbf{c}$ and $\mathbf{e}$ in (6). Note that in (9) derivatives of the dynamic model terms w.r.t. joint positions and velocities appear. These depend on the specific robot arm and can be computed automatically using symbolic manipulation languages (Neuman and Murray, 1984).

Numerical results

The proposed approach has been used to generate minimum time smooth spline trajectories for two different two-link SCARA-type robots. Programs were written in Fortran 77 on an AT personal computer and, at each iteration, the routine *E04NAF* of the NAG Workstation Library was used to solve the quadratic programming subproblem which provides the search direction. The dynamic model (6) of both arms takes on the explicit form

$$u_1 = (H_1 + 2H_2 \cos q_2)\ddot{q}_1 + (H_3 + H_2 \cos q_2)\ddot{q}_2 - H_2(2\dot{q}_1\dot{q}_2 + \dot{q}_2^2) \sin q_2$$
$$u_2 = (H_3 + H_2 \cos q_2)\ddot{q}_1 + H_3 \ddot{q}_2 + H_2 \dot{q}_1^2 \sin q_2,$$

TABLE 1. PARAMETERS OF THE TWO ROBOTS USED IN THE EXAMPLES

	l_1 (m)	l_2 (m)	d_2 (m)	m_2 (kg)	m_p (kg)	J_1 (kg m^2)	J_2 (kg m^2)	J_p (kg m^2)
Example 1	0.5	0.5	0.25	1	0	0.084	0.084	0
Example 2	0.4	0.25	0.125	15	6	1.6	0.34	0.01

with

$$H_1 = J_1 + J_2 + J_p + m_2 l_1^2 + m_p(l_1^2 + l_2^2),$$
$$H_2 = m_2 l_1 d_2 + m_p l_1 l_2, \quad H_3 = J_2 + J_p + m_p l_2^2.$$

Note that $\mathbf{e}(\mathbf{q}) = \mathbf{0}$ since the motion is constrained on a horizontal plane. In the above equations m_i, l_i and J_i $(i = 1, 2)$ are the mass, length and moment of inertia w.r.t. the axis of the driving joint for link i, while m_p and J_p are the mass and centroidal inertia of the payload. Also, d_2 is the distance between the axis of the second joint and the center of mass of the second link.

As a first example, a very light robot arm has been considered, whose parameters are reported in Table 1. The limit values are $V_1 = V_2 = 2\,\text{rad sec}^{-1}$ for the joint velocity, $U_1 = 7$, $U_2 = 2\,\text{Nm}$ for the torques. The robot task requires the arm to move along a C^2-spline trajectory passing through a sequence of six joint knots (in rads): $\{q_1\} = \{0, 0.5, 0.75, 1, 1.25, 1.5\}$, $\{q_2\} = \{0, -0.5, -1, -1.5, -1, 0.5\}$, with zero initial and final velocity. The chosen initial design parameter is $\mathbf{h} = [1, 0.5, 0.5, 0.5, 0.5]^T$, so that $t_n = 3\,\text{sec}$. On the resulting spline, velocity and torque of the second joint are both unfeasible, reaching the values of $4\,\text{rad sec}^{-1}$ and $-2.7\,\text{Nm}$. Figures 1–3 refer to the optimal solution found after 40 iterations. The optimal design parameter vector is $\mathbf{h}^* = [0.37, 0.25, 0.34, 0.43, 1.07]^T$, from which $t_n^* = 2.46\,\text{sec}$. Both torques are saturated at the initial instant, while the second joint velocity saturates twice, near the second and fourth trajectory knots.

The obtained results deserve some comments. It is known that the minimum time torque profile along a parameterized trajectory is such that at each instant at least one actuator provides its maximum torque (Bobrow *et al.*, 1985). The reason for the absence of these saturated "flat tops" in the obtained profiles is twofold. First, the presence of velocity bounds prevents the torques from reaching their maximum values. Second, the requirement of a continuous acceleration excludes a bang-bang form for the torques.

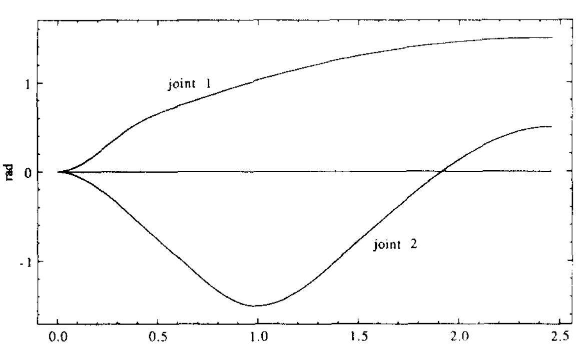

FIG. 1. Example 1. Optimal solution: position.

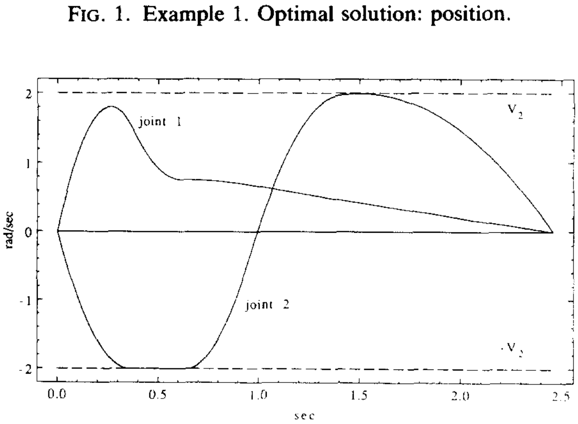

FIG. 2. Example 1. Optimal solution: velocity.

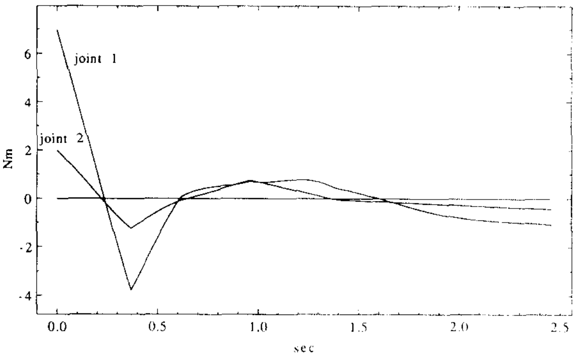

FIG. 3. Example 1. Optimal solution: torque.

As a second example, a planar motion of an IBM 7535 robot has been considered. Its parameters are shown in Table 1, while the bounds are $V_1 = V_2 = 1\,\text{rad sec}^{-1}$, and $U_1 = 9$, $U_2 = 25\,\text{Nm}$. The task is specified by a different sequence of six knots in the joint space: $\{q_1\} = \{q_2\} = \{0.1, 0.2, 0.25, 0.3, 0.35, 0.4\}$ (rads), again with zero initial and final velocity. The initial design parameter is $\mathbf{h} = [0.3, 0.3, 0.3, 0.3, 0.3]^T$, adding up to $t_n = 1.5\,\text{sec}$. The joint torques on the corresponding path are unfeasible, reaching the values of 45.6 Nm for the first joint and -12.2 Nm for the second. Feasibility is recovered after 5 iterations, with $t_n = 1.31\,\text{sec}$. Note that feasibility is recovered with a lower value of the objective function, as a result of the combined phase I–phase II algorithm. Figures 4–6 refer to the solution found after 34 iterations. The optimal design parameter is $\mathbf{h}^* = [0.29, 0.07, 0.07, 0.08, 0.2]^T$, giving a traveling time $t_n^* = 0.71\,\text{sec}$.

It is interesting to note that the obtained torques approximate a bang-bang behavior, although such a profile is outside the C^2-class. Two intermediate time intervals (i.e. h_2 and h_3) are in fact brought close to zero, while saturated torques are obtained in the first and in the last two intervals. These torque solutions are allowed since there is no velocity saturation in this case. If acceleration continuity is relaxed on our final path, the algorithm of Shin and McKay (1985) can be applied to obtain a lower minimum time solution.

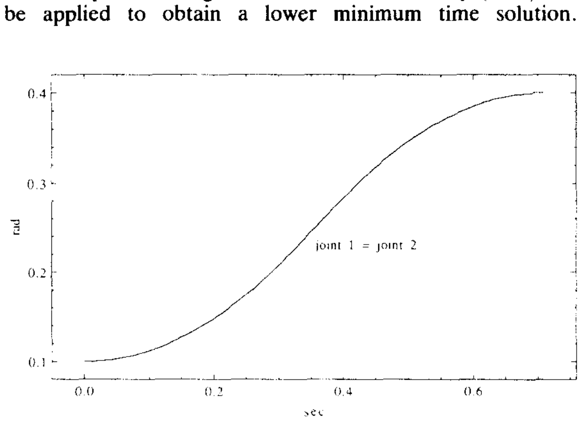

FIG. 4. Example 2. Optimal solution: position.

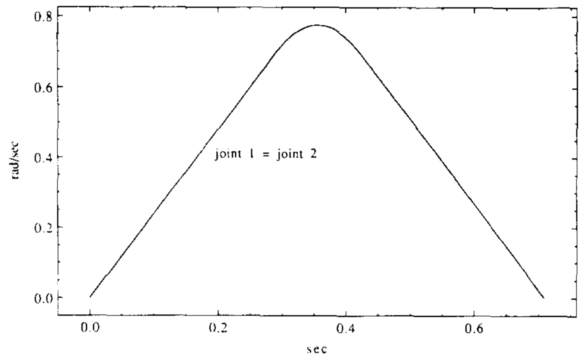

FIG. 5. Example 2. Optimal solution: velocity.

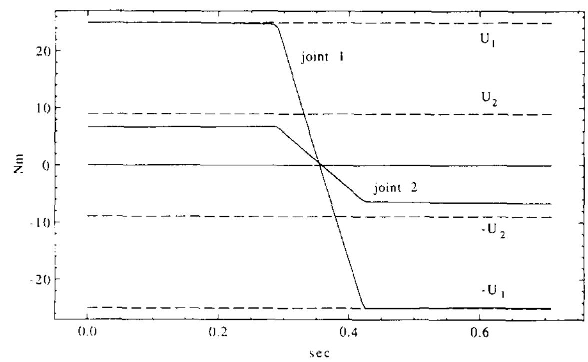

FIG. 6. Example 2. Optimal solution: torque.

However, only a negligible improvement is gained with respect to the proposed method, since $t_n^* = 0.707$ sec results. Other simulations, together with further details on the optimization algorithm, are reported in De Luca *et al.* (1988).

Conclusions

A new method has been presented for minimizing a general objective function along a spline trajectory under kinematic and dynamic constraints. In particular, minimization of the total traveling time along a path specified by a sequence of knots has been considered. The peculiarity of the proposed approach stands in the following aspects: (i) the problem is parameterized by a finite number of variables (i.e. the time intervals between the knots), thus allowing the use of an efficient nonlinear programming algorithm; (ii) general constraints on the robot state can be included directly in the formulation; (iii) the optimization is performed over the class of C^2-splines, thus ensuring smooth generated trajectories, which are graceful for the mechanical structure of the robot. The inclusion of constraints on both velocity and torque, together with the continuity requirement assumed for the trajectory acceleration, produces interesting minimum time torque profiles.

The numerical optimization algorithm used is very robust and efficient, being based on gradient information. The sensitivity of the spline with respect to changes of the time intervals between the knots has been explicitly derived. The obtained expressions can prove useful also in purely kinematic approaches to robot trajectory planning, as well as in other applications.

Use of the proposed method in optimization problems with more general objective functions under different types of constraints, like velocity-dependent torque bounds or joint limits, requires only little additional complexity. Nondifferentiable functions may also be treated, following the theoretical developments of the same basic algorithm as presented in (Polak *et al.*, 1983). This is of interest for tackling robot trajectory planning problems in the presence of obstacles, where nondifferentiable distance functions come into play (Gilbert and Johnson, 1985).

References

Bobrow, J. E. (1988). Optimal robot path planning using the minimum-time criterion. *IEEE J. Robotics Aut.*, **RA-4,** 443–450.

Bobrow, J. E., S. Dubowsky and J. S. Gibson (1985). Time-optimal control of robotic manipulators along specified paths. *Int. J. Robotics Res.*, **4,** 3–17.

de Boor, C. (1978). *A Practical Guide to Splines.* Springer, Berlin.

De Luca, A., L. Lanari and G. Oriolo (1988). Generation and computation of optimal smooth trajectories for robot arms. DIS Report 13.88, Università di Roma "La Sapienza", Rome.

Geering, H. P., L. Guzzella, S. A. R. Hepner and C. H. Onder (1986). Time-optimal motions of robots in assembly tasks. *IEEE Trans. Aut. Control,* **AC-31,** 512–518.

Gilbert, E. G. and D. W. Johnson (1985). Distance functions and their application to robot path planning in the presence of obstacles. *IEEE J. Robotics Aut.*, **RA-1,** 21–30.

Gonzaga, C., E. Polak and R. Trahan (1980). An improved algorithm for optimization problems with functional inequality constraints. *IEEE Trans. Aut. Control,* **AC-25,** 443–450.

Hollerbach, J. M. (1984). Dynamic scaling of manipulator trajectories. *ASME J. Dyn. Syst. Meas. Control,* **106,** 102–106.

Lin, C. S., P. R. Chang and J. Y. S. Luh (1983). Formulation and optimization of cubic polynomial joint trajectories for industrial robots. *IEEE Trans. Aut. Control,* **AC-28,** 1067–1073.

Neuman, C. P. and J. J. Murray (1984). Linearization and sensitivity functions of dynamic robot models. *IEEE Trans. Syst. Man Cybern.*, **SMC-14,** 805–818.

Polak, E., D. Q. Mayne and Y. Wardi (1983). On the extension of constrained optimization algorithms from differentiable to nondifferentiable problems. *SIAM J. Control and Optimiz.*, **21,** 179–203.

Sahar, G. S. and J. M. Hollerbach (1986). Planning of minimum-time trajectories for robot arms. *Int. J. Robotics Res.*, **5,** 90–100.

Shin, K. G. and N. D. McKay (1985). Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Trans. Aut. Control,* **AC-30,** 531–541.

Stoer, J. and R. Bulirsch (1980). *Introduction to Numerical Analysis.* Springer, Berlin.

Weinreb, A. and A. E. Bryson (1985). Optimal control of systems with hard control bounds. *IEEE Trans. Aut. Control,* **AC-30,** 1135–1138.