A new method for smooth trajectory planning of robot manipulators

A. Gasparetto *, V. Zanotto

Dipartimento di Ingegneria Elettrica, Gestionale e Meccanica, Università degli Studi di Udine, Via delle Scienze, 208-33100 Udine, Italy

Received 25 April 2005; received in revised form 3 April 2006; accepted 6 April 2006
Available online 5 June 2006

Abstract

A new method for smooth trajectory planning of robot manipulators is described in this paper. In order to ensure that the resulting trajectory is smooth enough, an objective function containing a term proportional to the integral of the squared jerk (defined as the derivative of the acceleration) along the trajectory is considered. Moreover, a second term, proportional to the total execution time, is added to the expression of the objective function. In this way it is not necessary to define the total execution time before running the algorithm. Fifth-order B-splines are then used to compose the overall trajectory. With respect to other trajectory optimization techniques, the proposed method enables one to set kinematic constraints on the robot motion, expressed as upper bounds on the absolute values of velocity, acceleration and jerk. The algorithm has been tested in simulation yielding good results, which have also been compared with those provided by another important trajectory planning technique.

© 2006 Elsevier Ltd. All rights reserved.

Keywords: Trajectory planning; Smoothness; Jerk; Robot manipulators; Optimization; B-splines

1. Introduction

A fundamental problem in robotics consists in trajectory planning, which may be defined in this way: find a temporal motion law along a given geometric path, such as certain requirements set on the trajectory properties are fulfilled. Trajectory planning is devoted to generate the reference inputs for the control system of the manipulator, so as to be able to execute the motion. The geometric path, the kinematic and dynamic constraints are the inputs of the trajectory planning algorithm, whereas the trajectory of the joints (or of the end effector), expressed as a time sequence of position, velocity and acceleration values, is the output.

The geometric path is usually defined in the operating space, i.e. with reference to the end effector of the robot, since both the task to perform and the obstacles to avoid can be more naturally described in this space.
On the other side, trajectory planning is normally carried out in the joint space of the robot, after a kinematic inversion of the given geometric path. The joint trajectories are then obtained by means of interpolating functions which meet the imposed kinematic and dynamic constraints.

Planning a trajectory in the joint space rather than in the operating space has a major advantage, namely that the control system acts on the manipulator joints rather than on the end effector, so it would be easier to adjust the trajectory according to the design requirements if working in the joint space. Moreover, trajectory planning in the joint space would allow to avoid the problems arising with kinematic singularities and manipulator redundancy.

The main disadvantage of planning the trajectory in the joint space is that, given the planned trajectory in the joint space, the motion actually performed by the robot end effector is not easily foreseeable, due to the non-linearities introduced when transforming the trajectories of the joints into the trajectories of the end effector through direct kinematics.

Apart from the particular strategy adopted, the motion laws generated by the trajectory planner must fulfill the constraints set a priori on the maximum values of the generalized joint torques, and must be such that no mechanical resonance mode is excited. This can be achieved by forcing the trajectory planner to generate smooth trajectories, i.e. trajectories with good continuity features: in particular, it would be desirable to obtain trajectories with continuous joint accelerations, so that the absolute value of the jerk (i.e. of the derivative of the acceleration) keeps bounded. Limiting the jerk is very important, because high jerk values can wear out the robot structure, and heavily excite its resonance frequencies. Vibrations induced by non-smooth trajectories can damage the robot actuators, and introduce large errors while the robot is performing tasks such as trajectory tracking. Moreover, low-jerk trajectories can be executed more rapidly and accurately.

Almost every technique found in the scientific literature on the trajectory planning problem is based on the optimization of some parameter or some objective function. The most significant optimality criteria are:

1. minimum execution time,
2. minimum energy (or actuator effort),
3. minimum jerk.

Besides the aforementioned approaches, some hybrid optimality criteria have also been proposed (e.g. time-energy optimal trajectory planning).

1.1. Minimum-time trajectory planning

Minimum-time algorithms were the first trajectory planning techniques proposed in the scientific literature because they were tightly linked to the need of increasing the productivity in the industrial sector. The first interesting methods of this kind [1,2] are developed in the position-velocity phase plane. The main idea is to use the curvilinear abscissa $\theta$ of the path as a parameter, in order to express the dynamic equation of the manipulator in a parametric form. An alternative approach is proposed in [3,4], where dynamic programming techniques are employed. However, the aforementioned techniques generate trajectories with discontinuous values of accelerations and joint torques, because the dynamic models used for trajectory computation assume the robot members as perfectly rigid and neglect the actuator dynamics. This leads to two undesired effects: first, the real actuators of the robot cannot generate discontinuous torques, which causes the joint motion to be always delayed with respect to the reference trajectory. The accuracy in trajectory following is then greatly reduced and the so-called chatter phenomenon may eventually occur, consisting in high frequency vibrations that can damage the manipulator structure. Second, the time-optimal control requires saturation of at least one robot actuator at any time instant, so the controller cannot correct the tracking errors arising from disturbances or modeling errors.

In order to overcome such problems, other approaches [5,6] impose some limits on the actuator jerks, defined as the variation rates of the joint torques. In this way, the generated trajectory will not be exactly time-optimal, albeit close to the optimality value; however, the generated trajectories can be effectively implemented and more advanced control strategies can be applied.
In order to generate trajectories with continuous accelerations, a common strategy is to use smooth trajectories, such as the spline functions, that have been extensively employed in the scientific literature on both kinematic and dynamic trajectory planning.

The first formalization of the problem of finding the optimal curve interpolating a sequence of nodes in the joint space, obtained through kinematic inversion of a discrete set of points representing position and orientation of the reference frame linked to the end effector of the robot, can be found in [7]. In [8] the same algorithm is presented, but the trajectories are expressed by means of cubic B-splines. In [9] the optimization algorithm proposed in [7] is modified, so that it can deal also with dynamic constraints and with objective functions of a more general type; however, the reported simulations still refer to the trajectory planning problem with kinematic constraints.

In [10] a method is proposed, to compute point-to-point minimum-time trajectories using uniform B-splines. The dynamic model of the manipulator is considered, and the problem of semi-infinite constraints resulting from the algorithm is bypassed by sampling a certain number of points.

All the algorithms based on the optimization of cubic splines mentioned in the foregoing yield a local optimal solution, which may or may not be the global optimum. So, some algorithms have been developed, aimed at finding a global optimal solution for the trajectory planning problem for robotic manipulators. Examples are [11], where the proposed algorithm is based on the so-called interval analysis, and [12] where a hybrid technique using genetic algorithms is put forward.

1.2. Minimum-energy trajectory planning

Planning the robot trajectory using energetic criteria provides several advantages. On one hand, it yields smooth trajectories resulting easier to track and reducing the stresses to the actuators and to the manipulator structure. Moreover, saving energy may be desirable in several applications, such as those with a limited capacity of the energy source (e.g. robots for spatial or submarine exploration).

Examples of energy optimal trajectory planning are provided in [13–15]. In [13,14] point-to-point trajectories with minimal energy are considered, with upper bounds on the amplitude of the control signals and the joint velocities. In [15] a trajectory with motion constraints set on the end effector is optimized. The cost function is given by the time integral of the squared joint torques, and the trajectories are expressed by means of cubic B-splines. The resulting motion minimizes the actuators' effort.

Paper [16] deals with time-energy optimal trajectories, i.e. the cost function has a term linked to the execution time and a term expressing the energy spent. The trade-off between the two needs can be adjusted by changing the respective weights.

Other examples of time-energy optimal trajectories are given in [17,18]. In [17] a cubic spline trajectory is considered, subject to kinematic constraints on the maximum values of velocity, acceleration and jerk. In [18] a point-to-point trajectory parametrized by means of cubic B-splines is considered.

1.3. Minimum-jerk trajectory planning

The importance of generating trajectories that do not require abrupt torque variations has already been remarked. In [5,6] this has been obtained by setting upper bounds on the rate of torque variations, but in this way the third-order dynamics of the manipulator must be computed. An indirect method to get the same results is to set an upper limit on the jerk, defined as the time derivative of the acceleration of the manipulator joints.

The positive effects induced by a jerk minimization are:

- errors during trajectory tracking are reduced,
- stresses to the actuators and to the structure of the manipulator are reduced,
- excitation of resonance frequencies of the robot is limited,
- a very coordinated and natural robot motion is yielded.

The last effect appears very interesting; indeed, some studies [19,20] show that the movement of a human arm seem to satisfy an optimization criterion linked to the rate of variation of the joint acceleration. So, it may
be inferred that minimum-jerk trajectories are an example of optimization according to physical criteria imitating the capacity of natural coordination of a human being.

In [21] the solutions for two point-to-point minimum-jerk trajectories are analytically obtained; the optimization, obtained through the Pontryagin principle, concerns two objective functions, namely the time integral of the squared jerk and the maximum absolute value of the jerk (minimax approach).

An interesting approach is described in [22], where the interpolation is done through trigonometric splines, that ensure the continuity of the jerk. The interpolation through trigonometric splines features a sound locality property, i.e. if the value of some node in the input sequence is changed, only the two splines having that node as their common point should be computed again, not the whole trajectory: this property can be usefully exploited for instance to bypass an obstacle in real time.

In [23,24] the so-called interval analysis is used to develop an algorithm that globally minimizes the maximum absolute value of the jerk along a trajectory whose execution time is set a priori: hence, an approach of the type minimax is used. The trajectories are expressed by means of cubic splines and the intervals between the via-points are computed so that the lowest possible jerk peak is produced. A comparison between the simulation results of the algorithm proposed in [23] and those of the method described in [22] is drawn.

An important remark must be done at the end of this literature overview: in the case of trajectory planning along a given path, all jerk-minimization algorithms that could be found consider an execution time set a priori and do not accept any kinematic constraint. On the other hand, the trajectory planning technique proposed in this paper does not require the execution time to be imposed; moreover, kinematic constraints are taken into account when generating the optimal trajectory.

With respect to minimum-jerk optimization techniques that could be found in the scientific literature, the method described in this work enables one to define kinematic constraint on the robot motion before running the algorithm. Such constraints are expressed as upper bounds on the absolute values of velocity, acceleration and jerk for all robot joints, so that any physical limitation of the real manipulator can be taken into account when planning its trajectory.

The paper is organized as follows: In Section 2 the optimization problem is formulated, namely the objective function to minimize is defined. Fifth-order B-splines are then chosen to define the trajectory which is to solve the optimization problem: the objective function and the kinematic constraints are rewritten accordingly. Quintic splines have been proposed for trajectory generation in some scientific papers (see for instance [25–28]). Section 3 describes the whole algorithm, which is based on an iterative minimization procedure, and Section 4 describes its execution. Section 5 shows the results of some simulation that have been carried out using the same input data as another important algorithm found in the literature, in order to be able to compare the results.

2. The optimization problem

As stated in the foregoing, it is desirable that trajectories generated by planning algorithms feature sufficient smoothness properties, so as to avoid to excite mechanical resonance modes of the manipulator structure. This can be achieved by if the acceleration of the planned trajectory is a continuous function, so that the value of the jerk (its derivative) keeps bounded. Limitation of the value of the jerk is an indirect way to bound the variation rate of the joint torques, without any need to keeping into account the dynamic model of the manipulator in the planning algorithm.

The trajectory planning technique described in this paper assumes that the geometric path, generated a priori by an upper-level path planner, is given in the form of a sequence of via-points in the operating space of the robot, which represent successive positions and orientations of the end effector of the manipulator; obstacle avoidance problems are assumed as already solved by the upper-level path planer and will not be considered. The proposed technique generates an optimal trajectory for the robot, by associating temporal information to the pre-planned geometric path, according to an optimality criterion based on the minimization of some objective function that will be defined below, and by taking into account any kinematic constraints, expressed by means of upper bounds on velocity, acceleration and jerk.

There are several algorithms where the value of the jerk appears in the objective function. For instance, Simon and Isik [22] minimize the integral of the squared jerk, while Piazzi and Visioli [23] use a minimax approach to minimize the maximum value of the jerk along the trajectory.
However, these techniques consider the execution time as known (and set a priori); moreover, it is not possible to set any kind of kinematic constraint on the trajectory, because they are not taken into account.

On the contrary, the algorithm presented in this paper generates an optimal trajectory such as

- it is not required to set the execution time a priori,
- kinematic constraints on the resulting trajectory (i.e. upper bounds on the values of velocity, acceleration and jerk of the robot joints) can be defined before running the algorithm.

The objective function adopted in proposed technique is given by the sum of two terms having opposite effects:

- the first term is proportional to the execution time,
- the second term is proportional to the integral of the squared jerk.

Of course, reducing the value of the first term of the objective function will lead to trajectories featuring large values of the kinematic quantities (velocity, acceleration and jerk), while reducing the second term will lead to a smoother trajectory.

The trade-off between these two tendencies can be performed by suitably adjusting the weights of the two terms of the objective function: a larger weight of the jerk term will lead to smoother but slower trajectories, while a larger weight of the time term will lead to faster but less smooth trajectories.

Hence, the optimal trajectory planning problem can be formulated as

\[
\begin{align*}
\text{find:} & \\
& \min k_T N \sum_{i=1}^{N-1} h_i + k_1 \sum_{j=1}^{N} \int_{0}^{t_f} (\dddot{q}_j(t))^2 \, dt \\
\text{subject to:} & \\
& |\dot{q}_j(t)| \leq V_{C_j}, \quad j = 1, \ldots, N \\
& |\ddot{q}_j(t)| \leq W_{C_j}, \quad j = 1, \ldots, N \\
& |\dddot{q}_j(t)| \leq J_{C_j}, \quad j = 1, \ldots, N
\end{align*}
\]

(1)

The meaning of the symbols appearing in (1) is explained in Table 1.

By solving the optimization problem (1), the vector of the time intervals \( h_i \) between any pair of consecutive via-points is computed.

Of course, the trajectory solving the above defined optimization problem must also meet the interpolation conditions for all the via-points, as well as the initial and final conditions for velocity, acceleration and jerk.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N )</td>
<td>Number of robot joints</td>
</tr>
<tr>
<td>( V_p )</td>
<td>Number of via-points</td>
</tr>
<tr>
<td>( h_i )</td>
<td>Time interval between two via-points</td>
</tr>
<tr>
<td>( \dot{q}_j(t) )</td>
<td>Velocity of the ( j )th joint</td>
</tr>
<tr>
<td>( \ddot{q}_j(t) )</td>
<td>Acceleration of the ( j )th joint</td>
</tr>
<tr>
<td>( \dddot{q}_j(t) )</td>
<td>Jerk of the ( j )th joint</td>
</tr>
<tr>
<td>( k_T )</td>
<td>Weight of the term proportional to the execution time</td>
</tr>
<tr>
<td>( k_1 )</td>
<td>Weight of the term proportional to the jerk</td>
</tr>
<tr>
<td>( t_f )</td>
<td>Total execution time of the trajectory</td>
</tr>
<tr>
<td>( V_{C_j} )</td>
<td>Velocity limit for the ( j )th joint (symmetrical)</td>
</tr>
<tr>
<td>( W_{C_j} )</td>
<td>Acceleration limit for the ( j )th joint (symmetrical)</td>
</tr>
<tr>
<td>( J_{C_j} )</td>
<td>Jerk limit for the ( j )th joint (symmetrical)</td>
</tr>
</tbody>
</table>
3. Definition of the trajectory by means of fifth-order B-splines

A technique to solve the optimization problem (1), based on fifth-order B-splines, will be described in the following.

We briefly recall here that a B-spline of degree \( p \) and order \( k = p + 1 \) is a spline curve \( B_p(t) \) expressed in the so-called \( B \)-form, namely as a linear combination of polynomials \( N_{i,p}(t) \) of degree \( p \), called base or blending functions, weighted by some coefficients \( Q_i \) named control points. The curve is built on a sequence of nodes \( t_i \) and the base functions are defined recursively by means of the De Boor formula [29]

\[
B_p(t) = \sum_{i=1}^{n+1} Q_i \cdot N_{i,p}(t)
\]

with

\[
N_{i,p}(t) = \begin{cases} 
\frac{t-t_i}{t_{i+p}-t_i} N_{i,p-1}(t) + \frac{t_{i+p+1}-t}{t_{i+p+1}-t_{i+1}} N_{i+1,p-1}(t) \\
1, & \text{for } t_i \leq t < t_{i+1} \\
0, & \text{elsewhere}
\end{cases}
\]

where \( k = p + 1 \) and \( m = n + p + 1 \).

Table 2 contains the definition of the symbols used in (2) and (3).

The sequence of nodes is said to be clamped when the values at the extremities have multiplicity \( k \), i.e. are repeated \( k \) times, so that the control points at both ends of the trajectory coincide with the initial and final via-points, respectively.

3.1. Imposing the interpolation and boundary conditions

In order to obtain trajectories with no jerk at both ends, two virtual points have been introduced at the second and the second-last position of the sequence.

Given a sequence, obtained through a kinematic inversion, of \( vp + 2 \) via-points in the joint space of dimension \( N \), \( m + 1 = (vp + 2) + 2p = vp + 12 \) nodes are necessary to get a complete interpolation. Being \( m = (vp + 12) - 1 = n + p + 1 = n + 6 \), the number of control points needed is given by: \( n + 1 = vp + 6 \). So, enough degrees of freedom are available to ensure that the trajectory passes through the via-points, and to impose the initial and final conditions for velocity, acceleration and jerk.

Some remarks about the derivative of a curve represented through a B-spline are now necessary, in order to properly define the resolution procedure of the interpolation problem.

The derivative of a B-spline of degree \( p \) is a B-spline of degree \( p - 1 \). Considering a single joint for sake of simplicity, let \( CPQ_i (i = 1, \ldots, n + 1) \) represent the control points of the trajectory; \( Uq_i, (i = 1, \ldots, m + 1) \) the nodes and \( CPV_i (i = 1, \ldots, n + 1) \) the control points of the trajectory derivative (i.e. the velocity).

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
<td>Degree of the B-spline</td>
</tr>
<tr>
<td>( k )</td>
<td>Order of the B-spline</td>
</tr>
<tr>
<td>( B_p(t) )</td>
<td>B-spline of degree ( p )</td>
</tr>
<tr>
<td>( N_{i,p}(t) )</td>
<td>Base function of degree ( p )</td>
</tr>
<tr>
<td>( Q_i )</td>
<td>Control point of the B-spline</td>
</tr>
<tr>
<td>( n + 1 )</td>
<td>Number of control points</td>
</tr>
<tr>
<td>( t_i )</td>
<td>Nodes</td>
</tr>
<tr>
<td>( m + 1 )</td>
<td>Number of nodes</td>
</tr>
</tbody>
</table>
The trajectory is expressed in the form

\[ q(t) = \sum_{i=1}^{n+1} \text{CPQ}_i \cdot N_{i,p}(t) \]  

(4)

and the following holds:

\[ \text{CPV}_i = \frac{p}{U_q_{i+p+1} - U_q_{i+1}} (\text{CPQ}_{i+1} - \text{CPQ}_i), \quad i = 1, \ldots, n \]  

(5)

The velocity can be defined on the sequence of nodes \( U_v \), built by discarding the values at the extremities of \( U_q \), thus obtaining a clamped sequence compatible with the degree \( p - 1 \) of the velocity

\[ v(t) = \sum_{i=1}^{n} \text{CPV}_i \cdot N_{i,p-1}(t) \]  

(6)

The boundary conditions on the velocity can then be imposed

\[ \text{CPV}_1 = \frac{p}{U_q_{p+2} - U_q_2} (\text{CPQ}_2 - \text{CPQ}_1) = -\frac{p}{U_q_{p+2} - U_q_2} \cdot \text{CPQ}_1 + \frac{p}{U_q_{p+2} - U_q_2} \cdot \text{CPQ}_2 = v_{\text{in}} \]

\[ \text{CPV}_n = \frac{p}{U_n_{p+1} - U_{n+1}} (\text{CPQ}_{n+1} - \text{CPQ}_n) \]

\[ = -\frac{p}{U_n_{p+1} - U_{n+1}} \cdot \text{CPQ}_n + \frac{p}{U_n_{p+1} - U_{n+1}} \cdot \text{CPQ}_{n+1} = v_{\text{fin}} \]  

(7)

Eq. (7) are to be added to the interpolation conditions of the via-points.

The same procedure is carried out in order to compute the control points for the acceleration. Let \( \text{CPA}_i \) \( (i = 1, \ldots, n - 1) \) be the control points of the acceleration curve. The following holds:

\[ \text{CPA}_i = \frac{p - 1}{U_a_{i+p} - U_a_{i+1}} (\text{CPV}_{i+1} - \text{CPV}_i), \quad i = 1, \ldots, n - 1 \]  

(8)

The acceleration can then be defined on the sequence of nodes \( U_a \), built by discarding the values at the extremities of \( U_v \), thus obtaining a clamped sequence compatible with the degree \( p - 2 \) of the acceleration

\[ a(t) = \sum_{i=1}^{n-1} \text{CPA}_i \cdot N_{i,p-2}(t) \]  

(9)

The boundary conditions on the accelerations can then be imposed

\[ \text{CPA}_1 = \frac{p - 1}{U_{a_{p+1}} - U_{a_2}} (\text{CPV}_2 - \text{CPV}_1) = a_{\text{in}} \]

\[ \text{CPA}_{n-1} = \frac{p - 1}{U_{a_{n+p-1}} - U_{a_{n+1}}} (\text{CPV}_n - \text{CPV}_{n-1}) = a_{\text{fin}} \]  

(10)

Using the (7), Eqs. (10) can be rewritten as

\[ \text{CPA}_1 = k_1 \cdot \text{CPQ}_1 + k_2 \cdot \text{CPQ}_2 + k_3 \cdot \text{CPQ}_3 = a_{\text{in}} \]

\[ \text{CPA}_{n-1} = k_{n-1} \cdot \text{CPQ}_{n-1} + k_n \cdot \text{CPQ}_n + k_{n+1} \cdot \text{CPQ}_{n+1} = a_{\text{fin}} \]  

(11)

Eqs. (11) are also to be added to the interpolation conditions of the via-points.

Again, the same procedure is carried out in order to compute the control points for the jerk. Let \( \text{CPJ}_i \) \( (i = 1, \ldots, n - 2) \) be the control points of the jerk curve, where

\[ \text{CPJ}_i = \frac{p - 2}{U_{a_{i+p-1}} - U_{a_{i+1}}}(\text{CPA}_{i+1} - \text{CPA}_i), \quad i = 1, \ldots, n - 2 \]  

(12)
The jerk can then be defined on the sequence of nodes \( U_j \), built by discarding the values at the extremities of \( U_a \), thus obtaining a clamped sequence compatible with the degree \( p - 3 \) of the jerk

\[
j(t) = \sum_{i=1}^{n-2} CPJ_i \cdot N_{i,p-3}(t)
\]  

(13)

The boundary conditions on the accelerations can then be imposed

\[
CPJ_1 = \frac{p - 2}{Ua_p - Ua_2} (CPA_2 - CPA_1) = f_{in}
\]

\[
CPJ_{n-2} = \frac{p - 2}{Ua_{n+p-3} - Ua_{n-1}} (CPA_{n-1} - CPA_{n-2}) = f_{fin}
\]

(14)

Using the (7) and the (11), Eqs. (14) can be rewritten as

\[
CPJ_1 = g_1 \cdot CPQ_{i1} + g_2 \cdot CPQ_{i2} + g_3 \cdot CPQ_{i3} + g_4 \cdot CPQ_{i4} = f_{in}
\]

\[
CPJ_{n-2} = g_{n-2} \cdot CPQ_{n-2} + g_{n-1} \cdot CPQ_{n-1} + g_n \cdot CPQ_n + g_{n+1} \cdot CPQ_{n+1} = f_{fin}
\]

(15)

Eqs. (15) are also to be added to the interpolation conditions of the via-points.

The general form of the linear system solving the interpolation problem using fifth-degree B-splines can now be obtained. Let \( VPR_{ij} \) and \( CPQ_{ij} \) be respectively the \( i \)th via-point and control point of the trajectory of the \( j \)th robot joint, the linear system (16) can be written for each joint \( j (j = 1, \ldots, N) \):

\[
\begin{cases}
- \frac{p}{Uq_{p+2} - Uq_2} \cdot CPQ_{j,1} + \frac{p}{Uq_{p+2} - Uq_2} \cdot CPQ_{j,2} = v_{j,in} \\
k_1 \cdot CPQ_{j,1} + k_2 \cdot CPQ_{j,2} + k_3 \cdot CPQ_{j,3} = a_{j,in} \\
g_1 \cdot CPQ_{j,1} + g_2 \cdot CPQ_{j,2} + g_3 \cdot CPQ_{j,3} + g_4 \cdot CPQ_{j,4} = j_{j,in} \\
\sum_{k=1}^{v+6} N_{p,k}(t_i) \cdot CPQ_{j,k} = VPR_{jj} \quad \text{with } i = 1, \ldots, vp \\
g_{n-2} \cdot CPQ_{j,n-2} + g_{n-1} \cdot CPQ_{j,n-1} + g_n \cdot CPQ_{j,n} + g_{n+1} \cdot CPQ_{j,n+1} = j_{j,fin} \\
k_{n-1} \cdot CPQ_{j,n-1} + k_n \cdot CPQ_{j,n} + k_{n+1} \cdot CPQ_{j,n+1} = a_{j,fin} \\
- \frac{p}{Uq_{n+p+1} - Uq_{n+1}} \cdot CPQ_{j,n} + \frac{p}{Uq_{n+p+1} - Uq_{n+1}} \cdot CPQ_{j,n+1} = v_{j,fin}
\end{cases}
\]

(16)

The \( N \) systems of the type (16) can be gathered into a compact form of the type

\[
A \Phi_j = B_j \quad \forall j = 1, \ldots, N
\]

(17)

where the unknowns of the linear system (17) are the values of the control points

\[
\Phi_j = \begin{bmatrix}
CPQ_{j,1} \\
\vdots \\
CPQ_{j,v+6}
\end{bmatrix} \quad \forall j = 1, \ldots, N
\]

(18)

The coefficient matrix \( A \), which results non-singular and band-diagonal, is unique for all joints, because it depends only on the intervals \( h_i = t_{i+1} - t_i \) between each pair of via-points. It must be noticed that \( A \) is not provided in analytical form, but requires the evaluation of the base functions at the time instants corresponding to the via-points; hence, a procedure for computation of the base functions must be included in the trajectory optimization algorithm.

3.2. Expression of the kinematic constraints

The expression of the kinematic constraints for a B-spline can be conveniently expressed recalling that any B-spline keeps within the convex hull associated to its control points.
This can proven thus: let \( f(t) = \sum_{i=1}^{n+1} Q_i \cdot N_{i,k}(t) \) be a B-spline (representing position, or velocity, or acceleration, or jerk) that is both upper- and lower-bounded (\( L_{\text{inf}} \leq f(t) \leq L_{\text{sup}} \)); the following relations hold:

\[
\begin{align*}
L_{\text{inf}} & \leq \sum_{i=1}^{n+1} Q_i \cdot N_{i,k}(t) \leq L_{\text{sup}} \\
L_{\text{inf}} \cdot \sum_{i=1}^{n+1} N_{i,k}(t) & \leq \sum_{i=1}^{n+1} Q_i \cdot N_{i,k}(t) \leq L_{\text{sup}} \cdot \sum_{i=1}^{n+1} N_{i,k}(t) \\
\sum_{i=1}^{n+1} L_{\text{inf}} \cdot N_{i,k}(t) & \leq \sum_{i=1}^{n+1} Q_i \cdot N_{i,k}(t) \leq \sum_{i=1}^{n+1} L_{\text{sup}} \cdot N_{i,k}(t)
\end{align*}
\]  

(19)

A sufficient condition for the validity of the (19) is given by

\[
L_{\text{inf}} \leq Q_i \leq L_{\text{sup}} \quad \forall i = 1, \ldots, n + 1
\]  

(20)

Hence, it has been proven that, in order to set upper- and lower-bounds on a B-spline (which can represent position, velocity, acceleration or jerk), it suffices to put some constraints on its control points. The advantage in doing that lies in the fact that the semi-infinite kinematic constraints of the type

\[
(L_{\text{inf}} \leq f(t) \leq L_{\text{sup}})
\]

imposed on the curves \( f(t) \) of velocity, acceleration and jerk, which would have to hold for any value of the time \( t \), are turned into a set of constraints (20) that must hold for just a finite number of points, namely the control points of the B-splines. This greatly reduces the computational complexity of the problem.

Namely, on the basis of the convex hull property and of the expressions for the control points of the derivative of a B-spline computed in the foregoing, the kinematic constraints can be easily formulated

\[
\begin{align*}
|CPV_{j,k}| & \leq VC_j, \quad k = 1, \ldots, n \\
|CPA_{j,k}| & \leq WC_j, \quad k = 1, \ldots, n - 1 \\
|CPJ_{j,k}| & \leq JC_j, \quad k = 1, \ldots, n - 2
\end{align*}
\]

(22)

where \( VC_j, WC_j, JC_j \) represent the (symmetric) constraints for velocity, acceleration and jerk of the \( j \)th joint, respectively. It should be remarked that the terms on the left-hand side are linked to the values of the control points \( CPQ_{j,k} \) through the (5), (8) and (12) which in turn are linked to the optimization parameters \( h_i \) through the (17).

It should be remarked again that (20) represents not a necessary, but just a sufficient condition for the validity of (21). So, if the finite set (20) is used to define the kinematic constraints of the trajectory, instead of the original ones (21), it should be kept in mind that a conservative approach is employed because not all the possibilities assumed in defining the kinematic model are fully exploited. In other words, the trajectories obtained by running the algorithm may result slower than those theoretically feasible.

Another type of kinematic constraint is due to the fact that the optimization parameters \( h_i \) are lower bound, because any interval between a pair of consecutive via-points cannot be run at infinite velocity. Hence, for each interval the relation

\[
\left| \frac{q_{j,i+1} - q_{j,i}}{h_i} \right| \leq VC_j
\]

(23)

must hold, so that the \( h_i \) must be such as

\[
h_i > w_i = \max_{j=1, \ldots, N} \left\{ \left| \frac{q_{j,i+1} - q_{j,i}}{VC_j} \right| \right\} > 0
\]

(24)

In other words, the execution time of every trajectory segment should be computed, corresponding to the maximum allowed velocity for each joint; the largest execution time, corresponding to the most restrictive condition, is then to be picked.
Finally, it remains to define the expression of the objective function (1) in order to be able to implement the algorithm and to run the simulations.

The generic form of the objective function for a trajectory defined in terms of B-splines is given by

$$\text{FOBJ} = k_1 \sum_{i=1}^{n+1} h_i + k_3 \sum_{j=1}^{N} \int_0^{t_f} \left( \sum_{k=1}^{n-2} \mathbf{CPJ}_{j,k} \cdot N_{p-3,k}(t) \right)^2 \, dt$$

(25)

However, the expression of the integral of the squared jerk appearing in (25) turns out to be very long and complex.

In order to compute the term \( \int_0^{t_f} \left( \sum_{k=1}^{n-2} \mathbf{CPJ}_{j,k} \cdot N_{p-3,k}(t) \right)^2 \, dt \), a first approach would be to use a numerical integration procedure (e.g. the \textit{quadl} function in MatLab), thus avoiding any form of analytical integration.

On the other hand, in order to get an exact instead of an approximate result, the analytical integration of the expression squared jerk has to be computed. This requires to determine first the analytical expression of the base functions \( N_{p-3,k} \) of degree 2, by means of the recursive De Boor formula (2). The final expression of the integral, obtained through a very lengthy and tedious analytical integration, would take many pages and will not be reported here. It must be noticed that running the algorithm using the exact expression of the squared jerk instead of a numerical integration procedure yields much better results in terms of execution time.

4. Running the algorithm

The trajectory planning algorithm described in the previous section can then be run in whatever simulation environment, by solving the minimization problem formulated in (25) through some dedicated software routine (usually, based on iteration).

The steps for running the algorithm are summarized below:

- starting from a given path in the operating space, a kinematic inversion is applied, so as to get a sequence of via-points in the joint space,
- the numeric values of kinematic constraints are set, on the basis of the structural constraints of the manipulator, and of design considerations,

<table>
<thead>
<tr>
<th>Joint</th>
<th>Via-points [deg]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>–10 Virtual</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>150</td>
</tr>
<tr>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>6</td>
<td>120</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Joint</th>
<th>V</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>–10</td>
<td>Virtual</td>
<td>60</td>
<td>20</td>
<td>Virtual</td>
<td>55</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>20</td>
<td></td>
<td>50</td>
<td>120</td>
<td></td>
<td>35</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>15</td>
<td></td>
<td>100</td>
<td>–10</td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>150</td>
<td></td>
<td>100</td>
<td>40</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>30</td>
<td></td>
<td>110</td>
<td>90</td>
<td></td>
<td>70</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>120</td>
<td></td>
<td>60</td>
<td>100</td>
<td></td>
<td>25</td>
</tr>
</tbody>
</table>

Table 3: Values of the via-point of the trajectory

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>2</td>
<td>95</td>
<td>60</td>
<td>66</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>75</td>
<td>85</td>
</tr>
<tr>
<td>4</td>
<td>150</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>5</td>
<td>130</td>
<td>90</td>
<td>75</td>
</tr>
<tr>
<td>6</td>
<td>110</td>
<td>80</td>
<td>70</td>
</tr>
</tbody>
</table>

Table 4: Values of the kinematic limits of the joints
• a suitable initial solution to start the iterative optimization algorithm is chosen,
• the expression of the objective function (25) and of the kinematic constraints are input to the optimization algorithm,
• the solution of the optimization problem is then obtained using Sequential Quadratic Programming techniques (for instance, the \texttt{fmincon} function of MatLab\textsuperscript{TM}).

As for all iterative routines, a crucial point is the choice of a suitable initial solution, because a wrong choice would affect the execution time and even the final result of the algorithm.

![Fig. 1. Planned trajectory for a six joints robot.](image)
A suitable initial solution can be found by considering the lower bound of the intervals \( h_i \) making a time scaling of the trajectory. Let \( \mathbf{H}_{lb} \) be the vector of the lower bound of the \( vp + I \) optimization variables \( h_i \) computed using (24). By substitution of \( \mathbf{H}_{lb} \) into (17) the \( N \) joint trajectories corresponding to \( \mathbf{H}_{lb} \) can be computed. If a scaled time variable \( \tau \) is defined as: \( \tau = \Delta t \); the value of \( \Delta \) can be set from the values of the control points corresponding to the curves of velocity, acceleration and jerk. Namely, if a set of coefficient \( \{ A_1, A_2, A_3 \} \) is defined as

Fig. 2. Velocity of the six joints for the planned trajectory.
\[ A_1 = \max_j \left\{ \max_k \left\{ \left| \frac{\text{CPV}_{jk}}{\text{VC}_j} \right| \right\}, \ j = 1, \ldots, N, k = 1, \ldots, n \right\} \]
\[ A_2 = \max_j \left\{ \max_k \left\{ \left| \frac{\text{CPA}_{jk}}{\text{WC}_j} \right| \right\}, \ j = 1, \ldots, N, k = 1, \ldots, n - 1 \right\} \]
\[ A_3 = \max_j \left\{ \max_k \left\{ \left| \frac{\text{CPJ}_{jk}}{\text{JC}_j} \right| \right\}, \ j = 1, \ldots, N, k = 1, \ldots, n - 2 \right\} \]

Fig. 3. Acceleration of the six joints for the planned trajectory.
the scaling factor \( A \) can then be obtained by setting

\[
A = \max \left\{ 1, A_1, A_1^{1/2}, A_1^{1/3} \right\}
\]

so that a suitable initial solution is

\[
H_0 = A H_{lb}
\]

Fig. 4. Jerk of the six joints for the planned trajectory.
5. Results of the algorithm and comparison

The algorithm described in this paper has been tested in simulation for a 6-joint robot, and a comparison with the results obtained from the algorithm proposed by those Authors has been made.

The input data are reported in the following tables. Table 3 contains the values of the via-point of the trajectory, while Table 4 contains the values of the kinematic limits of the joints.

As already stated in the foregoing, the technique proposed by Simon and Isik [22] takes the total execution time as given, while the algorithm described in this paper outputs the total execution time as a result; its value depending on the weights $k_T$ and $k_J$ appearing in the expression of the cost function (1). So, in order to be able to compare the results yielded by the two algorithms, the values of the weights $k_T$ and $k_J$ have been adjusted so that the execution time of the two algorithms would be the same (namely, 9.1 s).

The results of the simulations are reported in Figs. 1–4, showing the trajectories of the six joints and their derivatives (velocity, acceleration and jerk, respectively).

Table 5 reports the maximum kinematic values resulting from the optimization procedure; such values are compared with those yielded by the technique proposed by Simon and Isik [22]. It can be noticed that the results yielded by the technique described in this paper are well comparable with those provided by the technique [22] with respect to the maximum values of acceleration and jerk.

Table 6 reports the mean values of velocity, acceleration and jerk for all joints, compared again with those found in [22]. In this case, the technique presented in this work yields lower mean values for acceleration and jerk of almost all the robot joints.

The results reported above show the effectiveness of the algorithm proposed in performing an optimization of the trajectories. The comparison with one of the best trajectory planning algorithm found in the scientific literature proves that the kinematic values of the generated trajectories are good and kept under control.

### Table 5
Maximum kinematic values resulting from the optimization procedure

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Joint</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gasparetto–Zanotto</td>
<td>$V_{\text{max}}$</td>
<td>37.57</td>
<td>41.96</td>
<td>61.49</td>
<td>28.90</td>
<td>41.53</td>
<td>38.14</td>
</tr>
<tr>
<td></td>
<td>$A_{\text{max}}$</td>
<td>39.07</td>
<td>43.65</td>
<td>65.60</td>
<td>15.87</td>
<td>33.88</td>
<td>39.94</td>
</tr>
<tr>
<td></td>
<td>$J_{\text{max}}$</td>
<td>54.94</td>
<td>65.90</td>
<td>78.13</td>
<td>23.02</td>
<td>52.73</td>
<td>65.15</td>
</tr>
<tr>
<td>Simon–Isik [22]</td>
<td>$V_{\text{max}}$</td>
<td>39.84</td>
<td>47.67</td>
<td>57.54</td>
<td>28.67</td>
<td>44.29</td>
<td>43.00</td>
</tr>
<tr>
<td></td>
<td>$A_{\text{max}}$</td>
<td>39.03</td>
<td>46.40</td>
<td>62.05</td>
<td>19.89</td>
<td>35.61</td>
<td>43.73</td>
</tr>
<tr>
<td></td>
<td>$J_{\text{max}}$</td>
<td>54.82</td>
<td>59.28</td>
<td>80.84</td>
<td>25.99</td>
<td>49.04</td>
<td>61.47</td>
</tr>
</tbody>
</table>

### Table 6
Mean kinematic values resulting from the optimization procedure

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Joint</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gasparetto–Zanotto</td>
<td>$V_{\text{avg}}$</td>
<td>16.94</td>
<td>20.70</td>
<td>27.45</td>
<td>15.38</td>
<td>15.92</td>
<td>19.47</td>
</tr>
<tr>
<td></td>
<td>$A_{\text{med}}$</td>
<td>19.08</td>
<td>18.15</td>
<td>30.60</td>
<td>6.50</td>
<td>14.92</td>
<td>21.84</td>
</tr>
<tr>
<td></td>
<td>$J_{\text{med}}$</td>
<td>26.26</td>
<td>20.69</td>
<td>41.26</td>
<td>6.39</td>
<td>18.18</td>
<td>30.35</td>
</tr>
<tr>
<td></td>
<td>$A_{\text{med}}$</td>
<td>19.48</td>
<td>19.99</td>
<td>30.07</td>
<td>6.53</td>
<td>15.02</td>
<td>23.52</td>
</tr>
<tr>
<td></td>
<td>$J_{\text{med}}$</td>
<td>27.51</td>
<td>23.97</td>
<td>41.27</td>
<td>9.10</td>
<td>18.07</td>
<td>33.77</td>
</tr>
</tbody>
</table>
6. Conclusions

A new methodology for optimal trajectory planning of robotic manipulators has been described in this paper. The technique is based on minimization of an objective function that takes into account both the execution time and the integral of the squared jerk along the whole trajectory. Unlike many other trajectory planning techniques, the proposed method does not take the execution time as given a priori and takes into account kinematic constraints on the robot motion, expressed as upper bounds on the absolute values of velocity, acceleration, and jerk for all robot joints.

The algorithm has been applied considering to compose the trajectory by means of fifth-order B-splines connecting pairs of consecutive via-points. The expressions for the objective function and the kinematic constraints have been formulated in this case.

Finally, the algorithm has been run in simulation, taking as input data those found in the work by Simon and Isik [22]. Comparison of the results with those provided in [22] has shown that the effectiveness of the algorithm is effective in performing an optimal trajectory planning.

Future work will be devoted to apply the present technique to trajectories of different kind (like cubic splines, trigonometric splines, etc.), so as to evaluate the applicability the algorithm and its results.

Acknowledgements

This work was supported in part by the Italian Ministry of Instruction and University, in the framework of the PRIN 2004 project. The authors would like to thank Mr. Claudio Mantovani for his important contribution to this work.

References


