J Intell Robot Syst
DOI 10.1007/s10846-010-9533-5

Experimental Validation of Minimum Time-jerk Algorithms for Industrial Robots

Vanni Zanotto · Alessandro Gasparetto · Albano Lanzutti · Paolo Boscariol · Renato Vidoni

Received: 19 July 2010 / Accepted: 22 December 2010

Abstract In this paper, we present a minimum-time/jerk algorithm for trajectory planning and its experimental validation. The algorithm search for a trade-off between the need for a short execution time and the requirement of a sufficiently smooth trajectory, which is the well known necessary condition to limit the vibration during fast movements. The trade-off is achieved by adjusting the weight of two suitable functions, able to consider both the execution time and the squared-jerk integral along the whole trajectory. The main feature of this algorithm is its ability to smooth the trajectory's profile by adjusting the intervals between two consecutive via-points so that the overall time is minimally delayed. The practical importance of this technique lies in the fact that it can be implemented in any industrial manipulator without a hardware upgrade. The algorithm does not need for a dynamic model of the robot: only the mechanical constraints on the position, velocity and acceleration ranges have to be set a priori. The experimental proof is provided in this paper by comparing the results of the proposed algorithm with those obtained by adopting some classical algorithms.

Keywords Trajectory planning · Optimization · B-splines · Cubic splines · Vibration reduction

1 Introduction

The growth in the use of industrial robots is characterized by the fusion of a wide spectrum of techniques. Motion control is one the main targets in the development of this sector; mainly aimed at improving the robustness, the precision and the stability of robotic controllers. Moreover, the use of flexible mechanical structures with

V. Zanotto (✉) · A. Gasparetto · A. Lanzutti · P. Boscariol · R. Vidoni
Dipartimento di Ingegneria Elettrica, Gestionale e Meccanica,
Universita' di Udine, 33100 Udine, Italia
e-mail: vanni.zanotto@uniud.it

Published online: 20 January 2011

complex vibration modes leads the control system to take into account the dynamics of the manipulator, especially when the trajectory to be executed by the robot must be planned [1]. In robotic systems with high dynamic performance, the growing demand for greater productivity forces designers and manufacturers to reduce the inertia and the mass of the whole system. However, this reduction results in a loss of structural rigidity and an increase of flexibility, and it also affects the dynamic response of the system.

A possible "hardware" solution of this problem is to use materials and structures that damp vibrations: this option is rather expensive and not always fully satisfactory [1]. Two complementary "software" techniques are: a proper calibration of the control system and a dedicated action on the trajectory planning, in order to reduce the positioning errors [2, 3].

In robotics, trajectory planning is a fundamental problem. The problem can be defined in this way: find a temporal motion law along a given geometric path, in such a way that certain requirements on the trajectory properties are fulfilled. Planning trajectories which satisfy specific requirements concerning position, velocity, acceleration and jerk values is crucial to ensure optimal results from the viewpoint of motion performance, especially for high-speed operations required in many current applications.

Trajectory planning is devoted to generate the reference inputs for the control system of the manipulator, to execute the motion. The geometric path, the kinematic and dynamic constraints are the inputs of the planning algorithm, whereas the trajectory of the joints (or the end effector), expressed as a time sequence of position, velocity and acceleration values, is the output.

The required smoothness property of a trajectory implies that the trajectory generated by the planner must not excite the mechanical resonances of the manipulator. In order to satisfy this request, some of the trajectory derivatives must be continuous functions. In particular, it would be desirable to obtain trajectories with continuous joint accelerations, so that the absolute value of the derivative of the acceleration, namely the jerk, is bounded.

A smooth trajectory (i.e. a trajectory with a limited jerk value) allows to reduce the errors during the movement and the stress to the actuators and the manipulator's structure. Low-jerk trajectories can be executed more rapidly and accurately. Moreover, the jerk minimization allows to limit the excitation of resonance frequencies, since vibration induced by non-smooth trajectories can damage the robot actuators, and introduce large errors while the robot is performing tasks such as a trajectory tracking.

Trajectory planning techniques that can be found in the scientific literature aim at minimizing some objective functions, which usually are the execution time or the actuator effort or the jerk. The first proposed trajectory planning technique was the minimum-time algorithm [4–6], due to the need of increasing the productivity in the industrial sector. The main disadvantage of these algorithms, that introduce dynamic problems during the execution of the trajectory, is that the trajectories have discontinuous values of accelerations. In order to compensate this drawback the trajectories should be smoothed, for instance, by considering trajectories with continuous acceleration. Spline functions have been used to this end for both kinematic and dynamic trajectory planning. Algorithms that plan the robot trajectory by using energetic criteria have been also proposed in the scientific literature [7–10].

This kind of method, that minimizes the energy consumption instead of the execution time, provides several advantages. It yields a smooth trajectory which is easier to track and reduces the stress to the actuators and to the robot 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).

A third approach provides a jerk-optimal trajectory. The aim of this method is to minimize the value of the jerk during the execution, in this way reducing the path errors, the stress to the actuators and the structure of the manipulator. Some examples of this approach can be found in [11–16]. In [11] the solutions for two point-to-point minimum-jerk trajectories are analytically obtained; the Pontryagin principle is used to obtain minimum-jerk point-to-point trajectories through a minimax approach. In [12] trigonometric splines are used to interpolate the trajectory and provide the jerk continuity. In [13, 14] the so-called interval analysis is used in an algorithm that globally minimizes the maximum absolute value of the jerk along a trajectory whose execution time is set a priori. The trajectories are expressed by cubic splines and the intervals between the via-points are computed so that the lowest jerk peak is produced. In [15, 16] a time-jerk optimal trajectory planning algorithm is presented. This method is an extended version of the algorithm reported in [21], because into the optimization constraints, the joint variable limits and the instantaneous power consumption of the actuating motors are also considered.

In [17] a method is presented for the time optimal path-constrained motion subjected to velocity, acceleration and jerk constraints. The optimization problem is solved by using a hybrid optimization strategy, starting from the path description, the kinematic relations of the manipulator and the defined constraints.

Once the trajectory has been planned, it is necessary to verify through experimental tests its effective smoothness. The acceleration of the end effector (or those of a particular part of the robot, e.g. an arm) is a fundamental parameter that accounts for the vibration that affects the structure of the manipulator during the trajectory execution [2, 18, 19].

To the Authors' knowledge, most of the trajectory planning algorithms that can be found in the literature have no experimental validation. Some experimental results could be found in [2–4, 15, 16, 18, 19].

In this paper, a trajectory planning algorithm, whose theoretical definition has been given in [20, 21] is applied to a Cartesian manipulator, and the experimental results are evaluated.

This algorithm allows to define the constraints on the robot motion before its execution. The 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.

Moreover, unlike most of the jerk-minimization methods, this technique does not require an a priori setting of the total execution time. Important benefits of this type of limited-jerk trajectories are the reduced mechanical stress and the reduced vibration. In order to demonstrate these properties, experimental tests have been made by using an accelerometer with the aim to measure the acceleration induced by the vibration of the arms of the Cartesian manipulator during their motion.

The paper is organized as follows: the trajectory planning algorithm used in the tests is described in Section 2. Application of the technique by cubic splines or fifth-order B-splines is described in Section 3. Section 4 deals with the implementation of

the algorithm. The application of the algorithm and the discussion of the results are presented in Section 5.

2 The Minimum Time-jerk Trajectory Planning Algorithm

Jerk control is an indirect way to limit the variation rate of the joint torque, without any need to account for the dynamic model of the manipulator when the trajectory is being planned. The suggested algorithm (described with many details in [20, 21]) assumes that a geometric path is available, generated by an upper-level system.

The path consists of a sequence of via-points defined in the Operative Space (Cartesian Space) of the robot (i.e. a sequence of positions and orientations of the end-effector). As mentioned above, the execution time is not set a priori, and only the constraints on the upper bounds of velocity, acceleration and jerk are taken into account. The algorithm considers a hybrid objective function, made of two terms that have opposite effects: one being proportional to the execution time and the other to the integral of the squared jerk. By choosing the terms of the objective function, a suitable trade-off between the speed and the smoothness can be achieved. The optimal trajectory planning problem is defined as:

$$\begin{cases} \min\limits_{\mathbf{h}} \; k_T N \sum\limits_{i=1}^{v_p-1} h_i + k_J \sum\limits_{j=1}^{N} \int\limits_0^{t_f} \dddot{q}_j^{\,2}(t)\, dt \\ \text{subject to:} \\ V_{\min,j} \le \dot{q}_j(t) \le V_{\max,j} & j = 1 \ldots N \\ A_{\min,j} \le \ddot{q}_j(t) \le A_{\max,j} & j = 1 \ldots N \\ J_{\min,j} \le \dddot{q}_j(t) \le J_{\max,j} & j = 1 \ldots N \end{cases} \tag{1}$$

where:

$$\mathbf{h} = \left[h_1 \; h_2 \; \ldots \; h_{v_p-1} \right]^T$$

Table 1 Nomenclature

N	Number of robot joints
v_p	Number of via-points
h_i	Time interval between two via-points
$\dot{q}_j(t)$	Velocity of the jth joint
$\ddot{q}_j(t)$	Acceleration of the jth joint
$\dddot{q}_j(t)$	Jerk of the jth joint
k_T	Weight of the term proportional to the execution time
k_J	Weight of the term proportional to the jerk
t_f	Total execution time of the trajectory
V_j	Velocity limit for the jth joint (symmetrical)
A_j	Acceleration limit for the jth joint (symmetrical)
J_j	Jerk limit for the jth joint (symmetrical)

The solution of the system (1) provides a vector of time intervals $h_i = t_{i+1} - t_i$ between any pair of consecutive via-points that minimizes the objective function. Table 1 explains the meaning of the symbols appearing in Eq. 1.

In order to implement the algorithm, suitable primitives for building the trajectory must be chosen. Next section will consider cubic splines (SPL3J) and fifth-order B-splines (BSPL5J).

3 Definition of the Trajectory by Means of Cubic Splines and Fifth-order B-splines

We briefly explain here how the expression of the objective function (1) can be obtained by using cubic splines (SPL3J) $Q_{j,i}(t)$, i.e. the cubic polynomial for the j-th joint defined on the interval $I_i = [t_i, t_{i+1}]$.

A complete description of the procedure, shortened for brevity in this section, can be found in the papers [20, 21] mentioned above.

As stated above, the interval vector **h** is the output of the algorithm. If we assume that the constraints are constant and symmetric for velocities, accelerations and jerks, the explicit expression of the constraints are:

$$\begin{aligned} \left|\dot{Q}_{j,i}(t)\right| &\leq V_j \;\; \forall j = 1...N, \forall i = 1...(v_p - 1) \\ \left|\ddot{Q}_{j,i}(t)\right| &\leq A_j \;\; \forall j = 1...N, \forall i = 1...(v_p - 1) \\ \left|\dddot{Q}_{j,i}(t)\right| &\leq J_j \;\; \forall j = 1...N, \forall i = 1...(v_p - 1) \end{aligned} \tag{2}$$

where v_p is the number of the via-points of the path. The constraints hold for any value of the continuous variable t. It is convenient to transform them into finite constraints, which can be done in the following way.

First, the maximum velocity must be evaluated. For a cubic spline the velocity is a parabolic function in the interval I_i. Therefore, its maximum value must be either at one interval end, or at the time $t^*_{j,i}$ given by:

$$t^*_{j,i} = t_{j,i} + \frac{h_i \alpha_{j,i}}{\alpha_{j,i} - \alpha_{j,i+1}}$$

Consequently, the finite form of the velocity constraints becomes:

$$\max\left\{\left|\dot{Q}_{j,i}\left(t_{j,i}\right)\right|, \left|\dot{Q}_{j,i}\left(t_{j,i+1}\right)\right|, \left|\dot{Q}_{j,i}(t_{j,i}{}^*)\right|\right\} \leq V_j \tag{3}$$

$$\forall j = 1...N \; \forall i = 1...(v_p - 1)$$

where

$$\dot{Q}_{j,i}\left(t_{j,i}\right) = -\frac{\alpha_{j,i}}{2} h_i + \frac{q_{j,i+1} - q_{j,i}}{h_i} + \frac{\alpha_{j,i} - \alpha_{j,i+1}}{6} h_i$$

$$\dot{Q}_{j,i}\left(t_{j,i+1}\right) = \frac{\alpha_{j,i+1}}{2} h_i + \frac{q_{j,i+1} - q_{j,i}}{h_i} + \frac{\alpha_{j,i} - \alpha_{j,i+1}}{6} h_i$$

$$\dot{Q}_{j,i}\left(t_{j,i}{}^*\right) = -\frac{h_i}{2\left(\alpha_{j,i+1} - \alpha_{j,i}\right)} \alpha_{j,i}\alpha_{j,i+1} + \frac{q_{j,i+1} - q_{j,i}}{h_i} - \frac{h_i}{6}\left(\alpha_{j,i+1} - \alpha_{j,i}\right)$$

The same procedure can be followed for the constraints on acceleration and jerk. The acceleration is linear in the interval I_i, so that it must reach its maximum value at one of the interval ends. Therefore, the constraints can be written as follows:

$$\max\left\{\left|\alpha_{j,i}\right|, \left|\alpha_{j,i+1}\right|\right\} \leq A_j \ \forall j = 1...N \ \forall i = 1...(v_p - 1)$$

The jerk is constant in any interval and can be written as:

$$\dddot{Q}_{j,i}(t) = \frac{\alpha_{j,i+1} - \alpha_{j,i}}{h_i}$$

Therefore, the finite constraints are:

$$\left|\frac{\alpha_{j,i+1} - \alpha_{j,i}}{h_i}\right| \leq J_j \ \forall j = 1...N \ \forall i = 1...\left(v_p - 1\right)$$

There are additional constraints on the minimum duration of intervals I_i, because the following equation must hold:

$$\left|\frac{q_{j,i+1} - q_{j,i}}{h_i}\right| \leq V_j$$

As such, any interval cannot be smaller than:

$$h_i \geq h_{i,\min} = \max_{j=1...N}\left\{\frac{\left|q_{j,i+1} - q_{j,i}\right|}{V_j}\right\} \tag{4}$$

for all $j = 1...N$ and $i = 1...(v_p - 1)$.

Therefore, for trajectories made of cubic splines, the minimization problem (1) can be rewritten as:

$$\begin{cases} \min\limits_{\mathbf{h}}\left[k_T N \sum\limits_{i=1}^{v_p-1} h_i + k_J \sum\limits_{j=1}^{N}\sum\limits_{i=1}^{v_p-1}\left(\frac{\left(\alpha_{j,i+1} - \alpha_{j,i}\right)^2}{h_i}\right)\right] \\ \\ \text{subject to:} \\ \\ \max\left\{\left|\dot{Q}_{j,i}\left(t_{j,i}\right)\right|, \left|\dot{Q}_{j,i}\left(t_{j,i+1}\right)\right|, \left|\dot{Q}_{j,i}(t_{j,i}{}^*)\right|\right\} \leq V_j \\ \max\left\{\left|\alpha_{j,i}\right|, \left|\alpha_{j,i+1}\right|\right\} \leq A_j \\ \left|\frac{\alpha_{j,i+1} - \alpha_{j,i}}{h_i}\right| \leq J_j \\ h_i \geq \max\limits_{j=1...N}\left\{\frac{\left|q_{j,i+1} - q_{j,i}\right|}{V_j}\right\} \end{cases} \tag{5}$$

for all $j = 1...N$ and $i = 1...(v_p - 1)$.

By varying k_T and k_J the designer can choose between the shortest execution time and the smoothest trajectory. The limit conditions are:

1. $k_T = 0$

$$\min_{\mathbf{h}} \left(\sum_{j=1}^{N} \sum_{i=1}^{v_p - 1} \left(\frac{\left(\alpha_{j,i+1} - \alpha_{j,i}\right)^2}{h_i} \right) \right)$$

   As such, the minimization problem makes a minimum-jerk trajectory.

2. $k_J = 0$

$$\min_{\mathbf{h}} \left(\sum_{i=1}^{n-1} h_i \right)$$

   and the minimization problem makes a minimum-time trajectory.

Instead of using cubic splines, fifth-order B-splines (BSPL5J) can be used in Eq. 1. As it is well known, a B-spline of degree p and order $k = p + 1$ is a spline curve $B_p(t)$ expressed 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 (CPs).

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

The curve is built on a sequence of temporal instants t_i (named nodes) and the base functions are defined recursively by the De Boor formula [22]:

$$\begin{cases} N_{i,p}(t) = \begin{cases} 1 & t \in I_i = [t_i, t_{i+1}] \\ 0 & t \notin I_i \end{cases} & p = 0 \\ N_{i,p}\,(t) = \dfrac{t - t_i}{t_{i+p} - t_i} N_{i,p-1}\,(t) + + \dfrac{t_{i+p+1} - t}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}\,(t) & p > 0 \end{cases}$$

Table 2 Nomenclature for B-splines

Symbol	Definition
p	Degree of the B-*spline*
k	Order of the B-*spline*
$B_p(t)$	B-*spline* of degree p
$N_{i,p}(t)$	Base function of degree p
Q_i	Control point of the B-*spline*
n+1	Number of control points
t_i	Nodes
m+1	Number of nodes

The symbols that appear in the previous formulas are explained in Table 2. In this case the expressions of the constraints on velocity, acceleration and jerk (symmetric) are:

$$\begin{cases} \left|CPV_{j,k}\right| \leq V_j & k = 1...n \\ \left|CPA_{j,k}\right| \leq A_j & k = 1...(n-1) \\ \left|CPJ_{j,k}\right| \leq J_j & k = 1...(n-2) \end{cases}$$

where:

$$\begin{cases} CPV_{j,k} = \frac{p}{Uq_{j,k+p+1} - Uq_{j,k+1}} \left(CPQ_{j,k+1} - CPQ_{j,k}\right) \\ \quad for\ k = 1...n \\ \\ CPA_{j,k} = \frac{p-1}{Uv_{j,k+p} - Uv_{j,k+1}} \left(CPV_{j,k+1} - CPV_{j,k}\right) \\ \quad for\ k = 1...(n-1) \\ \\ CPJ_{j,k} = \frac{p-2}{Ua_{j,k+p-1} - Ua_{j,k+1}} \left(CPA_{j,k+1} - CPA_{j,k}\right) \\ \quad for\ k = 1...(n-2) \end{cases}$$

The symbols that appear in the previous formulas are explained in Table 3.

Another type of kinematic constraint is due to the fact that the optimization parameters h_is are lower bounded, similar to the one that is represented in Eq. 4. The expression of the objective function (1) for a trajectory made of fifth-order B-splines is thus given by:

$$\min_{\mathbf{h}} k_T N \sum_{i=1}^{v_p+1} h_i + k_J \sum_{j=1}^{N} \int_0^{t_f} \left(\sum_{k=1}^{n-2} CPJ_{j,k} \cdot N_{p-3,k}(t) \right)^2 dt$$

The same remarks of Eq. 5 about the values of k_T and k_J hold here, as well. The reader can refer to [20, 21] to find more details about the algorithm.

Table 3 Nomenclature for the BSPL5J algorithm

Symbol	Definition
CPQ	Control point of the trajectory
CPV	Control point of the velocity
CPA	Control point of the acceleration
CPJ	Control point of the jerk
Uq	Node of position
Uv	Node of velocity
Ua	Node of acceleration

4 Execution of the Algorithms and Choice of k_T and k_J

The steps for running the algorithms SPL3J and BSPL5J can be summarized as follows:

1. obtain a sequence of via-points in the joint space by applying a kinematic inversion to the path defined in the operating space;
2. set the speed, acceleration and jerk bounds at the joints by considering the manipulator structural constraints;
3. set k_T and k_J;
4. choose the initial solution of the iterative optimization process by considering the lowest values of the time intervals, h_i (given by Eq. 4);
5. run the algorithm by solving the optimization problem by means of a suitable optimization routine. The outputs of the algorithm is the vector of optimal time intervals $\mathbf{h}$.

An important point is to make a suitable choice for the coefficients k_T and k_J, whose aim is to weight the two opposite effects, so as to reach a trade-off between speed and smoothness. To this end, a criterion to make a suitable choice of the two weights has been designed and implemented.

For the SPL3J algorithm, the objective function can be written as follows:

$$\min_{\mathbf{h}} \sum_{j=1}^{N} \sum_{i=1}^{v_p-1} \left(k_T h_i + k_J \frac{\left(\alpha_{j,i+1} - \alpha_{j,i}\right)^2}{h_i} \right)$$

It can be noticed that the jerk is piecewise constant since in the ith interval I_i the difference $(\alpha_{j,i+1} - \alpha_{j,i})$ is constant. Therefore, for each interval I_i the value of k_J/k_T sets the ratio between the squared-jerk contribution and the time contribution. The above ratio has to be chosen according to the requirements set on the speed and the smoothness of the trajectory.

The procedure is made of the following steps:

1. calculate the maximum jerk, by setting $k_J = 0$ and solving the optimization problem;
2. set the new value of jerk, according to the requirements on the resulting trajectory;
3. the ratio is computed from the ratio between the two values of jerk defined at steps 2 and 1, respectively;
4. k_T and k_J can be chosen arbitrarily, provided that their ratio is equal to the value computed at step 3.

A procedure for the choice of k_T and k_J has been defined and implemented for the BSPL5J trajectory planning technique, as well. However, the expression of the objective function does not allow to obtain an explicit relation between the jerk term and the time term, as for the SPL3J algorithm. Therefore, an iterative procedure has been set up, based on the following steps:

1. set k_J to a fixed value;
2. run the BSPL5J algorithm by using different values for k_T (e.g starting from 0 and increasing it with discrete steps);

3. plot the values of the integral of the squared jerk versus the execution time and the weights;
4. choose the most suitable value for k_T by inspection of the plot.

5 Experimental Setup

The algorithms have been tested on a Cartesian 3-axes manipulator (Fig. 1), controlled by using a real time external device, based on the xPC-Target(TM) software.

All the manipulator's kinematic constraints are reported in Table 4.

Each axis is actuated by a brush-less AC servo-motor, Siemens 1FK6 (11.4 Nm maximum torque), coupled with the load by using a cogged belt and a geared transmission.

The link between the real-time controller and the robot is obtained through a Sensoray S626 multifunction PCI board.

The Cartesian manipulator is equipped with the Siemens Simodrive 611 industrial control unit. This is a control board that offers conventional drive functionality, such as torque and speed control, and positioning functionality.

The control scheme of each axis is shown in Fig. 2.

The real-time position controller has been implemented on an AMD Athlon(tm) XP 2400 (1.99 GHz) with 480 MB of RAM memory. The xPC-Target software provides a high-performance host-to-target environment. It makes the connection between the host PC (i.e. the user-interface device that allows to manage the basic controller functions, as the start and stop commands, and store the acquired data) and the dedicated target PC where the real time kernel runs.

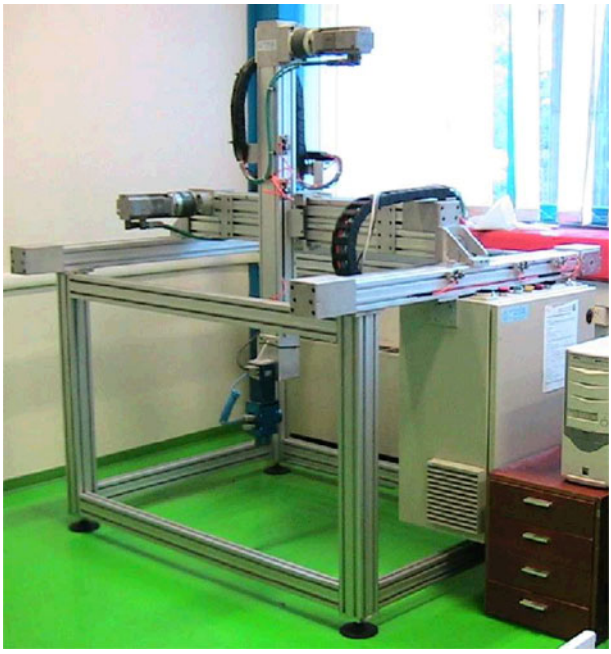

Fig. 1 Cartesian three-axes manipulator

Table 4 Kinematic constraints

Joint	Velocity	Acceleration	Jerk
1	225 mm/s	700 mm/s^2	2,400 mm/s^3
2	225 mm/s	700 mm/s^2	2,400 mm/s^3
3	225 mm/s	700 mm/s^2	2,400 mm/s^3

The target of the tests concerns the experimental validation of the BSPL5J and SPL3J algorithms by evaluating the effectiveness of the proposed algorithms in reducing the robot vibrations, with respect to a "classical" approach, in this case a spline trajectory planning algorithm (named PROP in the following sections and described in major detail in Section 4.2).

In order to accomplish this target, a piezoelectric accelerometer mounted on the Z axis robot arm has been used. The device's accuracy is 1036 mV/g and the maximum value of measurable acceleration is ±5g.

5.1 Control Scheme

The single axis control scheme is shown in Fig. 2. The Siemens Simodrive control unit provides the internal loops for the velocity and the torque by means two simple PI controllers. The external real-time controller makes closed the position loop through the position gain k_v and provides the feed-forward action by the gain k_{fv}, as shown in Fig. 2. This control scheme represents the general model for an industrial axis drive control [2].

Each shape of the trajectory takes effect both the controller's and the axis's vibration, as it has been discussed in [2]. These effects change with the mechanical properties of the axis, the position and feed-forward gains and, clearly, with the shape and the amplitude of the jerk. Moreover, the controller action can excite or filter the vibration, independently of the planned trajectory. Once the trajectory has been planned, the operator can adjust the controller action in order to obtain the suitable compromise between the trajectory tracking and the induced vibration.

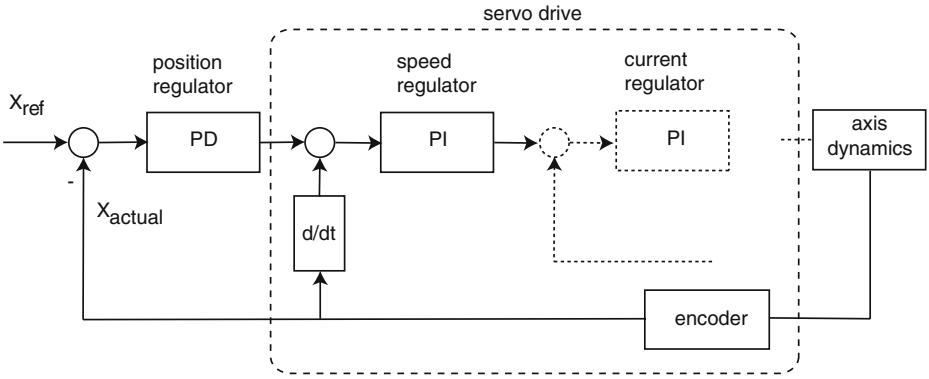

Fig. 2 Control scheme

Therefore, a strict comparison among several types of trajectories is hard to be done, because a proper tuning of the controller can mask the effects of the chosen trajectory.

In this work, we try to make such a comparison by keeping unchanged the controller's set up and looking for the different effects that each type of trajectory planner (PROP, SPL3J or BSPL5J) generates on the vibration. In order to make "unpolarized" this test, we optimize the control tuning only for the classical cubic planner, named PROP, by following the procedure presented in [2]. Eventually, the same controller is used for the trajectories planned by the SPL3J and BSPL5J algorithms. This procedure is repeated for each of the tasks under consideration.

5.2 Test Trajectories

In order to lead the test mentioned above, three different tasks (namely, Tk1, Tk2 and Tk3) have been simulated in MatLab ™ and then implemented in the real manipulator.

The first task (named Tk1) consists on moving the end-effector along a circle of radius $r = 150$ mm, centered at $C = [-500, -300, -600]^T$(mm) and tilted, with respect to the robot's reference frame, according to the following rotation matrix:

$$R = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{\sqrt{2}} \\ -\frac{1}{2} & -\frac{1}{2} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

Table 5 Via-points for Tk2

Via-point	X [mm]	Y [mm]	Z [mm]
1	0	0	0
2	Virtual		
3	100	100	−50
4	0	200	−100
5	−100	300	−180
6	0	400	−200
7	130	300	−120
8	0	220	−100
9	−50	70	−80
10	Virtual		
11	0	0	0

Table 6 Via-points for Tk3

Via-point	X [mm]	Y [mm]	Z [mm]
1	0	0	0
2	Virtual		
3	0	0	−170
4	10	12.9	−190
5	30	38.6	−200
6	175	225	−200
7	320	411.4	−200
8	330	437.1	−190
9	350	450	−170
10	Virtual		
11	350	450	0

The execution time of this task is 5.53 s. For the SPL3J algorithm k_T and k_J are set to 1,800 and 0.005, respectively, while for the BSPL5J algorithm they are set to 21 and 1. This choice allows the execution time to be the same for all the three algorithms.

The second task (Tk2) makes the end-effector describe an "eight"-shaped curve, with unevenly spaced via-points. The execution time is 8 s and the values of k_T and k_J are set to 775 and 0.002 for the SPL3J and to 10 and 1 for the BSPL5J, respectively.

The last task (Tk3) is a typical industrial application of pick-and-place, with an execution time of 7.4 s. For this trajectory, k_T and k_J are set to 860 and 0.005 for the SPL3J, and 10 and 1, respectively, for the BSPL5J.

Tables 5 and 6 contain the via-points of the last two trajectories.

The limit values of time, t_{MOV}, and the integral of the squared jerk, I_{MOV}, as they have been defined in Section 3.2, are reported in Tables 7 and 8.

As it has been stated in the previous section, the controller is tuned only for the trajectory planned with the PROP algorithm. The effectiveness of the controller is shown in Fig. 3.

This figure shows the simulated versus the measured positions of the X, Y and Z joints for task Tk3 implemented by using the PROP algorithm.

Table 7 Limit values of time and integral of the squared jerk for the SPL3J planner

	$k_T = 0$ $k_J = 1$		$k_T = 1$ $k_J = 0$		Test conditions	
	t_{MOV} [s]	I_{MOV} [mm^2/s^5]	t_{MOV} [s]	I_{MOV} [mm^2/s^5]	t [s]	I [mm^2/s^5]
Tk1	89.95	2.44	4.25	1.67×10^7	5.53	2.25×10^6
Tk2	99.49	22.54	6.14	2.07×10^7	8	3.72×10^6
Tk3	32.34	5.14×10^4	4.89	1.02×10^7	7.4	4.82×10^5

Table 8 Limit values of time and integral of the squared jerk for the BSPL5J planner

	$k_T = 0\ k_J = 1$		$k_T = 1\ k_J = 0$		Test conditions	
	t_{MOV} [s]	I_{MOV} [mm^2/s^5]	t_{MOV} [s]	I_{MOV} [mm^2/s^5]	t [s]	I [mm^2/s^5]
Tk1	311.93	0.0175	5.09	3.86×10^6	5.53	2.13×10^6
Tk2	387.21	0.0416	7.58	5.51×10^6	8	3.28×10^6
Tk3	249.58	3.73	5.66	3.10×10^6	7.4	4.64×10^5

5.3 Comparison of BSPL5J and SPL3J with a Classic Spline Algorithm

With the aim of evaluating the trajectories based on the SPL3J and BSPL5J algorithms, a "classic" spline algorithm has been implemented, based on the cubic spline objective function, with imposed time intervals (named PROP). The duration of the time intervals is proportional to the number of via-points and the execution time of the trajectory. In Table 9 the main properties of the three techniques can be shown.

Tables 10, 11, and 12 list the time intervals used for the three trajectories.

It must be pointed out that the PROP algorithm considers an execution time set a priori and does not accept any kinematic constraint. The SPL3J and the BSPL5J algorithms, on the other hand, do not require the execution time to be imposed and the kinematic constraints are taken into account when the optimal trajectory is planned.

The main target of the tests is to compare the jerk values and the vibration effects that are obtained by using the three different primitive (PROP, SPL3J and BSPL5J).

In order to obtain the same execution time for all the trajectories generated by the different algorithms, the SPL3J and the BSPL5J are simulated first of all. Therefore, the resulting time is imposed to the PROP technique.

This part of the tests consisted on an evaluation of the smoothness of the trajectories that have been planned with the three algorithms. Evaluation of the results obtained by running the algorithms SPL3J, BSPL5J and PROP shows the

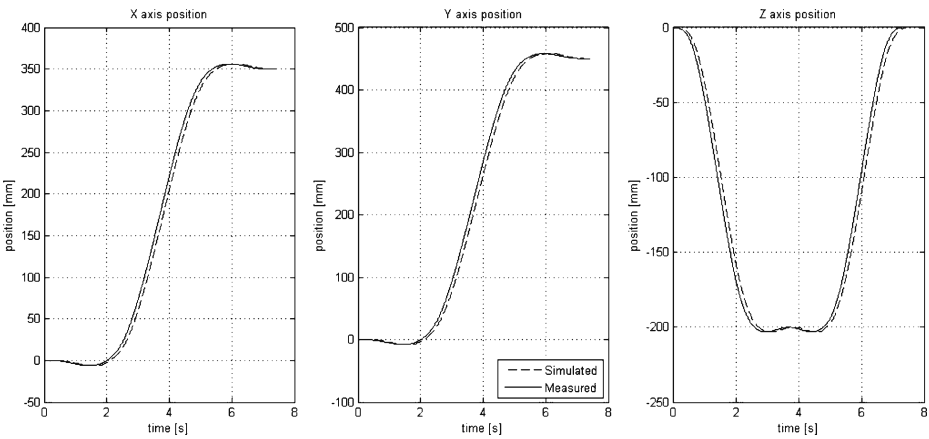

Fig. 3 Tk3: joints positions using the PROP algorithm (simulation vs experimental)

Table 9 Main properties of the planners

Algorithm	Primitive	Trajectory time length	Optimization	Notes
PROP	Cubic splines	Imposed	Jerk	
SPL3J	Cubic splines	Calculated	Jerk-time	Kinematic constraints
BSPL5J	Quintic B-splines	Calculated	Jerk-time	Kinematic constraints

Table 10 Time intervals (in *s*) for Tk1

	h1	h2	h3	h4	h5	h6	h7	h8	h9
SPL3J	0.3734	0.8611	0.6386	0.5854	0.5844	0.6019	0.6456	0.8691	0.3705
BSPL5J	0.0678	1.0930	0.7301	0.6244	0.6280	0.6013	0.6171	1.0703	0.0980
PROP	0.6144	0.6144	0.6144	0.6144	0.6144	0.6144	0.6144	0.6144	0.6144

Table 11 Time intervals (in *s*) for Tk2

	h1	h2	h3	h4	h5	h6	h7	h8	h9	h10
SPL3J	0.3634	1.1726	0.7109	0.6927	0.9571	1.1737	0.8368	0.8840	0.8561	0.3527
BSPL5J	0.4641	1.0957	0.6910	0.6645	0.9573	1.1182	0.8724	0.8575	1.0109	0.2684
PROP	0.8000	0.8000	0.8000	0.8000	0.8000	0.8000	0.8000	0.8000	0.8000	0.8000

Table 12 Time intervals (in *s*) for Tk3

	h1	h2	h3	h4	h5	h6	h7	h8	h9	h10
SPL3J	0.5766	1.4440	0.2822	0.3019	1.0953	1.0953	0.3019	0.2822	1.4440	0.5766
BSPL5J	0.5601	1.4829	0.2676	0.2863	1.0323	1.0312	0.2854	0.2664	1.1026	1.0852
PROP	0.7400	0.7400	0.7400	0.7400	0.7400	0.7400	0.7400	0.7400	0.7400	0.7400

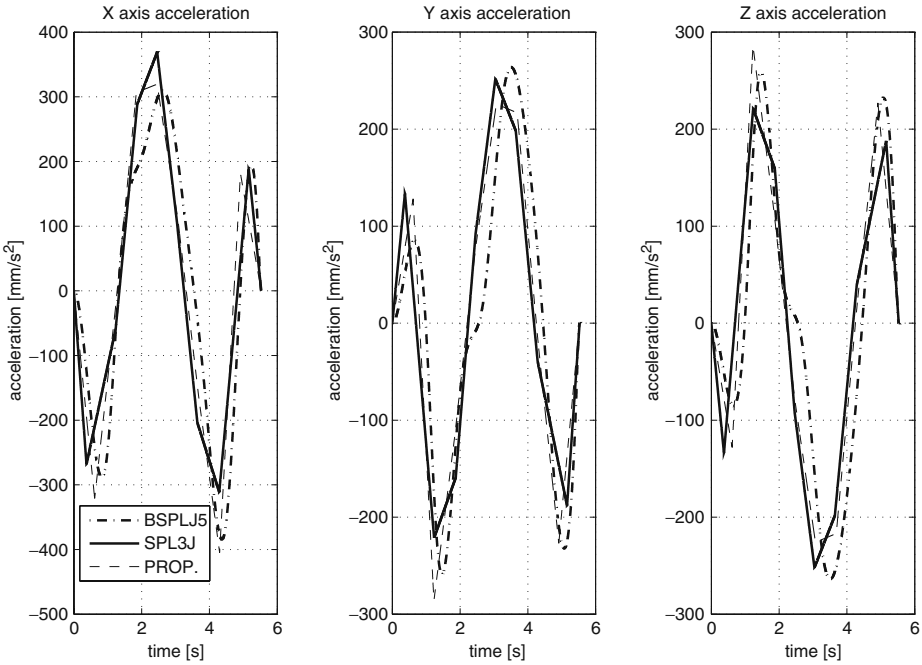

Fig. 4 Tk1: joints accelerations (BSPL5J vs SPL3J vs PROP)

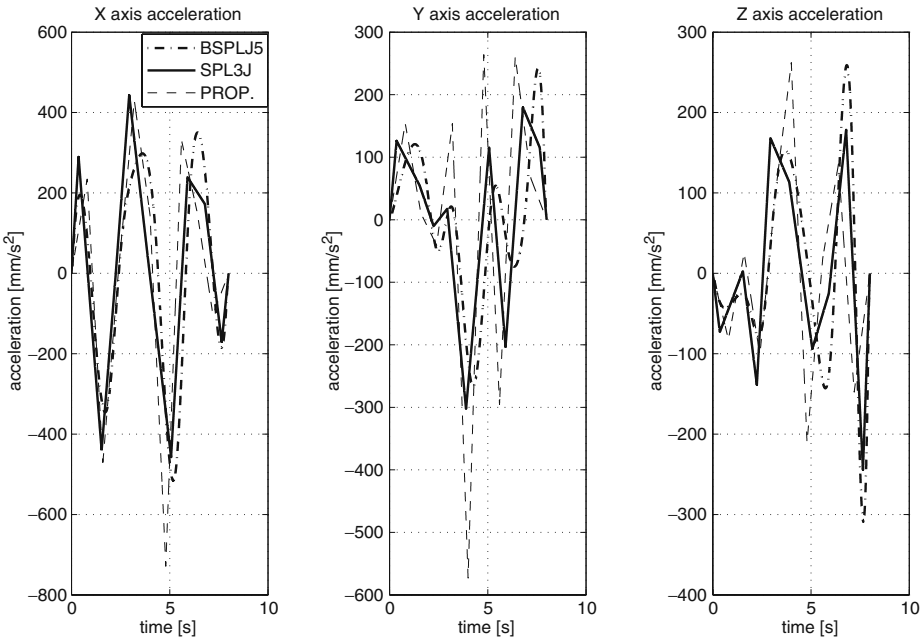

Fig. 5 Tk2: joints accelerations (BSPL5J vs SPL3J vs PROP)

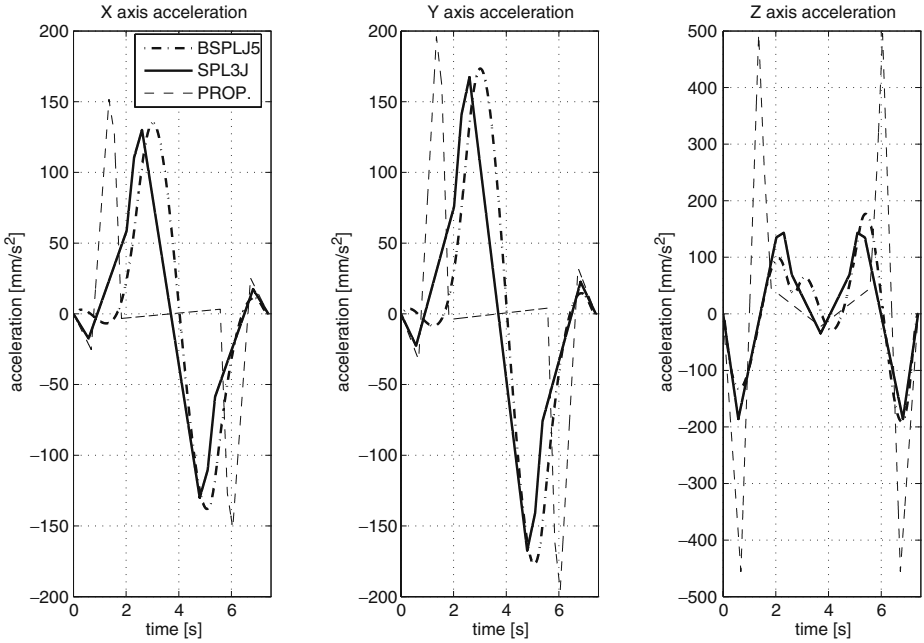

Fig. 6 Tk3: joints accelerations (BSPL5J vs SPL3J vs PROP)

improvements that could be obtained by using the technique presented in this work. Compared with a classic spline trajectory (PROP), the minimum time-jerk technique features a smoother profile. Indeed, both the SPL3J and BSPL5J algorithms feature lower acceleration absolute values in comparison with the ones of the PROP algorithm. Moreover, the mean value of the acceleration with the BSPL5J planner is comparable with the mean value of the acceleration with the SLP3J planner, although the BSPL5J technique produces a larger maximum absolute value.

This can be seen in Figs. 4, 5, and 6 where the joints accelerations for the three trajectories (Tk1, TkJ, and Tk3), generated by using the three techniques (SPL3J, BSPL5J and PROP) are reported. In particular, the plots of the acceleration of the Y joint in Fig. 5 and of the Z joint in Fig. 6 show the effectiveness of the proposed SPL3J and BSPL5J algorithms since they reduce the acceleration.

Tables 13, 14, and 15 show the *rms* and the *mean* value of the accelerations measured by means of the accelerometer mounted on the Z axis, for the three trajectories (Tk1, Tk2, Tk3), generated by using the three techniques (SPL3J,

Table 13 Comparison of *rms* and *mean* values of acceleration induced by vibration for Tk1

	rms value [m/s^2]	% improvement	Mean value [m/s^2]	% improvement
PROP	0.6352	–	0.4402	–
SPL3J	0.6087	4.17	0.4108	6.68
BSPL5J	0.6219	2.09	0.4156	5.59

Table 14 Comparison of *rms* and *mean* values of acceleration induced by vibration for Tk2

	rms value [m/s^2]	% improvement	Mean value [m/s^2]	% improvement
PROP	0.6096	–	0.4260	–
SPL3J	0.5424	11.02	0.3938	7.56
BSPL5J	0.5443	10.71	0.3919	8.00

Table 15 Comparison of *rms* and *mean* values of acceleration induced by vibration for Tk3

	rms value [m/s^2]	% improvement	Mean value [m/s^2]	% improvement
PROP	0.6797	–	0.3900	–
SPL3J	0.4742	30.23	0.3313	15.05
BSPL5J	0.5074	25.35	0.3360	13.85

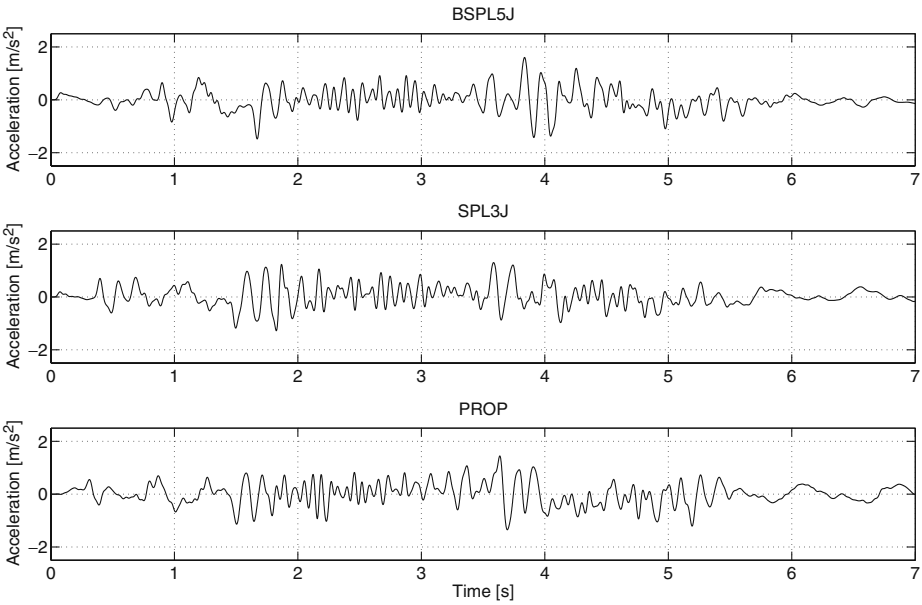

Fig. 7 Measured accelerations for Tk1

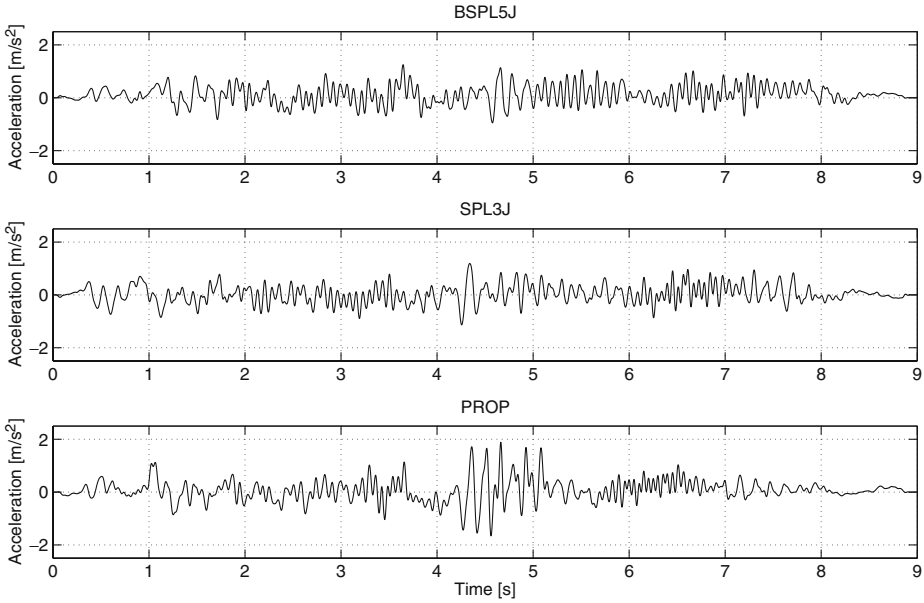

Fig. 8 Measured accelerations for Tk2

BSPL5J and PROP). Taking the *rms* values generated by the PROP algorithm as a reference, it can be noticed that an improvement of 4 or 2% for *Tk1*, of 11% for *Tk2* and of 30 or 25% for *Tk3* is obtained if SPL3J or BSPL5J are implemented. In

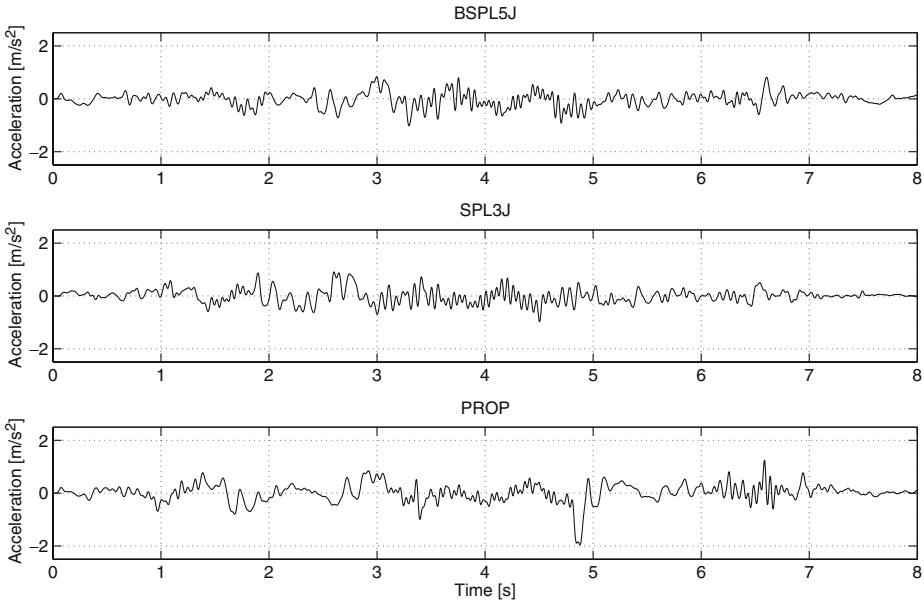

Fig. 9 Measured accelerations for Tk3

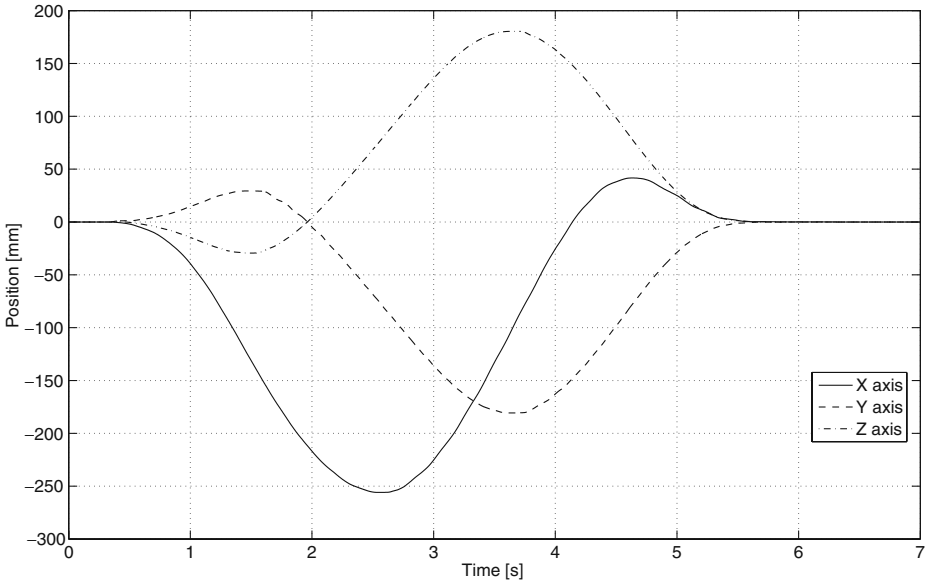

Fig. 10 Measured axes position for Tk1

Figs. 7, 8, and 9 the accelerations measured by means the accelerometer applied to the manipulator Z axis are shown.

If the graphs of the measured accelerations (Figs. 7–9) are examined together with the axes position plots (Figs. 10, 11, and 12), it is possible to notice that the

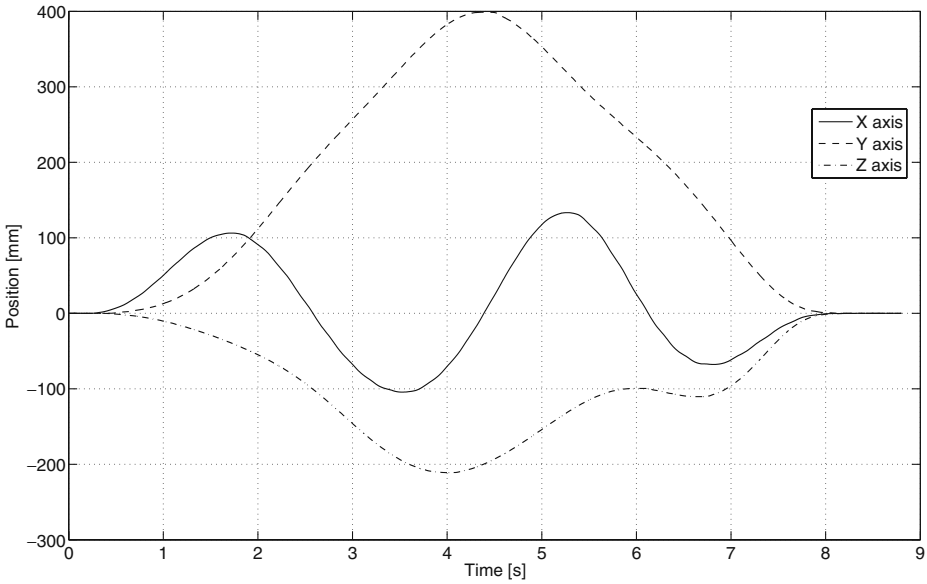

Fig. 11 Measured axes position for Tk2

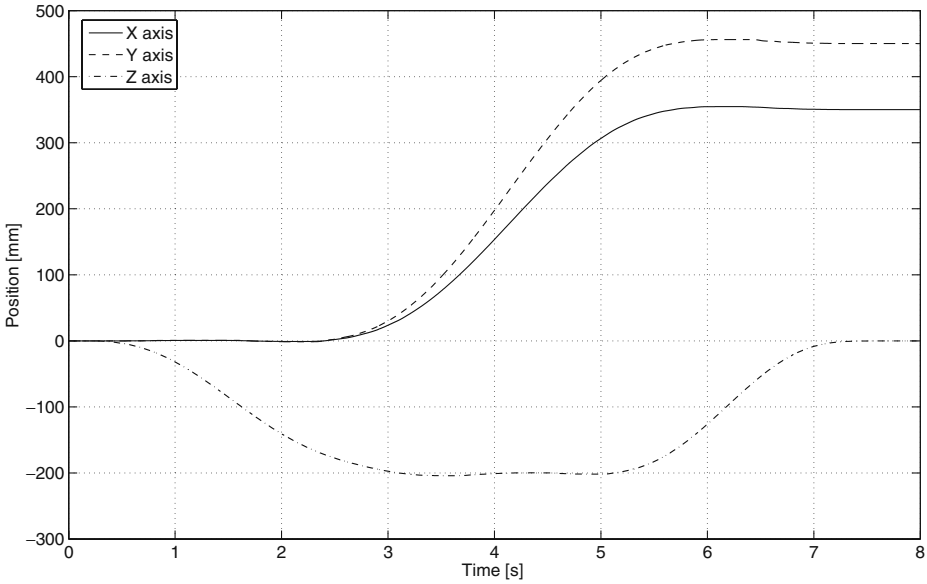

Fig. 12 Measured axes position for Tk3

increase of the oscillations amplitude (in the accelerations graphs) corresponds to the change-of-direction of the Y axis movement (the broken line in the position figures). This can be clearly observed for *Tk1* (approximately at time 1.5 s and 4 s) and *Tk2* (approximately at time 4.5 s), for which the Y axis inverts its movement, and it is instead less marked for *Tk3* (approximately at time 3 s and 5 s), for which the Y axis only "starts-and-stops" its movement. Finally, as a confirmation of the data reported in Tables 13–15, by considering the test examples reported in Figs. 7–9 and by taking the PROP case as a reference behavior (0.4133 m/s^2, 0.4177 m/s^2 and 0.3406 m/s^2 for trajectories Tk1, Tk2 and Tk3 respectively), the acceleration *rms* values show an improvement of 6.56%, 18.73% and 24.18 % if the case SPL3J is considered (0.3862 m/s^2, 0.3395 m/s^2 and 0.2583 m/s^2 for trajectories Tk1, Tk2 and Tk3 respectively) and of 3.89%, 13.58% and 22.82% if the case BSPL5J is evaluated (0.3972 m/s^2, 0.3610 m/s^2 and 0.2629 m/s^2 for trajectories Tk1, Tk2 and Tk3 respectively).

By considering the results mentioned above, an experimental validation of the actual effectiveness of the proposed trajectory planning algorithm has been made. The experimental tests prove that the minimum time-jerk trajectory planning technique allows to decrease the absolute acceleration values of the robot joints, and to obtain a reduction in the values of the acceleration due to the robot vibration, i.e. the reduction of the vibration of the robot arms during the trajectory execution.

6 Conclusion

In the present paper an experimental validation of the minimum time-jerk trajectory planning algorithm has been described. The algorithm takes into account both the execution time and the integral of the squared jerk along the whole trajectory and

allows to choose between the need for a quick execution or the need for a smooth trajectory by varying the values of two weights. It is not required to set the execution time a priori. The trajectories have been implemented on a Cartesian manipulator, and the results obtained have been evaluated and discussed in comparison with a "classic" spline algorithm. The performance has been evaluated by considering the acceleration of the Z joint of the robot measured by means of an accelerometer. Future work will be devoted to work with the DoFs of the robot wrist, as well as to compare the results also with other trajectory planning techniques found in the literature.

References

1. Benning, R.D., Hodgins, M.G., Zipfel, G.G.: 1997, Active control of mechanical vibrations. Bell Labs Tech. J. **2**(2), 246–257 (1997)
2. Barre, P.J., Bearee, R., Borne, P., Dumetz, E.: Influence of a jerk controlled movement law on the vibratory behaviour of high-dynamics systems. J. Intell. Robot. Syst. **42**(3), 275–293 (2005)
3. Bearee, R., Barre, P.J., Bloch, S.: Influence of high-speed machine tool control parameters on the contouring accuracy. Application to linear and circular interpolation. J. Intell. Robot. Syst. **40**(3), 321–342 (2004)
4. Constantinescu, D., Croft, E.A.: Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. J. Robot. Syst. **17**(5), 233–249 (2000)
5. Lin, C.S., Chang, P.R., Luh, J.Y.S.: Formulation and optimization of cubic polynomial joint trajectories for industrial robots. IEEE Trans. Automat. Contr. **28**(12), 1066–1073 (1983)
6. Piazzi, A., Visioli, A.: Global minimum-time trajectory planning of mechanical manipulators using interval analysis. Int. J. Contr. **71**(4), 631–652 (1998)
7. Von Stryk, O., Schlemmer, M.: Optimal control of the industrial robot Manutec r3. In: Bulirsch, R., Kraft, D. (eds.) Computational Optimal Control. International Series of Numerical Mathematics, vol. 115, pp. 367–382 (1994)
8. Field, G., Stepanenko, Y.: Iterative dynamic programming: an approach to minimum energy trajectory planning for robotic manipulators. In: Proc. of the IEEE International Conference on Robotics and Automation, vol. 3, pp. 2755–2760 (1996)
9. Saramago, S.F.P., Steffen Jr., V.: Optimization of the trajectory planning of robot manipulators tacking into account the dynamics of the system. Mech. Mach. Theory **33**(7), 883–894 (1998)
10. Saramago, S.F.P., Steffen Jr., V.: Optimal trajectory planning of robot manipulators in the presence of moving obstacles. Mech. Mach. Theory **35**(8), 1079–1094 (2000)
11. Kyriakopoulos, K.J., Saridis, G.N.: Minimum jerk path generation. In: Proc. of the 1988 IEEE International Conference on Robotics and Automation, vol. 1, pp. 364–369 (1988)
12. Simon, D., Isik, C.: A trigonometric trajectory generator for robotic arms. Int. J. Contr. **57**(3), 505–517 (1993)
13. Piazzi, A., Visioli, A.: Global minimum-jerk trajectory planning of robot manipulators. IEEE Trans. Ind. Electron. **47**(1), 140–149 (2000)
14. Piazzi, A., Visioli, A.: An interval algorithm for minimum-jerk trajectory planning of robot manipulators. In: Proc. of the 36th Conference on Decision and Control, vol. 2, pp. 1924–1927 (1997)
15. Lombai, F., Szederkenyi, G.: Trajectory tracking control of a 6-degree-of-freedom robot arm using nonlinear optimization. Advanced Motion Control, 2008 AMC '08. 10th IEEE International Workshop on 26–28 March 2008, pp. 655–660 (2008)
16. Lombai, F., Szederkenyi, G.: Throwing motion generation using nonlinear optimization on a 6-degree-of-freedom robot manipulator. In: Proc. of IEEE International Conference on Mechatronics (2009)
17. van Dijk, N.J.M., van de Wouw, N., Nijmeijer, H., Pancras, W.C.M.: Path-constrained motion planning for robotics based on kinematic constraints. In: Proc. of IDETC/CIE ASME (2007)
18. Dumetz, E., Dieulot, J.Y., Barre, P.J., Colas, F., Delplace, T.: Control of an industrial robot using acceleration feedback. J. Intell. Robot. Syst. **46**(2), 111–128 (2006)

19. Cano, T., Chapelle, F., Lavest, J.M., Ray, P.: A new approach to identifying the elastic behaviour of a manufacturing machine. Int. J. Mach. Tools Manuf. **48**(14), 1569–1577 (2008)
20. Gasparetto, A., Zanotto, V.: A new method for smooth trajectory planning of robot manipulators. Mech. Mach. Theory **42**(4), 455–471 (2007)
21. Gasparetto, A., Zanotto, V.: A technique for time-jerk optimal planning of robot trajectories. Robot. Comput.-Integr. Manuf. **24**(3), 415–426 (2008)
22. De Boor, C.: A Practical Guide to Splines. Springer-Verlag, New York (1978)