Advances in Engineering Software 41 (2010) 548–556

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

Optimal trajectory planning for industrial robots

A. Gasparetto *, V. Zanotto

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

ARTICLE INFO

Article history:
Received 6 October 2008
Received in revised form 19 June 2009
Accepted 6 November 2009
Available online 22 December 2009

Keywords:
Trajectory planning
Jerk
Robot manipulators
Optimization
B-splines
Cubic splines

ABSTRACT

An analysis of the results of an algorithm for optimal trajectory planning of robot manipulators is described in this paper. The objective function to be minimized is a weighted sum of the integral squared jerk and the execution time. Two possible primitives for building the trajectory are considered: cubic splines or fifth-order B-splines. The proposed technique allows to set constraints on the robot motion, expressed as upper bounds on the absolute values of velocity, acceleration and jerk. The described method is then applied to a 6-d.o.f. robot (a Cartesian gantry manipulator with a spherical wrist); the results obtained using the two different primitives are presented and discussed.

1. Introduction

The trajectory planning problem in Robotics is thus defined: find a motion law along a given geometric path, taking into account some predefined requirements, so as to be able to generate suitable reference inputs for the control system of the manipulator. The inputs of the trajectory planning are: the geometric path, the kinematic and dynamic constraints, while the output is the trajectory of the joints (or of the end-effector), expressed as a time sequence of position, velocity and acceleration values.

The trajectory generated by the planner must not excite the mechanical resonances of the manipulator: this is the case if it is *smooth*, i.e. the trajectory itself and some of its derivatives are continue functions. 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.

Most of the proposed trajectory planning algorithms are based on a minimization of an objective function, that depends on execution time, actuator effort, absolute value of the jerk, or a combination of these variables.

The first proposed trajectory planning techniques were minimum-time algorithms, due to the need of increasing the productivity in the industrial sector. Some versions of this kind of algorithms can be found in [1–3]. The main disadvantage of minimum-time algorithms is that the trajectories have discontinuous values of acceleration and joint torque; hence, dynamic problems arise during the execution of the trajectory. 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.

An alternative approach lies in minimizing the energy consumption instead of the execution time. This approach leads to smooth trajectories, with smaller stresses on the manipulator structure and on the actuators. Some examples of energy optimal trajectory planning can be found in [4–7].

Other approaches seek a jerk-optimal trajectory. This kind of approach allows to reduce the errors during trajectory tracking, the stresses to the actuators as well as to the mechanical structure of the robot, and the excitation of resonance frequencies. Some examples of this approach can be found in [8–11]. Kyriakopoulos and Saridis [8] exploit the Pontryagin principle to obtain minimum-jerk point-to-point trajectories through a minimax approach. Simon and Isik [9] use trigonometric splines to interpolate the trajectory ensuring the jerk continuity. Piazzi and Visioli [10,11] use an *interval analysis* method to generate trajectories which globally minimize 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

* Corresponding author. Tel.: +39 0432 558257; fax: +39 0432 558251.
E-mail address: gasparetto@uniud.it (A. Gasparetto).

0965-9978/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.advengsoft.2009.11.001

the intervals between the via-points are computed so that the lowest possible jerk peak is produced.

In this paper, the results of a trajectory planning algorithm (presented in [12,13]) applied to a 6-d.o.f. robot, namely a Cartesian manipulator with a spherical wrist, are evaluated. This algorithm makes possible to set constraints on the robot motion before 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 when planning its trajectory. Moreover, unlike most jerk-minimization methods, this technique does not require an *a priori* setting of the total execution time.

The paper is organized as follows: the trajectory planning algorithm under evaluation is described in Section 2. Application of the technique by means of cubic splines or fifth-order B-splines is described in Sections 3 and 4, respectively. Section 5 deals with the application of the algorithm to the kinematics of a laboratory manipulator: the results obtained are presented, compared and discussed.

2. The trajectory planning algorithm

The algorithm (described with more details in [12,13]) starts assuming that a geometric path is available, generated by an upper-level planner. The path consists of a sequence of via-points defined in the operating space of the robot (i.e. a sequence of positions and orientations of the end-effector).

Unlike most minimum-jerk trajectory planning techniques found in the scientific literature, the execution time is not forced *a priori*, and some constraints, namely the upper bounds on velocity, acceleration and jerk, are taken into account. In other words, the algorithm considers a hybrid objective function, made of two terms having opposite effects, one being proportional to the execution time and the other to the integral of the squared jerk. Minimization of the former term would yield large values of velocity, acceleration and jerk, while minimization of the latter would yield a smoother trajectory. By suitably choosing the two terms of the objective function, a balance between speed and smoothness can be reached.

The optimal trajectory planning problem is defined as:

$$\begin{cases} \text{find}: \\ \min k_r N \sum_{i=1}^{vp-1} h_i + k_J \sum_{j=1}^{N} \int_0^{t_f} (\dddot{q}_j(t))^2 dt \\ \text{subject to}: \\ |\dot{q}(t)| \leqslant VC_j \quad j = 1 \div N \\ |\ddot{q}(t)| \leqslant WC_j \quad j = 1 \div N \\ |\dddot{q}(t)| \leqslant JC_j \quad j = 1 \div N \end{cases} \tag{1}$$

Table 1
Meaning of the symbols appearing in Eq. (1).

Symbol	Meaning
N	Number of robot joints
vp	Number of via-points
h_i	Time interval between two via-points
$\dot{q}_j$	Velocity of the jth joint
$\ddot{q}_j$	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
VC_j	Velocity limit for the jth joint (symmetrical)
WC_j	Acceleration limit for the jth joint (symmetrical)
JC_j	Jerk limit for the jth joint (symmetrical)

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

The output of the trajectory planning technique is given by the vector of the time intervals h_i between any pair of consecutive via-points that minimizes the objective function (1).

In order to apply the algorithm, a suitable type of primitives for building the trajectory must be chosen. Cubic splines and fifth-order B-splines will be tested, respectively.

3. Definition of the trajectory by means of cubic splines

We recall that a cubic spline is a third-order polynomial which interpolates a sequence of vp via-points in the joint space, obtained by kinematic inversion from an original sequence in the operating space.

We briefly explain here how the expression of the objective function (1) can be obtained by substituting into (1) the expression of the cubic splines $Q_{j,i}(t)$, i.e. the cubic polynomial for the jth joint defined on the interval $[t_i, t_{i+1}]$. As stated in the foregoing, the intervals time $h_i = t_{i+1} - t_i$ are the output of the algorithm. If we assume that the constraints are constant and symmetric for velocities, accelerations and jerks, the explicit expressions of the constraints are:

$$\begin{aligned} &|\dot{Q}_{j,i}(t)| \leqslant VC_j \quad \forall j = 1 \div N, \forall i = 1 \div n-1 \\ &|\ddot{Q}_{j,i}(t)| \leqslant WC_j \quad \forall j = 1 \div N, \forall i = 1 \div n-1 \\ &|\dddot{Q}_{j,i}(t)| \leqslant JC_j \quad \forall j = 1 \div N, \forall i = 1 \div n-1 \end{aligned} \tag{2}$$

Such constraints may be called "semi-infinite", holding for any value of the continuous variable t. Transforming them into finite constraints, we obtain:

$$\begin{aligned} &\max\{|\dot{Q}_{j,i}(t_i)|, |\dot{Q}_{j,i}(t_{i+1})|, |\dot{Q}^*_{j,i}|\} \leqslant VC_j \quad j = 1 \div N,\ i = 1 \div n-1 \\ &\max\{|\alpha_{j,1}|, \ldots, |\alpha_{j,n}|\} \leqslant WC_j,\ \forall j = 1 \div N \\ &\left|\frac{\alpha_{j,i+1} - \alpha_{j,i}}{h_i}\right| \leqslant JC_j \forall \quad j = 1 \div N, \forall i = 1 \div n-1 \end{aligned} \tag{3}$$

There is an additional constraints on the length of the trajectory intervals h_i:

$$h_i > w_i = \max_{j=1,\ldots,N} \left\{ \frac{|q_{j,i+1} - q_{j,i}|}{VC_j} \right\} > 0 \tag{4}$$

The expression of the objective function (1) for a trajectory made of cubic splines is thus given by:

$$FOBJ = k_T \sum_{i=1}^{n-1} h_i + k_J \sum_{j=1}^{N} \sum_{i=1}^{n-1} \left[\frac{(\alpha_{j,i+1} - \alpha_{j,i})^2}{h_i} \right] \tag{5}$$

By varying the values of the weights k_T and k_J it is possible to choose between the need for a quick execution or the need for a smooth trajectory. The limit conditions are $k_T = 0$ (i.e. minimum-jerk trajectory) and $k_J = 0$ (i.e. minimum-time trajectory).

4. Definition of the trajectory by means of fifth-order B-splines

Instead of using cubic splines, fifth-order B-splines can be used in order to obtain another expression for the objective function (1). We recall 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 [14]: