Robotics and Computer-Integrated Manufacturing 29 (2013) 53–62

Contents lists available at SciVerse ScienceDirect

Robotics and Computer-Integrated Manufacturing

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

Integrated sliding-mode algorithms in robot tracking applications

Luis Gracia [a,*], Fabricio Garelli [b], Antonio Sala [a]

[a] Department of Systems Engineering and Control, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
[b] CONICET and Universidad Nacional de La Plata, C.C.91 (1900), La Plata, Argentina

ARTICLE INFO

Article history:
Received 7 February 2012
Received in revised form
9 July 2012
Accepted 26 July 2012
Available online 21 August 2012

Keywords:
Sliding mode
Robot control
Collision avoidance

ABSTRACT

An integrated solution based on sliding mode ideas is proposed for robotic trajectory tracking. The proposal includes three sliding-mode algorithms for speed auto-regulation, path conditioning and redundancy resolution in order to fulfill velocity, workspace and C-space constraints, respectively. The proposed method only requires a few program lines and simplifies the robot user interface since it directly deals with the fulfillment of the constraints to find a feasible solution for the robot trajectory tracking in a short computation time. The proposed approach is evaluated in simulation on the freely accessible 6R robot model PUMA-560, for which the main features of the method are illustrated.

1. Introduction

The main objective of robot control systems is the tracking of a reference trajectory, which involves the generation of a control signal to make the error between the robot position and the reference zero [1]. In this sense, this work presents an integrated solution for robotic trajectory tracking based on three sliding-mode algorithms recently proposed by the authors[1] for speed auto-regulation [2], path conditioning [3] and redundancy resolution [4] in order to fulfill velocity, workspace and C-space constraints, respectively. These constraints may be due to different reasons such as joint speed limits [5], joint angle limits [6], obstacle collision avoidance [7], etc.

The proposed approach, which only requires a few program lines, simplifies the user interface since the method directly deals with the fulfillment of the constraints specified by the robot end-user. Therefore, in case of relatively simple tasks the proposed method finds a feasible solution for the robot trajectory tracking in a short computation time.

The proposed approach can be useful for many type of robots and industrial applications, such as spray painting [8], arc welding [9], assembly [10], polishing [11], etc. For instance, in this work it is used a well-known and free-access six-revolute (6R) robot, the PUMA 560, for which the main distinctive features of the method are illustrated in a spray painting application.

The outline of the paper is as follows. Next section introduces some preliminaries, while Sections 3–5 present the three sliding-mode algorithms proposed for speed auto-regulation, path conditioning and redundancy resolution, respectively. A discussion about the method is given in Section 6. The proposed approach is applied in Section 7 to the PUMA-560 robot model in order to show the feasibility and effectiveness of the method. Finally, some concluding remarks are given.

2. Preliminaries and control scheme

2.1. Notation

Following the standard notation [12], consider a robot system with $\mathbf{q}=[q_1 \dots q_n]^{\mathrm{T}}$ being the robot *configuration* or *n*-dimensional joint position vector and $\mathbf{p}=[p_1 \dots p_m]^{\mathrm{T}}$ being the robot *pose* or *m*-dimensional workspace position vector. A robot is said to be *redundant* when the dimension m of the workspace is less than the dimension n of the configuration space (hereafter, C-space), i.e., $m<n$. The degree of kinematic redundancy is computed as $n-m$. For the rest of the paper it is assumed that the robot at hand is redundant.

The relationship between the robot configuration and the robot pose is highly nonlinear, generically expressed as

$$\mathbf{p}=\mathbf{l}(\mathbf{q}), \quad (1)$$

* Corresponding author. Tel.: +34 963879770; fax: +34 963879579.
E-mail addresses: luigraca@isa.upv.es (L. Gracia), fabricio@ing.unlp.edu.ar (F. Garelli), asala@isa.upv.es (A. Sala).

[1] The three algorithms were individually developed and tested by the authors in previous works [2–4]. However, all three are adapted in this work to be used in a more general framework than that of previous works. In this sense, this research effectively integrates the three algorithms in the same robot control scheme and shows their complementarity for robot trajectory tracking. Another important contribution of this work is the given pseudo-code for the proposed approach (including the three sliding-mode algorithms) so that it can be easily implemented on many actual robot platforms.

0736-5845/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.rcim.2012.07.007

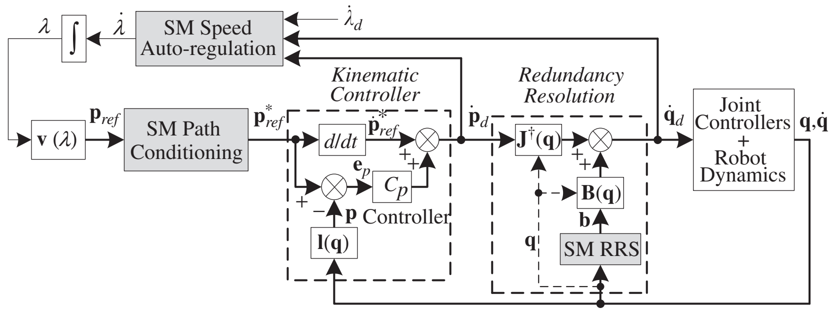

Fig. 1. Robotic trajectory tracking control scheme with SM Algorithms (shaded blocks).

where the function $\mathbf{l}$ is called the kinematic function of the robot model.

The first-order kinematics results in

$$\dot{\mathbf{p}} = \frac{\partial \mathbf{l}(\mathbf{q})}{\partial \mathbf{q}} \dot{\mathbf{q}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \tag{2}$$

where $\mathbf{J}(\mathbf{q})$ is denoted as the $m \times n$ Jacobian matrix or simply *Jacobian* of the kinematic function.

Let us denote as $\mathbf{p}_{ref}(t)$ the workspace reference, which can be usually expressed in terms of a desired path function $\mathbf{v}(\lambda)$ whose argument is the so-called motion parameter $\lambda(t)$ as

$$\mathbf{p}_{ref} = \mathbf{v}(\lambda). \tag{3}$$

Finally, the gradient of a scalar function $f(x_1, \ldots, x_n)$ will be denoted

$$\nabla f = \left[\frac{\partial f}{\partial x_1} \cdots \frac{\partial f}{\partial x_n}\right]^{\mathrm{T}}.$$

2.2. Control scheme

Fig. 1 shows the control scheme proposed in this work for robotic trajectory tracking, which contains three sliding-mode (SM) algorithms for speed auto-regulation, path conditioning and redundancy resolution. The SM speed auto-regulation block generates the motion rate parameter $\dot{\lambda}$ so that it is as close as possible to the desired value $\dot{\lambda}_d$ and that satisfies velocity constraints on the desired workspace velocity vector $\dot{\mathbf{p}}_d$, desired joint velocity vector $\dot{\mathbf{q}}_d$ and robot state $(\mathbf{q},\dot{\mathbf{q}})$, see Section 3. The SM path conditioning block generates a modified workspace reference $\mathbf{p}^*_{ref}$ to be sent to the robot kinematic controller so that it is as close as possible to the original value $\mathbf{p}_{ref}$ and that belongs to the allowed workspace, see Section 4. The redundancy resolution block computes the desired joint velocity vector $\dot{\mathbf{q}}_d$ in order to track the desired workspace velocity vector $\dot{\mathbf{p}}_d$ as primary task, while a secondary goal is achieved using redundancy in order to satisfy C-space constraints on the robot state $(\mathbf{q},\dot{\mathbf{q}})$, see Section 5. The kinematic controller generates the workspace velocity vector $\dot{\mathbf{p}}_d$ closing a loop using the robot state and the modified workspace reference $\mathbf{p}^*_{ref}$ in order to make the tracking error zero.

Kinematic controller: For this work, it is considered a classical kinematic controller utilized for robotic trajectory tracking [13], see Fig. 1, which consists of a two-degree of freedom (2-DOF) control that incorporates a correction based on the position error $\mathbf{e}_p = \mathbf{p}^*_{ref} - \mathbf{p}$ by means of the position loop controller C_p plus a feedforward term depending on the first-order time derivative of the modified workspace reference, i.e. $\dot{\mathbf{p}}^*_{ref}$. Note that the path function $\mathbf{v}(\lambda)$ needs to be differentiable due to the feedforward term. For instance, if the reference path is given by a set of tracking points generated by the robot operator, it can be made smooth and continuous by using spline or Bézier interpolation.

Classical redundancy resolution: The desired joint velocity vector $\dot{\mathbf{q}}_d$ is computed by the redundancy block in Fig. 1 in order to satisfy the first-order kinematic relation

$$\dot{\mathbf{p}}_d = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}_d. \tag{4}$$

In general, in the case of redundant robots an infinite number of solutions for $\dot{\mathbf{q}}_d$ satisfying (4) exist,[2] which are given by

$$\dot{\mathbf{q}}_d = \mathbf{J}^{\dagger}(\mathbf{q})\dot{\mathbf{p}}_d + \mathbf{B}(\mathbf{q})\mathbf{b}, \tag{5}$$

where $\mathbf{J}^{\dagger}(\mathbf{q})$ is the so-called *right pseudo-inverse* of $\mathbf{J}(\mathbf{q})$ (i.e., $\mathbf{J}^{\dagger} \equiv \mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1}$); $\mathbf{B}(\mathbf{q})$ is an $n \times n$ matrix whose last m column vectors are the n-dimensional null vector and whose first $n-m$ column vectors form an orthonormal basis for the null space of $\mathbf{J}(\mathbf{q})$ (e.g., this basis can be easily obtained from the singular value decomposition [15] of $\mathbf{J}(\mathbf{q})$); and $\mathbf{b}$ is the so-called *performance vector* which is an arbitrary n-dimensional column vector. The first term in (5) represents the minimum-norm solution or *base solution*, while the second term is the *homogeneous solution* that gives rise to infinite possible solutions for $\dot{\mathbf{q}}_d$ depending on the value of performance vector $\mathbf{b}$. In general, this vector can be expressed as a function of the robot state, i.e. $\mathbf{b}(\mathbf{q},\dot{\mathbf{q}})$. The reader is referred to literature for choices of performance vector [16,6,4].

2.3. Constrained control via sliding modes

Consider the following dynamical system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u + \boldsymbol{v}, \tag{6}$$

where $\mathbf{x}$ is the state vector, u is the control input (which has been assumed scalar for simplicity), $\mathbf{f}$ and $\mathbf{g}$ are two vector fields of $\mathbf{x}$ and vector $\boldsymbol{v}$ accounts for the system uncertainty.

Consider also the constraint

$$\sigma(\mathbf{x}) \le 0, \tag{7}$$

where σ is a function of the state vector whose first-order time-derivative is obtained as

$$\dot{\sigma} = \frac{\partial \sigma(\mathbf{x})}{\partial \mathbf{x}}^{\mathrm{T}} (\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u + \boldsymbol{v}). \tag{8}$$

Provided

$$\frac{\partial \sigma(\mathbf{x})}{\partial \mathbf{x}}^{\mathrm{T}} \mathbf{g}(\mathbf{x})\|_{\sigma(\mathbf{x})=0} \neq 0,$$

condition $\dot{\sigma} < 0$ can be ensured on the border $\sigma(\mathbf{x}) = 0$ by means of a high enough input u, so as to avoid violating constraint (7). In this sense, the following switching law

$$u = \begin{cases} u_{SM} & \text{if } \sigma(\mathbf{x}) \ge 0 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

[2] It is implicitly assumed that $\mathbf{J}(\mathbf{q})$ is full row rank, since otherwise the robot configuration $\mathbf{q}$ is said to be *singular* [14] and the desired workspace velocity vector $\dot{\mathbf{p}}_d$ in general cannot be achieved.