INTERNATIONAL JOURNAL OF ROBUST AND NONLINEAR CONTROL
Int. J. Robust. Nonlinear Control 2017; **00**:1–32
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/rnc

Sliding Mode Control for Robust and Smooth Reference Tracking in Robot Visual Servoing

Pau Muñoz-Benavent[1]*, Luis Gracia[1], J. Ernesto Solanes[1], Alicia Esparza[1], and Josep Tornero[1]

[1]*Instituto IDF, Universitat Politecnica de Valencia, Camino de Vera s/n, 46022 Valencia, Spain*

SUMMARY

An approach based on sliding mode is proposed in this work for reference tracking in robot visual servoing. In particular, two sliding mode controls are obtained depending on whether the joint accelerations or the joint jerks are considered as the discontinuous control action. Both sliding mode controls are extensively compared in 3D simulated environment to their equivalent well-known continuous controls, which can be found in the literature, in order to highlight their similarities and differences. The main advantages of the proposed method are smoothness, robustness and low computational cost. The applicability and robustness of the proposed approach is substantiated by experimental results using a conventional 6R industrial manipulator (the KUKA KR6 R900 sixx, Agilus) for positioning and tracking tasks.

Received . . .

KEY WORDS: Sliding mode control; robust control; nonlinear control; control applications; robot system

1. INTRODUCTION

Robot visual feedback control or visual servoing (VS) [1–3], has been being studied along the last four decades by many researchers, mostly from the areas of control and computer science. It is a viable method for robot control based on the utilization of visual information extracted from images to close the control loop. For this purpose, a computer vision algorithm must be used to obtain the *visual features* of the target object present in the scene and observed by the camera. This information is used to compute the robot control law in order to achieve the desired robot pose.

Taking into consideration the workspace in which the control law is computed, the following classification can be made [2]: position-based VS (PBVS), in which the control law is carried out in the operational space, and image-based VS (IBVS), in which the control law is directly computed in the image space.

Independently of the workspace in which the control is carried out, another classification can be done focusing on the control law nature: *continuous* or *discontinuous control laws* [4].

On the one hand, the most typical continuous control law used in VS applications for positioning or tracking tasks is based on computing a continuous joint velocity [2] to be commanded to the

*Correspondence to: Instituto IDF, Universitat Politecnica de Valencia, Camino de Vera s/n, 46022 Valencia, Spain. E-mail: pmunyoz@idf.upv.es

Contract/grant sponsor: Spanish Government and Conselleria d'Educacio Generalitat Valenciana; contract/grant number: BES-2010-038486, DPI2013-42302-R, and VALi+d APOSTD/2016/044

joint actuators in order to obtain an exponential decrease of the error signal. In this respect, we can find in the literature a vast number of approaches, as for instance: classic PID controllers [5–9]; optimal [10–13] and robust controllers [14–17]; based on learning [18–20]; etc. However, other continuous approaches that are based on computing the joint accelerations to be commanded to the joint actuators can be found, either in PBVS [21] or IBVS [22].

On the other hand, *discontinuous control laws* have been deeply studied in the context of sliding mode control (SMC) [23]. In particular, several works have studied the use of SMC for the main control law[†] of the VS system, mainly to increase its robustness against errors: authors in [25] used sliding mode (SM) theory to design a 3D vision based controller that is robust to bounded parametric estimation errors; in [26] and [27], a SMC strategy based on a switching scheme and monitoring function was developed to deal with the uncertainties in the camera calibration parameters; authors in [28] proposed a visual controller and a robot joint controller based on the SMC theory for a camera-in-hand planar two-link robot visual servo system in order to achieve strong robustness against parameter variations and external disturbances; in [29], the SMC was applied to IBVS in order to increase the robustness on the parametric uncertainties; in [30], a SMC together with an estimator based on unscented Kalman observer cascading with Kalman filter demonstrated to be a stable and robust structure in PBVS, considering system uncertainties existing in the estimation model and observation noise; in [31], a second order SMC for PBVS was presented in order to control the end effector pose of a 7 DoF robot arm in eye-to-hand configuration; in [32] and [33], a robust tracking control law under image noise and uncertainty of parameters was designed on the basis of SM theory for mobile robots using epipolar geometry in IBVS; in [34], the SMC was integrated with kernel-based visual servoing to improve the tracking error and expand the stability region; in [35], rotation and translation SMC using SIFT features were designed to solve the robot visual servoing problem; in [36], a two stage control scheme based on sliding surfaces was proposed for path-following and accurate positioning in PBVS for a robotic riveting system.

All works mentioned above use the joint velocities as the discontinuous control action for the SMC. Therefore, the objective of this work is to develop a SMC for VS that uses a high-order discontinuous control signal, i.e., joint accelerations or joint jerks, in order to obtain a *smoother behavior* and ensure the robot system stability. The proposed SMC approach is equivalent, in some sense, to the continuous control strategy mentioned above but has two main advantages: *robustness* and *low computational cost*; while its main limitation is the chattering drawback, although this problem becomes negligible for reasonable fast sampling rates.

The structure of the paper is as follows. Next section introduces some preliminaries, while Sec. 3 presents the basic SM theory used in this work. The proposed method for reference tracking is developed in Sec. 4, while it is theoretically compared to the classical VS strategy for reference tracking in Sec. 5. Subsequently, the main advantages of the proposed approach are discussed in Sec. 6, while some additional remarks are given in Sec. 7. Next, Sec. 8 presents the conditions considered for the simulations and experiments. Sec. 9 and Sec. 10 compare in simulation the proposed approach to its equivalent continuous counterpart to highlight the similarities and differences. The applicability, effectiveness and robustness of the proposed approach is substantiated by experimental results in Sec. 11 using a conventional 6R industrial manipulator (the KUKA KR6 R900 sixx, AGILUS) for positioning and tracking tasks. Finally, some concluding remarks are given.

2. PRELIMINARIES

Coordinate frames. Fig. 1 shows the coordinate frames involved in the VS problem: F base robot frame; E end-effector robot frame; C current camera frame; C^* desired camera frame; O object frame; $C2$ camera frame for eye-to-hand configuration (in this case the camera does not move with the robot, i.e., it is static, and frame C^* would be replaced by E^*).

[†]SMC has also been used in VS for other purposes. For instance, authors in [24] presented an approach in which a SM observer is applied to estimate the joint velocities of the VS system.

DOI: 10.1002/rnc

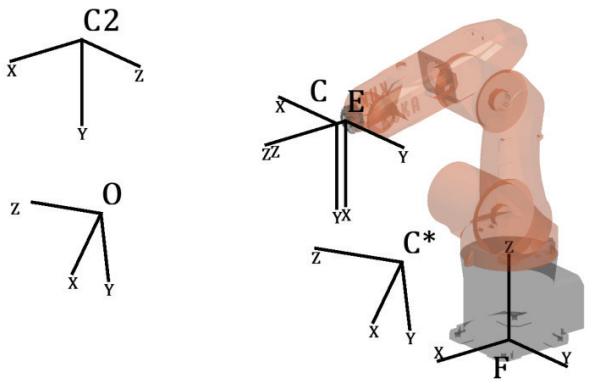

Figure 1. Frames involved in visual servoing.

Kinematics. Following the standard notation [2], the VS application is characterized by the visual *feature vector* $\mathbf{s}$ that depends on the robot *configuration* $\mathbf{q}$ and also explicitly on time for the general case of a *moving target*, that is:

$$\mathbf{s} = \mathbf{l}(\mathbf{q}, t), \tag{1}$$

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

The first-order kinematics of the feature vector $\mathbf{s}$ results in:

$$\dot{\mathbf{s}} = \frac{\partial \mathbf{l}(\mathbf{q}, t)}{\partial \mathbf{q}} \dot{\mathbf{q}} + \frac{\partial \mathbf{l}(\mathbf{q}, t)}{\partial t} = \mathbf{J}_s \dot{\mathbf{q}} + \partial \mathbf{s}/\partial t, \tag{2}$$

where $\partial \mathbf{s}/\partial t$ is due to the target motion and $\mathbf{J}_s$ is the resulting Jacobian matrix, which can be expressed as a concatenation of three different Jacobian matrices:

$$\mathbf{J}_s(\mathbf{q}, t) = \mathbf{L}_s(\mathbf{q}, t)\, {}^c\mathbf{V}_e\, {}^e\mathbf{J}_e(\mathbf{q}), \tag{3}$$

where $\mathbf{L}_s$ is the image Jacobian, also known as *interaction matrix*, related to the visual feature vector $\mathbf{s}$; ${}^c\mathbf{V}_e$ is the constant twist transformation matrix from the camera frame C to the robot end-effector frame E; and ${}^e\mathbf{J}_e$ is the robot Jacobian expressed in the end-effector frame. For more details about matrix $\mathbf{J}_s$ see [2].

The second- and third-order kinematics of the feature vector $\mathbf{s}$ result in:

$$\ddot{\mathbf{s}} = \mathbf{J}_s \ddot{\mathbf{q}} + \dot{\mathbf{J}}_s \dot{\mathbf{q}} + \partial \dot{\mathbf{s}}/\partial t. \tag{4}$$

$$\dddot{\mathbf{s}} = \mathbf{J}_s \dddot{\mathbf{q}} + 2\dot{\mathbf{J}}_s \ddot{\mathbf{q}} + \ddot{\mathbf{J}}_s \dot{\mathbf{q}} + \partial \ddot{\mathbf{s}}/\partial t. \tag{5}$$

Reference. The robot system should carry out a task, which in VS applications refers to achieving a reference value for the visual feature vector $\mathbf{s}$ and is given by the following equation:

$$\mathbf{s}(\mathbf{q}, t) = \mathbf{s}_{ref}(t), \tag{6}$$

where $\mathbf{s}_{ref}(t)$ is the reference trajectory for the visual feature vector.

Computer vision algorithm. This algorithm is composed of three parts: the first part consists of the *image processing* for obtaining the image plane coordinates (u_i, v_i) of all the feature points; the second part consists of the *coordinate transformation* for converting the pixel coordinates (u_i, v_i) to the corresponding value in the normalized image plane using the matrix of the camera intrinsic parameters; and the third part, which only applies for PBVS, consists of the *pose estimation* of the camera (eye-in-hand) or robot (eye-to-hand) from the feature points of the second part. The output of the computer vision algorithm, both for PBVS and IBVS, is the visual feature vector $\mathbf{s}$. This work assumes existence of this computer vision algorithm.

DOI: 10.1002/rnc

Robot control. This work also assumes the existence of an underlying robot control in charge of achieving a particular joint velocity, joint acceleration or joint jerk (depending on the control case) from a command. Nevertheless, the actual joint value will not be exactly the commanded one due to the dynamics of the low-level control loop and the inaccuracies because of disturbances. However, in this work it is assumed that the dynamics of the low-level control loop is fast enough compared to that of the joint commanded variable so that the relationships below hold approximately true, avoiding the need of including extra state variables. Therefore, depending on the control application (velocity, acceleration or jerk) the following equations are considered:

$$\dot{\mathbf{q}} = \dot{\mathbf{q}}_c + \mathbf{d}_{cv} \tag{7}$$

$$\ddot{\mathbf{q}} = \ddot{\mathbf{q}}_c + \mathbf{d}_{ca} \tag{8}$$

$$\dddot{\mathbf{q}} = \dddot{\mathbf{q}}_c + \mathbf{d}_{cj}, \tag{9}$$

where subscript c is used for the commanded variable and $\{\mathbf{d}_{cv}, \mathbf{d}_{ca}, \mathbf{d}_{cj}\}$ represent the inaccuracies of the low-level control loop for each case.

Note that, the *dynamic model* of the robot system should be taken into account to properly design the mentioned underlying joint controller. Obviously, for stability reasons, the bandwidth of this underlying robot control should be faster than that of the used kinematic control.

3. SLIDING MODE CONTROL THEORY

3.1. Sliding mode control action

Theorem 1
Consider the following dynamical system with n_x states and n_u inputs given by:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{d}) + \mathbf{g}(\mathbf{x})\,\mathbf{u}, \tag{10}$$

where $\mathbf{x}(t)$ is the state vector, $\mathbf{d}(t)$ is an unmeasured disturbance or model uncertainty, $\mathbf{u}(t)$ is the control input vector (possibly discontinuous), $\mathbf{f}$ is a vector field and $\mathbf{g}$ is a set of vector fields.

Consider also that the system state vector $\mathbf{x}$ is subject to user-specified equality constraints $\phi_i(\mathbf{x}) = 0$, $i = 1, \ldots, N$, where $\phi_i(\mathbf{x})$ is the ith equality constraint function. Thus, the region Φ of the state space compatible with the constraints on state $\mathbf{x}$ is given by:

$$\Phi = \{\mathbf{x} \mid \phi_i(\mathbf{x}) = 0\} \tag{11}$$

with $i = 1, \ldots, N$.

Then, assuming that the constraint function ϕ_i is differentiable, the variable structure control below guarantees that the system converges to Φ in finite time and remains there henceforth:

$$\mathbf{u} = -\mathbf{L}_\mathbf{g}\boldsymbol{\phi}^{\mathrm{T}}\,\mathrm{sign}(\boldsymbol{\phi})\,u^+ \tag{12}$$

$$u^+ > \|L_f\boldsymbol{\phi}\|_1 / \mathrm{eig}_{\min}(\mathbf{L}_\mathbf{g}\boldsymbol{\phi}\,\mathbf{L}_\mathbf{g}\boldsymbol{\phi}^{\mathrm{T}}), \tag{13}$$

where the scalar $L_f\phi_i$ and the row vector $\mathbf{L}_\mathbf{g}\phi_i$ denote the Lie derivatives of $\phi_i(\mathbf{x})$ in the direction of vector field $\mathbf{f}$ and in the direction of the set of vector fields $\mathbf{g}$, respectively, column vector $L_f\boldsymbol{\phi}$ contains the elements $L_f\phi_i$ of all constraints, matrix $\mathbf{L}_\mathbf{g}\boldsymbol{\phi}$ contains the row vectors $\mathbf{L}_\mathbf{g}\phi_i$ of all constraints, $\boldsymbol{\phi}$ is a column vector with all the constraint functions ϕ_i, $\mathrm{sign}(\cdot)$ represents the *sign function* (typically used in SMC), positive scalar u^+ is the so-called switching gain, which can be either constant or varying in time, $\|\cdot\|_1$ represents the 1-norm (also known as the Taxicab norm) and function $\mathrm{eig}_{\min}(\cdot)$ computes the minimum eigenvalue of a matrix.

Proof

Assuming that $\boldsymbol{\phi}(0) \neq \mathbf{0}$, the goal of this proof is to show that convergence to point $\boldsymbol{\phi} = \mathbf{0}$ is achieved in finite time.

DOI: 10.1002/rnc

Taking the time derivative of the constraint function ϕ_i results in:

$$\frac{d(\phi_i(\mathbf{x}))}{dt} = \frac{\partial \phi_i(\mathbf{x})^{\mathrm{T}}}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}, \mathbf{d}) + \frac{\partial \phi_i(\mathbf{x})^{\mathrm{T}}}{\partial \mathbf{x}} \mathbf{g}(\mathbf{x})\, \mathbf{u} = L_f \phi_i(\mathbf{x}, \mathbf{d}) + \mathbf{L_g} \phi_i(\mathbf{x}) \mathbf{u}.. \tag{14}$$

From (14) and (12), the column vector $\dot{\boldsymbol{\phi}}$ composed of the constraint function derivatives $\dot{\phi}_i$ is given by:

$$\dot{\boldsymbol{\phi}} = L_f \boldsymbol{\phi} - \left(\mathbf{L_g} \boldsymbol{\phi}\, \mathbf{L_g} \boldsymbol{\phi}^{\mathrm{T}}\right)\, \mathbf{z}\, u^+, \tag{15}$$

where $\mathbf{z}$ is a column vector with the ith-component $z_i = 1$ if $\phi_i > 0$ and $z_i = -1$ if $\phi_i < 0$.

Let $V = \mathbf{z}^{\mathrm{T}}\, \boldsymbol{\phi}$ be a Lyapunov function candidate. Vector $\boldsymbol{\phi}$ can be generically partitioned into two subvectors $\boldsymbol{\phi} = [\boldsymbol{\phi}^{a\,\mathrm{T}} \quad \boldsymbol{\phi}^{N-a\,\mathrm{T}}]^{\mathrm{T}}$, where SM occurs in the manifold given by $\boldsymbol{\phi}^a = \mathbf{0}_a$, whereas the components of vector $\boldsymbol{\phi}^{N-a}$ are not zero. Obviously, one of these two subvectors may be empty at a certain time. Since vector $\mathbf{z}^{N-a}$ is constant, the time derivative of V results in:

$$\dot{V} = \frac{d}{dt}\left(\mathbf{z}^{\mathrm{T}}\, \boldsymbol{\phi}\right) = \frac{d}{dt}\left(\begin{bmatrix} \mathbf{z}^a \\ \pm\mathbf{1}_{N-a} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \boldsymbol{\phi}^a \\ \boldsymbol{\phi}^{N-a} \end{bmatrix}\right) = \begin{bmatrix} \dot{\mathbf{z}}^a \\ \mathbf{0}_{N-a} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \mathbf{0}_a \\ \boldsymbol{\phi}^{N-a} \end{bmatrix} + \mathbf{z}^{\mathrm{T}}\, \dot{\boldsymbol{\phi}} = \mathbf{z}^{\mathrm{T}}\, \dot{\boldsymbol{\phi}}, \tag{16}$$

where $\pm\mathbf{1}$ represents a column vector with all its elements equal to 1 or -1.

Replacing vector $\dot{\boldsymbol{\phi}}$ with its value from (15), it is obtained:

$$\dot{V} = \mathbf{z}^{\mathrm{T}}\, L_f \boldsymbol{\phi} - \mathbf{z}^{\mathrm{T}}\, \left(\mathbf{L_g} \boldsymbol{\phi}\, \mathbf{L_g} \boldsymbol{\phi}^{\mathrm{T}}\right)\, \mathbf{z}\, u^+. \tag{17}$$

The components of vector $\mathbf{z}$ range from -1 to 1, hence the upper bound of the first term in (17) is given by $z_i = 1$ if $L_f \phi_i > 0$ and $z_i = -1$ if $L_f \phi_i < 0$, that is:

$$\mathbf{z}^{\mathrm{T}}\, L_f \boldsymbol{\phi} \leq \sum_{i=1}^{N} |L_f \phi_i| = \|L_f \boldsymbol{\phi}\|_1. \tag{18}$$

Assuming that $u^+ > 0$, the second term in (17) is negative, since matrix $\left(\mathbf{L_g} \boldsymbol{\phi}\, \mathbf{L_g} \boldsymbol{\phi}^{\mathrm{T}}\right)$ is positive definite, and its upper bound is given by:

$$-\,\mathbf{z}^{\mathrm{T}}\, \left(\mathbf{L_g} \boldsymbol{\phi}\, \mathbf{L_g} \boldsymbol{\phi}^{\mathrm{T}}\right)\, \mathbf{z}\, u^+ \leq -\mathrm{eig}_{\min}\left(\mathbf{L_g} \boldsymbol{\phi}\, \mathbf{L_g} \boldsymbol{\phi}^{\mathrm{T}}\right)\, \|\mathbf{z}\|_2^2\; u^+, \quad \text{where } \|\mathbf{z}\|_2 \geq 1\ \forall\, \boldsymbol{\phi} \neq \mathbf{0}_N, \tag{19}$$

because if vector $\boldsymbol{\phi}^{N-a}$ is not empty at least one component of vector $\mathbf{z}$ is equal to 1.

From (18) and (19), the upper bound of the time derivative of the Lyapunov function V results in:

$$\dot{V} \leq \|L_f \boldsymbol{\phi}\|_1 - \mathrm{eig}_{\min}\left(\mathbf{L_g} \boldsymbol{\phi}\, \mathbf{L_g} \boldsymbol{\phi}^{\mathrm{T}}\right)\, u^+. \tag{20}$$

Therefore, if u^+ fulfills (13) the Lyapunov function decays at a finite rate, it vanishes and collective SM in the intersection of the N constraints occurs after a finite time interval. □

It is important to remark that, using the transpose of matrix $\mathbf{L_g} \boldsymbol{\phi}$ for the SMC in (12) instead of its pseudoinverse represents a theoretical contribution of this approach. In fact, the transpose of the Jacobian matrix has already been used in robotics for continuous control laws due to its simplicity, see [37–40], among others. However, to the best of the authors knowledge, it has not been yet used for SMC, e.g., all the SMCs proposed in literature for robot VS [25–36] use the Jacobian pseudoinverse.

Using matrix inversion. Alternatively, instead of using the transpose of matrix $\mathbf{L_g} \boldsymbol{\phi}$ in (12), it could be also utilized the Moore-Penrose pseudoinverse [41], but at the expense of *higher computational load* and the condition that $\mathbf{L_g} \boldsymbol{\phi}$ must be now *full row rank*. In this case, the control law and the lower bound condition for u^+ result in:

$$\mathbf{u} = -\mathbf{L_g} \boldsymbol{\phi}^{\dagger}\, \mathrm{sign}(\boldsymbol{\phi})\, u^+ \tag{21}$$

$$u^+ > \|L_f \boldsymbol{\phi}\|_1, \tag{22}$$

DOI: 10.1002/rnc

where superscript † denotes the Moore-Penrose pseudoinverse

The control law given by (12), or alternatively (21), firstly makes the system converge to Φ in finite time, known as *reaching phase* [23]. And, subsequently, the control law will make $\mathbf{u}$ switch at a theoretically infinite frequency in order to keep the system on the so-called sliding surface Φ. This final phase is known as *sliding mode phase* [23]. Moreover, a *continuous equivalent control* [42] can be obtained for the SM phase, i.e., the control required to keep the system on the sliding surface. Hence, the SM generated by (12) or (21) produces such control action without explicit knowledge of it and with a low computational cost, which is a typical advantage of SM strategies [42].

3.2. Higher-order invariance

The above SMC produces a non-smooth $\mathbf{u}$. If a smooth control action is wished, the following approach can be used. Firstly, the initial constraint $\phi(\mathbf{x}) = 0$ is transformed to $\overline{\phi} = \phi(\mathbf{x}) + K\dot{\phi}(\mathbf{x}) = 0$. Thus, we obtain $\overline{\phi} = \phi(\mathbf{x}) + K\nabla\phi \cdot \dot{\mathbf{x}} = \phi(\mathbf{x}) + K\nabla\phi \cdot (\mathbf{f}(\mathbf{x}, \mathbf{d}) + \mathbf{g}(\mathbf{x})\mathbf{u})$. Hence, an augmented state is considered $\bar{\mathbf{x}}^{\mathrm{T}} = [\mathbf{x}^{\mathrm{T}}\ \mathbf{u}^{\mathrm{T}}]$, which includes the input $\mathbf{u}$, so that $\overline{\phi}$ is a function of the augmented state, i.e, $\overline{\phi}(\bar{\mathbf{x}}, \mathbf{d})$. Then, taking time derivatives[‡], we obtain $\dot{\overline{\phi}} = (\partial\overline{\phi}/\partial\bar{\mathbf{x}})^{\mathrm{T}}\dot{\bar{\mathbf{x}}} + (\partial\overline{\phi}/\partial\mathbf{d})^{\mathrm{T}}\dot{\mathbf{d}}$. Thus, since $\dot{\mathbf{u}}$ appears in $\dot{\bar{\mathbf{x}}}$, $\overline{\phi}$ is relative degree one in $\dot{\mathbf{u}}$, so considering $\dot{\mathbf{u}}$ as the "new" discontinuous input, the actual control $\mathbf{u}$ will be now smooth. Hence, the fulfillment of the new constraint $\overline{\phi} = 0$ gives rise to an exponential decrease of the original constraint ϕ towards zero: $\phi(t) = \phi(0)e^{-t/K}$, where K is a free design parameter to establish the rate of approach to the boundary of the original constraint.

Therefore, the proposed systematic procedure above, which is sequentially applied three times in next section, consists in increasing the constraint order and using first-order SMC. Alternatively, a different approach in the literature consists in using high-order SMC [43–45] so that the SM surface is not modified but its relative degree with respect to the discontinuous control action is increased to two or more. Then, the discontinuous control action is computed using complex expressions of the SM surface function and its derivatives, e.g., see the algorithms: twisting [46], super-twisting [47], sub-optimal [48], etc.

3.3. Order of the control action

In order to use the SMC above, the time-derivative of $\boldsymbol{\phi}$ must explicitly depend on the control action $\mathbf{u}$, see (14). That is, the sliding manifold must have *relative degree one* with respect to the control variable, as required by SMC theory [23]. Therefore, when the constraint function vector $\boldsymbol{\phi}$ is defined, the order of the corresponding discontinuous control action in (12) or (21) is also established. Nevertheless, if the order of the actual control action vector for the system at hand does not match the order of the mentioned discontinuous control action, a *filter* with the right order (or a set of integrators) can be used between both signals to meet the relative degree condition above.

For instance, if the constraint function depends on the joint positions and velocities, the discontinuous control action is an acceleration or second-order signal. For this example, if the actual control action is the joint velocity vector, a first-order filter[§] or a pure integrator has to be applied to the discontinuous control signal in order to compute the actual control action, which is continuous.

Note that in any case, the order of the actual control action must be equal to or lower than the order of the discontinuous control signal. If that would not be the case, the higher-order invariance described in Sec. 3.2 may be used to increase the order of the discontinuous control signal.

[‡]Note that to use this approach the original constraint function ϕ_i needs to be *twice* differentiable.
[§]If a filter is used, it has to be properly designed since it limits the bandwidth of the controlled system.

DOI: 10.1002/rnc

4. SLIDING MODE CONTROL FOR REFERENCE TRACKING

4.1. Procedure to use sliding mode control

This section gives an overview of the required steps to use SMC. This "recipe" will be used in the next subsections.

The first step consists in defining the equality constraint to be satisfied: $\boldsymbol{\phi} = \mathbf{0}$. Typically, this constraint is chosen to be an ordinary differential equation in order to obtain the desired dynamics for the controlled system. In particular, for reference tracking, this differential equation is expressed in terms of the error variable, i.e., the difference between the current value and the reference value for the controlled variable. Note that, when the order of the differential equation is is defined, the order of the corresponding discontinuous control action is also established, see Sec. 3.3. Therefore, the control design specification could be either the order of the differential equation or, alternatively, the order of the discontinuous control action.

The next step consists in obtaining the time-derivative of the constraint function vector $\boldsymbol{\phi}$ and identifying the Lie derivatives $L_f\boldsymbol{\phi}$ and $\mathbf{L_g}\boldsymbol{\phi}$ of the system at hand.

Finally, the control law given by (12), or alternatively (21) if the matrix inversion option is considered, has to be obtained.

4.2. Sliding mode control using joint velocities

This case considers the most simple equality constraint for reference tracking, which is straightforward obtained by rewriting (6) as:

$$\boldsymbol{\phi}_v(\mathbf{q}, t) = \mathbf{s}(\mathbf{q}, t) - \mathbf{s}_{ref}(t) = \mathbf{e} = \mathbf{0}, \tag{23}$$

where $\mathbf{e}$ represents the tracking position error of the visual feature vector $\mathbf{s}$.

Taking into account the first-order kinematics in (2), the time-derivative of the constraint function vector $\boldsymbol{\phi}_v$ is given by:

$$\dot{\boldsymbol{\phi}}_v = \dot{\mathbf{s}} - \dot{\mathbf{s}}_{ref} = \mathbf{J}_s\dot{\mathbf{q}} + \partial\mathbf{s}/\partial t - \dot{\mathbf{s}}_{ref} = \mathbf{J}_s\dot{\mathbf{q}} + \partial\mathbf{e}/\partial t, \tag{24}$$

where it can be noticed that it explicitly depends on the joint velocities. Hence, in order to satisfy the relative degree condition mentioned above, the control input vector $\mathbf{u}$ for this case is the commanded joint velocity $\dot{\mathbf{q}}_c$.

Therefore, from (24) and (7) the Lie derivatives in (14) for the constraint function vector $\boldsymbol{\phi}_v$ are given by:

$$\mathbf{L_g}\boldsymbol{\phi}_v = \mathbf{J}_s \tag{25}$$

$$L_f\boldsymbol{\phi}_v = \mathbf{J}_s\,\mathbf{d}_{cv} + \partial\mathbf{e}/\partial t. \tag{26}$$

Therefore, the control laws given by (12) and (21) result in:

$$\dot{\mathbf{q}}_c = -\mathbf{J}_s^{\mathrm{T}}\,\mathrm{sign}(\mathbf{e})\,u^+ \tag{27}$$

$$\dot{\mathbf{q}}_c = -\mathbf{J}_s^{\dagger}\,\mathrm{sign}(\mathbf{e})\,u^+. \tag{28}$$

It is interesting to note that the sliding surface given by (23) has already been used in VS for a SM velocity controller in [25, 28, 29, 32, 33].

4.3. Sliding mode control using joint accelarations

The above constraint $\boldsymbol{\phi}_v$ will be modified via the higher-order invariance described in Sec. 3.2 as follows:

$$\boldsymbol{\phi}_a(\mathbf{q}, \dot{\mathbf{q}}, t) = \boldsymbol{\phi}_v + K_a\dot{\boldsymbol{\phi}}_v = \mathbf{e} + K_a\dot{\mathbf{e}} = \mathbf{0}, \tag{29}$$

where K_a is a positive parameter that determines the time constant of the approach to $\boldsymbol{\phi}_v = \mathbf{0}$.

DOI: 10.1002/rnc

Taking into account the first- and second-order kinematics in (2) and (4), the time-derivative of the constraint function vector $\boldsymbol{\phi}_a$ is given by:

$$\dot{\boldsymbol{\phi}}_a = \dot{\mathbf{e}} + K_a\ddot{\mathbf{e}} = \dot{\mathbf{s}} - \dot{\mathbf{s}}_{ref} + K_a(\ddot{\mathbf{s}} - \ddot{\mathbf{s}}_{ref}) = \mathbf{J}_s\dot{\mathbf{q}} + \partial\mathbf{e}/\partial t + K_a(\mathbf{J}_s\ddot{\mathbf{q}} + \dot{\mathbf{J}}_s\dot{\mathbf{q}} + \partial\dot{\mathbf{e}}/\partial t), \tag{30}$$

where it can be noticed that it explicitly depends on the joint accelerations. Hence, in order to satisfy the relative degree condition mentioned above, the control input vector $\mathbf{u}$ for this case is the commanded joint acceleration $\ddot{\mathbf{q}}_c$.

Therefore, the main advantage of considering (29) instead of (23) is that the control is smoother: the joint velocity is continuous instead of discontinuous and the error equation is a first-order differential equation instead of an static equation.

From (30) and (8) the Lie derivatives in (14) for the constraint function vector $\boldsymbol{\phi}_a$ are given by:

$$\mathbf{L}_\mathbf{g}\boldsymbol{\phi}_a = K_a\mathbf{J}_s \tag{31}$$

$$L_f\boldsymbol{\phi}_a = (\mathbf{J}_s + K_a\dot{\mathbf{J}}_s)\dot{\mathbf{q}} + K_a\mathbf{J}_s\mathbf{d}_{ca} + \partial(\mathbf{e} + K_a\dot{\mathbf{e}})/\partial t. \tag{32}$$

Therefore, the control laws given by (12) and (21) result in:

$$\ddot{\mathbf{q}}_c = -K_a^{-1}\mathbf{J}_s^{\mathrm{T}}\,\mathrm{sign}(\mathbf{e} + K_a\dot{\mathbf{e}})\,u^+ \tag{33}$$

$$\ddot{\mathbf{q}}_c = -K_a^{-1}\mathbf{J}_s^{\dagger}\,\mathrm{sign}(\mathbf{e} + K_a\dot{\mathbf{e}})\,u^+. \tag{34}$$

4.4. Sliding mode control using joint jerks

As before, the constraint $\boldsymbol{\phi}_a$ will be modified via the higher-order invariance described in Sec. 3.2 as follows:

$$\boldsymbol{\phi}_j(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, t) = \boldsymbol{\phi}_a + K_j\dot{\boldsymbol{\phi}}_a = \mathbf{e} + K_a\dot{\mathbf{e}} + K_j\dot{\mathbf{e}} + K_aK_j\ddot{\mathbf{e}} = \mathbf{e} + K_{j1}\dot{\mathbf{e}} + K_{j2}\ddot{\mathbf{e}} = \mathbf{0}, \tag{35}$$

where K_j is a positive parameter that determines the time constant of the approach to $\boldsymbol{\phi}_a = \mathbf{0}$ and $K_{j1} = K_a + K_j$ and $K_{j2} = K_aK_j$ represent the coefficients of the differential equation above.

Taking into account (2), (4) and (5), the time-derivative of the constraint function vector $\boldsymbol{\phi}_j$ is given by:

$$\begin{aligned}\dot{\boldsymbol{\phi}}_j =& \dot{\mathbf{e}} + K_{j1}\ddot{\mathbf{e}} + K_{j2}\dddot{\mathbf{e}} = \dot{\mathbf{s}} - \dot{\mathbf{s}}_{ref} + K_{j1}(\ddot{\mathbf{s}} - \ddot{\mathbf{s}}_{ref}) + K_{j2}(\dddot{\mathbf{s}} - \dddot{\mathbf{s}}_{ref})\\ =& \mathbf{J}_s\dot{\mathbf{q}} + \partial\mathbf{e}/\partial t + K_{j1}(\mathbf{J}_s\ddot{\mathbf{q}} + \dot{\mathbf{J}}_s\dot{\mathbf{q}} + \partial\dot{\mathbf{e}}/\partial t) + K_{j2}(\mathbf{J}_s\dddot{\mathbf{q}} + 2\dot{\mathbf{J}}_s\ddot{\mathbf{q}} + \ddot{\mathbf{J}}_s\dot{\mathbf{q}} + \partial\ddot{\mathbf{e}}/\partial t),\end{aligned} \tag{36}$$

where it can be noticed that it explicitly depends on the joint jerks. Hence, in order to satisfy the relative degree condition mentioned above, the control input vector $\mathbf{u}$ for this case is the commanded joint jerk $\dddot{\mathbf{q}}_c$.

Therefore, as before, the main advantage of considering (35) instead of (29) is that the control is smoother: the joint acceleration is continuous instead of discontinuous and the error differential equation is of second-order instead of first-order.

From (36) and (9) the Lie derivatives in (14) for the constraint function vector $\boldsymbol{\phi}_j$ are given by:

$$\mathbf{L}_\mathbf{g}\boldsymbol{\phi}_j = K_{j2}\mathbf{J}_s \tag{37}$$

$$L_f\boldsymbol{\phi}_j = (\mathbf{J}_s + K_{j1}\dot{\mathbf{J}}_s + K_{j2}\ddot{\mathbf{J}}_s)\dot{\mathbf{q}} + (K_{j1}\mathbf{J}_s + 2K_{j2}\dot{\mathbf{J}}_s)\ddot{\mathbf{q}} + K_{j2}\mathbf{J}_s\mathbf{d}_{cj} + \partial(\mathbf{e} + K_{j1}\dot{\mathbf{e}} + K_{j2}\ddot{\mathbf{e}})/\partial t. \tag{38}$$

Therefore, the control laws given by (12) and (21) result in:

$$\dddot{\mathbf{q}}_c = -K_{j2}^{-1}\mathbf{J}_s^{\mathrm{T}}\,\mathrm{sign}(\mathbf{e} + K_{j1}\dot{\mathbf{e}} + K_{j2}\ddot{\mathbf{e}})\,u^+ \tag{39}$$

$$\dddot{\mathbf{q}}_c = -K_{j2}^{-1}\mathbf{J}_s^{\dagger}\,\mathrm{sign}(\mathbf{e} + K_{j1}\dot{\mathbf{e}} + K_{j2}\ddot{\mathbf{e}})\,u^+. \tag{40}$$

DOI: 10.1002/rnc

4.5. Chattering

Discrete-time implementations of any practical SMC makes the system leave the ideal SM and oscillate with finite frequency and amplitude inside a band around $\boldsymbol{\phi} = \mathbf{0}$, namely *chattering* [23]. The chattering band $\triangle\boldsymbol{\phi}$ of the proposal can be obtained using the Euler-integration of the discontinuous control action given by (21), that is:

$$\triangle\boldsymbol{\phi} = T_s \, |\mathbf{L_g}\boldsymbol{\phi} \, \mathbf{u}| = T_s \, u^+ \, \mathbf{1}, \tag{41}$$

where T_s is the sampling time of the robot control and $\mathbf{1}$ is the column vector with all its components equal to one.

5. COMPARISON WITH CLASSICAL CONTINUOUS CONTROL

5.1. Classical continuous control using joint velocities

Substituting the first-order kinematics of the robot system (2) and the low-level control equation (7) in the first-order differential equation of the error given by (29), it is obtained the following equation:

$$\mathbf{e} + K_a\dot{\mathbf{e}} = \mathbf{e} + K_a(\mathbf{J}_s\dot{\mathbf{q}} + \partial\mathbf{s}/\partial t - \dot{\mathbf{s}}_{ref}) = \mathbf{e} + K_a(\mathbf{J}_s(\dot{\mathbf{q}}_c + \mathbf{d}_{cv}) + \partial\mathbf{e}/\partial t) = \mathbf{0}, \tag{42}$$

and the commanded joint velocity vector results in:

$$\dot{\mathbf{q}}_c = -\mathbf{J}_s^{\dagger}(K_a^{-1}\mathbf{e} + \partial\mathbf{e}/\partial t) - \mathbf{d}_{cv}, \tag{43}$$

which is the most typical control law used in VS [2] in order to obtain an exponential decrease of the tracking error.

The partial derivative $\partial\mathbf{e}/\partial t$ is typically estimated (see [2]) using the first-order kinematics given by (2), yielding:

$$\partial\mathbf{e}/\partial t = \dot{\mathbf{e}} - \mathbf{J}_s\dot{\mathbf{q}}, \tag{44}$$

and, hence, the time-derivative of the error vector $\dot{\mathbf{e}}$ is also needed like in the SMC given by (33)–(34).

5.2. Classical continuous control using joint accelarations

Substituting the second-order kinematics of the robot system (4) and the low-level control equation (8) in the second-order differential equation of the error given by (35), it is obtained the following equation:

$$\mathbf{e} + K_{j1}\dot{\mathbf{e}} + K_{j2}\ddot{\mathbf{e}} = \mathbf{e} + K_{j1}\dot{\mathbf{e}} + K_{j2}(\mathbf{J}_s(\ddot{\mathbf{q}}_c + \mathbf{d}_{ca}) + \dot{\mathbf{J}}_s\dot{\mathbf{q}} + \partial\dot{\mathbf{e}}/\partial t) = \mathbf{0}, \tag{45}$$

and the commanded joint acceleration vector results in:

$$\ddot{\mathbf{q}}_c = -\mathbf{J}_s^{\dagger}(K_{j2}^{-1}\mathbf{e} + K_{j2}^{-1}K_{j1}\dot{\mathbf{e}} + \dot{\mathbf{J}}_s\dot{\mathbf{q}} + \partial\dot{\mathbf{e}}/\partial t) - \mathbf{d}_{ca}, \tag{46}$$

which represents the classical operational space robot control [49] that has already been used in VS appliactions by [21] for PBVS and by [22] for IBVS.

As above, the partial derivative $\partial\dot{\mathbf{e}}/\partial t$ can be estimated using the second-order kinematics given by (4), yielding:

$$\partial\dot{\mathbf{e}}/\partial t = \ddot{\mathbf{e}} - \mathbf{J}_s\ddot{\mathbf{q}} - \dot{\mathbf{J}}_s\dot{\mathbf{q}}, \tag{47}$$

and, hence, the second-order time-derivative of the error vector $\ddot{\mathbf{e}}$ is also needed like in the SMC given by (39)–(40).

DOI: 10.1002/rnc

5.3. *Equivalences between sliding mode control and classical continuous control*

Assuming that the SMC is in the SM phase, i.e., the reaching phase has finished and the system on the sliding surface, the SMCs proposed in Sec. 4.3 and Sec. 4.4 are equivalent to the classical velocity and acceleration controls described in Sec. 5.1 and Sec. 5.2, respectively, in the sense that both approaches give rise to the same first-order or second-order differential equation for the tracking error. In particular, if the SMC is in the SM phase and considering the same initial conditions, both the SMC and its continuous equivalent give rise to the same value for the joint velocities or joint accelerations, as will be shown in the simulations of Sec. 9 and Sec. 10.

However, despite the mentioned equivalences, the proposed SM strategy presents several significant advantages over its classical continuous counterpart that are discussed below.

6. ADVANTAGES OF THE METHOD

6.1. *Valid both for PBVS and IBVS*

In contrast to some SMCs proposed in literature for VS [29–36], the proposed SMC given by (33)–(34) or (39)–(40) is valid either if the visual feature vector $\mathbf{s}$ is defined in PBVS or IBVS domain. Obviously, each case yields a specific Jacobian matrix $\mathbf{J}_s$ to be used in (33)–(34) and (39)–(40) .

6.2. *Smoothness*

In contrast to the SMCs proposed in literature for VS [25–36], see Sec. 1, the proposed method yields *continuous joint velocities* given that the SM discontinuous control action are joint accelerations or joint jerks.

6.3. *Robustness*

Instead of using a SM discontinuous control action to enforce $\dot{\boldsymbol{\phi}}_a = \mathbf{0}$ or $\dot{\boldsymbol{\phi}}_j = \mathbf{0}$, in order to keep the system on the sliding surface, the *analytic computation* of $\ddot{\mathbf{q}}_c$ or $\dddot{\mathbf{q}}_c$, respectively, could be obtained solving (30) or (36), respectively. However, the accurate computation of these continuous control actions requires a *perfect knowledge* of the system model: Jacobian matrix $\mathbf{J}_s$ and its derivatives, partial derivative of the error vector $\partial\mathbf{e}/\partial t$ and its derivatives, joint velocities $\dot{\mathbf{q}}$, inaccuracies $\{\mathbf{d}_{ca}, \mathbf{d}_{cj}\}$ of the low-level control loop, etc. The same applies to the classical continuous control given by (43) and (46).

For instance, if the disturbances $\{\mathbf{d}_{ca}, \mathbf{d}_{cj}\}$ and/or the partial derivatives $\{\partial\mathbf{e}/\partial t, \partial\dot{\mathbf{e}}/\partial t\}$ given by a moving target are not known a priori, as common in practice, the accurate computation of the mentioned continuous control actions is not possible.

In contrast, the proposed SMC is *robust* [23] against $\{\mathbf{d}_{ca}, \mathbf{d}_{cj}\}$ and $\{\partial\mathbf{e}/\partial t, \partial\dot{\mathbf{e}}/\partial t\}$ since they are collinear with the discontinuous control action, see (8) and (9). The same applies to the remaining terms collinear with the discontinuous control action: time-derivative of the Jacobian matrix, etc.

Even more, although the Jacobian matrix $\mathbf{J}_s$ used in the proposed SMC is not collinear with the discontinuous control action, see (33)–(34) and (39)–(40), a non-accurate value of this matrix can be used as long as it provides a component perpendicular to the sliding surface given by $\boldsymbol{\phi} = \mathbf{0}$ in order for the SM control action to be able to switch the value of the constraint functions ϕ_i from positive to negative or vice versa.

It is worth to mention that, the proposed SMC is not robust against the error signal $\mathbf{e}$ (i.e., $\mathbf{s}$) and its derivatives since they are used to define the sliding surface in the switching control law, see (33)–(34) and (39)–(40). Obviously, the continuous equivalent control is neither robust against this error.

The robustness feature of the proposed SMC is illustrated in the simulation of Sec. 10.3 and in the experimental results of Sec. 11.3.

DOI: 10.1002/rnc

6.4. Low computational cost

The proposed SMC only requires to compute the Jacobian matrix $\mathbf{J}_s$ and the constraint function vector $\boldsymbol{\phi}$. In contrast, the classical continuous control given by (43) and (46) require to compute: the partial derivative $\partial\mathbf{e}/\partial t$ due to a moving target and its derivatives; the inaccuracies $\{\mathbf{d}_{ca}, \mathbf{d}_{cj}\}$ of the low-level control loop; the time derivative of the Jacobian matrix; etc. Therefore, the computational cost is reduced.

It is interesting to remark that, the partial derivative $\partial\mathbf{e}/\partial t$ due to a moving target is also used as a feedforward term by some SMCs proposed in literature for VS [25, 28–33], while the proposed SMC does not require any kind of feedforward, as commented above.

Note also that, in contrast to the classical continuous control, the inversion of the Jacobian matrix is not mandatory for the proposed SMC, since the transpose of the Jacobian matrix $\mathbf{J}_s^{\mathrm{T}}$ can also be used, as mentioned above. Therefore, the computational cost can be further reduced.

6.5. Sliding mode control using joint jerks

If the joint jerks are used for the proposed SMC instead of the joint accelerations, i.e., Eq. (39)–(40) instead of Eq. (33)–(34), two main advantages are obtained: the joint velocities are smoother, i.e., they are C^1 instead of C^0; and the error differential equation has one more degree-of-freedom, i.e., there are two poles to be assigned instead of one. However, practical implementations for this case may be affected if the sampling time is not small enough or significant measurement noise is present.

7. ADDITIONAL REMARKS

7.1. Time derivatives

The proposed approach requires the time derivatives of the error signal: $\dot{\mathbf{e}}$ (i.e., $\dot{\mathbf{s}}$) for the SMC using joint accelerations, see (33)–(34); and $\{\dot{\mathbf{e}}, \ddot{\mathbf{e}}\}$ (i.e., $\{\dot{\mathbf{s}}, \ddot{\mathbf{s}}\}$) for the SMC using joint jerks, see (39)–(40). However, this situation is not new in VS applications: the classical continuous controls given by (43)–(44) and (46)–(47) require the same time derivatives, together with other time derivatives (i.e., $\dot{\mathbf{q}}$ and $\dot{\mathbf{J}}_s$) not needed in the proposed SMC. As in many other applications, besides the use of advanced sensors (tachometers, accelerometers, etc.), the simplest way to deal with this issue consists in using numerical differentiation, e.g., the well-known backward Euler approximation. However, some kind of filtering should be previously applied to the actual variable when non-negligible noise is present. It is important to remark that the low-pass filter used for noise reduction must not limit the bandwidth of the control law. That is, the bandwidth of the control law should not exceed the bandwidth of the low-pass filter. In particular, for the proposed SMC, the theoretical frequency of the control law signal is equal to $(2T_s)^{-1}$ Hertz and, hence, the filter attenuation at this frequency should be relatively small.

7.2. Jacobian singularities

Concerning the Jacobian pseudoinverse used by the proposed SMCs in (34) and (40), it is recalled that this inversion gives rise to numerical problems when the determinant of the Jacobian vanishes, which occurs at singular points. Several approaches are discussed below to overcome this issue:

- The simplest option consists in using the Jacobian transpose $\mathbf{J}_s^{\mathrm{T}}$ instead of the the Jacobian pseudoinverse $\mathbf{J}_s^{\dagger}$, i.e., using {(33),(39)} instead of {(34),(40)}. The main advantage of this approach is the low computational cost, while its main drawback is that the lower bound for the switching gain u^+ is increased, see (13) and (22).
- Another alternative consists in using a classical type of matrix regularization: the damped least-squares (DLS) solution [50], which minimizes the square norm of the equation error together with the square norm of the solution weighted by a nonnegative damping factor λ. In

particular, the regularized Moore-Penrose pseudoinverse of matrix $\mathbf{J}_s$ using DLS results in:

$$\mathbf{J}_s^\sharp = \mathbf{J}_s^{\mathrm{T}}(\mathbf{J}_s\mathbf{J}_s^{\mathrm{T}} + \lambda^2\mathbf{I})^{-1}, \tag{48}$$

where $\mathbf{I}$ denotes the identity matrix of suitable size and superscript $\sharp$ denotes the so-called regularized or singularity-robust pseudoinverse. The main drawback of this method is that the direction of the Jacobian pseudoinverse is significantly modified around singular configurations.

- Another type of matrix regularization can be considered to preserve the direction of the Jacobian pseudoinverse but saturating its magnitude as follows:

$$\mathbf{J}_s^\sharp = \mathbf{J}_s^{\mathrm{T}}\,\mathrm{adjoint}(\mathbf{J}_s\mathbf{J}_s^{\mathrm{T}})\,\mathrm{sat}((\det(\mathbf{J}_s\mathbf{J}_s^{\mathrm{T}}))^{-1}, \epsilon), \tag{49}$$

where $\det(\cdot)$ is the determinant function and function $\mathrm{sat}(\cdot)$ is used to saturate the value of $(\det(\mathbf{J}_s\mathbf{J}_s^{\mathrm{T}}))^{-1}$ to $\pm\epsilon$, where ϵ represents a threshold to avoid extremely large values.

- Finally, the pseudoinverse can be computed via the singular value decomposition (SVD) method [41] and using a tolerance to set to zero the very small singular values in order to avoid extremely large values for the commanded variable. The main drawback of this approach compared to the previous options is the computational cost required to obtain the SVD.

7.3. Switching gain selection

The selection of the switching gain u^+ is a common issue in SMC applications. A number of options for this purpose are discussed below.

Firstly, a *big number* could be chosen for u^+ in order to ensure that it is greater than the lower bound given by (13) or (22), as usual in SMC applications. However, such big numbers may induce unnecessary chattering amplitude, see (41). Therefore, in order to avoid this drawback, a second option consists in *estimating* the mentioned lower bound and choosing a value slightly larger than the estimated one. However, since this lower bound depends on the system state, as usual in SMC applications, it may be difficult to a priori estimate it. Thus, many practical applications use a third option that consists in running the system application (either in simulation or experimentally) in order to *empirically tune* a proper value for u^+ so that the SM behavior is fulfilled at all times and the chattering amplitude is minimized.

Finally, taking into account that the switching gain u^+ can be varying in time, a fourth option consists in using an *adaptive switching gain* $u^+(t)$ on the basis that different parts of the system trajectory may require very different SM control actions. That is, the lower bound for u^+ may drastically change from one part of the trajectory to another, e.g., it is typically larger when the system trajectory executes abrupt maneuvers. The main advantage of this adaptive approach is that the chattering amplitude is minimized online according to the current part of the trajectory. Examples of adaptive algorithms for the switching gain can be found in [51–56], among others. In any case, this is out of the scope of this research and remains as further work.

7.4. Adaptation of the algorithm for dynamic robot control

The proposed SMC has been designed considering the existence of the underlying robot control described in Sec. 2. However, if the robot can be controlled directly commanding the joint torques, which is not the case of most industrial robots (e.g., the one used in the real experimentation of Sec. 11), the proposed SMC can be adapted as detailed below to perform the dynamic control of the robot.

The joint space dynamic model of the robot is typically given by [49]:

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{H}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{F}_s(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) + \mathbf{J}_s^{\mathrm{T}}(\mathbf{q})\mathbf{F}_{ee}, \tag{50}$$

where $\boldsymbol{\tau}$ represents the vector of joint torques, $\mathbf{M}$ the inertia matrix, $\mathbf{H}\dot{\mathbf{q}}$ the centrifugal and Coriolis torques, $\mathbf{F}_v$ a diagonal matrix with the viscous friction coefficients, $\mathbf{F}_s$ the static friction torques, $\mathbf{G}$ the torques generated by the links weights due to gravity and $\mathbf{F}_{ee}$ the vector of forces and moments exerted by the robot end-effector on the environment.

DOI: 10.1002/rnc

The SMC described in Sec. 4.3 can be modified as follows to consider the joint torques as the discontinuous control action instead of the joint accelerations. Solving for $\ddot{\mathbf{q}}$ in (50) and substituting into (30) yields the following modified Lie derivatives for the constraint function vector $\boldsymbol{\phi}_a$:

$$\overline{\mathbf{L_g}\boldsymbol{\phi}_a} = \mathbf{L_g}\boldsymbol{\phi}_a\mathbf{M}^{-1} \tag{51}$$

$$\overline{L_f\boldsymbol{\phi}_a} = L_f\boldsymbol{\phi}_a - K_a\mathbf{J}_s\mathbf{M}^{-1}(\mathbf{M}\dot{\mathbf{d}}_{ca} + \mathbf{H}\dot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{F}_s + \mathbf{G} + \mathbf{J}_s^{\mathrm{T}}\mathbf{F}_{ee}), \tag{52}$$

and, hence, the modified switching control law (21) results in:

$$\boldsymbol{\tau} = -K_a^{-1}\mathbf{M}\mathbf{J}_s^{\dagger}\,\mathrm{sign}(\mathbf{e} + K_a\dot{\mathbf{e}})\,u^{+}. \tag{53}$$

Similarly, the SMC described in Sec. 4.4 can be modified as follows to consider the time derivative of the joint torque vector as the discontinuous control action instead of the joint jerks. Taking the time derivative of (50) and solving for $\dddot{\mathbf{q}}$ results in:

$$\dddot{\mathbf{q}} = \mathbf{M}^{-1}(\dot{\boldsymbol{\tau}} - (\dot{\mathbf{M}} + \mathbf{H} + \mathbf{F}_v)\ddot{\mathbf{q}} - \dot{\mathbf{H}}\dot{\mathbf{q}} - \dot{\mathbf{F}}_s - \dot{\mathbf{G}} - \dot{\mathbf{J}}_s^{\mathrm{T}}\mathbf{F}_{ee} - \mathbf{J}_s^{\mathrm{T}}\dot{\mathbf{F}}_{ee}), \tag{54}$$

and substituting this result into (36) yields the following modified Lie derivatives for the constraint function vector $\boldsymbol{\phi}_j$:

$$\overline{\mathbf{L_g}\boldsymbol{\phi}_j} = \mathbf{L_g}\boldsymbol{\phi}_j\mathbf{M}^{-1} \tag{55}$$

$$\overline{L_f\boldsymbol{\phi}_j} = L_f\boldsymbol{\phi}_j - K_{j2}\mathbf{J}_s\mathbf{M}^{-1}(\mathbf{M}\dot{\mathbf{d}}_{cj} + (\dot{\mathbf{M}} + \mathbf{H} + \mathbf{F}_v)\ddot{\mathbf{q}} + \dot{\mathbf{H}}\dot{\mathbf{q}} + \dot{\mathbf{F}}_s + \dot{\mathbf{G}} + \dot{\mathbf{J}}_s^{\mathrm{T}}\mathbf{F}_{ee} + \mathbf{J}_s^{\mathrm{T}}\dot{\mathbf{F}}_{ee}), \tag{56}$$

and, hence, the modified switching control law (21) results in:

$$\dot{\boldsymbol{\tau}} = -K_{j2}^{-1}\mathbf{M}\mathbf{J}_s^{\dagger}\,\mathrm{sign}(\mathbf{e} + K_{j1}\dot{\mathbf{e}} + K_{j2}\ddot{\mathbf{e}})\,u^{+}. \tag{57}$$

Note that, as explained in Sec. 3.3, the discontinuous control action $\dot{\boldsymbol{\tau}}$ in (57) has to be integrated in order to obtain the actual control action of the robot, i.e., the torque vector $\boldsymbol{\tau}$.

Therefore, by comparing (53) with (34) and (57) with (40), it is concluded that only the inertia matrix $\mathbf{M}$ is required to use the proposed SMC for the dynamic control of the robot. Moreover, this SMC is robust (see Sec. 6.3) against the remaining terms of the robot dynamic model (centrifugal/Coriollis torques, viscous and static friction, gravity force, external forces) and its derivatives, since they are included in the Lie derivatives $\overline{L_f\boldsymbol{\phi}_a}$ and $\overline{L_f\boldsymbol{\phi}_j}$, i.e., they are collinear with the discontinuous control action.

8. CONDITIONS FOR THE SIMULATIONS AND EXPERIMENTS

The proposed approach can be used either for PBVS or IBVS. However, the simulations and experiments are focused on PBVS, since it is less robust to calibration and modeling errors than IBVS [2, 57, 58] and would get more benefit from the proposed SMC. Furthermore, the simulations and experiments have been developed for the eye-in-hand configuration, i.e., camera rigidly attached to the robot end-effector, although the method can also be used for eye-to-hand configuration (camera does not move with the robot).

The typical visual feature vector used in PBVS [2] is considered in this work:

$$\mathbf{s} = \begin{bmatrix} {}^{C^*}\mathbf{t}_C^{\mathrm{T}} & {}^{C^*}\theta\mathbf{u}_C^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \tag{58}$$

where the first element represents a translation vector and the second element gives the angle parameterization for the rotation (see [2] for further details), both between the desired camera pose and current camera pose.

For both simulations and experiments, it is used a classical 6R serial robot with spherical wrist: the Kuka KR6 R900 sixx manipulator, coined as Agilus. The robot is ceiling-mounted and its Jacobian

DOI: 10.1002/rnc

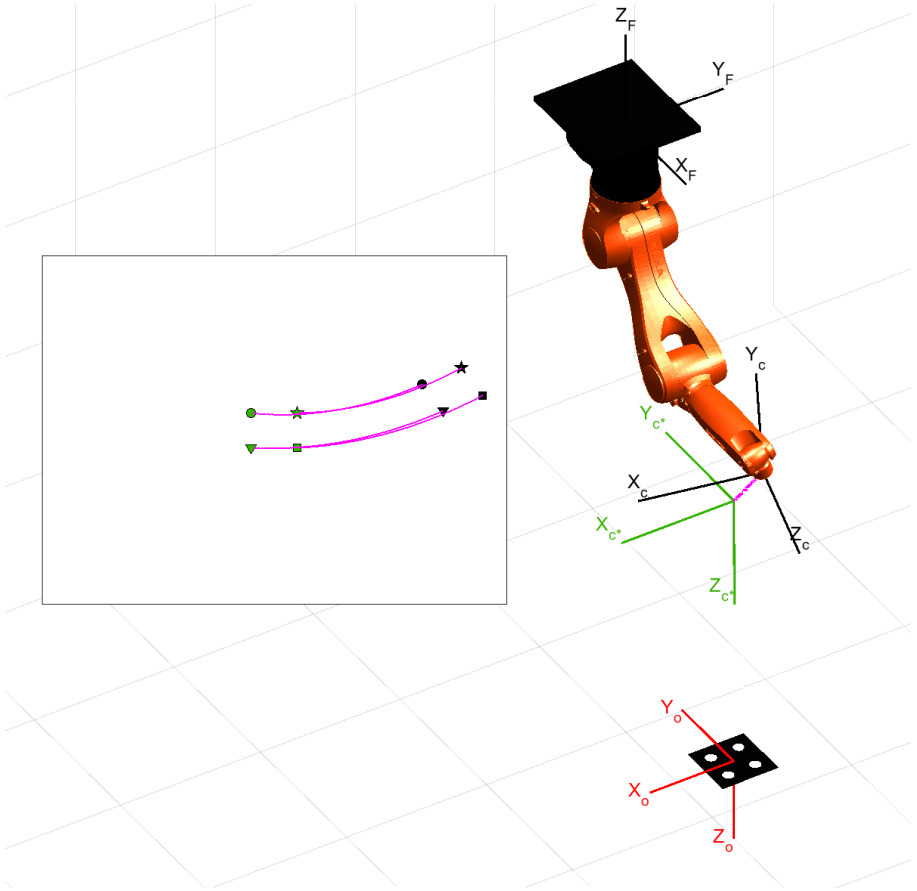

Figure 2. Positioning task: 6R robot, target object with four markers, involved coordinate frames, and resulting 3D trajectory of the camera (magenta-continuous). Image plane on the left-hand side: initial position (black-dark), desired position (green-light), and trajectory of the features (magenta-continuous). For clarity, only the trajectories for the SMC using joints acceleration and $\mathbf{J}_s^{\dagger}$ are depicted.

matrix ${}^{e}\mathbf{J}_{e}$ can be readily obtained [49] from the Denavit-Hartenberg (DH) parameters in Table I. Moreover, since the camera is rigidly attached to the end-effector of the robot, the twist matrix ${}^{c}\mathbf{V}_{e}$ is constant and can be computed from the camera to end-effector transformation matrix ${}^{c}\mathbf{M}_{e}$.

For both simulations and experiments presented below, the following two tasks are considered to demonstrate the general effectiveness and robustness of the method: (1) a *positioning task* consisting in moving the robot from the initial to the goal position, defined with respect to a still target object; (2) a *tracking task* consisting in moving the robot to follow a 3D trajectory defined by the object motion.

For the description of the parameters, let ${}^{S_2}\mathbf{M}_{S_1} = \begin{bmatrix} x & y & z & \alpha & \gamma & \theta \end{bmatrix}^{\mathrm{T}}$ be the compact notation adopted for detailing the values of the homogeneous transformation matrix from frame S_2 to frame S_1, where x, y and z are the Cartesian coordinates in meters, and α, γ and θ are the roll, pitch and roll angles, respectively, in radians. The simulation results presented below were obtained using MATLAB®.

9. SIMULATED POSITIONING TASK

Fig. 2 depicts the VS application in consideration for the simulated positioning task with the following elements: 6R robot, target object, as well as the involved frames: robot base frame F, object frame O, initial camera frame C and desired camera frame C^*.

DOI: 10.1002/rnc

9.1. *Conditions for the simulated positioning task*

Simulation for the positioning task was run under the following conditions:

i) Parameters used for the camera: focal lengths $f_x = 640$ and $f_y = 480$ pixels; principal point $[u_0, v_0] = [320, 240]$ pixels; and camera to end-effector transformation matrix ${}^c\mathbf{M}_e = \mathbf{I}_4$ where $\mathbf{I}_4$ represents the identity matrix of dimension 4, i.e., the camera pose is equivalent to the end-effector pose.
ii) Coefficients for the error differential equation: $K_a = 5$ for the first-order differential equation; $K_{j2} = 5$ and $K_{j1} = 3\sqrt{K_{j2}}$ (i.e., a value of 1.5 for the damping ration) for the second-order differential equation.
iii) The initial configuration considered for the robot is given by the joint position vector $\mathbf{q}(0) = \begin{bmatrix} 0 & -0.82 & 0.79 & -0.35 & -1.57 & -2.09 \end{bmatrix}^{\mathrm{T}}$ rad, yielding an initial camera pose given by the transformation matrix ${}^F\mathbf{M}_C(0) = \begin{bmatrix} 0.755 & 0.027 & 0.564 & -2.7923 & -0.0250 & 2.1038 \end{bmatrix}^{\mathrm{T}}$ in compact notation.
iv) A *static* target object is considered with the pose given by the transformation matrix ${}^F\mathbf{M}_O = \begin{bmatrix} 0.755 & 0.027 & 0.564 & \pi & 0 & \pi/2 \end{bmatrix}^{\mathrm{T}}$ in compact notation and with four markers given by the following points with respect to the object frame: ${}^O\mathbf{p}_1 = \begin{bmatrix} -0.05 & -0.05 & 0 \end{bmatrix}^{\mathrm{T}}$ m, ${}^O\mathbf{p}_2 = \begin{bmatrix} 0.05 & -0.05 & 0 \end{bmatrix}^{\mathrm{T}}$ m, ${}^O\mathbf{p}_3 = \begin{bmatrix} 0.05 & 0.05 & 0 \end{bmatrix}^{\mathrm{T}}$ m, ${}^O\mathbf{p}_4 = \begin{bmatrix} -0.05 & 0.05 & 0 \end{bmatrix}^{\mathrm{T}}$ m, that is, the four markers are the vertices of a square with a side length of 0.1 m.
v) The desired camera pose is given by the transformation matrix ${}^F\mathbf{M}_O = \begin{bmatrix} 0.624 & 0.001 & 0.377 & \pi & 0 & \pi/2 \end{bmatrix}^{\mathrm{T}}$ in compact notation.
vi) Switching gain u^+ for the SMC: $u^+ = 1$ when $\mathbf{J}_s^\dagger$ is used and $u^+ = 3$ when $\mathbf{J}_s^{\mathrm{T}}$ is used.
vii) For comparison purposes between the proposed SMC and the continuous equivalent, the initial value for the joint velocities and joint accelerations is chosen so that the system starts on the sliding surface Φ and, hence, the reaching phase is not present, see Sec. 5.3. In particular, in order to satisfy (29) when joint accelerations are used, the initial value for the joint velocities is computed as $\dot{\mathbf{q}}(0) = -\mathbf{J}_s^\dagger \dot{\mathbf{e}}(0)$ with $\dot{\mathbf{e}}(0) = -K_{T,p}\mathbf{e}(0)$. Similarly, in order to satisfy (35) when joint jerks are used, the initial value for the joint velocities is set to zero $\dot{\mathbf{q}}(0) = 0$, i.e., $\dot{\mathbf{e}}(0) = 0$, and the initial value for the joint accelerations is computed as $\ddot{\mathbf{q}}(0) = -\mathbf{J}_s^\dagger \ddot{\mathbf{e}}(0)$ with $\ddot{\mathbf{e}}(0) = -K_{T,p}\mathbf{e}(0)$.
viii) The simulation was carried out with a sampling time T_s of 1 millisecond.

9.2. *Simulation results for the positining task*

The results of the simulation are depicted at different figures. In particular, Fig. 2 shows that, as expected in PBVS approaches, the resulting camera trajectory in the 3D space is a straight line from the initial to the desired camera pose, whilst the trajectory of the features in the image plane describe a non-straight line. For clarity, only the trajectories for SMC using accelerations and $\mathbf{J}_s^\dagger$ are depicted. The differences between all the approaches is shown in the following figures.

Fig. 3 and Fig. 4 show the position error $\mathbf{e}$, the integral of the control action (i.e., $\dot{\mathbf{q}}$ or $\ddot{\mathbf{q}}$) and the constraint function vector (i.e., $\boldsymbol{\phi}_a$ or $\boldsymbol{\phi}_j$) for the SMC using joint accelerations and joint jerks, respectively. Note that the behavior of the position errors and joint velocities or accelerations is very similar when using $\mathbf{J}_s^\dagger$ and $\mathbf{J}_s^{\mathrm{T}}$, whereas the difference in the chattering band (see the constraint function) is due to the value used in each case for the switching gain u^+.

Fig. 5 compares the SMCs to their respective continuous equivalents in terms of joint speeds or joint accelerations, i.e., $\dot{\mathbf{q}}$ or $\ddot{\mathbf{q}}$, and in terms of the Euclidean norm (in the sequel, unless stated otherwise, it will be assumed the Euclidean norm) of the tracking error, i.e., $\|\mathbf{e}\|$. The differences are similar for the SMC using joint accelerations and joint jerks, and the norm of the tracking error increases around one order of magnitude when the transpose of the Jacobian matrix is used instead of the pseudoinverse.

DOI: 10.1002/rnc

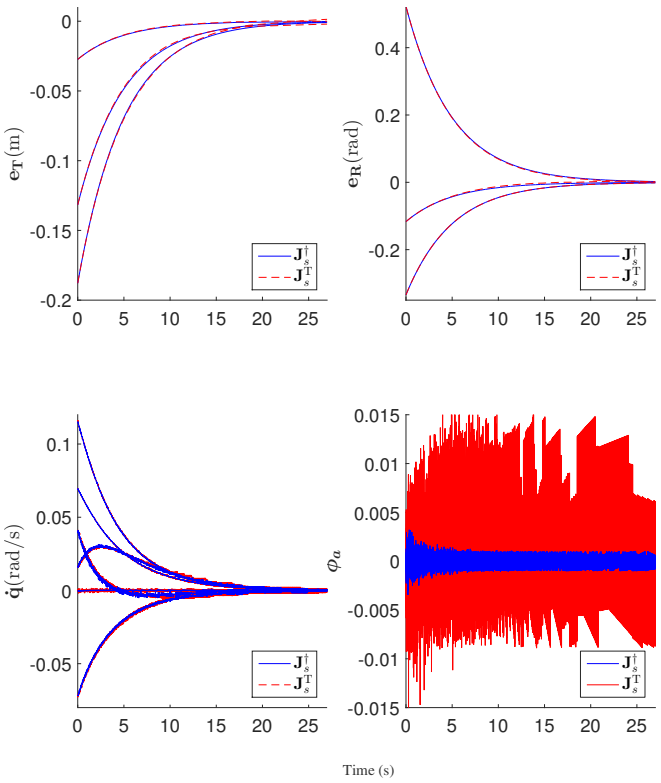

Figure 3. Simulated positioning task. SMC using joint accelerations. From top to bottom and left to right plots: (1) translation errors $\mathbf{e_T}$; (2) rotation errors $\mathbf{e_R}$; (3) joint speeds $\dot{\mathbf{q}}$; (4) Constraint function vector $\boldsymbol{\phi}_a$.

A video of this simulated positioning task for the SMC using joint accelerations and $\mathbf{J}_s^\dagger$ (the other cases are very similar) can be played at https://media.upv.es/player/?id=4f000e20-782b-11e7-90ea-23686ce0f1be.

10. SIMULATED TRACKING TASK

As above, Fig. 6 depicts the VS tracking application in consideration for the simulated tracking task.

10.1. Conditions for the simulated tracking task

Simulation was run under the same conditions as the positioning task except for the following:

i) Coefficients for the error differential equation: $K_a = 0.4$ for the first-order differential equation; $K_{j2} = 0.4$ and $K_{j1} = 3\sqrt{K_{j2}}$ for the second-order differential equation.
ii) The initial configuration considered for the robot is given by the joint position vector $\mathbf{q}(0) = \begin{bmatrix} 0 & 0.17 & -1.22 & 0 & -0.52 & -1.57 \end{bmatrix}^{\mathrm{T}}$ rad, yielding an initial camera pose given by the transformation matrix ${}^F\mathbf{M}_C(0) = \begin{bmatrix} 0.652 & 0 & 0.63 & \pi & 0 & \pi/2 \end{bmatrix}^{\mathrm{T}}$ in compact notation.
iii) The desired camera pose with respect to the object is given by the transformation matrix ${}^{C^*}\mathbf{M}_O = \begin{bmatrix} 0 & 0 & 0.5 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$ in compact notation, i.e., the camera must follow the target trajectory keeping a distance in the ${}^C Z$ axis equal to 0.5 meters.
iv) Switching gain u^+ for the SMC: $u^+ = 0.5$ when $\mathbf{J}_s^\dagger$ is used and $u^+ = 2$ when $\mathbf{J}_s^{\mathrm{T}}$ is used.
v) A *moving* target object is considered with four markers as described in previous section and with the following transformation matrix ${}^F\mathbf{M}_O(t) =$

DOI: 10.1002/rnc

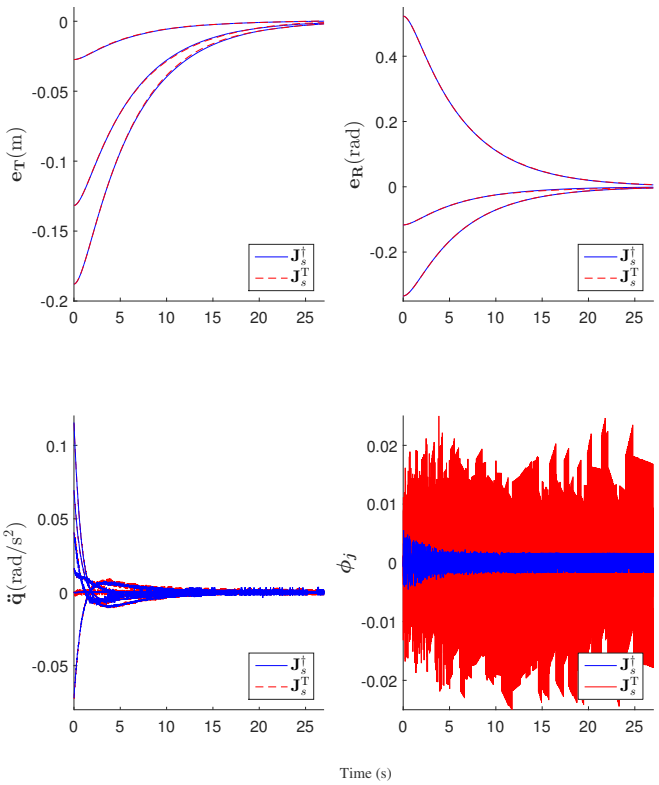

Figure 4. Simulated positioning task. SMC using joint jerks. From top to bottom and left to right plots: (1) translation errors $\mathbf{e_T}$; (2) rotation errors $\mathbf{e_R}$; (3) joint accelerations $\ddot{\mathbf{q}}$; (4) Constraint function vector $\boldsymbol{\phi}_a$.

$\begin{bmatrix}{}^F x_O & {}^F y_O & {}^F z_O & {}^F \alpha_O & {}^F \beta_O & {}^F \gamma_O\end{bmatrix}^{\mathrm{T}}$ in compact notation, where ${}^F x_O = 0.542 + 0.1\cos(t) + 0.01t + e^{-t}(0.1 - 0.09\cos(t))$, ${}^F y_O = -0.15(\sin(t) + e^{-t}\cos(t) - 1)$, ${}^F z_O = 0.105 + (t + e^{-t}\cos(t))/40$, ${}^F \alpha_O = \pi - 0.00763(t - 1 + e^{-t}\cos(t))$, ${}^F \beta_O = -0.00763(t - 1 + e^{-t}\cos(t))$ and ${}^F \gamma_O = \pi/2 + 0.061(t - 1 + e^{-t}\cos(t))$ which basically represents an ellipsoidal movement in the horizontal axes, plus a linear displacement for all linear and angular coordinates and plus a transient component (given by the term e^{-t}) in order for the target object to have initial velocity and acceleration equal to zero.

vi) The initial value for the joint velocities and joint accelerations is set to zero, that is $\dot{\mathbf{q}}(0) = \ddot{\mathbf{q}}(0) = \mathbf{0}$. Note that for these initial conditions, as before, the SMC starts on the sliding surface Φ, i.e., (29) or (35) are satisfied due to $\mathbf{e}(0) = \mathbf{0}$, see $\{{}^F\mathbf{M}_C(0), {}^{C^*}\mathbf{M}_O, {}^F\mathbf{M}_O(t)\}$, and $\dot{\mathbf{e}}(0) = \ddot{\mathbf{e}}(0) = \mathbf{0}$ since the initial velocity and acceleration for the joints and the target is equal to zero.

10.2. Simulation results for the tracking task

Fig. 6 shows the object and camera trajectories during the tracking process. For clarity, only the trajectories for SMC using accelerations and $\mathbf{J}_s^\dagger$ is depicted. The differences between all the approaches is shown in the following figures.

Fig. 7 and Fig. 8 show the integral of the control action (i.e., $\dot{\mathbf{q}}$ or $\ddot{\mathbf{q}}$), the tracking error $\mathbf{e}$ and the constraint function vector (i.e., $\boldsymbol{\phi}_a$ or $\boldsymbol{\phi}_j$) for the SMC using joint accelerations and joint jerks, respectively. Note that the tracking errors are always lower than 0.7 millimeters and that, as before, the chattering band is greater when the transpose of the Jacobian matrix is used instead of the pseudoinverse.

DOI: 10.1002/rnc

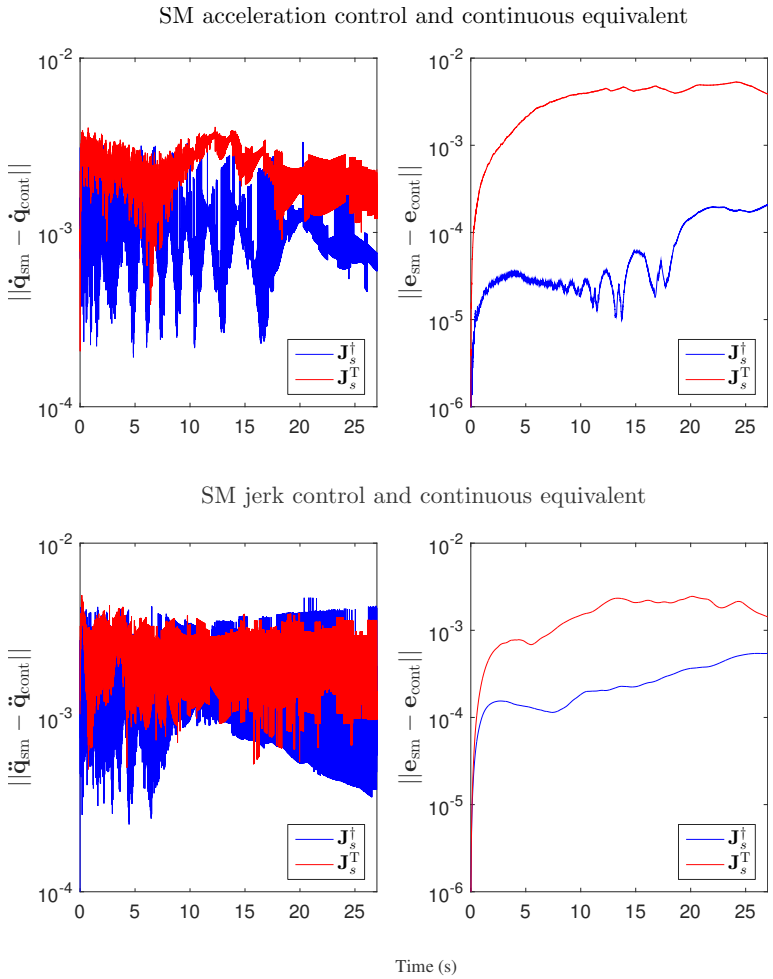

Figure 5. Simulated positioning task. SMC versus the continuos equivalent. From top to bottom and left to right plots: (1) Norm of the difference in $\dot{\mathbf{q}}$ between SMC using joint accelerations $\dot{\mathbf{q}}_{\text{sm}}$ and the continuous equivalent $\dot{\mathbf{q}}_{\text{cont}}$; (2) Norm of the difference in $\mathbf{e}$ between SMC using joint accelerations $\mathbf{e}_{\text{sm}}$ and the continuous equivalent $\mathbf{e}_{\text{cont}}$. (3) Norm of the difference in $\ddot{\mathbf{q}}$ between SMC using joint jerks $\ddot{\mathbf{q}}_{\text{sm}}$ and the continuous equivalent $\ddot{\mathbf{q}}_{\text{cont}}$; (4) Norm of the difference in $\mathbf{e}$ between SMC using joint jerks $\mathbf{e}_{\text{sm}}$ and the continuous equivalent $\mathbf{e}_{\text{cont}}$.

Fig. 9 compares the SMCs to their respective continuous equivalents in terms of the joint speeds or joint accelerations, i.e., $\dot{\mathbf{q}}$ or $\ddot{\mathbf{q}}$. The differences are similar for the SMC using joint accelerations and joint jerks.

A video of this simulated tracking task for the SMC using joint accelerations and $\mathbf{J}_s^\dagger$ (the other cases are very similar) can be played at https://media.upv.es/player/?id=96753c70-79d5-11e7-90ea-23686ce0f1be.

10.3. Robustness against errors

Three types of "modeling errors" are considered to analyze the robustness of the proposed SMC: low-level controller error, target estimation error and Jacobian matrix error.

In particular, the error in the Jacobian matrix $\mathbf{J}_s$ (3) can be due to the following sources of error: 1) image noise, which affects the interaction matrix $\mathbf{L}_s$; 2) error in the camera intrinsic parameters, which affects the interaction matrix $\mathbf{L}_s$; 3) error in the pose estimator, which affects the interaction matrix $\mathbf{L}_s$; 4) error in the target model, which affects matrix the interaction $\mathbf{L}_s$; 5) error in the camera to end-effector transformation matrix ${}^c\mathbf{M}_e$, which affects the twist matrix ${}^c\mathbf{V}_e$; 6) error

DOI: 10.1002/rnc

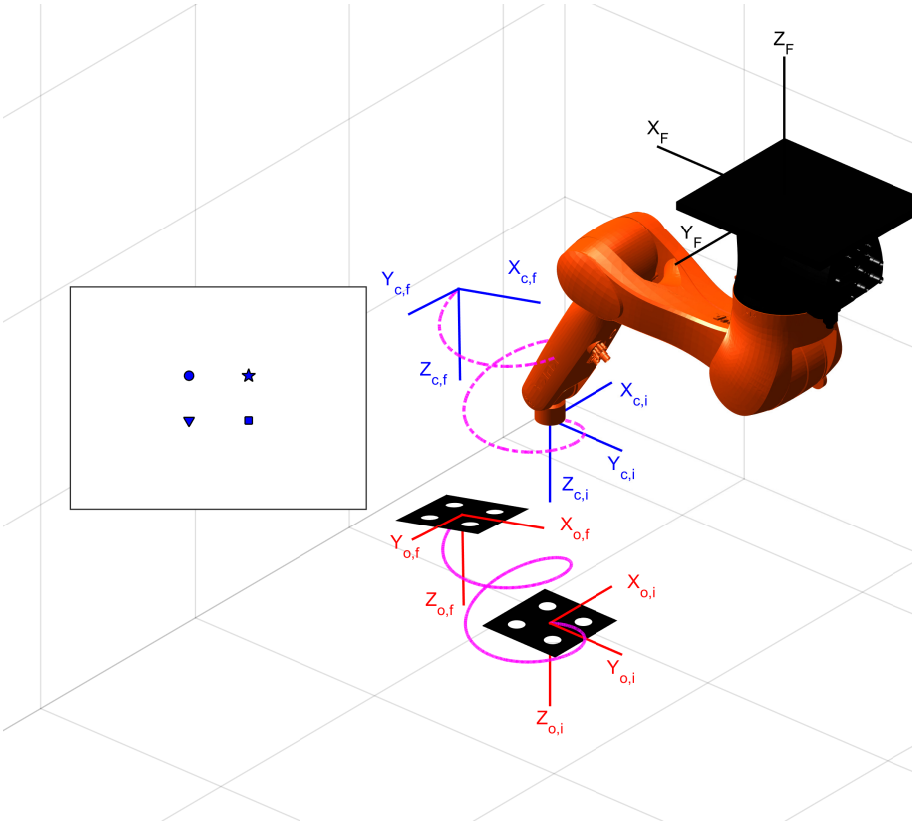

Figure 6. Tracking task: 6R robot, target object with four markers, initial and final object frames (O_i and O_f), initial and final camera frames (C_i and C_f) and resulting 3D trajectory of the object (magenta-continuous) and camera (magenta-dashed). For clarity, only the trajectories for the SMC using joints acceleration and $\mathbf{J}_s^\dagger$ is depicted. Image plane on the left-hand side: Constant desired features position.

in the DH parameters of the robot, which affects the robot Jacobian matrix ${}^e\mathbf{J}_e$; and 7) error in the robot configuration $\mathbf{q}$, which affects both the interaction matrix $\mathbf{L}_s$ and the robot Jacobian matrix ${}^e\mathbf{J}_e$. However, the first four sources of error listed above also affect the computation of the feature vector $\mathbf{s}$, which negatively influences[¶] both the proposed SMC for VS and its continuous equivalent, see Section 6.3. Therefore, in order to focus on highlighting the differences between both approaches, only the last three sources of error are simulated for the Jacobian matrix: camera to end-effector transformation matrix ${}^c\mathbf{M}_e$, which typically might have significant error in VS applications [60–63], specially if a coarse calibration is used [64–66]; DH parameters of the robot; and robot configuration $\mathbf{q}$.

For the simulations below, the norm of the tracking error $\mathbf{e}$ is used to compare three different cases: SMC with $\mathbf{J}_s^\dagger$, SMC with $\mathbf{J}_s^{\mathrm{T}}$ and the equivalent continuous control. The comparison is carried out for both SMCs using joint accelerations and joint jerks. Modeling errors are introduced with a signed variation in percentage of the actual value:

[¶]It is worth mentioning that, in VS applications the reference $\mathbf{s}_{ref}$ for the visual feature vector is typically specified to the robot system using the well-known *teaching by showing* method [1, 2, 59]. In this manner, the repetitive inaccuracies in $\mathbf{s}$ are also present in $\mathbf{s}_{ref}$ and, hence, they are mostly compensated when computing the error vector $\mathbf{e}$, which is the main signal used in both the discontinuous and continuous control law.

DOI: 10.1002/rnc

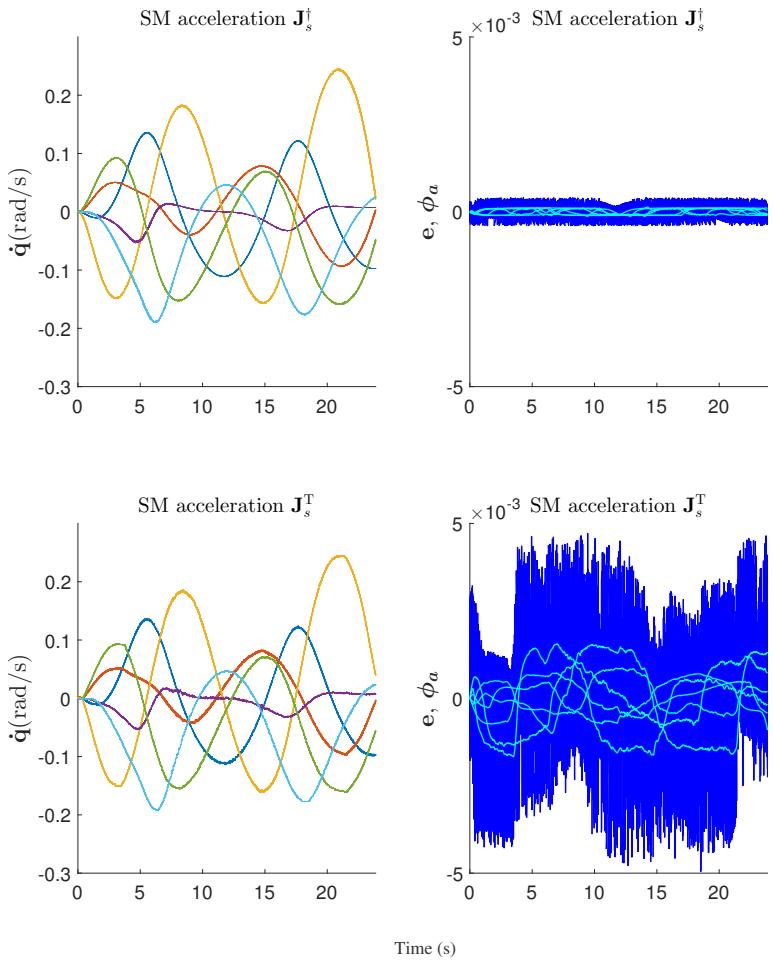

Figure 7. Simulated tracking task. SMC using joint accelerations. Integral of the control action, i.e., $\dot{\mathbf{q}}$, on the left-hand side, and error $\mathbf{e}$ and constraint function ϕ_a, on the right-hand side.

- Low-level controller error:

$$\ddot{\mathbf{q}}_e = \ddot{\mathbf{q}} + c_e \begin{bmatrix} -1 & 1 & -1 & -1 & -1 & 1 \end{bmatrix}^{\mathrm{T}} \circ |\ddot{\mathbf{q}}| \tag{59}$$

$$\dddot{\mathbf{q}}_e = \dddot{\mathbf{q}} + c_e \begin{bmatrix} -1 & 1 & -1 & -1 & -1 & 1 \end{bmatrix}^{\mathrm{T}} \circ |\dddot{\mathbf{q}}|. \tag{60}$$

- Target motion estimation error:

$$\partial \mathbf{s}_e/\partial t = \partial \mathbf{s}/\partial t + t_e \begin{bmatrix} -1 & 1 & -1 & -1 & -1 & 1 \end{bmatrix}^{\mathrm{T}} \circ |\partial \mathbf{s}/\partial t|. \tag{61}$$

- Error in the camera to end-effector transformation matrix:

$${}^c\mathbf{M}_{e,e} = {}^c\mathbf{M}_e + \begin{bmatrix} m_{de} \begin{bmatrix} 1 & -1 & 1 \end{bmatrix} & m_{re} \begin{bmatrix} 1 & -1 & -1 \end{bmatrix} \end{bmatrix}^{\mathrm{T}}. \tag{62}$$

- Error in the DH parameters (length vectors $\mathbf{d}$ and $\mathbf{a}$) of the robot:

$$\begin{bmatrix} \mathbf{d}_e^{\mathrm{T}} \\ \mathbf{a}_e^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \mathbf{d}^{\mathrm{T}} \\ \mathbf{a}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} + DH_e \begin{bmatrix} -1 & -1 & 1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 & -1 & 1 \end{bmatrix}^{\mathrm{T}} \circ \begin{bmatrix} |\mathbf{d}|^{\mathrm{T}} \\ |\mathbf{a}|^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}. \tag{63}$$

- Robot configuration error:

$$\mathbf{q}_e = \mathbf{q} + q_e \begin{bmatrix} -1 & 1 & -1 & -1 & -1 & 1 \end{bmatrix}^{\mathrm{T}}, \tag{64}$$

DOI: 10.1002/rnc

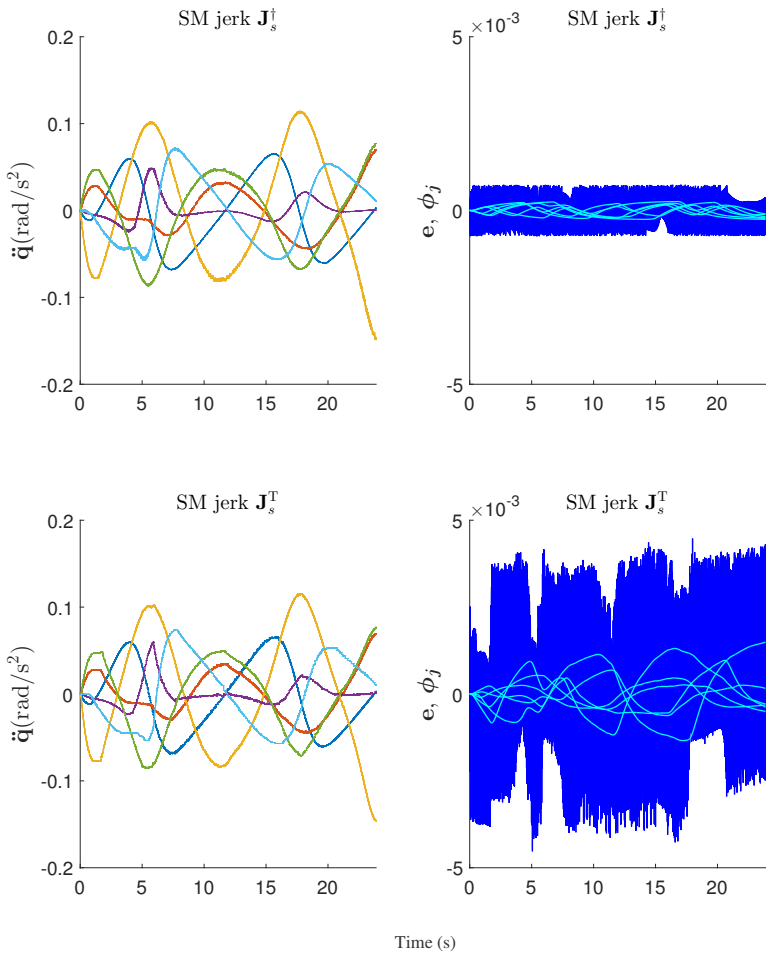

Figure 8. Simulated tracking task. SMC using joint jerks. Integral of the control action, i.e., $\ddot{\mathbf{q}}$, on the left-hand side, and error $\mathbf{e}$ and constraint function ϕ_j, on the right-hand side.

where $|\cdot|$ represents the absolute value function, symbol $\circ$ denotes the element-wise or Hadamard product, and $\{c_e, t_e, DH_e\}$ and $\{m_{de}, m_{re}, q_e\}$ represent the percentage errors and absolute errors, respectively, considered in each case.

Fig. 10 and Fig. 11 show the norm of the tracking error for (a) the ideal case, (b) a low-level controller error of $c_e = 20\%$, (c) a target motion estimation error of $t_e = 40\%$, (d) a camera to end-effector error of $m_{de} = 0.05$ m and $m_{re} = 0.35$ rad, (e) a DH parameters error of $DH_e = 20\%$, and (f) a robot configuration error of $q_e = 0.175$ rad, when using joint accelerations and joint jerks, respectively. Note that, the equivalent continuous control for each case has very small values for the tracking error in the ideal case, but this value dramatically increases (around three orders of magnitude) in the presence of modeling errors. In contrast, the value of the tracking error for the SMC using $\mathbf{J}_s^\dagger$ approximately remains the same regardless the considered modeling errors. When $\mathbf{J}_s^{\mathrm{T}}$ is used for the SMC the norm of the tracking error may be larger, since it represents an additional error to the already existing error due to using matrix transpose. However, this increment is significantly lower (up to one order of magnitude) than the one experienced by the continuous equivalent.

Nevertheless, the tracking errors for the SMC are due to the chattering band and, therefore, they can be reduced as much as desired by lowering the sampling time. In this sense, Fig. 12 shows a case that combines all the errors simultaneously $\{c_e = 20\%, t_e = 20\%, m_{de} = 0.02\,\mathrm{m}, m_{re} = 0.2\,\mathrm{rad}, DH_e = 10\%, q_e = 0.1\,\mathrm{rad}\}$, and the effect of reducing one order of magnitude the sampling time T_s. In particular, the norm of the tracking error for the SMCs is reduced around one order

DOI: 10.1002/rnc

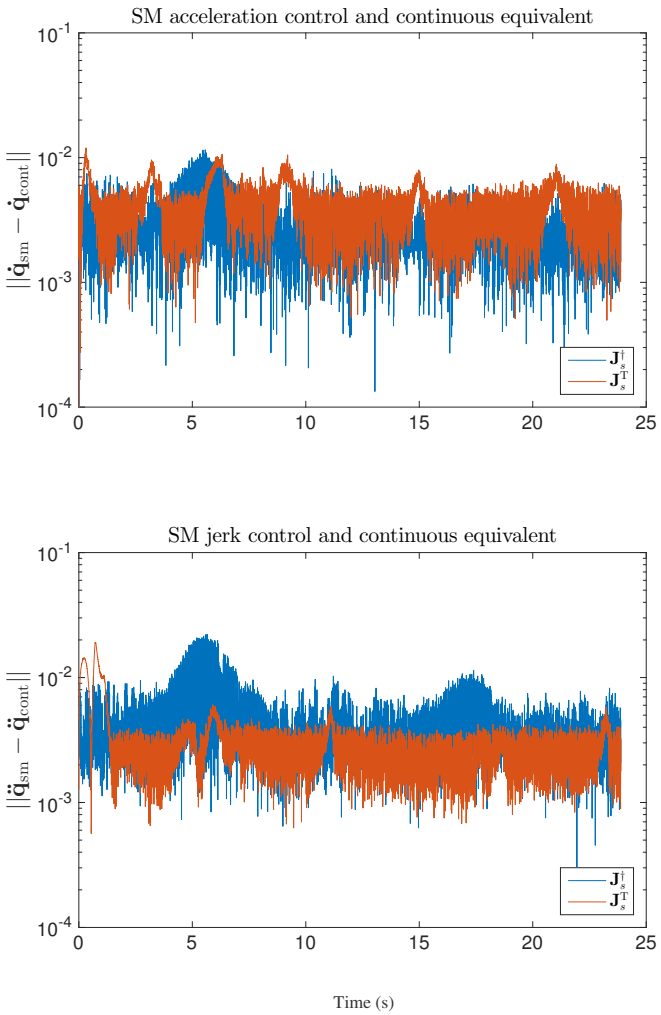

Figure 9. Simulated tracking task. Norm of the integral of the control action with respect to the continuous equivalent. SMC using joint accelerations, for $\mathbf{J}_s^{\dagger}$ and $\mathbf{J}_s^{\mathrm{T}}$, and SMC using joint jerks, for $\mathbf{J}_s^{\dagger}$ and $\mathbf{J}_s^{\mathrm{T}}$.

of magnitude, whereas that value for the continuous equivalent basically remains the same. This evidences that the tracking errors for the SMCs are a consequence of the chattering band and, hence, these controls are *robust* to modeling errors. In contrast, the tracking errors for the equivalent continuous controllers are not reduced by lowering the sampling time since the control law is not qualitatively robust against modeling errors.

The simulation results can be concluded as follows. On the one hand, the performance of the continuous control is good as long as no significant errors are present (in the Jacobian matrix, low-level controller, and target motion estimation), while the sampling period has no significant effect on it (obviously, the Nyquist-Shannon sampling theorem must be satisfied). On the other hand, the proposed SMC is robust against these errors but the sampling period compromises its performance due to the chattering drawback. Therefore, in order to acknowledge the advantages of each approach, a *fair guideline* for practical applications is using the continuous control when errors are small and using the proposed SMC with reasonable fast sampling rates otherwise.

DOI: 10.1002/rnc

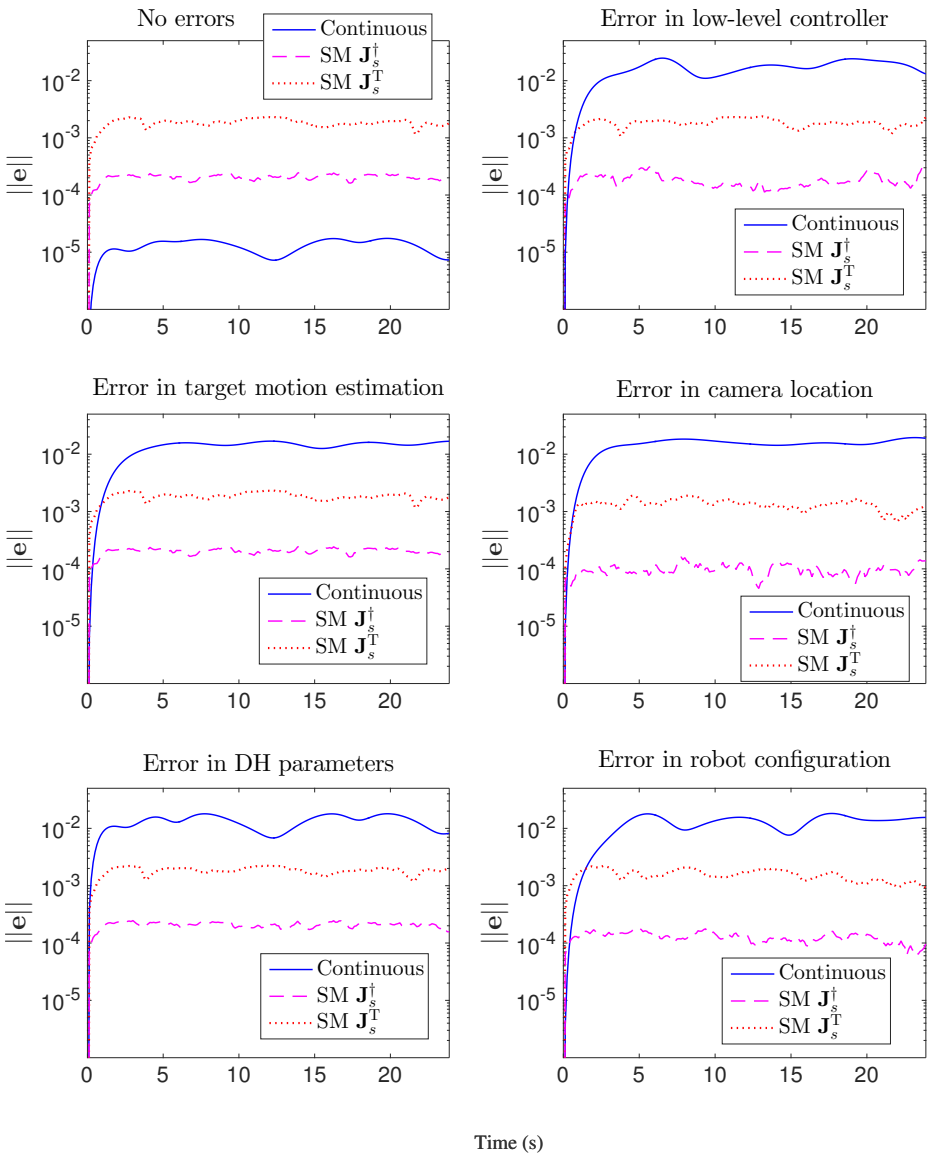

Figure 10. Simulated tracking task. Norm of the tracking error for SMC using joint accelerations and continuous equivalent in presence of modeling errors: Continuous (solid, blue), using $\mathbf{J}_s^\dagger$ (dashed, magenta) and using $\mathbf{J}_s^{\mathrm{T}}$ (dotted, red).

11. REAL EXPERIMENTATION

The proposed SMC has been implemented to obtain real experiments in order to demonstrate its feasibility and robustness. The following setup has been used (see Fig. 13): a Kuka KR6 R900 sixx robot manipulator, coined as Agilus, in ceiling-mounted position, is equipped with the Kuka.RobotSensorInterface (RSI) technology that allows external real-time communication using the Ethernet UDP protocol; a general purpose web cam rigidly attached to the robot end-effector (eye-in-hand consiguration), which is used for image acquisition; a screen, which is used to display the target object markers; and an external PC with Ubuntu 12.04 OS prompted with real time kernel that implements the computer vision and control algorithms proposed in this work. The position of the image features is updated using the dot tracker in ViSP (Visual Servoing Platform) [67], whilst the object pose is estimated to update the visual feature vector $\mathbf{s}$ and to compute $\mathbf{L}_s$.

DOI: 10.1002/rnc

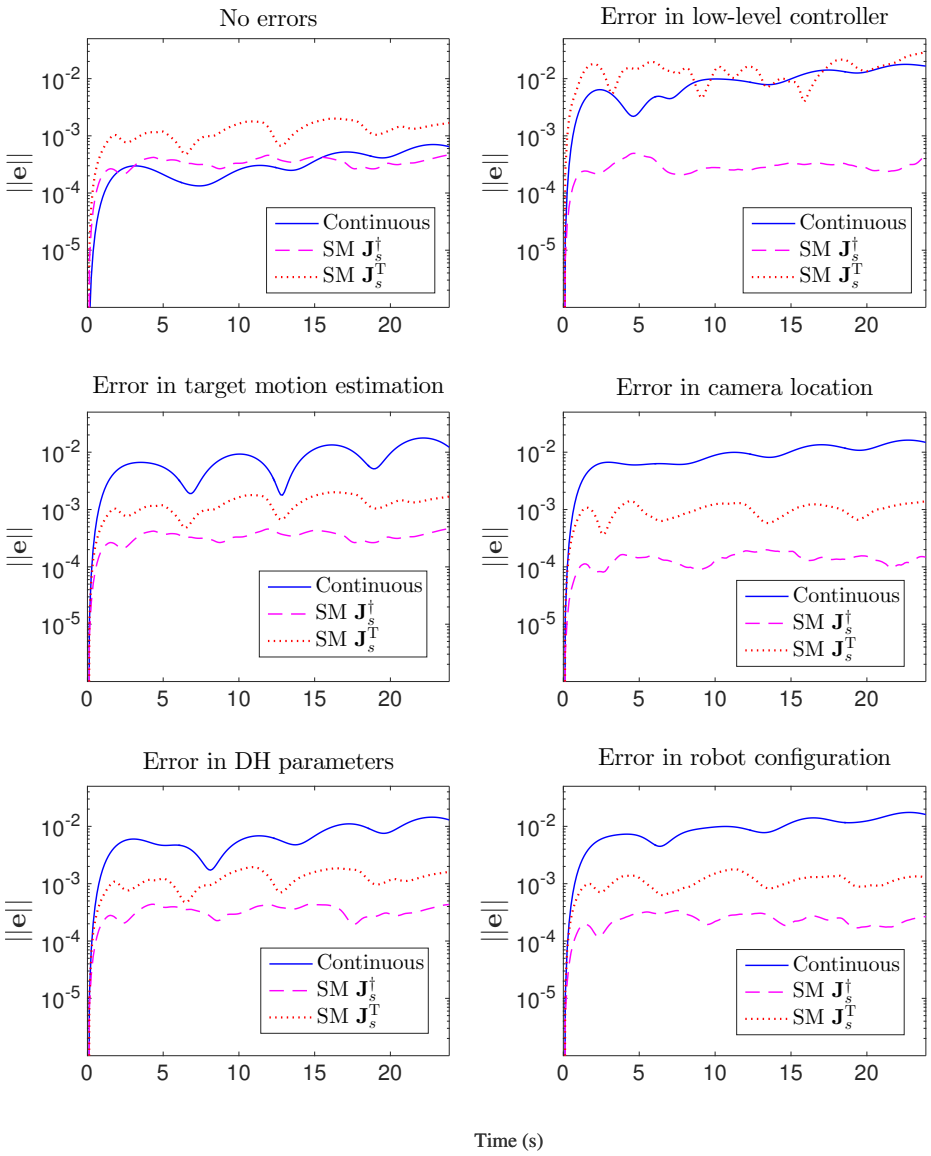

Figure 11. Simulated tracking task. Norm of the tracking error for SMC using joint jerks and continuous equivalent in presence of modeling errors: Continuous (solid, blue), using $\mathbf{J}_s^\dagger$ (dashed, magenta) and using $\mathbf{J}_s^{\mathrm{T}}$ (dotted, red).

Two experiments have been conducted to show the validity of the proposed approach: (1) a *positioning task* consisting of moving the robot from the initial to the goal position, defined with respect to a still target object; (2) a *tracking task* consisting of moving the robot to follow a circular trajectory defined by the object motion.

11.1. Experiment conditions and parameter values

Both experiments were run under the following conditions:

i) Both the proposed SMC using joint accelerations and its continuous equivalent were implemented using $\mathbf{J}_s^\dagger$.
ii) Three periodic threads are defined and scheduled following a fixed priority scheme, from highest to lowest: server, control and vision threads. The server period must be set to 4

DOI: 10.1002/rnc

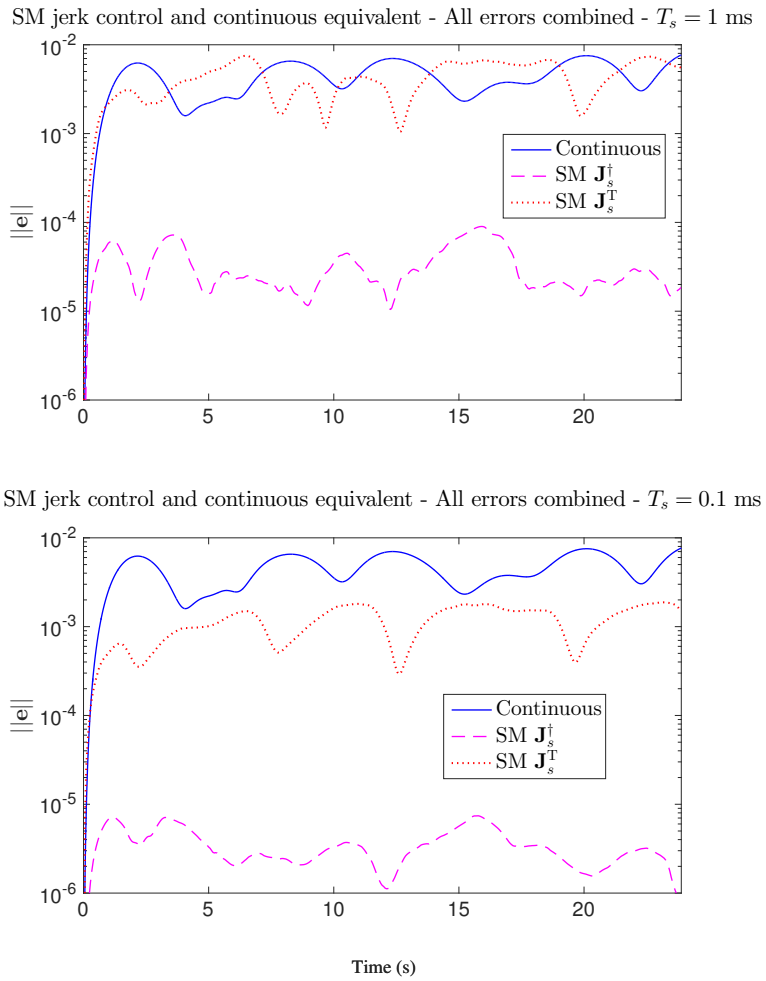

Figure 12. Simulated tracking task. Effect of sampling time in the tracking error for SMC using joint jerks and continuous equivalent in presence of modeling errors: Continuous (solid, blue), using $\mathbf{J}_s^{\dagger}$ (dashed, magenta) and using $\mathbf{J}_s^{\mathrm{T}}$ (dotted, red).

Figure 13. Experimental setup: 6R serial industrial manipulator in ceiling position with the camera rigidly attached to the robot end-effector (eye in hand configuration) and a screen to display the object markers.

DOI: 10.1002/rnc

milliseconds due to robot specification. Both the vision period and the control period T_s are set to 100 milliseconds to guarantee an appropriate scheduling.

iii) The commanded joint accelerations $\ddot{\mathbf{q}}_c$ computed by the proposed algorithm are double integrated (see Sec. 3.3) to obtain the commanded joint positions $\mathbf{q}_c$ sent to the robot controller.

iv) The perspective camera model without distortion is considered with the following parameters: the ratio between the focal length and the size of a pixel is $[F_u, F_v] = [710.1, 709.8]$ pixels, the resolution $[W_V, H_V] = [640, 480]$ pixels, and the central point $[u_0, v_0] = [310, 247]$ pixels. The camera to end-effector transformation matrix is ${}^{c}\mathbf{M}_e = \begin{bmatrix} 0 & 0.07 & -0.05 & 0 & 0 & -\pi/2 \end{bmatrix}^{\mathrm{T}}$, in compact notation.

v) Four markers define the object, representing the vertices of a square with a side length of 17 centimeters in both cases.

vi) Coefficient for the first-order error differential equation $K_a = 10$ and control action amplitude for the SMC $u^+ = 0.1$.

vii) A discrete first-order low-pass IIR filter (see Sec. 7) has been used with a pole at 0.4 to reduce the noise of the pose estimation signal before computing the time-derivative of the error signal.

viii) In the *positioning task*, the initial configuration is given by the robot joint position vector $\mathbf{q}(0) = \begin{bmatrix} 2.67 & -1.80 & 2.14 & 0.79 & -1.44 & -0.97 \end{bmatrix}^{\mathrm{T}}$ rad, and the visual feature vector $\mathbf{s}(0) = \begin{bmatrix} 0.111 & 0.006 & 0.045 & -0.049 & -0.012 & 0.014 \end{bmatrix}^{\mathrm{T}}$.

ix) In the *tracking task*, the initial configuration is given by the same robot joint position vector $\mathbf{q}(0)$, and the reference trajectory for the visual feature vector $\mathbf{s}_{ref}(t) = \mathbf{s}(0) = \begin{bmatrix} 0 & 0 & 0.5 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$, i.e., the object is aligned with the camera optical axis and at a distance of 0.5 meters. Moreover, the target object describes a circular trajectory whose radius is equal to 8.5 centimeters and whose angular velocity follows a trapezoidal profile with the following parameters: initial velocity equal to zero; nominal velocity equal to 0.1 rad/s; and acceleration to achieve the nominal velocity equal to 1 rad/s^2.

11.2. Experimental results with no errors

For the positioning task, Fig. 14 shows the position $\mathbf{e_T}$ and orientation $\mathbf{e_R}$ errors, the commanded joint accelerations $\ddot{\mathbf{q}}_c$ and the constraint function $\boldsymbol{\phi}_a$ obtained using the SMC. Note that, even with noisy signals and a sampling period of 0.1 s, the robot goal position is reached with $\mathbf{e}_{T,i} < 5$ mm and $\mathbf{e}_{R,i} < 0.002$ rad. Fig. 15 shows the trajectory of the object markers in the image plane and the 3D camera trajectory, which is very similar to the ideal trajectory, i.e., for PBVS approaches, a straight line from the initial C to the goal C^*. A video of this positioning experiment can be played at (video at double speed) https://media.upv.es/player/?id=a187cf40-7621-11e7-90ea-23686ce0f1be.

For the tracking task, Fig. 16 shows the joint speeds $\dot{\mathbf{q}}$, the constraint function $\boldsymbol{\phi}_a$ and the tracking error $\mathbf{e}$. Note that, the tracking errors are relatively small: below 0.01 m or radians. Fig. 17 shows the circular 3D camera trajectory. A video of this tracking experiment can be played at (video at double speed) https://media.upv.es/player/?id=6b8148d0-7622-11e7-90ea-23686ce0f1be.

11.3. Experimental robustness against errors

The robustness of the proposed approach is experimentally analyzed by adding a signed error in the Jacobian matrix as follows:

$$\mathbf{J}_{s,e} = \mathbf{J}_s + j_e \begin{bmatrix} -1 & -1 & -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & -1 \\ -1 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & -1 \\ 1 & 1 & -1 & 1 & 1 & 1 \end{bmatrix} \circ |\mathbf{J}_s|, \tag{65}$$

DOI: 10.1002/rnc

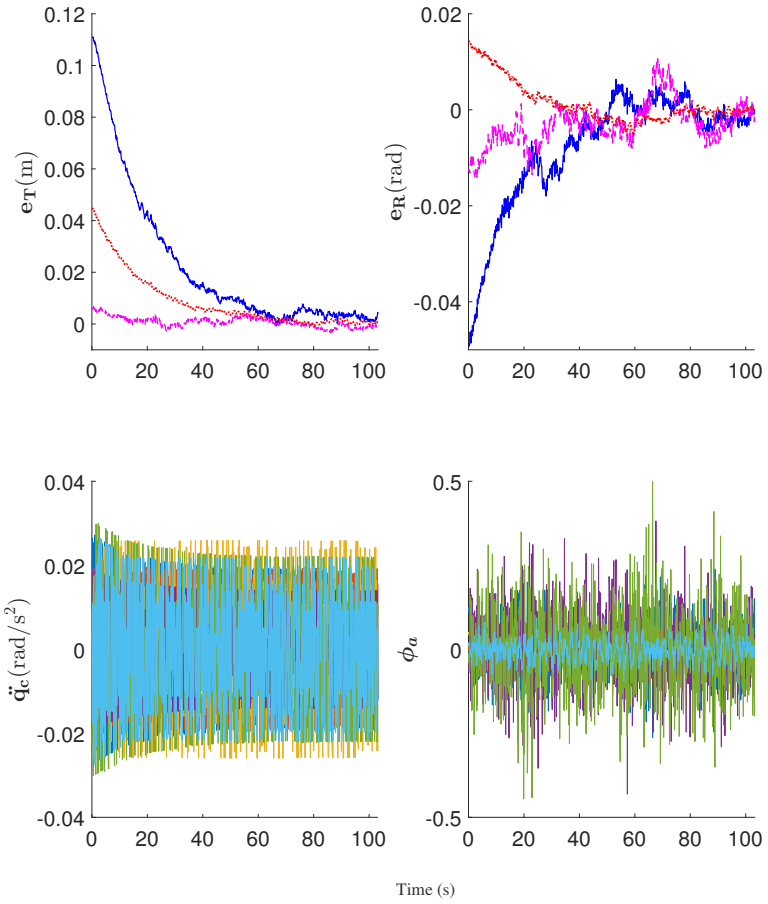

Figure 14. Real positioning experiment using the proposed SMC. From top to bottom and left to right plots: (1) translation errors $\mathbf{e_T}$; (2) rotation errors $\mathbf{e_R}$; (3) commanded joint accelerations $\ddot{\mathbf{q}}_c$; (4) Constraint function vector ϕ_a.

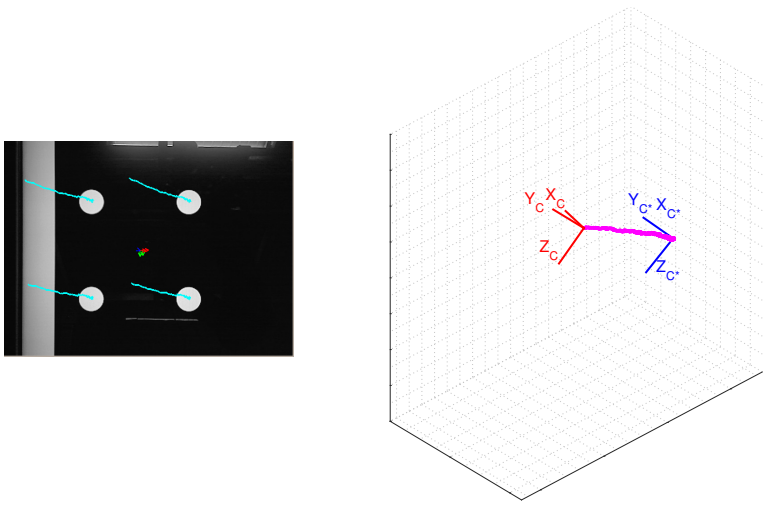

Figure 15. Real positioning experiment: left, trajectory of the object markers in the image plane; right, 3D camera trajectory.

with $j_e = 30\%$ being the considered percentage error.

Prepared using rncauth.cls
DOI: 10.1002/rnc

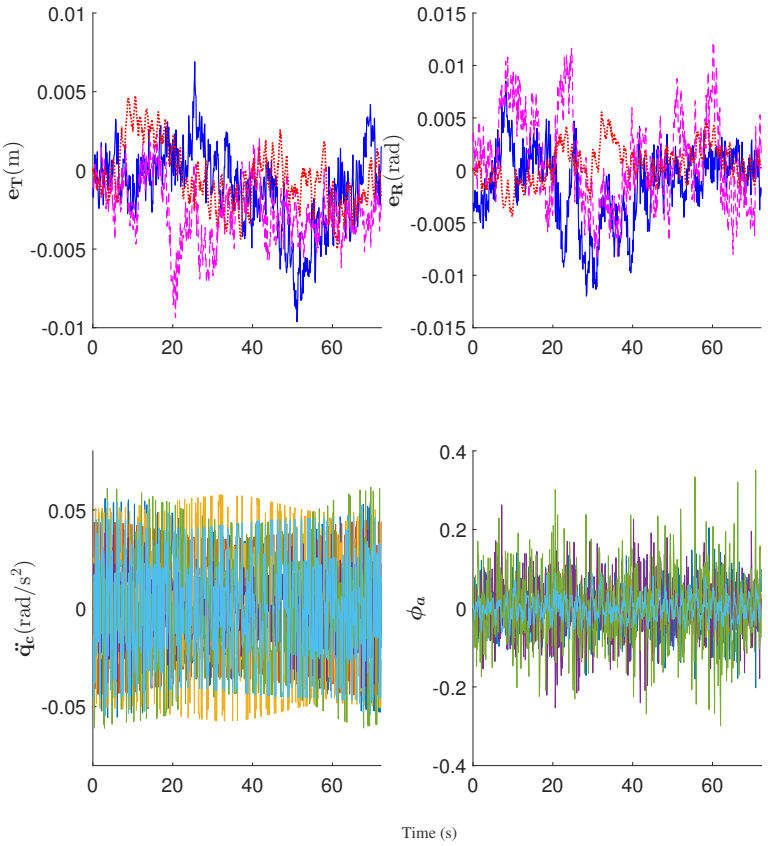

Figure 16. Real tracking experiment using the proposed SMC. From left to right plots: (1) translation errors $\mathbf{e_T}$; (2) rotation errors $\mathbf{e_R}$; (3) commanded joint accelerations $\ddot{\mathbf{q}}_c$; (4) Constraint function vector $\boldsymbol{\phi}_a$.

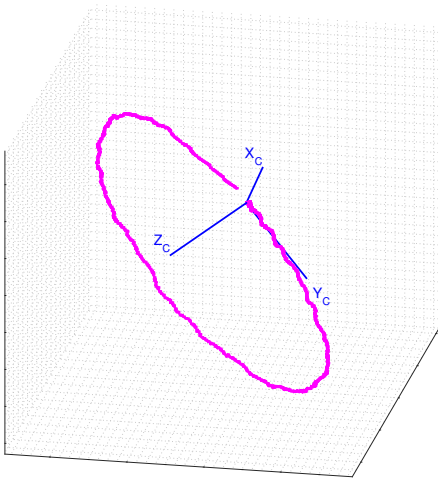

Figure 17. Real tracking experiment: 3D camera trajectory for a reference trajectory given by an almost closed circle.

Fig. 18 shows the tracking error $\mathbf{e}$ for SMC and the continuous equivalent both without and with errors in the Jacobian matrix. It can be seen that the tracking errors for the SMC are not significantly modified when the error in the Jacobina matrix is introduced: they are always below 0.010 m and 0.020 radians. Whereas for the continuous equivalent: position errors are approximately

DOI: 10.1002/rnc

doubled when the error is introduced (they reached values of up to 0.015 m); while orientation errors drastically increased when the error is introduced (they reached values of up to 0.047 radians).

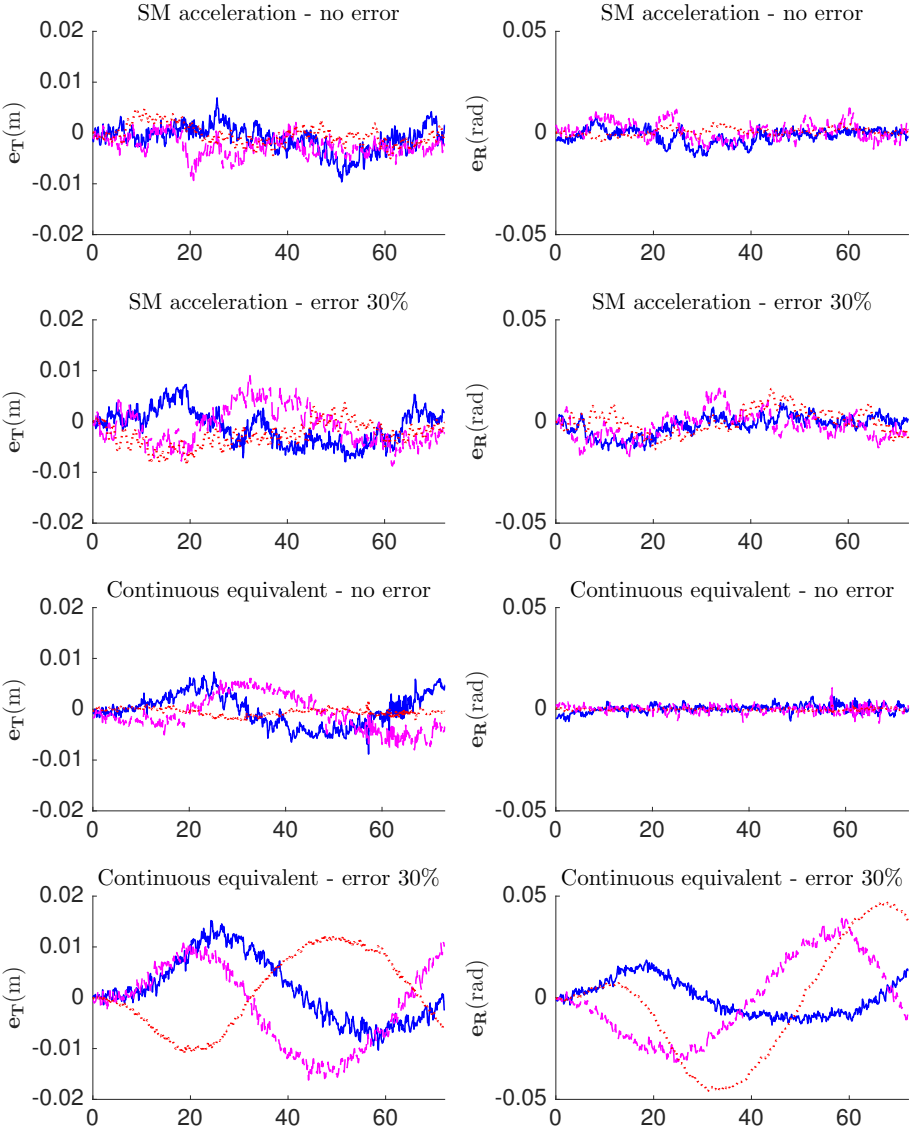

Figure 18. Real tracking experiment with 30% signed error in the Jacobian Matrix. Comparison of SMC using joint accelerations and the continuous equivalent.

A final consideration about the presented experiments is given as follows. As explained in Section 6.2, the proposed SMC for VS is smoother than those in literature [25–36] because yields continuous joint velocities. This characteristic can be *qualitatively* appreciated by comparing the experimental results above and those presented in the mentioned works. However, from a *quantitative* point of view, this advantage is fully exploited when considering large accelerations and abrupt maneuvers in the reference trajectory. Nevertheless, this has not been possible in the above VS experiments (neither in those presented in the listed works) due to the relatively large sampling time used for the SMC, which is required for image acquisition and processing.

DOI: 10.1002/rnc

12. CONCLUSIONS

An approach for reference tracking in visual servoing has been presented using a sliding mode strategy. In particular, two sliding mode controls have been obtained depending on whether the joint accelerations or the joint jerks are considered as the discontinuous control action. The main contributions of this research are listed as follows:

- From a theoretical point of view, it has been proven that the Jacobian transpose can be used for the sliding mode control of robotic systems, instead of the conventional Jacobian pseudoinverse.
- A systematic procedure has been proposed to increase the relative degree between the original constraint function and the discontinuous control action, which represents an alternative to the classical high-order sliding mode control.
- The proposed approach for robot visual servoing is smoother than the previous sliding mode controls used for this purpose since it yields continuous joint velocities.
- The proposed sliding mode controls have been compared theoretically and in simulation to their classical continuous counterparts.
- The applicability and feasibility of the proposed approach is substantiated by experimental results using a conventional 6R industrial manipulator for positioning and tracking tasks. In particular, the robustness of the method compared to the continuous equivalent has been successfully verified in the experiments by introducing an error in the Jacobian matrix.

Finally, the main advantages of the proposed sliding mode approach are smoothness, robustness and low computational cost, while its main limitation is the chattering drawback.

REFERENCES

1. Hutchinson S, Hager G, Corke P. A tutorial on visual servo control. *Robotics and Automation, IEEE Transactions on* 1996; **12**(5):651–670, doi:10.1109/70.538972.
2. Chaumette F, Hutchinson S. Visual servoing and visual tracking. *Springer Handbook of robotics* 2008; :563–583.
3. Corke P. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. Springer-Verlag: Berlin, Germany, 2011.
4. Ryan EP, Corless M. Ultimate boundedness and asymptotic stability of a class of uncertain dynamical systems via continuous and discontinuous feedback control. *IMA Journal of Mathematical Control and Information* 1984; **1**(3):223, doi:10.1093/imamci/1.3.223.
5. Chaumette F, Hutchinson S. Visual servo control. i. basic approaches. *IEEE Robotics Automation Magazine* Dec 2006; **13**(4):82–90, doi:10.1109/MRA.2006.250573.
6. Chaumette F, Hutchinson S. Visual servo control. ii. advanced approaches [tutorial]. *IEEE Robotics Automation Magazine* March 2007; **14**(1):109–118.
7. Bonfe M, Mainardi E, Fantuzzi C. Variable structure pid based visual servoing for robotic tracking and manipulation. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2002; 396–401 vol.1, doi:10.1109/IRDS.2002.1041421.
8. Solanes JE, Muñoz Benavent P, Girbés V, Armesto L, Tornero J. On improving robot image-based visual servoing based on dual-rate reference filtering control strategy. *Robotica* 2016; **34**(12):2842–2859, doi:10.1017/S0263574715000454.
9. Elena M, Cristiano M, Damiano F, Bonfe M. Variable structure pid controller for cooperative eye-in-hand/eye-to-hand visual servoing. *Proceedings of 2003 IEEE Conference on Control Applications, 2003. CCA 2003*, vol. 2, 2003; 989–994 vol.2, doi:10.1109/CCA.2003.1223145.
10. Hashimoto K, Ebine T, Kimura H. Visual servoing with hand-eye manipulator-optimal control approach. *IEEE Transactions on Robotics and Automation* Oct 1996; **12**(5):766–774, doi:10.1109/70.538981.
11. Chan A, Leonard S, Croft EA, Little JJ. Collision-free visual servoing of an eye-in-hand manipulator via constraint-aware planning and control. *Proceedings of the 2011 American Control Conference*, 2011; 4642–4648, doi:10.1109/ACC.2011.5991008.
12. Allibert G, Courtial E, Chaumette F. Visual servoing via Nonlinear Predictive control. *Visual Servoing via Advanced Numerical Methods*, Chesi G, Hashimoto K (eds.). LNCIS 401, Springer-Verlag, 2010; 375–394.
13. Hajiloo A, Keshmiri M, Xie WF, Wang TT. Robust Online Model Predictive Control for a Constrained Image-Based Visual Servoing. *IEEE Trans. on Industrial Electronics* 2016; **63**(4):2242–2250.
14. Kragic D, Christensen HI. Robust visual servoing. *The International Journal of Robotics Research* 2003; **22**(10-11):923–939, doi:10.1177/027836490302210009.
15. Mezouar Y, Chaumette F. Path planning in image space for robust visual servoing. *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 3, 2000; 2759–2764 vol.3, doi:10.1109/ROBOT.2000.846445.

16. Morel G, Zanne P, Plestan F. Robust visual servoing: bounding the task function tracking errors. *IEEE Transactions on Control Systems Technology* Nov 2005; **13**(6):998–1009, doi:10.1109/TCST.2005.857409.
17. Hammouda L, Kaaniche K, Mekki H, Chtourou M. Robust visual servoing using global features based on random process. *Int. J. Comput. Vision Robot.* Apr 2015; **5**(2):138–154, doi:10.1504/IJCVR.2015.068803.
18. Yang YX, Liu D, Liu H. Robot-self-learning visual servoing algorithm using neural networks. *Proceedings. International Conference on Machine Learning and Cybernetics*, vol. 2, 2002; 739–742 vol.2, doi:10.1109/ICMLC.2002.1174473.
19. Sadeghzadeh M, Calvert D, Abdullah HA. Self-learning visual servoing of robot manipulator using explanation-based fuzzy neural networks and q-learning. *Journal of Intelligent & Robotic Systems* 2015; **78**(1):83–104, doi:10.1007/s10846-014-0151-5.
20. Lee AX, Levine S, Abbeel P. Learning Visual Servoing with Deep Features and Fitted Q-Iteration. *ArXiv e-prints* Mar 2017; .
21. Fakhry H, Wilson W. A modified resolved acceleration controller for position-based visual servoing. *Mathematical and Computer Modelling* 1996; **24**(5-6):1–9.
22. Keshmiri M, Xie W, Mohebbi A. Augmented image-based visual servoing of a manipulator using acceleration command. *IEEE Trans. on Industrial Electronics* 2014; **61**(10):5444–5452.
23. Edwards C, Spurgeon S. *Sliding Mode Control: Theory and Applications*. 1st edn., Taylor & Francis: UK, 1998.
24. Yu. Stability Analysis of Visual Servoing with Sliding-mode Estimation and Neural Compensation. *Int. Journal of Control, Automation, and Systems* 2013; **4**(5):545–558.
25. Zanne P, Morel G, Piestan F. Robust vision based 3D trajectory tracking using sliding mode control. *Proceedings 2000 ICRA. Millennium Conference. IEEE Int. Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 3, IEEE, 2000; 2088–2093.
26. Oliveira TR, Peixoto AJ, Leite AC, Hsu L. Sliding mode control of uncertain multivariable nonlinear systems applied to uncalibrated robotics visual servoing. *2009 American Control Conference* 2009; (April 2017):71–76.
27. Oliveira TR, Leite AC, Peixoto AJ, Hsu L. Overcoming limitations of uncalibrated robotics visual servoing by means of sliding mode control and switching monitoring scheme. *Asian Journal of Control* 2014; **16**(3):752–764.
28. Li F, Xie HL. Sliding mode variable structure control for visual servoing system. *Int. Journal of Automation and Computing* 2010; **7**(3):317–323.
29. Kim J, Kim D, Choi S, Won S. Image-based visual servoing using sliding mode control. *Proceedings of the SICE-ICASE Int. Joint Conference*, 2006; 4996–5001.
30. Parsapour M, RayatDoost S, Taghirad H. A 3d sliding mode control approach for position based visual servoing system. *Scientia Iranica. Transaction B, Mechanical Engineering* 2015; **22**(3):844–853.
31. Burger W, Dean-Leon E, Cheng G. Robust second order sliding mode control for 6D position based visual servoing with a redundant mobile manipulator. *2015 IEEE-RAS 15th Int. Conference on Humanoid Robots (Humanoids)*, IEEE, 2015; 1127–1132.
32. Becerra HM, Sagüés C. Sliding Mode Control for Visual Servoing of Mobile Robots using a Generic Camera. *Sliding Mode Control*, Prof Andrzej Bartoszewicz (ed.). 2011; 221–236.
33. Becerra HM, López-Nicolás G, Sagüés C. A sliding-mode-control law for mobile robots based on epipolar visual servoing from three views. *IEEE Trans. on Robotics* 2011; **27**(1):175–183.
34. Parsapour M, Taghirad H. Kernel-based sliding mode control for visual servoing system. *IET Computer Vision* 2015; **9**(3):309–320.
35. Xin J, Ran BJ, Ma XM. Robot visual sliding mode servoing using SIFT features. *2016 35th Chinese Control Conference (CCC)*, IEEE, 2016; 4723–4729.
36. Zhao YM, Lin Y, Xi F, Guo S, Ouyang P. Switch-Based Sliding Mode Control for Position-Based Visual Servoing of Robotic Riveting System. *Journal of Manufacturing Science and Engineering* 2017; **139**(4):041 010 (11).
37. Moosavian S, Papadopoulos E. Modified transpose jacobian control of robotic systems. *Automatica* 2007; **43**(7):1226–1233.
38. Huang C, Wang X, Wang Z. A class of transpose jacobian-based NPID regulators for robot manipulators with an uncertain kinematics. *Journal of Field Robotics* 2002; **19**(11):527–539.
39. Sagara S, Taira Y. Digital control of space robot manipulators with velocity type joint controller using transpose of generalized jacobian matrix. *Artificial Life and Robotics* December 2008; **13**(1):355–358.
40. Khalaji A, Moosavian S. Modified transpose jacobian control of a tractor-trailer wheeled robot. *Journal of Mechanical Science and Technology* 2015; **29**(9):3961–3969.
41. Golub G, Van Loan C. *Matrix Computations*. 3rd edn., The Johns Hopkins University InPress: Baltimore, MD, 1996.
42. Utkin V, Guldner J, Shi J. *Sliding Mode Control in Electro-Mechanical Systems*. 2nd edn., Taylor & Francis: London, 2009.
43. Utkin V. Discussion aspects of high-order sliding mode control. *IEEE Trans. on Automatic Control* 2016; **61**(3):829–833.
44. Romdhane H, Dehri K, Nouri A. Discrete second-order sliding mode control based on optimal sliding function vector for multivariable systems with input-output representation. *International Journal of Robust and Nonlinear Control* 2016; **26**(17):3806–3830.
45. Sharma N, Janardhanan S. Optimal discrete higher-order sliding mode control of uncertain LTI systems with partial state information. *International Journal of Robust and Nonlinear Control* 2017; :In Press (DOI: 10.1002/rnc.3785).
46. Levant A. Sliding order and sliding accuracy in sliding mode control. *Int. Journal of Control* 1993; **58**(6):1247–1263.
47. Levant A. Higher-order sliding modes, differentiation and output-feedback control. *Int. Journal of Control* 2003; **76**(9-10):924–941.
48. Bartolini G, Ferrara A, Usai E. Chattering avoidance by second-order sliding mode control. *IEEE Trans. on Automatic Control* 1998; **43**(2):241–246.

49. Siciliano B, Sciavicco L, Villani L, Oriolo G. *Robotics: Modelling, Planning and Control*. Springer-Verlag: London, UK, 2009.
50. Deo A, Walker I. Overview of damped least-squares methods for inverse kinematics of robot manipulators. *Journal of Intelligent & Robotic Systems* 1995; **14**(1):43–68.
51. Wheeler G, Su C, Stepanenko Y. A sliding mode controller with improved adaption laws for the upper bounds on the norm of uncertainties. *Automatica* 1998; **34**(12):1657–1661.
52. Lu Y. Sliding-mode disturbance observer with switching-gain adaptation and its application to optical disk drives. *IEEE Transactions on Industrial Electronics* 2009; **56**(9):3743–3750.
53. Chen X, Shen W, Cao Z, Kapoor A. A novel approach for state of charge estimation based on adaptive switching gain sliding mode observer in electric vehicles. *Journal of Power Sources* 2014; **246**:667–678.
54. Cong B, Chen Z, Liu X. On adaptive sliding mode control without switching gain overestimation. *International Journal of Robust and Nonlinear Control* 2014; **24**(3):515–531.
55. Taleb M, Plestan F, Bououlid B. An adaptive solution for robust control based on integral high-order sliding mode concept. *International Journal of Robust and Nonlinear Control* 2015; **25**(8):1201–1213.
56. Zhu J, Khayati K. On a new adaptive sliding mode control for MIMO nonlinear systems with uncertainties of unknown bounds. *International Journal of Robust and Nonlinear Control* 2017; **27**(6):942–962.
57. Hafez AHA, Cervera E, Jawahar CV. Hybrid visual servoing by boosting ibvs and pbvs. *2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications*, 2008; 1–6.
58. Kermorgant O, Chaumette F. Combining IBVS and PBVS to ensure the visibility constraint. *2011 IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, IEEE, 2011; 2849–2854.
59. Corke P, Hutchinson S. A new partitioned approach to image-based visual servo control. *IEEE Trans. on Robotics and Automation* 2001; **17**(4):507–515.
60. Yang Z, Shen S. Monocular visual-inertial state estimation with online initialization and camera-imu extrinsic calibration. *IEEE Transactions on Automation Science and Engineering* January 2017; **14**(1):39–51.
61. Chesi G, Hashimoto K. Static-eye against hand-eye visual servoing. *In 41st IEEE Conference on Decision and Control*, vol. 3, 2002; 2854–2859.
62. Bourdis N, Marraud D, Sahbi H. Camera pose estimation using visual servoing for aerial video change detection. *In IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, IEEE, 2012; 3459–3462.
63. Shademan A, Janabi-Sharifi F. Sensitivity analysis of EKF and iterated EKF pose estimation for position-based visual servoing. *In IEEE Conference on Control Applications (CCA'05)*, IEEE: USA, 2005; 755–760.
64. Malis E, Mezouar Y, Rives P. Robustness of image-based visual servoing with a calibrated camera in the presence of uncertainties in the three-dimensional structure. *Robotics, IEEE Transactions on* February 2010; **26**(1):112–120.
65. Chen J, Behal A, Dawson D, Dixon W. Adaptive visual servoing in the presence of intrinsic calibration uncertainty. *In 42nd IEEE Conference on Decision and Control*, vol. 5, IEEE: USA, 2003; 5396–5401.
66. Mezouar Y, Malis E. Robustness of central catadioptric image-based visual servoing to uncertainties on 3d parameters. *In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, vol. 2, IEEE: USA, 2004; 1389–1394.
67. Marchand E, Spindler F, Chaumette F. ViSP for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robotics & Automation Magazine* dec 2005; **12**(4):40–52.

TABLES

Link i	θ_i (rad)	d_i (m)	a_i (m)	α_i (rad)
1	q_1	-0.400	0.025	$\pi/2$
2	q_2	0	-0.455	0
3	q_3	0	-0.035	$-\pi/2$
4	q_4	-0.420	0	$\pi/2$
5	q_5	0	0	$-\pi/2$
6	q_6	-0.080	0	π

Table I. Denavit-Hartenberg parameters for the 6R robot.