Human-robot cooperation for robust surface treatment using non-conventional sliding mode control

Abstract

This work presents a human-robot closely collaborative solution to cooperatively perform surface treatment tasks such as polishing, grinding, deburring, etc. The method considers two force sensors attached to the manipulator end-effector and tool: one sensor is used to properly accomplish the surface treatment task, while the second one is used by the operator to guide the robot tool. The proposed scheme is based on task priority and adaptive non-conventional sliding mode control. The applicability of the proposed approach is substantiated by experimental results using a redundant 7R manipulator: the Sawyer cobot.

Keywords: Cooperative task, force feedback, sliding mode control

1. Introduction

The automation of industrial processes has generated great improvements in terms of product quality, cost reduction and operator safety and comfort. However, there are currently many industrial processes that are carried out manually

5   due to their complexity. Nowadays robotic manipulators cannot compete with the adaptability of humans and, hence, there is currently a strong tendency to combine robots and humans to collaboratively accomplish complex tasks.

In surface treatment operations the tool has to be in contact with the product surface to apply a specific treatment (polishing, deburring, grinding, etc.) and,

10   hence, the forces exerted by the tool have to be properly controlled and it should be kept perpendicular to the surface to homogenize the pressure on all contact points [1]. Due to the complexity of the shape of the product surfaces, human operators have difficulties in maintaining a constant pressure and tool

perpendicularity to the surface at all times. Moreover, when the product to be
treated is part of a production line where it is in motion or its morphology is
constantly changing, it is difficult for robots to carry out automatic treatment
operations efficiently.

To mitigate the drawbacks of both, manual and robotic automatic surface treatment, this work proposes a human-robot closely collaborative solution
which adopts the form of a human operator performing the task of ''guiding''
the tool along the object surface, whilst operating in synergy with a robotic
manipulator in charge of automatically maintaining both the tool's pressure
on the surface and the tool's perpendicularity to the surface, hence ensuring a
flexible surface treatment. For this purpose, this work uses multi-task and a
novel adaptive non-conventional sliding mode control (SMC). The SMC is used
in this work to benefit from its inherent robustness [2] and low computational
cost, i.e., only the first-order kinematics of the robot is required for the proposed
approach, as detailed below.

Moreover, the proposed approach resorts to an arrangement with two force
sensors to accomplish the collaborative operation: one sensor is used to properly accomplish the surface treatment task, i.e., to attain the desired pressure
between the tool and the surface being treated as well as to keep the tool orientation perpendicular to the surface; while the second sensor is used by the
operator to guide the robot tool along the surface to be treated. Note that the
force sensor used to accomplish the surface treatment cannot be used simultaneously to guide the robot, since the forces exerted by the human operator would
represent a disturbance in an underdetermined sensing system that would prevent the satisfactory accomplishment of the surface treatment task. This would
be the case for instance should the operator exert a force in the direction away
from the surface to be treated larger than the desired pressure for the treatment
task. In this troubled scenario the robot would end up moving away from the
object to be treated and contact would be lost.

Next, a literature review is presented about the main aspects concerning
this work: automatic surface treatment, robot guidance and SMC techniques

2

<superscript>45</superscript> for robot force control and human-robot collaboration.

Many approaches can be found in literature tackling the problem of *automatic surface treatment* using robot manipulators with force feedback. For instance, in [1] an algorithm was proposed for planning the tool location together with a compliance force control. In [3] a method for maintaining a constant
<superscript>50</superscript> polishing pressure with a numerical control polishing system was proposed by controlling the force during the process. In [4] a dual position/force control loop based on fuzzy techniques was presented for robotic grinding applications. In [5] an analytically force overshoot-free approach based on impedance control was developed to perform force-tracking. In [6] a sensor-less force control technique
<superscript>55</superscript> was proposed for a parallel machine using the information about trajectories and forces applied by skilled workers.

The *motion guidance* for robot manipulators is typically obtained via a wrist-mounted force sensor which evaluates the forces exerted by the human operator. The most commonly used method to convert these measurements into kinematic
<superscript>60</superscript> instructions to the robot is through compliance control, which establishes a direct relationship between the measured forces and the changes in the robot position [7, 8]. Yet other variants and methods can be found in the literature. For instance, in [9] a force tracking method under the impedance control framework was extended to also account for uncertain human limb dynamics. In [10] a
<superscript>65</superscript> decision-and-control architecture was proposed for hand-arm systems with "soft robotics" capabilities via dedicated human-machine interfaces. In [11] a mathematical relation between the velocity of the human-robot interaction point and the force applied by the human operator was established using impedance control for handling tasks.

<superscript>70</superscript> Other approaches tackling the problems of robot force control and human-robot collaboration are based on *SMC techniques*. Concretely, in [12] SMC is used to suppress impact forces when contacting the environment and be able to continue with a stable robot motion. In [13] a hybrid position/force control scheme was proposed using first- and second-order SMC for position and force
<superscript>75</superscript> control, respectively. In [14] an impedance control structure was proposed for

3

monitoring the contact force between the end-effector and the environment, and a model-free fuzzy SMC strategy was employed to design the position and force controllers. In [15], several methods were developed to control a prosthetic hand and the best results in terms of unwanted force overshoot were obtained using a SMC with force, position and velocity feedback. In [16] a non-singular terminal SMC was developed to ensure trajectory tracking precision for the case of a lower limb rehabilitation parallel robot. In [17] a proxy-based SMC was proposed to obtain effective tracking during normal operations for flexible joint manipulators working close to humans. In [18] a robust SMC was proposed that relied on basic information from the human subject to handle model uncertainties due to biomechanical variation of patients using an upper limb rehabilitation robot. An SMC consisting of a PID sliding surface and a fuzzy hitting control law was developed in [19] to guarantee robust tracking performance and reduce the chattering effect for a class of robot-assisted therapeutic exoskeleton. A fuzzy SMC was presented in [20] using a non-linear model for trajectory tracking of micro robots in the human vasculature system. Moreover, SMC has been used in the field of robot force control not only to improve controller robustness but also to improve force estimation by means of a sliding perturbation observer to avoid the use of expensive force sensors, e.g., see [21].

Recently, a human-robot collaboration aimed at *manual* polishing operations was presented in [22]. In this application the robot holds the workpiece and the human operator is assumed equipped with an abrasive tool to perform the polishing operation. During this process the robot keeps the workpiece in a *fixed position*, whereas the operator can change its orientation by pushing the robot body, which is detected by a force sensor mounted at the robot wrist. Note that this "static" polishing application is substantially different from the cooperative solution proposed in this work for surface treatment, whereby the robot, not the operator, is in charge of applying the surface treatment with the tool, automatically maintaining both the desired pressure and the perpendicularity of the tool to the surface, whilst the operator's task is to guide the robot tool along the surface to be treated. One of the proposed method's primary

4

virtues is the ability to treat large surfaces, e.g., polishing a car body section, for which the above work is not a fitting solution.

Furthermore, the proposed controller has several distinctive features that sets it apart from other works in the literature. In particular, the combination of two sensors attached to the manipulator end-effector with an adaptive non-conventional SMC framework is a key novelty of the proposal.

The paper is organized as follows: next section introduces some preliminaries, while Section 3 develops the required SMC theory. The proposed method for robotic surface treatment is presented in Section 4. A simulation is presented in Section 5 in order to evaluate the proposed non-conventional SMC and to compare several switching gain laws. The actual implementation of the proposed method is detailed in Section 6, while its effectiveness is substantiated by experimental results in Section 7 using a redundant 7R manipulator: the Rethink Sawyer cobot. Finally, conclusions are drawn in Section 8.

2. Preliminaries

Kinematics. The robot *pose* $\mathbf{p}$ depends on the robot *configuration* $\mathbf{q}$ as follows:

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

where $\mathbf{l}$ is the nonlinear kinematic function of the robot. The first- and second-order kinematics of the pose vector $\mathbf{p}$ result in:

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

$$\ddot{\mathbf{p}} = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}, \tag{3}$$

where $\mathbf{J}$ is the Jacobian matrix of the robot.

Robot control. This work assumes the existence of a robot controller in charge of achieving a particular joint acceleration from the commanded vector $\ddot{\mathbf{q}}_c$, and

5

that its dynamics is fast enough compared to that of $\ddot{\mathbf{q}}_c$. Hence, the relationship:

$$\ddot{\mathbf{q}} = \ddot{\mathbf{q}}_c + \mathbf{d}_c \tag{4}$$

holds approximately true, where $\mathbf{d}_c$ represents inaccuracies due to disturbances. Note that the *dynamic model* of the robot system should be taken into account to properly design the mentioned underlying joint controller.

Task-priority scheme. The task-priority strategy [23] allows to tackle several objectives simultaneously assigning an order of priority to each one. Let us consider M tasks which consist in calculating the commanded joint acceleration vector $\ddot{\mathbf{q}}_c$ to fulfill the following equality constraints:

$$\mathbf{A}_i\ddot{\mathbf{q}}_c = \mathbf{b}_i, \quad i = 1, \ldots, M, \tag{5}$$

where matrix $\mathbf{A}_i$ and vector $\mathbf{b}_i$ of the ith task are assumed known and index i represents the priority order ($i = 1$ for highest priority). The solution $\ddot{\mathbf{q}}_{c,M}$ that hierarchically minimizes the error of equations in (5) is given by [24]:

$$\ddot{\mathbf{q}}_{c,i} = \ddot{\mathbf{q}}_{c,i-1} + (\mathbf{A}_i\mathbf{N}_{i-1})^{\dagger}(\mathbf{b}_i - \mathbf{A}_i\ddot{\mathbf{q}}_{c,i-1}) \tag{6}$$

$$\mathbf{N}_i = \mathbf{N}_{i-1}(\mathbf{I} - (\mathbf{A}_i\mathbf{N}_{i-1})^{\dagger}(\mathbf{A}_i\mathbf{N}_{i-1})), \tag{7}$$

$$\text{with } i = 1, \ldots, M, \ \ddot{\mathbf{q}}_{c,0} = \mathbf{0}, \ \mathbf{N}_0 = \mathbf{I},$$

where $\mathbf{I}$ and $\mathbf{0}$ denote the identity matrix and zero column vector, respectively, superscript $\dagger$ denotes the Moore-Penrose pseudoinverse and $\ddot{\mathbf{q}}_{c,i}$ and $\mathbf{N}_i$ are the solution vector and null-space projection matrix for the set of first i tasks.

3. Sliding Mode Control

3.1. Conventional SMC to satisfy equality constraints

For conventional SMC (see Fig. 1-left) the state space of the system is divided into two regions, $\mathbf{A}$ and $\mathbf{B}$, separated by the sliding surface. The value of the

6

control action $\mathbf{u}$ when the system is in region $\mathbf{A}$ is such that it "pushes" the system into region $\mathbf{B}$, namely $\mathbf{u}_B$. Analogously, when the system is in region $\mathbf{B}$,

135 the value of control action is such that it pushes the system into region $\mathbf{A}$, namely $\mathbf{u}_A$. Hence, regardless of whether the system starts in region $\mathbf{A}$ - $\mathbf{x}_A(0)$, or $\mathbf{B}$ - $\mathbf{x}_B(0)$, it evolves to the sliding surface in what is referred as *reaching mode* [2]. Once the system has reached the sliding surface, the system is kept on it by a control action $\mathbf{u}$ that switches between $\mathbf{u}_A$ and $\mathbf{u}_B$ at a theoretically infinite

140 frequency, which is known as *sliding mode* (SM) [2]. A *continuous equivalent control* [25] can be obtained for the SM phase. Therefore, SMC produces such control action without explicitly computing it and with low computational cost.

The theorem below presents a conventional SMC designed to satisfy equality constraints.

Theorem 1. *Consider the following dynamical system:*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{d}) + \mathbf{g}(\mathbf{x})\,\mathbf{u}, \tag{8}$$

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

Consider also that the state vector $\mathbf{x}$ is subject to equality *constraints $\phi_{eq,i}(\mathbf{x}) = 0$, $i = 1, \ldots, N_{eq}$, where $\phi_{eq,i}(\mathbf{x})$ is the ith equality constraint function. Thus, the region Φ_{eq} compatible with the constraints is given by:*

$$\Phi_{eq} = \{\mathbf{x} \mid \phi_{eq,i}(\mathbf{x}) = 0\}, \quad i = 1, \ldots, N_{eq}. \tag{9}$$

Then, assuming that the constraint functions $\phi_{eq,i}$ are differentiable, the control action $\mathbf{u}$ that fulfills the variable structure control below guarantees that the system converges to Φ_{eq} in finite time and remains there henceforth:

$$\mathbf{L_g}\phi_{eq}\mathbf{u} = -\mathbf{W}_{eq}sign(\phi_{eq})\,u_{eq}^+ \tag{10}$$

$$u_{eq}^+ > \|L_f\phi_{eq}\|_1 \big/ diag_{\min}(\mathbf{W}_{eq}), \tag{11}$$

where $\boldsymbol{\phi}_{eq}$ is a column vector with all the constraint functions $\phi_{eq,i}$, the scalar $L_f\phi_{eq,i} = \frac{\partial \phi_{eq,i}^T}{\partial \mathbf{x}}\mathbf{f}$ and the row vector $\mathbf{L_g}\phi_{eq,i} = \frac{\partial \phi_{eq,i}^T}{\partial \mathbf{x}}\mathbf{g}$ denote the Lie derivatives of $\phi_{eq,i}(\mathbf{x})$ in the direction of vector field $\mathbf{f}$ and in the direction of the set of vector fields $\mathbf{g}$, respectively, column vector $L_f\boldsymbol{\phi}_{eq}$ contains the elements $L_f\phi_{eq,i}$ of all equality constraints, matrix $\mathbf{L_g}\boldsymbol{\phi}_{eq}$ contains the row vectors $\mathbf{L_g}\phi_{eq,i}$ of all equality constraints, $sign(\cdot)$ represents the sign function, positive scalar u_{eq}^+ is the so-called switching gain, $\mathbf{W}_{eq}$ is a diagonal matrix representing the switching gain weights for the constraints, $\|\cdot\|_1$ represents the 1-norm norm and function $diag_{\min}(\cdot)$ computes the minimum value of the diagonal elements of a matrix.

Proof. The proof can be obtained straightforward from the Proof 2.1 in [25] and its generalization. Details omitted for brevity. $\qquad\square$

3.2. One-side SMC to satisfy inequality constraints

The one-side SMC proposed in this work is graphically represented in Fig. 1-right and is used to satisfy inequality constraints. In this case, the state space of the system is divided into the non-allowed region $\mathbf{A}$ and the allowed region $\mathbf{B}$, which are separated by the constraint boundary. Similarly to conventional SMC, when the system sate is in the non-allowed region $\mathbf{A}$ the control action $\mathbf{u} = \mathbf{u}_B$ pushes the system into the allowed region $\mathbf{B}$. But in contrast to conventional SMC, when the system state is in the allowed region $\mathbf{B}$, no control action is applied, i.e., $\mathbf{u} = \mathbf{0}$. Hence, if the system starts in region $\mathbf{A}$, i.e., $\mathbf{x}_A(0)$, it evolves in reaching mode to the sliding surface. Nevertheless, when the system starts in the allowed region $\mathbf{B}$, the system state can "freely" evolve according to some other criterion, e.g., a control law for reference tracking. Therefore, only when the state trajectory tries by itself to leave the allowed region, the one-side SMC will make $\mathbf{u}$ switch between $\mathbf{0}$ and $\mathbf{u}_B$ at a theoretically infinite frequency, which can be seen as an ideal SM behavior [2]. The theorem below presents the one-side SMC proposed to satisfy inequality constraints.

Theorem 2. *Considering the system* (8) *and that the state vector $\mathbf{x}$ is subject to* inequality *constraints $\phi_{in,i}(\mathbf{x}) \leq 0$, $i = 1, \dots, N_{in}$, where $\phi_{in,i}(\mathbf{x})$ is the ith*

8

constraint function, the region Φ_{in} compatible with the constraints is given by:

$$\Phi_{in} = \{\mathbf{x} \,|\, \phi_{in,i}(\mathbf{x}) \leq 0\}\,, \quad i = 1, \ldots, N_{in}. \tag{12}$$

Then, assuming that the constraint functions $\phi_{in,i}$ are differentiable, the control action $\mathbf{u}$ that fulfills the variable structure control below guarantees that the system converges to Φ_{in} in finite time and remains there henceforth:

$$v2dm\left(pos\left(\boldsymbol{\phi}_{in}\right)\right)\mathbf{L_g}\boldsymbol{\phi}_{in}\mathbf{u} = -\mathbf{W}_{in}\,pos\left(\boldsymbol{\phi}_{in}\right)\,u_{in}^+ \tag{13}$$

$$u_{in}^+ > \sum_{i=1}^{n_a}\left(\max(L_f\phi_{in,i}, 0)\right)/diag_{\min}(\mathbf{W}_{in}), \tag{14}$$

where function $v2dm(\cdot)$ converts a vector into a diagonal matrix[1], function $pos(\cdot)$ represents the positive *function (i.e., $pos(x) = 0$ if $x < 0$ and $pos(x) = 1$ otherwise), $\boldsymbol{\phi}_{in}$ is a column vector with all the inequality constraint functions $\phi_{in,i}$, matrix $\mathbf{L_g}\boldsymbol{\phi}_{in}$ contains the row vectors $\mathbf{L_g}\phi_{in,i}$ of all inequality constraints, the scalar $L_f\phi_{in,i} = \frac{\partial\phi_{in,i}^T}{\partial\mathbf{x}}\mathbf{f}$ and the row vector $\mathbf{L_g}\phi_{in,i} = \frac{\partial\phi_{in,i}^T}{\partial\mathbf{x}}\mathbf{g}$ denote the Lie derivatives of the inequality constraints in the direction of vector field $\mathbf{f}$ and in the direction of the set of vector fields $\mathbf{g}$, respectively, positive scalar u_{in}^+ is the switching gain, $\mathbf{W}_{in}$ is a diagonal matrix representing the switching gain weights and n_a is the number of active inequality constraints, i.e., those with $\phi_{in,i} \geq 0$.*

Proof. The inequality constraint vector is partitioned into two subvectors $\boldsymbol{\phi}_{in} = [\boldsymbol{\phi}_{in}^{n_a\,\mathrm{T}}\quad\boldsymbol{\phi}_{in}^{N_{in}-n_a\,\mathrm{T}}]^{\mathrm{T}}$, where the first one is composed of the n_a active inequality constraints and the second one of the remaining non-active constraints. Assuming that $\boldsymbol{\phi}_{in}^{n_a}(0) > \mathbf{0}$, the goal of this proof is to show that convergence to point $\boldsymbol{\phi}_{in}^{n_a} = \mathbf{0}$ is achieved in finite time. The column vector $\dot{\boldsymbol{\phi}}_{in}$ composed of the constraint function derivatives $\dot{\phi}_{in,i}$ is given by

$$\dot{\boldsymbol{\phi}}_{in} = \frac{\partial\boldsymbol{\phi}^{\mathrm{T}}}{\partial\mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{d}) + \frac{\partial\boldsymbol{\phi}^{\mathrm{T}}}{\partial\mathbf{x}}\mathbf{g}(\mathbf{x})\,\mathbf{u} = L_f\boldsymbol{\phi}_{in} + \mathbf{L_g}\boldsymbol{\phi}_{in}\mathbf{u}. \tag{15}$$

---

[1]The expression v2dm(pos($\boldsymbol{\phi}_{in}$)) on the left-side of (13) is used to obtain the trivial scalar equation $0 = 0$ for the non-active inequality constraints, i.e., those with $\phi_{in,i} < 0$.

Premultiplying (15) by v2dm $(\mathrm{pos}\,(\boldsymbol{\phi}_{in}))$ and substituting (13) yields:

$$\mathrm{v2dm}(\mathbf{z}_{in})\dot{\boldsymbol{\phi}}_{in} = \mathrm{v2dm}(\mathbf{z}_{in})L_f\boldsymbol{\phi}_{in} - \mathbf{W}_{in}\,\mathbf{z}_{in}\,u_{in}^+, \tag{16}$$

where $\mathbf{z}_{in}$ is a column vector with the ith-component $z_{in,i} = 1$ if $\phi_{in,i} > 0$ and $z_{in,i} = 0$ if $\phi_{in,i} < 0$. Let $V_{in} = \mathbf{z}_{in}^{\mathrm{T}}\,\mathrm{v2dm}(\mathbf{z}_{in})\,\boldsymbol{\phi}_{in}$ be a Lyapunov function candidate. Vector $\boldsymbol{\phi}_{in}^{n_a}$ can be generically partitioned into two subvectors $\boldsymbol{\phi}_{in}^{n_a} = [\boldsymbol{\phi}_{in}^{b\,\mathrm{T}} \quad \boldsymbol{\phi}_{in}^{n_a-b\,\mathrm{T}}]^{\mathrm{T}}$, where SM occurs in the manifold given by $\boldsymbol{\phi}_{in}^b = \mathbf{0}$, whereas the components of vector $\boldsymbol{\phi}_{in}^{n_a-b}$ are greater than zero. Since vectors $\mathbf{z}_{in}^{n_a-b} = \mathbf{1}$ and $\mathbf{z}_{in}^{N_{in}-n_a} = \mathbf{0}$ are constant, the time derivative of V results in:

$$\dot{V}_{in} = \frac{d}{dt}\left(\left(\begin{bmatrix} \mathbf{z}_{in}^b \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix}^{\mathrm{T}} \mathrm{v2dm}\left(\begin{bmatrix} \mathbf{z}_{in}^b \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix}\right)\right)\begin{bmatrix} \mathbf{0} \\ \boldsymbol{\phi}_{in}^{n_a-b} \\ \boldsymbol{\phi}_{in}^{N_{in}-n_a} \end{bmatrix} + \mathbf{z}_{in}^{\mathrm{T}}\,\mathrm{v2dm}(\mathbf{z}_{in})\,\dot{\boldsymbol{\phi}}_{in}\right.$$
$$= \mathbf{z}_{in}^{\mathrm{T}}\,\mathrm{v2dm}(\mathbf{z}_{in})\,\dot{\boldsymbol{\phi}}_{in}. \tag{17}$$

Substituting (16) in (17) yields:

$$\dot{V}_{in} = \mathbf{z}_{in}^{\mathrm{T}}\,\mathrm{v2dm}(\mathbf{z}_{in})\,L_f\boldsymbol{\phi}_{in} - \mathbf{z}_{in}^{\mathrm{T}}\,\mathbf{W}_{in}\,\mathbf{z}_{in}\,u_{in}^+. \tag{18}$$

Since $\mathbf{z}_{in}^{N_{in}-n_a} = \mathbf{0}$ and the components of vector $\mathbf{z}_{in}^{n_a}$ range from 0 to 1, the upper bound of the first term in (18) is given by $z_{in,i}^{n_a} = 1$ if $L_f\phi_{in,i}^{n_a} > 0$ and $z_{in,i}^{n_a} = 0$ if $L_f\phi_{in,i}^{n_a} < 0$, that is:

$$\mathbf{z}_{in}^{\mathrm{T}}\,\mathrm{v2dm}(\mathbf{z}_{in})\,L_f\boldsymbol{\phi}_{in} \leq \sum_{i=1}^{n_a}\left(\max(L_f\phi_{in,i},0)\right). \tag{19}$$

Assuming that $u_{in}^+ > 0$, the second term in (18) is negative, since matrix $\mathbf{W}_{in}$ is positive definite, and its upper bound is given by:

$$-\mathbf{z}_{in}^{\mathrm{T}}\,\mathbf{W}_{in}\,\mathbf{z}_{in}\,u_{in}^+ \leq -\mathrm{diag}_{\min}(\mathbf{W}_{in})\,\|\mathbf{z}_{in}\|_2^2\,u_{in}^+, \tag{20}$$

where $\|\mathbf{z}_{in}\|_2 \geq 1 \; \forall \; \boldsymbol{\phi}_{in} > \mathbf{0}$, because if vector $\boldsymbol{\phi}_{in}^{n_a-b}$ is not empty at least one component of vector $\mathbf{z}_{in}$ is equal to 1. From (19) and (20), the upper bound of the time derivative of the Lyapunov function V results in:

$$\dot{V}_{in} \leq \sum_{i=1}^{n_a} \left(\max(L_f \phi_{in,i}, 0) \right) - \mathrm{diag_{min}}(\mathbf{W}_{in}) \, u_{in}^+. \tag{21}$$

Therefore, if u_{in}^+ fulfills (14) the Lyapunov function decays at a finite rate, it vanishes and collective SM in the intersection of the n_a active inequality constraints occurs after a finite time interval.

$\square$

3.3. Modified constraints

Approaching the constraints at high speed is not advisable because, in general, large joint accelerations $\ddot{\mathbf{q}}$ would be required to slow down the robot motion in order to keep it on the constraint manifold. Therefore, the actual constraints $\sigma_{eq,i}$ and $\sigma_{in,i}$ will be modified to include the speed of movement as follows:

$$\phi_{eq,i} = \sigma_{eq,i} + K_{eq,i} \dot{\sigma}_{eq,i} = 0 \tag{22}$$

$$\phi_{in,i} = \sigma_{in,i} + K_{in,i} \dot{\sigma}_{in,i} \leq 0, \tag{23}$$

where $K_{eq,i}$ and $K_{in,i}$ are free design parameters that determine the rate of approach to the constraint manifold and constraint boundary, respectively.

3.4. Chattering

Discrete-time implementations of the proposed SMC makes the system leave the ideal SM and oscillate with finite frequency and amplitude inside a band around $\boldsymbol{\phi} = \mathbf{0}$, which is called *chattering* [2]. The upper bound for the chattering band $\triangle \boldsymbol{\phi}$ of the proposal can be obtained using the Euler-integration of the discontinuous control action given by Eq. (10), that is:

$$\triangle \boldsymbol{\phi} = T_s \, |\mathbf{L_g} \boldsymbol{\phi} \, \mathbf{u}| = T_s \, u^+ \, \mathrm{diag}(\mathbf{W}), \tag{24}$$

11

where T_s is the sampling time of the robot system and function $\text{diag}(\cdot)$ gives a column vector with the diagonal elements of a square matrix.

3.5. Adaptive sliding mode control

Since the switching gain u^+ can be varying in time, a common option consists in using an *adaptive switching gain* (ASG) in order to minimize its value online and, thus, the control effort and chattering amplitude are reduced.

3.5.1. Adaptive switching gain for the conventional sliding mode control.

Among the different ASG laws in the literature for conventional SMC, this work considers the following discrete algorithm proposed in [26]:

$$u^+(k) = |u^+(k-1) + T_s\, \mu_{eq}\, \text{sign}(\phi_{eq}(k))\, \text{sign}(\phi_{eq}(k-1))|, \qquad (25)$$

where $u^+(k)$ and $u^+(k-1)$ are the values of the switching gain for the current and the previous time steps, respectively, $\phi_{eq}(k)$ and $\phi_{eq}(k-1)$ are values of the equality constraint function for the current and the previous time steps, respectively, and μ_{eq} is a positive configuration parameter that determines the speed of the adaptation. Since the SMC consists in a first-order control law, the SM surface should be crossed in every successive time step and, hence, the ASG law in (25) increases the switching gain if the SM surface has not been crossed and decreases it otherwise. Thus, the method aims to lead the system to cross the SM surface while minimizing the switching gain.

3.5.2. Adaptive switching gain for the one-side sliding mode control.

In the same spirit of the ASG discrete algorithm (25), this work proposes the following ASG law for the one-side SMC:

$$u^+(k) = |u^+(k-1) + T_s\, \mu_{in}\, f_{ASG}(\phi_{in}(k), \phi_{in}(k-1))|, \qquad (26)$$

where μ_{in} is the speed adaptation parameter and $f_{ASG}(A, B)$ is equal to: 1 if $B > 0$ and $A > 0$; -1 if $B > 0$ and $A < 0$; and 0 for the remaining cases.

12

This ASG law is explained as follows: when the inequality constraint is unfulfilled at a given time step (i.e., $\phi_{in}(k-1) > 0$) the switching gain is decreased or increased depending on whether the constraint boundary is successfully crossed or not at the next time step. Hence, the method aims to lead the system to cross the boundary of the inequality constraint whenever it is unfulfilled while minimizing the switching gain. Thus, starting from any initial point in the non-allowed region, the system crosses the constraint boundary in finite time and, for constant conditions, the switching gain given by (26) converges to a bounded region. The proof can be obtained straightforward considering the extension described in [26] with $\rho^+(k) = u^+$ and $\rho^-(k) = 0$. Details omitted for brevity.

4. Proposed approach

The objective of this work is to obtain a robot control using the theory in Section 3 so that robot and human operator cooperatively perform a surface treatment task. For this purpose, some coordinates of the robot pose are controlled automatically to maintain the desired pressure between the robot tool and the object's surface as well as to assure that the tool is perpendicular to this surface by using a force sensor, namely *treatment sensor*, attached to the end of the robot tool. Meanwhile, other coordinates of the robot pose are controlled by the human operator in order to guide the polishing tool along the object's surface using a second force sensor, namely *guide sensor*, attached to some place of the robot end-effector.

4.1. System tasks

Four tasks with different priority levels are considered in this work:

- The first level (high-priority task) includes the *equality constraints* that must be satisfied at all times to properly accomplish the treatment on the surface with the robot tool.

13

- The second level (medium-high-priority task) includes the *inequality constraints* required for the robot to track the human operator's forces using the guide sensor.

- The third level (medium-low-priority task) is used to reduce to zero the speed of the robot tool when no operator's forces is detected by the guide sensor.

- The fourth level (low-priority task) is considered only for the case of *redundant* robots in order to keep the robot close to the *home* configuration.

4.2. Lie derivatives

In order to use the theory in Section 3, a dynamical system in the form of Eq. (8) is considered with the state vector $\mathbf{x} = \begin{bmatrix} \mathbf{q}^{\mathrm{T}} & \dot{\mathbf{q}}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$, the disturbance vector $\mathbf{d} = \mathbf{d}_c$ and the input vector $\mathbf{u} = \ddot{\mathbf{q}}_c$. Hence, the model is a double integrator, and from (4) the state equation results in:

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{O} & \mathbf{I} \\ \mathbf{O} & \mathbf{O} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \mathbf{0} \\ \mathbf{d}_c \end{bmatrix} + \begin{bmatrix} \mathbf{O} \\ \mathbf{I} \end{bmatrix} \mathbf{u}, \tag{27}$$

and, therefore, the Lie derivatives for the constraint function ϕ_i are given by:

$$\mathbf{L_g}\phi_i = \nabla\phi_i^{\mathrm{T}}\,\mathbf{g} = (\partial\phi_i/\partial\dot{\mathbf{q}})^{\mathrm{T}} \tag{28}$$

$$L_f\phi_i = \nabla\phi_i^{\mathrm{T}}\,\mathbf{f} = (\partial\phi_i/\partial\mathbf{q})^{\mathrm{T}}\,\dot{\mathbf{q}} + (\partial\phi_i/\partial\dot{\mathbf{q}})^{\mathrm{T}}\,\mathbf{d}_c. \tag{29}$$

4.3. Force model

The constraints for the first two levels are defined below depending on the vectors $\mathbf{F}_t$ and $\mathbf{F}_g$ of forces and torques measured by the treatment and guide sensors, respectively. In many applications, the interaction forces $\mathbf{F}$ between the robot/tool and the environment can be approximated by the ideal elastic model below [27]:

$$\mathbf{F} = \mathbf{K}_s\,\Delta\mathbf{s}(\mathbf{q}, \mathbf{p}_s) = \begin{bmatrix} F_x & F_y & F_z & F_\alpha & F_\beta & F_\gamma \end{bmatrix}^{\mathrm{T}}, \tag{30}$$

14

where vector $\mathbf{F}$ is relative to the tool coordinate system, $\mathbf{K}_s$ is a diagonal matrix with the stiffness coefficients for each tool axis and vector $\Delta\mathbf{s}$ is the mechanical deformation of the sensor relative to the tool coordinate system, which depends on the robot configuration $\mathbf{q}$ and the position and orientation $\mathbf{p}_s$ of the surface of the environment.

4.4. Level 1: Constraints for the surface treatment task

Three equality constraints are defined for the surface treatment as follows:

$$\sigma_{1,z}(\mathbf{F}_t) = \sigma_{1,z}(\mathbf{q}, t) = F_{t,z} - F_{z,ref} = 0 \tag{31}$$

$$\sigma_{1,\alpha}(\mathbf{F}_t) = \sigma_{1,\alpha}(\mathbf{q}, t) = F_{t,\alpha} = 0 \tag{32}$$

$$\sigma_{1,\beta}(\mathbf{F}_t) = \sigma_{1,\beta}(\mathbf{q}, t) = F_{t,\beta} = 0, \tag{33}$$

where $F_{t,z}$ is the linear force measured by the treatment sensor in the tool Z-axis, $F_{t,\alpha}$ and $F_{t,\beta}$ are the angular forces (i.e., torques) measured by the treatment sensor in the tool X- and Y-axes, and $F_{z,ref}$ is the desired force between the tool and the surface being treated in the tool Z-axis. Hence, the first equality constraint is used to attain the desired force $F_{z,ref}$ between the tool and the surface, whereas the last two equality constraints are used to keep the tool orientation perpendicular to the surface, since the torques in X- and Y-axes are zero if the tool is perfectly perpendicular to the surface.

Taking into account (22), (28) and (30)–(33), the Lie derivative $\mathbf{L_g}\phi_1$ required for the SMC in (10) is given by:

$$
\begin{aligned}
\mathbf{L_g}\phi_1 &= (\partial\phi_1/\partial\dot{\mathbf{q}})^{\mathrm{T}} = \mathbf{K}_1\,(\partial\boldsymbol{\sigma}_1/\partial\mathbf{q})^{\mathrm{T}} \\
&= \mathbf{K}_1
\begin{bmatrix}
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0
\end{bmatrix}
\mathbf{K}_{t,s}\mathbf{J}_n = \mathbf{K}_1\mathbf{H}_1\mathbf{K}_{t,s}\mathbf{J}_n,
\end{aligned}
\tag{34}
$$

where $\boldsymbol{\sigma}_1$ is a column vector composed of all equality constraints $\sigma_{1,i}$, $\mathbf{K}_1$ is a diagonal matrix composed of all the approaching parameters $K_{eq,i}$, see (22), $\mathbf{K}_{t,s}$

15

is the stiffness diagonal matrix for the treatment sensor and $\mathbf{J}_n$ is the geometric Jacobian relative to the tool coordinate system [27], i.e., the Jacobian matrix relating the joint velocities $\dot{\mathbf{q}}$ and the linear and angular velocities of the end-effector relative to the tool coordinate system.

Since the stiffness coefficients $\mathbf{K}_{t,s}$ in $\mathbf{L_g}\phi_1$ (34) may not be known, they can be included without loss of generality in the switching gain weight matrix $\mathbf{W}_1$, so that the conventional SMC given by (10) is modified as follows:

$$\mathbf{K}_1\mathbf{H}_1\mathbf{J}_n\ddot{\mathbf{q}}_c = -\overline{\mathbf{W}}_1\text{sign}(\phi_1)\, u_1^+ \;\;\rightarrow\;\; \mathbf{A}_1\ddot{\mathbf{q}}_c = \mathbf{b}_1, \tag{35}$$

where $\mathbf{A}_1$ and $\mathbf{b}_1$ are the matrix and vector for the first task in (5) and:

$$\overline{\mathbf{W}}_1 = \begin{bmatrix} W_{1,z}/K_{t,s,z} & 0 & 0 \\ 0 & W_{1,\alpha}/K_{t,s,\alpha} & 0 \\ 0 & 0 & W_{1,\beta}/K_{t,s,\beta} \end{bmatrix} = \begin{bmatrix} \overline{W}_{1,z} & 0 & 0 \\ 0 & \overline{W}_{1,\alpha} & 0 \\ 0 & 0 & \overline{W}_{1,\beta} \end{bmatrix}. \tag{36}$$

4.5. Level 2: Constraints to track human operator's forces

The following constraint is used to track the human operator's forces:

$$\sigma_2(\mathbf{F}_g) = \sigma_2(\mathbf{q}, t) = \sqrt{F_{g,x}^2 + F_{g,y}^2} - F_{l,0} = F_l - F_{l,0} \leq 0, \tag{37}$$

where $F_{g,x}$ and $F_{g,y}$ are the linear forces detected by the guide sensor in its X- and Y-axes, which are perpendicular to the robot end-effector, F_l is the magnitude of these linear forces and $F_{l,0}$ is a threshold so that the constraint becomes active when the magnitude F_l is larger than this threshold, in which case the robot tool is moved in the direction of the detected forces in order to fulfill the constraint.

Taking into account (23), (28) and (37), the Lie derivative $\mathbf{L_g}\phi_2$ for the

16

above constraint, which is required for the one-side SMC in (13), is given by:

$$\mathbf{L_g}\phi_2 = (\partial\phi_2/\partial\dot{\mathbf{q}})^{\mathrm{T}} = K_2\,(\partial\sigma_2/\partial\mathbf{q})^{\mathrm{T}}$$

$$= K_2\begin{bmatrix}\overline{F}_{g,x} & \overline{F}_{g,y} & 0 & 0 & 0 & 0\end{bmatrix}\mathbf{K}_{g,s}\mathbf{J}_n = K_2\mathbf{H}_2\mathbf{K}_{g,s}\mathbf{J}_n, \qquad (38)$$

where K_2 is the approaching parameter to the original constraint (37), see (23), $\overline{F}_{g,i} = F_{g,i}/F_l$ represents the normalized linear force in the i axis and $\mathbf{K}_{g,s}$ is the stiffness diagonal matrix for the guide sensor.

The acceleration equality for the second level results in:

$$\mathrm{v2dm}\,(\mathrm{pos}\,(\phi_2))\,\mathbf{L_g}\phi_2\ddot{\mathbf{q}}_c = -W_2\,\mathrm{pos}\,(\phi_2)\;u_2^+, \qquad (39)$$

where W_2 and u_2^+ are the switching gain weight and switching gain, respectively, for Level 2 (in this scalar case the switching gain weight could be omitted).

Since the stiffness coefficients $\mathbf{K}_{g,s}$ in $\mathbf{L_g}\phi_2$ (38) may not be known, they can be included in the switching gain weight W_2 in (39) as follows:

$$\mathrm{v2dm}\,(\mathrm{pos}\,(\phi_2))\,K_2\mathbf{H}_2\mathbf{J}_n\ddot{\mathbf{q}}_c = -\overline{W}_2\mathrm{pos}(\phi_2)\,u_2^+ \quad \rightarrow \quad \mathbf{A}_2\ddot{\mathbf{q}}_c = b_2, \qquad (40)$$

where $\mathbf{A}_2$ and b_2 are the values for the second task in (5) and:

$$\overline{W}_2 = W_2/K_{s,l}, \qquad (41)$$

where it has been assumed the same stiffness coefficient $K_{s,l}$ for both linear coordinates, i.e., $K_{s,l} = K_{g,s,x} = K_{g,s,y}$ and, hence, the linear motion of the robot tool given by (40) is in the same direction as the human operator's forces.

4.6. Level 3: Speed reduction

The following equality is considered to reduce to zero the tool speed:

$$\mathbf{J}\ddot{\mathbf{q}}_c = -\,K_{3,v}\dot{\mathbf{p}} - \mathrm{sign}(\dot{\mathbf{p}})u_3^+ \quad \rightarrow \quad \mathbf{A}_3\ddot{\mathbf{q}}_c = \mathbf{b}_3, \qquad (42)$$

17

where the tool speed $\dot{\mathbf{p}}$ is obtained from the first-order kinematics (2), $K_{3,v}$ is the velocity correction gain of the continuous term in the above control law, u_3^+ is the switching gain of the conventional SMC used in the second term, and $\mathbf{A}_3$ and $\mathbf{b}_3$ are the matrix and vector for the third task in (5). Basically, the above control law uses the first continuous term for speed reduction when $\dot{\mathbf{p}} \neq \mathbf{0}$ and the second switching term to compensate the term $\dot{\mathbf{J}}\dot{\mathbf{q}}$ of the robot second-order kinematics (3) when $\dot{\mathbf{p}} = \mathbf{0}$. This hybrid control law has several advantages: the time derivative of the robot Jacobian is not required; the switching gain u_3^+ can be relatively small, reducing the chattering effects, while a fast speed convergence to zero is obtained due to the continuous correction term.

4.7. Level 4: Home configuration

This level applies only for *redundant* robots, since otherwise there are no remaining degrees of freedom at this level. Among the different available options in the literature, this work considers "pushing" the robot to a home configuration $\mathbf{q}_0$ to avoid a bias robot self-motion which may lead achieving critical areas, joint limits, etc. For this purpose, the following equality is considered:

$$\ddot{\mathbf{q}}_c = -K_{4,v}\dot{\mathbf{q}} + K_{4,p}(\mathbf{q}_0 - \mathbf{q}) \;\; \rightarrow \;\; \mathbf{A}_4\ddot{\mathbf{q}}_c = \mathbf{b}_4, \tag{43}$$

where $K_{4,v}$ and $K_{4,p}$ are the gains used for the velocity and position corrections, respectively, and $\mathbf{A}_4$ and $\mathbf{b}_4$ are the matrix and vector for the fourth task in (5).

4.8. Additional remarks

Control action. In this work the joint accelerations are considered as the SM discontinuous control action, which yields two advantages: the joint velocities are continuous (smoother control) and it allows to reach smoothly the constraints manifold. If the actual control action are the joint velocities (or positions), a pure single (or double) integrator can be applied to the discontinuous control signal to compute the actual continuous control action.

Time derivatives. The method requires the time derivatives $\dot{\mathbf{F}}_t$ and $\dot{\mathbf{F}}_g$ for the SMC in the first and second levels and $\dot{\mathbf{q}}$ for the tasks in the remaining levels. The simplest way to deal with this issue consists in using numerical differentiation. However, some kind of filtering should be previously applied to the actual variable when non-negligible noise is present. Note that the low-pass filter used for noise reduction must not limit the bandwidth of the control law.

4.9. Advantages of the proposed method

The first advantage of the proposed multitask SMC is *complementarity*: one task is in charge of controlling the robot force to accomplish the surface treatment task, while another task is in charge of moving the robot along the surface according to the human operator's forces. Other advantages of the proposed method compared to traditional approaches are:

- *Smoothness*: firstly, the joint velocities are continuous since the SM control action are the joint accelerations; and, secondly, the boundary of the original constraints is reached progressively depending on a free design parameter.

- *Robustness*: the SMC algorithm is robust against the Lie derivatives $L_f\phi_i$ since they are collinear [2] with the discontinuous control action. Therefore, it is not affected by the terms included in $L_f\phi_i$, such as: the inaccuracies $\mathbf{d}_c$ of the low-level control loop; the pose $\mathbf{p}_s$ of the robot environment (i.e., the surface being treated and the human operator) and its derivative; the time derivative of the Jacobian matrix; etc.

- *Low computational cost*: Only partial information of the system model is used, i.e., the Lie derivatives $L_f\phi_i$ are not needed (see the terms listed above), only the Lie derivatives $\mathbf{L_g}\phi_i$ are required. In particular, the conventional and one-side SMCs in Level 1 and Level 2 only require the robot Jacobian and the constraint functions ϕ_i, which are computed from the measurements $\mathbf{F}_t$ and $\mathbf{F}_g$ of the treatment and guide sensors. Hence,

19

the proposed approach only requires a few program lines and has reduced
computation time, see Section 6.

5. Evaluation of the proposed control method

In this section, the main features of the proposed adaptive conventional and
one-side SMCs are illustrated by simulating a DC motor model, as it allows
to better highlighting the merits of the proposed controller with a traditional
example. Note that this model represents one joint of the robot system and
can be easily extrapolated to a given number of joints, as considered in the real
experimentation of Section 7. The simulation results presented in this section
were obtained using MATLAB$^{\circledR}$.

The DC motor model is given by:

$$\dot{\omega} = (K_{m,t}\, i_m - b_m\, \omega)/J_m \tag{44}$$

$$\dot{i}_m = (-R_m\, i_m + V_m - K_{m,e}\, \omega)/L_m, \tag{45}$$

were ω is the rotor angular velocity, i_m the armature current, V_m the voltage
source, J_m the moment of inertia of the rotor, b_m the motor viscous friction con-
stant, $K_{m,e}$ the electromotive force constant, $K_{m,t}$ the motor torque constant,
R_m the electric resistance and L_m the electric inductance.

For this model, the output of the system will be the rotor speed ω and the
input or control action the voltage V_m.

In order to track the reference velocity ω_{ref} the equality and inequality
constraints for the conventional and one-side SMC are defined in terms of the
error $e = \omega_{ref} - \omega$ as follows:

$$\sigma_{eq,m}(\omega) = e = 0, \qquad \phi_{eq,m}(\omega,\dot{\omega}) = \sigma_{eq,m} + K_{eq,m}\dot{\sigma}_{eq,m} = 0 \tag{46}$$

$$\sigma_{in,m}(\omega) = |e| - e_{\max} \leq 0, \qquad \phi_{in,m}(\omega,\dot{\omega}) = \sigma_{in,m} + K_{in,m}\dot{\sigma}_{in,m} \leq 0, \tag{47}$$

where $K_{eq,m}$ and $K_{in,m}$ are the approaching parameters to the original con-
straints and $e_{\max}$ is the maximum allowed tracking error considered for the

20

inequality constraints. Note that the relative degree between the constraint functions $\{\phi_{eq,m}, \phi_{in,m}\}$ and the control action V is equal to one, as required by SM control theory.

Therefore, the Lie derivative $L_g\phi$ and the control action V for the conventional and one-side SMCs are given by:

$$L_g\phi_{eq,m} = K_{eq,m}, \qquad\qquad V = K_{eq,m}^{-1}\text{sign}(\phi_{eq,m})u^+ \qquad (48)$$

$$L_g\phi_{in,m} = K_{in,m}\text{sign}(e), \qquad V = (K_{in,m}\text{sign}(e))^{-1}\text{pos}(\phi_{in,m})u^+. \qquad (49)$$

5.1. Results for the conventional sliding mode control

The following ASG laws in the literature are considered for comparison:

M1) Method in [28]:

$$u^+(t) = u^+(t_0) + K_{int}\int_{\tau=t_0}^{t}\|\phi\|\,d\tau, \qquad (50)$$

where $u^+(t_0)$ is the initial SMC switching gain and K_{int} is a positive configuration parameter that determines the speed of the adaptation.

M2) Method in [29]:

$$u^+(t) = \left|u^+(t_0) + K_{int}\int_{\tau=t_0}^{t}(\|\phi\| - \psi)\,d\tau\right|, \qquad (51)$$

where ψ is a positive constant lower than the SM band $\triangle\phi$.

M3) Method in [30]:

$$u^+(t) = \eta\left(e^{\|\phi\|} - 1\right) + K_{int}\int_{\tau=t_0}^{t}(\|\phi\| - \psi)\,d\tau, \qquad (52)$$

where η is a positive configurable parameter.

M4) Method in [31]:

$$u^+(t) = \eta\left(e^{\|\phi\|} - 1\right) + K_{int}\int_{\tau=t_0}^{t}\|\phi\|\,\text{sign}\left(\|\phi\| - \psi\right)d\tau. \qquad (53)$$

21

M5) Discrete method in [26], see (25)

The simulation was run under the following conditions: $T_s = 1\text{ms}$, $J_m = 0.1 \text{ kg.m}^2$, $b_m = 0.1$ N.m.s, $K_{m,e} = 0.01$ V.s, $K_{m,t} = 0.01$ N.m/A, $R_m = 1$ Ω, $L_m = 0.5$ H, $K_{eq,m} = 0.1$ and the switching gain parameters are:

- FSG (fixed switching gain): $u^+ = 100$

- M1: $u^+(t_0) = 100$, $K_{int} = 500$

- M2: $u^+(t_0) = 100$, $K_{int} = 500$, $\psi = 0.01$

- M3: $\eta = 160$, $K_{int} = 1000$, $\psi = 0.01$

- M4: $\eta = 160$, $K_{int} = 1000$, $\psi = 0.01$

- M5: $u^+(k_0) = 100$, $\mu_{eq,m} = 500$.

For the simulation, the reference velocity ω_{ref} consists of two segments (see the bottom plot in Fig. 2): for the first segment the reference is a sinusoidal wave $\omega_{ref}(t) = 0.5 + 0.5\sin(1.33\pi t)$ rad/s, whereas for the second segment the reference velocity is a constant value of 0.5 rad/s. Fig. 2 shows that once the initial error of both segments has been corrected, only the discrete method M5 properly modifies the switching gain according to the reference velocity (see the middle and bottom plots): for the sinusoidal segment the switching gain is approximately a periodic signal with the same period of the reference signal; whereas for the constant segment the switching gain remains constant. Note that despite that the system conditions are constant for the second segment, among the ASG methods only the discrete method M5 properly reaches a low constant value for the switching gain. Therefore, it can be concluded that the discrete ASG method M5 has the best performance, since it adapts promptly and effectively the switching gain and, hence, this method is used for the real experimentation in Section 7.

22

Since there are no ASG laws in the literature for the proposed one-side SMC, the proposed ASG law is compared below with a Fixed Switching Gain (FSG) strategy. The simulation was run under the same conditions indicated in Section 5.1 except for the following: $\omega_{ref}(t) = 0.5 + 0.5\sin(1.33\pi t - \pi/2)$, $K_{in,m} = 0.1$, $e_{\max} = 0.05$ and the switching gain parameters are as follows:

- FSG High: $u^+ = 200$

- FSG Low: $u^+ = 50$

- Proposed ASG: $u^+(k_0) = 50$ and $\mu_{in,m} = 3000$.

The comparison for this simulation is depicted in Fig. 3 and Fig. 4. In particular, the tracking error for the FSG case with a low gain is larger than the maximum allowed error $e_{\max}$ during some phases of the simulation, which means that the switching gain is not large enough. In contrast, for both the FSG case with a high gain and the ASG method the tracking error is kept within the allowed band, see the top plot in Fig. 3. However, since the ASG method uses a lower value for the switching gain, see the middle plot in Fig. 3, the chattering band obtained with this method is smaller than that obtained with the FSG case with a high gain, see the constraint functions $\phi_{in,m}$ in the detail view of Fig. 4(d). Therefore, the proposed ASG method for the one-side SMC has a good performance and adapts effectively the switching gain.

6. Controller implementation

The pseudo-code of the method is shown below, which uses the following auxiliary functions: kinematic function $\mathbf{l}(\mathbf{q})$; Jacobian matrices $\mathbf{J}$ and $\mathbf{J}_n$; Moore-Penrose pseudoinverse $(\cdot)^\dagger$, using a tolerance to set to zero the very small singular values [32]; $GetRobotStateAndForces()$, which returns the current robot state $\{\mathbf{q}, \dot{\mathbf{q}}\}$ and force vectors $\mathbf{F}_t$ and $\mathbf{F}_g$, which have already been filtered by the sensor electronics; and $SendToJointControllers(\ddot{\mathbf{q}}_c)$, which sends the current

23

commanded joint acceleration vector to the joint controllers. The computation time per iteration of the algorithm using compiled C code in a modern computer was around 15 microseconds for the experiment in Section 7.

Algorithm executed at sampling time of T_s seconds
1 $[\mathbf{q}, \dot{\mathbf{q}}, \mathbf{F}_t, \mathbf{F}_g]$ =GetRobotStateAndForces();
2 $\mathbf{p} = \mathbf{l}(\mathbf{q})$;   // Eq. (1)
3 $\dot{\mathbf{p}} = \mathbf{J}\dot{\mathbf{q}}$;   // Eq. (2)
4 $\dot{\mathbf{F}}_t = (\mathbf{F}_t - \mathbf{F}_{t,prev})/T_s$;   // Derivative
5 $\dot{\mathbf{F}}_g = (\mathbf{F}_g - \mathbf{F}_{g,prev})/T_s$;   // Derivative
6 $\boldsymbol{\phi}_1 = \begin{bmatrix} F_{t,z} - F_{z,ref} + K_{1,z}\dot{F}_{t,z} \\ F_{t,\alpha} + K_{1,\alpha}\dot{F}_{t,\alpha} \\ F_{t,\beta} + K_{1,\beta}\dot{F}_{t,\beta} \end{bmatrix}$;   // Eqs. (22),(31)-(33)
7 $\phi_2 = \sqrt{F_{g,x}^2 + F_{g,y}^2} - F_{l,0} + K_2\dfrac{\dot{F}_{g,x}F_{g,x} + \dot{F}_{g,y}F_{g,y}}{\sqrt{F_{g,x}^2 + F_{g,y}^2}}$;   // Eqs. (23),(37)
8 $\overline{W}_{1,i} = \left|\overline{W}_{1,i,prev} + T_s\,\mu_{1,i}\,\text{sign}(\phi_{1,i})\,\text{sign}(\phi_{1,prev,i})\right|$ for $i = z, \alpha, \beta$; // Eq. (25)
9 $\overline{W}_2 = \left|\overline{W}_{2,prev} + T_s\,\mu_2\,f_{ASG}(\phi_2, \phi_{2,prev})\right|$;   // Eq. (26)
10 $\mathbf{A}_1 = \mathbf{K}_1\mathbf{H}_1\mathbf{J}_n$;   // Eq. (35)
11 $\mathbf{b}_1 = -\overline{\mathbf{W}}_1\,\text{sign}\,(\boldsymbol{\phi}_1)\,u_1^+$;   // Eq. (35)
12 $\mathbf{A}_2 = \text{v2dm}\,(\text{pos}\,(\phi_2))\,K_2\mathbf{H}_2\mathbf{J}_n$;   // Eq. (40)
13 $\mathbf{b}_2 = -\overline{W}_2\,\text{pos}\,(\phi_2)\,u_2^+$;   // Eq. (40)
14 $\mathbf{A}_3 = \mathbf{J}$;   // Eq. (42)
15 $\mathbf{b}_3 = -K_{3,v}\dot{\mathbf{p}} - \text{sign}(\dot{\mathbf{p}})u_3^+$;   // Eq. (42)
16 $\mathbf{A}_4 = \mathbf{I}$;   // Eq. (43)
17 $\mathbf{b}_4 = -K_{4,v}\dot{\mathbf{q}} + K_{4,p}(\mathbf{q}_0 - \mathbf{q})$;   // Eq. (43)
18 $\ddot{\mathbf{q}}_{c,1} = \mathbf{A}_1^{\dagger}\mathbf{b}_1$;   // Eq. (6), $i = 1$
19 $\mathbf{N}_1 = \mathbf{I} - \mathbf{A}_1^{\dagger}\mathbf{A}_1$;   // Eq. (7), $i = 1$
20 $\ddot{\mathbf{q}}_{c,2} = \ddot{\mathbf{q}}_{c,1} + (\mathbf{A}_2\mathbf{N}_1)^{\dagger}(\mathbf{b}_2 - \mathbf{A}_2\ddot{\mathbf{q}}_{c,1})$;   // Eq. (6), $i = 2$
21 $\mathbf{N}_2 = \mathbf{N}_1(\mathbf{I} - (\mathbf{A}_2\mathbf{N}_1)^{\dagger}(\mathbf{A}_2\mathbf{N}_1))$;   // Eq. (7), $i = 2$
22 $\ddot{\mathbf{q}}_{c,3} = \ddot{\mathbf{q}}_{c,2} + (\mathbf{A}_3\mathbf{N}_2)^{\dagger}(\mathbf{b}_3 - \mathbf{A}_3\ddot{\mathbf{q}}_{c,2})$;   // Eq. (6), $i = 3$
23 $\mathbf{N}_3 = \mathbf{N}_2(\mathbf{I} - (\mathbf{A}_3\mathbf{N}_2)^{\dagger}(\mathbf{A}_3\mathbf{N}_2))$;   // Eq. (7), $i = 3$
24 $\ddot{\mathbf{q}}_{c,4} = \ddot{\mathbf{q}}_{c,3} + (\mathbf{A}_4\mathbf{N}_3)^{\dagger}(\mathbf{b}_4 - \mathbf{A}_4\ddot{\mathbf{q}}_{c,3})$;   // Eq. (6), $i = 4$
25 SendToJointControllers($\ddot{\mathbf{q}}_{c,4}$);
26 $\mathbf{F}_{t,prev} = \mathbf{F}_t$;   // For next iteration
27 $\mathbf{F}_{g,prev} = \mathbf{F}_g$;   // For next iteration
28 $\boldsymbol{\phi}_{1,prev} = \boldsymbol{\phi}_1$;   // For next iteration
29 $\phi_{2,prev} = \phi_2$;   // For next iteration
30 $\overline{\mathbf{W}}_{1,prev} = \overline{\mathbf{W}}_1$;   // For next iteration
31 $\overline{W}_{2,prev} = \overline{W}_2$;   // For next iteration

7. Real experimentation

7.1. Setup

The setup used for the experiment consists of (see Fig. 5): a Sawyer robot; a plastic T-shaped guiding handle composed of two prismatic portions of dimensions 148x30x6mm and 30x36x6mm; a force/torque sensor ATI Nano25 (guide sensor) located between the last link of the robot and the guiding handle; another force/torque sensor ATI Nano25 (treatment sensor) located between the guiding handle and the tool; a profiling tool consisting of a cylinder of 43x43x10mm; and a flat rectangular plastic object of 190x95x3mm as target.

The proposed controller has been implemented in an external PC workstation (Intel Core i5-3470 processor at 3.2GHz) using *Ubuntu 16.04* as Operative System, *ROS* (Robot Operating System) *Lunar* distribution as middleware to interface with the Sawyer robot, *Intera 5 SDK* software provided by the robot manufacturer (Rethink Robotics) and the *netft_rdt_driver* ROS package provided by the sensor manufacturer (ATI Industrial Automation). The Sawyer robot, force sensors and external PC workstation are connected to an Ethernet switch device and communicate via UDP protocol.

The SDK software allows the development of custom applications to run outside the robot controller [33]. In particular, the algorithm presented in Section 6 has been implemented in the external PC workstation using the `intera_interface` module provided by the SDK, which interfaces with the robot controller and each of the Nano25 box controllers via ROS.

7.2. Experiment conditions

i) The period T_s is set to 0.02s and the commanded accelerations $\ddot{\mathbf{q}}_c$ are integrated to obtain the commanded velocities $\dot{\mathbf{q}}_c$ sent to the robot controller.

ii) The signals of both force sensors are filtered using a low-pass filter with a cut-off frequency of 73Hz, which is implemented in the sensor electronics.

iii) Parameters used for Level 1 (Section 4.4): $F_{z,ref} = -10$N, $u_1^+ = 0.06$, $\overline{W}_{1,z} = 0.8$, $\overline{W}_{1,\alpha} = \overline{W}_{1,\beta} = 12$, $\mu_{1,i} = 0.02$ and $K_{1,i} = 0.15$.

iv) Parameters used for Level 2 (Section 4.5): $F_{l,0} = 3$N, $u_2^+ = 0.24$ (switching gain weight omitted for this scalar case), $\mu_2 = 0.04$ and $K_2 = 0.075$.

v) Parameters used for Level 3 (Section 4.6): $K_{3,v} = 0.01$ and $u_3^+ = 1$.

vi) Parameters used for Level 4 (Section 4.7): $K_{4,v} = 4.5$, $K_{4,p} = 0.75$ and $\mathbf{q}_0 = \begin{bmatrix} 1.04° & -36.41° & -1.39° & 95.24° & 2.08° & 30.19° & 188.29° \end{bmatrix}^{\mathrm{T}}$.

7.3. Results

In order to verify the performance of the proposed approach an experiment has been conducted, whose video can be played at https://media.upv.es/player/?id=c9e06e90-0f71-11e8-82c3-510d9467b019. Fig. 6 shows several frames from the video: Fig. 6(a) and 6(b) (interval 17s–23s in the video) show how the user guides the robot tool when the flat target object is resting on the table; Fig. 6(c) and 6(d) (interval 35s–37s) show how the robot tool smoothly stops when the user releases the robot guiding handle[2]; and Fig. 6(e), and 6(f) (interval 2m21s–2m25s) show how the user guides the robot tool when the flat object changes its position, orientation and stiffness, i.e., when the flat object is held above the table.

Fig. 7 shows the performance of the current approach in terms of constraint functions, switching gains and activation of the inequality constraint used to guide the robot tool. In particular, it can be seen that the equality constraints $\{\sigma_{1,z}, \sigma_{1,\alpha}, \sigma_{1,\beta}\}$ are switching around zero as expected. This means that the surface treatment is being done properly: the tool orientation is perpendicular to the object surface and the pressure with the tool on the surface is being kept around the desired value regardless the changes of the object position, orientation and stiffness. The figure also shows the variation of the computed switching gain $u_{1,z}^+$ to keep the pressure on the object surface regardless the changes in its stiffness. In particular, for the intervals 85s–136s and 155s–168s in the graph (1m40s–2m31s and 2m50s–3m03 in the video) the flat object is held

---

[2]A smooth stop has been considered for the robot tool. However, the desired speed reduction to stop the robot tool can be selected in Level 3.

above the table and, hence, its stiffness coefficient in the Z-axis is significantly reduced because it has no support. Thus, for this interval the ASG method increases the switching gain $u_{1,z}^+$ to properly keep the desired pressure, see the second plot in Fig. 7.

Fig. 7 also shows in the sixth plot the variation of the computed switching gain in Level 2, where it can be seen that it is increased by the ASG algorithm when large force magnitudes are detected, see the peaks in the fifth plot. Fig. 8 shows the trajectory followed by the robot tool, where it can be seen that a large area has been covered by the tool during the experiment: around 0.4 meters in both X- and Y-axes and 0.2 meters in Z-axis.

Note that the method is very sensitive to the human operator's forces: the robot tool is guided by the operator using very small forces. However, if a low threshold $F_{l,0}$ is considered, the constraint at Level 2 could be accidentally triggered given the effective weight supported by the guide sensor, yielding an undesired tool motion. Indeed, an accidental activation occurs in the interval 2m09s–2m12s of the video due to the large tilt angle of the flat object, causing the weight component acting on the surface plane to become larger than the mentioned threshold. This undesired movement is simply avoided when the user handles the guiding element again, as is expected to be the situation generally (see interval 2m12s–2m15s in the video). In practice, to avoid this weight-related force perturbation, it is convenient to minimize the weight supported by the guide sensor.

To further document this phenomenon, another experiment has been conducted using a different force sensors configuration readily adequate to avoid the accidental activation of the Level 2 constraint. Fig. 9(a) shows the new sensor configuration in which the weight supported by the guide sensor (sensor at the top) is significantly reduced, only supporting the weight of the plastic T-shaped guiding handle itself (shown in white). The video of the experiment can be played at https://media.upv.es/player/?id=c0d926f0-3d68-11e8-a61d-fb59fde74596, where it can be seen that in contrast to the previous case, when the flat

object is held above the table with a large tilt angle as before, and the user releases the robot guiding tool, the manipulator remains motionless, (interval 1m21s–1m30s, and Fig. 9(b) and Fig. 9(c)).

8. Conclusions

This work has proposed a human-robot closely collaborative solution to cooperatively perform surface treatment tasks such as polishing, grinding, deburring, etc. For this purpose, two force sensors attached to the manipulator end-effector and tool have been considered: one sensor is used to properly accomplish the surface treatment task, i.e., to attain the desired pressure between the tool and the surface being treated, as well as to keep the tool orientation perpendicular to the surface; whereas the second sensor is used by the human operator to guide the robot tool along the surface to be treated.

The proposed control scheme is based on task priority and adaptive sliding mode control. A distinctive feature of the method is that the controller was developed using not only conventional sliding mode control but also a novel one-side sliding mode control. In addition, an adaptive switching gain law was considered not only for the conventional sliding mode control but also for the proposed one-side sliding mode control.

The main advantages of the method are robustness and low computational cost. The feasibility and effectiveness of the proposed collaborative solution for robotic surface treatment have been substantiated by experimental results using a redundant 7R manipulator.

References

[1] F. Tian, Z. Li, C. Lv, G. Liu, Polishing pressure investigations of robot automatic polishing on curved surfaces, The International Journal of Advanced Manufacturing Technology 87 (2016) 639–646.

[2] C. Edwards, S. Spurgeon, Sliding Mode Control: Theory and Applications, 1st ed., Taylor & Francis, UK, 1998.

[3] Y. Shi, D. Zheng, L. Hu, Y. Wang, L. Wang, Nc polishing of aspheric surfaces under control of constant pressure using a magnetorheological torque servo, The International Journal of Advanced Manufacturing Technology 58 (2012) 1061–1073.

[4] X. Xie, L. Sun, Force control based robotic grinding system and application, in: 2016 12th World Congress on Intelligent Control and Automation (WCICA), 2016, pp. 2552–2555.

[5] L. Roveda, F. Vicentini, N. Pedrocchi, L. M. Tosatti, Impedance control based force-tracking algorithm for interaction robotics tasks: An analytically force overshoots-free approach, in: 2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO), volume 02, 2015, pp. 386–391.

[6] Y. Oba, Y. Kakinuma, Simultaneous tool posture and polishing force control of unknown curved surface using serial-parallel mechanism polishing machine, Precision Engineering 49 (2017) 24 – 32.

[7] F. Dimeas, N. Aspragathos, Online stability in human-robot cooperation with admittance control, IEEE Transactions on Haptics 9 (2016) 267–278.

[8] A. M. Khan, D.-w. Yun, K. M. Zuhaib, J. Iqbal, R.-J. Yan, F. Khan, C. Han, Estimation of desired motion intention and compliance control for upper limb assist exoskeleton, International Journal of Control, Automation and Systems 15 (2017) 802–814.

[9] Y. Li, S. S. Ge, Force tracking control for motion synchronization in human-robot collaboration, Robotica 34 (2016) 1260–1281.

[10] J. Vogel, S. Haddadin, B. Jarosiewicz, J. Simeral, D. Bacher, L. Hochberg, J. Donoghue, P. van der Smagt, An assistive decision-and-control architecture for force-sensitive hand-arm systems driven by human-machine interfaces, The International Journal of Robotics Research 34 (2015) 763–780.

29

[11] S. Jlassi, S. Tliba, Y. Chitour, An event-controlled online trajectory generator based on the human-robot interaction force processing, Industrial Robot: An International Journal 41 (2014) 15–25.

[12] S. Sakaino, K. Ohnishi, Sliding mode control based on position control for contact motion applied to hopping robot, in: 2006 IEEE International Conference on Industrial Technology, 2006, pp. 170–175.

[13] E. Bassi, F. Benzi, L. M. Capisani, D. Cuppone, A. Ferrara, Hybrid position/force sliding mode control of a class of robotic manipulators, in: Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference, 2009, pp. 2966–2971.

[14] S.-J. Huang, Y.-C. Liu, S.-H. Hsiang, Robotic end-effector impedance control without expensive torque/force sensor, International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering 7 (2013) 1446 – 1453.

[15] E. Engeberg, S. Meek, M. Minor, Hybrid force-velocity sliding mode control of a prosthetic hand, IEEE Transactions on Biomedical Engineering 55 (2008) 1572–1581.

[16] J. Zhou, Z. Zhou, Q. Ai, Impedance control of the rehabilitation robot based on sliding mode control, in: X. Li (Ed.), Mechanical Engineering and Control Systems (MECS2015), 2016, pp. 135–140.

[17] N. Kashiri, N. G. Tsagarakis, M. Van Damme, B. Vanderborght, D. G. Caldwell, Proxy-Based Sliding Mode Control of Compliant Joint Manipulators, Springer International Publishing, Cham, 2016, pp. 241–257.

[18] D. Yun, A. M. Khan, R.-J. Yan, Y. Ji, H. Jang, J. Iqbal, K. M. Zuhaib, J. Y. Ahn, J. Han, C. Han, Handling subject arm uncertainties for upper limb rehabilitation robot using robust sliding mode control, International Journal of Precision Engineering and Manufacturing 17 (2016) 355–362.

30

[19] Q. Wu, X. Wang, F. Du, Q. Zhu, Fuzzy sliding mode control of an upper limb exoskeleton for robot-assisted rehabilitation, in: 2015 IEEE International Symposium on Medical Measurements and Applications (MeMeA) Proceedings, 2015, pp. 451–456.

[20] A. Mitra, L. Behera, Development of a fuzzy sliding mode controller with adaptive tuning technique for a MRI guided robot in the human vasculature, in: 2015 IEEE 13th International Conference on Industrial Informatics (INDIN), 2015, pp. 370–377.

[21] N. Rahman, M. C. Lee, Reaction force separation method of surgical tool from unknown dynamics and disturbances by fuzzy logic and perturbation observer of SMCSPO algorithm, in: The SICE Annual Conference 2013, 2013, pp. 2536–2541.

[22] C. Gaz, E. Magrini, A. De Luca, A model-based residual approach for human-robot collaboration during manual polishing operations, Mechatronics , In Press (2018) DOI: 10.1016/j.mechatronics.2018.02.014.

[23] Y. Nakamura, H. Hanafusa, T. Yoshikawa, Task-priority based redundancy control of robot manipulators, Int. J. of Robotics Research 6 (1987) 3–15.

[24] B. Siciliano, J. Slotine, A general framework for managing multiple tasks in highly redundant robotic systems, in: Fifth Int. Conference on Advanced Robotics (ICAR'91), Pisa, Italy, 1991, pp. 1211–1216, 1991.

[25] V. Utkin, J. Guldner, J. Shi, Sliding Mode Control in Electro-Mechanical Systems, 2nd ed., Taylor & Francis, London, 2009.

[26] G. Monsees, J. Scherpen, Adaptive switching gain for a discrete-time sliding mode controller, International Journal of Control 75 (2002) 242–251.

[27] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics: Modelling, Planning and Control, Springer-Verlag, London, UK, 2009.

[28] T.-P. Leung, Q.-J. Zhou, C.-Y. Su, An adaptive variable structure model following control design for robot manipulators, IEEE Transactions on Automatic Control 36 (1991) 347–353.

[29] G. Wheeler, C. Su, Y. Stepanenko, A sliding mode controller with improved adaption laws for the upper bounds on the norm of uncertainties, Automatica 34 (1998) 1657–1661.

[30] J. Zhu, K. Khayati, On a new adaptive sliding mode control for mimo nonlinear systems with uncertainties of unknown bounds, International Journal of Robust and Nonlinear Control 27 (2017) 942–962.

[31] F. Plestan, Y. Shtessel, V. Bregeault, A. Poznyak, New methodologies for adaptive sliding mode control, Int. Journal of Control 83 (2010) 1907–1919.

[32] G. Golub, C. Van Loan, Matrix Computations, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.

[33] SDK website, http://sdk.rethinkrobotics.com/intera/SDK_System_Overview (Accessed 2018/04/12) Rethink Robotics.
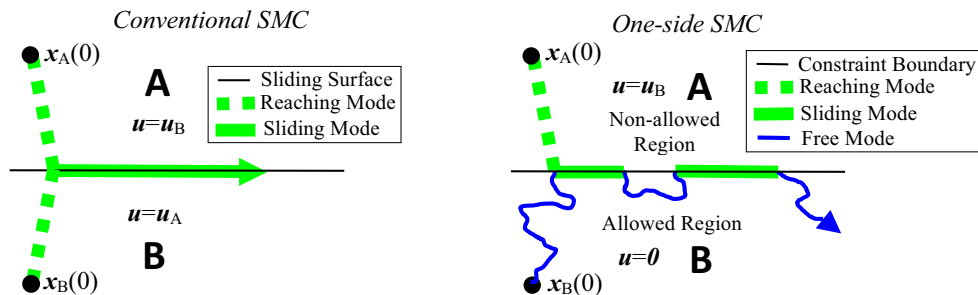
Fig. 1. Graphical comparison between conventional SMC (left) and one-side SMC (right).
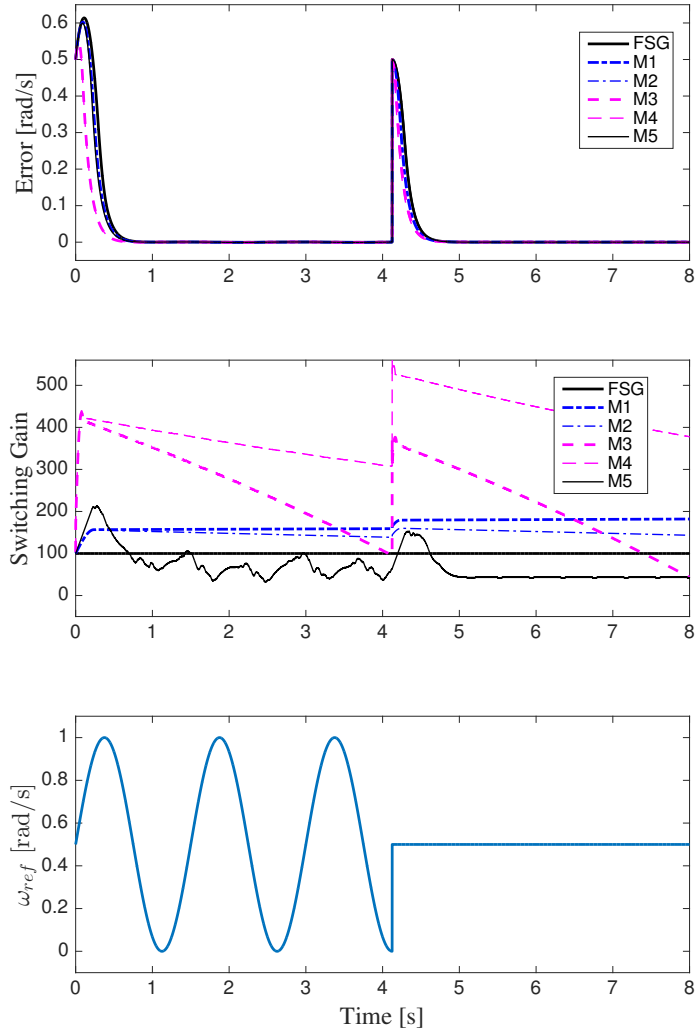
Fig. 2. Simulation with constant motor parameters and sinusoidal and constant reference velocity: comparison of the FSG and ASG methods.
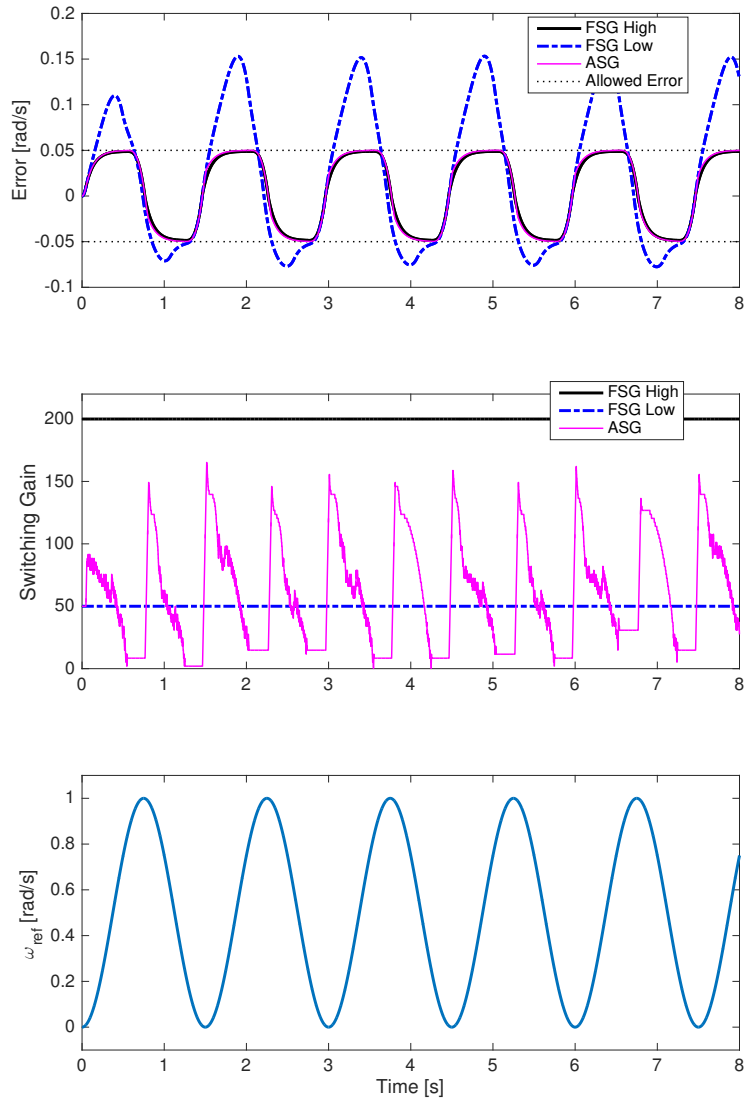
Fig. 3. Simulation of a DC motor control using one-side SMC with a sinusoidal reference velocity: comparison of FSG and ASG methods in terms of error and switching gain.

(a) FSG High

(b) FSG Low

(c) ASG

(d) Chattering band comparison
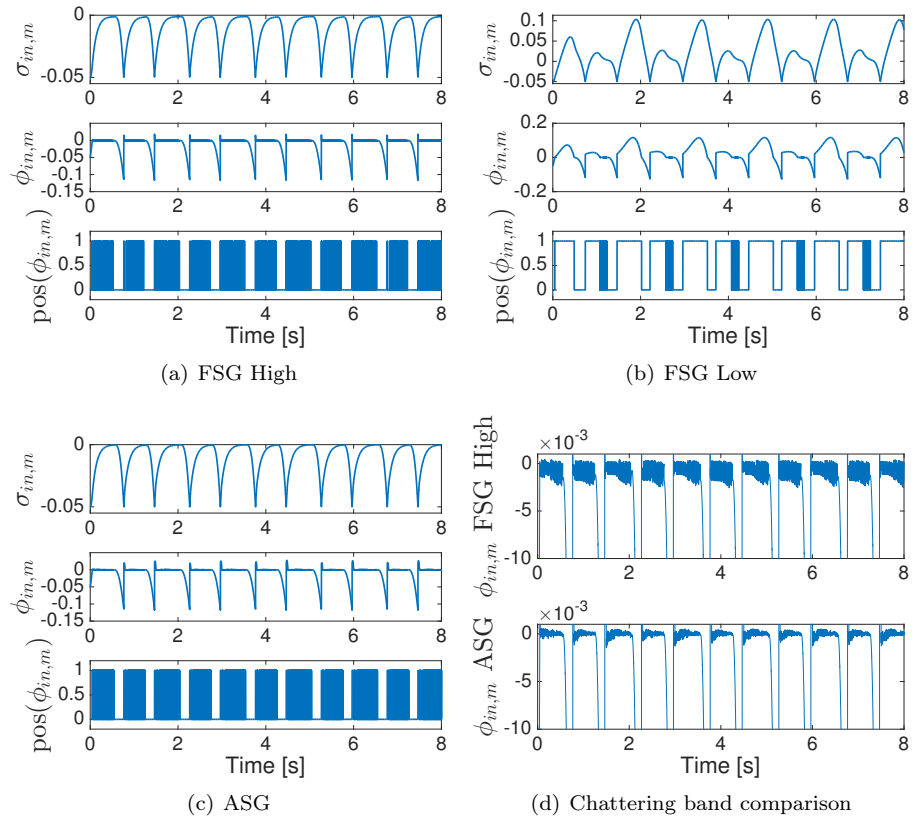
Fig. 4. Simulation of a DC motor control using one-side SMC: Detail of constraint and commutation functions for each method.
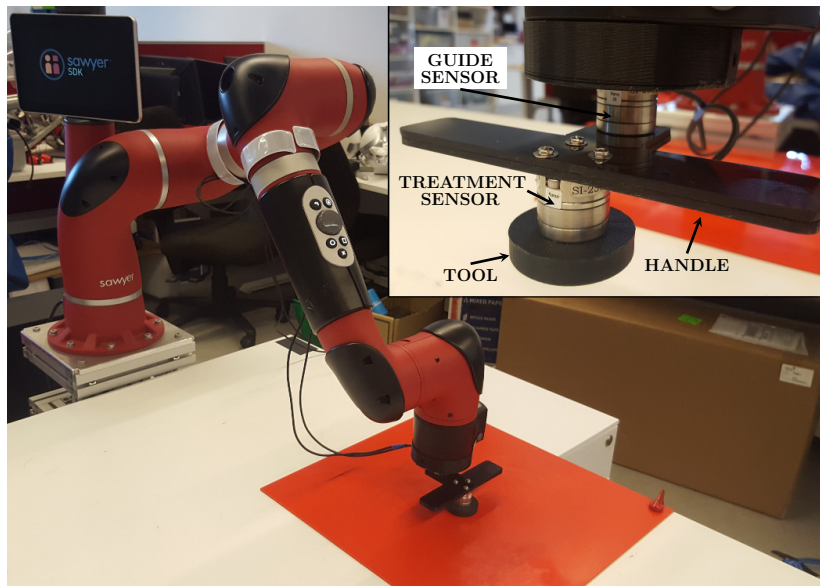
Fig. 5. Experimental setup: 7R serial manipulator with two force sensors (guide and treatment sensors), a plastic T-shaped guiding handle (black), a tool consisting of a cylinder (black) and a flat rectangular plastic object as target (red).
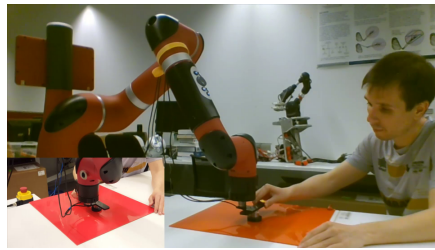
(a) video: 0m17s; graph: 2s

(b) video: 0m23s; graph: 8s
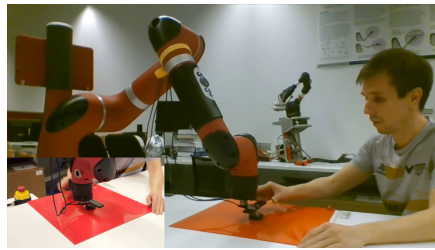
(c) video: 0m35s; graph: 20s
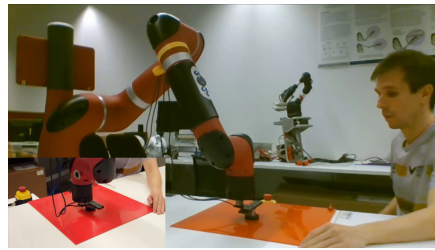
(d) video: 0m37s; graph: 22s
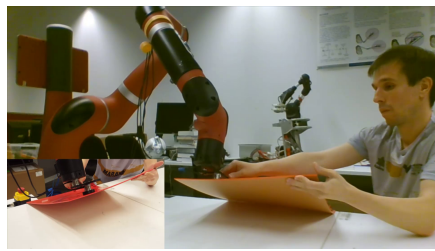
(e) video: 2m21s; graph: 126s
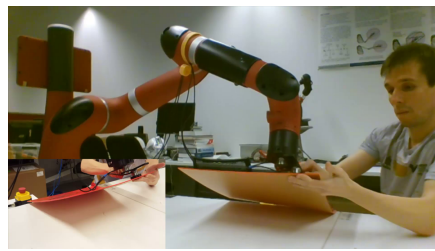
(f) video: 2m25s; graph: 130s

Fig. 6. Frames of the video of the experiment. The time instant is indicated for each frame.
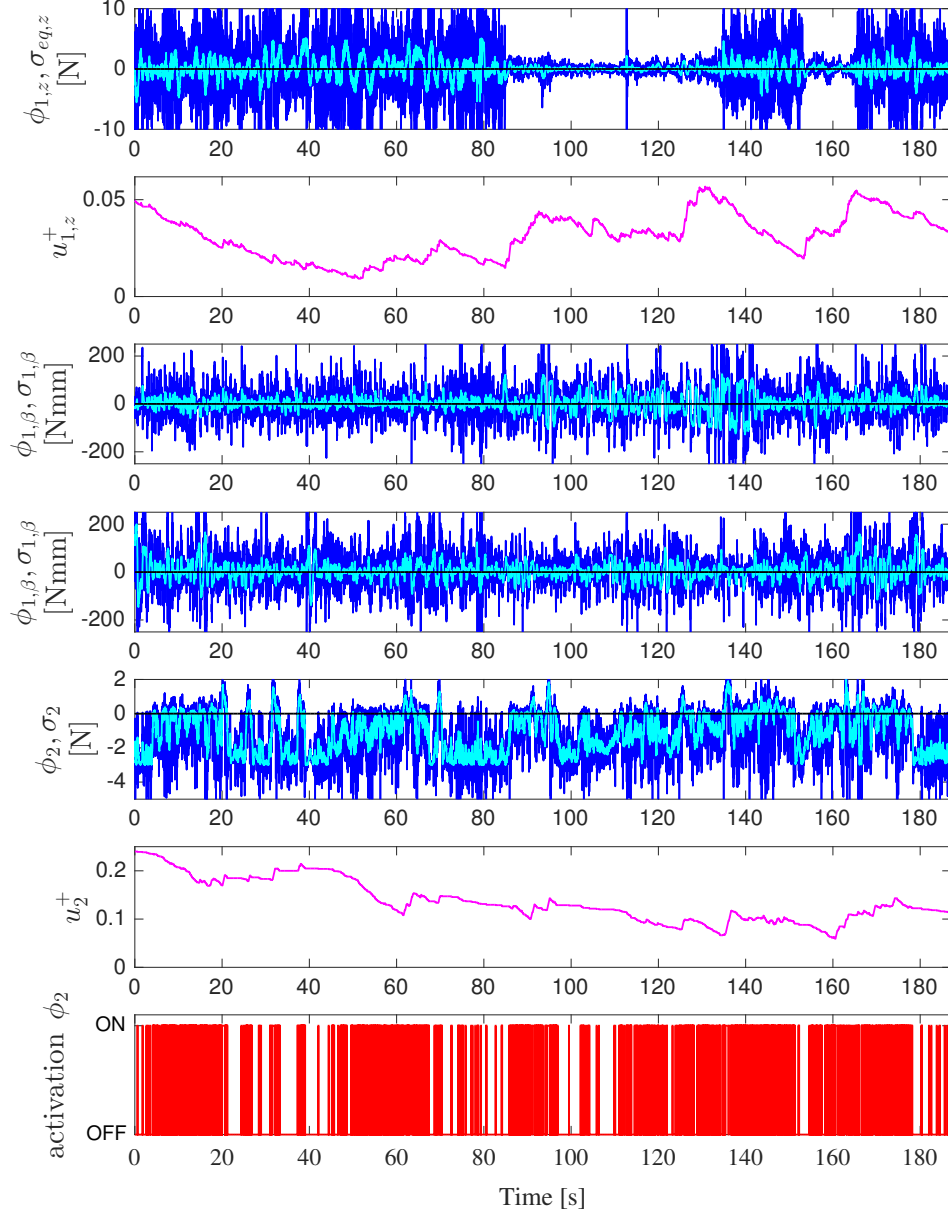
Fig. 7. Behavior of the constraints in the experiment. First, third, fourth and fifth plots: constraint functions for Level 1 and Level 2, the modified constraint function ϕ_i is in darkblue, whereas the original constraint function σ_i is in light-cyan. Second and sixth plots: switching gains for Level 1 ($u_{1,z}^+ = \overline{W}_{1,z} u_1^+$) and Level 2 ($u_2^+ = \overline{W}_2 u_2^+$), the switching gains $u_{1,\alpha}^+$ and $u_{1,\beta}^+$ are omitted because they do not present significant variations. Bottom plot: activation of the inequality constraint in Level 2.
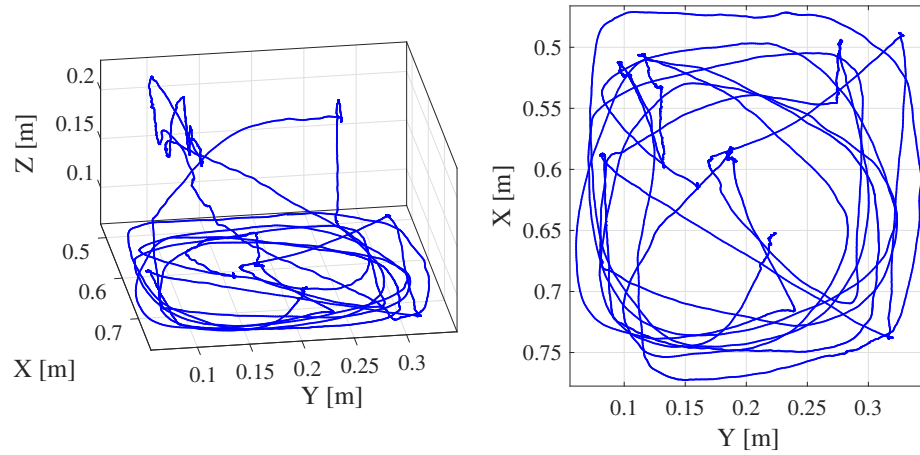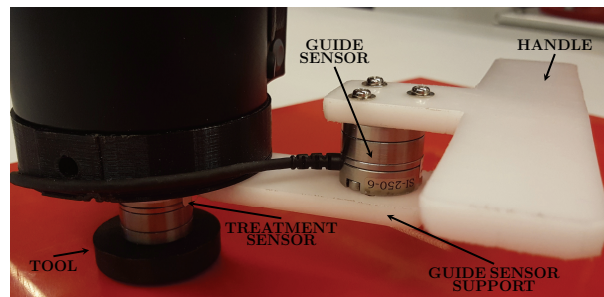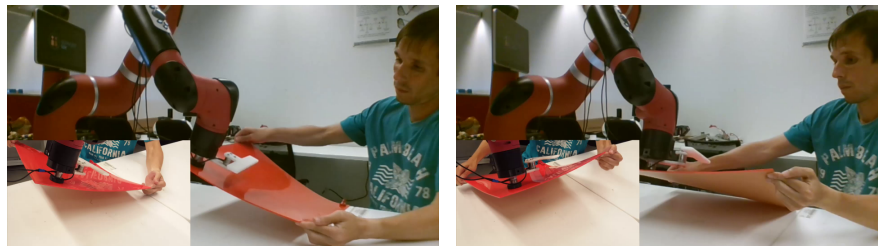
Fig. 8. 3D representation of the trajectory followed by the robot tool in the experiment.

(a) Detail view of the new configuration for the force sensors: the guide sensor only supports the weight of the plastic T-shaped guiding handle.

(b) Frame of the video at instant 1m22s

(c) Frame of the video at instant 1m25s

Fig. 9. Experiment with a different force sensors configuraton.