Reactive Sliding-Mode Algorithm for Collision Avoidance in Robotic Systems

Luis Gracia, Fabricio Garelli, and Antonio Sala, *Member, IEEE*

***Abstract*— This brief presents a reactive reference conditioning algorithm for robot collision avoidance based on geometric invariance and sliding-mode (SM) ideas. First, constraints are defined in terms of the measurements given by the robot's sensors in order to guarantee that collisions will not occur. Then, a supervisory loop ensures the fulfillment of the constraints modifying the reference trajectory as much as necessary by means of a discontinuous control law. The proposed algorithm activates only when the constraints are about to be violated and, thus, in contrast to conventional SM approaches, there exists no reaching mode to the limit surface of the constraints (sliding surface). The validity and effectiveness of the proposed approach is substantiated by simulation and experimental results using a mobile robot equipped with infrared sensors.**

***Index Terms*— Collision avoidance, mobile robots, motion planning, reference conditioning, sliding-mode (SM) control.**

I. Introduction

During robotic operations, there are some workspace constraints in the robot end-effector position and orientaion and joint coordinates, as well as other constraints in position of intermediate points, etc., which must be enforced in order to ensure a safe and reliable operation. Also, a relevant issue when tracking a reference trajectory in partially structured environments is the avoidance of collisions with unforeseen obstacles.

For this purpose, the reference trajectory previously generated by either an operator or a high-level planner [1] must be locally modified by a low-level reactive system [2], [3] according to the obstacles detected by the robot's sensors. In this sense, this brief presents a reference conditioning algorithm for collision avoidance using the information gathered by proximity sensors.

This brief complements previous works in which sliding-mode (SM) ideas were used to design reference conditioning algorithms in structured environments [4] with modifications for redundant robots [5], [6] in order to cope with end-effector position constraints. Here, these original ideas are extended and further exploited to deal with unstructured environments and more general constraints on the full robot configuration (end-effector position and orientation), as well as an experimental setup to verify its reactive behavior is provided.

The proposed algorithm is based on geometric invariance and SM control theory [7]–[9]. The basic idea is to define constraints [10] in terms of the measurements given by the proximity sensors in order to guarantee that collisions will not occur. In particular, a supervisory loop [11] is designed to properly modify the reference trajectory in order to fulfill the constraints while tracking the reference.

The algorithm is based on a variable structure control law that only becomes active only when the constraints are about to be violated. Thus, in contrast to conventional SM control applications [12]–[15], there exists no reaching mode to the sliding surface (i.e., the limit surface of the constraints) and it is the system itself (reference trajectory, robot, detected obstacles, etc.) which gives rise to the sliding mode by means of the activation of the constraints. Therefore, sliding regimes are exploited in this brief as a transitional mode of operation, in which the discontinuous signal is used for conditioning the reference trajectory in order to satisfy the constraints. The proposed approach can be interpreted as a limit case of the conventional potential-field-based approach for collision avoidance [16], [17], which requires using variable structure control concepts like such as those used in this brief.

The outline of this brief is as follows. Section II introduces notation, states the main problem to be addressed, and presents some general concepts on geometric invariance and sliding regimes. Section III develops the SM reference conditioning technique proposed in this brief for robot collision avoidance. Both simulation and experimental results are presented in Section IV using a mobile robot equipped with infrared sensors in order to show the viability and effectiveness of the proposed approach. Section V concludes this brief.

II. Preliminaries

A. Notation

Following the standard notation [18], consider a robot system with $\mathbf{q} = [q_1 \ldots q_n]^{\mathrm{T}}$ being the robot configuration or joint position vector and $\mathbf{p}$ being the robot pose or workspace position vector, e.g., the Cartesian position and Euler-angle orientation of the robot end-effector. A robot is said to be redundant when the dimension m of the workspace is less than the dimension n of the configuration space (hereafter, C-space). For the rest of this brief, it is assumed that the robot at hand is nonredundant,[1] i.e., $m = n$. The relationship

Manuscript received July 30, 2011; revised August 25, 2012; accepted November 26, 2012. Manuscript received in final form November 30, 2012. Date of publication January 17, 2013; date of current version October 15, 2013. This work was supported in part by research under Project DPI2011-27845-C02-01 (Spanish Government), Projects PAID-05-11-2640, and PAID-00-12-SP20120159, (Universitat Politècnica de València, Spain), and Projects ANPCyT PICT-2011-0888, CONICET PIP 112-2011-00361, and UNLP I164 in Argentina. Recommended by Associate Editor M. Zefran.

L. Gracia and A. Sala are with the Department of Systems Engineering and Control, Universitat Politècnica de València, Valencia 46022, Spain (e-mail: luigraca@isa.upv.es; asala@isa.upv.es).

F. Garelli is with CONICET and Universidad Nacional de La Plata, La Plata 1900, Argentina (e-mail: fabricio@ing.unlp.edu.ar).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCST.2012.2231866

[1] The ideas in [5] for redundant robots may also be applied to the case under discussion here, but the details are omitted for brevity.

1063-6536 © 2013 IEEE

between the robot configuration and the robot pose is generically expressed as

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

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

The first- and second-order kinematics result in

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

where $\mathbf{J}_p(\mathbf{q})$ is the Jacobian matrix of the kinematic function $\mathbf{l}(\mathbf{q})$.

Let us denote as $\mathbf{p}_{\text{ref}}(t)$ the workspace reference, generated by an operator or high-level planner.

B. Problem Statement

We consider now that the robotic system to be controlled is subjected to C-space constraints given by

$$\Phi_{CS}(\mathbf{q}) = \{\mathbf{q} \mid \sigma_i(\mathbf{q}) \le 0\}, \qquad i = 1, \ldots, N \tag{3}$$

where N is the number of constraints, Φ_{CS} is the allowed region of the C-space, and σ_i is a function of the robot configuration[2] $\mathbf{q}$, that which is positive only if the ith-constraint is not fulfilled. In order for some smoothness assumptions to hold in the solution later proposed in this brief, the functions σ_i need to be twice differentiable around the boundary given by $\sigma_i(\mathbf{q}) = 0$ and their gradients $\nabla\sigma_i \equiv \partial\sigma_i/\partial\mathbf{q}$ around this boundary should not vanish.

The main control goal, therefore, is to generate a modified position reference $\mathbf{p}^*_{\text{ref}}$ to be sent to the robot controller so that it is as close as possible to the user-input value $\mathbf{p}_{\text{ref}}$ and that the C-space constraints given by (3) are fulfilled.

C. Geometric Invariance via SM Conditioning

Consider the following dynamical system with n_x states and n_u inputs

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{d}) + \mathbf{g}(\mathbf{x})\,\mathbf{u} \tag{4}$$

where $\mathbf{x} \in X \subset \mathbb{R}^{n_x}$ is the state vector, $\mathbf{d} \in D \subset \mathbb{R}^{n_d}$ is an unmeasured disturbance or model uncertainty, $\mathbf{u} \in U \subset \mathbb{R}^{n_u}$ is the control input vector (possibly discontinuous), $\mathbf{f}\colon \mathbb{R}^{n_x+n_d} \to \mathbb{R}^{n_x}$ is a vector field defined in $X \bigcup D$, and $\mathbf{g}\colon \mathbb{R}^{n_x} \to \mathbb{R}^{n_x \times n_u}$ is a set of n_u vector fields defined in X.

The system state vector $\mathbf{x}$ has to be bounded so as to fulfill user-specified constraints $\phi_i(\mathbf{x}) \le 0$, $i = 1, \ldots, N$, where $\phi_i(\mathbf{x})$ is the ith constraint function. Thus, the region $\Phi(\mathbf{x})$ of the state space compatible with the bounds on state $\mathbf{x}$ is given by the set

$$\Phi(\mathbf{x}) = \{\mathbf{x} \mid \phi_i(\mathbf{x}) \le 0\}, \quad i = 1, \ldots, N \tag{5}$$

and the goal is then to find a control input $\mathbf{u}$ such that the region Φ becomes invariant (i.e., trajectories originating in Φ remain in Φ for all times t), while $\mathbf{x}$ is driven as close as possible to a desired trajectory $\mathbf{x}_{\text{ref}}$. To ensure the invariance of Φ, the control input $\mathbf{u}$ must guarantee that the right-hand side of the first equation in (4) points to the interior of Φ at all points in the boundary of Φ, denoted by $\partial\Phi$ and, defined as

$$\partial\Phi = \bigcup_{i=1}^{N} \partial\Phi_i, \quad \partial\Phi_i = \{\mathbf{x} \in \Phi \mid \phi_i(\mathbf{x}) = 0\}. \tag{6}$$

Assumption 1: ϕ_i is differentiable in the boundary $\partial\Phi_i$.

The previous assumption will allow the computing computation of the gradient vectors $\nabla\phi_i$.

Mathematically, the invariance of Φ is ensured by an input $\mathbf{u}$ such that

$$\begin{aligned}\frac{d(\phi_i(\mathbf{x}))}{dt} &= \nabla\phi_i^{\mathrm{T}}(\mathbf{x})\dot{\mathbf{x}} \\ &= \nabla\phi_i^{\mathrm{T}}(\mathbf{x})\mathbf{f}(\mathbf{x}, \mathbf{d}) + \nabla\phi_i^{\mathrm{T}}(\mathbf{x})\mathbf{g}(\mathbf{x})\,\mathbf{u} \\ &= L_f\phi_i(\mathbf{x}, \mathbf{d}) + \mathbf{L_g}\phi_i(\mathbf{x})\mathbf{u} \le 0 \quad \forall i \,|\, \phi_i(\mathbf{x}) = 0\end{aligned} \tag{7}$$

where the scalar $L_f\phi_i$ and the n_u-dimensional row vector $\mathbf{L_g}\phi_i$ 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. In general, any vector $\mathbf{u}$ pointing toward the interior of the allowed region can be used to satisfy (7). This vector could be computed, for instance, by solving a linear-programming optimization problem. However, the goal of this brief is to present a more simple strategy, involving only linear algebra. For this purpose, the proposition below is considered.

Proposition 1: We can make the set Φ invariant by means of the following variable structure control law

$$\mathbf{u} = \begin{cases} \mathbf{0}_{n_u}, & \text{if } \max\limits_i \{\phi_i(\mathbf{x})\} < 0 \\ \mathbf{u}_{\text{SM}}, & \text{otherwise} \end{cases} \tag{8}$$

where $\mathbf{0}_{n_u}$ denotes the n_u-dimensional null column vector and $\mathbf{u}_{\text{SM}}$ is chosen as follows:

$$\mathbf{u}_{\text{SM}} = -\mathbf{L_g}\boldsymbol{\phi}^{\mathrm{T}}\,\mathbf{1}_b\,u^+ \tag{9}$$

where b is the number of active constraints (i.e., those constraints with $\phi_i \ge 0$), $\mathbf{1}_b$ is the b-dimensional column vector with all its components equal to 1, matrix $\mathbf{L_g}\boldsymbol{\phi}$ contains the row vectors $\mathbf{L_g}\phi_i$ of all active constraints, and u^+ is a positive constant to be chosen high enough to satisfy (7). In particular, a sufficient, but not necessary, condition for making the set Φ invariant is

$$u^+ > \sum_{i=1}^{b}\left(\max(L_f\phi_i, 0)\right) \Big/ \operatorname{eig}_{\min}(\mathbf{L_g}\boldsymbol{\phi}\,\mathbf{L_g}\boldsymbol{\phi}^{\mathrm{T}}). \tag{10}$$

Proof: See Appendix. ■

Observe that, as long as the state trajectory tries by itself to leave the allowed region Φ, the above control law (8) will make $\mathbf{u}$ switch between $\mathbf{0}_{n_u}$ and $\mathbf{u}_{\text{SM}}$ at a theoretically infinite frequency, which can be seen as an ideal SM operation without the absence of open-loop phase (reaching mode) [7]. Thus, the switching law (8) does not seek for SM, but becomes transiently active only when the process is at the boundary of the allowed region Φ and about to leave it.

[2] In many practical applications, some restrictions are made on the robot pose $\mathbf{p}$. As they are functions of the robot configuration $\mathbf{q}$, such constraints are included in the general setting (3). For later sections in which gradients are needed, note that $\frac{\partial\sigma_i}{\partial\mathbf{q}} = \mathbf{J}_p^{\mathrm{T}}\frac{\partial\sigma_i}{\partial\mathbf{p}}$.

Once the SM is established on the boundary $\partial\Phi$ by the control action $\mathbf{u}$, a continuous equivalent control [7] can be obtained, which is the control required to keep the system just on the boundary manifold $\partial\Phi$. Consequently, the SM conditioning generated by (8) produces such control action without explicit knowledge of it and with a low computational cost; this is a distinctive advantage of SM conditioning strategies [10].

From the above discussions, the proposal could be seen as a one-side sliding control: if we are at $\phi > 0$, then we make $\phi = 0$ attractive in finite time in the same way as SM control. Otherwise, if $\phi < 0$, we disregard it. The sliding regime and the equivalent dynamics only occur if the uncontrolled system is pushing outwards of the allowed region, while the conditioning algorithm here pushes inwards to fulfill the constraints; therefore, sliding regimes are exploited in this brief as a transitional mode of operation, in contrast to the conventional sliding control where $\phi = 0$ is made attractive and invariant.

Interested readers may refer to [7]–[9] for further details on conventional SM control theory, and to [10] and [11] for constrained control applications of SM reference conditioning.

III. Proposal

A. Reference Conditioning Method

We are interested in exploiting the exact approximation to the constraint boundary that allows variable structure laws such as (8) to deal with the control problem stated in Section II-B. In particular, the ideas of Section II-C are employed to perform an on-line robotic reference conditioning so that the environment limits given by (3) are fulfilled.

To avoid reaching the boundary of the constrains at high speed, the actual constraint space (3) will be modified to include the speed of movement as follows:

$$\Phi^*_{CS}(\mathbf{q},\dot{\mathbf{q}}) = \left\{[\mathbf{q}^{\mathrm{T}}\ \dot{\mathbf{q}}^{\mathrm{T}}]^{\mathrm{T}} \mid \phi_i(\mathbf{q},\dot{\mathbf{q}}) = \sigma_i(\mathbf{q}) + K_i\frac{d\sigma_i(\mathbf{q})}{dt} = \sigma_i + K_i\,\nabla\sigma_i^{\mathrm{T}}\,\dot{\mathbf{q}} \le 0\right\},\quad i = 1,\ldots,N \tag{11}$$

where $\sigma_i(\mathbf{q})$ is the original ith constraint function and K_i is a free design parameter that determines the rate of approach to the boundary of the ith -constraint. Then, the variable structure control law (8) will be considered, with $\phi_i(\mathbf{q},\dot{\mathbf{q}})$ as defined in (11).

Fig. 1 shows the proposed reference conditioning algorithm (RCA) to fulfill the environment limits given by (11). The objective of this conditioning is to instantaneously modify the desired reference $\mathbf{p}_{\mathrm{ref}}$ to a possibly different value $\mathbf{p}^*_{\mathrm{ref}}$ which is sent to the robotic control system when there is a risk of violating a given constraint.

Assumption 2: The relative degree between $\ddot{\mathbf{q}}$ and $\ddot{\mathbf{p}}^*_{\mathrm{ref}}$ is zero.

This assumption is satisfied if the robot control includes the typical feedforward of the second derivative of the desired reference [19]. For instance, if the operational space control is considered [20], a control law in the form

$$\ddot{\mathbf{q}}_d = \mathbf{J}_p^{-1}(\mathbf{q})\ddot{\mathbf{p}}^*_{\mathrm{ref}} + \mathbf{f}_c(\mathbf{p}^*_{\mathrm{ref}},\dot{\mathbf{p}}^*_{\mathrm{ref}},\mathbf{q},\dot{\mathbf{q}}) \tag{12}$$

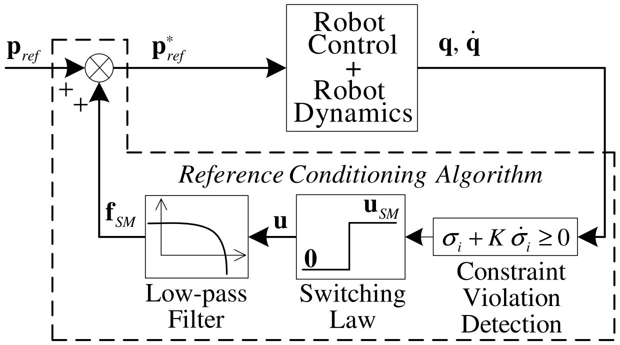

$\mathbf{p}_{ref}$
$\mathbf{p}^*_{ref}$
+
+
Robot Control + Robot Dynamics
$\mathbf{q}, \dot{\mathbf{q}}$
Reference Conditioning Algorithm
$\mathbf{f}_{SM}$
$\mathbf{u}$
$\mathbf{u}_{SM}$
0
$\sigma_i + K\,\dot{\sigma}_i \ge 0$
Low-pass Filter
Switching Law
Constraint Violation Detection

Fig. 1. Proposed SM reference conditioning loop to fulfill C-space constraints.

will verify the assumption, where $\ddot{\mathbf{q}}_d$ is the desired joint acceleration vector to be commanded as torque to the drives of each joint, and $\mathbf{f}_c$ is a generic vector function for error correction. Note that the actual joint acceleration $\ddot{\mathbf{q}} = \mathcal{C}\ddot{\mathbf{q}}_d + \mathbf{d}_c$ will not be exactly the commanded one ($\mathcal{C}$ represents the dynamics of the low-level robot control loop and $\mathbf{d}_c$ represents inaccuracies due to other torque disturbances; it will be assumed that the dynamics of $\mathcal{C}$ is fast enough compared to that of $\ddot{\mathbf{q}}_d$ so that the relationship $\ddot{\mathbf{q}} = \ddot{\mathbf{q}}_d + \mathbf{d}_c$ holds approximately true without the need of extra state variables in the model to be defined below).

To take advantage of the SM conditioning features described above, the commanded trajectory is shaped by modifying the position reference $\mathbf{p}_{\mathrm{ref}}$ as follows (Fig. 1):

$$\mathbf{p}^*_{\mathrm{ref}} = \mathbf{p}_{\mathrm{ref}} + \mathbf{f}_{\mathrm{SM}} \tag{13}$$

with $\mathbf{f}_{\mathrm{SM}}$ being the correcting term to the original reference trajectory that is generated by passing the discontinuous signal $\mathbf{u}$ through a low-pass filter, as shown in Fig. 1. This filter is needed for the sliding manifold to have unitary relative degree with respect to the discontinuous action, as required by SM theory [7]. In particular, for the problem at hand, the filter must be of second-order for $\dot{\phi}_i$ (and $\ddot{\mathbf{q}}$) to explicitly depend on $\mathbf{u}$, see (11)–(13). This filter also allows smoothing out the signal added to the main control loop depending on its cut-off frequency. Particularly, the following second-order Butterworth low-pass filter could be used:

$$\ddot{\mathbf{f}}_{\mathrm{SM}} = -\sqrt{2}\,\alpha\,\dot{\mathbf{f}}_{\mathrm{SM}} - \alpha^2\mathbf{f}_{\mathrm{SM}} + \alpha^2\mathbf{u} \tag{14}$$

with the scalar α being the filter cut-off frequency. Naturally, α should be taken for the filter to be significantly faster than the dynamics of the original position reference $\mathbf{p}_{\mathrm{ref}}$, in order to avoid degrading the performance of the RCA (see Section III-C.1).

The whole conditioning system can be cast within the theoretical framework of the previous section by constructing a dynamical system in the form of (4) with the state vector $\mathbf{x} = \left[\mathbf{f}_{\mathrm{SM}}^{\mathrm{T}}\ \mathbf{q}^{\mathrm{T}}\ \dot{\mathbf{f}}_{\mathrm{SM}}^{\mathrm{T}}\ \dot{\mathbf{q}}^{\mathrm{T}}\right]^{\mathrm{T}}$ and the disturbance vector $\mathbf{d} = \left[\mathbf{d}_c^{\mathrm{T}}\ \mathbf{p}_{\mathrm{ref}}^{\mathrm{T}}\ \dot{\mathbf{p}}_{\mathrm{ref}}^{\mathrm{T}}\ \ddot{\mathbf{p}}_{\mathrm{ref}}^{\mathrm{T}}\right]^{\mathrm{T}}$, and, therefore, from (12)–(14) we obtain the following state equation:

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{O}_n & \mathbf{O}_n & \mathbf{I}_n & \mathbf{O}_n \\ \mathbf{O}_n & \mathbf{O}_n & \mathbf{O}_n & \mathbf{I}_n \\ -\alpha^2\,\mathbf{I}_n & \mathbf{O}_n & -\sqrt{2}\alpha\,\mathbf{I}_n & \mathbf{O}_n \\ -\alpha^2\,\mathbf{J}_p^{-1}(\mathbf{q}) & \mathbf{O}_n & -\sqrt{2}\alpha\,\mathbf{J}_p^{-1}(\mathbf{q}) & \mathbf{O}_n \end{bmatrix}\mathbf{x} + \begin{bmatrix} \mathbf{0}_n \\ \mathbf{0}_n \\ \mathbf{0}_n \\ \mathbf{d}_c + \mathbf{f}_c'(\mathbf{x}, \mathbf{p}_{\text{ref}}, \dot{\mathbf{p}}_{\text{ref}}) + \mathbf{J}_p^{-1}(\mathbf{q})\ddot{\mathbf{p}}_{\text{ref}} \end{bmatrix} + \begin{bmatrix} \mathbf{O}_n \\ \mathbf{O}_n \\ \alpha^2\mathbf{I}_n \\ \alpha^2\mathbf{J}_p^{-1}(\mathbf{q}) \end{bmatrix}\mathbf{u} \qquad (15)$$

where $\mathbf{O}_n$ and $\mathbf{I}_n$ denote the null matrix and the identity matrix of dimensions $n \times n$, and function $\mathbf{f}_c'$ is obtained by substituting (13) into function $\mathbf{f}_c$.

Using the above state equation (15), the Lie derivatives of ϕ_i are given by

$$\nabla\phi_i = \partial\phi_i/\partial\mathbf{x} = \left[\mathbf{O}_n\ \nabla\sigma_i^{\mathrm{T}} + K_i\dot{\mathbf{q}}^{\mathrm{T}}\mathbf{H}_{\sigma i}\ \mathbf{O}_n\ K_i\nabla\sigma_i^{\mathrm{T}}\right]^{\mathrm{T}} \qquad (16)$$

$$\mathbf{L}_{\mathbf{g}}\phi_i = \nabla\phi_i^{\mathrm{T}}\,\mathbf{g} = \alpha^2\,K_i\,\nabla\sigma_i^{\mathrm{T}}\,\mathbf{J}_p^{-1} \qquad (17)$$

$$\begin{aligned} L_f\phi_i &= \nabla\phi_i^{\mathrm{T}}\,\mathbf{f} \\ &= \nabla\sigma_i^{\mathrm{T}}\,\dot{\mathbf{q}} + K_i\,\dot{\mathbf{q}}^{\mathrm{T}}\,\mathbf{H}_{\sigma i}\,\dot{\mathbf{q}} + K_i\,\nabla\sigma_i^{\mathrm{T}} \\ &\quad \times \Big(-\alpha^2\,\mathbf{J}_p^{-1}\,\mathbf{f}_{\mathrm{SM}} - \sqrt{2}\,\alpha\,\mathbf{J}_p^{-1}\,\dot{\mathbf{f}}_{\mathrm{SM}} \\ &\qquad + \mathbf{d}_c + \mathbf{f}_c' + \mathbf{J}_p^{-1}\,\ddot{\mathbf{p}}_{\mathrm{ref}}\Big) \end{aligned} \qquad (18)$$

where $\mathbf{H}_{\sigma i}$ denotes the Hessian matrix of second-order partial derivatives of σ_i, and thus the matrix $\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}$ becomes

$$\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi} = \alpha^2\mathbf{K}\nabla\boldsymbol{\sigma}^{\mathrm{T}}\mathbf{J}_p^{-1} \qquad (19)$$

where $\mathbf{K}$ is a diagonal matrix with diagonal entries K_i, and matrix $\nabla\boldsymbol{\sigma}$ contains the gradient vectors $\nabla\sigma_i$ of all active constraints.

Theorem 1: If $\ddot{\mathbf{p}}_{\text{ref}}$, $\mathbf{d}_c$, $\mathbf{f}_c'$, and $\mathbf{H}_{\sigma i}$ are bounded, and the matrix $\nabla\boldsymbol{\sigma}^{\mathrm{T}}\mathbf{J}_p^{-1}$ is full row rank, the reference conditioning proposed in Fig. 1 achieves invariance of the set Φ_{CS}^* and of the set Φ_{CS} using the variable structure control law (8)–(10) with the system Lie derivatives in (18) and (19).

The proof is easily crafted after considering the Lie derivatives in (18) and (19) and the results in previous section, and along with the fact that $\Phi_{\mathrm{CS}}^* \subset \Phi_{\mathrm{CS}}$ and that the conditions in the theorem statement ensure $L_f\phi_i$ is bounded.

Some important issues about the proposed conditioning approach are discussed below.

1) Sufficient Condition for SM: The matrix $\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}$ (19) must be full row rank to ensure that there exists a finite value of u^+ satisfying the sufficient condition (10) for the establishment of the proposed SM conditioning with control action (8), (9). From the above definitions and assumptions, the values of α, K_i, and $\nabla\sigma_i$ are not zero and, therefore, this condition for only one active constraint (i.e., $\mathbf{L}_{\mathbf{g}}\phi_i \neq \mathbf{0}_n^{\mathrm{T}}$) is satisfied for nonsingular configurations of the robot [21]. Furthermore, for multiple active constraints and nonsingular configurations ($\mathbf{J}_p^{-1}$ has full rank), the condition can be restated as requiring matrix $\nabla\boldsymbol{\sigma}^{\mathrm{T}}$ to be full row rank, i.e., as requiring the gradient vectors $\nabla\sigma_i$ of the active constraints to be linearly independent. In fact, the linear dependency of the gradient vectors could be seen as a "singularity" of the proposed method that yields two possibilities, which can be illustrated by the following simple example. Consider two parallel gradient vectors. On the one hand, if both gradient vectors are pointing in the same direction (i.e., they could be seen as redundant vectors), the proposed SM conditioning is properly established with control action (8), (9) although sufficient condition (10) is not satisfied. On the other hand, if both vectors are pointing in opposite directions (i.e., they could be seen as incompatible vectors), the proposed SM conditioning cannot be properly established and the values of $L_f\phi_i$ are relevant to ensure, if possible, the invariance of Φ.

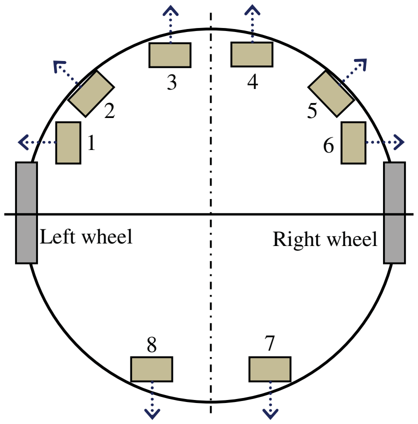
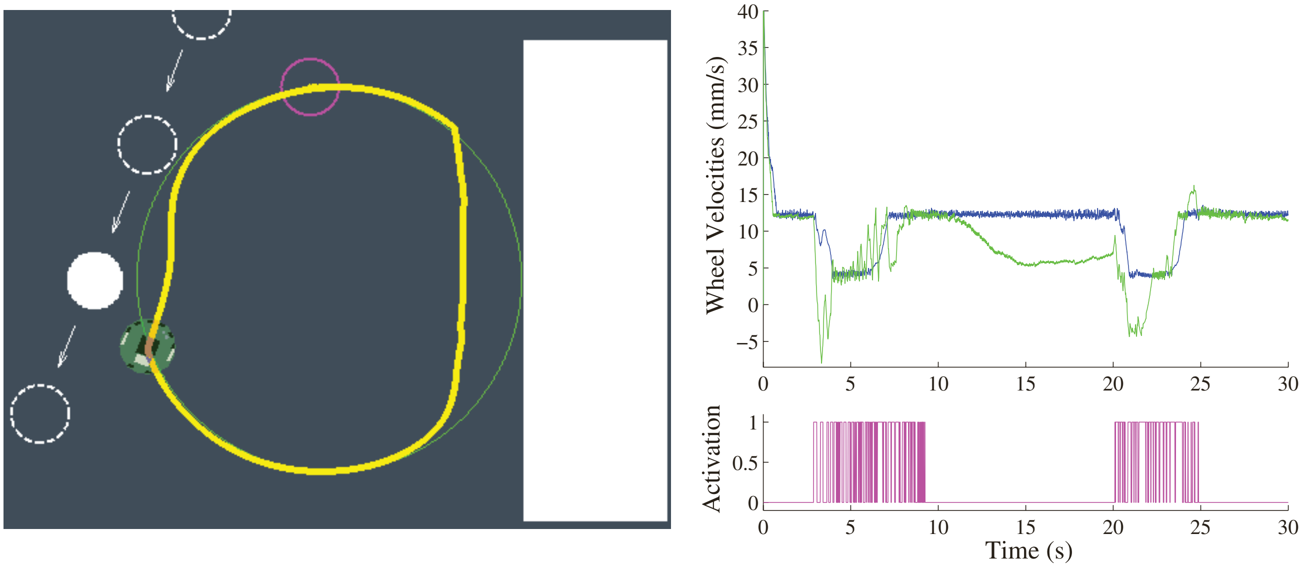


Fig. 2. Khepera mobile robot (schematic representation of the two differential drive wheels and the eight infrared sensors).

2) Robustness: First, it is worth mentioning that the proposed approach is akin to an auxiliary supervision of the reference signal in order to avoid violating constraints, and is not intended as the main robot control algorithm itself. Thus, intuitively, changing references to avoid violating constraints should not affect the internal robot control loop stability/robustness. Therefore, we discuss in this section the robustness of the proposed method to achieve geometric invariance, i.e., the robustness of the SM conditioning given by (8)–(10) to satisfy (7).

Although equations (10) and (18)–(19) derive an analytical lower bound for u^+ to be used in (9), this scalar factor can be set conservatively large in practice, as usual in SM applications. In this way, it is guaranteed that the term $\mathbf{L}_{\mathbf{g}}\phi_i(\mathbf{x})\mathbf{u}$ is dominating over the term $L_f\phi_i(\mathbf{x}, \mathbf{d})$ in (7). Hence, the geometric invariance given by (7) is satisfied regardless of the value of $L_f\phi_i$ (18) (as long as it is bounded) and without explicit knowledge of it. Therefore, it can be concluded that the proposed approach is robust against: the robot control (i.e., against $\mathbf{d}_c$ and $\mathbf{f}_c'$), as long as Assumption 2 is fulfilled, the workspace reference $\mathbf{p}_{\text{ref}}$ and its derivatives, and the Hessian matrices $\mathbf{H}_{\sigma i}$ of the constraint functions. In contrast, since the value of $\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}$ (19) is required to be known in order to compute $\mathbf{u}_{\mathrm{SM}}$ in (9), the proposed approach is sensitive to the gradient vector $\nabla\sigma_i$ of the constraint functions and the Jacobian matrix $\mathbf{J}_p$ of the robot. However, the proposed method allows the use of inaccurate values for $\nabla\sigma_i$ and $\mathbf{J}_p$ in



(a) (b)

Fig. 3. Simulation in KiKS environment. (a) Reference path (thin line), path followed by the robot (thick line), static obstacle (white rectangle), moving obstacle at 22 s (white circle), robot at 22 s (Khepera image), sequence followed by the moving obstacle (dashed circles and arrows) and robot's initial and final position (thick circle). (b) Wheel velocities and constraint activation.

control action (9) as long as $\mathbf{L_g}\boldsymbol{\phi}\widehat{\mathbf{L_g}\boldsymbol{\phi}}^{\mathrm{T}}\mathbf{1}_b > \mathbf{0}_b$, where $\widehat{\mathbf{L_g}\boldsymbol{\phi}}$ is the computed value of $\mathbf{L_g}\boldsymbol{\phi}$ using the inaccurate data.

The proposed method is also sensitive to noisy measurements of $\mathbf{q}$ and $\dot{\mathbf{q}}$ since the condition $\phi_i(\mathbf{q}, \dot{\mathbf{q}}) \geq 0$ must be evaluated in the variable structure control law (8) in order to properly switch between $\mathbf{0}_{n_u}$ and $\mathbf{u}_{\mathrm{SM}}$. Therefore, for security reasons, the constraint functions should be designed conservatively including a security margin to take into account possible differences between the actual value of ϕ_i and that obtained with noisy measurements. This security margin should also be designed to compensate the estimated chattering amplitude (see Section III-C.1), reference tracking errors (e.g., in case that Assumption 2 is not completely fulfilled), etc.

3) Moving Constraints: The proposed approach can also be used if there are moving constraints, e.g., moving obstacles. In this case, σ_i also depends explicitly on time, and, hence, the new expression ϕ_i^* for the constraint function and its derivative are given by

$$\phi_i^* = \sigma_i + K_i\left(\frac{\partial\sigma_i}{\partial\mathbf{q}}^{\mathrm{T}}\dot{\mathbf{q}} + \frac{\partial\sigma_i}{\partial t}\right) \tag{20}$$

$$\begin{aligned}\dot{\phi}_i^* = {} & \frac{\partial\sigma_i}{\partial\mathbf{q}}^{\mathrm{T}}\dot{\mathbf{q}} + \frac{\partial\sigma_i}{\partial t} + K_i\left(\frac{\partial\sigma_i}{\partial\mathbf{q}}^{\mathrm{T}}\ddot{\mathbf{q}} + \frac{\partial^2\sigma_i}{\partial\mathbf{q}\partial t}^{\mathrm{T}}\dot{\mathbf{q}} + \dot{\mathbf{q}}^{\mathrm{T}}\frac{\partial^2\sigma_i}{\partial\mathbf{q}^2}\dot{\mathbf{q}}\right)\\ & + K_i\left(\frac{\partial^2\sigma_i}{\partial t^2} + \frac{\partial^2\sigma_i}{\partial t\partial\mathbf{q}}^{\mathrm{T}}\dot{\mathbf{q}}\right),\end{aligned} \tag{21}$$

and comparing this result with that for stationary constraints, yields the following relation:

$$\begin{aligned}\dot{\phi}_i^* &= \dot{\phi}_i + \left(\frac{\partial\sigma_i}{\partial t} + 2K_i\frac{\partial^2\sigma_i}{\partial\mathbf{q}\partial t}^{\mathrm{T}}\dot{\mathbf{q}} + K_i\frac{\partial^2\sigma_i}{\partial t^2}\right)\\ &= \dot{\phi}_i + M_{\phi_i}\\ &= L_f\phi_i + \mathbf{L_g}\phi_i\,\mathbf{u} + M_{\phi_i}\\ &= L_f\phi_i^* + \mathbf{L_g}\phi_i\,\mathbf{u},\end{aligned} \tag{22}$$

where $\mathbf{L_g}\phi_i$ and $L_f\phi_i$ are given again by (17) and (18). Therefore, all developments remain unchanged except for changing $L_f\phi_i$ to $L_f\phi_i^* = L_f\phi_i + M_{\phi_i}$. Thus, only the value of the lower bound for u^+ is changed when moving constraints are considered and, hence, the iterative computation of the RCA remains the same. Again, as discussed above, a high-enough constant u^+ will suffice for practical implementation.

B. Reactive Reference Conditioning

The above algorithm can be directly applied in structured environments, where the constraints σ_i can be evaluated from the robot coordinates and a model of the environment.

To use the proposed RCA as an on-line reactive algorithm for collision avoidance using the robot proximity sensors, such as infrared or ultrasonic sensors, the following constraint function is considered for each sensor:

$$\sigma_i(\mathbf{q}) = \varepsilon_i - \rho_i(\mathbf{q}) \tag{23}$$

where ρ_i is the sensor's measurement, i.e., the distance from the sensor to the detected obstacle in the sensor's direction, and ε_i is the desired minimum distance to the obstacle including the security margin mentioned in Section III-A.2.

From (23), the switching function $\phi_i = \sigma_i + K_i\dot{\sigma}_i$ must be generated, i.e., $\dot{\rho}_i$ must be obtained either by additional physical measurements or by numerical differentiation plus low-pass filtering of the reading ρ_i. Many industrial robot sensors have low-level capabilities in this respect and provide position and speed readings with a particular bandwidth. The sensor bandwidth will, obviously, limit the bandwidth of the references that can be followed.

Additionally, the algorithm requires the partial derivatives of σ_i (i.e., those of ρ_i) with respect to $\mathbf{q}$ to obtain $\mathbf{L_g}\phi_i$ in (17). This will require the knowledge of the partial derivatives with respect to both position and orientation of the sensor as well as the geometry of the part of the obstacle subject to detection. In an unstructured environment, it will require estimating the normal to the surface at the detected point, denoted as $\mathbf{n}_i$, and solving a triangle geometry problem as follows. Let us denote as $\overline{\mathbf{p}}_{s,i}$ the Cartesian sensor position and $\overline{\mathbf{p}}_{d,i}$ that of the detected point, and $\mathbf{s}_i$ as the unit vector pointing in the sensor direction, i.e., $\overline{\mathbf{p}}_{d,i} - \overline{\mathbf{p}}_{s,i} = \rho_i \mathbf{s}_i$. For small variations of the sensor position and orientation, the distance is given by (the details of the derivation are omitted for brevity)

$$\rho_i = \frac{\mathbf{n}_i^{\mathrm{T}}(\overline{\mathbf{p}}_{d,i} - \overline{\mathbf{p}}_{s,i})}{\mathbf{n}_i^{\mathrm{T}}\mathbf{s}_i} \tag{24}$$

hence

$$\frac{\partial \sigma_i}{\partial \overline{\mathbf{p}}_{s,i}} = -\frac{\partial \rho_i}{\partial \overline{\mathbf{p}}_{s,i}} = \frac{\mathbf{n}_i}{\mathbf{n}_i^{\mathrm{T}}\mathbf{s}_i} \tag{25}$$

$$\frac{\partial \sigma_i}{\partial \mathbf{s}_i} = -\frac{\partial \rho_i}{\partial \mathbf{s}_i} = \frac{\mathbf{n}_i^{\mathrm{T}}(\overline{\mathbf{p}}_{d,i} - \overline{\mathbf{p}}_{s,i})}{(\mathbf{n}_i^{\mathrm{T}}\mathbf{s}_i)^2}\mathbf{n}_i = \rho_i \frac{\mathbf{n}_i}{\mathbf{n}_i^{\mathrm{T}}\mathbf{s}_i} \tag{26}$$

and the gradient vector results in

$$\begin{aligned}\nabla\sigma_i &= \frac{\partial \sigma_i}{\partial \mathbf{q}} \\ &= \left(\frac{\partial \overline{\mathbf{p}}_{s,i}}{\partial \mathbf{q}}\right)^{\mathrm{T}} \frac{\partial \sigma_i}{\partial \overline{\mathbf{p}}_{s,i}} + \left(\frac{\partial \mathbf{s}_i}{\partial \mathbf{q}}\right)^{\mathrm{T}} \frac{\partial \sigma_i}{\partial \mathbf{s}_i} \\ &= \left(\mathbf{J}_{ps,i}^{\mathrm{T}} + \mathbf{J}_{s,i}^{\mathrm{T}}\rho_i\right)\frac{\mathbf{n}_i}{\mathbf{n}_i^{\mathrm{T}}\mathbf{s}_i}\end{aligned} \tag{27}$$

where $\mathbf{J}_{ps,i}$ and $\mathbf{J}_{s,i}$ are the Jacobian matrices between the sensor position $\overline{\mathbf{p}}_{s,i}$ and direction $\mathbf{s}_i$, respectively, and the robot configuration $\mathbf{q}$, which are obtained from the robot kinematics.

In the case the vector $\mathbf{n}_i$ is not available,[3] a reasonable assumption would be assuming that the obstacle surface is perpendicular to the sensor direction (i.e., $\mathbf{n}_i \equiv \mathbf{s}_i$, so $\mathbf{n}_i^{\mathrm{T}}\mathbf{s}_i = 1$). In this case, the gradient vector results in

$$\nabla\sigma_i \approx \left(\mathbf{J}_{ps,i}^{\mathrm{T}} + \left(\mathbf{s}_i^{\perp}\right)^{\mathrm{T}}\rho_i\right)\mathbf{s}_i = \mathbf{J}_{ps,i}^{\mathrm{T}}\mathbf{s}_i \tag{28}$$

where $\mathbf{s}_i^{\perp}$ is a vector orthogonal to $\mathbf{s}_i$ (the derivatives of a vector of a constant modulus are always orthogonal to it). Note that using approximation (28) results in a lower bound for the switching action u^+, which, in general, will be, higher than that obtained with the actual vector $\mathbf{n}_i$.

C. Additional Remarks About the Reference Conditioning

1) Chattering: Although, in theory, sliding modes are produced by infinite-frequency commutation, in every practical SM implementation, finite-frequency commutation makes the system leave the ideal SM and oscillate with finite frequency and amplitude inside a band around $\boldsymbol{\phi} = \mathbf{0}$, which is known as chattering [8]. In the proposed software-based SM conditioning, the switching frequency is directly the inverse of the sampling period. An upper bound for the chattering band $\triangle\phi_i$ of the proposed RCA can be derived [10] using the Euler integration

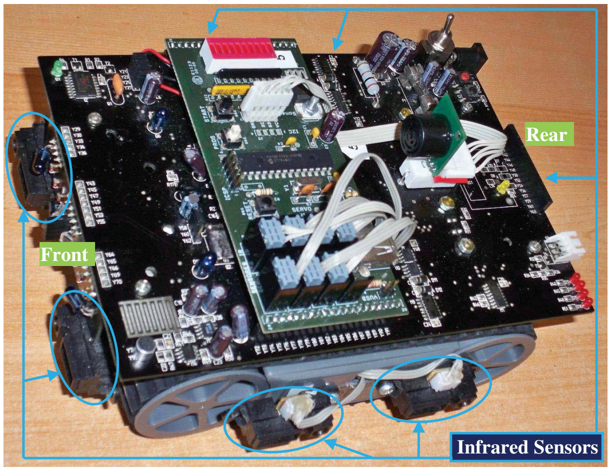


Fig. 4. Mobile robot used for the real experimentation.

$$\begin{aligned}\triangle\phi_i &= T_{\mathrm{RC}}\,|\mathbf{L_g}\phi_i\,\mathbf{u}_{SM}| \\ &= T_{\mathrm{RC}}\,\alpha^2\,K_i\,|\nabla\sigma_i^{\mathrm{T}}\,\mathbf{J}_p^{-1}\,\mathbf{u}_{SM}| \\ &\le T_{\mathrm{RC}}\,\alpha^2\,K_i\,\|\mathbf{u}_{SM}\|_2\,\|\nabla\sigma_i^{\mathrm{T}}\,\mathbf{J}_p^{-1}\|_2\end{aligned} \tag{29}$$

where T_{RC} is the sampling period of the RCA, $\|\mathbf{u}_{SM}\|_2$ is the amplitude of the control action, and $\|\nabla\sigma_i^{\mathrm{T}}\,\mathbf{J}_p^{-1}\|_2$ is the amplitude of a vector which depends on the gradient vector used in the control action and the robot configuration.

2) Trap Situations: Since the proposed RCA does not perform high-level planning, the robot may be trapped to guarantee the fulfillment of the constraints while the reference keeps moving forward. Note that trap situations can be detected easily in most cases by monitoring the difference between the user-supplied reference and the current position. This phenomenon is analogous to local minima in potential-field-based approaches and it can be avoided if the proposed approached is combined with a higher level planner [22] that detects the blocking situation and modifies the reference path that was originally planned. For instance, similar to potential-field-based approaches, a random walk [1] could be activated when the trap situation is detected in order to escape from it, see [4] for further details. Of course, this is only a basic low-level strategy which should be replaced by other more sophisticated ones if possible. There are some situations in which trap situations can be avoided.

Proposition 2: If the constraints are made only when the robot's pose $\mathbf{p}$ and the allowed workspace is star shaped,—i.e., there exists a point $\mathbf{p}_k$ (kernel), not necessarily unique, that can be connected to any point on the region boundary by a segment lying entirely in the interior of the region [23]—trap situations are avoided if (9) is replaced by

$$\mathbf{u}_{SM} = (\mathbf{p}_k - \mathbf{p})\,u^+ \tag{30}$$

that is, if a vector pointing toward a point of the region kernel is used (instead of the constraint gradient vector).

[3] In general, vector $\mathbf{n}_i$ could be estimated from three noncollinear detected points, or from a camera. These issues are, however, out of the scope of the present brief as the goal is to present a simple low-level algorithm with low computational cost. Approximation (28) is proposed.

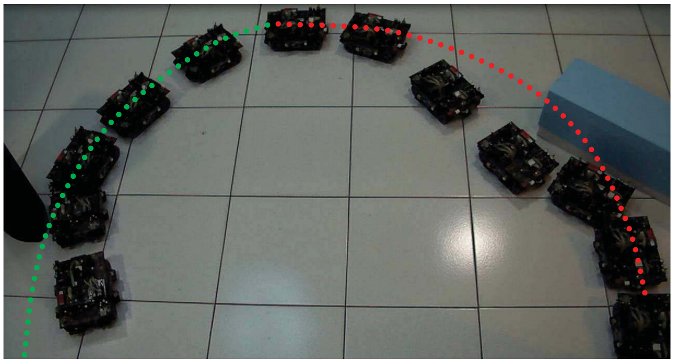


Fig. 5. Combination of frames of the real experiment with the mobile robot. Dotted line: reference path.

The proof is straightforward if one realizes that $\mathbf{u}_{\mathrm{SM}}$, by definition, points towards the interior of the allowed region, see Section II-C. Note also that, if the allowed workspace is convex (any point on a line between two allowed points is an allowed point), which is a particular case of star-shaped regions, all the allowed points belong to the kernel and, thus, any allowed point can be used as $\mathbf{p}_k$ in (30).

3) Speed Regulation: If the reference trajectory is expressed as a function of a parameter λ, i.e., $\mathbf{p}_{\mathrm{ref}} = \mathbf{f}_p(\lambda)$, the proposal in this brief can be combined with the speed auto-regulation technique in [24] in order to handle joint speed limits commanded to the lower-level control while tracking the required path as fast as possible. The details are omitted for brevity. Note, however, that combined handling of torque saturation and reference processing would need a different theoretical framework (high-level planning, prediction), as local gradients (as proposed here) have a direct relation (zero relative degree) to speed, but cannot predict significant changes of curvature far ahead.

IV. Results and Experiments

A simulation and an experiment have been provided to show the possibilities of the approach.

A. Simulation (Khepera)

The proposed approach has been tested in simulation for the Khepera mobile robot, which is a small robot (55 mm in diameter) with two differential driving wheels, and equipped with eight infrared sensors placed around it (Fig. 2). This mobile robot can track two-dimensional reference trajectories, e.g., the position of a point in the XY plane. For this section, it will be assumed that the tracking point (i.e., the point of the mobile robot that tracks the reference) is located on the robot symmetry axis and at a distance e from the wheel axle. This point of the robot represents the location of a tool, e.g., camera, flashlight, etc.

The behavior of this mobile robot has been simulated in MATLAB using the freely accessible software KiKS [25], which is a specific simulator for the Khepera robot. This software provides an accurate robot simulation since it includes a mathematical model for the robot dynamics, the wheel motors, the sensor noise, etc.

The proposed RCA has been implemented in MATLAB with the following parameter values: $\varepsilon_i = 40$ mm, $K_i = 0.3$ s, $\alpha = 1$ rad/s, and $T_{\mathrm{RC}} = 0.05$ s. For simplicity, the gradient vector approximation (28) has been used. Moreover, indirect control is considered for the mobile robot, where the classical trajectory control is used for the kinematic control loop in order to compute the desired wheel velocities [26]. For the simulation, the tracking point has been located with $e = 20$ mm, the reference trajectory has been generated as a circumference of 400 in diameter with a tangential velocity of $40\pi/3$ mm/s, and two obstacles have been considered: a static rectangular obstacle, and a moving circular obstacle. Fig. 3 shows the simulation results for the Khepera robot, where it can be seen how the RCA has effectively modified the original reference in order to satisfy the sensor constraints [Fig. 3(a)] and the resulting wheel velocities [Fig. 3(b)].

B. Experiment

Real experiments have been conducted using a self-developed differential-drive mobile robot, see Fig. 4. Based on the robby RP5 chassis (rectangular shape of 200 mm in length and 100 mm in width), the robot has been equipped with seven low-cost infrared sensors—two in the front, one in the rear, and two on both sides of the robot. This simple mobile robot allows to shows the viability of using the proposed RCA even on a platform that is neither sophisticated nor technologically advanced.

The proposed RCA and the robot control have been implemented in the robot microcontroller under the same conditions as the above simulation example, except for parameter ε_i which was set to 60 mm. For the experiment, the tracking point has been located, as in the previous simulation, at $e = 100$ mm of distance to the robot center in the longitudinal axis of symmetry. The reference trajectory consists of a quarter circular arc (950 mm in radius, depicted in red color in Fig. 5) and a half elliptical arc (950 mm in longer radius and 500 mm in shorter radius, depicted in green color) with a tangential velocity of 40 mm/s. Moreover, two obstacles have been considered. Fig. 5 shows the combination of sample video frames captured during the experiment, where it can be observed how the RCA has effectively modified the original reference for robot collision avoidance.

V. Conclusion

A variable structure algorithm for robotic reference conditioning was proposed using geometric invariance and SM concepts. The strategy acted as a supervisory loop, shaping the desired trajectory so that it is always compatible with the C-space constraints given by the measurements of the robot's sensors. In this manner, the algorithm activates when the desired reference is about to violate the C-space constraints, modifying it as much as necessary in order to fulfill all the constraints and reaching their limit surface at low speed.

The proposed approach can be easily added as an auxiliary supervisory loop to robotic systems, and its implementation is very easy in a few program lines of a microprocessor, see the RCA function in the Appendix. Moreover, the proposed algorithm improves the conventional conservative potential-field-based approach for collision avoidance in the sense that it fully exploits the robot workspace to maintain the faithfulness to the original reference trajectory.

Although the algorithm was illustrated both in simulation and experimentally for a particular mobile robot equipped with infrared sensors, the conclusions drawn for the reference conditioning method also apply to other robotic systems and proximity sensors.

Appendix

Proof: From (7)–(9), the column vector $\dot{\boldsymbol{\phi}}$ composed of the constraint function derivatives $\dot{\phi}_i$ of the b active constraints is given by

$$\dot{\boldsymbol{\phi}} = L_f\boldsymbol{\phi} - \left(\mathbf{L_g}\boldsymbol{\phi}\, \mathbf{L_g}\boldsymbol{\phi}^{\mathrm{T}}\right) \mathbf{z}\, u^+ \tag{31}$$

where $L_f\boldsymbol{\phi}$ is a column vector containing the elements $L_f\phi_i$ of the b active constraints, and $\mathbf{z}$ is a column vector whose ith-component z_i is defined to be 1 if $\phi_i \geq 0$ and 0 otherwise. Note that this switching function $z_i(\phi_i) = \{1, 0\}$ replaces the classical one used in conventional SMsliding mode [9], i.e., $\mathrm{sign}(\phi_i) = \{1, -1\}$, because of the fact that the proposed method only performs "one-side" sliding control, see Section II-C.

The goal of this proof is to show that $\boldsymbol{\phi} = \mathbf{0}_b$ is an asymptotically stable equilibrium point with finite time convergence. For this purpose, let $V = \mathbf{z}^{\mathrm{T}}\boldsymbol{\phi}$ be a Lyapunov function candidate. Vector $\boldsymbol{\phi}$ can be generically partitioned into two subvectors $\boldsymbol{\phi} = [\boldsymbol{\phi}^{a\,\mathrm{T}} \;\; \boldsymbol{\phi}^{b-a\,\mathrm{T}}]^{\mathrm{T}}$, where SM occurs in the manifold given by $\boldsymbol{\phi}^a = \mathbf{0}_a$, whereas the components of vector $\boldsymbol{\phi}^{b-a}$ are greater than zero. Obviously, one of these two subvectors may be empty at a certain time. According to the continuous equivalent control [7], vector $\mathbf{z}^a$ must be replaced by the function $\mathbf{z}^a_{eq}$ such that $\dot{\boldsymbol{\phi}}^a = \mathbf{0}_a$. Because $\boldsymbol{\phi}^a = \mathbf{0}_a$ in SM, the time derivative of V is given by

$$\begin{aligned}
\dot{V} &= \frac{d}{dt}\left(\mathbf{z}^{\mathrm{T}}\boldsymbol{\phi}\right) \\
&= \frac{d}{dt}\left(\begin{bmatrix}\mathbf{z}^a_{eq} \\ \mathbf{1}_{b-a}\end{bmatrix}^{\mathrm{T}} \begin{bmatrix}\boldsymbol{\phi}^a \\ \boldsymbol{\phi}^{b-a}\end{bmatrix}\right) \\
&= \begin{bmatrix}\dot{\mathbf{z}}^a_{eq} \\ \mathbf{0}_{b-a}\end{bmatrix}^{\mathrm{T}} \begin{bmatrix}\mathbf{0}_a \\ \boldsymbol{\phi}^{b-a}\end{bmatrix} + \mathbf{z}^{\mathrm{T}}\dot{\boldsymbol{\phi}} \\
&= \mathbf{z}^{\mathrm{T}}\dot{\boldsymbol{\phi}}.
\end{aligned} \tag{32}$$

Function RCA($\mathbf{p}_{\mathrm{ref}}, \mathbf{q}, \dot{\mathbf{q}}$)

```
$\mathbf{k} = \mathbf{0}_n$;
for $i \leftarrow 1$ to $N$ do
    if $\phi_i(\mathbf{q}, \dot{\mathbf{q}}) \geq 0$ then $\mathbf{k} = \mathbf{k} + \nabla\sigma_i(\mathbf{q})$;
end
$\mathbf{k} = -\left(\mathbf{J}_p^{-1}(\mathbf{q})\right)^{\mathrm{T}} \mathbf{k}$ ;                    // (9)
if $\|\mathbf{k}\|_2 \leq 10^{-6}$ then $\mathbf{u} = \mathbf{0}_n$;
else $\mathbf{u} = \mathbf{k}\|\mathbf{u}_{\mathrm{SM}}\|_2/\|\mathbf{k}\|_2$ ;                    // (8)
$\mathbf{f}_{\mathrm{SM}}$ = FilterSecondOrder ($\alpha, \mathbf{u}$);                    // (14)
$\mathbf{p}^*_{\mathrm{ref}} = \mathbf{p}_{\mathrm{ref}} + \mathbf{f}_{\mathrm{SM}}$;                    // (13)
return $\mathbf{p}^*_{\mathrm{ref}}$;
```

Replacing vector $\dot{\boldsymbol{\phi}}$ with its value from (31), we obtain

$$\dot{V} = \mathbf{z}^{\mathrm{T}}\, L_f\boldsymbol{\phi} - \mathbf{z}^{\mathrm{T}}\left(\mathbf{L_g}\boldsymbol{\phi}\, \mathbf{L_g}\boldsymbol{\phi}^{\mathrm{T}}\right) \mathbf{z}\, u^+. \tag{33}$$

The components of vector $\mathbf{z}$ range from 0 to 1, hence the upper bound of the first term in (33) is given by $z_i = 1$ if $L_f\phi_i > 0$ and $z_i = 0$ if $L_f\phi_i < 0$, that is

$$\mathbf{z}^{\mathrm{T}}\, L_f\boldsymbol{\phi} \leq \sum_{i=1}^{b}\left(\max(L_f\phi_i, 0)\right). \tag{34}$$

Assuming that $u^+ > 0$, the second term in (33) is negative, since the 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

$$-\mathbf{z}^{\mathrm{T}}\left(\mathbf{L_g}\boldsymbol{\phi}\, \mathbf{L_g}\boldsymbol{\phi}^{\mathrm{T}}\right)\mathbf{z}\, u^+ \leq -\mathrm{eig}_{\min}\left(\mathbf{L_g}\boldsymbol{\phi}\mathbf{L_g}\boldsymbol{\phi}^{\mathrm{T}}\right)\|\mathbf{z}\|_2^2 u^+ \tag{35}$$

where

$$\|\mathbf{z}\|_2 \geq 1 \quad \forall\, \boldsymbol{\phi} \neq \mathbf{0}_b. \tag{36}$$

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

From (34)–(36), the upper bound of the time derivative of the Lyapunov function V results in

$$\dot{V} \leq \sum_{i=1}^{b}\left(\max(L_f\phi_i, 0)\right) - \mathrm{eig}_{\min}\left(\mathbf{L_g}\boldsymbol{\phi}\, \mathbf{L_g}\boldsymbol{\phi}^{\mathrm{T}}\right) u^+. \tag{37}$$

Therefore, if u^+ fulfills (10), the Lyapunov function decays at a finite rate and, it vanishes, and collective SM in the intersection of the b active constraints occurs after a finite time interval. That is, the origin $\boldsymbol{\phi} = \mathbf{0}_b$ is an asymptotically stable equilibrium point with finite time convergence. ■

The pseudo-code of the proposed RCA (8), (9), (13), and (14) is shown in the function above. The algorithm uses the following auxiliary functions: the constraint functions $\phi_i(\mathbf{q}, \dot{\mathbf{q}})$; the gradient vectors $\nabla\sigma_i(\mathbf{q})$; the Jacobian matrix of the robot $\mathbf{J}_p(\mathbf{q})$; and function FilterSecondOrder $(\alpha, \mathbf{u})$, which is a discrete time implementation of the low-pass filter (14) (obviously, the filter implementation must take care of preserving its internal states between calls). Note that this implementation supports the claim made in this brief that the proposed approach only requires a few program lines. For the simulation example in Section IV-A, the computation time per iteration of the RCA in a modern computer using MATLAB R2009a (compiled C-MEX-file) was around 0.8 μs.

For reactive reference conditioning, evaluation of $\phi_i(\mathbf{q}, \dot{\mathbf{q}})$ would require reading the corresponding sensors, subtracting the threshold ε_i, and numerically estimating its derivative with well-known algorithms, see (23).

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments which helped us to improve the quality of this brief.

REFERENCES

[1] J.-C. Latombe, *Robot Motion Planning*. Boston: Kluwer, 1991.

[2] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. Syst., Man, Cybern.*, vol. 19, no. 5, pp. 1179–1187, Oct. 1989.

[3] F. Belkhouche and B. Bendjilali, "Reactive path planning for 3-D autonomous vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 1, pp. 249–256, Jan. 2012.

[4] L. Gracia, A. Sala, and F. Garelli, "A path conditioning method with trap avoidance," *Robot. Autonom. Syst.*, vol. 60, no. 6, pp. 862–873, 2012.

[5] L. Gracia, A. Sala, and F. Garelli, "A supervisory loop approach to fulfill workspace constraints in redundant robots," *Robot. Autonom. Syst.*, vol. 60, no. 1, pp. 1–15, 2012.

[6] L. Gracia, F. Garelli, and A. Sala, "Integrated sliding-mode algorithms in robot tracking applications," *Robot. Comput.-Integr. Manuf.*, vol. 29, no. 1, pp. 53–62, 2013.

[7] C. Edwards and S. Spurgeon, *Sliding Mode Control: Theory and Applications*, 1st ed. U.K.: Taylor and Francis, 1998.

[8] W. Perruquetti and J. Barbot, "Control Engineering Series," in *Sliding Mode Control in Engineering*, Eds., New York: Marcel Dekker, 2002.

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

[10] F. Garelli, R. Mantz, and H. De Battista, *Advanced Control for Constrained Processes and Systems*. London, U.K., IET, Control Engineering Series, 2011.

[11] F. Garelli, R. Mantz, and H. De Battista, "Limiting interactions in decentralized control of MIMO systems," *J. Process Control*, vol. 16, no. 5, pp. 473–483, 2006.

[12] V. Gazi and R. Ordonez, "Target tracking using artificial potentials and sliding mode control," *Int. J. Control*, vol. 80, no. 10, pp. 1626–1635, 2007.

[13] A. Ferrara and M. Rubagotti, "Second order sliding mode control of a mobile robot based on a harmonic potential field," *IET Control Theory Appl.*, vol. 2, no. 9, pp. 807–818, 2008.

[14] B. Park, S. Yoo, J. Park, and Y. Choi, "Adaptive neural sliding mode control of nonholonomic wheeled mobile robots with model uncertainty," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 1, pp. 207–2014, Jan. 2009.

[15] V. Sankaranarayanan and A. Mahindrakar, "Control of a class of underactuated mechanical systems using sliding modes," *IEEE Trans. Robot.*, vol. 25, no. 2, pp. 459–467, Feb. 2009.

[16] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robot. Res.*, vol. 5, no. 1, pp. 90–98, 1986.

[17] E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robot. Autom.*, vol. 8, no. 5, pp. 501–518, Oct. 1992.

[18] J. Angeles, *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*, 3rd ed., New York, NJ: Springer-Verlag, 2007.

[19] L. Sciavicco and B. Siciliano, *Modeling and Control of Robot Manipulators*. New York, NJ: McGraw-Hill, 1996.

[20] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. London, U.K.: Springer-Verlag, 2009.

[21] L. Gracia, J. Andres, and J. Tornero, "Trajectory tracking with a 6R serial industrial robot with ordinary and non-ordinary singularities," *Int. J. Control, Autom. Syst.*, vol. 7, no. 1, pp. 85–96, 2009.

[22] J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *Int. J. Robot. Res.*, vol. 10, no. 6, pp. 628–649, 1991.

[23] R. Larson and D. Falvo, *Elementary Linear Algebra*. Pacific Grove, CA: Brooks/Cole, 2008.

[24] F. Garelli, L. Gracia, A. Sala, and P. Albertos, "Sliding mode speed auto-regulation technique for robotic tracking," *Robot. Autonom. Syst.*, vol. 59, nos. 7–8, pp. 519–529, 2011.

[25] T. Nilsson, "KiKS is a Khepera Simulator," M.S. Thesis, Umeå Univ., Stockholm, Sweden, 2001.

[26] L. Gracia and J. Tornero, "Kinematic control of wheeled mobile robots," *Latin Amer. Appl. Res.*, vol. 38, no. 1, pp. 7–16, 2008.