A Supervisory Loop Approach to Fulfill Workspace Constraints in Redundant Robots

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

[a] *Dept. of Systems Engineering and Control, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain (e-mails: luigraca@isa.upv.es, asala@isa.upv.es). * Corresponding author*

[b] *CONICET and Universidad Nacional de La Plata, C.C.91 (1900), La Plata, Argentina (e-mail: fabricio@ing.unlp.edu.ar).*

Abstract

An approach based on geometric invariance and sliding mode ideas is proposed for redundancy resolution in robotic systems to fulfill configuration and workspace *constraints* caused by robot mechanical limits, collision avoidance, industrial security, etc. Some interesting features of the proposal are that: (1) it can be interpreted as a *limit case* of the classical potential field-based approach for collision avoidance which requires using variable structure control concepts, (2) it allows reaching the limit surface of the constraints *smoothly*, depending on a free design parameter, and (3) it can be easily added as a *supervisory* block to pre-existing redundancy resolution schemes. The algorithm is evaluated in simulation on a 6R planar robot and on the freely accessible 6R robot model PUMA-560, for which the main features of the method are illustrated.

Keywords: Sliding mode, collision avoidance, redundancy resolution

1. Introduction

The main objective of robot control systems is the tracking of a reference trajectory, which involves the generation of a control signal to make the error between the robot position and the reference zero. In the case of *redundant* robots, an additional *secondary* goal can be achieved by using the redundant degrees of freedom of the robot [1]. For instance, the redundancy can be used to avoid critical regions of the robot's configuration space (hereafter,

C-space) and/or workspace where collision might occur, the robot kinematics is singular, etc.

The framework of this research is a *redundant* robot operating in a *structured environment* [2], where the location of the robot, its operation region—additional constraints may be imposed to the original robot workspace in order to, for example, avoid the overlap with the workspace of other industrial machines placed close to the robot—and obstacles to avoid (real obstacles plus security distance) are known.

The use of *artificial potential fields* may be a way to address the problem. The potential field method has been studied extensively for autonomous mobile robot path planning in the past decades [3, 4]. The basic concept of the potential field method is to fill the robot's C-space/workspace with an artificial potential field in which the robot is attracted to its goal position and is repulsed away from the obstacles. There are many different variants based on this technique. When this technique is implemented as an on-line reactive behavior for collision avoidance [5], the robot reference velocity depends in part on the minimum distance from the robot position to the obstacles, which represent the constrained space. However, unless the field is designed to abruptly decay at a short distance of the obstacle, some regions of the C-space/workspace close to the boundaries will not be reached because of repulsion. Furthermore, collisions might occur if the robot approaches the obstacle at high speed, hence some corrective speed-related terms would be needed.

This paper proposes an alternative solution to the above problem as a supervisory block, in which the joint velocities commanded to the joint controllers may be different to those provided by the classical redundancy resolution in order to fulfill C-space and/or workspace constraints. The basic idea is to define a *discontinuous control law* inspired by the fact that, in the limit case, as the repulsion region decreases, a potential field could be characterized as a discontinuous force: zero away from the obstacle, a big value when touching its boundary.

Discontinuous control laws, as a particular case of variable-structure control strategies, have been deeply studied in the context of sliding-mode control [6, 7]. Indeed, many types of sliding-mode controllers have been developed in the last years for robotic systems [8, 9, 10, 11]. Hence, the objective of this paper is to design a *supervisory loop* [12] based on the mentioned discontinuous field idea and using sliding-mode control theory in order to modify the commanded joint velocities so that the robot fulfills the desired C-space

and/or workspace constraints. Sliding-mode-based supervisory blocks may also be used to handle joint speed limits. For instance, in [13] the authors discuss this issue for non-redundant robots.

The proposed algorithm does only activate when the robot is about to violate the constraints, modifying the commanded joint velocities as much as necessary to satisfy all the constraints. It also allows reaching the limit surfaces smoothly depending on a free design parameter. The strategy to be presented can be easily added as an auxiliary block to conventional redundancy resolution schemes.

The outline of the paper is as follows. Next section introduces some preliminaries and states the main problem to be addressed, while Section 3 presents some general concepts on geometric invariance and sliding regimes. Section 4 develops the sliding mode redundancy resolution scheme proposed to fulfill C-space and/or workspace constraints, while some important remarks about the methodology application are given in Section 5. The proposed approach is applied in Section 6 to a six-revolute (6R) planar robot and to the freely accessible 6R robot model PUMA-560, for which the main distinctive features of the method are illustrated. Finally, some concluding remarks are given.

2. Preliminaries and problem statement

Following the standard notation [14], consider a robot system with $\mathbf{q} = [q_1 \dots q_n]^{\mathrm{T}}$ being the robot *configuration* or n-dimensional joint position vector and $\mathbf{p} = [p_1 \dots p_m]^{\mathrm{T}}$ being the robot *pose* or m-dimensional workspace position vector. A robot is said to be *redundant* when the dimension m of the workspace is less than the dimension n of the C-space, i.e., $m < n$. The degree of kinematic redundancy is computed as $n - m$. For the rest of the paper it is assumed that the robot at hand is redundant.

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

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

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

The first order kinematics results in:

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

where $\mathbf{J}(\mathbf{q})$ is denoted as the $m \times n$ Jacobian matrix or simply *Jacobian* of the kinematic function. For more details see [14].

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

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

2.1. Algebraic problem of redundancy

Since the robot reference is given in the workspace and the robot control is based on controlling each joint, the inverse kinematic problem (IKP) has to be solved. The IKP at the *displacement* level is much more cumbersome as an infinite number of solutions may exist for a redundant robot [15]. For that reason, an iteratively approach at *joint rate* level is typically used [16]. Firstly, the desired workspace velocity vector $\dot{\mathbf{p}}_d$ is computed by a kinematic controller to make the tracking error zero. Secondly, the desired joint velocity vector $\dot{\mathbf{q}}_d$ has to be computed to satisfy the first order kinematic relation:

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

In general, in the case of redundant robots an infinite number of solutions for $\dot{\mathbf{q}}_d$ satisfying (4) exist. All of them can be obtained from the *singular value decomposition* (SVD) [17] of the robot Jacobian:

$$\begin{aligned} \mathbf{J} =& \mathbf{U}_{m\times m}\, \mathbf{\Sigma}_{m\times n}\, \mathbf{V}^{\mathrm{T}}_{n\times n} \\ =& [\overline{\mathbf{U}}_{m\times r} \quad \widetilde{\mathbf{U}}_{m\times m-r}] \begin{bmatrix} \overline{\mathbf{\Sigma}}_{r\times r} & \mathbf{O}_{r\times n-r} \\ \mathbf{O}_{m-r\times r} & \mathbf{O}_{m-r\times n-r} \end{bmatrix} \begin{bmatrix} \overline{\mathbf{V}}^{\mathrm{T}}_{n\times r} \\ \widetilde{\mathbf{V}}^{\mathrm{T}}_{n\times n-r} \end{bmatrix}, \end{aligned} \tag{5}$$

where the subscripts indicate the dimensions of the matrices; r is the rank of $\mathbf{J}$, i.e. the number of non-zero[1] singular values of the Jacobian; $\mathbf{O}_{i\times j}$ is the null matrix of dimensions $i \times j$; $\mathbf{\Sigma}$ is a diagonal matrix with non-negative singular values of $\mathbf{J}$ on the diagonal in decreasing order; and $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices containing the left and right singular vectors of $\mathbf{J}$, respectively.

[1]It could be defined a small *tolerance* below which singular values of $\mathbf{J}$ are treated as zero in order to avoid ill-conditioning.

The infinite number of solutions $\dot{\mathbf{q}}_d$ that minimize the square error of equation (4), i.e. $\|\dot{\mathbf{p}}_d - \mathbf{J}(\mathbf{q})\,\dot{\mathbf{q}}_d\|_2^2$, are given by:

$$\begin{aligned}\dot{\mathbf{q}}_d =& \overline{\mathbf{V}}\ \overline{\mathbf{\Sigma}}^{-1}\,\overline{\mathbf{U}}^{\mathrm{T}}\,\dot{\mathbf{p}}_d + [\widetilde{\mathbf{V}} \quad \mathbf{O}_{n\times r}]\,\mathbf{b} \\ =& \mathbf{J}^{\dagger}(\mathbf{q})\,\dot{\mathbf{p}}_d + \mathbf{B}(\mathbf{q})\,\mathbf{b}, \end{aligned} \tag{6}$$

where $\mathbf{J}^{\dagger}$ is the so-called *Moore-Penrose pseudo-inverse* of $\mathbf{J}$, $\mathbf{B}$ is an $n \times n$ matrix which first $n - r$ column vectors (i.e., $\widetilde{\mathbf{V}}$) form an orthonormal basis for the null space of $\mathbf{J}$ and $\mathbf{b}$ is an arbitrary n-dimensional column vector.

The first term in (6) represents the minimum-norm solution or *base solution*, while the second term is the *homogeneous solution* that gives rise to infinite possible solutions for $\dot{\mathbf{q}}_d$ depending on the value of vector $\mathbf{b}$. Note that the homogeneous solution vanishes (i.e. $\mathbf{B} = \mathbf{O}_{n\times n}$) when $r = n$, which is not possible for redundant robots.

If the Jacobian matrix $\mathbf{J}$ is full row rank (i.e. $r \geq m$) for a given configuration $\mathbf{q}$, the Moore-Penrose pseudo-inverse is equivalent to the so-called *right pseudo-inverse*, i.e. $\mathbf{J}^{\dagger} \equiv \mathbf{J}^{\mathrm{T}}(\mathbf{J}\,\mathbf{J}^{\mathrm{T}})^{-1}$. In such circumstances the solution $\dot{\mathbf{q}}_d$ given by (6) satisfies equation (4). Otherwise, the robot configuration $\mathbf{q}$ is said to be *singular* and the desired workspace velocity vector $\dot{\mathbf{p}}_d$ in general cannot be achieved. That is, the square error of equation (4) given by $\dot{\mathbf{q}}_d$ in (6) is non-zero unless $\dot{\mathbf{p}}_d = \mathbf{U}\,\xi$, i.e. $\dot{\mathbf{p}}_d$ lies in the column space of $\mathbf{J}$, which represents a non-ordinary singularity [18].

The expression (4) can be generalized weighting the equations by matrix a $\mathbf{W}_p$ and weighting joint speeds by the robot inertia matrix $\mathbf{W}_q$ (in order to take into account the joint power requirements), that is:

$$\begin{aligned}\mathbf{W}_p\,\dot{\mathbf{p}}_d &= (\mathbf{W}_p\,\mathbf{J}\,\mathbf{W}_q^{-1})\,\dot{\overline{\mathbf{q}}}_d \\ \dot{\overline{\mathbf{p}}}_d &= \overline{\mathbf{J}}\ \dot{\overline{\mathbf{q}}}_d. \end{aligned} \tag{7}$$

Therefore, $\dot{\overline{\mathbf{q}}}_d$ is computed from the SVD of $\overline{\mathbf{J}}$ by an expression analogous to that in (6) and then $\dot{\mathbf{q}}_d$ is computed as $\mathbf{W}_q^{-1}\,\dot{\overline{\mathbf{q}}}_d$.

One common approach in the literature to limit joint speeds is the *damped least-squares inverse* [19] which can be easily casted in the previous formu-

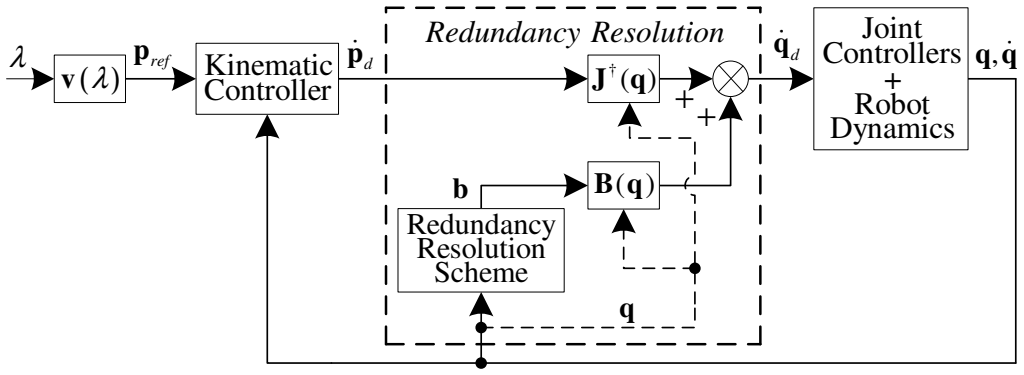

Fig. 1. Robotic trajectory tracking control scheme with redundancy resolution.

lation by augmenting (7) to include joint speeds:

$$\begin{bmatrix} \dot{\overline{\mathbf{p}}}_d \\ \mathbf{0}_n \end{bmatrix} = \begin{bmatrix} \overline{\mathbf{J}} \\ \rho\, \mathbf{I}_n \end{bmatrix} \dot{\overline{\mathbf{q}}}_d$$
$$\dot{\widetilde{\mathbf{p}}}_d = \widetilde{\mathbf{J}}\ \dot{\overline{\mathbf{q}}}_d, \tag{8}$$

where $\mathbf{0}_n$ is the n-dimensional null column vector, $\mathbf{I}_n$ is the n-dimensional identity matrix and ρ is the damping factor. Note that $\widetilde{\mathbf{J}}$ has full column rank, i.e. its null space is empty.

2.2. Trajectory tracking scheme with redundancy resolution

Fig. 1 shows the typical kinematic control scheme used for robotic trajectory tracking with the classical redundancy resolution [16]. The desired joint velocity vector $\dot{\mathbf{q}}_d$ is the command to the robot joint controllers and is obtained from the redundancy resolution block. The desired workspace velocity vector $\dot{\mathbf{p}}_d$ is the input to the redundancy resolution block and is obtained from the kinematic controller, which closes a loop using the robot state $(\mathbf{q}, \dot{\mathbf{q}})$ and the workspace reference $\mathbf{p}_{ref}$. Alternatively, the desired workspace velocity vector $\dot{\mathbf{p}}_d$ could also be supplied on-line by an operator using a joystick and visual feedback of the robot's position.

The redundancy resolution used in Fig. 1 is given by (6), which means that the robot is required to track the desired workspace velocity vector $\dot{\mathbf{p}}_d$ as primary task, while a secondary goal can be achieved by properly choosing vector $\mathbf{b}$ to be projected into robot *self-motions*, i.e. $\mathbf{b}$ changes the robot configuration but does not change the robot pose. In general, this arbitrary

vector can be expressed as a function of the robot state, i.e. $\mathbf{b}(\mathbf{q}, \dot{\mathbf{q}})$.

Assumption 1. It will be assumed the *kinematic framework*, i.e. the dynamics given by the joint controllers is negligible compared to the dynamics of the workspace reference $\mathbf{p}_{ref}$, which implies that the actual joint velocity vector $\dot{\mathbf{q}}$ is approximately equal to the desired joint velocity vector $\dot{\mathbf{q}}_d$.

To fulfill the previous assumption, the values of the motion rate parameter $\dot{\lambda}$ of the workspace reference $\mathbf{p}_{ref}$ and its derivative $\ddot{\lambda}$ have to be bounded in order to limit the radial and tangential acceleration of the workspace reference, respectively.

2.3. Redundancy resolution schemes in the literature

Several redundancy resolution schemes (RRS) have been developed in the literature to select *performance vector* $\mathbf{b}$, which can be considered as a virtual force that attempts to push the configuration of the robot away from a critical region. The most common approach is the *gradient projection* method (GPM) [1], which minimizes a configuration-dependent scalar, the performance index s, by means of its gradient[2] vector:

$$\mathbf{b} = -k_s \, \nabla s(\mathbf{q}), \tag{9}$$

where k_s is an arbitrary constant.

It is important to remark that, only the robot configuration $\mathbf{q}$ is used in the GPM, as opposed to the approach proposed in this work, in which joint speeds $\dot{\mathbf{q}}$ are also considered, see Section 4.1.

Different options have been proposed in the literature for the selection of performance index s. For instance, the weighted square distance to a *reference configuration* $\mathbf{q}_{ref}$ is used in [1] as performance index:

$$s = \frac{1}{2} \left(\mathbf{q} - \mathbf{q}_{ref}\right)^{\mathrm{T}} \mathbf{W}_s \left(\mathbf{q} - \mathbf{q}_{ref}\right), \tag{10}$$

where $\mathbf{W}_s$ is a diagonal weighting matrix. Thus, joint-limit avoidance is achieved by selecting $\mathbf{q}_{ref} = (\mathbf{q}_{max} + \mathbf{q}_{min})/2$ and $\mathbf{W}_s = \mathrm{diag}(2/(q_{i\,max,} - q_{i\,min}))$, were $\mathbf{q}_{min}$ and $\mathbf{q}_{max}$ are the lower and upper joint limits, respectively. Alternatively, collision avoidance is achieved by selecting a reference configuration $\mathbf{q}_{ref}$ away from the obstacles.

[2]The gradient of a scalar function $f(x_1, \ldots, x_n)$ is denoted ∇f where ∇ is the vector differential operator, i.e. $\nabla f = [\frac{\partial f}{\partial x_1} \cdots \frac{\partial f}{\partial x_n}]^{\mathrm{T}}$.

Similar indexes of quadratic forms in the robot configuration $\mathbf{q}$ have been considered in the literature together with the robot *manipulability* [20] or the *condition number* of the Jacobian matrix [21] in order to avoid singular configurations. Furthermore, a performance index based on *artificial potential fields* [5] could also be considered to keep the robot away from the obstacles.

2.4. Problem statement

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

$$\Phi_{CS}(\mathbf{q}) = \{\mathbf{q} \,|\, \sigma_i(\mathbf{q}) \leq 0\}\,,\ i = 1, ..., N, \tag{11}$$

where σ_i is a function of the robot configuration[3] $\mathbf{q}$ that is positive if and only if the ith-constraint is not fulfilled. Note that, $\sigma_i(\mathbf{q}) = 0$ represents the boundary of the ith-constraint. For instance, a constraint $\sigma_{plane} = \mathbf{n}_{plane}^{\mathrm{T}}(\mathbf{q} - \mathbf{q}_{plane}) \leq 0$ would indicate that the boundary of the C-space is a plane with normal vector $\mathbf{n}_{plane}$ and passing through point $\mathbf{q}_{plane}$.

In order for some smoothness assumptions to hold in the solution later proposed in this work, the functions σ_i need to be twice differentiable around the boundary given by $\sigma_i(\mathbf{q}) = 0$ and their gradients $\nabla\sigma_i$ around this boundary should not vanish. For non-differentiable constraints, there are techniques in literature [22] that may be used to enclose such non-smooth regions by smooth mathematical objects with an arbitrary degree of precision.

The main control goal can therefore be stated as to generate a joint velocity vector $\dot{\mathbf{q}}_d$ to be sent to the robot joint controllers so that the desired workspace velocity vector $\dot{\mathbf{p}}_d$ is tracked using the non-redundant degrees of freedom of the robot, while the remaining redundant degrees of freedom are used to implement a classical RRS together with a supervisory block to guarantee that $\mathbf{q}$ belongs to the allowed C-space Φ_{CS} given by (11).

3. Development of the theoretical framework

In order to address the above control problem from a general framework, we present in this section some important concepts on geometric invariance

[3]The constraints could also be defined in terms of any other vector related to the robot configuration, e.g. the Cartesian position $\bar{\mathbf{p}}_j = [x_j \ \ y_j \ \ z_j]^{\mathrm{T}}$ of a point j of the robot, i.e. $\sigma_i(\bar{\mathbf{p}}_j) = \sigma_i'(\mathbf{q})$. Also, the approaching speed to those constraints can be incorporated in the framework, see Section 4.1.

and sliding regimes. In particular, we study the necessary conditions to confine a dynamical system to an invariant region of the state-space, and we then explore the relationship between these conditions and the resulting dynamics of a system operating in sliding mode (SM), i.e. when the system input consists of a high frequency discontinuous signal.

3.1. Geometric invariance

Consider the following dynamical system

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{d}) + \mathbf{g}(\mathbf{x})\,\mathbf{u} \\ \mathbf{y} = \mathbf{h}(\mathbf{x}), \end{cases} \tag{12}$$

where $\mathbf{x} \in X \subset \mathbb{R}^{n'}$ is the state vector, $\mathbf{d} \in D \subset \mathbb{R}^{n'}$ an unmeasured disturbance or model uncertainty, $\mathbf{u} \in U \subset \mathbb{R}^{m'}$ the control input vector (possibly discontinuous), $\mathbf{f} : \mathbb{R}^{2n'} \to \mathbb{R}^{n'}$ a vector field defined in $X \bigcup D$, $\mathbf{g} : \mathbb{R}^{n'} \to \mathbb{R}^{n' \times m'}$ a set of m' vector fields defined in X, and $\mathbf{h} : \mathbb{R}^{n'} \to \mathbb{R}^{b}$ a vector field defined in X.

The variable $\mathbf{y}$ denotes the system output vector, which has to be bounded so as to fulfill user-specified constraints. The corresponding bounds on $\mathbf{y}$ are given by the set:

$$\Phi(\mathbf{x}) = \{\mathbf{x} \mid \phi_i(\mathbf{y}) \leq 0\}, \quad i = 1, ..., N. \tag{13}$$

Since the set Φ specifies the region of the state space compatible with the bounds on output $\mathbf{y}$, 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 $\mathbf{y}$ is driven as close as possible to its desired value $\mathbf{y}_{ref}$.

To ensure the invariance of Φ, the control input $\mathbf{u}$ must guarantee that the right hand side of the first equation in (12) points to the interior of Φ at all points in the boundary of Φ, denoted by $\partial\Phi$, defined as:

$$\partial\Phi = \bigcup_{i=1}^{N} \partial\Phi_i, \quad \partial\Phi_i = \{\mathbf{x} \in \Phi \mid \phi_i(\mathbf{y}) = 0\}. \tag{14}$$

For later developments, the following assumption will be needed.

Assumption 2. It will be assumed that all ϕ_i functions are differentiable in the boundary $\partial\Phi_i$.

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

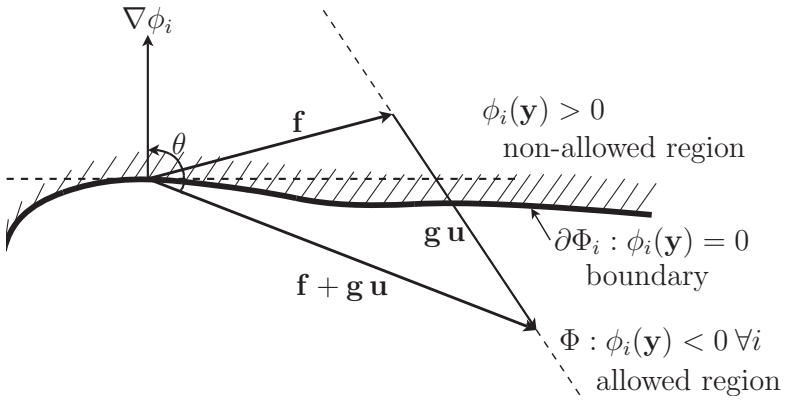

Fig. 2. Geometrical interpretation of invariance condition $cos(\theta) \leq 0$.

Mathematically, the invariance of Φ is ensured by an input such that, for all i, $\dot{\phi}_i \leq 0$ when $\phi_i(\mathbf{y}) = 0$, i.e., a condition on the dot product between the gradient of an active constraint with respect to the state and the time derivative of the state:

$$\begin{aligned}\dot{\phi}_i(\mathbf{x}, \mathbf{d}, \mathbf{u}) &= \nabla\phi_i^{\mathrm{T}}\dot{\mathbf{x}} = \|\nabla\phi_i\|\,\|\mathbf{f} + \mathbf{g}\,\mathbf{u}\|\,cos\theta \\ &= \nabla\phi_i^{\mathrm{T}}\mathbf{f} + \nabla\phi_i^{\mathrm{T}}\mathbf{g}\,\mathbf{u} \\ &= L_f\phi_i + \mathbf{L_g}\phi_i\mathbf{u} \leq 0, \qquad \forall\mathbf{x} \in \partial\Phi_i, \quad i = 1, ..., N, \end{aligned} \tag{15}$$

where $L_f\phi_i$ and $\mathbf{L_g}\phi_i$ denote the Lie derivatives of $\phi_i(\mathbf{y})$ in the direction of vector field $\mathbf{f}(\mathbf{x}, \mathbf{d})$ and in the direction of the set of vector fields $\mathbf{g}(\mathbf{x})$, respectively. Note that, $\mathbf{L_g}\phi_i$ is an m'-dimensional row vector.

The previous condition is geometrically depicted in Fig. 2 for $n' = 2$, and may be written in standard form as:

$$\inf_{\mathbf{u}}\{\dot{\phi}_i(\mathbf{x}, \mathbf{d}, \mathbf{u}) \leq 0, \;\; \forall\mathbf{x} \in \partial\Phi_i\}, \quad i = 1, ..., N, \tag{16}$$

which is known as *implicit invariance condition* [23].

Solving (15) for $\mathbf{u}$ gives rise to the *explicit invariance condition* for system (12) and a particular constraint ϕ_i. The set of feasible solutions of (15),

will be:

$$\mathcal{U}_i(\mathbf{x},\mathbf{d}) = \begin{cases} \mathbf{u} \in \{U | L_f\phi_i + \mathbf{L_g}\phi_i\mathbf{u} \leq 0\} \\ \qquad\qquad : \mathbf{x} \in \partial\Phi_i \text{ and } \mathbf{L_g}\phi_i \neq \mathbf{0}_{m'}^{\mathrm{T}} \\ \text{empty} \quad : \mathbf{x} \in \partial\Phi_i \text{ and } \mathbf{L_g}\phi_i = \mathbf{0}_{m'}^{\mathrm{T}} \text{ and } L_f\phi_i > 0 \\ \mathbf{u} = \text{free} \quad : \mathbf{x} \in \partial\Phi_i \text{ and } \mathbf{L_g}\phi_i = \mathbf{0}_{m'}^{\mathrm{T}} \text{ and } L_f\phi_i \leq 0 \\ \mathbf{u} = \text{free} \quad : \mathbf{x} \in \Phi \diagdown \partial\Phi_i, \end{cases} \tag{17}$$

where $\mathbf{0}_{m'}$ denotes the m'-dimensional null column vector and, evidently, the first set corresponding to $\mathbf{L_g}\phi_i \neq \mathbf{0}_{m'}^{\mathrm{T}}$ is always non-empty[4].

Note that, the control $\mathbf{u}$ in the interior of Φ can be freely assigned. Particularly, $\mathbf{u} = \mathbf{0}_{m'}$ could be taken so that the system evolves autonomously throughout the interior of Φ. Then, the control action becomes active only when some constraint becomes active, i.e. when the state trajectory reaches the boundary $\partial\Phi$ trying to leave the set Φ. Then, the invariance condition will hold if the intersection $\bigcap_i \mathcal{U}_i(\mathbf{x})$ for all constraints of the solution sets $\mathcal{U}_i(\mathbf{x})$ is not empty.

3.2. Geometric invariance via sliding regimes

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

$$\mathbf{u} = \begin{cases} \mathbf{0}_{m'} & \text{if} \quad \max_i \{\phi_i(\mathbf{y})\} < 0 \\ \mathbf{u}_{SM} & \text{otherwise,} \end{cases} \tag{18}$$

where $\mathbf{u}_{SM}$ is arbitrarily chosen from $\bigcap_i \mathcal{U}_i(\mathbf{x})$, if non-empty.

The above control law leads to a sliding regime [6] (i.e., control signal $\mathbf{u}$ switches between $\mathbf{0}_{m'}$ and $\mathbf{u}_{SM}$ with a theoretically infinite frequency) on the boundary of the constraint set, which can be described by $\{\mathbf{x} \,|\, \max_i \{\phi_i(\mathbf{y})\} = 0\}$.

[4]If a control action $\mathbf{u} \in \{U | L_f\phi_i + \mathbf{L_g}\phi_i\mathbf{u} \leq -\gamma\}$ for some arbitrarily chosen positive constant γ were applied when the state is *strictly outside* Φ, finite-time convergence to Φ can be achieved [7]. This is helpful when dealing with initial conditions out of the desired workspace.

3.2.1. Single active constraint

If the constraint Φ_i is considered, then a sliding regime around the boundary $\mathbf{x} \in \partial\Phi_i$ will establish if the following inequalities are locally satisfied around $\partial\Phi_i$:

$$\dot{\phi}_i = \begin{cases} L_f\phi_i > 0 & \text{if} \quad \phi_i(\mathbf{y}) < 0 \\ L_f\phi_i + \mathbf{L_g}\phi_i\mathbf{u}_{SM} < 0 & \text{otherwise.} \end{cases} \tag{19}$$

On the one hand, the first inequality in (19) is locally satisfied whenever the system tries by itself to leave the set Φ. Thus, the switching law (18) does not seek for sliding mode (SM), but it becomes active if the process is at the boundary of the allowed region and about to leave it. On the other hand, the second inequality of (19) implies that for a sliding regime to be established on the boundary $\partial\Phi_i$,

$$\mathbf{L_g}\phi_i = \nabla\phi_i^{\mathrm{T}}\,\mathbf{g} \neq \mathbf{0}_{m'}^{\mathrm{T}}, \tag{20}$$

must hold locally on the manifold $\phi_i(\mathbf{y}) = 0$. The necessary condition (20) for SM, which is known as the *transversality condition* [24], imposes that the sliding manifold must have *unitary relative degree* with respect to the discontinuous action, i.e., its first-order time derivative ($\dot{\phi}_i$) must explicitly depend on $\mathbf{u}$. If this is not the case, an auxiliary subset $\Phi^{\diamond} \subset \Phi$ which satisfies this condition should be properly defined [24].

For only one active constraint ϕ_i and $\mathbf{L_g}\phi_i \neq \mathbf{0}_{m'}^{\mathrm{T}}$, the minimum-Euclidean-norm control action $\mathbf{u}_{SM}$ in $\mathcal{U}_i(\mathbf{x})$ is given by a vector parallel to $\mathbf{L_g}\phi_i^{\mathrm{T}}$:

$$\mathbf{u}_{SM\,min\,i} = -\mathbf{L_g}\phi_i^{\mathrm{T}}\,\frac{L_f\phi_i}{\mathbf{L_g}\phi_i\,\mathbf{L_g}\phi_i^{\mathrm{T}}}\,. \tag{21}$$

Therefore, for only one active constraint it is proposed to use the following expression for the control action $\mathbf{u}_{SM}$:

$$\mathbf{u}_{SM} = -\mathbf{L_g}\phi_i^{\mathrm{T}}\,u^{+}, \tag{22}$$

where u^{+} is a positive constant to be chosen high enough to establish a SM on the boundary $\partial\Phi_i$, i.e., the second inequality in (19) must hold whenever the first inequality in (19) is locally satisfied. To fulfill that, the scalar factor

u^+ must be:

$$u^+ > \frac{\max(L_f\phi_i, 0)}{\mathbf{L_g}\phi_i\,\mathbf{L_g}\phi_i^{\mathrm{T}}} = u^{\phi_i}. \tag{23}$$

Once the switching SM is established on the boundary $\partial\Phi_i$ by the control action $\mathbf{u}_{SM}$ in (22), the continuous equivalent control [6] is obtained as $\mathbf{u}_{eq} = \mathbf{u}_{SM\,min\,i}$, which according to (17) is the control required to keep the system just on the boundary manifold $\partial\Phi_i$. Consequently, the sliding regime generated by switching law (18) produces the minimal value u^{ϕ_i} (without explicit knowledge of it) for the continuous equivalent of u^+ in order to achieve the invariant condition $\dot{\phi}_i \leq 0$. Moreover, the necessary condition (20) for SM on boundary $\partial\Phi_i$ guarantees that the invariant control exists in (17) for ith-constraint.

3.2.2. Multiple active constraints

In case several constraints (say h constraints) are active, a function vector $\boldsymbol{\phi}$ and its time derivative $\dot{\boldsymbol{\phi}}$ composed of all active constraints should be considered in (15), so the geometric invariance constraints can be kept holding if the linear system of inequalities:

$$L_f\boldsymbol{\phi} + \mathbf{L_g}\boldsymbol{\phi}\,\mathbf{u} \leq \mathbf{0}_h, \tag{24}$$

which actually describes $\bigcap_i \mathcal{U}_i(\mathbf{x})$ for i indexing *only the h active constraints*, admits a solution. Indeed, the non-active constraints admit a free $\mathbf{u}$, hence only the active ones should be considered in the control law computation. Of course, such set of active constraints changes with time.

In the spirit of the discussion for the above one-constraint case, one of such control laws could be given by the solution to:

$$\mathbf{u}_{SM} = \arg\min_{\widetilde{\mathbf{u}}} \left(\widetilde{\mathbf{u}}^{\mathrm{T}}\widetilde{\mathbf{u}} \mid \mathbf{L_g}\boldsymbol{\phi}\,\widetilde{\mathbf{u}} \leq -\mathbf{1}_h\,u^+\right), \tag{25}$$

where $\mathbf{1}_h$ is the h-dimensional column vector with all its components equal to one and u^+ is again a positive constant to be chosen high enough to establish a SM on the boundary $\partial\Phi$. In particular, the proposition below provides a lower bound for u^+ in (25).

Proposition 1. *One set of sufficient conditions for a "collective" sliding mode to arise in the intersection of h active constraints whose derivative*

function vector is given by:

$$\dot{\boldsymbol{\phi}} = L_f \boldsymbol{\phi} - \mathbf{M}\,\mathbf{z}\,u^+, \tag{26}$$

where $z_i = 1$ *if* $\phi_i > 0$ *and* $z_i = 0$ *if* $\phi_i < 0$*, is that matrix* $\mathbf{M}$ *is positive definite and*

$$u^+ > \sum_{i=1}^{h} \left(\max(L_f \phi_i, 0)\right) \Bigg/ \text{eig}_{min}\left(\frac{\mathbf{M} + \mathbf{M}^T}{2}\right), \tag{27}$$

PROOF. Let $V = \mathbf{z}^{\mathrm{T}}\,\boldsymbol{\phi}$ be a Lyapunov function candidate. Partition vector $\boldsymbol{\phi}$ into two subvectors $\boldsymbol{\phi} = [\boldsymbol{\phi}^{a\,\mathrm{T}} \quad \boldsymbol{\phi}^{h-a\,\mathrm{T}}]^{\mathrm{T}}$, assuming that SM occurs in the manifold given by $\boldsymbol{\phi}^a = \mathbf{0}_a$, whereas the components of vector $\boldsymbol{\phi}^{h-a}$ are greater than zero.

According to the continuous equivalent control [6], 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 results in:

$$\begin{aligned}\dot{V} =& \frac{d}{dt}\left(\mathbf{z}^{\mathrm{T}}\,\boldsymbol{\phi}\right) = \frac{d}{dt}\left(\begin{bmatrix} \mathbf{z}^a_{eq} \\ \mathbf{1}_{h-a} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \boldsymbol{\phi}^a \\ \boldsymbol{\phi}^{h-a} \end{bmatrix}\right) \\ &= \begin{bmatrix} \dot{\mathbf{z}}^a_{eq} \\ \mathbf{0}_{h-a} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \mathbf{0}_a \\ \boldsymbol{\phi}^{h-a} \end{bmatrix} + \mathbf{z}^{\mathrm{T}}\,\dot{\boldsymbol{\phi}} = \mathbf{z}^{\mathrm{T}}\,\dot{\boldsymbol{\phi}}. \end{aligned} \tag{28}$$

Replacing vector $\dot{\boldsymbol{\phi}}$ with its value from (26), it is obtained:

$$\dot{V} = \mathbf{z}^{\mathrm{T}}\,L_f\boldsymbol{\phi} - \mathbf{z}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{z}\,u^+. \tag{29}$$

The components of vector $\mathbf{z}$ range from 0 to 1, hence the upper bound of the first term in (29) is given by $z_i = 1$ if $L_f\phi_i > 0$ and $z_i = 0$ if $L_f\phi_i < 0$, that is:

$$\mathbf{z}^{\mathrm{T}}\,L_f\boldsymbol{\phi} \leq \sum_{i=1}^{h} \left(\max(L_f\phi_i, 0)\right) \tag{30}$$

Assuming that $u^+ > 0$, the second term in (29) is negative if matrix $\mathbf{M}$ is positive definite, in which case the upper bound of this term is given by:

$$-\mathbf{z}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{z}\,u^+ = -\,\mathbf{z}^{\mathrm{T}}\,\frac{\mathbf{M}+\mathbf{M}^{\mathrm{T}}}{2}\,\mathbf{z}\,u^+ \leq -\text{eig}_{min}\left(\frac{\mathbf{M}+\mathbf{M}^{\mathrm{T}}}{2}\right)\,\|\mathbf{z}\|_2^2\;u^+, \tag{31}$$

where

$$\|\mathbf{z}\|_2 \geq 1 \quad \forall\, \boldsymbol{\phi} \neq \mathbf{0}_h, \tag{32}$$

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

From (30), (31) and (32), the upper bound of the time derivative of the Lyapunov function V results in:

$$\dot{V} \leq \sum_{i=1}^{h} \left(\max(L_f \phi_i, 0)\right) - \text{eig}_{min} \left(\frac{\mathbf{M} + \mathbf{M}^{\mathrm{T}}}{2} \right) u^+. \tag{33}$$

Therefore, if u^+ fulfills (27) the Lyapunov function decays at a finite rate, it vanishes and collective SM in the intersection of the h active constraints occurs after a finite time interval. That is, the origin $\boldsymbol{\phi} = \mathbf{0}_h$ is an asymptotically stable equilibrium point with finite time convergence. □

According to Proposition 1, the scalar factor u^+ in (25) must be:

$$u^+ > \sum_{i=1}^{h} \left(\max(L_f \phi_i, 0)\right) = u^{\boldsymbol{\phi}}, \tag{34}$$

to satisfy sufficient conditions for collective sliding mode.

Computing solution (25) on-line might be impractical so, trying to solve the Least-Squares problem $\mathbf{L_g}\boldsymbol{\phi}\mathbf{u} = -\mathbf{1}_h\, u^+$ via the Moore-Penrose pseudo-inverse may be a more reasonable solution for implementation at a fast-enough rate, giving:

$$\mathbf{u}_{SM} = -\mathbf{L_g}\boldsymbol{\phi}^{\dagger}\, \mathbf{1}_h\, u^+, \tag{35}$$

which requires SVD computations (see Section 2.1) and where u^+ is a high-enough scalar fulfilling (34) so if $\mathbf{L_g}\boldsymbol{\phi}$ is full row rank, sufficient conditions for collective sliding mode are satisfied.

In this case, (20) must be now changed to the "multiple-constraint" transversality condition of $\mathbf{L_g}\boldsymbol{\phi}$ having *full row rank*, which indeed covers the previous single-constraint case (a one-row matrix is full rank if there exists a non-zero element).

A third option can be considered which does not involve computing any inverse, SVD or solution to linear equations, even if there would be no major

problem in doing that with moderately-sized problems in modern computers. Such option consists in generalizing (22) to:

$$\mathbf{u}_{SM} = -\mathbf{L_g}\boldsymbol{\phi}^{\mathrm{T}}\,\mathbf{1}_h\,u^{+}, \tag{36}$$

where:

$$u^{+} > \sum_{i=1}^{h}\left(\max(L_f\phi_i, 0)\right) \Big/ \mathrm{eig}_{min}(\mathbf{L_g}\boldsymbol{\phi}\,\mathbf{L_g}\boldsymbol{\phi}^{\mathrm{T}}) = u^{\boldsymbol{\phi}}, \tag{37}$$

i.e., its value depends on the inverse of the minimum singular value of $\mathbf{L_g}\boldsymbol{\phi}$. As before, if the multiple-constraint transversality condition is fulfilled, sufficient conditions for collective sliding mode are satisfied.

Note that in numerical implementations, computation of $\mathbf{u}_{SM}$ should be implemented at a fast enough frequency to approximate the ideal continuous-time behavior [7] and the active constraints will be all those in which $\phi_i(\mathbf{y}) > 0$ (discrete-time sliding modes are not attained at $\phi_i(\mathbf{y}) = 0$ but only approximately, in a band around zero whose width depends on sampling period, see [7]). Indeed, the column vector $\mathbf{1}_h$ in (25), (35) and (36) could be replaced by a weighted unit vector whose ith-component is a function of ϕ_i. Thus, the unfulfilled constraints with higher value ϕ_i have greater impact on control action $\mathbf{u}_{SM}$ and, therefore, the minimum value of u^{+} required for SM may be reduced.

As in the previous single-constraint case, once the switching SM is established on the boundary $\partial\Phi$ by the control action $\mathbf{u}_{SM}$, a continuous equivalent control [6] is obtained which is the control required to keep the system just on the boundary manifold $\partial\Phi$. Consequently, the sliding regime generated by switching law (18) produces such equivalent control without explicit knowledge of it, with a reasonably low computational cost; this is a distinctive advantage of sliding-mode strategies.

4. The Proposal

4.1. Redundancy resolution scheme

We are interested in exploiting the exact approximation to the constraint boundary that allow variable structure laws such as (18) to deal with the control problem stated in Section 2. In particular, we will employ the ideas of Section 3 to perform an on-line robotic redundancy resolution so that C-space limits given by equation (11) are fulfilled. The objective of this

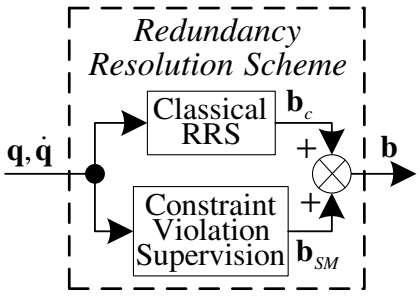

Fig. 3. Redundancy resolution scheme of the proposal.

redundancy resolution is to instantaneously modify the desired joint velocity vector $\dot{\mathbf{q}}_d$ which is sent to the joint controllers of the robot when there is a risk of violating a given constraint.

In particular, the RRS proposed in this research, see Fig. 3, consists of the combination of two signals:

$$\mathbf{b} = \mathbf{b}_c + \mathbf{b}_{SM}, \tag{38}$$

where $\mathbf{b}_c$ is the performance vector of a classical RRS, see Section 2.3, and $\mathbf{b}_{SM}$ is a discontinuous signal generated by a supervisor block proposed in this work to fulfill C-space constraints. It is important to remark that, in contrast to classical RRS, the mentioned supervisor block takes into account not only joint positions but also joint speeds, as discussed below.

At this point it is important to consider the following rationale. Approaching the constraints at high speed is impractical because collisions might occur if the *joint accelerations* $\ddot{\mathbf{q}}$ required to slow down the motion of the robot towards the constraint boundary cannot be achieved by the robot actuators due to power limitations [25]. This is of particular significance in mechanical or robotics systems in which the inertia is large. Hence, the actual constraint space (11) will be modified to also include the speed of movement in the following way:

$$\Phi^*_{CS}(\mathbf{q}, \dot{\mathbf{q}}) = \Big\{ [\mathbf{q}^{\mathrm{T}}\ \dot{\mathbf{q}}^{\mathrm{T}}]^{\mathrm{T}} \mid \phi_i(\mathbf{q}, \dot{\mathbf{q}}) = \sigma_i(\mathbf{q}) + K\frac{d\sigma_i(\mathbf{q})}{dt} = \sigma_i + K\ \nabla\sigma_i^{\mathrm{T}}\ \dot{\mathbf{q}} \leq 0 \Big\},\ i = 1, ..., N, \tag{39}$$

where $\sigma_i(\mathbf{p})$ is the original ith workspace constraint and K is the *constraint*

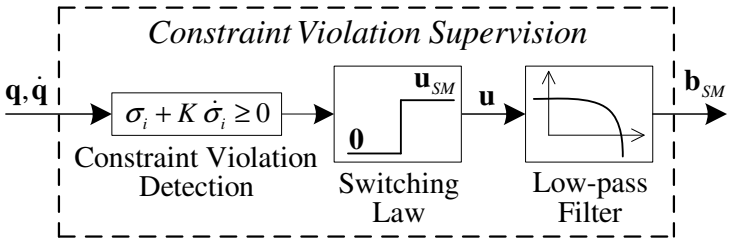

Fig. 4. Supervisor block proposed to fulfill C-space constraints.

approaching parameter, which is a free design parameter that determines the rate of approach to the boundary of the constraints. Thus, expression (39) introduces an additional degree of freedom necessary to reach the limit in a controlled fashion. That is, the term $K\dot{\sigma}_i$ is used to anticipate when the robot is about to violate the ith-constraint in order to initiate an early corrective action, which will increase the numerical stability of the redundancy resolution algorithm. Note that $\dot{\sigma}_i = \nabla\sigma_i^{\mathrm{T}}\,\dot{\mathbf{q}}$. Thus, for low speeds or small K values $\Phi_{CS}^{*} \approx \Phi_{CS}$ given by (11). Note also that K may take different values for different constraints, if so wished.

4.2. Sliding-mode supervisor to fulfill C-space constraints

In order to apply the theoretical framework of the previous section a dynamic system will be constructed whose *state* is $\mathbf{x} = [\,\mathbf{q}^{\mathrm{T}} \quad \mathbf{b}_{SM}^{\mathrm{T}}\,]^{\mathrm{T}}$; its *output* vector is $\mathbf{y} = [\,\mathbf{q}^{\mathrm{T}} \quad \dot{\mathbf{q}}^{\mathrm{T}}\,]^{\mathrm{T}}$; its *disturbance* input are the performance vector $\mathbf{b}_c$ of a classical RRS and the desired workspace velocity vector $\dot{\mathbf{p}}_d$ (which tends to the time derivative of the workspace reference $\dot{\mathbf{p}}_{ref}$ as the tracking error tends to zero); and a control *input* vector $\mathbf{u}$ will also be crafted in such a way that $\dot{\phi}_i$ directly depends on $\mathbf{u}$ (transversality condition), i.e., the second derivative of $\mathbf{q}$ should explicitly depend on the input. Note that the dimensions of this dynamic system are $n' = b = 2n$ and $m' = n$.

To take advantage of the SM features described above, the supervisor block of Fig. 4 is proposed. The signal $\mathbf{b}_{SM}$ is generated by passing the discontinuous signal $\mathbf{u}$ through a low-pass filter. This filter must be of first-order for $\ddot{\mathbf{q}}_d$, see (6) and (38), to explicitly depend on $\mathbf{u}$:

$$\dot{\mathbf{b}}_{SM} = -\alpha\,\mathbf{b}_{SM} + \alpha\,\mathbf{u}, \tag{40}$$

with the scalar α being the filter cutoff frequency. Naturally, α should be taken for the filter to be faster than the dynamics of the desired workspace

velocity vector $\dot{\mathbf{p}}_d$, in order to avoid degrading the tracking performance.

The dynamic system described above is given by equations (6) and (40), which yield, by the kinematic framework assumption $\dot{\mathbf{q}} \approx \dot{\mathbf{q}}_d$ made in Section 2.2, the following state space representation:

$$\begin{aligned}\dot{\mathbf{x}} &= \begin{bmatrix}\mathbf{O}_n & \mathbf{B} \\ \mathbf{O}_n & -\alpha\,\mathbf{I}_n\end{bmatrix}\mathbf{x} + \begin{bmatrix}\mathbf{J}^\dagger\,\dot{\mathbf{p}}_d + \mathbf{B}\,\mathbf{b}_c \\ \mathbf{0}_n\end{bmatrix} + \begin{bmatrix}\mathbf{O}_n \\ \alpha\,\mathbf{I}_n\end{bmatrix}\mathbf{u} \\ \mathbf{y} &= \begin{bmatrix}\mathbf{I}_n & \mathbf{O}_n \\ \mathbf{O}_n & \mathbf{B}\end{bmatrix}\mathbf{x} + \begin{bmatrix}\mathbf{0}_n \\ \mathbf{J}^\dagger\,\dot{\mathbf{p}}_d + \mathbf{B}\,\mathbf{b}_c\end{bmatrix},\end{aligned} \tag{41}$$

where $\mathbf{O}_n$ and $\mathbf{I}_n$ denote the null matrix and the identity matrix of dimensions $n \times n$.

For the proposed approach, the variable structure control law in (18) is considered.

Using the above state space representation (41), the Lie derivatives of ϕ_i are given by:

$$\begin{aligned}\mathbf{L}_\mathbf{g}\phi_i =& \nabla\phi_i^{\mathrm{T}}\,\mathbf{g} = \begin{bmatrix}\nabla\sigma_i^{\mathrm{T}} + K\,\dot{\mathbf{q}}^{\mathrm{T}}\,\mathbf{H}_i & K\,\nabla\sigma_i^{\mathrm{T}}\,\mathbf{B}\end{bmatrix}\begin{bmatrix}\mathbf{O}_n \\ \alpha\,\mathbf{I}_n\end{bmatrix} \\ =& \alpha\,K\,\nabla\sigma_i^{\mathrm{T}}\,\mathbf{B}\end{aligned} \tag{42}$$

$$\begin{aligned}L_f\phi_i =& \nabla\phi_i^{\mathrm{T}}\,\mathbf{f} = \begin{bmatrix}\nabla\sigma_i^{\mathrm{T}} + K\,\dot{\mathbf{q}}^{\mathrm{T}}\,\mathbf{H}_i & K\,\nabla\sigma_i^{\mathrm{T}}\,\mathbf{B}\end{bmatrix}\begin{bmatrix}\dot{\mathbf{q}} \\ -\alpha\,\mathbf{b}_{SM}\end{bmatrix} \\ =& (1-\alpha\,K)\,\nabla\sigma_i^{\mathrm{T}}\,\dot{\mathbf{q}} + K\,\dot{\mathbf{q}}^{\mathrm{T}}\,\mathbf{H}_i\,\dot{\mathbf{q}} + \alpha\,K\,\nabla\sigma_i^{\mathrm{T}}\,(\mathbf{J}^\dagger\,\dot{\mathbf{p}}_d + \mathbf{B}\,\mathbf{b}_c),\end{aligned} \tag{43}$$

where $\mathbf{H}_i$ denotes the Hessian matrix of second-order partial derivatives of σ_i. Note that the last term of $L_f\phi_i$ depends on the disturbance $\dot{\mathbf{p}}_d$ and $\mathbf{b}_c$.

From the above definitions and assumptions, the values of α, K, $\nabla\sigma_i$ and $\mathbf{B}$ are not zero[5] and, thus, the transversality condition (20) for only one active constraint is written as:

$$\nabla\sigma_i^{\mathrm{T}}\,\mathbf{B} \neq \mathbf{0}_n^{\mathrm{T}}, \tag{44}$$

which means that the projection of gradient $\nabla\sigma_i$ onto the null space of $\mathbf{J}$ must

[5]If the chosen value of K had been zero, i.e., no speed limitations were desired, then the low-pass filter should have been removed, and the state $\mathbf{x}$ would only include $\mathbf{q}$, in order to fulfill (20). Details omitted for brevity.

be non-zero, i.e. the null space of $\mathbf{J}$ must not be tangent to the boundary of the ith-constraint.

For multiple active constraints, the control action $\mathbf{u}_{SM}$ is computed using (25) or (35) or (36) depending on the computation time requirements (see Section 3.2) with $\mathbf{L_g}\boldsymbol{\phi} = \alpha K \nabla\boldsymbol{\sigma}^{\mathrm{T}}\mathbf{B}$, where $\nabla\boldsymbol{\sigma}$ contains the gradient vectors $\nabla\sigma_i$ of all active constraints. The multiple-constraint transversality condition is that matrix $\nabla\boldsymbol{\sigma}^{\mathrm{T}}\mathbf{B}$ has to be full row rank; in practice, to avoid numerical ill-conditioning, this condition is satisfied when the smallest singular value of this matrix is larger than a predetermined small threshold.

Note that although equations (34), (37) and (42)–(43) propose a lower bound for u^+ to be used in (25), (35) and (36), the selection of this scalar factor can be made in a simple manner by choosing a high-enough constant. In this way, fast computation can be achieved. This is a distinctive advantage of sliding-mode algorithms [7], as discussed below.

Indeed, from the potential field approach point of view, the value u^+ is interpreted as the repulsion of the discontinuous field. This SM approach has the advantage that the magnitude $\|\mathbf{b}_{SM}\|_2$ of the "repulsive force" (correcting action) required to avoid the constrained space given by Φ^*_{CS} (39) is robustly *auto-regulated* to the continuous equivalent control [6]. This magnitude could also be computed at each sample analytically using (43), but it would require much more computational power than, plainly, setting the scalar u^+ to a big number which, due to the equivalent-control principle, computes the required quantity by a high-frequency switching law without explicit knowledge of the Hessian matrices $\mathbf{H}_i$, the joint velocity vector $\dot{\mathbf{q}}$, the desired workspace velocity vector $\dot{\mathbf{p}}_d$, the performance vector $\mathbf{b}_c$, etc.

4.3. Minimum-amplitude auto-regulation

The amplitude of the control action $\mathbf{u}_{SM}$ of the constraint supervision algorithm in Fig. 4 is given by u^+. Although the value of u^+ can be chosen in a conservative manner setting it to a big number, as discussed above, when the sampling time T_s of the robotic system at hand is not small enough, it is advisable to use the minimum possible value of u^+ in order to minimize the "chattering" effects due to the time-discretization. In this sense, the minimum value of u^+ required for SM, i.e. $u^{\boldsymbol{\phi}}$, could be reduced on-line in order to use a small value of u^+ without loosing the sliding regime. To achieve this goal, the values of $\dot{\mathbf{p}}_d$ (desired workspace velocity vector) and $\mathbf{b}_c$ (performance vector of a classical RRS), which are responsible for the

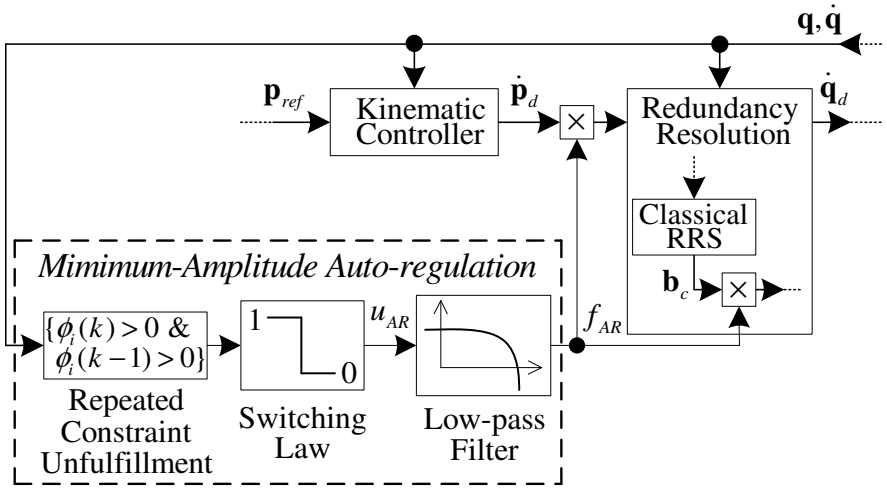

Fig. 5. Supervisor block proposed for minimum-amplitude auto-regulation.

necessity of control action $\mathbf{u}_{SM}$, have to be reduced whenever the SM of the constraint supervision algorithm is not working properly.

In numerical implementations, the SM is not working properly when some constraint is repeatedly unfulfilled in consecutive time steps (i.e. for active constraints the sign of ϕ_i must immediately switch from positive to negative), which means that the term $\mathbf{L_g}\boldsymbol{\phi}\ \mathbf{u}$ is not dominating over the term $L_f\boldsymbol{\phi}$ in (24).

Therefore, it is proposed to use the *minimum-amplitude auto-regulation* shown in Fig. 5 in order for $u^{\boldsymbol{\phi}}$ to be less than the programmed value of u^+. This algorithm is also based on a switching law and SM theory. The discontinuous signal u_{AR} is equal to 0 at sample time k if some constraint is repeatedly unfulfilled, i.e. $\exists\ i \mid \phi_i(k) > 0$ and $\phi_i(k-1) > 0$, and equal to 1 otherwise. The signal f_{AR} is generated by passing discontinuous signal u_{AR} through a low-pass filter. This filter must be of first-order to have a unitary relative degree between ϕ_i and the discontinuous action u_{AR}, as required by SM theory [24] (see Section 3.2). Finally, the signal f_{AR} acts as a "scale factor" to $\dot{\mathbf{p}}_d$ and $\mathbf{b}_c$ in order to reduced $u^{\boldsymbol{\phi}}$.

If the workspace reference $\mathbf{p}_{ref}(\lambda)$ is known, instead of scaling $\dot{\mathbf{p}}_d$ (which is the output of the kinematic controller), it could be scaled the motion rate parameter $\dot{\lambda}$ (which acts as an input to the kinematic controller) in order to directly stop the reference and, thus, avoid tracking errors. See Section 6.1 for an example.

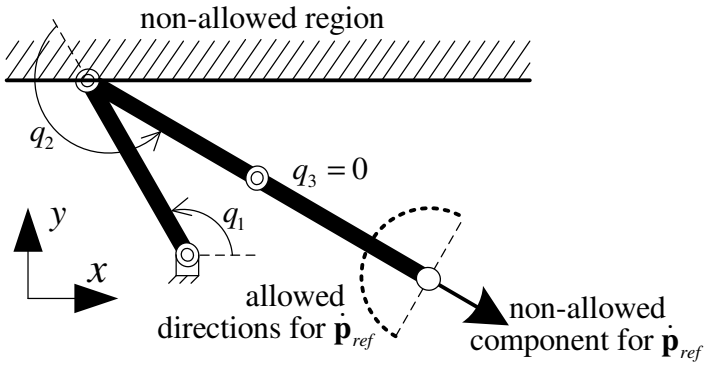

Fig. 6. Example of blocking situation for a 3R planar robot.

4.4. Blocking situation

If the invariant condition $\dot{\phi} \leq 0$ for all active constraints cannot be achieved by control vector $\mathbf{u}$, i.e. no solution of (24) exists, the robot motion must be stopped (i.e. $\dot{\mathbf{q}}_d$ must be set to zero) to avoid violating some constraint(s), giving rise to a blocking situation. For instance, such situation arises when the transversality condition (20) of an active constraint i is not satisfied and $L_f \phi_i > 0$.

If the minimum-amplitude auto-regulation algorithm described in Section 4.3 is used, the scale factor f_{AR} is self-regulated to zero when a blocking situation is found because some active constraint is permanently unfulfilled. In this case, in general, the value of control action $\mathbf{u}_{SM}$ must also be set to zero to completely stop the robot motion.

Another supervisor block could be used to set $\dot{\mathbf{q}}_d$ and/or $\mathbf{u}_{SM}$ to zero by checking if, for instance, some constraint has been repeatedly unfulfilled for a certain number of consecutive time steps or, alternatively, if the value of some constraint function ϕ_i has exceeded a certain predetermined threshold.

In order to illustrate the blocking situation, it is considered an open-chain planar mechanism composed by four links (the first of them is fixed) connected serially by three revolute joints, i.e. a 3R planar robot. For the configuration $\mathbf{q} = [2\pi/3 \quad 7\pi/6 \quad 0]^{\mathrm{T}}$ of this robot depicted in Fig. 6, the

Jacobian matrix $\mathbf{J}$ and matrix $\mathbf{B}$ result in:

$$\mathbf{J} = L \begin{bmatrix} -\sqrt{3}/2 + 1 & 1 & 1/2 \\ -1/2 + \sqrt{3} & \sqrt{3} & \sqrt{3}/2 \end{bmatrix} \tag{45}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 \\ -1/\sqrt{5} & 0 & 0 \\ 2/\sqrt{5} & 0 & 0 \end{bmatrix}, \tag{46}$$

where the same length L is considered for the three moving links.

Only constraint 1 above the robot has become active and its transversality condition is not fulfilled:

$$\nabla\sigma_1^{\mathrm{T}}\,\mathbf{B} = [-\sqrt{2}/2 \quad 0 \quad 0]\,\mathbf{B} = \mathbf{0}_n^{\mathrm{T}}. \tag{47}$$

The time derivative of the constraint function ϕ_1 results in:

$$\dot{\phi}_1 = \dot{\sigma}_1 = \nabla\sigma_1^{\mathrm{T}}\,\dot{\mathbf{q}} = \nabla\sigma_1^{\mathrm{T}}\,\mathbf{J}^{\dagger}\,\dot{\mathbf{p}}_d, \tag{48}$$

where $K = 0$ and $\mathbf{b}_c = \mathbf{0}_n$ have been used for simplicity.

Thus for zero tracking error, i.e. $\mathbf{e}_p = \mathbf{0}_m$, the condition $L_f\phi_1 \leq 0$ to avoid the blocking situation can be written as:

$$\dot{\phi}_1 = (\sqrt{2}/L)\,[\sqrt{3}/2 \quad -1/2]\,\dot{\mathbf{p}}_{ref} \leq 0, \tag{49}$$

which means that the angle of vector $\dot{\mathbf{p}}_{ref}$ must be within the interval $[\pi/3, 4\pi/3]$, see Fig. 6.

4.5. Workspace constraints

In practical applications with redundant robots one common objective is that the Cartesian position $\bar{\mathbf{p}}_j = [x_j \;\; y_j \;\; z_j]^{\mathrm{T}}$ of every point j of the robot[6] belongs to the allowed workspace $\Phi_{WS}(\bar{\mathbf{p}}_j) = \{\bar{\mathbf{p}}_j \,|\, \sigma_i(\bar{\mathbf{p}}_j) \leq 0 \;\forall\, i\}$. Thus,

[6]Obviously, the Cartesian position $\bar{\mathbf{p}}_{ee}$ of the robot end-effector or tracking point (i.e., the point that tracks the workspace reference $\mathbf{p}_{ref}$) must also belong to the allowed workspace Φ_{WS}. However, the tracking point does not satisfy the transversality condition (20) because from (42) $\mathbf{L_g}\phi_{ee} = \alpha\, K\, \nabla\sigma_{ee}^{\mathrm{T}}\,\mathbf{B}$, using the chain rule $\nabla\sigma_{ee}^{\mathrm{T}} = \frac{\partial\sigma_{ee}}{\partial\mathbf{q}}^{\mathrm{T}} = \frac{\partial\sigma_{ee}}{\partial\mathbf{p}_{ee}}^{\mathrm{T}} \frac{\partial\mathbf{l}_{ee}}{\partial\mathbf{q}} = \frac{\partial\sigma_{ee}}{\partial\mathbf{p}_{ee}}^{\mathrm{T}} \mathbf{J}$ and by definition (Section 2.1) $\mathbf{J}\,\mathbf{B} = \mathbf{O}_n$. Therefore, a blocking situation arises when the end-effector reaches the boundary $\partial\Phi_{WS}$ unless vector $\dot{\mathbf{p}}_{ref}$ fulfills a certain condition in order to obtain $L_f\phi_{ee} \leq 0$, see Section 4.4.

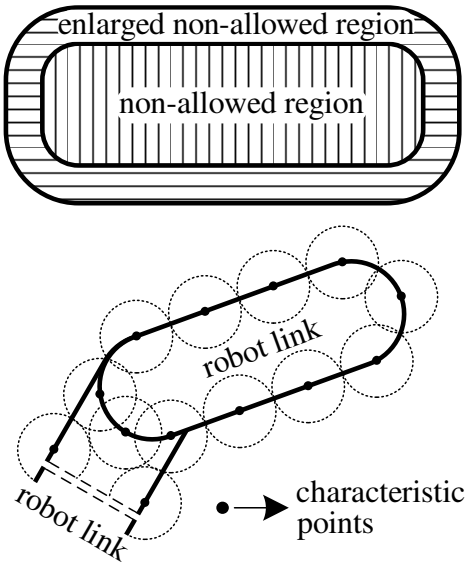

Fig. 7. Example of characteristic points of the robot.

the allowed C-space results in $\Phi_{CS}(\mathbf{q}) = \{\mathbf{q} \,|\, \sigma_i(\mathbf{l}_j(\mathbf{q})) \leq 0 \;\; \forall\, i, j\}$, where $\mathbf{l}_j$ is the kinematic function of the Cartesian position of point j.

The infinite number of points of the robot to be considered in the above expression can reduced to a set of *characteristic points* such that the distance from every point on the boundary surface of the robot links to the closest characteristic point is less than a predetermined value which is used to enlarge the constrained region of the workspace, see Fig. 7.

Some simplifications can be made in case the allowed workspace is *convex*. In such circumstances, the links could be enclosed with polyhedrons and the characteristic points to be considered are those on their vertices, whereas the original constrained region of the workspace does not have to be enlarged. Moreover, if the width of the robot links is negligible, the characteristic points to be considered are reduced to the end-points of the links.

4.6. Non-static environment

The proposed approach can also be used if there are moving constraints, e.g. moving obstacles with *known* trajectories. In this case σ_i also depends explicitly on time and, hence, the derivative of ϕ_i in equation (15) must be replaced by $\dot{\phi}_i = \widetilde{L_f \phi}_i + \mathbf{L_g}\phi_i \, \mathbf{u}$, where $\widetilde{L_f \phi}_i$ is equal to $L_f\phi_i + \partial\sigma_i/\partial t + K(\partial^2\sigma_i/\partial t^2 + 2\,\partial \nabla\sigma_i^{\mathrm{T}}/\partial t\;\dot{\mathbf{q}})$, and $\mathbf{L_g}\phi_i$ and $L_f\phi_i$ are given again by (42)

and (43), respectively (the explicit derivation of the above expression is omitted for brevity). Therefore, all developments keep unchanged except for changing $L_f\phi_i$ to $\widetilde{L_f\phi_i}$. Thus, only the value of the lower bound for u^+ is changed when moving constraints are considered and, hence, the iterative computation of the SM supervisor block of Section 4.2 remains the same.

4.7. Switching frequency and chattering

As in all SM controls, the theoretically infinite switching frequency cannot be achieved in practice because all physical systems have finite bandwidth. In computer implementations, the switching frequency is directly the inverse of the sampling period. Finite-frequency commutation makes the system leave the theoretical SM and, instead, its states oscillates with finite frequency and amplitude inside a "band" around $\boldsymbol{\phi} = \mathbf{0}$, which is known as 'chattering'.

For active constraints, the chattering band $\triangle\phi_i$ due to the SM supervisor block of Section 4.2 is given, using the Euler-integration, by:

$$\begin{aligned} \triangle\phi_i &= T_s \, |\mathbf{L_g}\phi_i \, \mathbf{u}_{SM}| = T_s \, \alpha \, K \, |\nabla\sigma_i^{\mathrm{T}} \, \mathbf{B} \, \mathbf{u}_{SM}| \\ &\leq T_s \, \alpha \, K \, \|\mathbf{u}_{SM}\|_2, \end{aligned} \tag{50}$$

where T_s is the sampling period of the proposed RRS and $\|\mathbf{u}_{SM}\|_2$ is the amplitude of the control action. (Note that $\|\nabla\sigma_i^{\mathrm{T}} \, \mathbf{B}\|_2 \leq 1$, see Section 2.1.)

The value of the original constraint function σ_i is obtained by passing signal ϕ_i through a first-order low-pass filter whose cutoff frequency is equal to $1/K$, see (39). This filter smooths out the chattering band of ϕ_i. In the worst case the chattering band $\triangle\sigma_i$ is equal to $\triangle\phi_i$ and it is reduced as the chattering frequency ω_ϕ and/or the constraint approaching parameter K increase. Thus the upper bound $\overline{\sigma}_{\max}$ for signal σ_i results in:

$$\overline{\sigma}_{\max} = T_s \, \alpha \, K \, \|\mathbf{u}_{SM}\|_2. \tag{51}$$

5. Additional remarks

5.1. Guidelines for designing the algorithm parameters

5.1.1. Constraint approaching parameter

The value of K can be interpreted as the *time constant* of the "braking" process when approaching the boundary of the original constraints σ_i, i.e., when approaching a constraint at high speed, the constraint will be reached in approximately $3K$ seconds and transversal speed will be also lowered to zero after that time has elapsed.

5.1.2. Cutoff frequency

The value of α must be higher than the frequency bandwidth of the desired workspace velocity vector $\dot{\mathbf{p}}_d$ in order to obtain a good approximation of the theoretical SM behavior, but not too high to avoid significant chattering (50).

5.1.3. Amplitude of the control action

The value of $\|\mathbf{u}_{SM}\|_2$ (which is directly related to u^+) has to be as close as possible to its lower bound given by (34) or (37) (with, perhaps, some safety margin) in order to have reduced chattering amplitude and high chattering frequency, see Section 4.7. Note that if the minimum-amplitude autoregulation of Section 4.3 is used, the value of $\|\mathbf{u}_{SM}\|_2$ can be made as small as desired at the expense of not guaranteeing the achievement of the desired values for the workspace velocity vector $\dot{\mathbf{p}}_d$ and performance vector $\mathbf{b}_c$.

5.1.4. Sampling period

The sampling period T_s has to be small enough in order for the discrete implementation of the filter to work properly, i.e. $T_s \ll \pi/\alpha$, and have small chattering amplitude (50).

5.2. Computational cost

The redundancy resolution used in this research is mainly based on the SVD of the robot Jacobian in order to obtain matrices $\overline{\mathbf{U}}$, $\overline{\Sigma}$, $\overline{\mathbf{V}}$ and $\widetilde{\mathbf{V}}$ to be used in equation (6). The computational complexity of the Golub-Reinsch SVD algorithm is $14mn^2 + 8n^3$ floating point operations for an $m \times n$ matrix [17]. For instance, the SVD computation of a 50×50 matrix in a modern computer takes about 3 milliseconds using MATLAB®'s *svd* function.

For on-line implementation, matrix $\mathbf{L_g}\boldsymbol{\phi}$ must be computed, which is plainly the multiplication of the null-space matrix $\mathbf{B} = [\widetilde{\mathbf{V}} \quad \mathbf{O}_{n\times r}]$ by the gradients of the constraints which can be preprogrammed. Hence, moderately-sized redundant robots can be managed with the on-line computation of (35) at sampling periods below one millisecond. Applications with harder real-time requirements can be managed with the on-line implementation of (36) where the only operation to be carried out is a matrix multiplication, so only one SVD of the Jacobian needs to be executed at each iteration.

5.3. Constraints definition

It is advisable to properly define all σ_i functions so that their orders of magnitude are comparable, for instance, being related to the minimum distance from $\mathbf{q}$ to the boundary of the ith-constraint. For instance, we may consider the constraint $\sigma_{plane} = \mathbf{n}_{plane}^{\mathrm{T}} (\mathbf{q} - \mathbf{q}_{plane}) \leq 0$ to indicate that a plane with normal vector $\mathbf{n}_{plane}$ and passing through point $\mathbf{q}_{plane}$ is included in the boundary of the C-space. However, instead of using σ_{plane} we could have used $5\sigma_{plane}$, or σ_{plane}^3, etc. Therefore, in this case the first option might be advised if similar criteria are used with the rest of constraints.

5.4. Security margin

In case it is needed for security reasons, the original constraint functions σ_i may be designed conservatively, taking into account the estimated chattering amplitude $\overline{\sigma}_{\max}$ and any other additional extra margin to cater for possible inaccuracies in the robot control or in the environment description.

5.5. Differentiability of the constraint functions

As stated in Assumption 2, the constraint functions σ_i must be twice differentiable and their second-order derivatives must be reasonably bounded in order to fulfill (34) or (37). If this assumption is not satisfied at a certain time, the SM behavior of the supervisor block (Section 4.2) is temporarily lost and the constraints may be unfulfilled.

5.6. Redundancy resolution with planning

The proposed redundancy resolution can be implemented on-line because it does not require future values of the reference trajectory and because it has a reasonably low computational cost: only linear algebra is used, no Hessian matrices are required (only gradients are used), etc. However, if future values of the reference trajectory were known, advanced robotic planning [2] could be used in the redundancy resolution to solve nonlinear optimization problems in order to, for example, avoid blocking situations, see Section 4.4. However, this issue is out of the scope of this work, where knowledge of future reference trajectories is *not* assumed.

5.7. Direct control of the robot

The proposed redundancy resolution can also be used with direct control of the robot, e.g. the classical model-based computed torque control scheme [26]. In particular, equation (4) is replaced by the the second order kinematics of the robot:

$$\ddot{\mathbf{p}} = \mathbf{J}(\mathbf{q})\,\ddot{\mathbf{q}} + \dot{\mathbf{q}}^{\mathrm{T}}\,\mathbf{H}_R\,\dot{\mathbf{q}} \;\rightarrow\; \ddot{\mathbf{p}}_d - \dot{\mathbf{q}}^{\mathrm{T}}\,\mathbf{H}_R\,\dot{\mathbf{q}} = \mathbf{J}(\mathbf{q})\,\ddot{\mathbf{q}}_d, \tag{52}$$

where $\mathbf{H}_R$ denotes the Hessian matrix of second-order partial derivatives of the robot kinematic function $\mathbf{l}(\mathbf{q})$, $\ddot{\mathbf{q}}_d$ is the desired workspace acceleration vector and $\ddot{\mathbf{q}}_d$ is the desired joint acceleration vector.

The robot control is as follows. First, vector $\ddot{\mathbf{q}}_d$ is obtained by the kinematic controller (Fig. 1) from the workspace reference $\mathbf{p}_{ref}$ and the robot state $(\mathbf{q}, \dot{\mathbf{q}})$. Next, vector $\ddot{\mathbf{q}}_d$ is computed in (52) from vector $\ddot{\mathbf{q}}_d$ and the robot state using the SVD of the robot Jacobian as described in Section 2.1. Finally, the torque vector $\boldsymbol{\tau}$ for the robot is computed from the desired joint acceleration vector $\ddot{\mathbf{q}}_d$ and the robot state $(\mathbf{q}, \dot{\mathbf{q}})$ using the robot inverse dynamical model (model-based computed torque control scheme). In this approach, no filter is required (Fig. 4) to have a unitary relative degree between ϕ_i and the discontinuous action $\mathbf{u}$, see Section 3.2.

The major advantage of this approach is that the kinematic framework assumption (Section 2.2) is not required and the main disadvantages are that the computation of the robot Hessian is needed and that, in general, the minimum-amplitude auto-regulation described in Section 4.3 cannot be used.

6. Simulation

In this section the main features of the constraint supervision algorithm developed in Section 4 are illustrated for a planar robot and for the classical PUMA-560 robot through simulation results obtained using MATLAB®.

6.1. Kinematic controller

For the simulations of this section, it is used a classical kinematic controller utilized for robotic trajectory tracking [27], see Fig. 8, which consists of a two-degree of freedom (2-DOF) control that incorporates a correction based on the position error $\mathbf{e}_p = \mathbf{p}_{ref} - \mathbf{p}$ by means of the position loop controller C_p plus a feedforward term depending on the first-order time derivative of the workspace reference, i.e. $\dot{\mathbf{p}}_{ref}$. For the simulations, the position

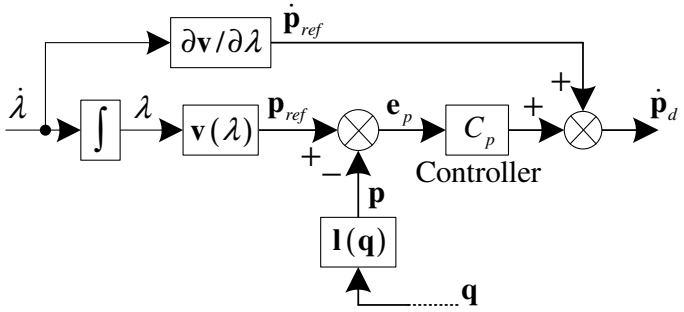

Fig. 8. Classical kinematic controller.

controller is simply implemented as a proportional controller with correction gain K_p. Therefore, the desired workspace velocity vector $\dot{\mathbf{p}}_d$ generated by the kinematic controller results in:

$$\dot{\mathbf{p}}_d = \frac{\partial \mathbf{v}(\lambda)}{\partial \lambda}\,\dot{\lambda} + K_p\,(\mathbf{p}_{ref} - \mathbf{p}). \tag{53}$$

Since $\dot{\mathbf{p}}_d$ is multiplied in the proposal by the scale factor f_{AR} obtained from the minimum-amplitude auto-regulation algorithm (see Section 4), it will be used $\dot{\lambda} = f_{AR}\,\dot{\lambda}_{max}$ and $K_p = f_{AR}\,K_{p\,max}$, where $\dot{\lambda}_{max}$ and $K_{p\,max}$ are the desired maximum values of the motion rate parameter and correction gain, respectively.

6.2. First example: 6R planar robot

In this first example, it is considered an open-chain planar mechanism composed by seven links (the first of them is fixed) connected serially by six revolute joints, i.e. a 6R planar robot. This robot is an extension of that shown in Fig. 6 with three more joints and links. Two elements are considered for the robot workspace vector: the cartesian coordinates $[x_6\ y_6]^{\mathrm{T}}$ of the end-effector position $\mathbf{p}_{ee} \equiv \mathbf{p}_6$. Therefore, this robot has four $(6-2)$ redundant degrees of freedom.

For this 6R robot, the kinematic function $\mathbf{l}_i(\mathbf{q})$ of the point located at the end of the ith moving link is given by:

$$\mathbf{p}_i(\mathbf{q}) = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = L \begin{bmatrix} \sum_{j=1}^{i} \cos(q_1 + \cdots + q_j) \\ \sum_{j=1}^{i} \sin(q_1 + \cdots + q_j) \end{bmatrix}, \quad i = 1, \ldots, 6, \tag{54}$$

where the origin of the reference frame is located at the first joint of the 6R robot and the same length L has been considered for the six moving links. Note that the robot Jacobian is readily obtained from (54) with $i = 6$.

6.2.1. Constraints

It will be considered that the allowed workspace for the robot is a circle whose center is (x_c, y_c) and radius is R_c. Since the allowed workspace is convex and assuming that the width of the robot links is negligible, the following constraints must be fulfilled to guarantee that every part of the 6R robot is inside the allowed workspace (see Section 4.5):

$$\sigma_i = -1 + (1/R_c)\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \leq 0, \quad i = 1, \ldots, 6. \tag{55}$$

Another constraint is considered to enforce the extension of the 6R robot:

$$\sigma_7 = 1 - (1/R_3)\sqrt{(x_3 - x_c)^2 + (y_3 - y_c)^2} \leq 0, \tag{56}$$

which means that point $\mathbf{p}_3$ must be *outside* of a circle whose center is (x_c, y_c) and radius is R_3 which, obviously, must be less than R_c to simultaneously fulfill (55).

The first three moving links of the 6R robot are considered to lie in the same plane. Therefore, in order to limit the values of the second and third joints for collision avoidance, the following two constraints are also considered:

$$\sigma_8 = -1 + |q_2|/q_{2\,max} \leq 0, \tag{57}$$

$$\sigma_9 = -1 + |q_3|/q_{3\,max} \leq 0, \tag{58}$$

where $q_{2\,max}$ and $q_{3\,max}$ are the maximum allowed values for the second and third joints, respectively.

6.2.2. Reference

The reference path is given by the following expression:

$$\mathbf{p}_{ref}(\lambda) = \begin{bmatrix} x_{ref}(\lambda) \\ y_{ref}(\lambda) \end{bmatrix} = \begin{bmatrix} x_c + R_c \sin(\lambda) \\ y_c + (R_c/2)\sin(2\lambda) \end{bmatrix}, \tag{59}$$

with $\lambda = 0...2\pi$.

Note that this reference path has the shape of the mathematical "infinity" symbol (∞), is centered within the allowed workspace and is tangent to its boundary at the two points given by $\lambda = \pi/2$ and $\lambda = 3\pi/2$.

6.2.3. Simulation conditions and parameter values

Simulation was run under the following conditions:

i) The kinematic framework was considered, i.e. $\dot{\mathbf{q}} \approx \dot{\mathbf{q}}_d$, see Assumption 1 in Section 2.2.

ii) No classical RRS was simulated (i.e. $\mathbf{b}_c = \mathbf{0}_n$) in order to focus on the behavior of the proposed SM algorithms.

iii) The control action $\mathbf{u}_{SM}$ was computed using (36), which is the equation that requires less computation time.

iv) The constraint functions ϕ_i were computed using a constraint approaching parameter K of 0.1 seconds.

v) The constraint supervision algorithm in Fig. 4 was implemented using a cutoff frequency α of 100 rad/s for the first-order low-pass filter and an amplitude $\|\mathbf{u}_{SM}\|_2 = 1$ for the switching law.

vi) The minimum-amplitude auto-regulation algorithm shown in Fig. 5 was implemented using a cutoff frequency of 100 rad/s for the first-order low-pass filter.

vii) The kinematic controller was implemented using a gain correction $K_{p\,max}$ of 20 s^{-1} in both coordinates and a maximum motion rate $\dot{\lambda}_{max}$ of 4 rad/s.

viii) All the algorithms were implemented with a sampling time T_s of half millisecond.

ix) The link length L was set to 1.

x) The center (x_c, y_c) and the radius R_c of the allowed workspace were set to $(0, 0)$ and 2.5, respectively.

xi) The constraint ϕ_7 was computed using a radius $R_3 = 2$.

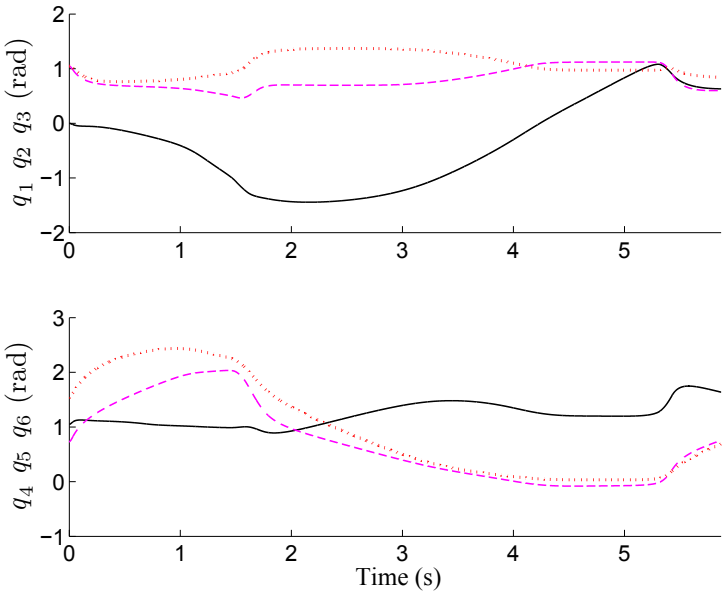

Fig. 9. Resulting joint positions $\mathbf{q}$: $\{q_1,q_4\}$ (solid), $\{q_2,q_5\}$ (dashed) and $\{q_3,q_6\}$ (dotted).

xii) The maximum allowed values for the second and third joints were set to $q_{2\,max} = 1.12$ and $q_{3\,max} = 1.37$, respectively.

xiii) We considered an initial robot position error $\mathbf{e}_p(0) = [0.1 \;\; -0.3]^{\mathrm{T}}$ and the initial robot configuration $\mathbf{q}(0) = [0 \;\pi/3\; \pi/3\; \pi/3\; 0.73\; 1.54]^{\mathrm{T}}$ rad.

6.2.4. Simulation results

Fig. 9 to Fig. 11 show the simulated behavior of the global system. In particular, the robot configuration at each time step is given by the joint positions shown in Fig. 9. For better visualization, twelve frames of the robot configuration at regular intervals of λ are shown in Fig. 10a–10b. Note that the tracking error is made zero and that all constraints[7] are fulfilled, i.e. $\min(\phi_i) <= 0$, see Fig. 11. Note also that the constraint supervision

[7]The constraints ϕ_1 and ϕ_2 are not represented in Fig. 11 because they are always fulfilled since $x_c = y_c = 0$ and $R_c > 2L$. Moreover the constraint ϕ_6, which is given by the tracking point $\mathbf{p}_6$, is neither represented because it does not satisfy the transversality condition (20) (see Section 3 and Section 4.5) and, therefore, it is not affected by control action $\mathbf{u}$.

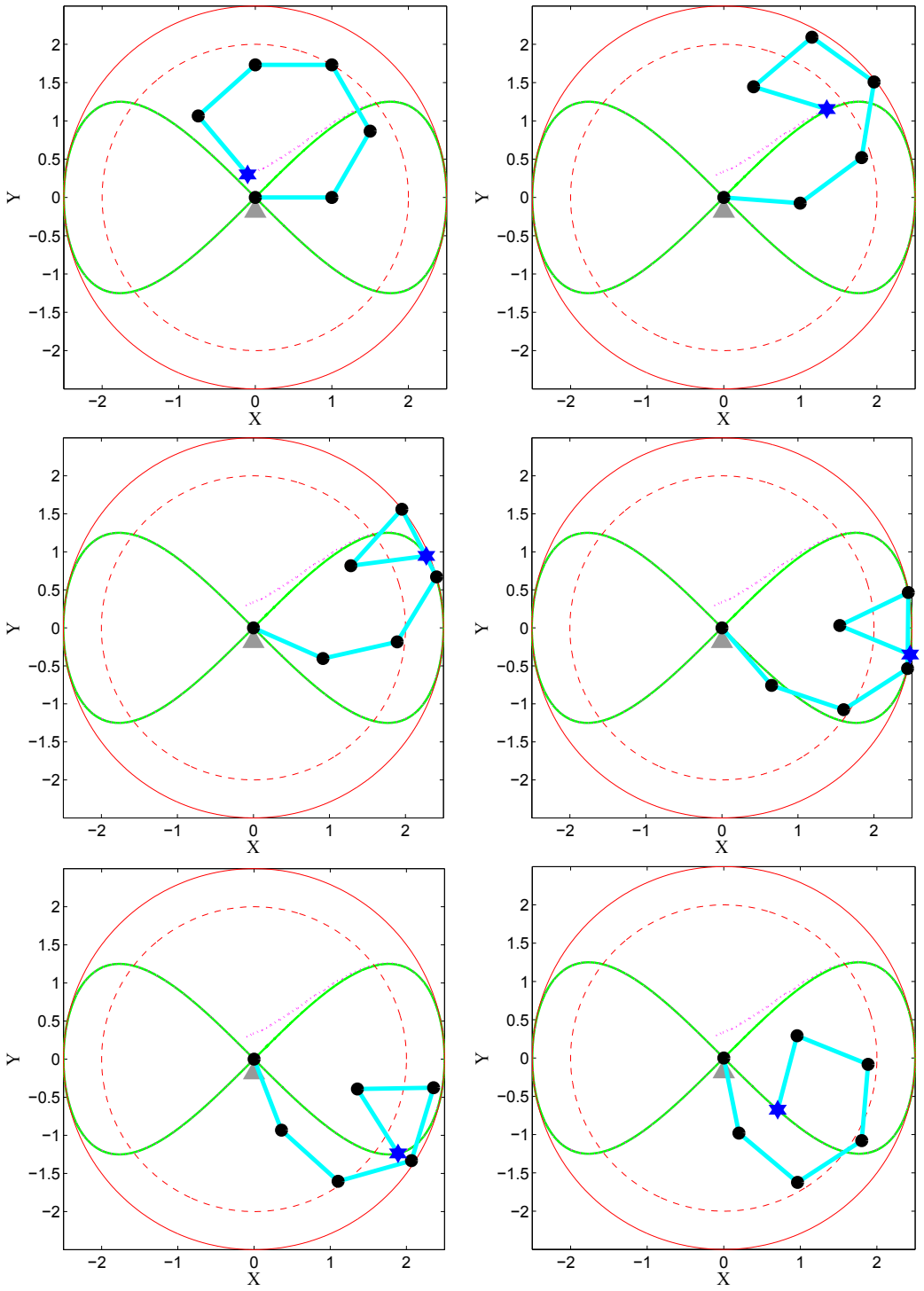

Fig. 10a. Frames 1 (top left) to 6 (bottom right) of the robot configuration at regular intervals of λ: reference path (thick solid line), path followed by the robot end-effector (dotted line), boundary of the allowed workspace (thin solid line), boundary of the constraint ϕ_7 for point $\mathbf{p}_3$ (thin dashed line), robot joints (solid discs), end-effector (solid star), moving links (extra thick solid lines) and fixed link (solid triangle). In Fig. 11 the active constraints at each frame are also shown.

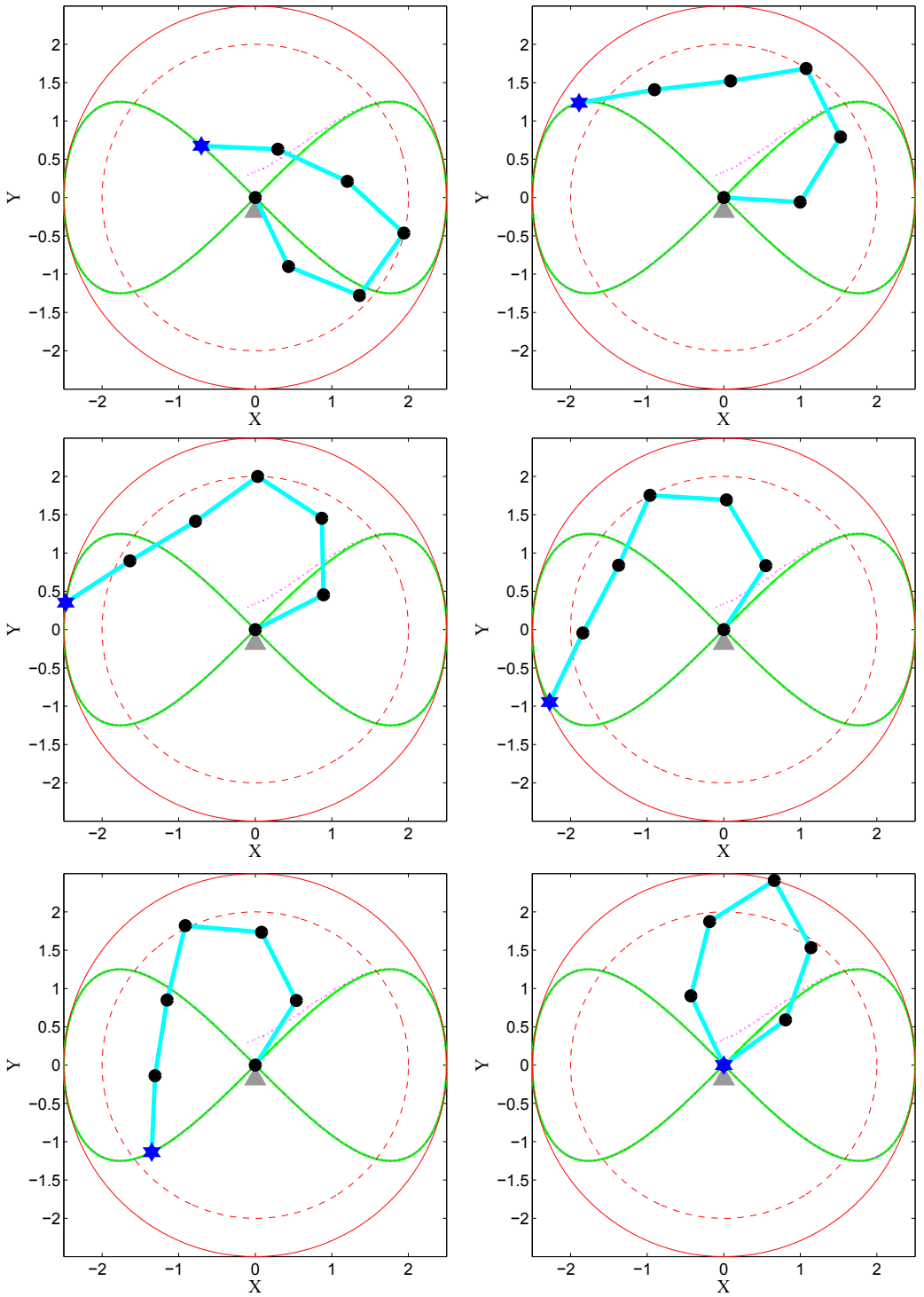

Fig. 10b. Frames 7 (top left) to 12 (bottom right) of the robot configuration at regular intervals of λ: reference path (thick solid line), path followed by the robot end-effector (dotted line), boundary of the allowed workspace (thin solid line), boundary of the constraint ϕ_7 for point $\mathbf{p}_3$ (thin dashed line), robot joints (solid discs), end-effector (solid star), moving links (extra thick solid lines) and fixed link (solid triangle). In Fig. 11 the active constraints at each frame are also shown.

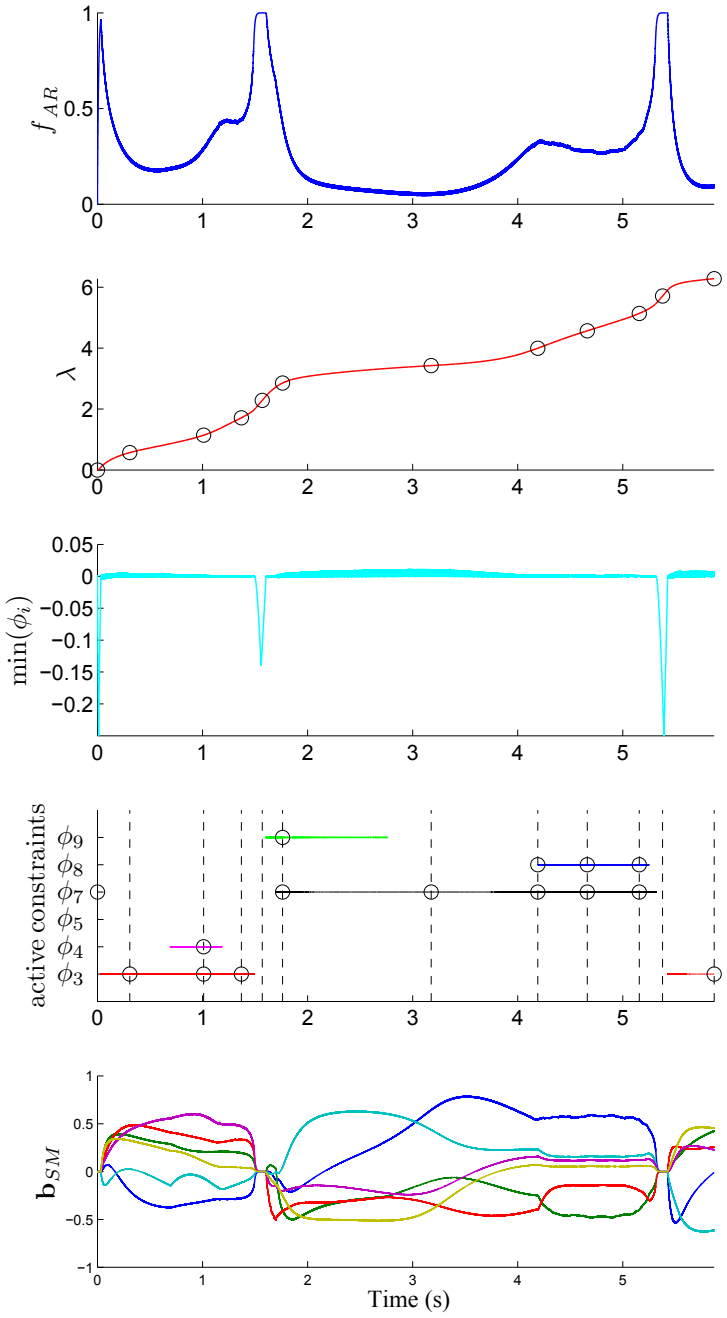

Fig. 11. Scale factor f_{AR} for the desired workspace velocity vector $\dot{\mathbf{p}}_d$; motion parameter λ (the circles correspond to the time instants of the frames in Fig. 10a–10b); minimum value of the constraint functions ϕ_i; horizontal lines indicating when a constraint is active (the dashed vertical lines correspond to the time instants of the frames in Fig. 10a–10b and the circles indicate the active constraints at those instants); filtered value $\mathbf{b}_{SM}$ of the control action $\mathbf{u}$ generated by the constraint supervision algorithm.

algorithm operates in the SM (in some phases with several active constraints) except in two phases around $t = 1.5$ seconds and $t = 5.3$ seconds where no constraint is active, see Fig. 11. Moreover, the minimum-amplitude auto-regulation algorithm scales (i.e. $f_{AR} < 1$) the desired workspace velocity vector $\dot{\mathbf{p}}_d$ except in the two mentioned phases where $f_{AR} = 1$. Furthermore, the filtered value $\mathbf{b}_{SM}$ of the control action $\mathbf{u}$ generated by the constraint supervision algorithm is zero in these two phases because no constraint is active.

6.3. Case study: PUMA robot

In this case study, the well-known 6DOF robotic arm PUMA-560 is considered, which is a classical 6R serial manipulator with spherical wrist. The results shown have been obtained with the freely accessible *Robotics Toolbox* (Release 7.1) for MATLAB® developed by P. Corke [28]. This Toolbox includes the kinematic model of the PUMA-560 robot, which has been used to generate the results.

Three elements are considered for the robot workspace vector: the cartesian coordinates $[x_6\ y_6\ z_6]^{\mathrm{T}}$ of the end-effector position $\mathbf{p}_{ee} \equiv \mathbf{p}_6$, i.e., there is no reference for the end-effector orientation. Assuming that the end-effector position lies as usual on the last joint axis, the angle q_6 of the last joint has no influence on the end-effector position and, hence, the last joint will not be considered. Therefore, the robot has two $(5 - 3)$ degrees of redundancy.

6.3.1. Constraints

It will be considered that the boundary of the allowed workspace is given by three vertical planes a, b and c. Since the allowed workspace is convex and assuming for simplicity that the width of the robot links is negligible, the following constraints must be fulfilled to guarantee that every part of the PUMA robot is inside the allowed workspace (see Section 4.5):

$$\sigma_{ai} = y_i - y_a \leq 0, \tag{60}$$

$$\sigma_{bi} = -(y_i - y_b) \leq 0, \tag{61}$$

$$\sigma_{ci} = x_i - x_c \leq 0, \qquad i = 1, \ldots, 6, \tag{62}$$

where the subindex i is associated with the end-point of the ith moving link (e.g., $\mathbf{p}_i$ is the position of the end-point of the ith moving link) and y_a, y_b and x_c are the parameters of the vertical planes a, b and c, respectively.

The following constraints are also considered for the joint limits:

$$\sigma_{qi} = -1 + |q_{norm\,i}| \leq 0, \quad i = 1, \ldots, 5. \tag{63}$$

where $q_{norm\,i} = (q_i - q_{mid\,i})/(\Delta q_{max\,i}/2)$ is the normalized joint position obtained using the mid joint position $q_{mid\,i}$ and the joint maximum range of motion $\Delta q_{max\,i}$.

6.3.2. Reference

The end-effector reference path is given by the following modified helicoidal expression:

$$\mathbf{p}_{ref}(\lambda) = \begin{bmatrix} x_{ref}(\lambda) \\ y_{ref}(\lambda) \\ z_{ref}(\lambda) \end{bmatrix} = \begin{bmatrix} 0.6277 + 0.15\left(\cos(\lambda) - \sin^2(\lambda) - 1\right) \\ -0.1501 + 0.15\sin(\lambda) \\ 0.0926 - 0.0375\lambda \end{bmatrix}, \tag{64}$$

with $\lambda = 0...4\pi$, where the units for linear and angular dimensions are meters and radians, respectively. It is important to recall that for the PUMA-560 manipulator the Z-axis of the robot base frame is aligned with the first joint and its origin is located at the same height of the second joint, i.e., the shoulder joint.

6.3.3. Simulation conditions and parameter values

Simulation was run under the following conditions:

xiv) The same conditions i) to viii) of the first example were used (see Section 6.2.3), except for the constraint approaching parameter K of the joint limit constraints which was set to 0.02 seconds.

xv) The tool length was set to 94 mm, i.e., the distance from the end-effector position to the wrist center is equal to 150 mm.

xvi) The workspace constraints were computed with $y_a = 0.05$ m, $y_b = -0.32$ m and $x_c = 0.645$ m.

xvii) The joint limit constraints were computed using a mid joint position vector $\mathbf{q}_{mid} = [0\ \pi/2\ -\pi/2\ \pi/6\ 0]^{\mathrm{T}}$ rad and a joint maximum range of motion $\Delta\mathbf{q}_{max} = [5.55\ 4.643\ 4.957\ 4.887\ 3.491]^{\mathrm{T}}$ rad, see [29, 30].

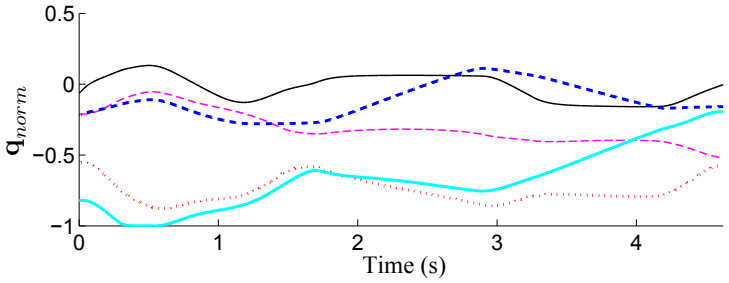

Fig. 12. Resulting normalized joint positions $\mathbf{q}_{norm}$: $q_{norm\,1}$ (thin solid line), $q_{norm\,2}$ (thin dashed line), $q_{norm\,3}$ (dotted line), $q_{norm\,4}$ (thick dashed line) and $q_{norm\,5}$ (thick solid line).

xviii) We considered an initial robot pose error $\mathbf{e}_p(0) = [0.05\ 0.1\ 0]^{\mathrm{T}}$ m and the initial robot configuration $\mathbf{q}(0) = [-0.168\ 1.084\ -2.935\ 0\ -1.29]^{\mathrm{T}}$ rad.

6.3.4. Simulation results

Fig. 12 to Fig. 14 show the simulated behavior of the global system. The robot configuration at each time step is given by the normalized joint positions shown in Fig. 12, where it can be seen that the joint limit constraints are fulfilled, i.e., $|q_{norm\,i}| \leq 1$. Fig. 13 shows the paths followed by the reference and the end-points of the moving links[8], where it can bee seen that the tracking error is made zero and that the workspace constraints are also fulfilled. Fig. 14 shows that, as expected, all constraints are fulfilled (i.e., $\min(\phi_i) <= 0$) and that the constraint supervision algorithm operates in the SM (i.e., the value of $\mathbf{b}_{SM}$ is non-zero) when some constraint is active, which occurs in six phases of different lengths.

7. Conclusions

A variable structure algorithm for redundancy resolution was proposed using sliding mode related concepts. The strategy acts as a supervisory loop, shaping the commanded joint velocities in order to fulfill C-space and/or

[8]The path followed by point $\mathbf{p}_5$ is not shown in Fig. 13 because this point lies on the straight line connecting the points $\mathbf{p}_4$ and $\mathbf{p}_6$ (i.e., $\mathbf{p}_5$ fulfills the workspace constraints if both $\mathbf{p}_4$ and $\mathbf{p}_6$ fulfill them, see Section 4.5). Note also that point $\mathbf{p}_1$ remains static.

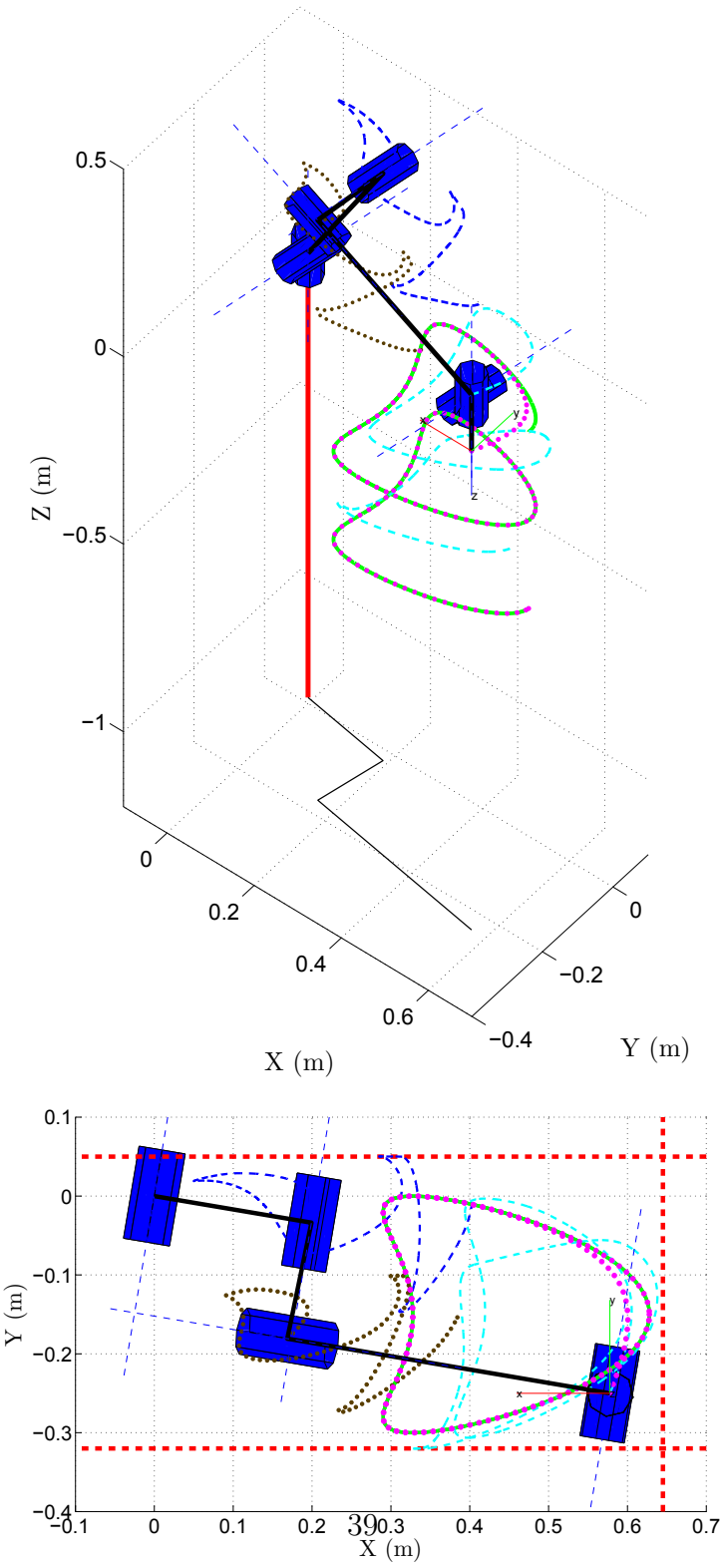

Fig. 13. 3D and top views of the PUMA robot in its initial configuration and the paths followed by the reference (thick solid line), robot end-effector $\mathbf{p}_6$ (thick dotted line) and points $\mathbf{p}_2$ (dark dashed line), $\mathbf{p}_3$ (thin dotted line) and $\mathbf{p}_4$ (light dashed line). The limit planes of the workspace constraints are shown as thick dotted lines in the top view.

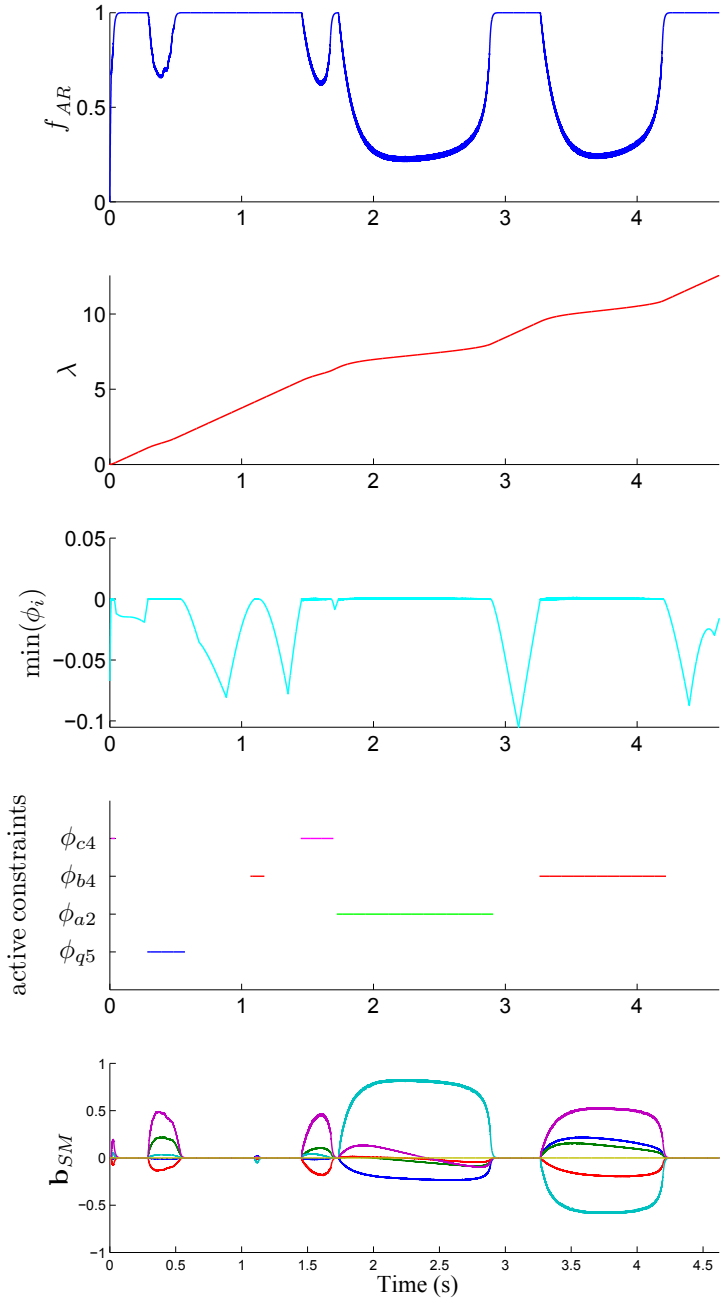

Fig. 14. Scale factor f_{AR} for the desired workspace velocity vector $\dot{\mathbf{p}}_d$; motion parameter λ; minimum value of the constraint functions ϕ_i; horizontal lines indicating when a constraint is active (only the constraints which become active at least once are shown); filtered value $\mathbf{b}_{SM}$ of the control action $\mathbf{u}$ generated by the constraint supervision algorithm.

workspace constraints. In this manner, the algorithm activates when the robot is about to violate the constraints, modifying the commanded joint velocities as much as necessary in order to fulfill all the constraints and reaching their limit surface smoothly depending on a free design parameter. The proposal also includes an additional sliding-mode supervisor block in order to auto-regulate the minimum-amplitude required for the main supervisory loop to achieve the sliding regime.

The proposal can be easily added as an auxiliary supervisory loop to conventional redundancy resolution schemes. Moreover, the proposed algorithm improves the classical conservative potential field-based approach for collision avoidance in the sense that it fully exploits the robot workspace and allows an additional secondary task while the constraints are fulfilled.

Although the algorithm was illustrated for a particular kinematic controller and two particular robots (6R planar robot and PUMA-560 robot), the conclusions drawn for the redundancy resolution method also apply to other kinematic controllers and/or redundant robots.

Acknowledgements

This research is partially supported by DISICOM project PROMETEO 2008/088 of Generalitat Valenciana (Spain), research project DPI2008-06731-C02-01 of the Spanish Government (Spain), Technical University of Valencia (Spain), and the Argentinian Government (UNLP 11I127, CONICET PIP 112-200801-0, ANPCyT PICT 2007 00535).

References

[1] A. Liégeois, Automatic supervisory control of the configuration and behavior of multibody mechanisms, IEEE Transactions on Systems, Man and Cybernetics 7 (1977) 868–871.

[2] J.-C. Latombe, Robot Motion Planning, Kluwer, Boston, 1991.

[3] K. AlSultan, M. Aliyu, A new potential field-based algorithm for path planning, Journal of Intelligent and Robotic Systems 17 (1996) 265–282.

[4] E. Rimon, D. Koditschek, Exact robot navigation using artificial potential functions, IEEE Transactions on Robotics and Automation 8 (1992) 501–518.

[5] J. Juang, Collision avoidance using potential fields, Industrial Robot 25 (1998) 408–415.

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

[7] W. Perruquetti, J. Barbot (Eds.), Sliding Mode Control in Engineering, Control Engineering Series, Marcel Dekker, 2002.

[8] L.-G. Garcia-Valdovinos, V. Parra-Vega, M. Arteaga, Observer-based sliding mode impedance control of bilateral teleoperation under constant unknown time delay, Robotics and Autonomous Systems 55 (2007) 609–617.

[9] W. Bessa, M. Dutra, E. Kreuzer, Depth control of remotely operated underwater vehicles using an adaptive fuzzy sliding mode controller, Robotics and Autonomous Systems 56 (2008) 670–677.

[10] V. Sankaranarayanan, A. Mahindrakar, Control of a class of underactuated mechanical systems using sliding modes, IEEE Transactions on Robotics 25 (2009) 459–467.

[11] W. Bessa, M. Dutra, E. Kreuzer, An adaptive fuzzy sliding mode controller for remotely operated underwater vehicles, Robotics and Autonomous Systems 58 (2010) 16–26.

[12] F. Garelli, R. Mantz, H. De Battista, Limiting interactions in decentralized control of MIMO systems, Journal of Process Control 16 (2006) 473–483.

[13] F. Garelli, L. Gracia, A. Sala, P. Albertos, Sliding mode speed autoregulation technique for robotic tracking, Robotics and Autonomous Systems 59 (2011) 519–529.

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

[15] O. Yashi, K. Ozgoren, Minimal joint motion optimization of manipulators with extra degrees of freedom, Mechanism and Machine Theory 19 (1984) 325–330.

[16] D. Whitney, Resolved motion rate control of manipulators and human prostheses, IEEE Transactions on Man-Machine Systems 10 (1969) 47–53.

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

[18] L. Gracia, J. Andres, J. Tornero, Trajectory tracking with a 6R serial industrial robot with ordinary and non-ordinary singularities, International Journal of Control, Automation, and Systems 7 (2009) 85–96.

[19] C. Wampler, Manipulator inverse kinematic solutions based on vector formulations and damped least squares methods, IEEE Transactions on Systems, Man, and Cybernetics 16 (1986) 93–101.

[20] G. Marani, J. Kim, J. Yuhl, W. K. Chung, A real-time approach for singularity avoidance in resolved motion rate control of robotic manipulators, in: Proceedings of IEEE International Conference on Robotics and Automation, volume 2, Washington, DC, USA, pp. 1973–1978.

[21] L. Huo, L. Baron, The joint-limits and singularity avoidance in robotic welding, Industrial Robot: An International Journal 35 (2008) 456–464.

[22] E. Bernabeu, J. Tornero, Optimal geometric modeler for robot motion planning, Journal of Robotic Systems 17 (2000) 593–608.

[23] H. Amann, Ordinary differential equations, Walter de Gruyter, Berlin, 1st edition, 1990.

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

[25] S. Ma, A kinetostatic index to measure the task-executing ability of robotic manipulators with limit-driven characteristics of actuators, Advanced Robotics 18 (2004) 401–414.

[26] C. Canudas de Wit, B. Siciliano, G. Bastin, Theory of Robot Control, Springer-Verlag, Secaucus, NJ, 1996.

[27] W. Khalil, E. Dombre, Modeling, Identification and Control of Robots, Taylor & Francis Inc., Bristol, PA, 2002.

[28] P. Corke, A robotics toolbox for matlab, IEEE Robotics and Automation Magazine 3 (1996) 24–32.

[29] S. Elgazzar, Efficient kinematic transformations for the PUMA 560 robot, IEEE Journal of Robotics and Automation 1 (1985) 142–151.

[30] Y. Zhang, J. Wang, A dual neural network for constrained joint torque optimization of kinematically redundant manipulators, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 32 (2002) 654–662.