Robotics and Computer-Integrated Manufacturing 30 (2014) 74–89

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

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

Robot coordination using task-priority and sliding-mode techniques

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

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

ARTICLE INFO

Article history:
Received 3 May 2013
Received in revised form
26 August 2013
Accepted 30 August 2013

Keywords:
Multi-robot systems
Cooperative robots
Robot control
Collision avoidance

ABSTRACT

In this work, an approach based on task-priority redundancy resolution and sliding mode ideas is proposed for robot coordination. In particular, equality and inequality constraints representing the coordination of the multi-robot system are considered as mandatory (for instance, rigid-body manipulation constraints to distance between the end-effectors of several robot arms, or other inequality constraints guaranteeing safe operation of a robotic swarm or confining the robot's workspace to avoid collision and joint limits). Besides the mandatory constraints, other constraints with lower priority are considered for the tracking of the workspace reference and to achieve secondary goals. Thus, lower-priority constraints are satisfied only in the null space of the higher-priority ones. The fulfillment of the constraints is achieved using geometric invariance and sliding mode control theory. The validity and effectiveness of the proposed approach are substantiated by 2D and 3D simulation results using two 3R planar robots and two 6R PUMA-762 robots, respectively.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

Research topics for multi-robot systems (MRSs) [1], also known as distributed [2] or cooperative [3] or multi-agent [4] robotic systems, have significantly increased in recent years: biological inspirations, robotic swarms, object transport and manipulation, reconfigurable robots, communication and architectures, reactivity and social deliberation, motion coordination, etc. In general, a team of robots provides redundancy and contributes cooperatively to solve the task for which they were conceived in a more reliable or faster way than what is possible with a single robot.

Coordination is an important issue in robotics and integrated manufacturing systems [5–7]. Typically, robot coordination [8] can be expressed in terms of equality and inequality constraints. These constraints could be attempted to solve analytically, but in most cases a closed-form solution to the problem is not possible, specially if inequality constraints are present.

This work is focused on robot coordination within the context of MRSs. In order to fulfill equality and inequality constraints, or reaching such feasible regions in finite time if not initially in them, sliding-mode (SM) control theory has been used in the literature [9,10], in particular, geometric-invariance approaches [11]. Previous work by the authors discusses how sliding-mode geometric invariance can be used to solve some problems in single-robot systems [12–14], while the third author has preliminarily evaluated its effectiveness for the coordination of single scalar realizable references commanding input-constrained dynamical systems [15,16]. The objective of this work is generalizing the above ideas to deal with MRSs as well as incorporating constraint prioritization situations. Other ways to tackle constraint-satisfaction and coordination problems in robotic systems are artificial intelligence [17], neural networks [18], fuzzy logic [19], genetic algorithms [20], etc.

This paper proposes a solution to the coordination problem based on task-priority redundancy resolution [21] and sliding mode (SM) control theory. The basic idea is to define mandatory constraints for the MRS in order to satisfy them via SM conditioning [22]. For this purpose, a coordination supervisor, which is aware of the configuration of each robot, generates the commanded joint accelerations to be sent to the joint controllers of the robots (which are controlled independently using their own control architecture) so that the constraints are fulfilled and that the reference tracking error is made as small as possible. The proposed method only requires a few program lines, has a short computation time and simplifies the user interface since the method directly deals with the fulfillment of the constraints specified by the robot end-user. However, since the proposed approach does not include high-level planning, other more sophisticated solutions in the literature for robot coordination [17–19] could lead to a better performance in certain cases, e.g., when facing trap situations [12], at the expense, however, of a much higher computational cost.

* Corresponding author. Tel.: +34 963877007.
E-mail addresses: luigraca@isa.upv.es (L. Gracia), asala@isa.upv.es (A. Sala), fabricio@ing.unlp.edu.ar (F. Garelli).

0736-5845/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.rcim.2013.08.003

The structure of the paper is as follows. Next section introduces some preliminaries and notation, while Section 3 states the main problem to be addressed. Next, Section 4 presents the basic theory used in this work. The proposed method for robot coordination is developed in Section 5, while some important remarks about the method are given in Section 6. The proposed approach is applied in Sections 7 and 8 to the model of two three-revolute (3R) planar robots and two 6R PUMA-762 robots, respectively, in order to show its feasibility and effectiveness. Finally, some concluding remarks are given.

2. Preliminaries and notation

Let us describe the different elements of a multi-robot system, composed of r robots to be coordinated.

Kinematics of the individual robots. Following the standard notation [23], consider the i-th robot of a MRS the n_i-dimensional joint position vector $\mathbf{q}_i=[q_1 \ldots q_{n_i}]^{\mathrm{T}}$ being its *configuration*, and $\mathbf{p}_i=[p_1 \ldots p_{m_i}]^{\mathrm{T}}$ being its *pose* expressed as an m_i-dimensional workspace position vector. As widely known, the relationship between the configuration of the robot and its pose is generically expressed as

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

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

$$\dot{\mathbf{p}}_i=\frac{\partial \mathbf{l}_i(\mathbf{q}_i)}{\partial \mathbf{q}_i}\dot{\mathbf{q}}_i=\mathbf{J}_{q_i}(\mathbf{q}_i)\dot{\mathbf{q}}_i, \quad \ddot{\mathbf{p}}_i=\mathbf{J}_{q_i}(\mathbf{q}_i)\ddot{\mathbf{q}}_i+\dot{\mathbf{J}}_{q_i}(\mathbf{q}_i)\dot{\mathbf{q}}_i, \tag{2}$$

where $\mathbf{J}_{q_i}(\mathbf{q}_i)$ is the $m_i \times n_i$ Jacobian matrix, or simply *Jacobian*, of the kinematic function $\mathbf{l}_i$.

Kinematics of the multi-robot system. Considering that the MRS is composed of r robots, the juxtaposition of vectors $\mathbf{q}=[\mathbf{q}_1^{\mathrm{T}} \ldots \mathbf{q}_r^{\mathrm{T}}]^{\mathrm{T}}$ and $\mathbf{p}=[\mathbf{p}_1^{\mathrm{T}} \ldots \mathbf{p}_r^{\mathrm{T}}]^{\mathrm{T}}$ are the composite configuration and composite pose of the MRS, respectively. Therefore, the first- and second-order kinematics of the MRS result in

$$\dot{\mathbf{p}}=\begin{bmatrix} \mathbf{J}_{q_1}(\mathbf{q}_1) & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{O} & \mathbf{J}_{q_2}(\mathbf{q}_2) & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O} & \mathbf{O} & \cdots & \mathbf{J}_{q_r}(\mathbf{q}_r) \end{bmatrix}\dot{\mathbf{q}}=\mathbf{J}_q(\mathbf{q})\dot{\mathbf{q}}, \tag{3}$$

$$\ddot{\mathbf{p}}=\mathbf{J}_q(\mathbf{q})\ddot{\mathbf{q}}+\dot{\mathbf{J}}_q(\mathbf{q})\dot{\mathbf{q}}, \tag{4}$$

where $\mathbf{O}$ denotes the zero matrix of suitable size and $\mathbf{J}_q(\mathbf{q})$ defines the MRS Jacobian, which is an $m \times n$ block diagonal matrix, where $m=m_1+\cdots+m_r$ and $n=n_1+\cdots+n_r$.

Robot control. In this paper, it will be assumed that there exists an underlying robot control hardware and software which will be in charge of achieving a particular joint acceleration from an acceleration command $\ddot{\mathbf{q}}_c$. In general, the actual joint acceleration $\ddot{\mathbf{q}}=\mathcal{C}\ddot{\mathbf{q}}_c+\mathbf{d}_c$ will not be exactly the commanded one $\ddot{\mathbf{q}}_c$, where $\mathcal{C}$ represents the dynamics of the low-level robot control loop and $\mathbf{d}_c$ represents inaccuracies due to other torque disturbances. However, we will assume that the dynamics of $\mathcal{C}$ is fast enough compared to that of $\ddot{\mathbf{q}}_c$ so that the relationship:

$$\ddot{\mathbf{q}}=\ddot{\mathbf{q}}_c+\mathbf{d}_c, \tag{5}$$

holds approximately true, avoiding the need for complicating the model with extra state variables.

Hard workspace constraints. Several types of inequality constraints are present in MRSs whatever the task to be accomplished. We will denote as hard constraints those which must be satisfied at all times for reasons of safety, physical limits, etc., so that if they are not fulfilled, the robot operation must be aborted. On one hand, each individual robot has its own constraints. For instance, an inequality constraint of the form $\sigma(\mathbf{q}_i)\leq 0$, where σ is a generic constraint function, can be used to avoid collision between robot i and the obstacles of the environment, to avoid exceeding the physical joint limits of robot i, etc. On the other hand, there are hard constraints that involve several robots. For instance, an inequality constraint of the form $\sigma(\mathbf{q}_i,\mathbf{q}_j)\leq 0$ can be used to avoid collisions between robot i and robot j. Other constraints regarding speed of movement could also be stated, in expressions in the form $\phi(\mathbf{q},\dot{\mathbf{q}})\leq 0$.

3. Problem definition and objectives

Task specification. The above-defined robot system with its dynamics, controllers and hard workspace constraints should carry out a coordinated task, which can be thought of achieving a reference value for a certain set of functions of the global pose. Hence, the coordinated task for the MRS will be expressed by the following equation:

$$\mathbf{v}_{ref}(t)=\mathbf{v}(\mathbf{p}(t)), \tag{6}$$

where $\mathbf{v}$ is the *coordination vector*, which is a function of the composite pose $\mathbf{p}$, and $\mathbf{v}_{ref}(t)$ is the reference trajectory for this vector generated by an external operator or high-level planner. The number of equalities in (6) will usually be lower than the degrees of freedom of the MRS but these degrees of freedom are reduced if the above mentioned hard constraints become active.

The first- and second-order kinematics of $\mathbf{v}$ result in

$$\dot{\mathbf{v}}=\frac{\partial \mathbf{v}(\mathbf{p})}{\partial \mathbf{p}}\dot{\mathbf{p}}=\frac{\partial \mathbf{v}(\mathbf{p})}{\partial \mathbf{p}}\mathbf{J}_q(\mathbf{q})\dot{\mathbf{q}}=\mathbf{J}_{pq}(\mathbf{q})\dot{\mathbf{q}}, \tag{7}$$

$$\ddot{\mathbf{v}}=\mathbf{J}_{pq}(\mathbf{q})\ddot{\mathbf{q}}+\dot{\mathbf{J}}_{pq}(\mathbf{q})\dot{\mathbf{q}}, \tag{8}$$

where $\mathbf{J}_{pq}(\mathbf{q})$ is defined as the product of matrices $\partial\mathbf{v}/\partial\mathbf{p}$ and $\mathbf{J}_q$ from chain rule as, from (1), obviously, the composite pose $\mathbf{p}$ is a function of the composite configuration $\mathbf{q}$.

With the above setup, the coordination task given by (6) represents an equality constraint for the MRS. The components of $\mathbf{v}_{ref}$ can be either constant or varying in time.

For example, one of the coordination vector components can be used to ensure that the distance between the end-effectors of robot i and robot j remains unchanged when they are used to cooperatively carry an object while satisfying its rigid body constraint, see Fig. 1. In the same example, another components

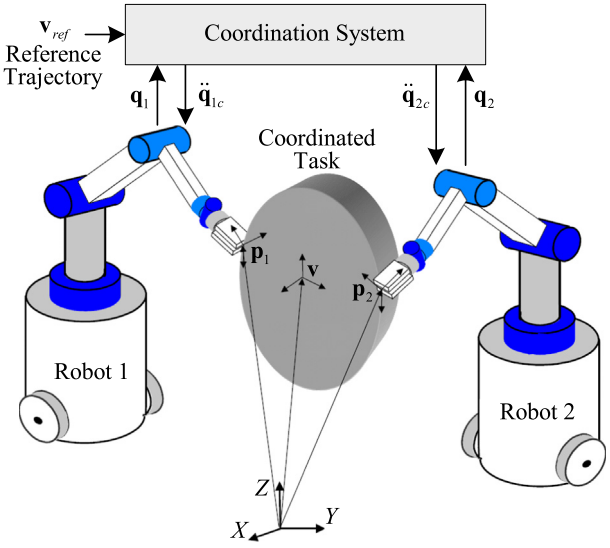

Fig. 1. Coordination of multi-robot systems.

can be used, for instance, to ensure that the carried object tracks a reference trajectory and orientation.

Note: Once the main ideas are presented, in Section 5.7, inspired on [24], the setting will be generalized to one in which the reference $\mathbf{v}_{ref}(t)$ will be expressed as $\mathbf{v}_{ref}(s(t))$ $s(t)$ being a scalar path parameter whose speed $\dot{s}$ can be varied in order to adapt the speed of the task execution. For better readability, such an issue will not be considered for the moment.

Constraint prioritization. In general, different levels of priority can be assigned to the constraints of the MRS. Such a priority would indicate that a lower-level constraint can be violated if it is necessarily needed in order to enforce a higher-level one.

In particular, the following levels are considered in this work:

- The first level includes the hard inequality constraints mentioned in Section 2 (collision avoidance, joint limits, etc.) as well as those considered in the specification of the coordinated task, for instance, keeping fixed distance and relative orientation between effectors in order to avoid breaking a transported object (rigid-body constraints).
- The second level includes the remaining constraints of the coordination task.
- Additionally, other soft constraints can be included into this level (or in separate ones): for instance, keeping non-critical distance bounds, soft constraints on the speed of movement, etc.

Therefore, the constraints of the first level will be considered mandatory for the MRS, while those belonging to second (and lower, if so introduced) levels will be satisfied, if possible, using the remaining degrees of freedom of the MRS. As later discussed, second-level constraints can be *weighted* in order to apportion the amount of violation if needed to fulfill mandatory ones. Obviously, there is no need to define such weights in the first-level ones as they will always be fulfilled.

As discrete-time implementations of sliding regimes have a small chattering band around the constraint (see [9] and Section 5.6), mandatory constraints should take into account such a residual error in order to be cautiously defined. Note that these issues are not too relevant if the proposed strategies are evaluated at fast sampling rates.

3.1. Problem statement

Once the main concepts on the multi-robot setup and the coordinated task to be executed have been presented (and are assumed to be specified a priori), let us precisely state the problem to be addressed in the following sections of this work.

Constraint space. From the above considerations, we will assume that the MRS to be controlled is subjected to inequality constraints[1] with different levels of priority. This will give rise to lth-level feasible sets denoted as

$$\Phi_l = \{[\mathbf{q}^T \ \dot{\mathbf{q}}^T]^T | \phi_{l,i}(\mathbf{q},\dot{\mathbf{q}}) \le 0\}, \quad l = 1,\ldots,M, \quad i = 1,\ldots,C_l, \tag{9}$$

where Φ_l is the allowed region of the MRS's state-space given by the constraints of the lth-level of priority, M is the number of priority levels, C_l is the number of constraints in the lth-level of priority, and $\phi_{l,i}$ is a (nonlinear) function of the MRS configuration[2] $\mathbf{q}$ and its derivative $\dot{\mathbf{q}}$.

[1] Note that an equality constraint can be readily expressed as two inequality constraints, i.e., the constraint $\phi_i = 0$ is equivalent to $\phi_i \le 0$ and $-\phi_i \le 0$. Therefore, the above equivalence will be implicitly used for the rest of the paper.

[2] In many practical applications, some restrictions are usually made on the MRS pose $\mathbf{p}$ and its derivative $\dot{\mathbf{p}}$. As the MRS pose $\mathbf{p}$ is, actually, a function of the MRS configuration $\mathbf{q}$, such constraints are included in the general setting (9).

For the solution later proposed in this work for geometric invariance, the functions $\phi_{l,i}$ need to be differentiable around the region given by $\phi_{l,i}(\mathbf{q},\dot{\mathbf{q}}) = 0$ and the partial derivatives $\partial\phi_{l,i}/\partial\dot{\mathbf{q}}$ around this boundary should not vanish.

Constraint qualification. The mandatory constraints $\phi_{1,i}$ included in the first priority level must define a 1st-level feasible set Φ_1 which must fulfill the following two conditions in order to allow a "safe" abort:

$$\exists [\mathbf{q}^T \ \mathbf{0}^T]^T \in \Phi_1 \tag{10}$$

$$[\mathbf{q}^T \ \dot{\mathbf{q}}^T]^T \in \Phi_1 \Rightarrow [\mathbf{q}^T \ \mathbf{0}^T]^T \in \Phi_1 \tag{11}$$

where $\mathbf{0}$ represents the zero column vector of suitable size. The meaning of such conditions is that (a) Φ_1 should be non-empty and (b) if a point of the configuration $[\mathbf{q}^T \ \dot{\mathbf{q}}^T]^T$ belongs to the feasible set Φ_1, the same position with zero speed also belongs to the feasible set. In this way, safe aborts can be achieved by decelerating before reaching a point where the hard constraints cannot be simultaneously satisfied (see Section 5.3 for details). Obviously, it is assumed that the starting state of the MRS belongs to Φ_1.

Objective. The goal of this work is to design a coordination supervisor (see Fig. 1) that is aware of the configuration $\mathbf{q}_i$ of each robot and that generates the commanded joint acceleration vector $\ddot{\mathbf{q}}_{ic}$ to be sent to the joint controllers of each robot, so that

- the *first-level* constraints are fulfilled;
- the actual coordination vector $\mathbf{v}$ of the MRS is as close as possible (via a weighted measure) to the user-input value $\mathbf{v}_{ref}$ in the *second-level* constraints;
- other third-level goals are achieved by using the remaining degrees of freedom of the MRS.

The coordination system to be developed is based on task-priority redundancy resolution, geometric invariance and sliding mode ideas.

The basic underlying idea in sliding-mode geometric invariance approach is the ability to instantaneously change the derivative of a particular scalar quantity related to a constraint. It will allow a simple computer implementation, in the line of other proposals by the authors [13,14], fulfilling the constraints specified by the robot end-user without recourse to high-level planning.

In partially unstructured environments, the inequality constraints corresponding to the collision avoidance with unforeseen obstacles cannot be known a priori. However, these constraints can be expressed in terms of the measurements given by the proximity sensors of the robots (e.g., infrared or ultrasonic sensors) in order to guarantee that the distance to the detected obstacles does not fall below a certain threshold. If the unknown obstacles do not change position, the constraint qualification still holds (i.e., quickly braking to zero speed is a feasible way of remaining in position bounds), the reactive setting in [25] may be adapted to the case here considered (details omitted for brevity).

Note that, in exchange for the simple implementation, however, complicated tasks might not be solvable with such an approach and, in certain cases, trap situations may occur [12] due to the absence of long-term planning. In that case, more sophisticated solutions in the literature for robot coordination [17–19] would be needed.

4. Background theory

The proposal in the next section of this work uses literature results from priority-based redundancy resolution and singular value decomposition, as well as geometric invariance via sliding

mode. Well-known ideas on these issues will be briefly reviewed next as preliminaries.

4.1. Task-priority based redundancy resolution

A classical approach to redundancy resolution in robotic systems consists in *augmenting* the task vector in order to tackle several (possibly incompatible) objectives simultaneously [26]. In this framework of multiple tasks, it is useful to consider the task-priority strategy, which consists of assigning a suitable order of priority to the given tasks. Thus, a lower-priority task is satisfied only by using the degrees of freedom in the null space of the higher-priority ones [21]. When an exact solution is not possible for a given task at a particular priority level, its error is minimized. The formulation for this approach is as follows. Let us consider the case of M tasks which consist of computing a command vector $\ddot{\mathbf{q}}_c$ (i.e., in this work, the commanded joint acceleration vector) while trying to fulfill the following acceleration equality constraints:

$$\mathbf{A}_i\ddot{\mathbf{q}}_c = \mathbf{b}_i, \quad i = 1, \ldots, M, \tag{12}$$

where matrix $\mathbf{A}_i$ and vector $\mathbf{b}_i$ of the ith task are assumed to be known, and the ordering given by index variable i reflects the priorities from highest priority ($i=1$) to lowest ($i=M$).

The solution $\ddot{\mathbf{q}}_{c,M}$ that hierarchically minimizes the error of equations in (12) is given by the following recursive formulation, proposed in [27]:

$$\begin{aligned}&\ddot{\mathbf{q}}_{c,i} = \ddot{\mathbf{q}}_{c,i-1} + (\mathbf{A}_i\mathbf{N}_{i-1})^{\dagger}\,(\mathbf{b}_i - \mathbf{A}_i\ddot{\mathbf{q}}_{c,i-1})\\ &\mathbf{N}_i = \mathbf{N}_{i-1}(\mathbf{I} - (\mathbf{A}_i\mathbf{N}_{i-1})^{\dagger}(\mathbf{A}_i\mathbf{N}_{i-1})), \quad i = 1, \ldots, M, \;\; \ddot{\mathbf{q}}_{c,0} = \mathbf{0}, \;\; \mathbf{N}_0 = \mathbf{I},\end{aligned} \tag{13}$$

where $\mathbf{I}$ and $\mathbf{0}$ denote the identity matrix and zero column vector, respectively, of suitable size, the superscript † denotes the Moore–Penrose pseudoinverse [28], and $\ddot{\mathbf{q}}_{c,i}$ and $\mathbf{N}_i$ are the solution vector and null-space projection matrix, respectively, for the set of first i tasks.

4.1.1. Regularization

Although the above solution does solve the required problem, a concern for it [28] is that excessively large values of $\ddot{\mathbf{q}}_{c,i}$ are obtained around the singularities of matrix $\mathbf{A}_i\mathbf{N}_{i-1}$, i.e., when its minimum singular value tends to zero.

In mandatory tasks it is indeed a serious problem due to ill-conditioning, requiring speed reduction or task aborts. However, if minor deviations can be allowed (for instance, in lower-level tasks), this drawback can be overcome using matrix regularization for $\mathbf{A}_i\mathbf{N}_{i-1}$ in the computation of $\ddot{\mathbf{q}}_{c,i}$ in (13). A classical type of regularization is the damped least-squares (DLS) solution [29] that consists of minimizing the square norm of the equation error together with the square norm of the solution weighted by a nonnegative damping factor λ. In particular, the regularized Moore–Penrose pseudoinverse of a matrix $\mathbf{H}$ using DLS results in

$$\mathbf{H}^{\sharp} = \mathbf{H}^{\mathrm{T}}(\mathbf{H}\mathbf{H}^{\mathrm{T}} + \lambda^2\mathbf{I})^{-1} = (\mathbf{H}^{\mathrm{T}}\mathbf{H} + \lambda^2\mathbf{I})^{-1}\mathbf{H}^{\mathrm{T}}, \tag{14}$$

where the superscript ♯ denotes the so-called regularized or singularity-robust pseudoinverse and it is computationally more efficient to use the first expression above if $\mathbf{H}$ is a matrix with more columns than rows and to use the second one otherwise. The value chosen for the damping factor can be a small constant [30] or other more sophisticated proposals in the literature can also be considered [31,32].

4.2. Geometric invariance via sliding mode conditioning

This section reviews the basic ideas of geometric invariance theory [11]. Such a theory has been successfully adapted by the authors to tackle some particular problems in single-robot configurations [12,13] and is also at the heart of the proposals in this work.

Consider the following dynamical system with n_x states and n_u inputs:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{d}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \tag{15}$$

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

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

$$\Phi = \{\mathbf{x} | \phi_i(\mathbf{x}) \le 0\}, \quad i = 1, \ldots, N \tag{16}$$

and ϕ_i are assumed to be defined in such a way that Φ is non-empty.

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). Later, the case of driving $\mathbf{x}$ as close as possible to a desired trajectory $\mathbf{x}_{ref}$ will be integrated into the approach.

To ensure the invariance of Φ, the control input $\mathbf{u}$ must guarantee that the right hand side of (15) points to the interior of Φ at all points in the boundary of Φ.

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

$$\begin{aligned}\frac{d(\phi_i(\mathbf{x}))}{dt} &= \nabla\phi_i^{\mathrm{T}}(\mathbf{x})\dot{\mathbf{x}} = \nabla\phi_i^{\mathrm{T}}(\mathbf{x})\mathbf{f}(\mathbf{x}, \mathbf{d}) + \nabla\phi_i^{\mathrm{T}}(\mathbf{x})\mathbf{g}(\mathbf{x})\mathbf{u}\\ &= L_f\phi_i(\mathbf{x}, \mathbf{d}) + \mathbf{L}_{\mathbf{g}}\phi_i(\mathbf{x})\mathbf{u} \le 0, \quad \forall\; i | \phi_i(\mathbf{x}) > 0,\end{aligned} \tag{17}$$

where ∇ denotes the gradient vector, the scalar $L_f\phi_i$ and the n_u-dimensional row vector $\mathbf{L}_{\mathbf{g}}\phi_i$ denote the Lie derivatives of $\phi_i(\mathbf{x})$ in the direction of vector field $\mathbf{f}$ and in the direction of the set of vector fields $\mathbf{g}$, respectively. Note that constraints such that $\phi_i(\mathbf{x}) > 0$ conform the set of *active* constraints. The number of active constraints will be denoted as b.

In general, any vector $\mathbf{u}$ pointing toward the interior of the allowed region can be used to satisfy (17), i.e., any vector $\mathbf{u}$ such that the scalar $\mathbf{L}_{\mathbf{g}}\phi_i\mathbf{u}$ is negative. This vector could be computed, for instance, solving a linear programming optimization problem for a fixed $\mathbf{x}$. Although such a direct solution may be plausible with high-capacity hardware, the goal of this work is presenting a simpler strategy, involving only gradient computation and a handful of matrix operations. In particular, we can make the set Φ invariant by means of the following variable structure control law:

$$\mathbf{u} = \begin{cases} \mathbf{0} & \text{if } \max_i\{\phi_i(\mathbf{x})\} \le 0\\ \mathbf{u}_c & \text{otherwise,}\end{cases} \tag{18}$$

where vector $\mathbf{u}_c$ is chosen to satisfy:

$$\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}\mathbf{u}_c = -\mathbf{1}_b u^+, \tag{19}$$

where matrix $\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}$ contains the row vectors $\mathbf{L}_{\mathbf{g}}\phi_i$ of all active inequality constraints (i.e., those constraints with $\phi_i > 0$), b is the number of active constraints, $\mathbf{1}_b$ is the b-dimensional column vector with all its components equal to one and u^+ is a positive constant to be chosen high enough to satisfy (17). In particular, one set of *sufficient*, but not necessary, conditions for making the set Φ invariant is that matrix $\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}$ is full row rank and that [13]:

$$u^+ > \sum_{i=1}^{b} (\max(L_f\phi_i, 0)). \tag{20}$$

As long as the state trajectory tries by itself to leave the allowed region Φ, the above control law (18) will make $\mathbf{u}$ to switch between $\mathbf{0}$ and $\mathbf{u}_c$ at a theoretically infinite frequency, which can be seen as an ideal sliding mode (SM) operation in the absence of open-loop phase (reaching mode) [9]. Therefore, sliding regimes are exploited here as a transitional mode of operation, contrarily to conventional sliding control where $\phi=0$ is made attractive and invariant. In fact, the above approach could be seen as a "one-side" sliding control: if we are at $\phi_i>0$, then we make $\phi_i=0$ attractive in finite-time just as sliding-mode control does, otherwise (i.e., when $\phi_i\le 0$) we disregard it.

Note that the two inequality constraints obtained from an original equality constraint actually give rise to the classical "permanent" sliding regime, i.e., we have a "two-sided" conventional sliding control where $\phi_i=0$ is made attractive and invariant. In this case, (19) will alternatively contain a row with different signs in $\mathbf{L}_\mathbf{g}\phi_i$. This fact can be interpreted as adding conditions $\mathbf{L}_\mathbf{g}\phi_i\mathbf{u}_c=-\text{sign}(\phi_i)u^+$ to (19) for the (always active) equality constraints. This is well-known in the conventional sliding-mode control literature to which the reader is referred for further details [10].

Once SM is established on the boundary of Φ by the control action $\mathbf{u}$, a continuous equivalent control [9] can be obtained, which is the control required to keep the system on the boundary of Φ. Consequently, the SM conditioning generated by (18) produces such a control action without explicit knowledge of it and with a low computational cost; this is a distinctive advantage of SM conditioning strategies [10].

Interested readers are referred to [9,10] for further details on conventional SM control theory, and to [11] for geometric-invariance based constrained control applications of SM reference conditioning, as well as the previously cited works by the authors [12,13].

4.2.1. Higher-order invariance

The above presented framework produces a non-smooth $\mathbf{u}$. If a smooth control action is wished, the following proposal can be applied.

First, the initial constraint $\phi(\mathbf{x})\le 0$ can be transformed to $\overline{\phi}=\phi(\mathbf{x})+K\dot{\phi}(\mathbf{x})\le 0$. Then, we have $\overline{\phi}=\phi(\mathbf{x})+K\nabla\phi\cdot\dot{\mathbf{x}}=\phi(\mathbf{x})+K\nabla\phi\cdot(\mathbf{f}(\mathbf{x},\mathbf{d})+\mathbf{g}(\mathbf{x})\mathbf{u})$. Hence, if we introduce an augmented state denoted as $\overline{\mathbf{x}}^\mathrm{T}=[\mathbf{x}^\mathrm{T}\ \mathbf{u}^\mathrm{T}]$, which includes the input $\mathbf{u}$, then $\overline{\phi}$ is actually a function of the augmented state, i.e., $\overline{\phi}(\overline{\mathbf{x}},\mathbf{d})$.

Now, taking time derivatives, we have $\dot{\overline{\phi}}=(\partial\overline{\phi}/\partial\overline{\mathbf{x}})^\mathrm{T}\dot{\overline{\mathbf{x}}}+(\partial\overline{\phi}/\partial\mathbf{d})^\mathrm{T}\dot{\mathbf{d}}$; hence, as $\dot{\mathbf{u}}$ appears in $\dot{\overline{\mathbf{x}}}$, then $\overline{\phi}$ is relative degree one in $\dot{\mathbf{u}}$, so considering $\dot{\mathbf{u}}$ as the "new" input, the actual control $\mathbf{u}$ will be now smooth.[3] Indeed, it is $\dot{\mathbf{u}}$ the variable which will now have a switching behavior if sliding-mode approaches are again used. Alternatively, if the model were precisely known, one could directly compute the range of $\mathbf{u}$ fulfilling $\overline{\phi}\le 0$ without differentiating $\overline{\phi}$.

When the constraint is active $\overline{\phi}=0$ entails exponential decrease in the original ϕ toward $\phi=0$ which is the original switching surface, i.e., $\phi(t)=\phi(0)e^{-K^{-1}t}$. Hence, the value of K^{-1} represents a sort of pole-assignment limiting the control action bandwidth. This idea is revisited in Section 5.4.1.

5. Proposal

5.1. Overview

We are interested in exploiting the task-priority based redundancy resolution and the geometric invariance setups described in the previous section to address the problem stated in Section 3.1.

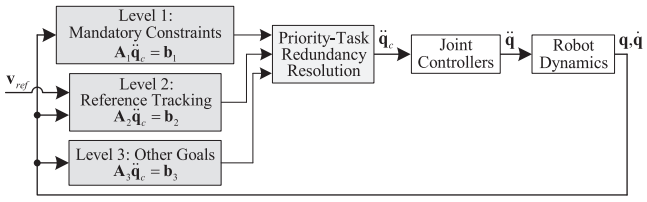

Fig. 2. Overview of the proposed approach.

It is important to remark that, in general, the number of priority levels and the constraints considered in each level can be freely chosen by the MRS end-user. In particular, three priority levels are considered in this section,[4] see Fig. 2.

The first level includes the *mandatory* constraints whose violation would result in collisions, invasion of forbidden space, breaking a transported object, etc. The second priority level includes the constraints related with the *tracking* of a reference trajectory by the MRS: deviations from the planned trajectory are allowed if such deviations are needed to enforce the high-priority constraints. Finally, the third priority level, which is the one with the lowest priority, is considered to achieve a *tertiary* goal (for instance, minimizing joint speeds, as later proposed) using the remaining degrees of freedom of the MRS, if any. Following the notation in (9), we will assume constraints are suitably grouped into $\phi_{1,i}$, $\phi_{2,i}$, $\phi_{3,i}$ for each of the three levels.

In an abstract setting, the input to the priority levels (Fig. 2) is the state $\{\mathbf{q},\dot{\mathbf{q}}\}$ of the MRS and each level gives an acceleration equality $\mathbf{A}_i\ddot{\mathbf{q}}_c=\mathbf{b}_i$ (12) whose square error must be minimized. These acceleration equalities are obtained using the geometric invariance theory presented Section 4.2, in order to fulfill the corresponding constraints. The commanded joint acceleration vector $\ddot{\mathbf{q}}_c$ is determined by the task-priority redundancy resolution block, where (13) is implemented, and serves as input to the joint controllers of the robots.

5.2. Dynamical system and Lie derivatives

In order to use the geometric invariance theory in Section 4.2, a dynamical system in the form of (15) is constructed with the state vector $\mathbf{x}=[\mathbf{q}^\mathrm{T}\ \dot{\mathbf{q}}^\mathrm{T}]^\mathrm{T}$, the disturbance vector $\mathbf{d}=\mathbf{d}_c$ and the input vector $\mathbf{u}=\ddot{\mathbf{q}}_c$. Therefore, the model is a double integrator and the following state equation is obtained from (5):

$$\dot{\mathbf{x}}=\begin{bmatrix}\mathbf{O}&\mathbf{I}\\ \mathbf{O}&\mathbf{O}\end{bmatrix}\mathbf{x}+\begin{bmatrix}\mathbf{0}\\ \mathbf{d}_c\end{bmatrix}+\begin{bmatrix}\mathbf{O}\\ \mathbf{I}\end{bmatrix}\mathbf{u}, \tag{21}$$

and, hence, in order to set up (17), the Lie derivatives of each of the constraint functions ϕ_i are given by

$$\mathbf{L}_\mathbf{g}\phi_i=\nabla\phi_i^\mathrm{T}\mathbf{g}=(\partial\phi_i/\partial\dot{\mathbf{q}})^\mathrm{T} \tag{22}$$

$$L_f\phi_i=\nabla\phi_i^\mathrm{T}\mathbf{f}=(\partial\phi_i/\partial\mathbf{q})^\mathrm{T}\dot{\mathbf{q}}+(\partial\phi_i/\partial\dot{\mathbf{q}})^\mathrm{T}\mathbf{d}_c. \tag{23}$$

5.3. Level 1: mandatory constraints

As previously mentioned, the first level includes the hard inequality constraints mentioned in Section 2 (collision avoidance, joint limits, etc.), as well as those so considered in the specification of the coordinated task, for instance keeping fixed distance and relative orientation between effectors in order to avoid breaking a transported object (rigid-body constraints). To satisfy these mandatory constraints, the SM conditioning of Section 4.2 is

[3] The assumption that $\dot{\mathbf{d}}$ is bounded is also needed as $\dot{\mathbf{d}}$ appears in $\dot{\overline{\phi}}$. The assumption is fulfilled, for instance, if $\mathbf{d}$ has finite bandwidth.

[4] Although a specific 3-level setup is considered in this section, other different setups can be easily defined, adapting the guidelines presented here.

considered with the dynamical system and Lie derivatives (22) and (23), the constraint functions ϕ_i being those of the mandatory constraints $\phi_{1,i}$.

Note that many mandatory constraints depend only on the MRS configuration $\mathbf{q}$, i.e., $\sigma_{1,i}(\mathbf{q}) \leq 0$. In order to achieve smooth speeds and using acceleration as control variable, such constraints will be modified as proposed in Section 4.2.1, in order for the sliding manifold to have relative degree one with respect to $\ddot{\mathbf{q}}$, that is

$$\phi_{1,i}(\mathbf{q},\dot{\mathbf{q}}) = \sigma_{1,i}(\mathbf{q}) + K_{1,i}\frac{d\sigma_{1,i}(\mathbf{q})}{dt} = \sigma_{1,i} + K_{1,i}\nabla\sigma_{1,i}^{\mathrm{T}}\dot{\mathbf{q}} \leq 0, \tag{24}$$

where $K_{1,i}$ is an arbitrary strictly positive parameter that determines the rate of approach to the boundary of the original constraint $\sigma_{1,i}(\mathbf{q}) \leq 0$.

From (5), it follows that the sliding manifold has relative degree one with respect to the discontinuous action $\mathbf{u}$, as required by SM theory [9], since $\dot{\phi}_{1,i}$ (and $\ddot{\mathbf{q}}$) explicitly depends on signal $\ddot{\mathbf{q}}_c$.

The partial derivatives of $\phi_{1,i}$ for the original mandatory constraints $\sigma_{1,i}$ depending only on the MRS configuration $\mathbf{q}$, see (24), are given by

$$(\partial\phi_{1,i}/\partial\mathbf{q})^{\mathrm{T}} = \nabla\sigma_{1,i}^{\mathrm{T}} + K_{1,i}\dot{\mathbf{q}}^{\mathrm{T}}\mathbf{H}_{\sigma i} \tag{25}$$

$$(\partial\phi_{1,i}/\partial\dot{\mathbf{q}})^{\mathrm{T}} = K_{1,i}\nabla\sigma_i^{\mathrm{T}}, \tag{26}$$

where $\mathbf{H}_{\sigma 1,i}$ denotes the Hessian matrix of second-order partial derivatives of $\sigma_{1,i}$.

Of course, if there were constraints in this level which directly depended on joint speeds, they would abide to the general case (22) and (23) without the need of higher-order-invariance modifications.

Eq. (19), for the first priority level results in

$$\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}_1\ddot{\mathbf{q}}_c = -\mathbf{1}_b u_1^+, \tag{27}$$

where matrix $\mathbf{L}_{\mathbf{g}}\phi_1$ contains the row vectors $\mathbf{L}_{\mathbf{g}}\phi_{1,i}$, see (22), of all active mandatory constraints and u_1^+ is the value of u^+ chosen for the first priority level.

Note that, if all these constraints originally depended only on the MRS configuration $\mathbf{q}$, from (22) and (26) matrix $\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}_1$ could be written as

$$\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}_1 = \mathbf{K}_1\nabla\boldsymbol{\sigma}_1^{\mathrm{T}}, \tag{28}$$

where $\mathbf{K}_1$ is a diagonal matrix with diagonal entries $K_{1,i}$ and matrix $\nabla\boldsymbol{\sigma}_1$ contains the gradient vectors $\nabla\sigma_{1,i}$ of all active mandatory constraints.

5.3.1. Unfulfillment of the constraints

If the solution given by the task-priority redundancy resolution (13) does not satisfy the mandatory constraints the robots must be stopped to avoid violating the constraints.

This situation may occur due to two causes: either matrix $\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}_1$ is not full row rank (e.g., this situation arises when the number of active constraints is greater than the total number of joints n), or u_1^+ has not been chosen high enough (see Section 5.6).

Note that (24) fulfills the constraint qualification conditions (10) and (11). Then, when the above situation arises, we know that $\sigma_{1,i} < 0$ (indeed, a first-order system $\sigma + K\dot{\sigma} \leq 0$ with initial condition $\sigma(0) < 0$ will verify $\sigma(t) \leq \sigma(0)e^{-K^{-1}t} < 0$ for all $t \geq 0$; hence, while the system had solution for $\ddot{\mathbf{q}}_c$ we can ensure that $\sigma_{1,i} < 0$ and the speed is also bounded $\dot{\sigma} \leq -\sigma/K$). So, at the moment (27) ceases to have a solution we have some time to carry out a brake maneuver (in a fast enough way) before reaching the position limits $\sigma_{1,i} = 0$, further helped by the fact that the closer we are to the limit, the slower the speed will be.

An option to reduce the robot speed in a "controlled" way is slowing down the reference trajectory; this is later addressed in Section 5.7.

5.4. Level 2: reference tracking

The second priority level includes the equality constraint (6) related with the tracking of a time-varying reference trajectory by the coordination vector $\mathbf{v}$ of the MRS. This equality constraint can be rewritten as

$$\boldsymbol{\sigma}_2(\mathbf{q}) = \mathbf{v}_{ref} - \mathbf{v}(\mathbf{p}) = \mathbf{e} = \mathbf{0}, \tag{29}$$

where $\mathbf{e}$ represents the tracking position error of the coordination vector.

As before, the above constraint will be modified via higher-order invariance (Section 4.2.1) in order for the sliding manifold to have relative degree one with respect to $\ddot{\mathbf{q}}$, that is

$$\boldsymbol{\phi}_2(\mathbf{q},\dot{\mathbf{q}}) = \boldsymbol{\sigma}_2 + K_{2,1}\frac{d\boldsymbol{\sigma}_2}{dt} = \mathbf{e} + K_{2,1}\dot{\mathbf{e}} = 0, \tag{30}$$

where $K_{2,1}$ is a positive parameter representing the desired time constant for correcting deviations from the reference trajectory (if level 1 constraints do not hinder it).

From (30) and (29), the time derivative of the vector constraint function $\boldsymbol{\phi}_2$ results in

$$\dot{\boldsymbol{\phi}}_2 = \dot{\mathbf{e}} + K_{2,1}\ddot{\mathbf{e}} = \dot{\mathbf{v}}_{ref} - \dot{\mathbf{v}} + K_{2,1}(\ddot{\mathbf{v}}_{ref} - \ddot{\mathbf{v}}). \tag{31}$$

Substituting (7) and (8) into Eq. (31) gives

$$\dot{\boldsymbol{\phi}}_2 = \dot{\mathbf{v}}_{ref} - \mathbf{J}_{pq}\dot{\mathbf{q}} + K_{2,1}(\ddot{\mathbf{v}}_{ref} - \mathbf{J}_{pq}\ddot{\mathbf{q}} - \dot{\mathbf{J}}_{pq}\dot{\mathbf{q}}). \tag{32}$$

So, from (32) and (5), the Lie derivatives of $\boldsymbol{\phi}_2$ result in

$$\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}_2 = -K_{2,1}\mathbf{J}_{pq} \tag{33}$$

$$L_f\boldsymbol{\phi}_2 = (\dot{\mathbf{v}}_{ref} + K_{2,1}\ddot{\mathbf{v}}_{ref}) - (\mathbf{J}_{pq} + K_{2,1}\dot{\mathbf{J}}_{pq})\dot{\mathbf{q}} - K_{2,1}\mathbf{J}_{pq}\mathbf{d}_c \tag{34}$$

Therefore, Eq. (19) for the second priority level results in

$$\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}_2\ddot{\mathbf{q}}_c = -\mathrm{sign}(\boldsymbol{\phi}_2)u_2^+, \tag{35}$$

where u_2^+ is the value of u^+ chosen for the second priority level and $\mathrm{sign}(\cdot)$ is the sign function discussed in Section 4.2 for equality constraints, i.e., the tracking of the reference trajectory using (35) is equivalent to the conventional sliding-mode control with switching command acceleration $\ddot{\mathbf{q}}_c$ (with, of course, no need for computing (34) due to the inherent robustness of the sliding-mode approach).

5.4.1. Smooth acceleration options

If a smooth acceleration were wished, the higher-order invariance[5] mentioned in Section 4.2.1 can be used more than one time, to obtain the new sliding manifold:

$$\begin{aligned}\overline{\boldsymbol{\phi}}_2(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) &= \boldsymbol{\phi}_2 + K_{2,2}\frac{d\boldsymbol{\phi}_2}{dt} = \mathbf{e} + K_{2,1}\dot{\mathbf{e}} + K_{2,2}(\dot{\mathbf{e}} + K_{2,1}\ddot{\mathbf{e}}) \\ &= \mathbf{e} + (K_{2,1} + K_{2,2})\dot{\mathbf{e}} + K_{2,1}K_{2,2}\ddot{\mathbf{e}} = 0,\end{aligned} \tag{36}$$

where $K_{2,2}$ determines the time constant of the approach to $\boldsymbol{\phi}_2 = 0$. Equivalently, scaling $\overline{\boldsymbol{\phi}}_2$, we obtain the sliding manifold as

$$K_{2,p}\mathbf{e} + K_{2,v}\dot{\mathbf{e}} + \ddot{\mathbf{e}} = 0, \tag{37}$$

with $K_{2,p} = (K_{2,1}K_{2,2})^{-1}$ and $K_{2,v} = (K_{2,1}^{-1} + K_{2,2}^{-1})$. If sliding-mode approaches are again used (arising from the expression of the time-derivative of $\overline{\boldsymbol{\phi}}_2$) the new control input $\dddot{\mathbf{q}}_c$ (i.e., the commanded jerk vector) will have a switching behavior but acceleration $\ddot{\mathbf{q}}_c$ will not.

Indeed, as an alternative to the previous jerk approach, the smooth commanded acceleration vector $\ddot{\mathbf{q}}_c$ could be directly computed from solving (37) without differentiating $\overline{\boldsymbol{\phi}}_2$.

[5] Note that this approach requires that the other priority levels consider also the jerk as the control input instead of the acceleration, using higher-order invariance, too.

In particular, substituting (29), (8) and (5) into Eq. (37), gives

$$K_{2,p}\mathbf{e}+K_{2,v}\dot{\mathbf{e}}+\ddot{\mathbf{v}}_{ref}-\ddot{\mathbf{v}}=K_{2,p}\mathbf{e}+K_{2,v}\dot{\mathbf{e}}+\ddot{\mathbf{v}}_{ref}-\mathbf{J}_{pq}\ddot{\mathbf{q}}-\dot{\mathbf{J}}_{pq}\dot{\mathbf{q}}$$
$$=K_{2,p}\mathbf{e}+K_{2,v}\dot{\mathbf{e}}+\ddot{\mathbf{v}}_{ref}-\mathbf{J}_{pq}(\ddot{\mathbf{q}}_c+\mathbf{d}_c)-\dot{\mathbf{J}}_{pq}\dot{\mathbf{q}}=0. \quad (38)$$

Solving $\mathbf{J}_{pq}\ddot{\mathbf{q}}_c$ from (38) results in

$$\mathbf{W}_2\mathbf{J}_{pq}\ddot{\mathbf{q}}_c=\mathbf{W}_2(\ddot{\mathbf{v}}_{ref}+K_{2,p}\mathbf{e}+K_{2,v}\dot{\mathbf{e}})-\mathbf{W}_2(\mathbf{J}_{pq}\mathbf{d}_c+\dot{\mathbf{J}}_{pq}\dot{\mathbf{q}}), \quad (39)$$

where $\mathbf{W}_2$ is an arbitrary diagonal weighting matrix.

Remark. Note that the acceleration equality (39) is equivalent to the classical operational space robot control [33]. Moreover, the commanded acceleration $\ddot{\mathbf{v}}_c$ in (39) for the coordination vector (i.e., $\ddot{\mathbf{v}}_c=\ddot{\mathbf{v}}_{ref}+K_{2,p}\mathbf{e}+K_{2,v}\dot{\mathbf{e}}$) represents a classical kinematic controller utilized for robotic trajectory tracking [34] (i.e., a correction based on the position and velocity errors plus a feedforward of the second-order time derivative of the reference) and introduces the poles given by the roots of the polynomial with coefficients $[1\ K_{2,v}\ K_{2,p}]$. The main drawback of the equation-solving approach is that all the terms in (39) should be known (i.e., the system model). For example, if the disturbance $\mathbf{d}_c$ is not known a priori, as common in practice, the accurate computation of $\ddot{\mathbf{q}}_c$ with (39) is not possible, whereas the computation of $\ddot{\mathbf{q}}_c$ with (35) is *robust* [9] against $\mathbf{d}_c$ since it is collinear with the discontinuous control action, see (5).

5.5. Level 3: other goals

The third priority level is used to achieve a secondary goal by means of the remaining degrees of freedom of the MRS, if any. Among the different available options in the literature, in this work we have chosen to stop moving unnecessary joints, i.e., trying to fulfill level-3 constraints $\phi_{3,i}(q,\dot{q})$ defined as $\dot{\mathbf{q}}=0$. As in the above case, this can be achieved in three ways: (a) sliding-mode settings in acceleration, or for smooth accelerations, (b) choosing sliding-mode in jerk, or (c) solving the equality

$$\ddot{\mathbf{q}}_c=-K_{3,red}\dot{\mathbf{q}}, \quad (40)$$

where $K_{3,red}$ is the desired time constant for the joint speed reduction.

5.6. Chattering

In theory, sliding regimes are produced by infinite-frequency commutation. However, as widely known, the finite-frequency commutation of any practical SM implementation makes the system to leave the ideal SM and oscillate with finite frequency and amplitude inside a "band" around $\boldsymbol{\phi}=\mathbf{0}$, which is called *chattering* [9]. Similar to other robotic applications based on SM conditioning [12–14], an upper bound for the chattering band $\Delta\boldsymbol{\phi}$ of the proposed approach can be derived using the Euler-integration of the discontinuous action given by (19), that is

$$\Delta\boldsymbol{\phi}=T_s|\mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}\mathbf{u}_c|=T_s u^+\mathbf{1}_b, \quad (41)$$

where T_s is the sampling period of the robotic system and the value of u^+ is u_1^+ for the first priority level, u_2^+ for the second priority level, etc. This chattering amplitude must be less than the error allowed in the constraint inequality fulfillment. For instance, this allowable error could be given by the looseness of the joints, the mechanical rigidity between the robot gripper and the object being held, the security margin used in the definition of the constraints for collision avoidance, etc. The value of u^+ can be chosen in a conservative manner setting it to a "big" number to ensure that it is greater than the lower bound given by (20) and (23), as usual in SM applications. However, from (41), such big numbers may induce unnecessary chattering amplitude, so it is a design compromise. Also, there is a relationship between the choice of u^+ and the speed at which a particular trajectory can be followed. This issue is discussed next.

5.7. Reference speed auto-regulation (optional)

The above sections propose a complete multi-level sliding-mode based robot coordination. Anyway, as outlined in Section 3, if the reference trajectory is expressed as a function of a parameter $s(t)$, i.e., $\mathbf{v}_{ref}(s)$, the above scheme can be *optionally* refined in order to regulate $\dot{s}$ (speed of task execution) in order to better fulfill speed-related constraints with reduced error: based on the constraint qualification (11) slow movement will help entering the feasible zone, hence, if speed of reference can be reduced there is greater opportunity for mandatory constraints to be fulfilled with lower errors and avoiding emergency stop procedures (Section 5.3.1), reducing the chattering amplitude, too. In order to accomplish the above goal, the procedures in [24] are adapted in the context of the present problem below.

First, note that reducing the value of the joint speeds $\dot{\mathbf{q}}$, the term $(\partial\phi_i/\partial\mathbf{q})^{\mathrm{T}}\dot{\mathbf{q}}$ in (23) gets smaller. Hence, from (20), if disturbances are not large, this will allow using a smaller value of u^+, beneficial for reducing chattering while keeping invariance conditions.

Note now that, if tracking error and disturbances are small and (39) is fulfilled, then from equation (37) the velocity error $\dot{\mathbf{e}}$ converges to zero exponentially with time constant $K_{2,v}^{-1}$, i.e., $\dot{\mathbf{v}}$ converges to $\dot{\mathbf{v}}_{ref}$, and $\dot{\mathbf{v}}$ is proportional (via suitable Jacobians) to $\dot{\mathbf{q}}$.

In summary, under the above conditions $\dot{\mathbf{q}}$ is approximately proportional to $\dot{\mathbf{v}}_{ref}$ at a particular position $\mathbf{q}(s)$. As $\dot{\mathbf{v}}_{ref}$ is, too, proportional to $\dot{s}$, this motivates the idea of slowing down the advance of s (i.e., reducing $\dot{s}$ and, henceforth $\dot{\mathbf{q}}$) in order to avoid significant unfulfillment of mandatory constraints without needing to increase u_1^+.

In numerical implementations, checking that the value of u_1^+ is not large enough can be carried out by detecting that a particular constraint is repeatedly unfulfilled in consecutive time steps because, for active constraints, (27) entails that the sign of ϕ_i must immediately switch from positive to negative in one sample.

If such a situation is detected, it means that the joint speeds (i.e., $\dot{s}$) must be reduced in order to decrease the minimum value of u_1^+ required for invariance. In order to recover and accelerate again, if such a repeated unfulfillment situation does not occur, it means that there is room for increasing $\dot{s}$ again (up to a reasonable maximum $\dot{s}_{\max}$), as u_1^+ is higher than needed.

Therefore, the following equations governing $\dot{s}$ are proposed, with design parameters $\dot{s}_{\max}$ and τ_{AR} indicating, respectively, the maximum speed of the tracking task and the time allowed for completely braking to zero speed (or accelerating to $\dot{s}_{\max}$):

$$u_{AR}=\begin{cases}-1 & \text{if } \exists i|(\phi_{1,i}(k)>0) \text{ and } (\phi_{1,i}(k-1)>0)\\ 1 & \text{otherwise}\end{cases} \quad (42)$$

$$f_{AR}=\tau_{AR}^{-1}\int u_{AR} \quad \text{with } f_{AR}\in[0,1] \quad (43)$$

$$\dot{s}=f_{AR}\dot{s}_{\max}. \quad (44)$$

In the above equations, u_{AR} is a signal indicating whether the speed should be increased or decreased depending on the detection of repeated unfulfillment, f_{AR} is the accumulated[6] effect of u_{AR}, saturated at zero (full stop) and one (full speed $\dot{s}_{\max}$), used in (44) as a scale factor for the motion rate parameter.

[6] Obviously, integral in (43) is actually implemented as summation of $u_{AR}T_s$.

Note that in normal operation (when the optimal speed is in intermediate values) u_{AR} will be a high-frequency switching signal between -1 and 1 whose average value (integral has a low-pass effect) will transparently regulate $\dot{s}$ to achieve the commanded task without violation of mandatory constraints. In this way, the speed autoregulation integrates nicely in the overall sliding-mode control framework.

Note that, however, as $\dot{\mathbf{q}}$ depends, too, on disturbance signals and possible nonzero initial error, reference speed autoregulation does not preclude a situation in which emergency stops might be required due to these situations.

6. Additional remarks

Some remarks on the proposed method are given below.

6.1. Advantages and disadvantages of the proposed approach

Main advantages of the proposed method:

- Only requires a few program lines (see Appendix A) and has *reduced computation time* since only linear algebra is used.
- Simplifies the *user interface* since the method directly deals with the fulfillment of the constraints specified by the robot end-user.
- Uses *partial information* of the system model, i.e., the Lie derivatives $L_f\phi_i$ (23) are not needed, only the Lie derivatives $\mathbf{L_g}\phi_i$ (22) are required. Therefore, only first order derivatives (gradient vectors, Jacobian matrices, etc.) are needed, see (26) and (33), and no second-order derivatives (Hessian matrices, derivative of Jacobians, etc.) are required, see (25) and (34).

Let us now comment on the possible limitations:

- The SM algorithm used in this work uses linear extrapolation to predict the value of the constraint functions at the next step. This implies that only local data of first-order derivatives are used. Therefore, since higher-order derivatives are ignored, the algorithm may be blocked in *trap situations*. In some cases, these situations could be avoided using a planner with the complete geometric data of the problem in order to "simulate" for a large prediction horizon. However, the complexity of this planner and its computational cost are substantially greater than those of the algorithms proposed in this work, see Appendix A.
- Similar to other SM control applications, the proposed method suffers from the *chattering* problem, see Section 5.6. However, this drawback becomes negligible for reasonable fast sampling rates, see (41).

6.2. Guidelines for designing the algorithm parameters

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

Amplitude of the control action. The value of u^+ has to be as close as possible to its lower bound given by (20) (with, perhaps, some safety margin) in order to have reduced chattering amplitude and high chattering frequency, see Section 5.6.

Sampling period. The sampling period T_s has to be small enough in order to have small chattering amplitude (41). The minimum value for the sampling period is determined by the computation time of one iteration of the proposed algorithm, which is around 5 μs for the case study in Section 8 (see Appendix A).

6.3. Differentiability of the constraint functions

As stated in Section 3.1, the constraint functions $\phi_{l,i}$ must be differentiable. If this assumption is not satisfied at a certain time, the SM behavior of the proposed approach (Section 4.2) is temporarily lost and the constraints may be unfulfilled.

6.4. Moving constraints

The proposed approach can also be used if there are moving constraints, e.g., moving obstacles with *known* trajectories. In this case ϕ_i also depends explicitly on time and, hence, the derivative of ϕ_i in Eq. (17) 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 $L_f\phi_i + \partial\phi_i/\partial t$, and $\mathbf{L_g}\phi_i$ and $L_f\phi_i$ are given again by (22) and (23), respectively. 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 proposed algorithm remains the same and a high-enough constant u^+ will suffice for practical implementation.

7. Simulation: first example

Although the proposed approach for robot coordination can be applied to any MRS, in this section a simple two-dimensional example is considered for better illustration of the main features of the algorithm. The simulation results presented in this section were obtained using MATLAB®. A real-time animation of the resulting movement for this first simulation example can be visualized at http://politube.upv.es/play.php?vid=58151.

Details of pseudo-code and computing time for actual implementation of the proposed strategy appear in Appendix A at the end of the paper.

7.1. Description of the 3R planar double robot setup for the first example

In the proposed first simulation example, coordination between two open-chain planar mechanisms composed of four links (the first of them is fixed) connected serially by three revolute joints, i.e., two 3R planar robots, is considered. The end-effector of each robot is used to grasp the end of a rigid bar and, therefore, both robots must be properly coordinated to carry the bar.

Fig. 3 depicts the MRS in consideration, as well as the notation for the different coordinates of the robots and the transported object. In the figure, L_i is the length of the ith moving link and L_{sep} is the distance between the first joint of both robots, and $[x_i\ y_i]^T$ denote the cartesian planar position of the point located at the end of the ith moving link ($i=1,2,3$ for the first 3R robot and $i=4,5,6$ for the second 3R robot).

The origin of the reference frame has been located at the midpoint between the first joint of both robots and the X-axis is aligned along the line joining these joints, see Fig. 3. The bar angle with respect to the horizontal axis is denoted as θ_V.

It will be assumed that the transported bar can freely rotate around the grippers located at both ends of the bar, i.e., there is an extra revolute joint between the end-effectors of each robot and the bar.

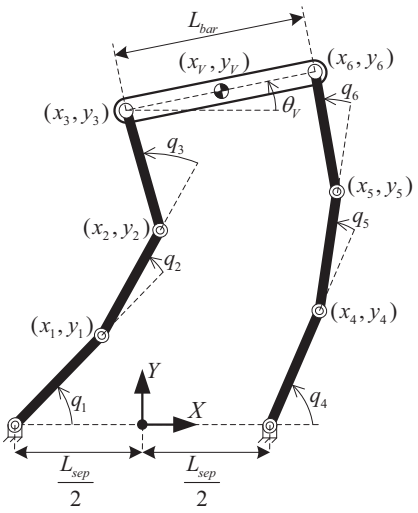

Fig. 3. Multi-robot system used for 2D simulation: two 3R planar robots carrying a bar.

From the referred figure and reference frame, the positions $[x_i\ y_i]^T$ are given by the equations:

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} -(L_{sep}/2) + \sum_{j=1}^{i} L_j \cos(q_1 + \cdots + q_j) \\ \sum_{j=1}^{i} L_j \sin(q_1 + \cdots + q_j) \end{bmatrix}, \quad i = 1, 2, 3, \tag{45}$$

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} (L_{sep}/2) + \sum_{j=4}^{i} L_j \cos(q_4 + \cdots + q_j) \\ \sum_{j=4}^{i} L_j \sin(q_4 + \cdots + q_j) \end{bmatrix}, \quad i = 4, 5, 6. \tag{46}$$

The pose vector considered for both 3R robots is the cartesian position of their end-effectors, that is

$$\mathbf{p}_1 = \mathbf{l}_1(\mathbf{q}_1) = \mathbf{l}_1(q_1, q_2, q_3) = [x_3\ \ y_3]^T \tag{47}$$

$$\mathbf{p}_2 = \mathbf{l}_2(\mathbf{q}_2) = \mathbf{l}_2(q_4, q_5, q_6) = [x_6\ \ y_6]^T. \tag{48}$$

Furthermore, three elements are considered for the coordination vector $\mathbf{v}$ of the MRS: the cartesian coordinates (x_V, y_V) of the midpoint of the bar (namely, the tracking point, i.e., the point that tracks the reference) and the bar angle θ_V. Therefore, the 3-element coordination vector of the MRS is given by

$$\mathbf{v}(\mathbf{p}) = \mathbf{v}(x_3, y_3, x_6, y_6) = \begin{bmatrix} x_V \\ y_V \\ \theta_V \end{bmatrix} = \begin{bmatrix} (x_3 + x_6)/2 \\ (y_3 + y_6)/2 \\ \arctan\left(\frac{y_6 - y_3}{x_6 - x_3}\right) \end{bmatrix}. \tag{49}$$

7.2. Constraints for the first example

As previously discussed, level 3 constraints have been considered to be $\dot{\mathbf{q}} = 0$. The constraints arising in the two upper levels are discussed next.

7.2.1. Mandatory constraints (level 1) for the first example

Three sets of constraints are considered in the first priority level, i.e., that with the highest priority.

First, the rigid body constraint associated with the bar carried by the robots must be fulfilled at all times. This equality constraint is given by

$$\sigma_{1,R} = L_{bar}^2 - ((x_6 - x_3)^2 + (y_6 - y_3)^2) = 0, \tag{50}$$

where L_{bar} is the length of the bar.

Second, it will be considered that the boundary of the allowed workspace is given by two perpendicular straight lines. Such lines will depart from the point $(x_{\max}, y_{\min})$ and will be parallel to the reference frame axis, i.e., $x_{\max}$ and $y_{\min}$ are the maximum and minimum allowed values for the x- and y-coordinates, respectively.

Since the allowed workspace is convex, assuming for simplicity that the width of the robot links and carried bar is negligible, the following inequality constraints must be fulfilled to guarantee that every point of the 3R robots and carried bar is inside the allowed workspace:

$$\sigma_{1,i} = x_i - x_{\max} \le 0, \quad i = 1, \ldots, 6, \tag{51}$$

$$\sigma_{1,i} = -y_{i-6} + y_{\min} \le 0, \quad i = 7, \ldots, 12. \tag{52}$$

As a last mandatory constraint, the following inequality is also considered for the angle of the bar carried by the robots:

$$\sigma_{1,\theta} = |\theta_V| - \theta_{\max} \le 0, \tag{53}$$

where $\theta_{\max}$ is the maximum allowed value for the bar angle deviation from the horizontal direction.

7.2.2. Reference tracking (level 2) for the first example

For the simulations, the reference path for the bar's center is given by the following expression:

$$\mathbf{v}_{ref}(s) = \begin{bmatrix} x_{V,ref}(s) \\ y_{V,ref}(s) \\ \theta_{V,ref}(s) \end{bmatrix} = \begin{bmatrix} 2\cos(s + \pi/4) - 2\sin^2(s + \pi/4) \\ 2.6\sin(s + \pi/4) \\ 0 \end{bmatrix}, \tag{54}$$

with $s = 0 \ldots (2\pi - 0.1)$. The resulting reference trajectory appears in green lines in Fig. 6 to be later discussed.

7.3. Simulation conditions and parameter values for the first example

Simulation was run under the following conditions:

(i) The length L_i of the robot links, the length L_{bar} of the bar, and the distance L_{sep} between the first joint of both robots were set to 1.

(ii) The acceleration equalities used for the first, second and third priority levels are (27), (39) and (40), respectively. Therefore, by comparing these acceleration equalities with Eq. (12), it is clear that $\mathbf{A}_1 = \mathbf{L}_{\mathbf{g}}\boldsymbol{\phi}_1$, $\mathbf{b}_1 = -\mathbf{1}_b u_1^+$, $\mathbf{A}_2 = \mathbf{W}_2\mathbf{J}_{pq}$, $\mathbf{b}_2 = \mathbf{W}_2(\ddot{\mathbf{v}}_c - \dot{\mathbf{J}}_{pq}\dot{\mathbf{q}})$ (the disturbance vector $\mathbf{d}_c$ has been considered zero), $\mathbf{A}_3 = \mathbf{I}$ and $\mathbf{b}_3 = -K_{red}\dot{\mathbf{q}}$.

(iii) For the mandatory constraints in the first priority level, the following parameter values were used: an amplitude u_1^+ of 10 for the switching law; a constraint approaching parameter $K_{1,i}$ of 0.1 s for all constraint functions; and the parameter values $x_{\max} = 2.2$, $y_{\min} = -2.2$ and $\theta_{\max} = 0.1$ rad.

(iv) For the reference tracking in the second priority level, the following parameter values were used: a diagonal weighting matrix $\mathbf{W}_2 = \mathrm{diag}([1\ 1\ 0.1])$, a correction gain $K_{2,p}$ of 400 s^{-2} and a correction gain $K_{2,v}$ of 40 s^{-1}.

(v) For the joint speed reduction in the third priority level, a reduction gain $K_{3,red}$ of 50 s^{-1} was used.

(vi) The speed auto-regulation algorithm described in Section 5.7 was implemented using a maximum speed $\dot{s}_{max}$ for the tracking task of 1 s^{-1} and an integration time constant τ_{AR} of 0.05 s.

(vii) The commanded joint acceleration vector $\ddot{\mathbf{q}}_c$ was computed with a sampling period T_s of 0.5 ms, using (13) but replacing the Moore–Penrose pseudoinverse by the singularity-robust pseudoinverse (14). A constant damping factor $\lambda = 0.01$ was used.

(viii) We considered the initial MRS configuration vector $\mathbf{q}(0) = [\pi/2\ 0.4930\ -1.9056\ \pi/2\ -1.4126\ 1.9056]^{\mathrm{T}}$ rad, yielding an initial robot position error vector $\mathbf{e}(0) = [-0.1\ -\ 0.2\ 0]^{\mathrm{T}}$.

7.4. Simulation results for the first example

The results of the simulation are depicted at different figures: Fig. 4 and 5 show the simulated behavior of the global system regarding tracking errors and constraints; 12 snapshot frames of the robot configuration during the reference tracking at different time instants are shown in Fig. 6.

Fig. 4 shows how initial error is corrected with the dynamics (37), and how deviations from the reference trajectory are forced by hitting level-1 constraints in later instants; the mandatory angle limitation is hit at the time interval 7–9 s.

Fig. 5 depicts the detail of different parameters of the proposed strategy (speed autoregulation, active constraints). The figure shows that

- the minimum-amplitude auto-regulation algorithm is active (i.e., $f_{AR} < 1$) during most of the simulation (see two upper plots in the figure and actual joint speeds in the third plot);
- all mandatory constraints are fulfilled, i.e., $\max(\phi_{1,i}) \le 0$ (fourth plot) – note that the bar length constraint is an equality, hence, always equal to zero so it has been omitted from the plots;
- in some phases of the simulation there are up to three active constraints at a time (bottom plot).

For further clarity, details on each of the presented frames from the achieved movements in Fig. 6 are discussed below:

- first and second frames (t=0 and t=0.35, respectively): the initial position error is made zero;
- third and fourth frames (t=0.7 and t=1.7, respectively): tracking error arises because the robot end-effectors reach the mechanical boundary of their workspace (i.e., the robots are completely extended);
- fifth and sixth frames (t=2.3 and t=3, respectively): the tracking error is made zero again;
- seventh frame (t=3.7): tracking error arises because the end-effectors reach again their fully stretched mechanical limits;
- eighth frame (t=5.45): the workspace constraint on the y-coordinate becomes simultaneously active for both end-effectors;
- ninth frame (t=6.3): the tracking error is zero again;

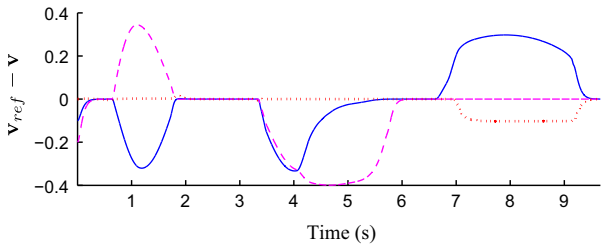

Fig. 4. Position tracking error $\mathbf{e} = \mathbf{v}_{ref} - \mathbf{v}$: e_x (solid, blue), e_y (dashed, magenta) and e_θ (dotted, red). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

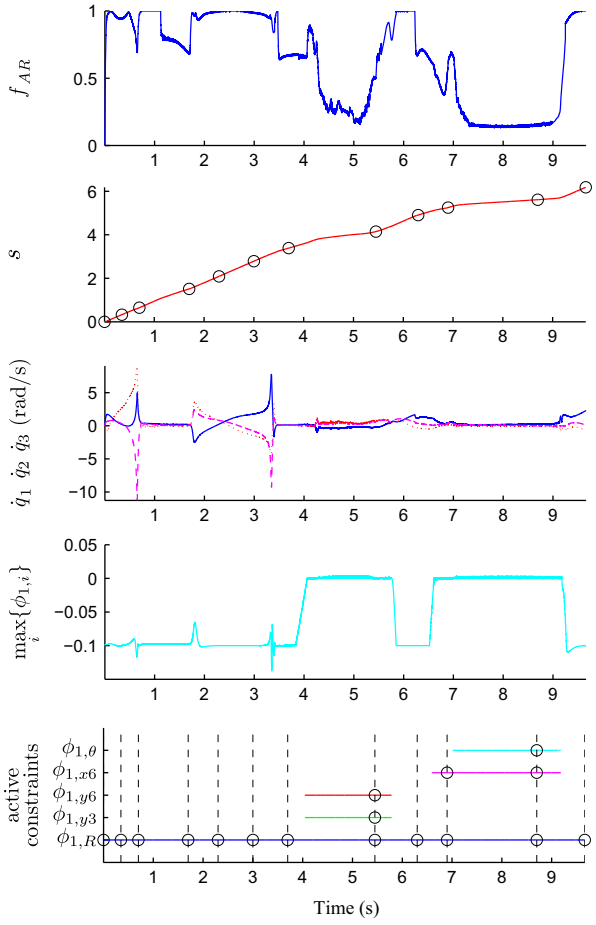

Fig. 5. From top to bottom plots: (1) Scale factor f_{AR} of the speed auto-regulation algorithm; (2) motion parameter s (circles correspond to time instants of frames in Fig. 6); (3) joint speeds $\{\dot{q}_1, \dot{q}_2, \dot{q}_3\}$ for the first 3R robot (those for the second robot are omitted for brevity); (4) maximum value of the mandatory constraint functions $\phi_{1,i}$ (excluding the equality constraint function $\phi_{1,R}$, always zero); and (5) horizontal lines indicating when a constraint is active (the dashed vertical lines correspond to the time instants of the frames in Fig. 6 and the circles indicate the active constraints at those instants).

- tenth frame (t=6.9): the workspace constraint on the x-coordinate becomes active for one end-effector;
- eleventh frame (t=8.7): the workspace constraint on the x-coordinate and the bar angle constraint are simultaneously active;
- twelfth frame (t=9.7): the end point of the reference path is achieved.

8. Simulation: case study

The two-dimensional example shown in the previous section was developed for better illustration of the main features of the algorithm. However, one could argue that it cannot be taken as evidence of generality of the proposed method. Therefore, a three-dimensional case study is presented in this section to demonstrate the effectiveness and real applicability of the method. As before, simulation results presented in this section were obtained using MATLAB® and a real-time animation of the resulting movement for this case study can be visualized at http://politube.upv.es/play.php?vid=58153. Details of pseudo-code and computing time for actual implementation of the proposed strategy appears in Appendix A at the end of the paper.

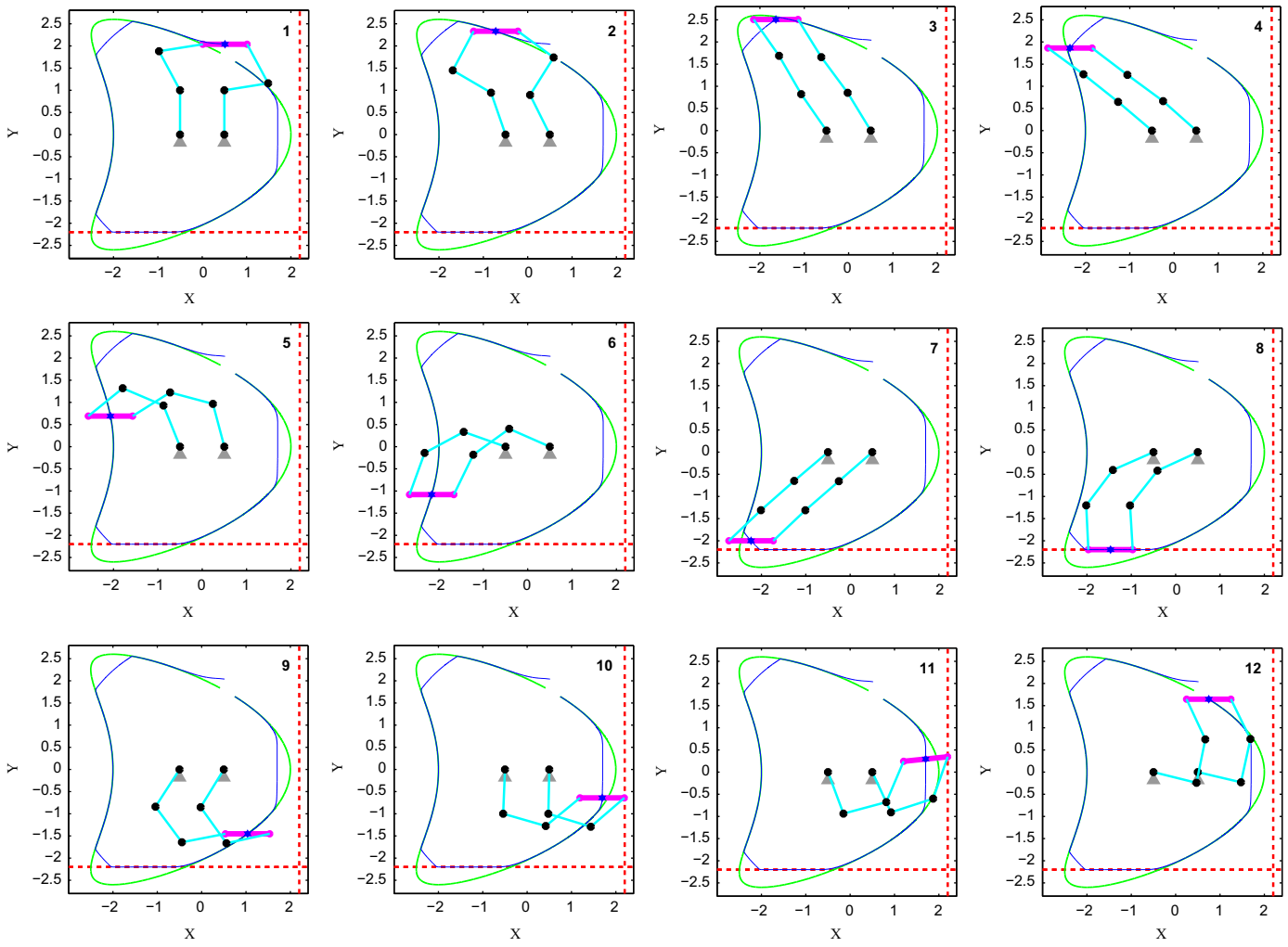

Fig. 6. Sequence of frames (frames 1–12) showing the 2D MRS configuration during the task simulation: reference path (thick line, green), path followed by the tracking point (thin line, blue), boundary of the allowed workspace (dashed red lines), robot joints (solid black discs), robot moving links (thin cyan bars), fixed link (triangles), carried bar (thick magenta bar) and tracking point (solid blue star). Active constraints at each frame shown in Fig. 5. (A video of the simulation can be played at http://politube.upv.es/play.php?vid=58151). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

8.1. Description of the robot setup for the case study

In the proposed case study, coordination between two PUMA-762 robots is considered. The PUMA-762 robot is a classical 6R serial manipulator with spherical wrist, which is widely used in industrial applications. The grippers of both robotic arms are used to grasp the end of a rigid bar in the shape of a rectangular prism and, therefore, both robots must be properly coordinated to carry the bar. Fig. 7 depicts the MRS in consideration for the case study, where letters "A" and "B" stand for the robot on the left and right sides, respectively.

The location of both robots is as follows. The Z-axis of the reference frame is aligned with the first joint of robot A and its origin is located at the same height of the second joint, i.e., the shoulder joint. Robot B is located 2 m away from robot A along the Y -axis of the reference frame and is rotated by π rad around the Z-axis of the reference frame, see Fig. 7.

The pose vector considered for both PUMA-762 robots is composed of the cartesian coordinates (x, y, z) describing the end-effector position and the roll-pitch-yaw Euler angles (α, β, γ) describing the end-effector orientation, that is

$$\mathbf{p}_A = \mathbf{l}_A(\mathbf{q}_A) = \mathbf{l}_A(q_1, q_2, q_3, q_4, q_5, q_6) = [x_A \;\; y_A \;\; z_A \;\; \alpha_A \;\; \beta_A \;\; \gamma_A]^{\mathrm{T}} \quad (55)$$

$$\mathbf{p}_B = \mathbf{l}_B(\mathbf{q}_B) = \mathbf{l}_B(q_7, q_8, q_9, q_{10}, q_{11}, q_{12}) = [x_B \;\; y_B \;\; z_B \;\; \alpha_B \;\; \beta_B \;\; \gamma_B]^{\mathrm{T}}, \quad (56)$$

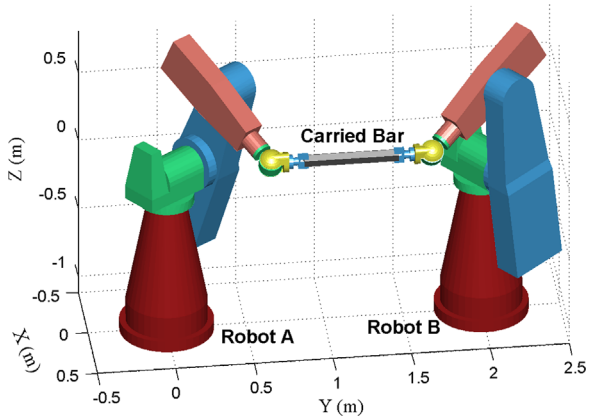

Fig. 7. Multi-robot system used for 3D simulation: two 6R PUMA-762 robots carrying a bar. (The geometry data used for the 3D representation of the PUMA-762 robot were taken from http://www.mathworks.com/matlabcentral/fileexchange/14932-3d-puma-robot-demo).

where the units for linear and angular dimensions are meters and radians, respectively, and $\mathbf{l}_A$ and $\mathbf{l}_B$ represent the kinematic function of robot A and robot B, respectively, which can be readily obtained

using the Denavit–Hartenberg method [33] (the Denavit–Hartenberg parameters of the PUMA-762 robot are shown in Table 1).

Furthermore, six elements are considered for the coordination vector $\mathbf{v}$ of the MRS: the cartesian coordinates (x_V, y_V, z_V) of the midpoint of the bar (i.e., the tracking point) and the roll-pitch-yaw Euler angles $(\alpha_V, \beta_V, \gamma_V)$ describing the bar orientation. Therefore, the 6-element coordination vector of the MRS is given by

$$\mathbf{v}(\mathbf{p}) = \mathbf{v}(\mathbf{p}_A, \mathbf{p}_B) = [x_V \;\; y_V \;\; z_V \;\; \alpha_V \;\; \beta_V \;\; \gamma_V]^{\mathrm{T}}. \tag{57}$$

Assuming that the rigid body constraints of the bar (which are defined below) are satisfied, the midpoint of the bar and its orientation can be computed, for instance, using only the pose vector of robot A.

8.2. Constraints for the case study

The constraints arising in the two upper levels are discussed below and, again, level 3 constraints have been considered to be $\dot{\mathbf{q}} = 0$. However, note that if the six rigid body constraints in level 1 (Section 8.2.1) together with the six constraints in level 2 (Section 8.2.2) associated with the reference tracking are independent, no remaining degrees of freedom of the MRS will be available for level 3.

8.2.1. Mandatory constraints (level 1) for the case study

As in the first simulation example, three sets of constraints are considered in the first priority level.

The first set is composed of the equality rigid body constraints associated with the bar carried by the robots. These equality constraints are given by

$$\sigma_{1,eq,L} = L_{bar}^2 - ((x_B - x_A)^2 + (y_B - y_A)^2 + (z_B - z_A)^2) = 0 \tag{58}$$

$$\sigma_{1,eq,u} = [x_B - x_A \;\; y_B - y_A \;\; z_B - z_A] u_A = 0 \tag{59}$$

$$\sigma_{1,eq,v} = [x_B - x_A \;\; y_B - y_A \;\; z_B - z_A] v_A = 0 \tag{60}$$

$$\sigma_{1,eq,\alpha} = \alpha_B - \alpha_A = 0 \tag{61}$$

$$\sigma_{1,eq,\beta} = \beta_B - \beta_A = 0 \tag{62}$$

$$\sigma_{1,eq,\gamma} = \gamma_B + \pi - \gamma_A = 0 \tag{63}$$

where L_{bar} is the length of the bar and vectors u_A and v_A represent the directions of the X- and Y-axis, respectively, of the tool frame of robot A. These vectors are computed from the Euler angles $(\alpha_A, \beta_A, \gamma_A)$ using the expression:

$$u_A = [\cos(\alpha_A) \;\; \cos(\beta_A) \;\; \sin(\alpha_A) \;\; \cos(\beta_A) \;\; -\sin(\beta_A)]^{\mathrm{T}} \tag{64}$$

$$v_A = \begin{bmatrix} -\sin(\alpha_A)\cos(\gamma_A) + \cos(\alpha_A)\sin(\beta_A)\sin(\gamma_A) \\ \cos(\alpha_A)\cos(\gamma_A) + \sin(\alpha_A)\sin(\beta_A)\sin(\gamma_A) \\ \cos(\beta_A)\sin(\gamma_A) \end{bmatrix} \tag{65}$$

Table 1
Denavit–Hartenberg parameters of the PUMA-762 robot using the standard convention (the value of d_6 includes the gripper length).

Link i	α_{i-1} (rad)	a_{i-1} (m)	d_i (m)	θ_i
1	0	0	0	q_1
2	$-\pi/2$	0	0	q_2
3	0	0.65	0.19	q_3
4	$-\pi/2$	0	0.6	q_4
5	$\pi/2$	0	0	q_5
6	$-\pi/2$	0	0.211	q_6

The meaning of the above rigid body constraints is as follows. The equality (58) forces the distance between the tools of both robots to be constant and equal to the bar length. The equalities (59) and (60) guarantee that the tool of robot A points to the position of the tool of robot B. Finally, the equalities (61)–(63) ensure that the tool of robot B points in the opposite direction of the tool of robot A and has the same rotation.

A second set of inequality constraints is considered to avoid collision between the carried bar and a sphere obstacle. In particular, the Cartesian position $[x_{cb,i} \; y_{cb,i} \; z_{cb,i}]^{\mathrm{T}}$ of every point i of the carried bar must fulfill the following inequality constraint:

$$R_{so} - \sqrt{(x_{cb,i} - x_{so})^2 + (y_{cb,i} - y_{so})^2 + (z_{cb,i} - z_{so})^2} \le 0 \quad \forall i, \tag{66}$$

where R_{so} and $[x_{so} \; y_{so} \; z_{so}]^{\mathrm{T}}$ are the radius and center, respectively, of the sphere obstacle.

The infinite number of points of the carried bar to be considered in (66) for collision avoidance can reduced to a set of *characteristic points* P_i such that the distance from every point of the carried bar to the closest characteristic point is less than a predetermined value, namely *security margin*, which is used to enlarge the constrained region of the sphere obstacle. Therefore, the following set of inequality constraints is considered:

$$\sigma_{1,in,Pi} = d_{sm} + R_{so} - \sqrt{(x_{cb,Pi} - x_{so})^2 + (y_{cb,Pi} - y_{so})^2 + (z_{cb,Pi} - z_{so})^2} \le 0 \quad i = 1, \ldots, N_{cp}, \tag{67}$$

where d_{sm} is the security margin, N_{cp} is the number of characteristic points considered and $[x_{cb,Pi} \; y_{cb,Pi} \; z_{cb,Pi}]^{\mathrm{T}}$ is the Cartesian position of the characteristic point P_i of the carried bar.

As a last mandatory constraint, the following inequality is also considered for the angle θ_{cb} between the carried bar and the horizontal plane to not exceed the maximum allowed value $\theta_{\max}$:

$$\sigma_{1,in,\theta} = |\theta_{cb}| - \theta_{\max} = \left| \arctan(w_{Az} / \sqrt{w_{Ax}^2 + w_{Ay}^2}) \right| - \theta_{\max} \le 0, \tag{68}$$

where vector $w_A = [w_{Ax} \; w_{Ay} \; w_{Az}]^{\mathrm{T}}$ represents the direction of the Z-axis of the tool frame of robot A and is computed from the Euler angles $(\alpha_A, \beta_A, \gamma_A)$ using the expression:

$$w_A = \begin{bmatrix} \sin(\alpha_A)\sin(\gamma_A) + \cos(\alpha_A)\sin(\beta_A)\cos(\gamma_A) \\ -\cos(\alpha_A)\sin(\gamma_A) + \sin(\alpha_A)\sin(\beta_A)\cos(\gamma_A) \\ \cos(\beta_A)\cos(\gamma_A) \end{bmatrix}. \tag{69}$$

8.2.2. Reference tracking (level 2) for the case study

For the case study, the reference path for the coordination vector is a straight line parallel to the X-axis of the reference frame given by the following expression:

$$\mathbf{v}_{ref}(s) = \begin{bmatrix} x_{V,ref}(s) \\ y_{V,ref}(s) \\ \theta_{V,ref}(s) \\ \alpha_{V,ref}(s) \\ \beta_{V,ref}(s) \\ \gamma_{V,ref}(s) \end{bmatrix} = \begin{bmatrix} 0.51 - s/10 \\ 1.111 \\ 0.35 \\ 0 \\ 0 \\ -\pi/2 \end{bmatrix}, \tag{70}$$

with $s = 0 \ldots 10$.

8.3. Simulation conditions and parameter values for the case study

Simulation was run under the same conditions as the first example (Section 7.3) except for the following:

(i) The dimensions of the carried bar (i.e., the rectangular prism) are the following: length (L_{bar}) 0.6 m, width 0.08 m and height 0.04 m.

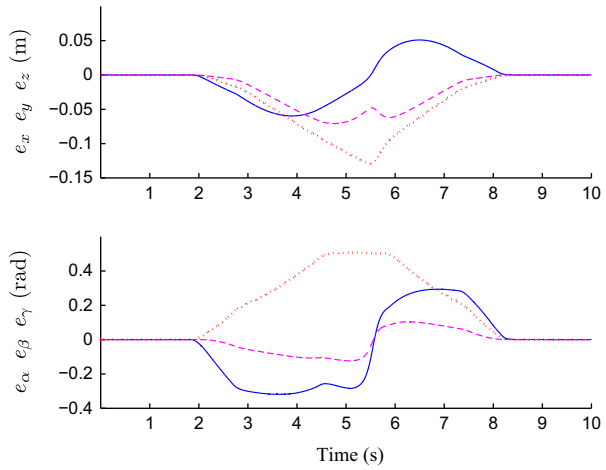

Fig. 8. Position tracking error $\mathbf{e} = \mathbf{v}_{ref} - \mathbf{v}$: e_x and e_α (solid, blue), e_y and e_β (dashed, magenta) and e_z and e_γ (dotted, red). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

(ii) For the mandatory constraints in the first priority level, the following parameter values were used: $R_{so} = 0.25$ m, $[x_{so}\ y_{so}\ z_{so}]^T = [0\ 0.95\ 0.2]^T$ m, $N_{cp} = 7$ (characteristic points for collision avoidance located uniformly along the symmetry axis of the bar), $d_{sm} = 0.1$ m[7] and $\theta_{\max} = 0.5$ rad.

(iii) For the reference tracking in the second priority level, a diagonal weighting matrix $\mathbf{W}_2 = \mathrm{diag}([1\ 1\ 1\ 0.1\ 0.1\ 0.1])$ was used.

(iv) The commanded joint acceleration vector $\ddot{\mathbf{q}}_c$ was computed with a sampling period T_s of 0.2 ms.

(v) We considered the initial MRS configuration given by $\mathbf{q}_A(0) = [0.6226\ -1.2196\ 0.0976\ -1.2689\ -1.0176\ 2.6065]^T$ rad and $\mathbf{q}_B(0) = [2.1998\ -1.4427\ 0.3852\ 0.9771\ -0.7893\ -2.3775]^T$ rad, yielding a zero initial robot position error $\mathbf{e}(0) = \mathbf{0}$.

8.4. Simulation results for the case study

As in the first example, the results of the simulation are depicted at different figures: Fig. 8 and 9 show the simulated behavior of the global system regarding tracking errors and constraints; 12 snapshot frames of a 3D representation of the MRS (robot A, robot B, carried bar, sphere obstacle) during the reference tracking at different time instants are shown in Fig. 10.

Fig. 8 shows how deviations from the reference trajectory arise when the mandatory inequality constraints for collision avoidance (i.e., $\phi_{1,in,Pi}$) become active at the time interval 1.8–8.2 s, see the fifth plot in Fig. 9. Fig. 8 also shows that the deviation e_γ for the yaw angle of the carried bar stops increasing once it reaches 0.5 rad since the inequality constraint for the bar angle θ_{cb} (i.e., $\phi_{1,in,\theta}$) becomes active, which occurs at the time interval 4.4–5.8 s, see the fifth plot in Fig. 9.

In contrast to the first example, the scale factor f_{AR} of the speed auto-regulation algorithm has been omitted from the plots of Fig. 9 because its value is almost constant at 1 during the whole simulation. Therefore, the motion rate parameter $\dot{s}$ is constant at full speed $\dot{s}_{\max}$, which means that the chosen amplitude u_1^+ for the switching law of the mandatory constraints is large enough to track the whole reference trajectory at maximum speed.

Fig. 9 also shows that

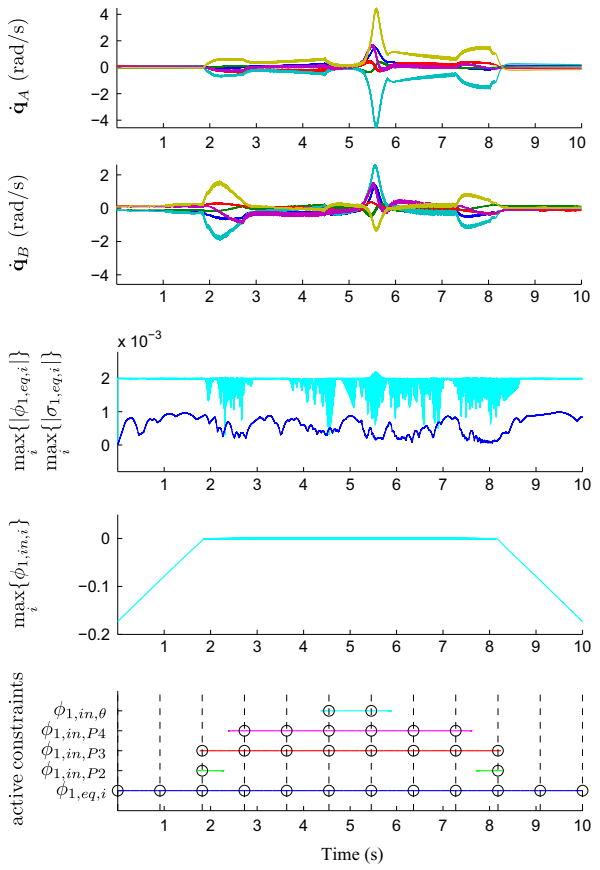

Fig. 9. From top to bottom plots: (1) joint speeds $\dot{\mathbf{q}}_A$ for robot A; (2) joint speeds $\dot{\mathbf{q}}_B$ for robot B; (3) maximum absolute value of the equality constraint functions $\phi_{1,eq,i}$ (light, cyan) and $\sigma_{1,eq,i}$ (dark, blue); (4) maximum value of the inequality constraint functions $\phi_{1,in,i}$; and (5) horizontal lines indicating when a constraint is active (the dashed vertical lines correspond to the time instants of the frames in Fig. 10 and the circles indicate the active constraints at those instants). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

- the joint speeds for robot A and robot B are less than 4 rad/s and 2 rad/s, respectively (see two upper plots in the figure);
- the error in the fulfillment of the mandatory equality constrains (rigid body constraints of the carried bar) is below 2×10^{-3}, i.e., $\max_i(|\phi_{1,eq,i}|) \leq 2 \times 10^{-3}$, which agrees with the value given by (41) for the chattering band: $T_s u_1^+ = 2 \times 10^{-4} \times 10 = 2 \times 10^{-3}$. Note also that the maximum error in the fulfillment of the original constraint functions $\sigma_{1,eq,i}$ is even smaller,[8] around 10^{-3} m (linear coordinates) and rad (angular coordinates), which is a very reasonable value for the maximum error[9] in the fulfillment of the rigid body constraints (third plot);
- all inequality constraints of level 1 are fulfilled, i.e., $\max_i(\phi_{1,in,i}) \leq 0$ (fourth plot);
- in some phases of the simulation there are up to three active inequality constraints at a time, i.e., up to nine constraints of level 1 are active at a time (bottom plot).

For further clarity, details on each of the presented frames from the achieved movements in Fig. 10 are discussed below:

[7] Since there are seven characteristic points located uniformly along the symmetry axis of the bar and the dimensions of the bar are $(L \times W \times H) = (0.6 \times 0.08 \times 0.04)$, the distance from a point on the boundary surface of the carried bar to the closest characteristic point ranges from 0.02 m to $\sqrt{0.05^2 + 0.04^2 + 0.02^2} = 0.0671$ m. Therefore the security margin must be larger than 0.0671 m.

[8] Note that $\sigma_{1,eq,i}$ is obtained by passing signal $\phi_{1,eq,i}$ through a first-order low-pass filter whose cutoff frequency is equal to $1/K_{1,i}$, see (24). Therefore, this filter smooths out the chattering band of $\phi_{1,eq,i}$ as the chattering frequency and/or the constraint approaching parameter $K_{1,i}$ increase.

[9] If needed, smaller errors can be obtained by increasing the sampling rate.

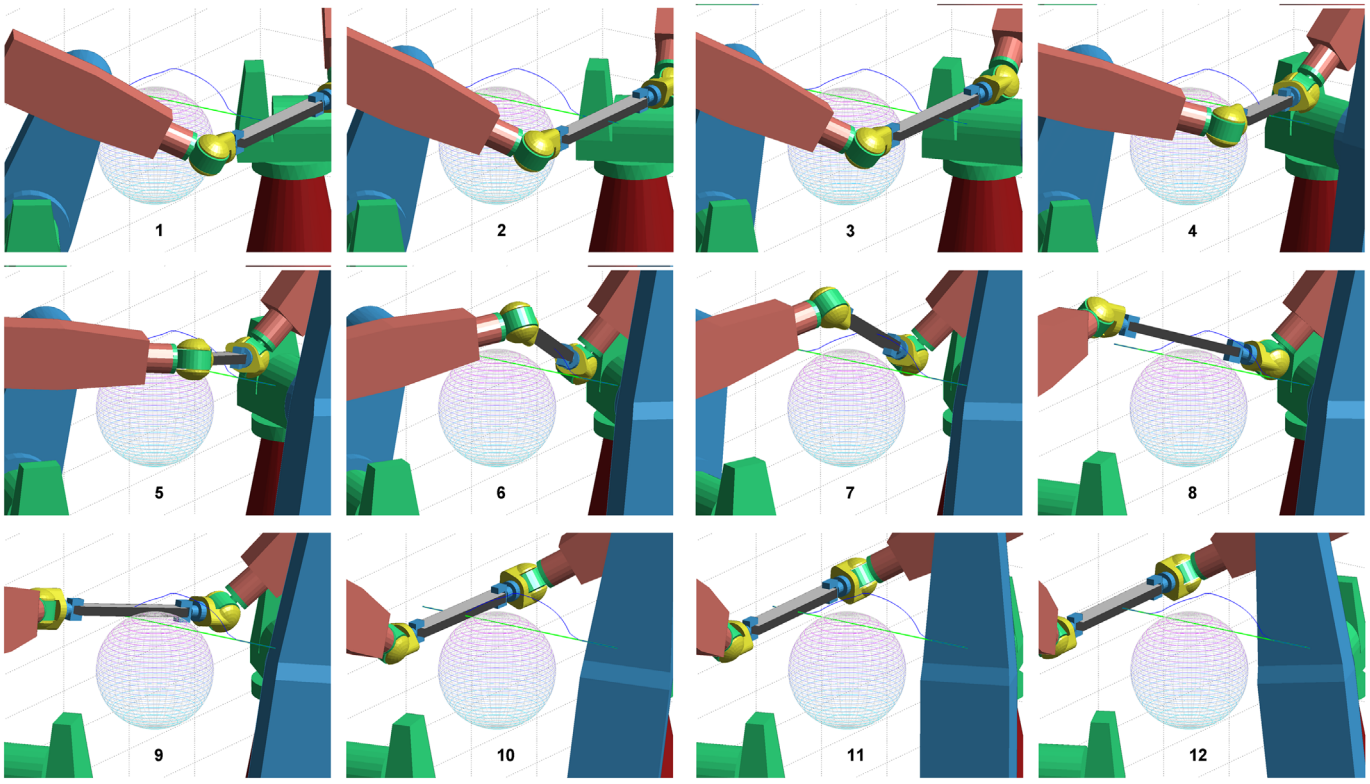

Fig. 10. Sequence of frames (frames 1–12) showing a detailed view of the 3D MRS operation area during the task simulation: reference path (thick line, green), path followed by the tracking point (thin line, blue), obstacle (sphere) and carried bar (prism, gray). Active constraints at each frame shown in Fig. 9. (A video of the simulation from different perspectives can be played at http://politube.upv.es/play.php?vid=58153.) (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

- first and second frames ($t=0$ and $t=0.91$): there is no tracking error;
- third frame ($t=1.82$): tracking error arises because two inequality constraints of level 1 become active in order to avoid collision with the sphere obstacle;
- fourth to ninth frames ($t=2.73$, $t=3.64$, $t=4.55$, $t=5.45$, $t=6.36$, $t=7.27$): there is tracking error;
- tenth frame ($t=8.18$): the tracking error is made zero again since no inequality constraint is active because the sphere obstacle has been overcome;
- eleventh frame ($t=9.09$): there is no tracking error;
- twelfth frame ($t=10$): the end point of the reference path is achieved.

9. Conclusions

An approach for robot coordination was developed using task-priority and sliding mode related concepts. In particular, the proposal is based on defining mandatory constraints representing the coordination of the multi-robot system, non-mandatory reference tracking ones and third-level goals. Sliding mode conditioning was proposed to ensure the satisfaction of the constraints, at different levels, using joint acceleration (or jerk, if so wished) as control input. If the model is known, lower level constraints can be also satisfied via model inversion.

The pseudo-code of the proposed approach appears in Appendix A, which simplifies the robot user interface since the method directly deals with the fulfillment of the constraints. Although the method was illustrated for two particular multi-robot systems (two 3R planar robots and two 6R PUMA-762 robots), the conclusions drawn for the proposed approach also apply to other settings.

Acknowledgements

This work was supported in part by research under Project DPI2011-27845-C02-01 of the Spanish Government and Projects ANPCyT PICT-2011-0888, CONICET PIP 112-2011-00361 and UNLP I164 in Argentina.

Appendix A. Computer implementation

The pseudo-code of the proposed approach for robot coordination is shown below. The algorithm, which is executed at a sampling period of T_s seconds, uses the following auxiliary functions:

- Constraint functions and gradient vectors for the first priority level: $\phi_{1,i}(\mathbf{q},\dot{\mathbf{q}})$ and $\nabla\sigma_{1,i}(\mathbf{q})$.
- Kinematic functions and Jacobian matrix: $\mathbf{l}_i(\mathbf{q}_i)$ and $\mathbf{J}_{pq}(\mathbf{q})$.
- Coordination vector and reference path: $\mathbf{v}([\mathbf{p}_1^{\mathrm{T}}\ldots\mathbf{p}_r^{\mathrm{T}}]^{\mathrm{T}})$ and $\mathbf{v}_{ref}(s)$.
- Moore–Penrose pseudoinverse and regularized pseudoinverse functions (Section 4.1): $(\cdot)^{\dagger}$ and $(\cdot)^{\sharp}$.
- Sensors: *GetRobotState*(), which returns the current robot state given by $\mathbf{q}$ and $\dot{\mathbf{q}}$.
- Actuators: *SendToJointControllers* ($\ddot{\mathbf{q}}_c$), which sends the current commanded joint acceleration vector to the joint controllers.

The computation time per iteration of the algorithm in a computer with the Intel Core i7-870 processor at 2.93 GHz clock frequency using MATLAB® R2009a (compiled C-MEX-file) was around 1.2 μs and 5 μs for the first simulation example in Section 7 and the case study example in Section 8, respectively.

Algorithm. Algorithm executed at sampling period of T_s seconds.

while $s < s_{end}$ **do**
  $[\mathbf{q}, \dot{\mathbf{q}}] = \texttt{GetRobotState}()$;
  if $\exists i | (\phi_{1,i}(\mathbf{q}, \dot{\mathbf{q}}) > 0)$ *and* $(\phi_{1,i,prev} > 0)$ **then** $u_{AR} = -1$;
  else $u_{AR} = 1$; // Eq. (42)
  $f_{AR} = f_{AR} + (T_s/\tau_{AR})u_{AR}$; // Integrator, Eq. (43)
  if $f_{AR} > 1$ **then** $f_{AR} = 1$; // Integrator saturation
  if $f_{AR} < 0$ **then** $f_{AR} = 0$; // Integrator saturation
  $\dot{s} = f_{AR}\dot{s}_{max}$; // Speed auto-regulation, Eq. (44)
  $s = s + T_s\dot{s}$; // Discrete-time (D-t) integration
  $\dot{\mathbf{v}}_{ref} = (\mathbf{v}_{ref}(s) - \mathbf{v}_{ref,prev})/T_s$; // D-t derivative
  $\ddot{\mathbf{v}}_{ref} = (\dot{\mathbf{v}}_{ref} - \dot{\mathbf{v}}_{ref,prev})/T_s$; // D-t derivative
  $\dot{\mathbf{J}}_{pq} = (\mathbf{J}_{pq}(\mathbf{q}) - \mathbf{J}_{pq,prev})/T_s$; // D-t derivative
  $\ddot{\mathbf{v}}_c = \ddot{\mathbf{v}}_{ref} + K_{2,v}(\dot{\mathbf{v}}_{ref} - \mathbf{J}_{pq}(\mathbf{q})\dot{\mathbf{q}}) + K_{2,p}(\mathbf{v}_{ref}(s) - \mathbf{v}([\mathbf{l}_1(\mathbf{q}_1)^T \ldots \mathbf{l}_r(\mathbf{q}_r)^T]^T)$; // Kinematic controller, Eq. (39)
  $\mathbf{A}_1 = \mathbf{K}_1 \nabla\sigma_1^T(\mathbf{q})$, $\nabla\sigma_1$ with all $\nabla\sigma_{1,i} | \phi_{1,i} > 0$; // Eq. (28)
  $\mathbf{b}_1 = -\mathbf{1}_b u_1^+$; // Eq. (27)
  $\mathbf{A}_2 = \mathbf{W}_2\mathbf{J}_{pq}(\mathbf{q})$; // Eq. (39)
  $\mathbf{b}_2 = \mathbf{W}_2(\ddot{\mathbf{v}}_c - \dot{\mathbf{J}}_{pq}\dot{\mathbf{q}})$; // Eq. (39)
  $\mathbf{A}_3 = \mathbf{I}$; // Eq. (40)
  $\mathbf{b}_3 = -K_{3,red}\dot{\mathbf{q}}$; // Eq. (40)
  $\ddot{\mathbf{q}}_{c,1} = \mathbf{A}_1^{\sharp}\mathbf{b}_1$; // Eq. (13), $i=1$
  $\mathbf{N}_1 = \mathbf{I} - \mathbf{A}_1^{\dagger}\mathbf{A}_1$; // Eq. (13), $i=1$
  $\ddot{\mathbf{q}}_{c,2} = \ddot{\mathbf{q}}_{c,1} + (\mathbf{A}_2\mathbf{N}_1)^{\sharp}(\mathbf{b}_2 - \mathbf{A}_2\ddot{\mathbf{q}}_{c,1})$; // Eq. (13), $i=2$
  $\mathbf{N}_2 = \mathbf{N}_1(\mathbf{I} - (\mathbf{A}_2\mathbf{N}_1)^{\dagger}(\mathbf{A}_2\mathbf{N}_1))$; // Eq. (13), $i=2$
  $\ddot{\mathbf{q}}_{c,3} = \ddot{\mathbf{q}}_{c,2} + (\mathbf{A}_3\mathbf{N}_2)^{\sharp}(\mathbf{b}_3 - \mathbf{A}_3\ddot{\mathbf{q}}_{c,2})$; // Eq. (13), $i=3$
  $\phi_{1,i,prev} = \phi_{1,i}(\mathbf{q}, \dot{\mathbf{q}}) \forall i$; // For next iteration
  $\mathbf{v}_{ref,prev} = \mathbf{v}_{ref}(s)$; // For next iteration
  $\dot{\mathbf{v}}_{ref,prev} = \dot{\mathbf{v}}_{ref}$; // For next iteration
  $\mathbf{J}_{pq,prev} = \mathbf{J}_{pq}(\mathbf{q})$; // For next iteration
  $\texttt{SendToJointControllers}(\ddot{\mathbf{q}}_{c,3})$;
end

Appendix B. Supplementary material

Supplementary data associated with this article can be found in the online version of http://dx.doi.org/10.1016/j.rcim.2013.08.003.

References

[1] Arai T, Pagello E, Parker L. Editorial: advances in multi-robot systems. IEEE Transactions on Robotics and Automation 2002;18:655–61.
[2] Parker L. Current state of the art in distributed autonomous mobile robotics. Distributed Autonomous Robotic Systems 2000;4:3–12.
[3] Cao Y, Fukunaga A, Kahng A. Cooperative mobile robotics: antecedents and directions. Autonomous Robots 1997;4:7–27.
[4] Dudek G, Jenkin M, Milios E, Wilkes D. A taxonomy for multi-agent robotics. Autonomous Robots 1996;3:375–97.
[5] Kim S. Coordination of multiagent systems through explicit valuation of action. Robotics and Computer-Integrated Manufacturing 1991;8:265–91.
[6] Basile F, Caccavale F, Chiacchio P, Coppola J, Curatella C. Task-oriented motion planning for multi-arm robotic systems. Robotics and Computer-Integrated Manufacturing 2012;28:569–82.
[7] Bozma H, Kalalıoğlu M. Multirobot coordination in pick-and-place tasks on a moving conveyor. Robotics and Computer-Integrated Manufacturing 2012;28:530–8.
[8] Fierro R, Das A, Spletzer JR. A framework and architecture for multi-robot coordination. International Journal of Robotics Research 2008;21:977–95.
[9] Edwards C, Spurgeon S. Sliding mode control: theory and applications. 1st ed.. UK: Taylor & Francis; 1998.
[10] Utkin V, Guldner J, Shi J. Sliding mode control in electro-mechanical systems. 2nd ed.. London: Taylor & Francis; 2009.
[11] Garelli F, Mantz R, De Battista H. Advanced control for constrained processes and systems. IET, Control Engineering Series. London, UK; 2011.
[12] Gracia L, Sala A, Garelli F. A path conditioning method with trap avoidance. Robotics and Autonomous Systems 2012;60:862–73.
[13] Gracia L, Sala A, Garelli F. A supervisory loop approach to fulfill workspace constraints in redundant robots. Robotics and Autonomous Systems 2012;60:1–15.
[14] Gracia L, Garelli F, Sala A. Integrated sliding-mode algorithms in robot tracking applications. Robotics and Computer-Integrated Manufacturing 2013;29:53–62.
[15] Vignoni A, Picó J, Garelli F, De Battista H. Dynamical systems coordination via sliding mode reference conditioning. In: Proceedings of the 18th IFAC world congress. Milano, Italy; 2011.
[16] Vignoni A, Picó J, Garelli F, De Battista H. Sliding mode reference conditioning for coordination in swarms of non-identical multi-agent systems. In: Proceedings of the 12th IEEE international workshop on variable structure systems (VSS). Mumbai; 2012. p. 231–6.
[17] Parker L. Distributed intelligence: overview of the field and its application in multi-robot systems. Journal of Physical Agents 2008;2:5–14.
[18] Cui X, Shin K. Direct control and coordination using neural networks. IEEE Transactions on Systems, Man and Cybernetics 1996;23:686–97.
[19] Tunstel E, de Oliveira M, Berman S. Fuzzy behavior hierarchies for multi-robot control. International Journal of Intelligent Systems 2002;17:449–70.
[20] Shibata T, Fukuda T. Coordination in evolutionary multi-agent-robotic system using fuzzy and genetic algorithm. Control Engineering Practice 1994;2:103–11.
[21] Nakamura Y, Hanafusa H, Yoshikawa T. Task-priority based redundancy control of robot manipulators. International Journal of Robotics Research 1987;6:3–15.
[22] Garelli F, Mantz R, De Battista H. Limiting interactions in decentralized control of MIMO systems. Journal of Process Control 2006;16:473–83.
[23] Angeles J. Fundamentals of robotic mechanical systems: theory, methods and algorithms. 3rd ed.. New York, NJ: Springer-Verlag; 2007.

[24] Garelli F, Gracia L, Sala A, Albertos P. Sliding mode speed auto-regulation technique for robotic tracking. Robotics and Autonomous Systems 2011;59:519–529.
[25] Gracia L, Garelli F, Sala A. Reactive sliding-mode algorithm for collision avoidance in robotic systems. IEEE Transactions on Control Systems Technology, in press. http://dx.doi.org/10.1109/TCST.2012.2231866.
[26] Chiaverini S, Oriolo G, Walker I. Kinematically redundant manipulators. Springer handbook of robotics; 2008. p. 245–68.
[27] Siciliano B, Slotine J. A general framework for managing multiple tasks in highly redundant robotic systems. In: Proceedings of the fifth international conference on advanced robotics (ICAR'91). Pisa, Italy; 1991. p. 1211–6.
[28] Golub G, Van Loan C. Matrix computations. 3rd ed.. Baltimore, MD: The Johns Hopkins University Press; 1996.
[29] Deo A, Walker I. Overview of damped least-squares methods for inverse kinematics of robot manipulators. Journal of Intelligent & Robotic Systems 1995;14:43–68.
[30] Wampler C. Manipulator inverse kinematic solutions based on vector formulations and damped least squares methods. IEEE Transactions on Systems, Man, and Cybernetics 1986;16:93–101.
[31] Nakamura Y, Hanafusa H. Inverse kinematic solutions with singularity robustness for robot manipulator control. ASME Journal of Dynamic Systems, Measurement, and Control 1986;108:163–71.
[32] Maciejewski A, Klein C. Numerical filtering for the operation of robotic manipulators through kinematically singular configurations. Journal of Robotic Systems 1988;5:527–52.
[33] Siciliano B, Sciavicco L, Villani L, Oriolo G. Robotics: modelling, planning and control. London, UK: Springer-Verlag; 2009.
[34] Khalil W, Dombre E. Modeling, identification and control of robots. Bristol, PA: Taylor & Francis Inc.; 2002.