Copyright Information

Journal of Intelligent & Robotic Systems manuscript No.
(will be inserted by the editor)

A Novel Practical Technique to Integrate Inequality Control Objectives and Task Transitions in Priority Based Control

Enrico Simetti · Giuseppe Casalino

Received: date / Accepted: date

Abstract The task priority based control is a formalism which allows to create complex control laws with nice invariance properties, i.e. lower priority tasks do not affect the execution of higher priority ones. However, the classical task priority framework (Siciliano and Slotine) lacked the ability of enabling and disabling tasks without causing discontinuities. Furthermore, tasks corresponding to inequality control objectives could not be efficiently represented within that framework. In this paper we present a novel technique to integrate both the activation and deactivation of tasks and the inequality control objectives in the priority based control. The technique, called iCAT (inequality control objectives, activations and transitions) task priority framework, exploits novel regularization methods to activate and deactivate any row of a given task in a prioritized hierarchy without incurring in practical discontinuities, while maintaining as much as possible the invariance properties of the other active tasks. Finally, as opposed to other techniques, the proposed approach has a linear cost in the number of tasks. Simulations, experimental results and a time analysis are presented to support the proposed technique.

Keywords task priority control · redundant robots, · kinematics · hierarchical control.

This work has been supported by the MIUR (Ministry of Education, University and Research) through the MARIS prot. 2010FBLHRJ project and by the European Commission through H2020-BG-06-2014-635491 DexROV project.

E. Simetti
DIBRIS - University of Genova, Via Opera Pia 13, 16145 Genova, Italy
Tel.: +39 010 3532091
Website: `http://www.graal.dibris.unige.it`
E-mail: enrico.simetti@unige.it

G. Casalino
DIBRIS - University of Genova, Via Opera Pia 13, 16145 Genova, Italy
E-mail: giuseppe.casalino@unige.it

1 Introduction

Nowadays, redundant robotic systems are becoming increasingly more common than a few decades ago. As these systems increased with popularity, the computational power available to control them has scaled from the 40 MHz Motorola 68040 CPUs that were used to control the 7 d.o.f. AMADEUS arms [26], where a computation of a 6×7 pseudo-inverse required around 150 ms, to dual or quad core GHz systems where the same operation now takes a few microseconds. The increased computational power has allowed to exploit the high versatility offered by such systems, due to their high number of degrees of freedom (d.o.f.), in order to perform different, complex tasks. Examples of such systems are humanoid robots [9,14,17], aerial vehicles with manipulation capabilities [21], mobile manipulators [47] and autonomous underwater vehicle manipulator systems [33, 40].

Given the high versatility and the fact that such systems need to complete different goals, it became important to have a simple and effective way to specify the control objectives. The task-based control [37], also known as operational space control [25], allows one to define the control objectives in a coordinate system that is directly relevant to the task that needs to be performed, rather than in the generalized coordinates of the robotic system.

With the possibility of specifying the objectives directly in the operational space, it became even more important to efficiently exploit the redundancy of the system. The first studies on this subject date back to the seminal works [29, 50], where the residual d.o.f., after accomplishing the end-effector position control, were used in order to tackle the problem of singularity and obstacle avoidance for an industrial manipulator. Another seminal work [24] introduced the use of potential functions in order to efficiently implement the obstacle avoidance for manipulators and mobile robots.

Since the control objectives do not always have the same importance, the task-based control was enhanced by the introduction of the concept of task priority [36]. A primary task was executed, and a secondary task was accomplished (or attempted) only in the null space of the primary one, in order to guarantee the invariance of the main task w.r.t. (with respect to) the secondary one. This concept was later generalized to any number of task-priority levels in the seminal work [42].

Differing from the above works, [15] proposed a suboptimal approach, i.e. to compute the secondary task as if it was alone, and projecting it in the null space of the higher priority one, using a variable damping factor as proposed in [37] to deal with singularities. This was later generalized in [2] and called null-space-based behavioral control. The main advantage of this approach is that it avoids the problem of algorithmic singularities that can occur due to rank loss caused by the projection matrix. Further works on achieving a task priority control robust to singularities are found in [22, 34, 35].

Since those times, the task-priority framework has been applied to numerous robotic systems: redundant manipulators, as was obvious in each of the works mentioned until now; mobile manipulators such as [3, 4, 13, 33, 51] and multiple coordinated mobile manipulators [39, 47]; modular robots [11]; and also humanoid robots [41, 48] just to cite a few examples. Furthermore, a stability analysis for several prioritized inverse kinematics algorithms can be found in [1].

An important limitation of the classical task priority framework, which is evident in all the previous mentioned works, is that inequality control objectives (e.g.

avoiding the joint limits) were never treated as such. Indeed, the corresponding task was always active, even when the joints were sufficiently far away from their limits. The reason for this is the fact that activating (inserting) or deactivating (deleting) a task implies a discontinuity in the null space projector, which leads to a discontinuity in the control law [28]. As a consequence, such control objectives could be only considered as secondary tasks, otherwise they would have consumed many d.o.f. even when not necessary (i.e. joints far away from their limit). This led to an undesired situation where safety related tasks had a lower priority w.r.t. mission-centric tasks such as the end-effector position control.

Thus, in the last decade, major research efforts have been spent in order to incorporate inequality control objectives in the task-based control paradigm in a more efficient way. In [32] a new inversion operator is introduced for the computation of a smooth inverse with the ability of enabling and disabling tasks in the context of visual servoing. That work only dealt with the activation and deactivation of rows of a single multidimensional task (i.e. a single level of priority). The extension of the operator to the case of a hierarchy of tasks is provided in [31]. However, the major problem within that work is that it requires the computation of all the combinations of possible active and inactive tasks, which grows exponentially as the number of tasks increases.

Another interesting approach is given in [28]. The idea is to modify the reference of each task that is inserted or being removed, in order to comply with the already present ones, in such a way to smooth out any discontinuity. However, the algorithm requires $m!$ pseudo inverses with m number of tasks. The authors provide also approximate solutions, which are suboptimal whenever more than one task is being activated or deactivated (in *transition*).

A further way to solve a hierarchy of tasks is proposed in [23], which generalizes the earlier work [20] to any number of priority levels. Differing from the aforementioned works, [23] directly incorporates the inequality control objectives as inequality constraints in a Quadratic Program (QP). At each priority level, they solve a QP problem finding the optimal solution in a least-squares sense, using slack variables to incorporate inequality constraints in the minimization process. If the solution contains a slack variable different from zero, it means that the corresponding inequality constraint is not satisfied. Then they propagate to the next level all the inequality constraints that were satisfied, and transform each one that was not into an equality one (to prevent lower priority tasks from changing the best least-square trade-off found). A similar process is done for the equality constraints. The main drawbacks of this approach are that the cascade of QP problems grows in dimension and that the work cannot handle the activation or deactivation of tasks. Such a property is required whenever temporal sequences of tasks are used, as it happens whenever complex behaviour such as assembling objects is required [7,38]. A similar approach is given in [16], where again the inequality constraints are directly tackled within a constrained minimization problem. However, no explicit way to activate and deactivate an equality task is given.

Building on the results of [23], a hierarchical active set search is proposed in [19]. The main difference with [23] is that instead of a cascade of QP problems, the authors solve a single problem finding the active set of all the constraints at the same time. This allows to speed up the process quite a bit w.r.t. [23]. Due to its iterative nature, the authors propose to limit the iterations number to achieve a boundary on the computation times and be more suitable for a real-time

implementation, although the resulting solution is not optimal. Similarly as [16,23], the work does not have an explicit way to activate and deactivate an equality task.

An extension of the null space behavioural control which includes set-based objectives has also been proposed in [6]. The paper does not deal with activation and deactivation of tasks, and further works are needed to cope with its computational load.

Previous works of the authors dealt with the control of underwater free floating manipulators [12,45,46] in the context of the TRIDENT Framework Programme 7 (FP7) project. In such works, all the tasks except the end-effector position control were represented by scalar inequality objectives. The activation and deactivation of scalar tasks was tackled in the prioritized control. While proving sufficient and well working for the problem at hand also during the field trials, such works lacked the ability to deal with the activation and deactivation of multidimensional tasks, i.e. multiple scalar tasks at the same priority level.

In this work, we present a simple and generalized framework that has the capability of activating and deactivating tasks without incurring in practical discontinuities. It retains the simplicity of the original task-priority framework [42] since it only uses pseudo inverses. Tasks are activated and deactivated via the use of an activation matrix. The possible discontinuities that can arise with the use of an activation matrix [31, 32] are eliminated with the use of a novel task-oriented regularization and the singular value oriented one.

With the possibility of activating and deactivating tasks, it becomes possible to easily do the following:

- treat inequality control objectives efficiently: a suitable reference rate driving the system toward the region where the inequality objective is satisfied is generated, however the task (i.e. tracking such a reference rate) is deactivated whenever the system is inside the validity region, thus no over-constraining of the system is done;
- manage task transitions in a simple way: whenever a temporal sequence of tasks must be performed, it is possible to deactivate the tasks that are not anymore necessary and activate the new ones using the same activation function mechanism, which can be applied to equality control objectives as well as inequality ones in a uniform manner.

The work is structured as follows. Section 2 introduces some definitions and recalls some basic principles about the regularized pseudo inverses. Section 3 presents the basic idea to activate and deactivate rows of a single multidimensional task without practical discontinuities. Section 4 generalizes the approach to any number of multidimensional tasks inserted in multiple priority levels. Section 5 presents the activation functions and a deep discussion on task transitions. Simulation and experimental results, together with a computation time analysis, are presented in Section 6, and some final conclusions are given in Section 7. Finally, Appendix A gives further insights on how to construct the activation functions to implement inequality control objectives and task transitions.

2 Definitions and Preliminaries

Before starting the discussion we first introduce some notation in the next subsection. Then some definitions are given in the successive subsection, while some preliminaries on the regularized pseudo inverses are given in the last one.

2.1 Notation

Vectors and matrices are expressed with a bold face character, such as $\boldsymbol{M}$, whereas scalar values are represented with a normal font such as γ. Given a matrix $\boldsymbol{M}$:

- $M_{(i,j)}$ indicates the element of $\boldsymbol{M}$ at the i-th row and j-th column;
- $\boldsymbol{M}_{\{k\}}$ refers to the k-th row of $\boldsymbol{M}$;
- $\boldsymbol{M}^{\#}$ is the exact generalized pseudo-inverse (see [8] for a review on pseudo-inverses and their properties), i.e. the pseudo inverse of $\boldsymbol{M}$ performed without any regularizations.

Further, less used, notation is introduced as needed.

2.2 Definitions

We report hereafter some definitions often used in this paper:

- the system configuration vector $\boldsymbol{c} \in \mathbb{R}^n$ of a robotic system. For example, this vector is formed by the collection of all the joint positions if we consider just a robotic arm, the position and three orientation parameters if we consider an underwater vehicle, or both if we consider an underwater vehicle manipulator system;
- the system velocity vector $\dot{\boldsymbol{q}} \in \mathbb{R}^n$. Note that the system velocity vector may be different from the derivative of the system configuration vector. An example of such a case is an underwater vehicle, where the system velocity vector is the collection of the three linear and three angular velocities, where the latter are not the derivatives of the orientation parameters;
- a configuration dependant scalar variable $x(\boldsymbol{c})$ is said associated to an *equality control objective* when it is required to eventually satisfy, if possible,

$$x(\boldsymbol{c}) = x_0, \tag{1}$$

or to an *inequality control objective* when it is required to eventually satisfy, if possible,

$$x(\boldsymbol{c}) \geq x_m \tag{2}$$

or

$$x(\boldsymbol{c}) \leq x_M. \tag{3}$$

The case where a variable needs to stay within an interval can be represented by two separate objectives. Examples of such variables are the arm joints q_i, which are required to be within the joint limits, or the manipulability measure μ, which is required to be above a certain minimum threshold.

However, we can also consider as $x(\boldsymbol{c})$ the modulus of a certain vector $\boldsymbol{p}$. This allows to control its norm to a particular value (for instance if we want to nullify some error vector or having a vector $\boldsymbol{p}$ lying on a sphere surface), to be below a given threshold (if we just need to keep a reasonable error bound), or to be above a threshold (for instance if such a vector is a distance between two objects to avoid collisions).
In addition, we can also consider $x(\boldsymbol{c})$ to be the $i-$th component of a vector $\boldsymbol{p} \in \mathbb{R}^m$. Then if we consider its m different variables $x_i(\boldsymbol{c})$, it is possible to ask vector $\boldsymbol{p}$ to be inside/outside a polyhedral surface, or just achieving any desired value.
For the remainder of the paper, we will drop the dependency of x from $\boldsymbol{c}$ to ease the notation;

– for such variables, we also consider the existing Jacobian relationship between x and the system velocity vector $\dot{\boldsymbol{q}}$ as

$$\dot{x} = \boldsymbol{g}^T(\boldsymbol{c})\dot{\boldsymbol{q}}, \tag{4}$$

where $\boldsymbol{g} \in \mathbb{R}^n$ is a vector. Again, in the rest of the paper we will drop the dependency of $\boldsymbol{g}$ from $\boldsymbol{c}$;

– we define as a reference rate $\dot{\bar{x}}$ associated to a control objective any rate capable of driving the associated variable x toward the corresponding control objective. For equality control objectives, a suitable reference rate $\dot{\bar{x}}$ that drives x toward x_0 is

$$\dot{\bar{x}} \triangleq \kappa(x_0 - x),\ \kappa > 0, \tag{5}$$

where κ is a positive gain to control the convergence speed. For inequality control objectives, a suitable reference rate $\dot{\bar{x}}$ is any rate that drives x toward any arbitrary point inside the region where the inequality is satisfied. For instance, consider an inequality objective of the type $x \leq x_M$ and consider a point x^* such that $x^* \leq x_M$, then a suitable reference rate that drives x toward its corresponding objectives is

$$\dot{\bar{x}} \triangleq \kappa(x^* - x),\ \kappa > 0. \tag{6}$$

As it is clear from the above equations, the main difference between the two kind of objectives lies in the arbitrariness of the choice of x^* compared to x_0;

– an *active task* is defined as the requirement of tracking at best a reference rate $\dot{\bar{x}}$ in the current operational conditions, while an *inactive task* is trivially defined as the the absence of any tracking requirement of a reference rate $\dot{\bar{x}}$. Independently from the active or inactive state, the two categories will however be indicated via the common term *task*.
To clarify the above definitions, we can observe how when an inequality objective is not satisfied, then the associated task must be active to drive the variable toward the region where the inequality is satisfied, while when it is satisfied it can also be inactive. For equality objectives the corresponding task is always active, unless external conditions (i.e independent from the achievement or not of the equality, for example related to a change of mission phase) are allowed to intervene to activate or deactivate the corresponding task. Such a situation can of course be also applied to tasks associated to inequality objectives.

In the following we shall see how suitable combinations of continuous activation functions will reveal as the necessary tool for guaranteeing continuous transitions of tasks from active to inactive states and vice versa. During a transition the corresponding task will be therefore denoted as being *in transition*;
- the control objectives may have different *priorities* and the same holds for their associated tasks. Whenever multiple scalar tasks are considered with the same priority they form a so called *multidimensional* task, where its related variable $\boldsymbol{x}$ results as the stacking of all the scalar variables x, and the same goes for its reference rate $\dot{\bar{\boldsymbol{x}}}$. We will term the associated Jacobian as $\boldsymbol{J}$, which is the stacking of all the vector Jacobians $\boldsymbol{g}^T$. Note that a multidimensional task may be formed, at a given time, by any combination of *active*, *inactive* and *in transition* scalar tasks;
- all the *active* scalar tasks within a multidimensional task should be achieved simultaneously, if possible. Scalar or multidimensional tasks with lower priority should not interfere with the achievement of any *active* tasks with higher priority. A set of tasks with different priorities is also called a *hierarchy of tasks*.
- a *practical discontinuity* is a mathematically continuous rapidly varying behaviour that becomes a discontinuity or chattering phenomena once implemented in a discretized control system for practical values of the control gains. As it will be clear in the rest of the paper, most of the present work is aimed at mitigating the possible effects of practical discontinuities in order to implement task transitions without strong impacts on the control performances.

2.3 Regularized Pseudo Inverse

Before introducing the core of this work, it is useful to recall the fundamentals of pseudo inverses and the regularization mechanism. Toward that end, let us consider the following Jacobian relationship

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}}, \tag{7}$$

with $\boldsymbol{J} \in \mathbb{R}^{m\times n}$, $\dot{\boldsymbol{q}} \in \mathbb{R}^n$ and $\dot{\boldsymbol{x}} \in \mathbb{R}^m$.

Given a reference velocity vector $\dot{\bar{\boldsymbol{x}}}$, it is well known that the minimum norm velocity vector $\dot{\boldsymbol{q}}$ that realizes $\dot{\bar{\boldsymbol{x}}}$ at best, in a least-squares sense, can be found as the minimum norm solution of the following minimization problem

$$\min_{\dot{\boldsymbol{q}}} \left\| \dot{\bar{\boldsymbol{x}}} - \boldsymbol{J}\dot{\boldsymbol{q}} \right\|^2. \tag{8}$$

Such a solution can be found computing the square and taking the derivative equal to zero, which leads to the condition

$$(\boldsymbol{J}^T\boldsymbol{J})\dot{\boldsymbol{q}} = \boldsymbol{J}^T\dot{\bar{\boldsymbol{x}}}, \tag{9}$$

which can be solved using the pseudo inverse as

$$\dot{\boldsymbol{q}} = (\boldsymbol{J}^T\boldsymbol{J})^{\#}\boldsymbol{J}^T\dot{\bar{\boldsymbol{x}}}. \tag{10}$$

Since for the rest of the work we will focus on exploiting the residual arbitrariness in the solution, it is worth to consider the actual manifold of all solutions, by also including the null space, that is the manifold:

$$\dot{\boldsymbol{q}} = (\boldsymbol{J}^T\boldsymbol{J})^{\#}\boldsymbol{J}^T\dot{\bar{\boldsymbol{x}}} + (\boldsymbol{I} - (\boldsymbol{J}^T\boldsymbol{J})^{\#}\boldsymbol{J}^T\boldsymbol{J})\dot{\boldsymbol{z}},\ \forall\dot{\boldsymbol{z}}. \tag{11}$$

Then, by exploiting the following identities involving pseudo inverses

$$\boldsymbol{G}^{\#} = (\boldsymbol{G}^T\boldsymbol{G})^{\#}\boldsymbol{G}^T, \tag{12}$$

$$\boldsymbol{G}^{\#} = \boldsymbol{G}^T(\boldsymbol{G}\boldsymbol{G}^T)^{\#}, \tag{13}$$

we note that (11) can be equivalently rewritten in the following, more usual form

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\#}\dot{\bar{\boldsymbol{x}}} + (\boldsymbol{I} - \boldsymbol{J}^{\#}\boldsymbol{J})\dot{\boldsymbol{z}},\ \forall\dot{\boldsymbol{z}}. \tag{14}$$

Performing a regularization means changing the original minimization problem (8) by adding an additional regularization cost, as in the following

$$\min_{\dot{\boldsymbol{q}}} \left\|\dot{\bar{\boldsymbol{x}}} - \boldsymbol{J}\dot{\boldsymbol{q}}\right\|^2 + \|\dot{\boldsymbol{q}}\|^2_{\boldsymbol{R}}\ ;\ \boldsymbol{R} \geq \boldsymbol{0}, \tag{15}$$

where the notation $\|\dot{\boldsymbol{q}}\|^2_{\boldsymbol{R}}$ indicates the weighted norm, i.e. $\dot{\boldsymbol{q}}^T\boldsymbol{R}\dot{\boldsymbol{q}}$.

This is usually done to deal with the ill-definition of the solution near a singularity of the matrix $\boldsymbol{J}$. The solution of the regularized problem is found along the same steps performed for the original one, as the minimum norm solution of the following equation

$$(\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R})\dot{\boldsymbol{q}} = \boldsymbol{J}^T\dot{\bar{\boldsymbol{x}}} \tag{16}$$

that is

$$\dot{\boldsymbol{q}} = (\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R})^{\#}\boldsymbol{J}^T\dot{\bar{\boldsymbol{x}}}, \tag{17}$$

while the manifold of all solutions is

$$\dot{\boldsymbol{q}} = (\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R})^{\#}\boldsymbol{J}^T\dot{\bar{\boldsymbol{x}}} + (\boldsymbol{I} - (\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R})^{\#}(\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R}))\dot{\boldsymbol{z}},\ \forall\dot{\boldsymbol{z}}. \tag{18}$$

However it is important to note that the above projection matrix $(\boldsymbol{I} - (\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R})^{\#}(\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R}))$ preserves the overall minimum of problem (15), including that of the regularization cost $\|\dot{\boldsymbol{q}}\|^2_{\boldsymbol{R}}$. This means that if $\boldsymbol{R} > \boldsymbol{0}$ then the projection matrix becomes equal to $\boldsymbol{0}$ (e.g. in the case of damped least squares, see the next subsection) and no arbitrariness is available.

For the above reason, the following substitution is instead typically performed

$$\boldsymbol{J}^{\#} \rightarrow (\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R})^{\#}\boldsymbol{J}^T \tag{19}$$

and a manifold of solutions with $\boldsymbol{J}$ regularized by $\boldsymbol{R}$ is consequently constructed by analogy with (14) as

$$\dot{\boldsymbol{q}} = (\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R})^{\#}\boldsymbol{J}^T\dot{\bar{\boldsymbol{x}}} + (\boldsymbol{I} - (\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R})^{\#}\boldsymbol{J}^T\boldsymbol{J})\dot{\boldsymbol{z}},\ \forall\dot{\boldsymbol{z}}. \tag{20}$$

It is noteworthy to see that the projection matrix $(\boldsymbol{I} - (\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{R})^{\#}\boldsymbol{J}^T\boldsymbol{J})$ is not anymore an orthogonal projector on the null space of $\boldsymbol{J}$ whenever $\boldsymbol{R} \neq \boldsymbol{0}$.

Important remark: In some works, notwithstanding the introduction of a regularization of $\boldsymbol{J}$ with $\boldsymbol{R}$ in the minimum norm solution, the null space projection matrix is calculated without the regularization matrix $\boldsymbol{R}$, and thus is still an orthogonal projection. However, since exploiting the non-orthogonality of the projection matrix is actually one of the key ideas of this work, we will always consider it computed with $\boldsymbol{J}$ regularized by $\boldsymbol{R}$ as in (20). Furthermore, if the projection matrix is calculated without any regularization, a discontinuity occurs whenever an exact singularity of the matrix $\boldsymbol{J}$ is encountered.

2.3.1 Damped Least Squares

Among the different regularization we can cite the damped least squares (DLS)

$$\min_{\dot{\boldsymbol{q}}} \left\|\dot{\tilde{\boldsymbol{x}}} - \boldsymbol{J}\dot{\boldsymbol{q}}\right\|^2 + \gamma^2 \left\|\dot{\boldsymbol{q}}\right\|^2, \tag{21}$$

where thus $\boldsymbol{R} = \gamma^2 \boldsymbol{I}$. In this case, finding the solution to the original problem (8) is balanced with the requirement of also maintaining all the components of the velocity vector $\dot{\boldsymbol{q}}$ limited. The scalar value γ balances this trade-off: smaller values of γ will favor accomplishing the original minimization, while greater values will maintain $\dot{\boldsymbol{q}}$ much more limited. A more detailed discussion on DLS can be found in the book [27] and in [49].

2.3.2 Singular Value Oriented Regularization

Another popular regularization is the SVD-based one (see the book [36] for a deeper investigation). In this case, the Jacobian matrix is first decomposed along its singular values:

$$\boldsymbol{J} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \tag{22}$$

with $\boldsymbol{U} \in \mathbb{R}^{m\times m}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times n}$. $\boldsymbol{\Sigma}$ is a rectangular diagonal matrix, whose diagonal elements are the singular values. $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal matrices, where the m columns of $\boldsymbol{U}$ and the n columns of $\boldsymbol{V}$ are called the left-singular vectors and right-singular vectors of $\boldsymbol{J}$ and are respectively the eigenvectors of $\boldsymbol{J}\boldsymbol{J}^T$ and $\boldsymbol{J}^T\boldsymbol{J}$.

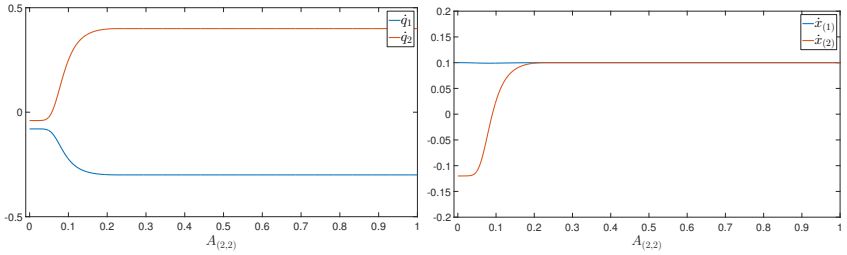
It is easy to see that, if a singular value is small, there is a need for a big effort along corresponding control direction in $\boldsymbol{V}$ to accomplish a given task along the corresponding task direction of $\boldsymbol{U}$. To prevent the control from growing unbounded as the singular value approaches zero, a diagonal singular value oriented regularization matrix $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ can be computed as follows: each diagonal elements $p_{(i,i)}$ of $\boldsymbol{P}$ is a bell-shaped, finite support function of the corresponding singular value, or it is zero if $i > m$ (i.e. the corresponding singular value does not exist).

With this definition, the singular value oriented (SVO) regularization problem becomes that of minimizing

$$\min_{\dot{\boldsymbol{q}}} \left\|\dot{\tilde{\boldsymbol{x}}} - \boldsymbol{J}\dot{\boldsymbol{q}}\right\|^2 + \left\|\boldsymbol{V}^T\dot{\boldsymbol{q}}\right\|_{\boldsymbol{P}}^2, \tag{23}$$

where thus $\boldsymbol{R} = \boldsymbol{V}\boldsymbol{P}\boldsymbol{V}^T$. In this case it is possible to see that the regularized solution is

$$\dot{\boldsymbol{q}} = \boldsymbol{V}(\boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \boldsymbol{P})^{\#}\boldsymbol{\Sigma}^T\boldsymbol{U}^T\dot{\tilde{\boldsymbol{x}}}, \tag{24}$$



(a) (b)

Fig. 1 Example of task activation using only the SVO regularization as the second row of the task (28) is deactivated: a practical discontinuity can be seen in both (a) control vector $\dot{\boldsymbol{q}}$ and (b) task velocities $\dot{\boldsymbol{x}}$.

and that the projection matrix becomes

$$\boldsymbol{V}(\boldsymbol{I} - (\boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \boldsymbol{P})^{\#}\boldsymbol{\Sigma}^T\boldsymbol{\Sigma})\boldsymbol{V}^T. \tag{25}$$

Given the definition of $\boldsymbol{P}$ it is simple to see that only the directions with a small singular values, such that $p_{(i,i)} \neq 0$, are affected by the regularization. Finally, it is evident how with the use of the SVO regularization the matrix $(\boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \boldsymbol{P})$ is always of constant rank m. An application of such a regularization method can be found in [37].

3 Activation and Deactivation of Tasks

In this section, we present the methodology for activating and deactivating any of the scalar tasks composing a single multidimensional task. We first show how classical regularization methods are not sufficient to eliminate practical discontinuities, without greatly impacting on control performances. Instead, the introduction of a novel regularization, called task oriented, coupled with the SVO one and a final minimization on the control vector allows to eliminate any practical discontinuity. The extension to the hierarchy of priority levels is given in a later section.

3.1 Classical Regularization Methods

Let us now consider a scenario where, for some reason, one or more of the scalar tasks composing a multidimensional task start losing importance, even to the point where they should not be anymore considered. To represent this fact, we consider a scalar value $0 \leq a_{(i)} \leq 1$, associated to i-th scalar task, whose meaning is the following:

- $a_{(i)} = 1$ implies that the corresponding scalar task must be exactly assigned if possible, i.e. the goal is to have $\dot{x}_{(i)} = \dot{\bar{x}}_{(i)}$ (*active task*);
- $a_{(i)} = 0$ implies that the corresponding scalar task should not be considered, i.e. $\dot{x}_{(i)}$ should be unconstrained (*inactive task*);
- $0 < a_{(i)} < 1$ implies that $\dot{x}_{(i)}$ should smoothly evolve between the two previous cases (*task in transition*).

Then, the straightforward initial idea is to modify the problem (8) by inserting a weight diagonal matrix $\boldsymbol{A}$, whose diagonal elements are defined as above, as hereafter indicated

$$\min_{\dot{\boldsymbol{q}}} \left\| \boldsymbol{A}(\dot{\bar{\boldsymbol{x}}} - \boldsymbol{J}\dot{\boldsymbol{q}}) \right\|^2, \tag{26}$$

whose corresponding manifold of non-regularized solutions is

$$\dot{\boldsymbol{q}} = (\boldsymbol{A}\boldsymbol{J})^{\#}\boldsymbol{A}\dot{\bar{\boldsymbol{x}}} + (\boldsymbol{I} - (\boldsymbol{A}\boldsymbol{J})^{\#}\boldsymbol{A}\boldsymbol{J})\dot{\boldsymbol{z}},\ \forall \dot{\boldsymbol{z}}. \tag{27}$$

Equation (27) unfortunately exhibits discontinuities in the $\dot{\boldsymbol{q}}$ when $\boldsymbol{A}$ is varied, since the resulting weighted pseudo-inverse $(\boldsymbol{A}\boldsymbol{J})^{\#}\boldsymbol{A}$ is invariant to the weights on the rows of $\boldsymbol{J}$ that are linearly independent [18]. More specifically, the discontinuity occurs whenever a value of $A_{(i,i)}$ changes from 0 to $\epsilon > 0$ and vice versa.

To deal with this discontinuity problem, the use of the DLS and SVO regularizations can be therefore proposed, where the latter is certainly better, since it specifically acts only along the singular directions. However, in both cases, since there is not a straightforward relationship between the activation and the regularization damping values, this in turn requires either to have high damping values, which have a detrimental impact on the performances, or small ones, which do not prevent the issues with "practical" discontinuities, as highlighted in [32].

An example is reported in Fig. 1 for the case of

$$\boldsymbol{J} = \begin{bmatrix} -1 & -0.5 \\ 1 & 1 \end{bmatrix}, \tag{28}$$

with a velocity reference $\dot{\bar{\boldsymbol{x}}} = \begin{bmatrix} 0.1 & 0.1 \end{bmatrix}^T$ and by varying the value of $A_{(2,2)}$, i.e. activating/deactivating the second task, represented by the second row of $\boldsymbol{J}$.

Remark 1: Despite being continuous, the use of such rapidly varying control vectors and task velocities can easily create chattering phenomena in discrete control. See Fig. 8 and its related simulation in Section 6 or see the details in [32].

Remark 2: It is important to note how for a large part of the variation of $A_{(2,2)}$ the value of the control vectors are constant, due to the aforementioned reasons. This exemplifies the fact that a straightforward relationship between the activation and the regularization damping values does not actually exist. This in turn makes the transition between the solution where the second task is inactive ($\dot{\boldsymbol{q}} = \begin{bmatrix} -0.08 & -0.04 \end{bmatrix}^T$) and where it is active ($\dot{\boldsymbol{q}} = \begin{bmatrix} -0.3 & 0.4 \end{bmatrix}^T$) compressed in the small interval of variation between $A_{(2,2)} = 0.03$ and $A_{(2,2)} = 0.2$.

3.2 Task Oriented Regularization

Since solely imposing a weight $\boldsymbol{A}$ is insufficient to obtain the desired continuous behavior of activating and deactivating some rows of $\boldsymbol{J}$ without discontinuity, and that both the DLS and SVO regularizations, even if assuring a theoretical continuity, are however affected by practical discontinuities, the additional idea is therefore to modify the original minimization problem (26) by introducing a novel regularization, the here called *task oriented regularization*:

$$\min_{\dot{\boldsymbol{q}}} \left[\left\| \boldsymbol{A}(\dot{\bar{\boldsymbol{x}}} - \boldsymbol{J}\dot{\boldsymbol{q}}) \right\|^2 + \|\boldsymbol{J}\dot{\boldsymbol{q}}\|^2_{\boldsymbol{A}(\boldsymbol{I}-\boldsymbol{A})} \right]. \tag{29}$$

The above problem is a special instance of the general regularized problem (15), where the regularization matrix $\boldsymbol{R}$ results to be

$$\boldsymbol{R} = \boldsymbol{J}^T \boldsymbol{A}(\boldsymbol{I} - \boldsymbol{A})\boldsymbol{J}. \tag{30}$$

The rationale of this choice is that we want to preserve the rows that have their corresponding $A_{(i,i)} = 1$, while at the same time we want to introduce a regularization on those with $A_{(i,i)} < 1$. The choice of using $\boldsymbol{A}(\boldsymbol{I} - \boldsymbol{A})$ guarantees that a regularization is added only to the rows in transition. Indeed, it is easy to see that the cost vanishes for all the rows with $A_{(i,i)} = 0$ or $A_{(i,i)} = 1$. Thus, if $\boldsymbol{A}$ is made of only such values, the regularization cost vanishes and the solution just corresponds to the pseudo inverse of the active rows, as it will be evident in the following discussion.

The definition (30), substituted into (20), gives the following form for the manifold of regularized solutions

$$\begin{aligned} \dot{\boldsymbol{q}} &= (\boldsymbol{J}^T \boldsymbol{A}\boldsymbol{J})^{\#} \boldsymbol{J}^T \boldsymbol{A}\boldsymbol{A}\dot{\boldsymbol{x}} + (\boldsymbol{I} - (\boldsymbol{J}^T \boldsymbol{A}\boldsymbol{J})^{\#} \boldsymbol{J}^T \boldsymbol{A}\boldsymbol{A}\boldsymbol{J})\dot{\boldsymbol{z}} \\ &\triangleq \boldsymbol{\rho} + \boldsymbol{Q}\dot{\boldsymbol{z}}, \ \forall \dot{\boldsymbol{z}}. \end{aligned} \tag{31}$$

In order to analyze the properties of (31) we now need to to analyze the so called residual orthogonality properties of the projection matrix $\boldsymbol{Q}$, as hereafter reported in the next subsection.

3.2.1 Residual Orthogonality Properties of the Projector

In the classical task priority framework the projection operator is orthogonal, and guarantees that lower priority task can only act inside the null space of the higher priority ones. This ensures the invariance of the main task w.r.t. lower priority ones. However, in the here proposed case, the projection matrix is not orthogonal whenever any of the activation values are different from 0 and 1, i.e. whenever some task is in the transition zone. As we shall see, this is actually a positive fact, since the d.o.f. that are being released from the higher priority tasks can be exploited, at least partially, by the lower priority ones. For the time being, let us prove the fact that, under some assumptions, the projection matrix admits some residual orthogonality property: it is orthogonal with respect to the active rows of $\boldsymbol{J}$, i.e. $(\boldsymbol{J}\boldsymbol{Q})_{\{i\}} = \boldsymbol{0}$ for every i-th row for which $A_{(i,i)} = 1$.

To simplify the subsequent analysis, it is convenient to note that

$$\begin{aligned} &(\boldsymbol{J}^T \boldsymbol{A}\boldsymbol{J})^{\#} \boldsymbol{J}^T \boldsymbol{A} \\ =&(\boldsymbol{J}^T \sqrt{\boldsymbol{A}}\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#} \boldsymbol{J}^T \sqrt{\boldsymbol{A}}\sqrt{\boldsymbol{A}} \\ =&(\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#} \sqrt{\boldsymbol{A}}, \end{aligned} \tag{32}$$

where the identity (12) has been used. Substituting this relationship in the manifold (31) yields

$$\begin{aligned} \dot{\boldsymbol{q}} &= (\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#} \sqrt{\boldsymbol{A}}\boldsymbol{A}\dot{\boldsymbol{x}} + (\boldsymbol{I} - (\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#} \sqrt{\boldsymbol{A}}\boldsymbol{A}\boldsymbol{J})\dot{\boldsymbol{z}} \\ &\triangleq \boldsymbol{\rho} + \boldsymbol{Q}\dot{\boldsymbol{z}}, \ \forall \dot{\boldsymbol{z}}. \end{aligned} \tag{33}$$

Let us start the discussion by considering $A_{(i,i)} > 0, \forall i$ and $\boldsymbol{J}$ full rank, which implies that $\sqrt{\boldsymbol{A}}\boldsymbol{J}$ is full row rank. Let us compute $(\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#}$:

$$\begin{aligned}(\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#} &= \boldsymbol{J}^T\sqrt{\boldsymbol{A}}(\sqrt{\boldsymbol{A}}\boldsymbol{J}\boldsymbol{J}^T\sqrt{\boldsymbol{A}})^{-1} \\ &= \boldsymbol{J}^T\sqrt{\boldsymbol{A}}\sqrt{\boldsymbol{A}}^{-1}(\boldsymbol{J}\boldsymbol{J}^T)^{-1}\sqrt{\boldsymbol{A}}^{-1} \\ &= \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T)^{-1}\sqrt{\boldsymbol{A}}^{-1} \\ &= \boldsymbol{J}^{\#}\sqrt{\boldsymbol{A}}^{-1}\end{aligned} \tag{34}$$

which substituted into (33) gives

$$\begin{aligned}\dot{\boldsymbol{q}} &= \boldsymbol{J}^{\#}\boldsymbol{A}\dot{\tilde{\boldsymbol{x}}} + (\boldsymbol{I} - \boldsymbol{J}^{\#}\boldsymbol{A}\boldsymbol{J})\dot{\boldsymbol{z}} \\ &\triangleq \boldsymbol{\rho} + \boldsymbol{Q}\dot{\boldsymbol{z}},\ \forall \dot{\boldsymbol{z}}.\end{aligned} \tag{35}$$

It is easy to see that the projection matrix $(\boldsymbol{I} - \boldsymbol{J}^{\#}\boldsymbol{A}\boldsymbol{J})$ is actually orthogonal to the active rows. Indeed, multiplying the projector by $\boldsymbol{J}$ yields

$$\boldsymbol{J}(\boldsymbol{I} - \boldsymbol{J}^{\#}\boldsymbol{A}\boldsymbol{J}) = (\boldsymbol{J} - \boldsymbol{J}\boldsymbol{J}^{\#}\boldsymbol{A}\boldsymbol{J}) = (\boldsymbol{I} - \boldsymbol{A})\boldsymbol{J} \tag{36}$$

since $\boldsymbol{J}\boldsymbol{J}^{\#} = \boldsymbol{I}$ under the above assumptions. The above result implies that, for every row where $A_{(i,i)} = 1$, then $(\boldsymbol{J}\boldsymbol{Q})_{\{i\}} = \boldsymbol{0}$ as it was claimed. However, the projection matrix $\boldsymbol{Q}$ does not prevent $\dot{\boldsymbol{z}}$ from influencing the other rows. As said before, this is a positive fact, as those rows are being deactivated and thus there is not anymore any need to guarantee the fulfillment of their corresponding velocity reference. Indeed, when $A_{(i,i)}$ reaches zero, the corresponding velocity should be unconstrained.

Let us now drop the two assumptions of full rankness of $\boldsymbol{A}$ and $\boldsymbol{J}$. If $\boldsymbol{A}$ contains any value equal to zero in its diagonal, then the above formula cannot be used, since it contains $(\sqrt{\boldsymbol{A}})^{-1}$ which cannot be computed. Instead, without losing generality, let us suppose that the rows with $A_{(i,i)} = 0$ are at the bottom, and let us partition $\boldsymbol{A}$ and $\boldsymbol{J}$ in the following way

$$\boldsymbol{A} = \begin{bmatrix}\bar{\boldsymbol{A}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\ ;\ \boldsymbol{J} = \begin{bmatrix}\bar{\boldsymbol{J}} \\ \hat{\boldsymbol{J}}\end{bmatrix}, \tag{37}$$

where $\bar{\boldsymbol{J}}$ only contains the i-th rows for which $A_{(i,i)} \neq 0$.

Let us still compute the value of $(\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#}$. Using (13) let us write it as

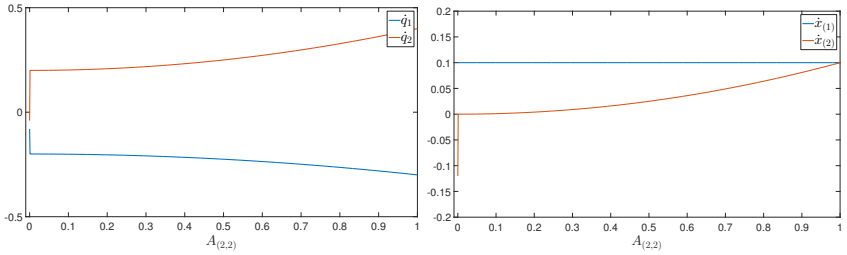
$$(\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#} = \boldsymbol{J}^T\sqrt{\boldsymbol{A}}(\sqrt{\boldsymbol{A}}\boldsymbol{J}\boldsymbol{J}^T\sqrt{\boldsymbol{A}})^{\#}. \tag{38}$$

Then using the definition (37) it is simple to compute

$$\sqrt{\boldsymbol{A}}\boldsymbol{J} = \begin{bmatrix}\sqrt{\bar{\boldsymbol{A}}}\bar{\boldsymbol{J}} \\ \boldsymbol{0}\end{bmatrix} \tag{39}$$

and

$$\sqrt{\boldsymbol{A}}\boldsymbol{J}\boldsymbol{J}^T\sqrt{\boldsymbol{A}} = \begin{bmatrix}\sqrt{\bar{\boldsymbol{A}}}\bar{\boldsymbol{J}}\bar{\boldsymbol{J}}^T\sqrt{\bar{\boldsymbol{A}}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix} \tag{40}$$



(a) (b)

Fig. 2 Example of task activation using only the task oriented regularization as the second row of the task (28) is deactivated: a discontinuity occurs near $A_{(2,2)} = 0$. History of (a) $\dot{\boldsymbol{q}}$ and (b) history of $\dot{\boldsymbol{x}}$ as $A_{(2,2)}$ is varied.

After some simple algebra, the formula for the pseudo-inverse $(\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#}$ becomes:

$$\begin{aligned}(\sqrt{\boldsymbol{A}}\boldsymbol{J})^{\#} &= \boldsymbol{J}^T\sqrt{\boldsymbol{A}}(\sqrt{\boldsymbol{A}}\boldsymbol{J}\boldsymbol{J}^T\sqrt{\boldsymbol{A}})^{\#} \\ &= \begin{bmatrix}\bar{\boldsymbol{J}}^{\#} & \boldsymbol{0}\end{bmatrix}\sqrt{\boldsymbol{A}}^{\#}\end{aligned} \tag{41}$$

which substituted into the control law (33) yields

$$\begin{aligned}\dot{\boldsymbol{q}} &= \begin{bmatrix}\bar{\boldsymbol{J}}^{\#} & \boldsymbol{0}\end{bmatrix}\boldsymbol{A}\dot{\tilde{\boldsymbol{x}}} + (\boldsymbol{I} - \begin{bmatrix}\bar{\boldsymbol{J}}^{\#} & \boldsymbol{0}\end{bmatrix}\boldsymbol{A}\boldsymbol{J})\dot{\boldsymbol{z}} \\ &\triangleq \boldsymbol{\rho} + \boldsymbol{Q}\dot{\boldsymbol{z}}, \forall \dot{\boldsymbol{z}}.\end{aligned} \tag{42}$$

This could not be otherwise, as the added cost in (29) vanishes whenever $\boldsymbol{A}$ is composed only by ones and zeros, and thus the obtained solution is just the pseudo inverse of $\boldsymbol{J}$ with only the relevant rows.

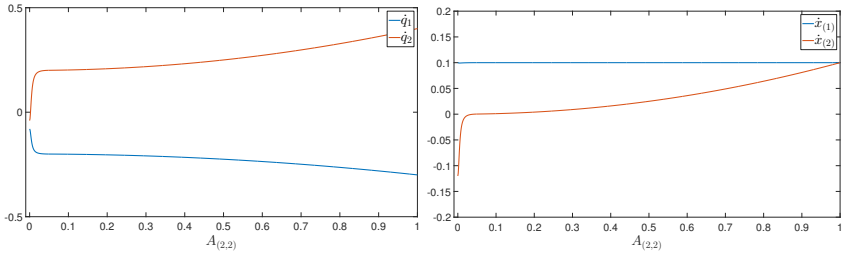
The projection matrix $(\boldsymbol{I} - \begin{bmatrix}\bar{\boldsymbol{J}}^{\#} & \boldsymbol{0}\end{bmatrix}\boldsymbol{A}\boldsymbol{J})$ is still orthogonal to the active rows. Indeed,

$$\begin{aligned}\boldsymbol{J}\boldsymbol{Q} &= (\boldsymbol{I} - \begin{bmatrix}\bar{\boldsymbol{J}} \\ \hat{\boldsymbol{J}}\end{bmatrix}\begin{bmatrix}\bar{\boldsymbol{J}}^{\#} & \boldsymbol{0}\end{bmatrix}\boldsymbol{A})\boldsymbol{J} \\ &= (\boldsymbol{I} - \begin{bmatrix}\bar{\boldsymbol{J}}\bar{\boldsymbol{J}}^{\#} & \boldsymbol{0} \\ \hat{\boldsymbol{J}}\bar{\boldsymbol{J}}^{\#} & \boldsymbol{0}\end{bmatrix}\boldsymbol{A})\boldsymbol{J} \\ &= \begin{bmatrix}(\boldsymbol{I} - \bar{\boldsymbol{J}}\bar{\boldsymbol{J}}^{\#}\bar{\boldsymbol{A}})\bar{\boldsymbol{J}} \\ (\boldsymbol{I} - \hat{\boldsymbol{J}}\bar{\boldsymbol{J}}^{\#}\bar{\boldsymbol{A}})\bar{\boldsymbol{J}}\end{bmatrix}.\end{aligned} \tag{43}$$

We have the following results:

- if a row i is linearly independent from all the others, and its $A_{(i,i)} = 1$, then $(\boldsymbol{J}\boldsymbol{Q})_{\{i\}} = \boldsymbol{0}$ as it was claimed;
- for a set of rows of $\boldsymbol{J}$ that are linearly dependent between each other, but linearly independent from the others, the orthogonality property holds if and only if all the corresponding $A_{(i,i)}$ are equal to one.

The clear advantage of the proposed regularization method is that the activation value is directly linked with the regularization itself, unlike in the SVO case. This allows to exploit the full range of variation of the activation value to perform the transition, clearly mitigating the possible practical discontinuities. Furthermore, the above analysis shows that the regularization is oriented along the task



(a) (b)

Fig. 3 Example of task activation using the task oriented regularization combined with the SVO one as the second row of the task (28) is deactivated: the discontinuity of Fig. 2 is removed however a practical one still persist in both (a) control vector $\dot{\boldsymbol{q}}$ and (b) task velocities $\dot{\boldsymbol{x}}$.

in transitions, preserving those that are active (whenever they are linearly independent from those in transition).

Before moving to the next section, we immediately note that using (31) an inevitable discontinuity occurs both in the minimum norm solution $\boldsymbol{\rho}$ and in the projection matrix $\boldsymbol{Q}$ whenever one of the $A_{(i,i)}$ becomes zero. This is clear when comparing its specialization (35) (all rows are active) and (42) (some rows are now deactivated). This is depicted in Fig. 2 where the same example (28) of Fig. 1 has been used.

3.3 Combining the Task Oriented and the SVO Regularizations

In this section we will show how to solve the problem of the residual discontinuity of the control law (31). The idea is simply that of combining the previously introduced task oriented regularization with the SVO one. As highlighted in Section 2.3.2, the SVO regularization ensures the theoretical continuity of the pseudo inverse, making the matrix to be inverted always of constant rank m. However, as stated in Section 3.2, the problem of using only the SVO regularization is that, in order to obtain a practical continuous transition, high values of the regularization parameters are actually necessary, with a clear impact on the performances of the control. Instead, in the previous sections we have shown how the task oriented regularization acts as soon as the activation is lower than one, thus smoothing the transition within the whole interval $0 \leq a_{(i)} \leq 1$, without impacting on the other rows, but without preventing the change of rank of the matrix to be inverted, which always occurs in the close vicinities of $a_{(i)} = 0$. The idea is thus to combine the task oriented regularization with the SVO one, thus combining the best of both regularizations:

- the task oriented regularization acts as soon as the task is being deactivated, thus immediately releasing its corresponding control directions, with a consequent immediate increase in the arbitrariness space of the solution in the whole interval $0 \leq a_{(i)} < 1$;
- the singular value oriented regularization instead ensures the continuity in the close vicinities of $a_{(i)} = 0$, even with small regularization parameter values, because of the contemporaneous presence of the task oriented one.

The minimization problem (29) thus is modified as follows

$$\min_{\dot{\boldsymbol{q}}} \left[\left\| \boldsymbol{A}(\dot{\bar{\boldsymbol{x}}} - \boldsymbol{J}\dot{\boldsymbol{q}}) \right\|^2 + \|\boldsymbol{J}\dot{\boldsymbol{q}}\|^2_{\boldsymbol{A}(\boldsymbol{I}-\boldsymbol{A})} + \left\| \boldsymbol{V}^T\dot{\boldsymbol{q}} \right\|^2_{\boldsymbol{P}} \right], \tag{44}$$

where $\boldsymbol{V}^T$ is the right orthonormal matrix of the SVD decomposition of $\boldsymbol{J}^T\boldsymbol{A}\boldsymbol{J} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, and $\boldsymbol{P}$ is the SVO regularization matrix described in Section 2.3.2. The manifold of solutions of the above problem can be thus written as

$$\begin{aligned} \dot{\boldsymbol{q}} &= (\boldsymbol{J}^T\boldsymbol{A}\boldsymbol{J} + \boldsymbol{V}^T\boldsymbol{P}\boldsymbol{V})^{\#}\boldsymbol{J}^T\boldsymbol{A}\boldsymbol{A}\dot{\bar{\boldsymbol{x}}} \\ &\quad + (\boldsymbol{I} - (\boldsymbol{J}^T\boldsymbol{A}\boldsymbol{J} + \boldsymbol{V}^T\boldsymbol{P}\boldsymbol{V})^{\#}\boldsymbol{J}^T\boldsymbol{A}\boldsymbol{A}\boldsymbol{J})\dot{\boldsymbol{z}} \\ &\triangleq \boldsymbol{\rho} + \boldsymbol{Q}\dot{\boldsymbol{z}},\ \forall \dot{\boldsymbol{z}}, \end{aligned} \tag{45}$$

and coincides with (31) whenever $\boldsymbol{P} = \boldsymbol{0}$.

From the above formula it is clear that the matrix $\boldsymbol{Q}$ evolves smoothly with $\boldsymbol{A}$, since the $\boldsymbol{V}^T\boldsymbol{P}\boldsymbol{V}$ regularization maintains the matrix to be inverted always of the same rank.

The addition of the SVO regularization to the task oriented one therefore allows to eliminate the residual discontinuity of Fig. 2 by transforming it into the practical one of Fig. 3, just emerging in the proximity of $a_{(i)} = 0$, due to the use of small SVO regularization parameters. In the following section we shall show how we can finally eliminate also such a practical discontinuity.

3.4 Minimization of the Control Vector

The main idea to eliminate any practical discontinuities is to exploit the arbitrariness of the solution $\dot{\boldsymbol{z}}$ in order to minimize the resulting control vector $\dot{\boldsymbol{q}}$. The rationale is simple. At the extreme values of $\boldsymbol{A}$ (i.e. its diagonal elements are either one or zero) the solution corresponds to the pseudo inverse of only the active rows, which is the minimum norm solution obtainable while fulfilling the given active tasks. Thus, the idea is to minimize the control vector as much as possible also during a task activation/deactivation transition, in order to smoothly join these two minimum-norm extrema. The above idea can be represented by the following minimization problem

$$\min_{\dot{\boldsymbol{z}}} \|\boldsymbol{\rho} + \boldsymbol{Q}\dot{\boldsymbol{z}}\|^2 \tag{46}$$

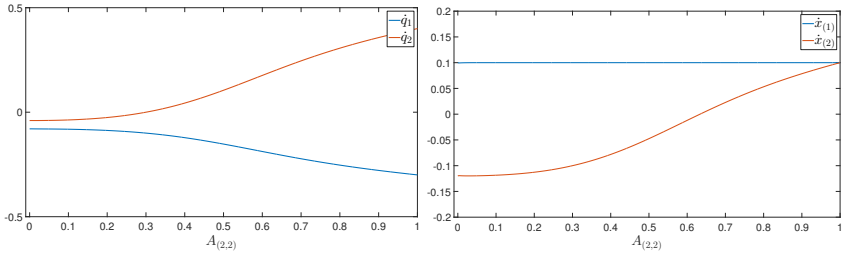
The solution of the above problem is simply

$$\dot{\boldsymbol{z}} = -\boldsymbol{Q}^{\#}\boldsymbol{\rho}, \tag{47}$$

which substituted in (45) gives

$$\dot{\boldsymbol{q}} = \boldsymbol{\rho} - \boldsymbol{Q}\boldsymbol{Q}^{\#}\boldsymbol{\rho}. \tag{48}$$

It is clear that the above formulation cannot be applied. Indeed, as soon as one eigenvalue of $\boldsymbol{Q}$ changes from zero to a value $\epsilon > 0$, the above solution would instantly cancel out any part of $\dot{\boldsymbol{q}}$ along the corresponding eigenvector direction.



(a) (b)

Fig. 4 Example of task activation using the proposed algorithm as the second row of the task (28) is deactivated: the resulting control vector and task velocity are now free of practical discontinuities (using $\eta = 10$). History of (a) control vector $\dot{\boldsymbol{q}}$ and (b) task velocities $\dot{\boldsymbol{x}}$.

To solve the aforementioned discontinuity, let us consider the following minimization problem in lieu of (46):

$$\min_{\dot{\boldsymbol{z}}} \left[\|\boldsymbol{\rho} + \boldsymbol{Q}\dot{\boldsymbol{z}}\|^2 + \eta \|(\boldsymbol{I} - \boldsymbol{Q})\dot{\boldsymbol{z}}\|^2 \right]. \tag{49}$$

The introduction of the second cost allows to penalize the use of the control directions characterized by an eigenvalue strictly less than one, due to the complementary nature of the eigenvalues of $\boldsymbol{Q}$ and $\boldsymbol{I} - \boldsymbol{Q}$. The scalar value $\eta > 0$ allows to balance the two needs. The solution of (49) is

$$\dot{\boldsymbol{z}} = -(\boldsymbol{Q}^T\boldsymbol{Q} + \eta(\boldsymbol{I} - \boldsymbol{Q})^T(\boldsymbol{I} - \boldsymbol{Q}))^{\#}\boldsymbol{Q}^T\boldsymbol{\rho}, \tag{50}$$

which substituted into the manifold $\boldsymbol{\rho} + \boldsymbol{Q}\dot{\boldsymbol{z}}$ yields

$$\begin{aligned} \dot{\boldsymbol{q}} &= \boldsymbol{\rho} - \boldsymbol{Q}(\boldsymbol{Q}^T\boldsymbol{Q} + \eta(\boldsymbol{I} - \boldsymbol{Q})^T(\boldsymbol{I} - \boldsymbol{Q}))^{\#}\boldsymbol{Q}^T\boldsymbol{\rho} \\ &= \left(\boldsymbol{I} - \boldsymbol{Q} \left(\boldsymbol{Q}^T\boldsymbol{Q} + \eta(\boldsymbol{I} - \boldsymbol{Q})^T(\boldsymbol{I} - \boldsymbol{Q}) \right)^{\#} \boldsymbol{Q}^T \right) \boldsymbol{\rho} \\ &\triangleq \boldsymbol{M}\boldsymbol{\rho}. \end{aligned} \tag{51}$$

Figure 4 shows the resulting final behaviour of the proposed algorithm as the same example (28) of the previous figures has been used. Summing up, for the example (28), we have:

- classical regularization methods such as SVO or DLS suffer of practical discontinuities, as highlighted in Fig. 1;
- the novel task oriented regularization has nice properties, as reported in Section 3.2.1, but has a discontinuity near zero as seen in Fig. 2;
- combining the task oriented and the SVO regularization allows to have a continuous, non-orthogonal projection matrix $\boldsymbol{Q}$ and to remove the discontinuity near $A_{1_{(2,2)}} = 0$, although issues of practical discontinuities arise and can be seen in Fig. 3;
- finally, the continuous, non-orthogonal projection matrix $\boldsymbol{Q}$ is exploited with a final minimization on the control vector $\dot{\boldsymbol{q}}$, and allows to join the two minimum norm extrema without practical discontinuities issues, as shown in Fig. 4.

Finally, let us remark how the properties that have been highlighted in the Section 3.2.1 hold only when $\boldsymbol{P} = \boldsymbol{0}$, i.e. when the corresponding SVO regularization is not active. The only drawback due to the introduction of the SVO regularization is that whenever it is active, it is not possible to ensure the invariance of the active tasks. However, since the required values for the SVO regularization can be chosen small, this drawback is almost negligible, as can be appreciated in Fig. 4. This indeed allows to reach a good compromise between simplicity of the algorithm, continuity of the obtained control and performances.

4 Extension to Task Priority Framework

In the previous section, the methodology for activating and deactivating rows of a single task has been presented, with the introduction of the so called task oriented regularization, the use of the SVO one and a final minimization on the control vector to ensure the continuity of the solution. In this section, we shall first tackle the possible discontinuity that can arise when a second level of priority is introduced, and then generalize the proposed framework to any number of priority levels.

4.1 Removing the Discontinuities in the Prioritized Control

Let us now consider another task that has to be executed with lower priority, represented by the following Jacobian relationship

$$\dot{\boldsymbol{x}}_2 = \boldsymbol{J}_2 \dot{\boldsymbol{q}}, \tag{52}$$

with $\boldsymbol{J}_2 \in \mathbb{R}^{m_2 \times n}$, $\dot{\boldsymbol{q}} \in \mathbb{R}^n$ and $\dot{\boldsymbol{x}}_2 \in \mathbb{R}^{m_2}$. The minimization for this second task must be performed taking into account that $\dot{\boldsymbol{q}}$ has been partially fixed by the higher priority task. The manifold of solutions of the first task is given by (45) and is $\dot{\boldsymbol{q}} = \boldsymbol{\rho}_1 + \boldsymbol{Q}_1 \dot{\boldsymbol{z}}_1$. Let us remark how, for the time being, we are not considering the minimization on the control vector, because that completely consumes any residual arbitrariness. We shall later see in Section 4.2 how that minimization is used in a hierarchy of tasks.

With that in mind, the second minimization problem can only exploit the arbitrariness of $\dot{\boldsymbol{z}}_1$, leading to

$$\begin{aligned} \min_{\dot{\boldsymbol{z}}_1} \Bigg[& \left\| \boldsymbol{A}_2 (\dot{\tilde{\boldsymbol{x}}}_2 - \boldsymbol{J}_2 \boldsymbol{Q}_1 \dot{\boldsymbol{z}}_1) \right\|^2 + \| \boldsymbol{J}_2 \boldsymbol{Q}_1 \dot{\boldsymbol{z}}_1 \|^2_{\boldsymbol{A}_2 (\boldsymbol{I} - \boldsymbol{A}_2)} \\ & + \left\| \boldsymbol{V}_2^T \dot{\boldsymbol{z}}_1 \right\|^2_{\boldsymbol{P}_2} \Bigg], \end{aligned} \tag{53}$$

with the definition $\dot{\tilde{\boldsymbol{x}}}_2 \triangleq \dot{\tilde{\boldsymbol{x}}}_2 - \boldsymbol{J}_2 \boldsymbol{\rho}_1$, and where the same task oriented and SVO regularizations have been employed to deal with the activation matrix $\boldsymbol{A}_2$.

Whenever a second level of priority is considered, a new source of possible discontinuities is represented by the non-orthogonal projection matrix $\boldsymbol{Q}_1$. To focus only on the discontinuities created by the projection matrix and to simplify the notation, let us suppose, without loss of generality, $\boldsymbol{A}_2 = \boldsymbol{I}$ and for the moment

let us neglect the presence of the SVO regularization. Then, the solution of the previous minimization is

$$\dot{\boldsymbol{z}}_1 = (\boldsymbol{J}_2\boldsymbol{Q}_1)^{\#}\dot{\tilde{\boldsymbol{x}}}_2 \tag{54}$$

which substituted into the first control law leads to

$$\boldsymbol{\rho}_2 = \boldsymbol{\rho}_1 + \boldsymbol{Q}_1(\boldsymbol{J}_2\boldsymbol{Q}_1)^{\#}\dot{\tilde{\boldsymbol{x}}}_2. \tag{55}$$

Note that $\boldsymbol{Q}_1(\boldsymbol{J}_2\boldsymbol{Q}_1)^{\#}$ is in actuality the weighted pseudo-inverse, with weights $\boldsymbol{Q}_1^{-1}$ on the control vector $\dot{\boldsymbol{q}}$ [36]. However, the above solution, while weighting the control directions, thus preferring to use those that have an eigenvalue of $\lambda_i = 1$ (i.e. unconstrained), fails under certain conditions. Indeed, there are cases where control directions with an eigenvalue $0 < \lambda_i < 1$ are treated as if $\lambda_i = 1$, because of possible invariance of the minimization with respect to the weights [18]. A very simple example can be constructed by considering $\boldsymbol{Q}_1 = \alpha\boldsymbol{I}$. It is easy to see that α cancels out in $\boldsymbol{Q}_1(\boldsymbol{J}_2\boldsymbol{Q}_1)^{\#}$ unless it is zero. This means that, even if the corresponding control direction only begins to be released by the higher priority tasks, the current priority level would consider it as totally free, with an inevitable discontinuity when it becomes zero.

To solve this problem, the idea is to compute a new task reference in lieu of $\dot{\tilde{\boldsymbol{x}}}_2$. In particular, we want to find which is the best velocity obtainable minimizing the use of control directions in transition. Toward that end, we exploit the following auxiliary problem

$$\begin{aligned}\min_{\dot{\boldsymbol{u}}_1} \Bigg[&\left\| \boldsymbol{A}_2(\dot{\tilde{\boldsymbol{x}}}_2 - \boldsymbol{J}_2\boldsymbol{Q}_1\dot{\boldsymbol{u}}_1) \right\|^2 + \|\boldsymbol{J}_2\boldsymbol{Q}_1\dot{\boldsymbol{u}}_1\|^2_{\boldsymbol{A}_2(\boldsymbol{I}-\boldsymbol{A}_2)} \\ &+ \eta \|(\boldsymbol{I} - \boldsymbol{Q}_1)\dot{\boldsymbol{u}}_1\|^2 + \left\| \hat{\boldsymbol{V}}_2^T\dot{\boldsymbol{u}}_1 \right\|^2_{\hat{\boldsymbol{P}}_2} \Bigg],\end{aligned} \tag{56}$$

where this time $\hat{\boldsymbol{V}}_2^T$ is the right orthonormal matrix of the SVD decomposition of

$$\boldsymbol{Q}_1^T\boldsymbol{J}_2^T\boldsymbol{A}_2\boldsymbol{J}_2\boldsymbol{Q}_1 + \eta(\boldsymbol{I} - \boldsymbol{Q}_1)^T(\boldsymbol{I} - \boldsymbol{Q}_1).$$

The corresponding task velocity is

$$\dot{\boldsymbol{x}}_2^* = \boldsymbol{J}_2\boldsymbol{Q}_1\dot{\boldsymbol{u}}_1 \triangleq \boldsymbol{W}_2\dot{\tilde{\boldsymbol{x}}}_2, \tag{57}$$

which is then used as a reference velocity in (53). We then have the following results:

- if $\dot{\boldsymbol{u}}_1$ is such that $\dot{\boldsymbol{x}}_2^* = \dot{\tilde{\boldsymbol{x}}}_2$, then $\dot{\boldsymbol{z}}_1$ will be the minimum norm solution that gives $\dot{\tilde{\boldsymbol{x}}}_2$, just as in (54). Furthermore, in this case $\boldsymbol{W}_2 = \boldsymbol{I}$;
- conversely, $\dot{\boldsymbol{z}}_1 = \dot{\boldsymbol{u}}_1$ and the obtained $\dot{\boldsymbol{x}}_2$ will necessarily differ from $\dot{\tilde{\boldsymbol{x}}}_2$ because not enough unconstrained control directions are available to obtain the desired velocity.

In practice, $\boldsymbol{Q}_1(\boldsymbol{J}_2\boldsymbol{Q}_1)^{\#}\boldsymbol{W}_2\dot{\tilde{\boldsymbol{x}}}_2$ operates in this way: first it finds the best $\dot{\boldsymbol{x}}_2^* = \boldsymbol{W}_2\dot{\tilde{\boldsymbol{x}}}_2$ that can be obtained minimizing the use of control directions in transition, and then it exploits the standard weighted pseudo-inverse to obtain the corresponding weighted minimum norm solution (i.e. $\boldsymbol{Q}_1(\boldsymbol{J}_2\boldsymbol{Q}_1)^{\#}\dot{\boldsymbol{x}}_2^*$).

4.2 Minimization of the Control Vector as the Final Task

In the previous section we have seen how to deal with a secondary task, exploiting (56) to cope with fact that the standard weighted pseudo inverse $\boldsymbol{Q}_1(\boldsymbol{J}_2\boldsymbol{Q}_1)^{\#}$ is insufficient.

Let us now go back to the minimization of the control vector presented in Section 3.3. It is clear that such a minimization can actually be seen as another task to be executed. This can be simply done considering $\boldsymbol{J}_2 = \boldsymbol{I}$, $\boldsymbol{A}_2 = \boldsymbol{I}$ and $\dot{\tilde{\boldsymbol{x}}}_2 = 0$. Then, we can see how (49) is just a special instance of (56).

Then, this task should be placed as the very last task of the hierarchy, consuming all the residual arbitrariness for minimizing the control vector, eliminating any practical discontinuities during task activations.

4.3 Unifying Formula for the Pseudo Inverse

Before proceeding to the extension to any number of priority levels, let us first introduce a more compact notation by introducing the operator $(\boldsymbol{X})^{\#,\boldsymbol{A},\boldsymbol{Q}}$ as in the following:

$$\boldsymbol{X}^{\#,\boldsymbol{A},\boldsymbol{Q}} \triangleq \left(\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X} + \eta(\boldsymbol{I}-\boldsymbol{Q})^T(\boldsymbol{I}-\boldsymbol{Q}) + \boldsymbol{V}^T\boldsymbol{P}\boldsymbol{V}\right)^{\#}\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{A} \tag{58}$$

where $\boldsymbol{V}$ is the right orthonormal matrix of the SVD decomposition of $\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X} + \eta(\boldsymbol{I}-\boldsymbol{Q})^T(\boldsymbol{I}-\boldsymbol{Q})$.

The task oriented regularization (44) can be simply obtained by writing $\boldsymbol{J}^{\#,\boldsymbol{A},\boldsymbol{I}}$ and the auxiliary problem (56) can be obtained by writing $(\boldsymbol{J}\boldsymbol{Q})^{\#,\boldsymbol{A},\boldsymbol{Q}}$.

4.4 Extension to Any Number of Priority Levels

Putting all the pieces together, and with the definition of the pseudo inverse given in the previous section, the extension to any number of priority levels is straightforward. With the initializations

$$\boldsymbol{\rho}_0 = \boldsymbol{0}, \quad \boldsymbol{Q}_0 = \boldsymbol{I}, \tag{59}$$

then for $k = 1, \ldots, p$, where p is the total number of priority levels:

$$\begin{aligned}
\boldsymbol{W}_k &= \boldsymbol{J}_k\boldsymbol{Q}_{k-1}(\boldsymbol{J}_k\boldsymbol{Q}_{k-1})^{\#,\boldsymbol{A}_k,\boldsymbol{Q}_{k-1}} \\
\boldsymbol{Q}_k &= \boldsymbol{Q}_{k-1}(\boldsymbol{I} - (\boldsymbol{J}_k\boldsymbol{Q}_{k-1})^{\#,\boldsymbol{A}_k,\boldsymbol{I}}\boldsymbol{J}_k\boldsymbol{Q}_{k-1}) \\
\boldsymbol{T}_k &\triangleq (\boldsymbol{I} - \boldsymbol{Q}_{k-1}(\boldsymbol{J}_k\boldsymbol{Q}_{k-1})^{\#,\boldsymbol{A}_k,\boldsymbol{I}}\boldsymbol{W}_k\boldsymbol{J}_k) \\
\boldsymbol{\rho}_k &= \boldsymbol{T}_k\boldsymbol{\rho}_{k-1} + \boldsymbol{Q}_{k-1}(\boldsymbol{J}_k\boldsymbol{Q}_{k-1})^{\#,\boldsymbol{A}_k,\boldsymbol{I}}\boldsymbol{W}_k\dot{\tilde{\boldsymbol{x}}}_k
\end{aligned} \tag{60}$$

thus ending up with the final control law

$$\dot{\boldsymbol{q}} = \boldsymbol{\rho}_p \tag{61}$$

because, according to Section 4.2, the p-th and final task should be the minimization of the control vector $\dot{\boldsymbol{q}}$, which consumes any residual arbitrariness.

5 Details on Activation Functions and Transitions

We have described in the previous sections the framework for activating and deactivating tasks, even when they are distributed within different priority levels, without issues of practical discontinuities. We started the discussion by referring to an activation value a for each row of such tasks that defines if a task is active, in transition or deactivated. This in turn implies if the corresponding reference signal should or should not be tracked exactly. We first show how the activation functions can be defined, and how they can be exploit to manage task transitions and implement inequality control objectives. Then a detailed discussion on the closed loop dynamics in task space whenever one or more tasks are in transition is presented, using simple geometrical tasks.

5.1 Activation Functions

Let us consider a multidimensional task and the activation value associated to each i-th of its components, called $a_{(i)}$, and let us construct it as the product of two functions

$$a_{(i)} \triangleq a^p_{(i)} a^s_{(i)}, \tag{62}$$

which have the following specific purposes:

- $a^s_{(i)}$ is used to activate/deactivate the task based on its internal state, i.e. the current value of the actual i-th component $x_{(i)}$;
- $a^p_{(i)}$ is instead used to activate/deactivate the task based on external parameters to the task itself, e.g. an activation based on the current elapsed mission time.

For each inequality control objective, we consider as activation function $a^s_{(i)}$ the one defined as follows for objectives of the type $x_{(i)} \leq x_{(i),M}$ (a similar function can be constructed for objectives $x_{(i)} \geq x_{(i),m}$):

$$a^s_{(i)} \triangleq \begin{cases} 1, & x_{(i)} > x_{(i),M} \\ s_i(x), & x_{(i),M} - \beta_{(i)} \leq x_{(i)} \leq x_{(i),M} \\ 0, & x_{(i)} < x_{(i),M} - \beta_{(i)} \end{cases} \tag{63}$$

where $s_i(x)$ is any sigmoid function with a continuous behaviour from 0 to 1 when $x_{(i),M} - \beta_{(i)} \leq x_{(i)} \leq x_{(i),M}$. The $\beta_{(i)}$ value allows to create a buffer zone, where the inequality is already satisfied, but the activation value is still greater than zero. This is necessary to prevent any chattering problem around the inequality control objective threshold. On the other hand, note that for equality control objectives it clearly holds that $a^i_{(i)} = 1$.

The activation value $a^p_{(i)}$ is instead a value which can be exploited to perform task transitions based on variables external to the task itself. For example, let us consider a mobile manipulator, moving on a horizontal plane, performing a grasping task. The end-effector position control (which is an equality control objective) should be activated only when the vehicle is sufficiently close to the object to be grasped, since it does not make sense to move the arm when the object is far away. Thus the activation function $a^p_{(i)}$ of the end-effector position control task would

be a function of the horizontal distance d between the vehicle and the object. This allows to surpass sequential approaches, where the vehicle is first commanded to get close and only successively the arm is commanded to grasp the object. Instead, the robot will accomplish both tasks, enabling the arm end-effector control smoothly as the vehicle approaches the object. A simulation trial implementing this seamless task transition is presented in Section 6.2.

Remark: We have implicitly considered objectives of the type $x_m < x < x_M$ as two separate ones. Note that if x_m and x_M are sufficiently spaced, i.e. $x_m+\beta < x_M-\beta$, then they can be considered together by using as activation function the sum of the two activation functions, and by choosing an arbitrary point inside the validity of both inequality to construct the common reference rate in (6). This is actually what is done for the joint limits task implementation, since the minimum and maximum limits satisfy the above conditions.

5.2 Discussion on Transitions

Let us for now discuss what happens when one or more tasks at the same or different priority levels are in transition. Toward that end, let us consider a simple two dimensional example. A point-wise robot moves on the plane (χ, γ) and has the following inequality control objectives to satisfy:

- objective 1: $\gamma > 3$;
- objective 2: $\chi > 1$;
- objective 3: $\gamma - \chi < 0$.

These objectives are grouped in two different priority levels:

- priority level 1: objective 1 and 2, with their respective velocity references stacked in the vector $\dot{\tilde{\boldsymbol{x}}}_1$ and their activation values in the matrix $\boldsymbol{A}_1$, where the subscript indicates the priority level as done in the previous sections;
- priority level 2: objective 3, with a velocity reference denoted as $\dot{\tilde{\boldsymbol{x}}}_2$ and activation matrix as $\boldsymbol{A}_2$.

The activation functions corresponding to these control objectives have been chosen with buffer zone size $\beta = 1$ (see Appendix A for the actual choice of sigmoid function).

In the first example the starting point of the robot is $(-4, 4.5)$. Figure 5(a) shows the trajectory of the robot on the (χ, γ) plane, as well as the inequality control objectives with a solid line and their corresponding buffer zone with a dashed line. Figure 5(b) reports the time behavior of the activation functions, while Fig. 5(c) reports the reference velocities for each task (dashed lines) and the actual task velocities (solid lines).

In the initial point of the first example, the first objective is already satisfied and the corresponding activation value is $A_{1_{(1,1)}} = 0$, while the second and third objectives have their corresponding activation values equal to one. At the same time, the velocities of the task 2 and 3 are tracked exactly, since the first task is completely deactivated. As the robot moves, it enters the buffer zone of the first task at approximately $t = 0.04$ s, which implies an activation of the task 1. For this reason, since the robot has only 2 d.o.f., the tracking of task 3 cannot be

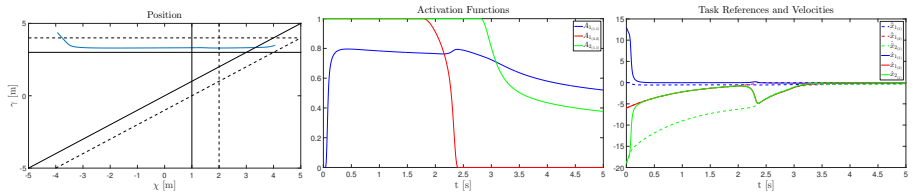


Fig. 5 Example 1: the robot initial position is $(-4, 4.5)$. (a) robot trajectory, where the solid lines represent the objectives boundaries and the dashed lines represent the thresholds at which the activation functions become different from zero (b) the time history of the activation functions (c) the reference and actual velocities of each of the three tasks.

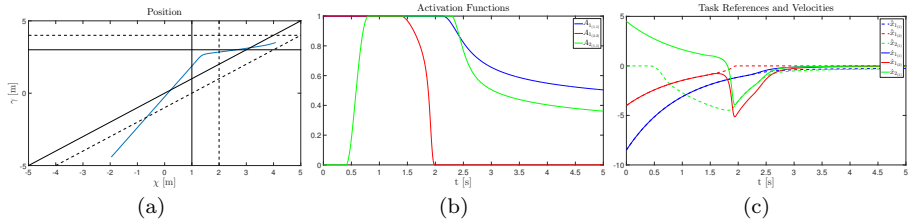


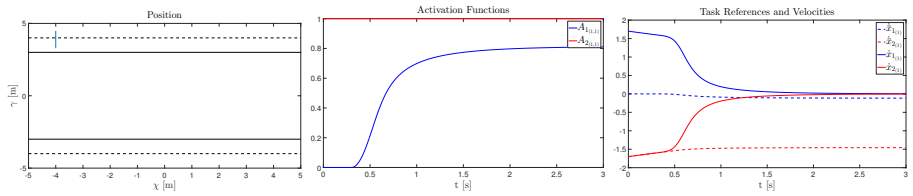
Fig. 6 Example 2: the robot initial position is $(-2, -4.5)$. (a) robot trajectory, where the solid lines represent the objectives boundaries and the dashed lines represent the thresholds at which the activation functions become different from zero (b) the time history of the activation functions (c) the reference and actual velocities of each of the three tasks.

anymore guaranteed, because it is at a lower priority level w.r.t. task 1 and 2. This is evident when looking at 5(c), since the reference $\dot{\tilde{x}}_{2_{(1)}}$ is not anymore tracked exactly. We further note how the tracking of the task 2 reference is still exactly fulfilled.

As the simulation progresses, task 1 remains in the transition zone and partially conflicts with task 3, for what concerns the motion on the γ axis. However, as can be noted from Fig. 5(a), a balance between the two tasks is reached and essentially the robot only moves on the χ axis, heading toward the achievement of both tasks 2 and 3. As soon as task 2 is completed (at approximately $t = 2.5$ s), the task 3 is again tracked exactly, despite task 1 is still in the transition zone.

In the second example the robot starts at $(-2, -4.5)$. As before, Figure 6(a) shows the trajectory of the robot on the (χ, γ) plane, as well as the inequality control objectives with a solid line and their corresponding buffer zone with a dashed line, Fig, 6(b) reports the time behavior of the activation functions, and Fig. 6(c) reports the reference velocities for each task (dashed lines) and the actual task velocities (solid lines).

In this case, tasks 1 and 2 are fully activated at the start, while task 3 is deactivated. Since tasks 1 and 2 have the highest priority, their velocity reference is tracked exactly, while the actual velocity of task 3 is imposed by the first two and actually drives the robot away from the region where the objective 3 is satisfied. Near $t = 0.4$ s, task 3 starts being activated, however as there are not enough d.o.f. available, its actual velocity remains the one set by task 1 and 2. However, as soon as task 2 enters its transition region at $t = 1.39$ s, then the actual velocity



(a) (b) (c)

Fig. 7 Example 3: the robot initial position is $(-4, 4.5)$. (a) robot trajectory, where the solid lines represent the objectives boundaries and the dashed lines represent the thresholds at which the activation functions become different from zero (b) the time history of the activation functions (c) the reference and actual velocities of each of the two tasks.

of task 3 begins to converge to its desired one. An exact tracking is accomplished at $t = 2.06$ s, when task 2 is finally completely deactivated.

A further example is proposed, where this time only two objectives are present and are clearly impossible to satisfy contemporaneously:

- objective 1: $\gamma > 3$;
- objective 2: $\gamma < -3$.

These two objectives are distributed within two priority levels. Such a situation should be clearly avoided in a real scenario, and is presented only for sake of a complete discussion.

At the start of the simulation the robot is placed in $(-4, 4.5)$, as can be seen in Fig. 7(a). In such a situation, $A1_{(1,1)} = 0$ and $A2_{(1,1)} = 1$, as can be seen in Fig. 7(b) thus the robot moves vertically towards the second objective. Furthermore, the velocity reference of the second task is tracked exactly, since the first task is inactive (see Fig. 7(c)). However, as soon as the robot enters the transition zone of the first task, the robot starts slowing down and actually stops in an equilibrium position where the first task is in the transition zone (and thus still $\gamma > 3$) and the second task is unsatisfied.

We conclude this preliminary discussion by noting how the proposed regularization method allows to share d.o.f. between the priority levels whenever tasks are in transition, without affecting the other *active* higher priority tasks. Furthermore, this sharing happens as soon as the activation values are different from zero and one, allowing to exploit the full extent of the transition, as opposed to SVO only regularization methods, generally only acting in a small zone of the transition interval. The first two examples shows that if a region where all the tasks are satisfied exists, then the system will evolve towards that zone. Instead, as the third example shows, if such a zone does not exist, then an equilibrium point within the transition zone will be established, without any chattering around it, in a configuration where the higher priority tasks are mainly satisfied.

Naturally, the fact that the task reference gains cannot be increased independently of the control sample time still holds true as in any discrete control system. In practice, once implemented in a discrete control system, the discontinuity-mitigation action which emerges from the proposed technique allows to have higher gains, and thus better control performances, before chattering or limit cycle problems arise when using SVO-only regularization solutions.

6 Simulation and Experimental Results

In this section we present some simulation and experimental results obtained using the proposed task-priority framework. The section is structured as follows:

- in Section 6.1 we present a simple case study, with the same 2-d.o.f. planar robot of [31]. We first show how with the use of classical regularization methods based only on SVO regularization the practical discontinuity issues generate chattering phenomena. Then, we compare our approach with the one provided in [31] showing similar results, despite our approach scaling linearly in the number of tasks;
- in Section 6.2 we move to the control of an underwater vehicle manipulator system. We first show the capability of the proposed framework to handle the activation and deactivation of tasks at any point of the hierarchy and within the same priority level. Then, another simulation is presented, showing an example of task transition for equality tasks. In particular, the end-effector position control (whose error must be zeroed) is activated only when the vehicle is actually within grasping distance;
- in Section 6.3 an experimental result obtained with the AMADEUS 7 d.o.f. manipulator is presented;
- finally, Section 6.4 presents a computational time analysis of the proposed framework.

6.1 Simple Case Study: 2-d.o.f. Planar Robot

In this first basic example the same 2-d.o.f. planar robot of [31] is employed, with joint limits equal to 1 and -1 for both joints. The starting position of the robot is $\boldsymbol{q} = [\ 0.3 \quad 0.1\ ]^T$ and the desired final position for the end-effector is $\bar{\boldsymbol{x}} = [\ 0.1 \quad 0.7\ ]^T$. Two tasks are implemented: the joint limit avoidance task and the end-effector position one.

The velocity reference for the joint limits avoidance task is simply

$$\dot{\bar{\boldsymbol{x}}}_{jl} = -\gamma_{jl}\boldsymbol{q}, \tag{64}$$

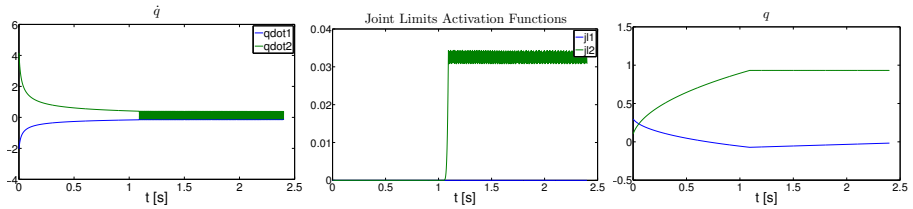
and it is activated by an activation function as defined in [31] eq. (55) with a buffer zone β of 0.1 rad. This means that the activation function is zero between $-0.9 \leq q_i \leq 0.9$ and then smoothly goes one for $0.9 \leq q_i \leq 1$ and the same for the negative part.

The position task is described by the following velocity reference

$$\dot{\bar{\boldsymbol{x}}}_{ee} = -\gamma_{ee}(\bar{\boldsymbol{x}} - \boldsymbol{x}). \tag{65}$$

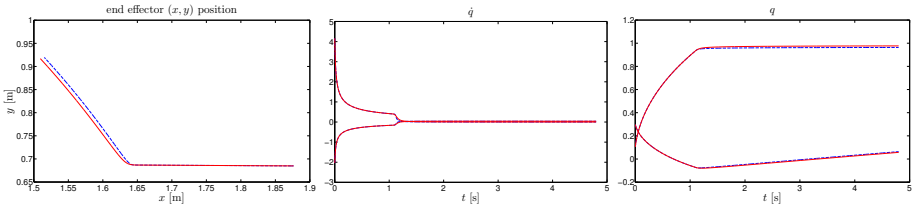
First we show how with the sole use of SVO regularization, the issues of practical discontinuities become chattering phenomena in discrete control. Figure 8(a) shows that as soon as the activation function reaches a critical value shown in Fig. 8(b), the control begins chattering around the threshold creating an undesired oscillating behaviour.

Then we proceed to compare our approach to the one presented in [31]. In particular, Fig. 9(a) shows the position of the end-effector obtained as result of the application of the control law [31] in a red solid line, while the proposed one



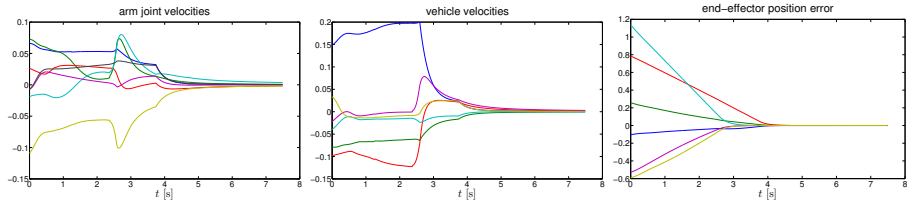
(a) (b) (c)

Fig. 8 Practical discontinuities create chattering phenomena around the activation thresholds: (a) the generated joint velocities, (b) the activation function value, (c) joints position.



(a) (b) (c)

Fig. 9 Comparison between the proposed approach (blue dashed line) and [31] (red solid line): (a) (x, y) position of the end-effector, (b) arm joints velocities, (c) joints position.



(a) (b) (c)

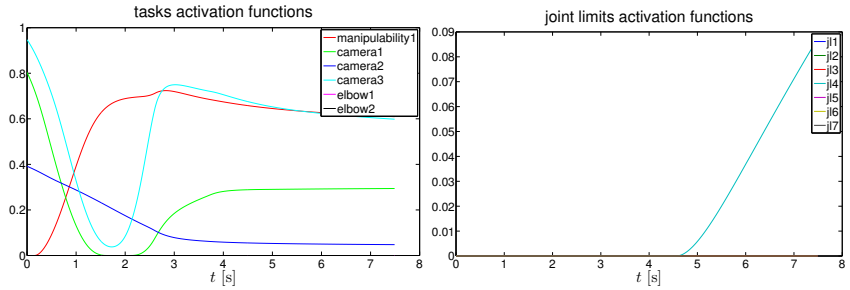
Fig. 10 Free floating vehicle simulation: (a) arm joints velocities, (b) vehicle velocities, (c) end-effector positioning error.

is presented with a dashed blue one. Figure 9(b) shows the resulting $\dot{\boldsymbol{q}}$, while the final Fig. 9(c) represents the time history of $\boldsymbol{q}$. As can be seen, the resulting behaviour is very similar. However, the approach in [31] scales exponentially and thus is difficult to apply as the number of tasks and d.o.f. increases.

6.2 Free Floating Underwater Manipulator

In this second simulation the reference scenario for the TRIDENT FP7 project [40] is studied. A free floating vehicle (6 d.o.f.) endowed with a redundant arm (7 d.o.f.) must recover a blackbox from the sea floor. The main task is thus represented by a position control of the end-effector (6 d.o.f.). However, the system must also simultaneously satisfy different safety and operational-enabling objectives [45]:

1. avoid the arm's joint limits (a 7 dimensional task, one for each joint);



(a) (b)

Fig. 11 Free floating vehicle simulation: (a) activation functions for manipulability, camera and elbow tasks, (b) activation functions for the joint limits task.

2. maintain a good arm dexterity to avoid singular postures (a scalar task);
3. keep the object grossly centered in the camera visual cone and keep the center of the object within given horizontal and vertical distances, to improve the performances of the visual tracking algorithm (a 3 dimensional task);
4. maintain the arm's elbow away from the camera visual cone, to avoid unnecessary occlusions of the object (a 2 dimensional task).

All these inequality control objectives would require 13 d.o.f. if their corresponding tasks are always active, as was done in the classical task-priority framework, which coupled with the main task of the end-effector position control would sum up to 19 d.o.f., surpassing the capabilities of the system. However, with the possibility of activating the tasks as seen in the previous sections, it becomes possible to activate these tasks only when strictly necessary, i.e. when the corresponding inequalities are not yet satisfied. Furthermore, with the proposed approach they can be given an higher priority w.r.t. to the position control of the end-effector, in order to guarantee their accomplishment and the respect of the corresponding inequality control objectives.

Figure 10(a) shows the arm's joint velocities and Fig. 10(b) shows the vehicle velocities obtained during the simulation, demonstrating the continuity of the control. Furthermore, Fig. 10(c) shows the six components of the position error of the end-effector, showing how it converges to zero.

Figure 11(a) reports the time history of the activation functions for the manipulability, camera and elbow tasks, while Fig. 11(b) reports the seven values of the activation functions of the joint limits task. As can be seen, different tasks are in transition during the trial. However, thanks to the proposed approach, the system manages to maintain the inequality tasks within the required thresholds, while at the same time accomplishing the final objective of the mission, by having the arm in the desired grasping position (see Fig. 10(c)).

The successive simulation is again performed with an underwater vehicle manipulator system. This simulation shows how with the method presented in Section 3 it is possible to activate and deactivate also equality tasks. For this simulation, the tasks are:

1. avoid the arm's joint limits (a 7 dimensional task, one for each joint);
2. maintain a good arm dexterity to avoid singular postures (a scalar task);

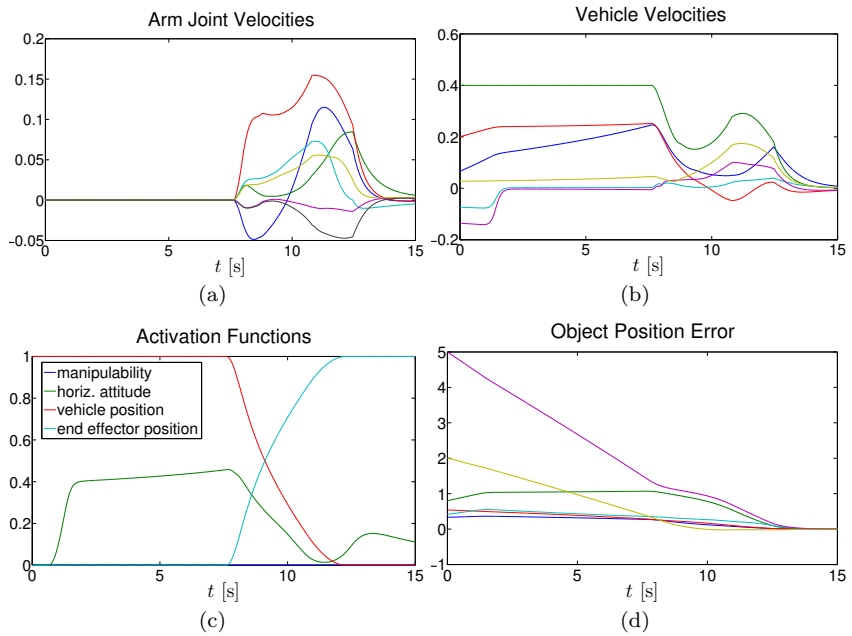


Fig. 12 Example of task transition: the end-effector position control is activated only when the vehicle is close to the object (a) time history of the joint velocities, (b) time history of the vehicle velocities (c) time history of the activation functions and (d) Cartesian position error between the end-effector and object frame.

3. keep the vehicle grossly horizontal (a scalar task)
4. precisely control the end-effector on top of the object frame (a 6 dimension task);
5. get the vehicle close to the object in a given position (a 6 dimension task).

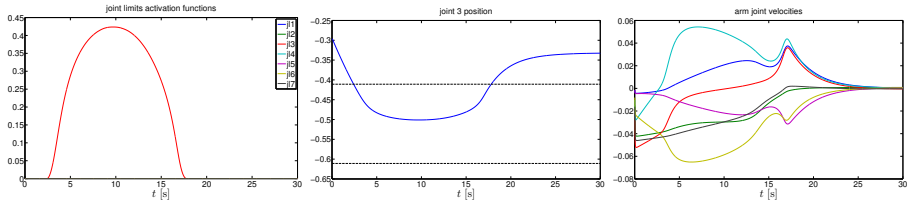
In particular the last two task are both of equality type. However, since it is not needed to move the arm while the object is away from the arm's availability, the end-effector position control is enabled only when the vehicle is close to the object to be grasped. At the same time, the vehicle position control is deactivated, since there is not a specific position needed.

Figure 12 shows the results of this simulation, presenting the joint and vehicle velocities generated, the activation functions showing this seamless task transition between getting close to the object and grasping it, while the last figure shows the convergence of the end-effector position control error. The simulation thus shows how sequential approaches can be implemented in a much smoother way.

6.3 Experimental Results with the AMADEUS Redundant Manipulator

In this section an experimental trial with the AMADEUS 7 d.o.f. manipulator (visible in Fig. 13) is shown. The tasks are, in order of priority: keeping away from joint limits, keeping the manipulability measure above a given threshold and finally reaching the desired end-effector position. The robot starts from an initial

Fig. 13 The AMADEUS 7 d.o.f. manipulator used during the experimental trials.



(a) (b) (c)

Fig. 14 AMADEUS experimental trial: (a) activation functions for the joint limits task, (b) values of the joint near its physical limit (c) time history of the joint velocities during the trial.

joint and Cartesian position which are, respectively,

$$\boldsymbol{q} = \begin{bmatrix} 0 \\ 0 \\ -0.31 \\ -1.08 \\ 0 \\ -0.63 \\ 0 \end{bmatrix}, \boldsymbol{x} = \begin{bmatrix} 1.568 \\ 0.008 \\ 1.742 \\ 1.197 \\ 0.001 \\ -0.999 \end{bmatrix}$$

where the Cartesian components represent the yaw, pitch, roll angles, followed by the x, y, z position of the end-effector. The robot is commanded to perform a diagonal movement on the horizontal plane, namely a -0.4 m movement along the x axis, and a 0.4 m one along the y axis, in order to reach the desired final position.

Figure 14(a) shows that during such a movement, one of the joint's arm approaches its end of race, causing the activation of the corresponding row in the joint limits avoidance task. Figure 14(b) shows the joint's position and the bounds for the start of task activation (-0.41 rad) and its complete activation (-0.61 rad). The figure thus show how the task successfully avoids the joint limit. The generated joint velocities are reported in Fig. 14(c), which are continuous and free of chattering phenomena even during the activation of the joint limit task.

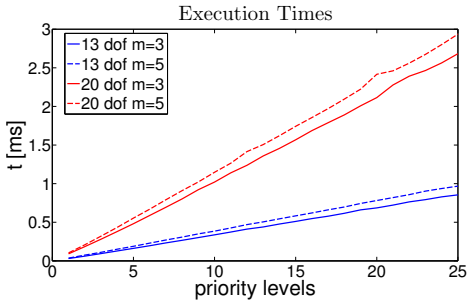


Fig. 15 Computation times as the number of priority levels (with m tasks for each level) is increased: with 13 d.o.f. the execution times with 25 priority levels is below 1 ms even with $m = 5$ tasks for each level, while with 20 d.o.f. is 2.68 ms with $m = 3$ and 2.98 ms with $m = 5$.

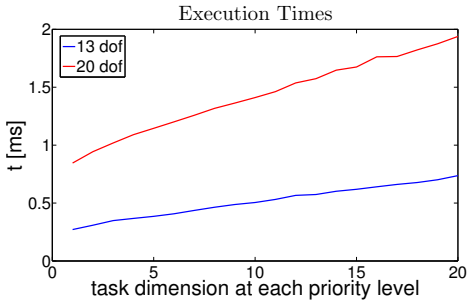


Fig. 16 Computation times as the number m of tasks for each level is increased, considering 10 priority levels.

6.4 Execution Times Analysis

The effectiveness from a computational point of view has been tested running the proposed algorithm on a test machine equipped with a Intel Core i7-4790 3.6 GHz CPU. Figure 15 shows the execution times as the number of priority level is increased, and each level is represented by a 3 or 5 dimensional task. The tests depicted in the figure have been done with 13 and 20 control variables, i.e. the number of d.o.f. available with an underwater vehicle manipulator system with one or two arms. The results show how with 13 d.o.f. the control can be run at 1 kHz even with 25 different priority levels with 5 dimensional task at each level, for a total of 125 scalar tasks. For a 20 d.o.f. system, a 1kHz control rate can be achieved with 9 priority levels each with 5 dimensional task for a total of 45 scalar tasks.

The successive Fig. 16 shows the linear scaling considering a fixed (10) amount of priority levels, as the dimension m of the tasks at each level is increased. The proposed approach can handle more than 200 scalar tasks divided equally at each priority level for a 13 d.o.f. system in less than 1 ms.

The final Fig. 17 shows the cubic scaling of the algorithm (due to the pseudo inverse) w.r.t. the number of d.o.f., considering a fixed (5) number of priority levels. The figure shows the computational times with different values of m. Despite the cubic scaling, the figure shows how the proposed algorithm can handle 100 scalar

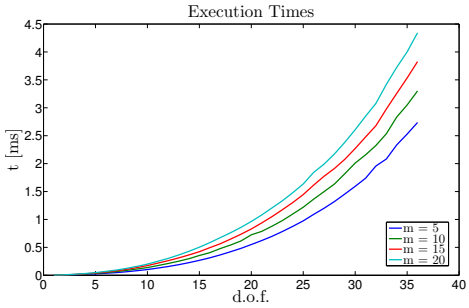


Fig. 17 Computation times as the number n of degrees of freedom is increased, considering 5 priority levels with different number m of task dimensions.

tasks for a 20 d.o.f. system in under 1 ms, while the number decreases to 25 scalar tasks for a 25 d.o.f. system.

Remark 1: For practical applications, this means that a mobile manipulator (3 + 7 d.o.f. for vehicle and redundant arm), a free floating vehicle manipulator system (6 + 7 d.o.f.) and dual arm systems (6 + 7 + 7 d.o.f.) are all capable to be controlled at 1 kHz, while considering at least 100 scalar tasks. It is important to stress the fact that the algorithm has a deterministic computational time, thus is perfectly suited for a real-time implementation. For these reasons, the proposed algorithm is being used by the authors in the context of the MARIS project [10] for the control of single, dual arm and cooperative underwater vehicle manipulator systems [30, 43, 44].

Remark 2: Finally, it must be noted that the implementation only uses one core of the CPU, leaving the other cores free to be used for other processes.

7 Conclusions

This paper has presented a novel framework for the task-priority control. The major novelty consists in the ability of activating and deactivating tasks without incurring into practical discontinuity problems, while at the same time keeping much of the simplicity of the original task-priority framework. Indeed, the proposed algorithm has a linear scaling in both the number of tasks and priority levels. Furthermore, it is deterministic and thus perfectly suited for a real-time implementation. Despite the cubic scaling in the number of degrees of freedom, we have shown that many redundant systems can be controlled up to 1 kHz with a significant number of tasks (at least 100).

The ability of activating and deactivating tasks has two main benefits. On the one hand it allows one to efficiently deal with inequality control objectives. Indeed, the corresponding task can be simply deactivated whenever inside the validity region of the inequality control objectives, without any over-constraining of the system mobility and without incurring in practical discontinuities or chattering problems. On the other hand, the same technique can be used to implement

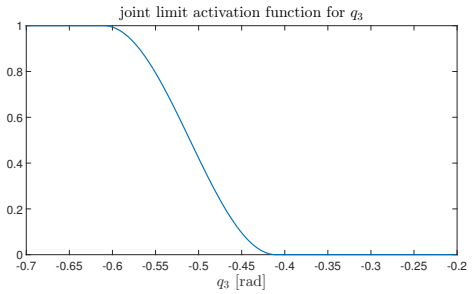


Fig. 18 Example of activation function for a joint limit task, where $q_3 > -0.66$

a temporal sequence of tasks. This is due to the fact that the mechanism for activating the tasks, i.e. the activation functions, can be also parametrized by some external variable (e.g. time, but also distance, etc.), providing an effective way to transition from a set of tasks to another. Future works may include the addition of velocity saturations in the prioritized control, in a similar manner as developed in [5].

A Appendix: Selection of Activation Functions

We have reported in Section 5 the general definition of the activation functions. We hereafter report the actual sigmoid function that we have used:

$$s_g(x) \triangleq \begin{cases} 1, & x < x_m \\ \frac{1}{2} * (\cos(\frac{(x-x_m)*\pi}{\beta}) + 1), & x_m \leq x \leq x_m + \beta \\ 0, & x > x_m + \beta \end{cases} \tag{66}$$

for objectives of the type $x > x_m$ and

$$s_l(x) \triangleq 1 - s_g(x) \tag{67}$$

for objectives of the type $x < x_M$.

We report here an example of activation function used in the implementation of the AMADEUS experiment presented in Section 6.3. Let us consider the joint limit task and in particular for the third joint lower the objective is

$$q_3 > -0.61$$

since the hard joint limit is -0.66 rad.

In this case the variable of interest is $x = q_3$, and the Jacobian of this task is simply $J_{jl3} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$. In accordance with the definitions given in Section 2, we have computed the task reference for q_3 as

$$\dot{\tilde{x}}_{jl_3} = \kappa(-0.41 - q_3),$$

where $\kappa > 0$ is a positive scalar gain. Then by using (66) with $x_m = -0.61$ and $\beta = 0.2$ we have that whenever $q_3 > -0.41$ the task is completely deactivated, $\dot{q}_3$ is freely used for other tasks and the fact that $\dot{\tilde{x}}_{jl_3}$ is negative does not matter. Figure 18 reports the specific activation function for this example.

The values of x_m and x_M can be tuned to increase or decrease the length of the transition, which can be important w.r.t. the velocities of the robot in the context of discrete control to avoid chattering phenomena: in practice, larger transitions are needed as the sample time of the control is increased.

References

1. Antonelli, G.: Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems. IEEE Transactions on Robotics **25**(5), 985–994 (2009)
2. Antonelli, G., Arrichiello, F., Chiaverini, S.: The null-space-based behavioral control for autonomous robotic systems. Intelligent Service Robotics **1**(1), 27–39 (2008)
3. Antonelli, G., Chiaverini, S.: Task-priority redundancy resolution for underwater vehicle-manipulator systems. In: Proc. IEEE International Conference on Robotics and Automation, vol. 1, pp. 768–773. Leuven, Belgium (1998). DOI 10.1109/ROBOT.1998.677070
4. Antonelli, G., Chiaverini, S.: Fuzzy redundancy resolution and motion coordination for underwater vehicle-manipulator systems. IEEE Trans. on Fuzzy Systems **11**(1), 109–120 (2003). DOI 10.1109/TFUZZ.2002.806321
5. Antonelli, G., Indiveri, G., Chiaverini, S.: Prioritized closed-loop inverse kinematic algorithms for redundant robotic systems with velocity saturations. In: Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, pp. 5892–5897. IEEE (2009)
6. Antonelli, G., Moe, S., Pettersen, K.: Incorporating set-based control within the singularity-robust multiple task-priority inverse kinematics. In: 23th Mediterranean Conference on Control and Automation, pp. 1132–1137. Torremolinos, Spain (2015)
7. Baerlocher, P., Boulic, R.: An inverse kinematics architecture enforcing an arbitrary number of strict priority levels. Visual Computer **20**(6), 402–417 (2004)
8. Ben-Israel, A., Greville, T.: Generalized inverses: theory and applications, vol. 15. Springer Verlag (2003)
9. Borst, C., Wimbock, T., Schmidt, F., Fuchs, M., Brunner, B., Zacharias, F., Giordano, P.R., Konietschke, R., Sepp, W., Fuchs, S., Rink, C., Albu-Schaffer, A., Hirzinger, G.: Rollin' justin - mobile platform with variable base. In: IEEE International Conference on Robotics and Automation, pp. 1597–1598 (2009)
10. Casalino, G., Caccia, M., Caiti, A., Antonelli, G., Indiveri, G., Melchiorri, C., Caselli, S.: Maris: A national project on marine robotics for interventions. In: Control and Automation (MED), 2014 22nd Mediterranean Conference of, pp. 864–869. IEEE (2014)
11. Casalino, G., Turetta, A., Sorbara, A., Simetti, E.: Self-organizing control of reconfigurable manipulators: a distributed dynamic programming based approach. In: ASME/IFToMM International Conference on Reconfigurable Mechanisms and Robots (ReMAR 2009), pp. 632–640. London, UK (2009)
12. Casalino, G., Zereik, E., Simetti, E., Torelli, S., Sperindé, A., Turetta, A.: Agility for underwater floating manipulation task and subsystem priority based control strategy. In: International Conference on Intelligent Robots and Systems (IROS 2012), pp. 1772–1779. Vilamoura, Portugal (2012). DOI 10.1109/IROS.2012.6386127
13. Casalino, G., Zereik, E., Simetti, E., Torelli, S., Sperindé, A., Turetta, A.: A task and subsystem priority based control strategy for underwater floating manipulators. In: IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles (NGCUV 2012), pp. 170–177. Porto, Portugal (2012). DOI 10.3182/20120410-3-PT-4028.00029
14. Cheng, G., Hyon, S., Morimoto, J., Ude, A., Hale, J.G., Colvin, G., Scroggin, W., Jacobsen, S.C.: Cb: A humanoid research platform for exploring neuroscience. Advanced Robotics **21**(10), 1097–1114 (2007)
15. Chiaverini, S.: Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. Robotics and Automation, IEEE Transactions on **13**(3), 398–410 (1997). DOI 10.1109/70.585902
16. Decre, W., Smits, R., Bruyninckx, H., De Schutter, J.: Extending iTaSC to support inequality constraints and non-instantaneous task specification. In: IEEE International Conference on Robotics and Automation, pp. 964–971. Kobe, Japan (2009)
17. Diftler, M.A., Mehling, J.S., Abdallah, M.E., Radford, N.A., Bridgwater, L.B., Sanders, A.M., Askew, R.S., Linn, D.M., Yamokoski, J.D., Permenter, F.A., Hargrave, B.K., Platt, R., Savely, R.T., Ambrose, R.O.: Robonaut 2 - the first humanoid robot in space. In: IEEE International Conference on Robotics and Automation, pp. 2178–2183 (2011)
18. Doty, K.L., Melchiorri, C., Bonivento, C.: Theory of generalized inverses applied to robotics. International Journal of Robotics Research **12**(1), 1–19 (1993)
19. Escande, A., Mansard, N., Wieber, P.B.: Hierarchical quadratic programming: Fast online humanoid-robot motion generation. The International Journal of Robotics Research **33**(7), 1006–1028 (2014). DOI 10.1177/0278364914521306

20. Faverjon, B., Tournassoud, P.: A local based approach for path planning of manipulators with a high number of degrees of freedom. In: Proc. IEEE Int. Conf. Robotics and Automation, vol. 4, pp. 1152–1159. Raleigh, NC, USA (1987)
21. Fink, J., Michael, N., Kim, S., Kumar, V.: Planning and control for cooperative manipulation and transportation with aerial robots. International Journal of Robotics Research **30**(3), 324–334 (2011)
22. Flacco, F., De Luca, A.: A reverse priority approach to multi-task control of redundant robots. In: Proc. IEEE International Conference on Intelligent Robots and Systems (2014)
23. Kanoun, O., Lamiraux, F., Wieber, P.B.: Kinematic control of redundant manipulators: generalizing the task-priority framework to inequality task. IEEE Transactions on Robotics **27**(4), 785–792 (2011)
24. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. The international journal of robotics research **5**(1), 90–98 (1986)
25. Khatib, O.: A unified approach for motion and force control of robot manipulators: The operational space formulation. Robotics and Automation, IEEE Journal of **3**(1), 43–53 (1987)
26. Lane, D.M., Davies, J.B.C., Casalino, G., Bartolini, G., Cannata, G., Veruggio, G., Canals, M., Smith, C., O'Brien, D.J., Pickett, M., Robinson, G., Jones, D., Scott, E., Ferrara, A., Angelleti, D., Coccoli, M., Bono, R., Virgili, P., Pallas, R., Gracia, E.: Amadeus: advanced manipulation for deep underwater sampling. IEEE Robot Autom Mag **4**(4), 34–45 (1997)
27. Lawson, C.L., Hanson, R.J.: Solving least squares problems, vol. 161. SIAM (1974)
28. Lee, J., Mansard, N., Park, J.: Intermediate desired value approach for task transition of robots in kinematic control. IEEE Transactions on Robotics **28**(6), 1260–1277 (2012)
29. Macjeiesky, A.A., Klein, C.A.: Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. International Journal of Robotic Research **4**(5), 109–117 (1985)
30. Manerikar, N., Casalino, G., Simetti, E., Torelli, S., Sperindé, A.: On autonomous cooperative underwater floating manipulation systems. In: International Conference on Robotics and Automation (ICRA 15), pp. 523–528. Seattle, WA (2015). DOI 10.1109/ICRA.2015.7139229
31. Mansard, N., Khatib, O., Kheddar, A.: A unified approach to integrate unilateral constraints in the stack of tasks. IEEE Trans. Robot. **25**(3), 670–685 (2009)
32. Mansard, N., Remazeilles, A., Chaumette, F.: Continuity of varying-feature-set control laws. IEEE Trans. on Automatic Control **54**(11), 2493–2505 (2009)
33. Marani, G., Choi, S.K., Yuh, J.: Underwater autonomous manipulation for intervention missions AUVs. Ocean Engineering **36**, 15–23 (2008)
34. Marani, G., Kim, J., Yuh, J., Chung, W.K.: A real-time approach for singularity avoidance in resolved motion rate control of robotic manipulators. In: Proc. IEEE International Conference on Robotics and Automation ICRA '02, vol. 2, pp. 1973–1978 (2002). DOI 10.1109/ROBOT.2002.1014830
35. Marani, G., Kim, J., Yuh, J., Chung, W.K.: Algorithmic singularities avoidance in task-priority based controller for redundant manipulators. In: Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on, vol. 4, pp. 3570–3574 vol.3 (2003). DOI 10.1109/IROS.2003.1249709
36. Nakamura, Y.: Advanced Robotics: Redundancy and Optimization. Addison Wesley (1991)
37. Nakamura, Y., Hanafusa, H.: Inverse kinematic solutions with singularity robustness for robot manipulator control. Journal of Dynamic Systems, Measurement and Control **108**(3), 163–171 (1986)
38. Nenchev, D., Sotirov, Z.: Dynamic task-priority allocation for kinematically redundant robotic mechanisms. In: IEEE/RSJ/GI International Conference on Intelligent Robots and Systems(IROS 94), vol. 1, pp. 518–524 (1994). DOI 10.1109/IROS.1994.407429
39. Padir, T.: Kinematic redundancy resolution for two cooperating underwater vehicles with on-board manipulators. In: Proc. IEEE Int Systems, Man and Cybernetics Conf, vol. 4, pp. 3137–3142. Waikolo, HI, USA (2005)
40. Sanz, P., Ridao, R., Oliver, G., Casalino, P., Insaurralde, C., Silvestre, C., Melchiorri, M., Turetta, A.: Trident: Recent improvements about autonomous underwater intervention missions. In: Proceedings of the IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles (NGCUV2012), Porto, Portugal (2012)
41. Sentis, L., Khatib, O.: Control of free-floating humanoid robots through task prioritization. In: Proceedings of the 2005 IEEE International Conference on Robotics & Automation, pp. 1718–1723. Barcelona, Spain (2005)

42. Siciliano, B., Slotine, J.J.E.: A general framework for managing multiple tasks in highly redundant robotic systems. In: Proc. Fifth Int Advanced Robotics 'Robots in Unstructured Environments', 91 ICAR. Conf, pp. 1211–1216. Pisa, Italy (1991)
43. Simetti, E., Casalino, G.: Whole body control of a dual arm underwater vehicle manipulator system. Annual Reviews in Control (2015). DOI 10.1016/j.arcontrol.2015.09.011. Accepted
44. Simetti, E., Casalino, G., Manerikar, N., Torelli, S., Sperindé, A., Wanderlingh, F.: Cooperation between autonomous underwater vehicle manipulations systems with minimal information exchange. In: IEEE/MTS OCEANS 2015. Genova, Italy (2015)
45. Simetti, E., Casalino, G., Torelli, S., Sperindé, A., Turetta, A.: Floating underwater manipulation: Developed control methodology and experimental validation within the trident project. Journal of Field Robotics **31**(3), 364–385 (2014). DOI 10.1002/rob.21497
46. Simetti, E., Casalino, G., Torelli, S., Sperindé, A., Turetta, A.: Underwater floating manipulation for robotic interventions. In: IFAC World Congress 2014, pp. 3358–3363 (2014). DOI 10.3182/20140824-6-ZA-1003.00503
47. Simetti, E., Turetta, A., Casalino, G.: Distributed control and coordination of cooperative mobile manipulator systems. In: H. Asama, H. Kurokawa, J. Ota, K. Sekiyama (eds.) Distributed Autonomous Robotic Systems 8, pp. 315–324. Springer Berlin Heidelberg (2009). DOI 10.1007/978-3-642-00644-9_28
48. Sugiura, H., Gienger, M., Janssen, H., Goerick, C.: Real-time collision avoidance with whole body motion control for humanoid robots. In: Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems IROS 2007, pp. 2053–2058. San Diego, CA, USA (2007)
49. Wampler, C.W.: Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. Systems, Man and Cybernetics, IEEE Transactions on **16**(1), 93–101 (1986)
50. Yoshikawa, T.: Analysis and control of robot manipulators with redundancy. In: M. Brady, R.Paul (eds.) Robotic Research: The First International Symposium, pp. 735–747. MIT Press (1984)
51. Zereik, E., Sorbara, A., Merlo, A., Simetti, E., Casalino, G., Didot, F.: Space robotics supporting exploration missions: Vision, force control and coordination strategy for crew assistants. Intelligent Service Robotics **4**(1), 39–60 (2011). DOI 10.1007/s11370-010-0084-1