Generalized Hierarchical Control

Mingxing Liu, Yang Tan, Vincent Padois

Generalized Hierarchical Control

Mingxing Liu, Yang Tan, and Vincent Padois

Most existing techniques to handle strict task priorities in hierarchical control are based on null-space projectors or a sequence of quadratic programs; whereas non strict task priorities are usually handled by optimization based on a weighting strategy. This paper proposes a novel approach to handle both strict and non-strict priorities of an arbitrary number of tasks, and to achieve multiple priority rearrangements simultaneously. A generalized projector, which makes it possible to completely project a task into the null-space of a set of tasks, while partially projecting it into the null-space of some other tasks, is developed for priority modulation. Priority transitions are achieved by smooth variations of the generalized projector. The control input is computed by solving one quadratic programming problem, where generalized projectors are adopted to maintain a task hierarchy, and constraints can be implemented (e.g. dynamic equilibrium, actuation capabilities, joint limits, obstacle avoidance, contact constraints, etc.). The effectiveness of this approach is demonstrated on a simulated robotic manipulator in a dynamic environment.

Index Terms—**Redundant robots, task hierarchy, priority switching, dynamics, torque-based control.**

I. Introduction

Redundant robots, such as humanoids, are nowadays expected to perform complex missions in weakly structured environments (*e.g.* human environments, construction sites, nuclear dismantling zones, etc.). Even though robot redundancy makes it possible for these robots to perform multiple tasks simultaneously, task conflicts may still occur when all the task objectives cannot be satisfied at the same time. In order to handle conflicts, tasks are usually assigned with different priority levels. Therefore, to control a complex robotic system the controller must be able to handle multiple prioritized tasks simultaneously and to respect various constraints imposed by the robot body (joint limits, actuation capabilities, etc.) and the environment (contacts to maintain, obstacles to avoid, etc.).

A large number of hierarchical control frameworks are presented in the robotics literature for the management of multiple operational task objectives.

- Some of them deal with *strict task hierarchies*, such as analytical methods based on null-space projectors [1]–[5] and hierarchical quadratic programming approaches [6,7]. These approaches can ensure that critical tasks are fulfilled with higher priorities and lower-priority tasks are performed only in the null-space of higher priority tasks.
- Other approaches handle *non-strict task hierarchies*, such as those using weighting strategies [8]–[12]. In a non-strict task hierarchy, a lower priority task is not restricted in the null-space of higher priority tasks, thus it may still affect their performances. The solution of these approaches is a compromise among task objectives of different weights.

In a more general context, the robot may need to deal with both strict and non-strict hierarchies. Moreover, for robots acting in dynamically changing contexts, non-strict priorities between tasks may become strict ones and task priorities may have to be switched in order to cope with changing situations.

With the aim of handling both strict and non-strict hierarchies simultaneously, a novel control framework called Generalized Hierarchical Control (GHC) is presented in this paper. The contributions of this work are as follows: 1) the development of a generic dynamic control framework, which solves a single quadratic program (QP) to account for an arbitrary number of strict and non-strict task priorities; 2) the development of a generalized projector, which ensures desired task priorities, their transitions as well as an elegant way of inserting and deleting tasks among those to be performed. The implementation of such a projector is not restricted to the dynamic control framework presented in this paper. In fact, it can be implemented in many analytical and optimization-based control frameworks. Task priorities can be handled by the modulation of a priority matrix, without the necessity of modifying the control problem formulation each time the priorities change.

This paper is organized as follows. Related works are described in Section II. The robot model considered in this paper and the tasks and constraints to be handled by the controller are presented in Section III. The GHC framework is developed in Section IV, where detailed explanations of the generalized projector are provided. Some experimental results are presented in Section V to demonstrate the framework capabilities, and comparisons with the results using some other approaches are provided. Several characteristics of this framework and future research directions regarding the computational aspect and the potential application of the proposed approach are presented in Section VII.

II. Related Works

Approaches to maintain a desired task hierarchy using a multi-objective controller draw a lot of interest. This Section reviews some classical types of hierarchical control frameworks, as well as the methods for priority transitions within these frameworks.

A. Approaches for handling a strict hierarchy

Analytical methods based on null-space projections can ensure that lower priority tasks are executed only in the null-space of higher-priority tasks, by means of the appropriate design of a null-space projector [13]. Such an idea is applied

The authors are with Institut des Systèmes Intelligents et de Robotique, CNRS UMR 7222 & Université Pierre et Marie Curie, 4 place Jussieu, 75252 Paris cedex 05, France. e-mail: {liu, tan, padois}@isir.upmc.fr

in prioritized inverse kinematics [2,14], in acceleration based control [3,4], and in joint torque based control [1,5,15]. A generic framework, from which several existing control laws can be derived, is presented in [16]. Projected inverse dynamics schemes are developed for constrained systems in [17,18], where the dynamics equation is projected into the null-space of the Jacobian of constraint equations.

Inequality constraints are usually difficult to be directly dealt with in analytical approaches using pseudo-inverses and projection matrices. A common method is to transform inequality constraints into task objectives by applying artificial potential fields [19], from which repulsive forces are derived to prevent the robot from entering into activation zones of the inequality constraints [19]–[24]. However, performing these tasks cannot guarantee that these inequality constraints are actually met. The approach presented in [25] integrates unilateral constraints at any priority level, albeit time consuming. The algorithm introduced in [4,26] proposes to disable the most critical joint and redistribute joint motion commands to guarantee the satisfaction of some hard bounds of joint variables. However, this algorithm deals with inequality constraints only at the joint level. Furthermore, the optimal solution satisfying the control problem may require the movement of a joint which has unfortunately been disabled.

To deal with prioritized inequality constraints more easily, hierarchical quadratic programming (HQP) approaches use numerical QP solvers to solve a Hierarchical Quadratic Program [6]. The idea of HQP is to first solve a QP to obtain a solution for a higher priority task objective; and then to solve another QP for a lower priority task, without increasing the obtained minimum of the previous task objective. This prioritization process corresponds to solving lower-priority tasks in the null-space of higher-priority tasks while trying to satisfy lower-priority tasks at best. The HQP algorithm is applied for solving prioritized inverse dynamics [7] and is also applied to whole-body motion control under unilateral constraints [27]. It requires to solve as many QPs as priority levels, which can be quite time consuming. The computation cost of hierarchical inverse kinematics with inequality constraints is improved by an algorithm developed in [28], which permits real time control of a humanoid robot.

Generally, for an approach based on strict hierarchy, the relative importance of one task with respect to another one of different priority level is parametrized in a binary way: either strictly higher or strictly lower. However, in many contexts, organizing tasks by assigning them with strict priorities is not generic, *i.e.* can have some limitations. First, a strict priority is just an extreme case of the relations of task importance levels. In fact, a task may not always have a strict priority over another one and it is usually difficult to define a strict hierarchy among a set of tasks. Second, strict priorities can sometimes be too conservative so that they may completely block lower-priority tasks. Unlike a discrete parametrization of task priorities, a continuous parametrization is richer and more informative. Therefore, this work handles task priorities, which can be strict or non-strict, by using a continuous parametrization. Moreover, priorities are defined here by pairs of tasks: this choice extends the classical notion of priority in Robotics while still making it possible to represent standard lexicographic orders as defined in [27].

B. Approaches for handling a non-strict hierarchy

Non-strict priorities are usually handled by control approaches using weighting strategies [8]–[11,29]. These control frameworks solve all the constraints and task objectives in one QP and provide a trade-off among task objectives with different importance levels. As the performances of higher priority tasks cannot be guaranteed by simply adjusting the weights of task objectives, a prioritized control framework is proposed in [12] to ensure the performance of a higher-priority task within a user defined tolerance margin. However, this approach handles priorities of only two levels. In approaches based on weighting strategies, task priorities can be parametrized continuously. Nonetheless, even though the work in [30] on soft constraints in model predictive control could probably be adapted to provide a way to reach the extreme case of strict priorities, the existing robotic applications of these frameworks do not extend to strict hierarchies.

The control framework proposed in this paper is based on these frameworks: it formulates and solves all tasks and constraints in one QP. It also largely outperforms them by permitting priorities to change continuously from a non-strict case to a strict case.

C. Task transitions

Earlier versions of analytical methods and HQP approaches can ensure strict priorities among tasks; however, a change in the task set, such as a switch of task priorities, may lead to discontinuity. Recently, different methods have been developed to handle task transition problems. An approach to smooth priority rearrangement between two tasks is proposed in [31,32]. Approaches for continuous and simultaneous transitions of multiple tasks are developed in [33,34]. A specific inverse operator is proposed in [33] to ensure continuous inverse in the analytical computation of control laws. The approach presented in [34] is based on intermediate desired values in the task space. When the number of task transitions increases, this approach suggests to apply an approximation to reduce the computational cost. An approach of hierarchical control with continuous null-space projections is presented in [35]. In this approach, an activator associated to directions in the right singular vectors of a task Jacobian matrix is regulated to activate or deactivate these directions. However, the design of such an activator makes this approach difficult to be implemented for the separate handling of different task directions. On the other hand, task transitions can be easily achieved within a non-strict hierarchy by the continuous variation of task weights [29].

The control framework proposed in this paper allows an arbitrary number of task priority transitions. This framework uses continuous priority parametrization, and the extreme case of strict priorities can be achieved. The idea to achieve this goal is based on the construction of a novel generalized projection matrix, which regulates to what extent a lower-priority task is projected into the null-space of a higher-priority task. In other words, this generalized projector allows a task to be completely, partially, or not at all projected into the null-space of some other tasks. The priority levels can be changed by the simple modulation of the generalized projector. The implementation of this generalized projector in multi-objective control frameworks based on optimization provide them with a mechanism to regulate task priorities more precisely, so that both strict and non-strict priorities can be handled by solving only one optimization problem.

III. Modeling

Consider a robot as an articulated mechanism with n degrees of freedom (DoF) including n_a actuated DoF. The dynamics of the robot in terms of its generalized coordinates $\boldsymbol{q} \in \mathbb{R}^n$ is written as follows

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = J_c(\boldsymbol{q})^T \boldsymbol{\chi}, \tag{1}$$

where $M(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is the generalized inertia matrix; $\dot{\boldsymbol{q}} \in \mathbb{R}^n$ and $\ddot{\boldsymbol{q}} \in \mathbb{R}^n$ are the vector of velocity and the vector of acceleration in generalized coordinates, respectively; $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^n$ is the vector of Coriolis, centrifugal and gravity induced joint torques; $\boldsymbol{\chi} = \begin{bmatrix}\boldsymbol{w}_c^T & \boldsymbol{\tau}^T\end{bmatrix}^T$ is the vector of the actuation torques ($\boldsymbol{\tau} \in \mathbb{R}^{n_a}$) and the external contact wrenches applied to the robot ($\boldsymbol{w}_c = \begin{bmatrix}\boldsymbol{w}_{c,1}^T & \dots & \boldsymbol{w}_{c,n_c}^T\end{bmatrix}^T$), with n_c the number of contact points; $J_c(\boldsymbol{q})^T = \begin{bmatrix}J_{c,1}(\boldsymbol{q})^T & \dots & J_{c,n_c}(\boldsymbol{q})^T & S(\boldsymbol{q},\dot{\boldsymbol{q}})^T\end{bmatrix}$ is the transpose of a Jacobian matrix, with $J_{c,n_\beta}(\boldsymbol{q})$, the Jacobian matrix associated to a contact point β and $S(\boldsymbol{q},\dot{\boldsymbol{q}})^T \in \mathbb{R}^{n\times n_a}$, a selection matrix for the actuated DoF. In the control problem considered in this paper, the vector $\boldsymbol{\chi}$ is called the **action variable**.

A. Task definition

A task in Robotics can be defined as a function of the considered robotic systems [36,37]. This function relates the control level in operational/task space, to the control level in joint space. More specifically, consider a robot controlled by joint torques at the dynamics level[1], a task i can be defined by the following characteristics:

- A physical frame $\mathcal{F}_i$, *i.e.* a frame attached to a part of the robot body that should be controlled for performing an operational task.
- An associated **task variable** $\boldsymbol{\xi}_i \in \mathbb{R}^{m_i}$ that can be expressed in terms of some high level goals to be achieved by the frame $\mathcal{F}_i$ in the task space, such as a desired position or orientation. m_i is the dimension of a task i.
- A forward model linearly relating the second order derivative of the vector of generalized coordinates to that of the task variable for a given state $(\boldsymbol{q}, \dot{\boldsymbol{q}})$

$$\ddot{\boldsymbol{\xi}}_i = J_i(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{J}_i(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \tag{2}$$

where $J_i(\boldsymbol{q})$ is the Jacobian matrix, *i.e* the differential kinematics mapping from joint space to task space, and $\dot{J}_i(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ is the task space drift vector.

[1]The velocity kinematics version of this problem can be trivially derived from this more general case.

- A local controller $\boldsymbol{r}_i$, the goal of which is to correct task errors and ensure the convergence of the task variable $\boldsymbol{\xi}_i$ towards its desired trajectory $\boldsymbol{\xi}_i^\star$

$$\ddot{\boldsymbol{\xi}}_i^d = \boldsymbol{r}_i\left(\boldsymbol{\xi}_i, \dot{\boldsymbol{\xi}}_i, \boldsymbol{\xi}_i^\star, \dot{\boldsymbol{\xi}}_i^\star, \ddot{\boldsymbol{\xi}}_i^\star\right). \tag{3}$$

For task motion control, the local controller can take the form of a proportional-integral-derivative controller with a feed-forward term

$$\ddot{\boldsymbol{\xi}}_i^d = \ddot{\boldsymbol{\xi}}_i^\star + K_p\boldsymbol{e} + K_d\dot{\boldsymbol{e}} + K_i\int \boldsymbol{e}dt, \tag{4}$$

where $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$ are errors of $\boldsymbol{\xi}_i$ and $\dot{\boldsymbol{\xi}}_i$, respectively; and K_p, K_d and K_i are symmetric, positive definite gain matrices.

For task wrench control, the local controller can take the form of a proportional-integral controller with a feed-forward term

$$\boldsymbol{w}_i^d = \boldsymbol{w}_i^\star + K_{w,p}\boldsymbol{e}_w + K_{w,i}\int \boldsymbol{e}_w dt, \tag{5}$$

where $\boldsymbol{w}_i^\star$ is the desired task wrench, $\boldsymbol{e}_w$ is the error of task wrench, and $K_{w,p}$ and $K_{w,i}$ are symmetric, positive definite gain matrices. The wrench task can be expressed as a motion task using the inverse of the operational space inertia matrix $\Lambda_i(\boldsymbol{q}) = \left[J_i(\boldsymbol{q})M(\boldsymbol{q})^{-1}J_i(\boldsymbol{q})^T\right]^{-1}$ [38,39]

$$\ddot{\boldsymbol{\xi}}_i^d = \Lambda_i(\boldsymbol{q})^{-1}\boldsymbol{w}_i^d \tag{6}$$

which maps the desired task wrench $\boldsymbol{w}_i^d$ to a desired acceleration $\ddot{\boldsymbol{\xi}}_i^d$ at $\mathcal{F}_i$.

- A set of relative importance levels with respect to n_t tasks, including task i, characterized by a priority matrix α_i

$$\alpha_i = diag\left(\alpha_{i1}I_{m_1}, \dots, \alpha_{ij}I_{m_j}, \dots, \alpha_{in_t}I_{m_{n_t}}\right) \tag{7}$$

where α_i is a diagonal matrix, the main diagonal blocks of which are square matrices: $\alpha_{ij}I_{m_j}$. I_{m_j} is the $m_j \times m_j$ identity matrix, and $\alpha_{ij} \in [0,1]$. By convention, the coefficient α_{ij} indicates the priority of task j with respect to task i.
  - $\alpha_{ij} = 0$ corresponds to the case where task j has strict lower priority with respect to task i.
  - $0 < \alpha_{ij} < 1$ corresponds to a soft (non-strict) priority between the two tasks: the greater the value of α_{ij}, the higher the importance level of task j with respect to task i.
  - $\alpha_{ij} = 1$ corresponds to the case where task j has a strict higher priority with respect to task i.

B. Constraint definition

Even though the set of task attributes is specific to each task, the control space to joint space forward mapping is task independent and can be written as

$$\ddot{\boldsymbol{q}} = M(\boldsymbol{q})^{-1}\left(J_c(\boldsymbol{q})^T\boldsymbol{\chi} - \boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}})\right). \tag{8}$$

The equation of motion (8) constitutes an equality constraint, which relates the joint space acceleration to the action variable for a given state $(\boldsymbol{q}, \dot{\boldsymbol{q}})$.

The other constraints considered in this work reflect the physical limitations of the system in terms of:

- actuation capabilities (maximum actuators torques and velocities);
- geometrical limits (joint limits, Cartesian space obstacles);
- contact wrenches (contact existence conditions, bounds on the norms of contact wrenches).

Assuming that approximations such as $\dot{\boldsymbol{q}}_{k+1} = \dot{\boldsymbol{q}}_k + \Delta t \ddot{\boldsymbol{q}}_k$ and $\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \Delta t \dot{\boldsymbol{q}}_k + \frac{\Delta t^2}{2} \ddot{\boldsymbol{q}}_k$ hold for one control period Δt, these specific constraints can generally be expressed as a linear inequality of the form

$$G(\boldsymbol{q}, \dot{\boldsymbol{q}}) \begin{pmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\chi} \end{pmatrix} \leq \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{9}$$

where G and $\boldsymbol{h}$ are the matrix and vector which express these inequality constraints of physical limitations.

IV. Control problem formulation

In this work, multiple tasks with different priority levels subject to equality and inequality constraints have to be handled. This kind of multi-objective control problem can be formulated as a Linear Quadratic Programming problem (LQP). This is the approach adopted here, where all the task objectives and constraints are solved simultaneously in one LQP.

This Section first briefly reviews the LQP control framework in IV-A, then developes a generalized projector in IV-B, which is implemented in a LQP-based control framework in IV-C for handling both strict and non-strict priorities, as well as for priority transitions.

A. Control framework based on Linear Quadratic Programming

When only non-strict task hierarchies are considered, weighting strategies, such as those proposed in [9]–[11,29], can be applied to handle the relative importance levels of multiple elementary tasks. In this case, the control problem can be formulated as a Linear Quadratic Programming (LQP) problem as

$$\underset{\ddot{\boldsymbol{q}}, \boldsymbol{\chi}}{\arg\min} \quad \sum_{i=1}^{n_t} \left\| \boldsymbol{f}_i \left(\ddot{\boldsymbol{q}}, \ddot{\boldsymbol{\xi}}_i^d \right) \right\|_{Q_i}^2 + \left\| \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\chi} \end{bmatrix} \right\|_{Q_r}^2 \tag{10a}$$

$$\text{subject to} \qquad \text{constraints (8), (9)} \tag{10b}$$

where $Q_i = \omega_i I_{m_i}$ is a diagonal weighting matrix to regulate the importance level of task i, $Q_r = \omega_r I_{n+n_a+3n_c}$ is the weighting matrix of the regularization term, ω_i is the weight of each task objective i, and $\boldsymbol{f}_i \left(\ddot{\boldsymbol{q}}, \ddot{\boldsymbol{\xi}}_i^d \right) = J_i(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{J}_i(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \ddot{\boldsymbol{\xi}}_i^d$ is the objective function which measures the error of task i. The task objective functions are minimized to achieve a compromise among all the weighted tasks. The regulation term minimizes the norm of accelerations and action variables. For a redundant robot with many solutions satisfying the same task objective, the regulation term is useful for ensuring the uniqueness of the solution [29]. As this regulation term may increase task error, its weight value ω_r is usually very small compared to task objective weights. In this optimization problem, $\ddot{\boldsymbol{q}}$ is an overabundant variable, which can be eliminated by using the equality constraint defined by (8).

B. Projectors for hierarchical control

The control framework based on weighting strategy (10) can qualitatively regulate the relative importance levels of tasks by weighting task objectives, but it cannot ensure strict priorities among tasks. This control framework is extended in this paper, the goal of which is to handle both strict and non-strict task priorities. To achieve this goal, a generalized projector, which can precisely regulate how much a task is affected by other tasks, is developed. In other words, this generalized projector can be regulated to *completely, partially, or not* project a task into the null-space of other tasks.

The following part of this subsection first looks at several forms of projectors, then the analysis of these projectors leads to the development of the generalized projector.

1) Review of existing projectors for hierarchical control

Strict priorities can be handled by analytical methods using a null-space projector N_j defined as

$$N_j = I - J_j^{\dagger} J_j, \tag{11}$$

where $J_j^{\dagger}$ is the Moore-Penrose pseudo-inverse of the Jacobian J_j[2]. The projection of a task i into the null-space of another task j can ensure that task i is satisfied only in the null-space of task j. Such projection-based approaches can ensure that a lower-priority task is performed without producing any motion for a higher-priority task. To handle priorities between one task i and a set of other tasks with higher priorities, task i is projected into the null-space of an augmented Jacobian J of all the higher priority tasks [40,41]

$$J = \left[J_1^T \dots J_j^T \dots J_{n_t}^T \right]^T \tag{12}$$

where the augmented Jacobian concatenates the Jacobian matrices of all the n_t tasks.

To achieve smooth priority transitions, the null-space projector (11) is replaced by the following matrix in [31,32]

$$N_j^{'}(\alpha_{ij}) = I - \alpha_{ij} J_j^{\dagger} J_j, \tag{13}$$

where a scalar parameter $\alpha_{ij} \in [0, 1]$ is used to regulate the priority between two tasks i and j. This matrix leads to smooth transitions of task priorities through the smooth change of the scalar parameter α_{ij}:

- when $\alpha_{ij} = 1$, $J_i N_j^{'} = J_i N_j$, task i is completely projected into the null-space of task j;
- when $0 < \alpha_{ij} < 1$, task i is partially projected into the null-space of task j;
- when $\alpha_{ij} = 0$, $J_i N_j = J_i$, task i is not at all projected into the null-space of task j.

This method can handle priority transitions between only two levels of tasks, and it can hardly be extended to the case of simultaneous transitions among multiple levels of task priorities.

[2]The dependence to $\boldsymbol{q}$ is omitted for clarity reasons.

Another projection matrix $N^{''}$ is proposed in [35] for continuous null-space projections

$$N^{''} = I - VAV^T \tag{14}$$

with $V \in \mathbb{R}^{n\times n}$ the right singular vectors of J and $A \in \mathbb{R}^{n\times n}$ a diagonal activation matrix. The j-th diagonal element of A, a_{jj}, refers to the j-th column vector in V:

- when $a_{jj} = 1$, the j-th direction in V is activated in $N^{''}$;
- when $0 < a_{jj} < 1$, the j-th direction in V is partially deactivated in $N^{''}$;
- when $a_{jj} = 0$, the j-th direction in V is deactivated in $N^{''}$.

As mentioned in [35], for any one-dimensional task j ($J_j \in \mathbb{R}^{1\times n}$), the matrix (14) becomes

$$N_j^{''} = I - a_{j,j}\frac{J_j^T}{\|J_j\|}\frac{J_j}{\|J_j\|}, \tag{15}$$

which can be applied to achieve smooth activation or deactivation of task j direction in the projection matrix by the smooth variation of the scalar a_{jj}. When extended to tasks of m directions ($J \in \mathbb{R}^{m\times n}$), this method allows us to apply the same transition to all the m directions of J, but its application for achieving the separate regulation of each task direction is not easy. This is because generally, each activator a_{jj} is directly referred to the j-th direction in the right singular vectors V of J, but not directly referred to a specific direction in J.

2) Generalized projector

In order to achieve variations of multiple task priorities simultaneously among an arbitrary number of tasks, and to be able to ensure strict priorities, an approach to the computation of a novel projector is developed here. Similar to the form of the matrix $N^{''}$ in the case of considering a one-dimensional task (15), the form of this novel projector is obtained without the necessity of the computation of pseudo-inverse matrices. Its computation is based on orthonormal basis computation, and it is simple to implement this novel projector. Moreover, the new projector allows us to regulate the activation of each task directions in a more intuitive way, by regulating the priority matrix α that is more closely related to task directions than the activator A in (14).

Consider a hierarchy of n_t tasks, the joint space acceleration $\ddot{\boldsymbol{q}}_i^\star$ for achieving each task i should be modified to account for the hierarchy information contained in α_i. The idea is to achieve this goal by a generalized projector $P_i(\alpha_i) \in \mathbb{R}^{n\times n}$, which projects the joint acceleration ($P_i(\alpha_i)\ddot{\boldsymbol{q}}_i^\star$) to satisfy the desired hierarchy.

In order to compute the generalized projector $P_i(\alpha_i)$, a preliminary processing of the matrices J and α_i is carried out according to the priorities of the n_t tasks with respect to task i. As each row of J is associated to α_{ij}, the rows of J can be sorted in descending order with respect to the values of the diagonal elements in α_i. The resulting matrix J_{s_i} is thus constructed so that tasks which should be the least influenced by task i appear in its first rows, while tasks which can be the most influenced by task i appear in its last rows. The values in α_i are sorted accordingly, leading to α_i^s, the diagonal elements of which are organized in descending order starting from the first row.

Based on J_{s_i}, a projector into the null space of J can be computed. This can be done by first computing a matrix $B_i(J_{s_i}) \in \mathbb{R}^{r\times n}$, where $r = rank(J_{s_i})$ is the rank of J_{s_i}. The rows of $B_i(J_{s_i})$ form an orthonormal basis of the joint space obtained using elementary row transformations on J_{s_i}. Then this projector can be computed as $P_i^{'} = I_n - B_i^T B_i$. When performing task i by using the projected acceleration $P_i^{'}\ddot{\boldsymbol{q}}_i^\star$, the projector $P_i^{'}$ basically cancels any acceleration that impacts all the n_t tasks, including task i itself.

The computation of the projector $P_i^{'}$ can be modified such that tasks having strict priority over task i are perfectly accounted for; tasks over which task i has a strict priority are not considered; and all other tasks with soft priorities are accounted for, according to the value of their respective priority parameters in α_i. Inspired by how the matrix $N^{''}$ is computed in (15), the generalized projector is given by

$$P_i(\alpha_i) = I_n - B_i(J_{s_i})^T \alpha_{i,r}^s(\alpha_i, \boldsymbol{origin}) B_i(J_{s_i}), \tag{16}$$

where $\alpha_{i,r}^s$ is a diagonal matrix of degree r. The vector $\boldsymbol{origin} \in \mathbb{R}^r$ is a vector of the row indexes of J_{s_i} selected during the construction of the orthonormal basis B_i. Each of these r rows in J_{s_i} is linearly independent to all the previously selected ones. The diagonal elements of $\alpha_{i,r}^s$ are restricted to the r diagonal elements of α_i^s, which correspond to the r rows of J_{s_i}, the row indexes of which belong to $\boldsymbol{origin}$.

Algorithm (1) and (2) summarize the construction of the generalized projector $P_i(\alpha_i)$. As any numerical scheme, tolerances are used for numerical comparison, such as ϵ, which is defined as the smallest value greater than zero in line #11 of Algorithm (2).

Algorithm 1: Generalized projector computation - task i

```
  Data: $\alpha_i$, $J$
  Result: $P_i$
1 begin
2                   $n \longleftarrow GetNbCol(J)$
3               $index \longleftarrow GetRowsIndexDescOrder(\alpha_i)$
4                 $\alpha_i^s \longleftarrow SortRows(\alpha_i, index)$
5                $J_{s_i} \longleftarrow SortRows(J, index)$
6     $B_i, \boldsymbol{origin}, r \longleftarrow GetOrthBasis(J_{s_i})$ ▷Alg. (2)
7               $\alpha_{i,r}^s \longleftarrow GetSubDiagMatrix(\alpha_i^s, \boldsymbol{origin})$
8                  $P_i \longleftarrow I_n - B_i^T \alpha_{i,r}^s B_i$
9     return $P_i$
```

Note that by varying the value of each α_{ij} in α_i, one can regulate the priority of each task j in the n_t tasks with respect to task i separately.

3) Task insertion and deletion

There is a particular case induced by the proposed formulation and corresponding to the influence of task i on itself. Even though not intuitive, this self-influence has to be interpreted in terms of task existence, modulated by α_{ii}. If $\alpha_{ii} = 1$ then task i is projected into its own null-space, *i.e.* it is basically canceled out. Decreasing α_{ii} continuously to 0 is a simple and elegant way to introduce the task in the set of tasks. Conversely, increasing α_{ii} continuously from 0 to 1 provides

Algorithm 2: Orthonormal basis computation - $GetOrthBasis(A)$

```
   Data: $A$, $\epsilon$
   Result: $B$, $\boldsymbol{origin}$, $r$
1  begin
2      $n \longleftarrow GetNbCol(A)$
3      $m \longleftarrow GetNbRow(A)$
4      $i \longleftarrow 0$
5      for $k \leftarrow 0$ to $m-1$ do
6          if $i \geq n$ then
7              break
8          $B[i,:] \longleftarrow A[k,:]$
9          for $j \leftarrow 0$ to $i-1$ do
10             $B[i,:] \longleftarrow B[i,:] - \left(B[i,:]B[j,:]^T\right) B[j,:]$
11         if $norm(B[i,:]) > \epsilon$ then
12             $B[i,:] \longleftarrow B[i,:]/\,norm(B[i,:])$
13             $\boldsymbol{origin}[i] \longleftarrow k$
14             $i \longleftarrow i+1$
15 $r \longleftarrow i$
16 return $B$, $\boldsymbol{origin}$, $r$
```

with a proper task deletion procedure. When being added or suppressed, the influence of task i with respect to other tasks also has to be defined but here again this can be done in a continuous manner.

C. Generalized hierarchical control framework

The control problem that solves one task i, while taking into account the constraints as well as the influence of a set of other tasks over it, can be written as follows

$$\underset{\ddot{\boldsymbol{q}},\boldsymbol{\chi}}{\arg\min} \quad \left\|\boldsymbol{f}_i\left(\ddot{\boldsymbol{q}}, \ddot{\boldsymbol{\xi}}_i^d\right)\right\|^2 + \left\|\begin{bmatrix}\ddot{\boldsymbol{q}}\\ \boldsymbol{\chi}\end{bmatrix}\right\|^2_{Q_r} \tag{17a}$$

subject to

$$J_c(\boldsymbol{q})^T\boldsymbol{\chi} = M(\boldsymbol{q})P_i(\alpha_i)\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{17b}$$

$$G(\boldsymbol{q},\dot{\boldsymbol{q}})\begin{pmatrix}P_i(\alpha_i)\ddot{\boldsymbol{q}}\\ \boldsymbol{\chi}\end{pmatrix} \leq \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{17c}$$

where the generalized projector defined by (16) is applied in the constraints to handle task priorities. Here, the task objective weighting matrix Q_i is omitted, as it is set to the identity matrix; and the matrix $Q_r = \omega_r I_{n+n_a+3n_c}$ is set to a diagonal matrix with the weight value ω_r being very small compared to 1.

Now consider the control problem for solving n_t tasks. A joint acceleration variable $\ddot{\boldsymbol{q}}'_i$ is associated to each task i, such that the overall joint space acceleration accounting for the sets of relative importance parameters $(\alpha_1,\ldots,\alpha_{n_t})$ is given by

$$\ddot{\boldsymbol{q}} = \sum_{i=1}^{n_t} P_i(\alpha_i)\ddot{\boldsymbol{q}}'_i. \tag{18}$$

The GHC framework solves the LQP problem formulated as

$$\underset{\ddot{\boldsymbol{q}}',\boldsymbol{\chi}}{\arg\min} \quad \sum_{i=1}^{n_t}\left\|\boldsymbol{f}_i\left(\ddot{\boldsymbol{q}}'_i, \ddot{\boldsymbol{\xi}}_i^d\right)\right\|^2 + \left\|\begin{bmatrix}\ddot{\boldsymbol{q}}'\\ \boldsymbol{\chi}\end{bmatrix}\right\|^2_{Q_r} \tag{19a}$$

subject to

$$J_c(\boldsymbol{q})^T\boldsymbol{\chi} = M(\boldsymbol{q})P\ddot{\boldsymbol{q}}' + \boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{19b}$$

$$G(\boldsymbol{q},\dot{\boldsymbol{q}})\begin{pmatrix}P\ddot{\boldsymbol{q}}'\\ \boldsymbol{\chi}\end{pmatrix} \leq \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}), \tag{19c}$$

with $\ddot{\boldsymbol{q}}' = \begin{bmatrix}\ddot{\boldsymbol{q}}'_1\\ \vdots\\ \ddot{\boldsymbol{q}}'_{n_t}\end{bmatrix}$ and $P = [P_1(\alpha_1)\ldots P_{n_t}(\alpha_{n_t})]$..

This optimization problem minimizes the objective function of each task as well as the magnitude of the control input, subject to a set of linear constraints. By solving this optimization problem, the solution of joint accelerations and the action variable $\boldsymbol{\chi}$ can be obtained. The solution of joint torques is extracted from the optimal value of $\boldsymbol{\chi}$. The overall joint space acceleration $\ddot{\boldsymbol{q}}$ is optimized to achieve all the tasks according to the set of their relative importance $(\alpha_1,\ldots,\alpha_{n_t})$. Especially, it is proved in Appendix A that this GHC framework can handle strict task hierarchies represented by standard lexicographic orders.

This control approach is robust to both kinematic and algorithmic singularities. In the GHC framework based on LQP formulation, tasks are expressed in a forward way and most LQP solvers do not require the explicite inversion of task Jacobian matrices. Therefore, the GHC framework does not have problems of numerical singularities due to kinematic singularities. Moreover, unlike approaches using the pseudo-inverse of projected Jacobians (J_iN_j), which requires special treatment for handling algorithmic singularities when the projected Jacobian drops rank [42], the GHC framework does not necessite the inversion of projected Jacobians. Therefore, the framework does not have to handle such kind of algorithmic singularities.

V. RESULTS

The proposed GHC framework (19) is applied to the control of a 7-DoF Kuka LWR robot. The experiments are conducted in the Arboris-Python simulator [43], which is a rigid multi-body dynamics and contacts simulator written in Python. The LQP problem is solved by a QP solver included in CasADi-Python [44], which is a symbolic framework for dynamic optimization.

In the experiments, three tasks are defined: task 1 for the control of the three dimensional position of the end-effector, task 2 for the three dimensional position of the elbow, and task 3 for the posture. Any wrench task is transformed into a motion task by applying (6). Targets of the three tasks are not compatible with one another. The elbow task target is a fixed target position and the posture task target is a fixed posture during all the experiments.

The GHC framework (19) is applied, with $n_t = 3$. For each task i, an optimization variable $\ddot{\boldsymbol{q}}'_i \in \mathbb{R}^7$ is defined. A local controller (4) is used to ensure the convergence of each task variable towards its target. More precisely, a proportional-derivative controller $\ddot{\boldsymbol{\xi}}_i^d$ is applied for each task. When a task target is static, $\ddot{\boldsymbol{\xi}}_i^d = K_p e_i + K_d \dot{e}_i$ with $K_p = 30s^{-2}$ and $K_d = 20s^{-1}$. When tracking a desired trajectory $\ddot{\boldsymbol{\xi}}_i^*$, $\ddot{\boldsymbol{\xi}}_i^d = \ddot{\boldsymbol{\xi}}_i^* + K_p e_i + K_d \dot{e}_i$ with $K_p = 100s^{-2}$ and $K_d = 20s^{-1}$. The priority parameter matrices associated with the three tasks are: $\alpha_1 = diag\,(\alpha_{11} I_3, \alpha_{12} I_3, \alpha_{13} I_7)$, $\alpha_2 = diag\,(\alpha_{21} I_3, \alpha_{22} I_3, \alpha_{23} I_7)$, $\alpha_3 = diag\,(\alpha_{31} I_3, \alpha_{32} I_3, \alpha_{33} I_7)$. The regularization weight Q_r is chosen as 0.01.

In the rest of the paper, the notation $i \rhd j$ indicates that task i has a strict higher priority over task j, and the notation $\Rightarrow$ stands for a transition of hierarchy setting. The following function is used for the smooth variation of an α_{ij} (conversely α_{ji}) from 0 to 1 during the transition time period ($[t_1, t_2]$)

$$\begin{aligned} &\alpha_{ij}(t) = 0.5 - 0.5\cos\left(\frac{t - t_1}{t_2 - t_1}\pi\right), \text{ with } t \in [t_1, t_2], \\ &\alpha_{ji}(t) = 1 - \alpha_{ij}(t). \end{aligned} \tag{20}$$

A. Priority switching subject to constraints

This experiment is carried out to demonstrate that GHC allows handling task priorities subject to a variety of constraints. All the task targets are static. An obstacle plane is inserted between the initial position of the end-effector and its target position (see Fig. 1). The robot should avoid penetration into the obstacle while performing tasks. A threshold value of $0.02m$ is chosen as the minimum authorized distance between the end-effector and the obstacle plane.

The optimization variables are $\ddot{\boldsymbol{q}}_1'$, $\ddot{\boldsymbol{q}}_2'$, $\ddot{\boldsymbol{q}}_3'$, and τ. The inequality constraints (19c) are:

- bounds on joint velocities (bjv), with $G_{bjv} = \begin{bmatrix} I_7 & 0_7 \\ -I_7 & 0_7 \end{bmatrix}$ and $h_{bjv} = \begin{bmatrix} (\overline{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}})/\Delta t \\ (-\underline{\dot{\boldsymbol{q}}} + \dot{\boldsymbol{q}})/\Delta t \end{bmatrix}$, where $\overline{(\cdot)}$ and $\underline{(\cdot)}$ denote the upper and lower bounds, respectively;
- bounds on joint torques (bjt), with $G_{bjt} = \begin{bmatrix} 0_7 & I_7 \\ 0_7 & -I_7 \end{bmatrix}$ and $h_{bjt} = \begin{bmatrix} \overline{\tau} \\ -\underline{\tau} \end{bmatrix}$;
- and obstacle avoidance (obs), with $G_{obs} = [\boldsymbol{n}_{obs}^T J_1 \;\, 0_7]$ and $h_{obs} = (\overline{v_1}(d_{obs}) - n_{obs}^T J_1 \dot{\boldsymbol{q}})/\Delta t$, where $\boldsymbol{n}_{obs}$ is the unit normal vector pointing from the end-effector to the obstacle plane, J_1 is the Jacobian of the end-effector task, and $\overline{v_1}$ is the bound on the end-effector velocity towards the plane, which depends on the distance (d_{obs}).

For the sake of clarity, joint limits and obstacle avoidance between the bodies of the robot (other than the end-effector) and the environment are not considered in this example.

At the beginning, the tasks, in a priority level decreasing order, are the elbow task, the end-effector task, and the posture task. Then the end-effector task priority increases and becomes the task with the highest priority. Afterwards, the priorities of the posture task and the elbow task are switched. Then the priorities of the posture task and the end-effector task are switched. At the end, the posture task becomes the task with the highest priority. The evolution of the task hierarchy is defined as: $2 \rhd 1 \rhd 3 \Rightarrow 1 \rhd 2 \rhd 3 \Rightarrow 1 \rhd 3 \rhd 2 \Rightarrow 3 \rhd 1 \rhd 2$.

The task errors are presented in Fig. 2. The desired priority switches are successfully performed and desired task priorities are well maintained. The resulting end-effector trajectory and the measured distance between the end-effector and the obstacle are presented in Fig. 3. The resulting joint velocities and joint torques are shown in Figures 4 and 5, respectively. It can be seen from these figures that joint velocity bounds and joint torques bounds are respected. Also the end-effector does not penetrate into the obstacle while trying to move towards its target position. The results of this experiment illustrate the fact that GHC can maintain desired task priorities while satisfying all these constraints. A video of this experiment is attached to this paper.

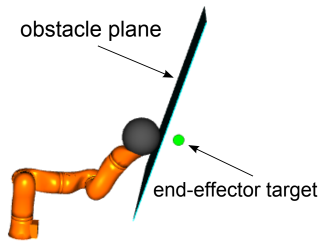


Fig. 1: The end-effector moves towards its target position while avoiding penetration into the obstacle plane.

B. Contact force control

In this experiment, the end-effector is expected to move towards a plane, and then to apply a desired contact force against the plane in the vertical direction (see Fig. 6). Before the establishment of the contact with the plane, the end-effector task is a motion task with its task target located on the surface of the plane. Once the end-effector contacts the plane, the end-effector task is a composition of a position task in the horizontal plane and a force task in the vertical direction. The end-effector starts from an initial position, which is above its target position and pointing upwards, then it moves towards the target and starts to apply a contact force to the plane.

The evolution of task hierarchy is $2 \rhd 1 \rhd 3 \Rightarrow 1 \rhd 2 \rhd 3$. At the beginning of this experiment, the elbow task has the highest priority, then the priorities between the elbow task and the end-effector task switches. The change of α, the positions errors, and the actual and desired contact forces are shown in Fig. 7.

When the end-effector task becomes the task with the highest priority, the end-effector position error is small and the generated contact force follows the references of the contact force, except for when the contact is established between the two rigid bodies. This result illustrates the fact that the highest priority task of the end-effector is maintained after the application of the contact force.

C. Empirical comparison with other approaches

In this Section, the GHC approach is compared with other approaches dedicated to hierarchical control subject to inequality constraints.

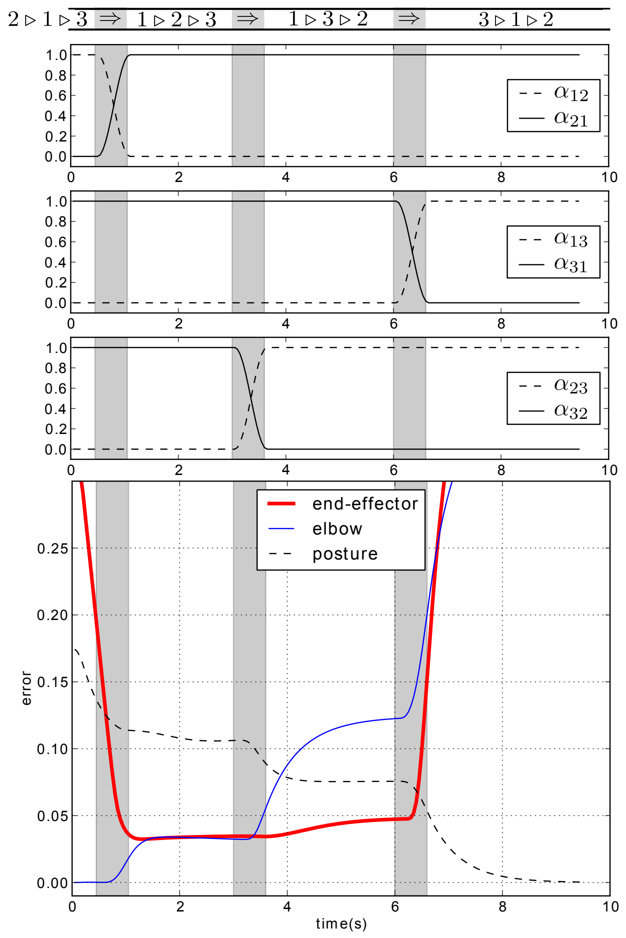


Fig. 2: Evoution of αs (top) and task errors (bottom) during priority switching subject to constraints. Task priorities are switched continuously with the continuous change of αs. The end-effector task error is not decreased to 0 when its task priority is the highest: the obstacle avoidance constraint is respected, and the end-effector cannot arrive at its target position behind the obstacle plane.

1) Comparison with HQP

In this experiment, GHC is compared with the HQP approach [45]. Task hierarchy is changed four times (see Fig. 8) and joint velocity and joint torques bounds are imposed. The evolution of the task hierarchy is 3⊳2⊳1 ⇒ 1⊳2⊳3 ⇒ 2⊳1⊳3 ⇒ 1⊳3 ⇒ 1⊳2⊳3. In the beginning, the tasks, in the priority level decreasing order, are the posture task, the elbow task, and the end-effector task. Then the end-effector task priority increases and the posture task priority decreases simultaneously. During the second priority switching, the priorities of the end-effector task and the elbow task are switched. Then the elbow task is removed. Finally, the elbow task is inserted between the end-effector task and the posture task.

The experiment is carried out first using fixed task targets, then using a desired end-effector trajectory with a lemniscate shape.

The results corresponding to the use of fixed task targets are presented in Fig. 9 to 11. Task errors by using GHC and HQP are shown in Figures 9 and 10, respectively. Fig. 11 shows the integration of the absolute value of each resulting joint jerk $\left(\int_0^t \frac{|\mathrm{d}^3 q|}{\mathrm{d}t^3} dt\right)$ using these two approaches. Steady state task errors for each task hierarchy configuration are shown in Table I, where the results using GHC and HQP are included.

When a lemniscate-shaped end-effector trajectory is used, the end-effector task is to move along this lemniscate orbit periodically, with an orbital period of $2\pi s$. The desired and the resulting end-effector trajectory is shown in Fig. 12. In this case, the results of task errors and the integration of the absolute values of joint jerks are presented in Fig. 13 and 14, respectively. A video of this experiment that presents the main

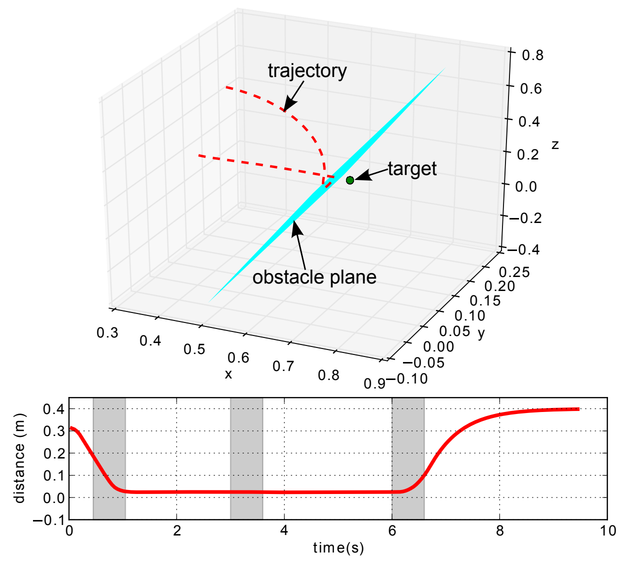


Fig. 3: The resulting end-effector trajectory (top) and the distance between the end-effector and the obstacle (bottom). The end-effector stops moving toward the obstacle plane when its distance to the obstacle decreases to the threshold value of $0.02m$.

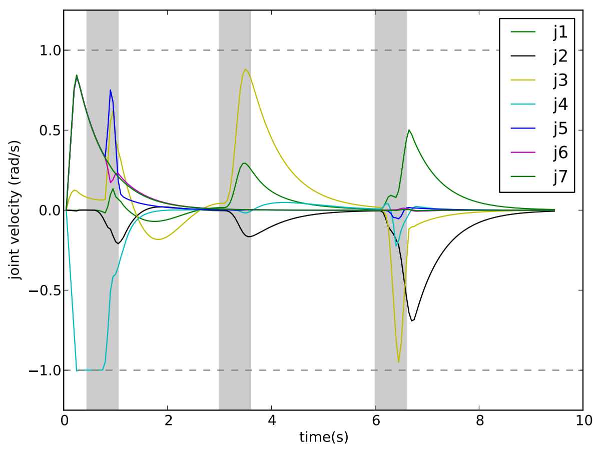


Fig. 4: Evolution of the joint velocities. The upper and lower bounds of $\dot{\boldsymbol{q}}$ are $1\ rad/s$ and $-1\ rad/s$, respectively. These bounds are voluntarily set low in order to easily illustrate the fact that they are respected.

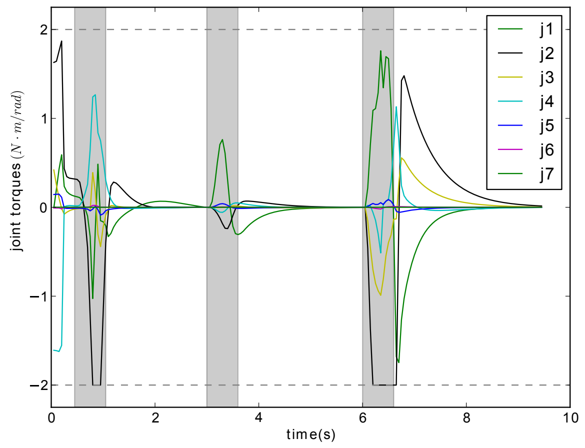


Fig. 5: The resulting joint torques. The upper and lower bounds of τ are $2\ N \cdot m/rad$ and $-2\ N \cdot m/rad$, respectively. These bounds are voluntarily set low in order to easily illustrate the fact that they are respected.

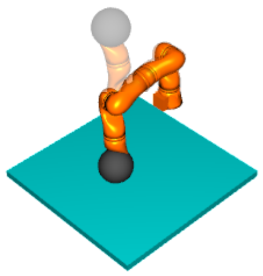


Fig. 6: The target position for the end-effector is on the plane. The end-effector starts from an initial position, which is above the target position and pointing upwards, then it moves towards the target and starts to apply a contact force to the plane.

TABLE I: Steady state task errors for each task hierarchy configuration

priority	$3 \rhd 2 \rhd 1$		
task	1	2	3
GHC	0.46	0.40	2.2e-30
HQP	0.46	0.40	2.8e-10
priority	$1 \rhd 2 \rhd 3$		
task	1	2	3
GHC	1.0e-6	0.46	1.8
HQP	4.5e-7	0.46	1.8
priority	$2 \rhd 1 \rhd 3$		
task	1	2	3
GHC	0.42	2.6e-6	3.0
HQP	0.42	2.7e-6	3.1
priority	$1 \rhd 3$		
task	1	2	3
GHC	3.9e-6	0.55	0.79
HQP	4.5e-6	0.55	0.79

features of GHC (priority transitions, the insertion and deletion of tasks) is attached to this paper.

GHC provides similar results in terms of task errors compared with HQP, as can be observed in Fig. 9, 10, and 13. The

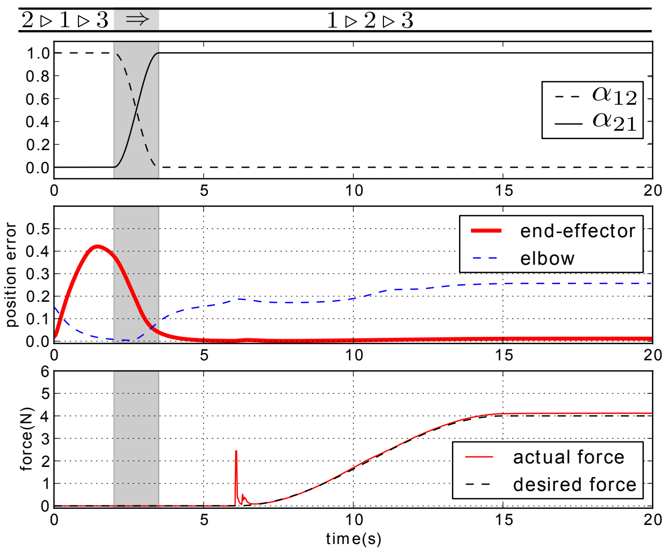


Fig. 7: Results of contact force control. The top figure shows the change of αs. Task 1 is the end-effector task and task 2 is the elbow task. The figure in the middle shows the end-effector position error in the horizontal plane as well as the elbow position error in 3-d. The end-effector starts from an initial position above the target position, then it moves towards the target and starts to apply a contact force to the plane. The bottom figure represents the actual and desired contact forces between the end-effector and the plane.

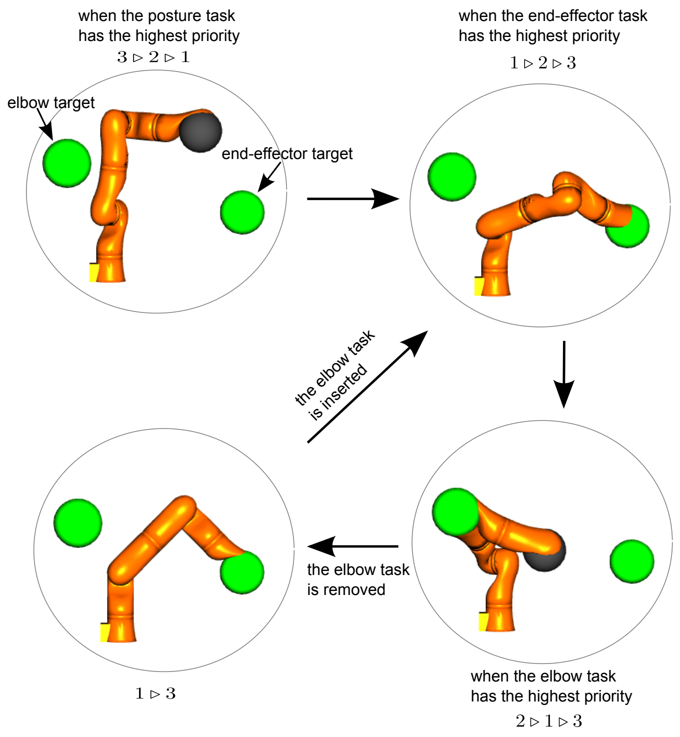


Fig. 8: Experiment of priority switching for the comparison of the HQP and GHC approaches.

results of task errors in Table I show that both GHC and HQP can ensure strict priority. When controlled by GHC and HQP, errors of the task with the highest priority are very small.

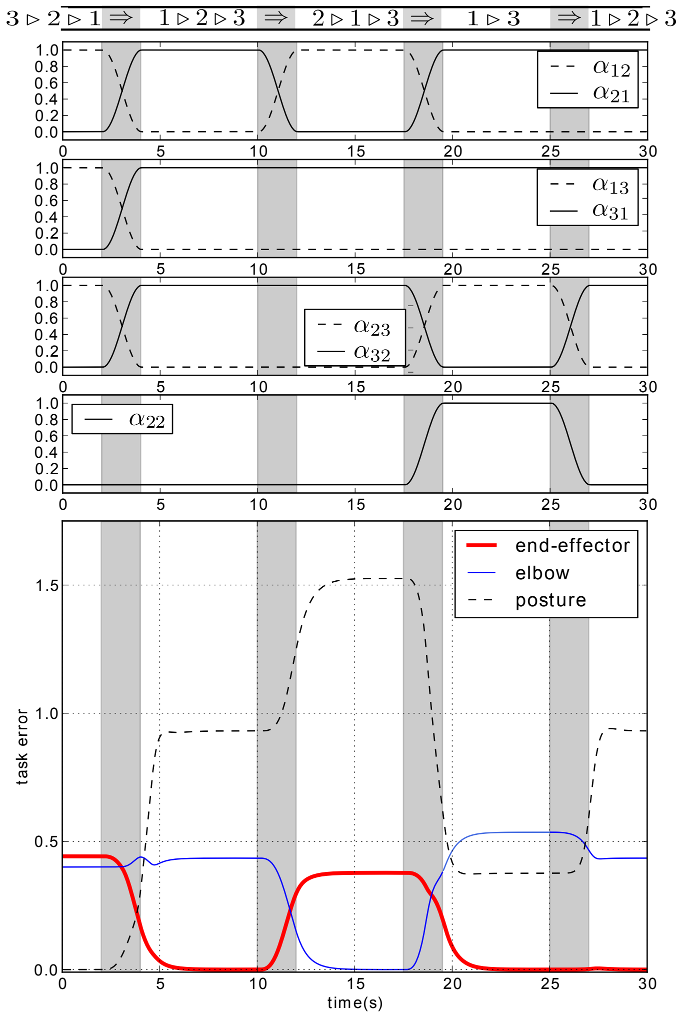


Fig. 9: Evolution of αs (top) and task errors (bottom) using GHC, with fixed task targets. Priority transitions as well as the insertion and deletion of the elbow task are performed. Strict priorities are well respected and the error of the highest priority task is maintained at 0 during steady states.

2) *Comparison with a non-strict hierarchy strategy*

The evolution of the task hierarchy in this experiment is $1 \rhd 2 \rhd 3 \Rightarrow 2 \rhd 1 \rhd 3$. The priorities of the end-effector task and the elbow task are switched once. During priority switching, the task objective weights of the end-effector task (w_1) and the elbow task (w_2) are changed smoothly when the weighting strategy is applied, and the priority parameters α_{12} and α_{21} are changed smoothly when GHC is applied. When the weighting strategy is used, two pairs of the weights ($w_1 = 1$, $w_2 = 0.1$) and ($w_1 = 1$, $w_2 = 0.001$) are applied.

Fig. 15 presents how task errors change with the priority parameters using GHC, as well as how they change with different task objective weights using the weighting strategy described by (10). It can be seen in this figure that a continuous change of corresponding values of αs can generate similar variations of task errors as a continuous change of task weights does. The priority of the end-effector task decreases gradually with respect to the elbow task, either by the continuous decrease of the weight of the end-effector (w_1) and the increase of the weight of the elbow task (w_2), or by the continuous increase of α_{21} and decrease of α_{12}. Moreover, the larger the difference between the maximum and the minimum values of task weights are, the closer the task performances are to those generated by GHC. This is because an increase of the difference between task weights makes non-strict task hierarchies evolve towards the extreme case of strict task

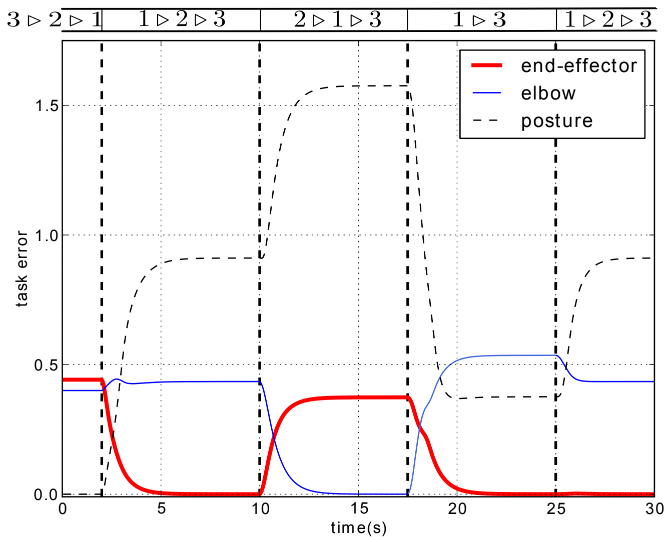


Fig. 10: Task errors using HQP, with fixed task targets. HQP provides similar results in terms of task errors as GHC (Fig. 9).

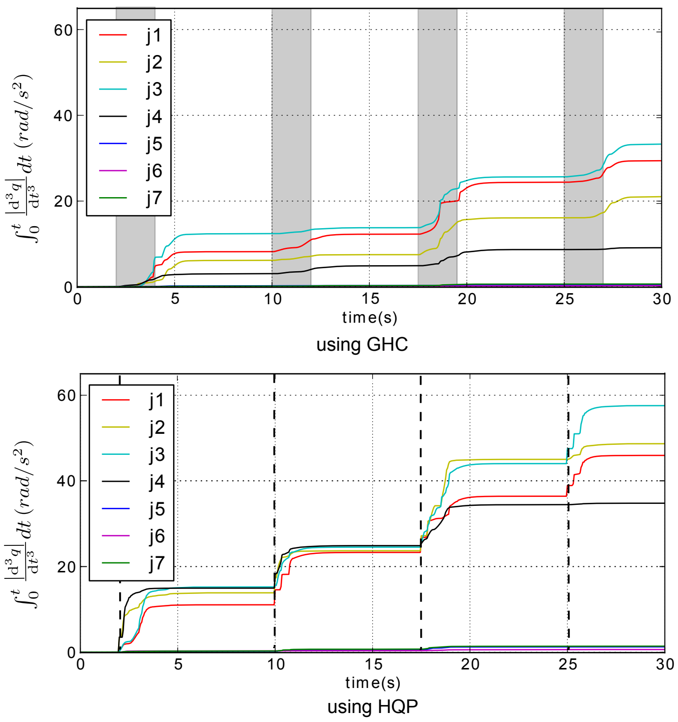


Fig. 11: Integration of the absolute values of joint jerks using GHC (top) and HQP (bottom), with fixed task targets. GHC generates smaller joint jerks than HQP does, while the latter one provides larger jerks each time task hierarchy is changed.

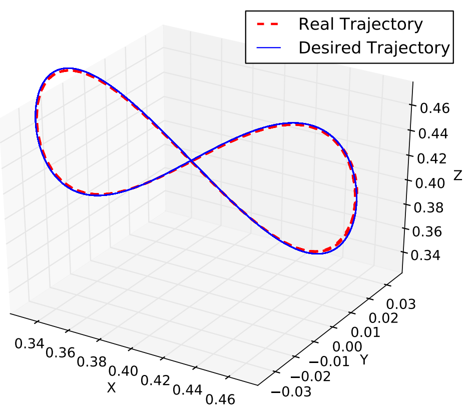


Fig. 12: The desired and the resulting end-effector trajectory provided by GHC, when the end-effector task has the highest priority. The end-effector moves along the lemniscate-shaped trajectory with an orbital period of $2\pi s$.

hierarchies. However, if a large number of importance levels has to be handled, then a huge difference between the weight of the highest priority task and the one of the lowest priority task has to be used. On the contrary, when GHC is applied, strict priorities can be easily achieved by setting the relevant α_{ij} to its limit values 0 or 1.

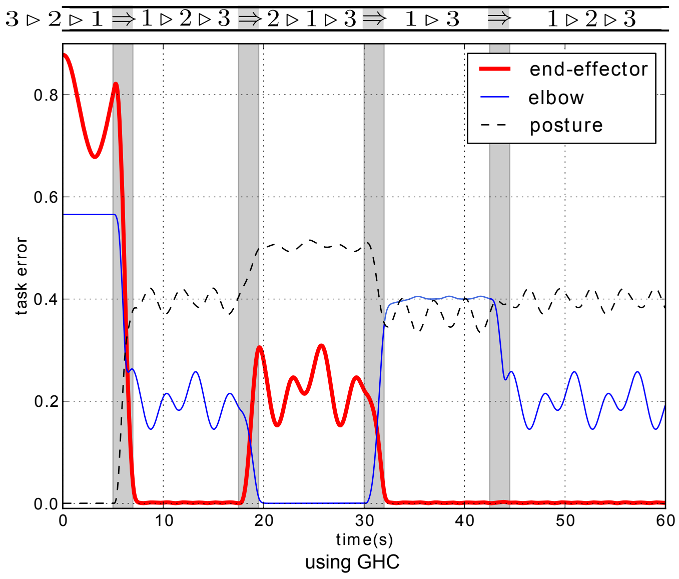
VI. Discussion

In this section, the computation cost and the continuity aspects of this approach are discussed.

A. Computation time

For a robot of n DoFs performing k priority levels of tasks with a total dimension of m, the computation cost by using the HQP solver [46] is dominated by the hierarchical complete orthogonal decomposition, whose cost is equivalent to $n^2m + nm^2 + \sum_{i=1}^{k}(m_i - r_i)m_i^2$, with m_i and r_i being respectively the dimension of tasks and the rank of task jacobian in the i-th hierarchy. By using the GHC strategy, the magnitude order of optimization variables is kn, since a joint acceleration variable $\ddot{q}_i' \in \mathbb{R}^n$ is associated to each task i. In this case, one level of QP (19) needs to be solved, so the computation cost is in $O((kn)^2m + knm^2 + (m-r)m^2)$, with r being the rank of the augmented task jacobian.

The computational cost of the current GHC strategy is sensitive to the number of DoFs of the robot and the number of tasks. For a fixed-based KUKA robot with 7 DoFs performing n_1 motion tasks of different priority levels, a set of joint acceleration variables $\ddot{q}' \in \mathbb{R}^{7n_1}$ and the joint torques $\tau \in \mathbb{R}^7$ needs to be solved for. For a fixed-based humanoid robot iCub with 32 DoFs performing n_2 tasks, the number of variables would be $32(n_2 + 1)$. Fig. 16 shows the computation time of using GHC to solve randomly selected hierarchical control problems for the KUKA robot and the iCub robot performing



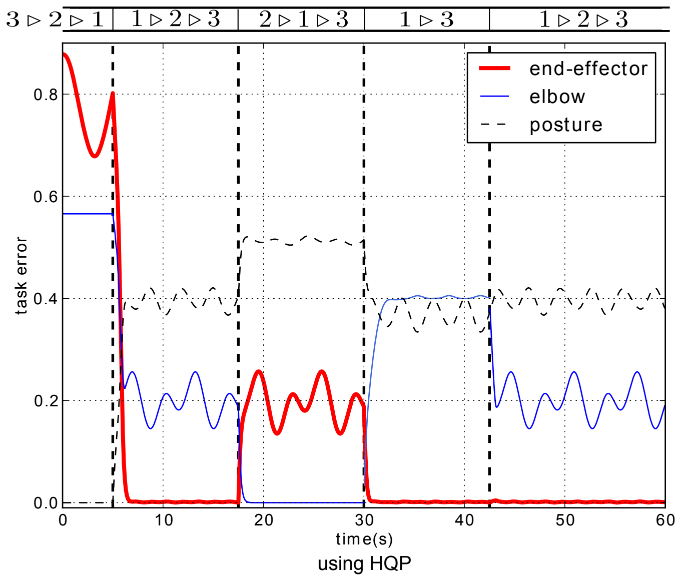


Fig. 13: Task errors using GHC (top) and HQP (bottom), with the end-effector tracking a lemniscate-shaped trajectory. Both approaches achieve desired priority transitions as well as the insertion and deletion of the elbow task, and both of them can maintain strict priorities.

different numbers of tasks. Each control problem consists of the constraint (19b), a posture task with random joint goal positions, and a set of 3-dimensional Cartesian motion tasks with random goal positions. For the KUKA robot performing totally 5 tasks, the mean computation time per iteration is 2.7 ms; for the iCub robot performing the same number of tasks, the mean computation time is 88ms. These results correpond to a C++ implementation of the controller on a standard Linux PC.

B. Continuity

It can be seen in Fig. 11 and 14 that GHC generates smaller joint jerks than HQP does, which implies that GHC provides smoother priority transitions. Basically, the solution

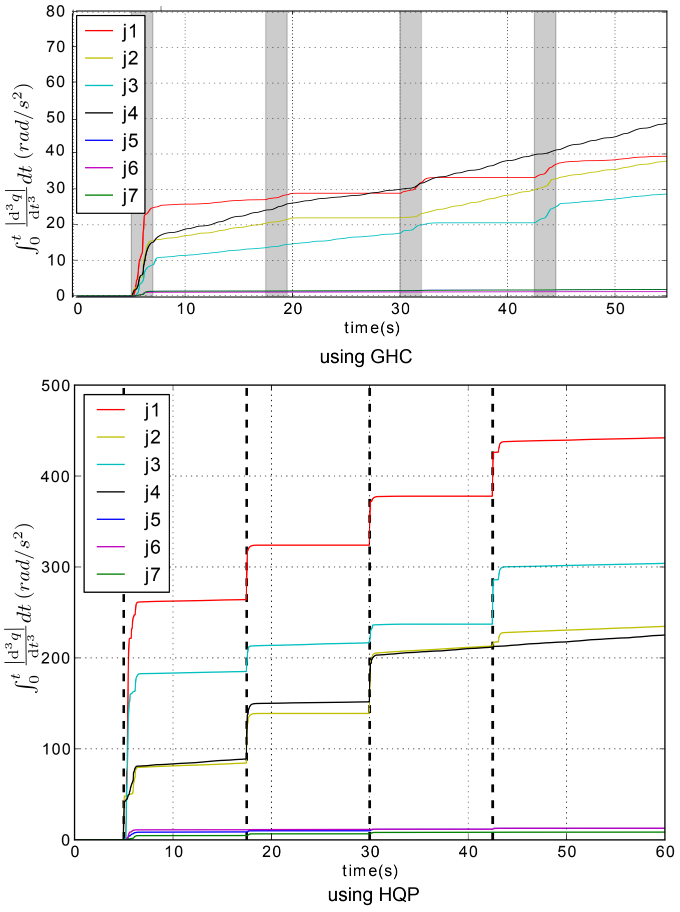


Fig. 14: Integration of the absolute values of joint jerks using GHC (top) and HQP (bottom), with the end-effector tracking a lemniscate-shaped trajectory. GHC generates smaller joint jerks than HQP does, while the latter one provides larger jerks each time task hierarchy is changed.

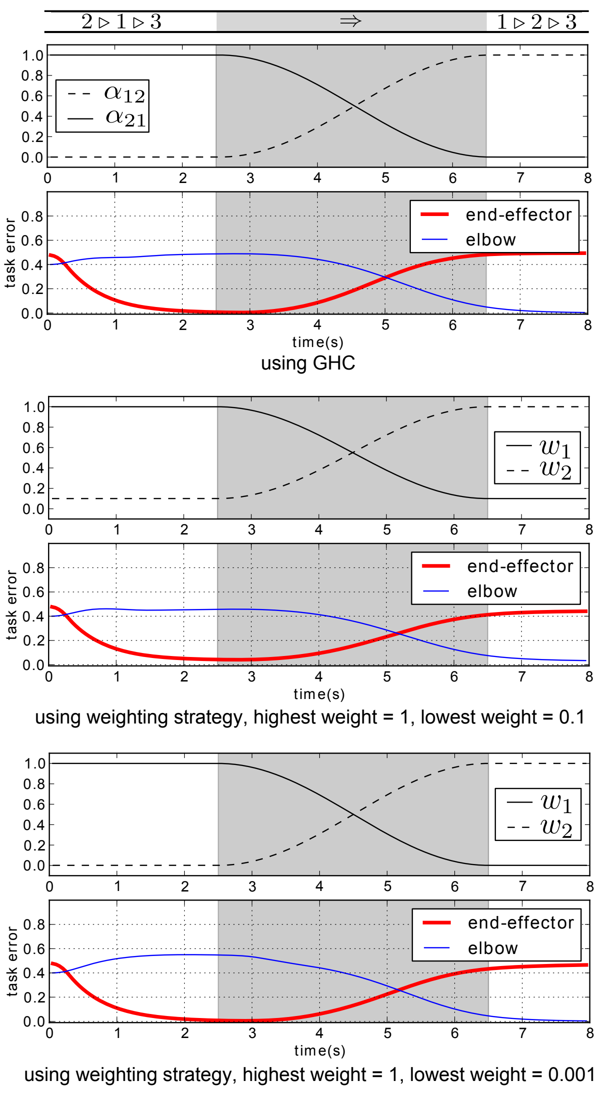


Fig. 15: Evolution of task errors with respect to the evolution of αs using GHC (top) and with respect to different weights by using the weighting strategy (middle and below).

of GHC is continuous, even during hierarchy rearrangements, if the vector ***origin*** in (16) remains the same before and after the rearrangements. Indeed, in this case, the basis B_i used to compute the generalized projector varies continuously with J_i, and the generalized projector varies continuously with B_i and α_i. However, similarly to the HQP algorithm, discontinuity may still occur during the switch of priorities or during the insertion and deletion of tasks. In GHC, such a discontinuity is due to the change of the basis B_i during hierarchy rearrangements.

VII. Conclusions and Future Works

This paper proposes a novel and unifying generalized hierarchical control approach for handling multiple tasks with strict and soft priorities. A generalized projector is developed. It can precisely regulate how much a task can influence or be influenced by other tasks through the modulation of a priority matrix: a task can be completely, partially, or not at all projected into the null-space of other tasks. Multiple simultaneous changes of task priorities can be achieved by using this generalized projector and, using the same mechanism, tasks can be easily inserted or deleted. Moreover, the GHC approach can maintain and switch task priorities while respecting a set of equality and inequality constraints.

Several experiments are conducted to demonstrate that GHC allows task insertion and deletion, as well as the handling of task priorities subject to constraints. Both motion and contact force tasks can be handled by GHC. These experiments emphasize several characteristics of this approach:

1) Priorities among tasks can be maintained by applying the generalized projectors. Through the modulation of the priority matrices $\alpha_1, \ldots, \alpha_{n_t}$ (and consequently of the associated generalized projectors), GHC can behave as a controller that takes into account a strict hierarchy (by setting some $\alpha_{ij} = 0$ or 1) and as a controller that uses a weighting strategy (by setting some $\alpha_{ij} \in]0, 1[$). In other words, the controller can be configured to control simultaneously tasks assigned with strict priorities, as well as tasks with different weights (non-strict priorities).

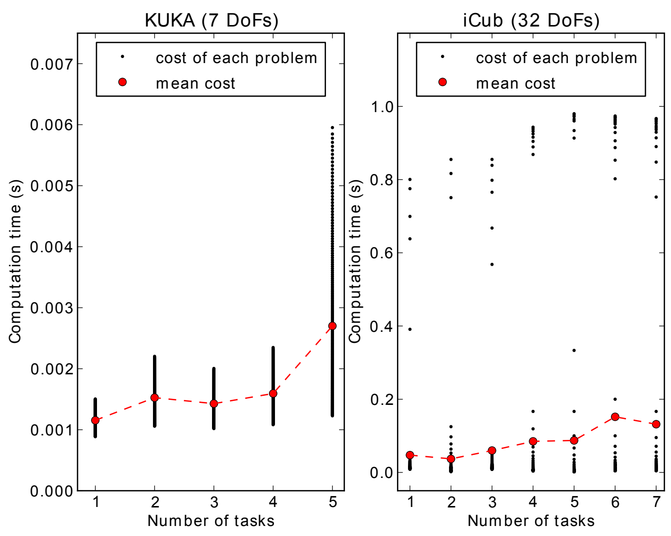


Fig. 16: Computation time per iteration when using GHC to solve randomly selected hierarchical control problems for a fixed-based KUKA robot and a fixed-based iCub robot. Each control problem consists of a posture task and a set of 3D Cartesian motion tasks (0 to 4 motion tasks for KUKA and 0 to 6 motion tasks for iCub), subject to the whole-body equilibrium constraint (19b). The computation time tends to increase with the number of DoFs of the robot and the number of tasks.

2) Simultaneous rearrangements of multiple task priorities can be achieved easily by the variations of relevant entries in the generalized projectors associated to these tasks.

In this work, the GHC approach is illustrated at the dynamic level; however, the generalized projector introduced here is not restricted to this case. In fact, it can also be used in other types of controllers, such as a velocity kinematics controller. The basic idea is to associate each task with a task variable in joint space ($\dot{\boldsymbol{q}}_i'$, $\ddot{\boldsymbol{q}}_i'$, τ_i', etc.), then to apply generalized projectors to these task variables, and finally the global joint space variable is the sum of each projected task variables($P_i(\alpha_i)\dot{\boldsymbol{q}}_i'$, $P_i(\alpha_i)\ddot{\boldsymbol{q}}_i'$, $P_i(\alpha_i)\tau_i'$, etc.).

Immediate future work includes the reduction of the computational cost of GHC to achieve real-time control of complex robots with a high number of DoF. The continuity problem also clearly remains an open problem to tackle in future work. Finally, the use of robot learning techniques to incrementally learn and improve the tuning of the relative influence of each task with respect to others is also of great interest. Finally,

Appendix A
Proof of the maintenance of strict hierarchies represented by standard lexicographic orders subject to constraints

This section proves that the proposed GHC approach (19) can maintain strict task hierarchies represented by standard lexicographic orders while accounting for linear constraints.

Suppose there are n_t tasks that should be organized in a way such that each task i has a strict lower priority than task $i-1$ with i = $2,\dots,n_t$. In this case, the generalized projector P_i of a task i is in fact a null-space projector, which projects a task Jacobian into the null-space of all the previous $i-1$ tasks, and each α_i is an identity matrix. The dependence of P_i to α_i is omitted in this proof for clarity reasons. Let each task objective function be $\boldsymbol{f}_i = J_i\boldsymbol{x}_i' - \boldsymbol{x}_i^d$, with $\boldsymbol{x}_i'$ being a joint space task variable. Moreover, the global variable $\boldsymbol{x} = \sum_i P_i\boldsymbol{x}_i'$ should satisfy linear equality or inequality constraints $A\boldsymbol{x} \leq \boldsymbol{b}$.

At the first stage, the regulation term is neglected, and the optimization problem can be written as follows

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(n_t)}}{\arg\min} \sum_{i=1}^{n_t} \left\| J_i\boldsymbol{x}_i' - \boldsymbol{x}_i^d \right\|^2 \\ &\text{subject to } A\sum_{i=1}^{n_t} P_i\boldsymbol{x}_i' \leq \boldsymbol{b} \end{aligned} \tag{21}$$

where $\boldsymbol{x}'_{(n_t)} = \{\boldsymbol{x}_1', \boldsymbol{x}_2', \dots \boldsymbol{x}_{n_t}'\}$, and the solution to (21) is denoted as $\boldsymbol{x}^*_{(n_t)} = \{\boldsymbol{x}_1^*, \boldsymbol{x}_2^*, \dots \boldsymbol{x}_{n_t}^*\}$.

Consider the case of $n_t = 1$, then the optimization problem can be written as

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(1)}}{\arg\min} \left\| J_1\boldsymbol{x}'_{(1)} - \boldsymbol{x}_1^d \right\|^2 \\ &\text{subject to } A\boldsymbol{x}'_{(1)} \leq \boldsymbol{b}. \end{aligned} \tag{22}$$

The solution to this problem $\boldsymbol{x}^*_{(1)}$ is the same as the one to the problem formulated by HQP.

When $n_t = k$, then the optimization problem is formulated as

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(k)}}{\arg\min} \sum_{i=1}^{k} \left\| J_i\boldsymbol{x}_i' - \boldsymbol{x}_i^d \right\|^2 \\ &\text{subject to } A\sum_{i=1}^{k} P_i\boldsymbol{x}_i' \leq \boldsymbol{b}. \end{aligned} \tag{23}$$

Suppose the solution $\boldsymbol{x}^*_{(k)}$ can maintain the strict task hierarchy: if a task $k+1$ is inserted with lowest priority with respect to the set of k tasks, then the optimization problem with the $k+1$ tasks can be written as

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(k+1)}}{\arg\min} \sum_{i=1}^{k} \left\| J_i\boldsymbol{x}_i' - \boldsymbol{x}_i^d \right\|^2 + \left\| J_{k+1}\boldsymbol{x}_{k+1}' - \boldsymbol{x}_{k+1}^d \right\|^2 \\ &\text{subject to } A\left(\sum_{i=1}^{k} P_i\boldsymbol{x}_i' + P_{k+1}\boldsymbol{x}_{k+1}'\right) \leq \boldsymbol{b}. \end{aligned} \tag{24}$$

As $P_kP_{k+1} = P_{k+1}$, the term $\sum_{i=1}^{k} P_i\boldsymbol{x}_i' + P_{k+1}\boldsymbol{x}_{k+1}'$ in the constraint in (24) is equivalent to $\sum_{i=1}^{k-1} P_i\boldsymbol{x}_i' + P_k\varsigma_k$, with

$$\varsigma_k = \boldsymbol{x}_k' + P_{k+1}\boldsymbol{x}_{k+1}'. \tag{25}$$

Then problem (24) can be written as

$$\begin{aligned} \underset{\boldsymbol{x}'_{(k)},\varsigma_k,\boldsymbol{x}_{k+1}}{\arg\min} & \sum_{i=1}^{k-1} \left\| J_i \boldsymbol{x}'_i - \boldsymbol{x}_i^d \right\|^2 + \left\| J_k \varsigma_k - \boldsymbol{x}_k^d \right\|^2 + \\ & \left\| J_{k+1} \boldsymbol{x}'_{k+1} - \boldsymbol{x}_{k+1}^d \right\|^2 \\ \text{subject to } & A \left(\sum_{i=1}^{k-1} P_i \boldsymbol{x}'_i + P_k \varsigma_k \right) \le \boldsymbol{b} \\ & \varsigma_k = \boldsymbol{x}'_k + P_{k+1} \boldsymbol{x}'_{k+1}. \end{aligned} \tag{26}$$

$\boldsymbol{x}'_k$ in (26) is a free variable, and this problem can be separated into two sub-problems. The first sub-problem is

$$\begin{aligned} \underset{\boldsymbol{x}'_{(k-1)},\varsigma_k}{\arg\min} & \sum_{i=1}^{k-1} \left\| J_i \boldsymbol{x}'_i - \boldsymbol{x}_i^d \right\|^2 + \left\| J_k \varsigma_k - \boldsymbol{x}_k^d \right\|^2 \\ \text{subject to } & A \left(\sum_{i=1}^{k-1} P_i \boldsymbol{x}'_i + P_k \varsigma_k \right) \le \boldsymbol{b}. \end{aligned} \tag{27}$$

The optimal solution $\sum_{i=1}^{k-1} P_i \boldsymbol{x}_i^{*,\prime} + P_k \varsigma_k^*$ to this problem is equivalent to the one of (23). Indeed, these two solutions have the same effect on task k

$$J_k \sum_{i=1}^{k} P_i \boldsymbol{x}_i^{*,\prime} = J_k \left(\sum_{i=1}^{k-1} P_i \boldsymbol{x}_i^{*,\prime} + P_k \varsigma_k^* \right). \tag{28}$$

To prove (28), one needs to notice that $J_i P_j = \mathbf{0}$ with $j \ge i$. The second sub-problem is given by

$$\underset{\boldsymbol{x}_{k+1}}{\arg\min} \left\| J_{k+1} \boldsymbol{x}'_{k+1} - \boldsymbol{x}_{k+1}^d \right\|^2. \tag{29}$$

Therefore, the insertion of a lower priority task $k+1$ does not change the optima of the k previous task objectives. In other words, the strict task hierarchy of an arbitrary number of tasks subject to linear constraints can be maintained.

We have proved that each lower priority task will not increase the obtained optima of all the previous tasks. The rest of this proof explains the roles of the regulation term. As mentioned in Section IV-A, the use of a regulation term, which minimizes the norm of each task variable, helps to ensure the uniqueness of the solution. As each task objective i is assigned with the weight $\omega_i = 1$, which is much greater than the weight of the regulation term ($\omega_r << 1$), the task variables are optimized to mainly satisfy task objectives. Moreover, in GHC, this regulation term also helps to improve the performance of lower priority tasks. Consider $k+1$ levels of tasks to handle, as $J_i P_j = \mathbf{0}$ with $j \ge i$, the final solution is $\sum_{i=1}^{k} P_i \boldsymbol{x}_i^* + P_{k+1} \boldsymbol{x}_{k+1}^*$. Denoting the elements required by task i as $\boldsymbol{x}_i^{i,*}$ and the rest elements that are are not effectively handled by task objective i as $\boldsymbol{x}_i^{f,*}$, the final solution can be rewritten as $S = \sum_{i=1}^{k} P_i^i \boldsymbol{x}_i^{i,*} + \sum_{i=1}^{k} P_i^f \boldsymbol{x}_i^{f,*} + P_{k+1} \boldsymbol{x}_{k+1}^*$, with P_i^i and P_i^f the columns in P_i that correspond to $\boldsymbol{x}_i^{i,*}$ and $\boldsymbol{x}_i^{f,*}$ respectively. The term $\sum_{i=1}^{k} P_i^f \boldsymbol{x}_i^{f,*}$ that is not required by the k previous tasks may contribute to task $k+1$ and affect its task performance. The minimization of the norm of $\boldsymbol{x}_i^f$ in the regulation term improves the performance of task $k+1$ by making S closer to $\sum_{i=1}^{k} P_i^i \boldsymbol{x}_i^{i,*} + P_{k+1} \boldsymbol{x}_{k+1}^*$, where $P_i^i \boldsymbol{x}_i^{i,*}$ are used to perform the k previous tasks and $P_{k+1} \boldsymbol{x}_{k+1}^*$ is used to perform the $(k+1)$-th task in the null-space of all the higher priority tasks.

Acknowledgment

This work was partially supported by the European Commission, within the CoDyCo project (FP7-ICT-2011-9, No. 600716) and by the RTE company through the RTE/UPMC chair Robotics Systems for field intervention in constrained environments held by Vincent Padois.

References

[1] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 1, pp. 43–53, 1987.

[2] M. Mistry, J. Nakanishi, and S. Schaal, "Task space control with prioritization for balance and locomotion," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 2007, pp. 331–338.

[3] P. Hsu, J. Mauser, and S. Sastry, "Dynamic control of redundant manipulators," *Journal of Robotic Systems*, vol. 6, no. 2, pp. 133–148, 1989. [Online]. Available: http://dx.doi.org/10.1002/rob.4620060203

[4] F. Flacco, A. De Luca, and O. Khatib, "Prioritized multi-task motion control of redundant robots under hard joint constraints," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, pp. 3970–3977.

[5] L. Sentis and O. Khatib, "Prioritized multi-objective dynamics and control of robots in human environments," in *4th IEEE/RAS International Conference on Humanoid Robots*, vol. 2, 2004, pp. 764–780 Vol. 2.

[6] O. Kanoun, F. Lamiraux, P.-B. Wieber, F. Kanehiro, E. Yoshida, and J.-P. Laumond, "Prioritizing linear equality and inequality systems: Application to local motion planning for redundant robots," in *IEEE International Conference on Robotics and Automation (2009)*, may 2009, pp. 2939–2944.

[7] L. Saab, N. Mansard, F. Keith, J.-Y. Fourquet, and P. Soueres, "Generation of dynamic motion for anthropomorphic systems under prioritized equality and inequality constraints," in *IEEE International Conference on Robotics and Automation (ICRA)*, may 2011, pp. 1091–1096.

[8] Y. Abe, M. da Silva, and J. Popović, "Multiobjective control with frictional contacts," in *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2007, pp. 249–258.

[9] C. Collette, A. Micaelli, C. Andriot, and P. Lemerle, "Dynamic balance control of humanoids for multiple grasps and non coplanar frictional contacts," in *7th IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 81–88.

[10] M. Liu, A. Micaelli, P. Evrard, A. Escande, and C. Andriot, "Interactive dynamics and balance of a virtual character during manipulation tasks," in *IEEE International Conference on Robotics and Automation (ICRA)*, may 2011, pp. 1676–1682.

[11] K. Bouyarmane and A. Kheddar, "Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configurations," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, 2011, pp. 4414–4419.

[12] M. Liu, A. Micaelli, P. Evrard, A. Escande, and C. Andriot, "Interactive virtual humans: A two-level prioritized control framework with wrench bounds," *IEEE Transactions on Robotics*, vol. 28, no. 6, pp. 1309–1322, 2012.

[13] A. Liégeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7, no. 12, pp. 868–871, dec. 1977.

[14] M. Mistry, J. Nakanishi, G. Cheng, and S. Schaal, "Inverse kinematics with floating base and constraints for full body humanoid robot control," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, 2008, pp. 22–27.

[15] L. Sentis, J. Park, and O. Khatib, "Compliant control of multi-contact and center of mass behaviors in humanoid robots," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 483–501, june 2010.

[16] J. Peters, M. Mistry, F. Udwadia, J. Nakanishi, and S. Schaal, "A unifying framework for robot control with redundant dofs," *Autonomous Robots*, vol. 24, no. 1, pp. 1–12, 2008. [Online]. Available: http://dx.doi.org/10.1007/s10514-007-9051-x

[17] F. Aghili, "A unified approach for inverse and direct dynamics of constrained multibody systems based on linear projection operator: applications to control and simulation," *Robotics, IEEE Transactions on*, vol. 21, no. 5, pp. 834–849, 2005.

[18] O. Khatib, L. Sentis, and J.-H. Park, "A unified framework for whole-body humanoid robot control with multiple constraints and contacts," in *European Robotics Symposium 2008*, ser. Springer Tracts in Advanced Robotics. Springer Berlin / Heidelberg, 2008, vol. 44, pp. 303–312.

[19] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[20] L. Sentis and O. Khatib, "Task-oriented control of humanoid robots through prioritization," in *IEEE RAS/RSJ International Conference on Humanoid Robots*, 2004.

[21] ——, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 02, no. 04, pp. 505–518, 2005.

[22] O. Stasse, A. Escande, N. Mansard, S. Miossec, P. Evrard, and A. Kheddar, "Real-time (self)-collision avoidance task on a hrp-2 humanoid robot," in *IEEE International Conference on Robotics and Automation (ICRA)*, may 2008, pp. 3200–3205.

[23] V. Padois, J.-Y. Fourquet, P. Chiron *et al.*, "Kinematic and dynamic model-based control of wheeled mobile manipulators: A unified framework for reactive approaches," *Robotica*, vol. 25, no. 2, p. 157, 2007.

[24] L. Saab, P. Soueres, and J.-Y. Fourquet, "Coupling manipulation and locomotion tasks for a humanoid robot," in *International Conference on Advances in Computational Tools for Engineering Applications (ACTEA)*, july 2009, pp. 84–89.

[25] N. Mansard, O. Khatib, and A. Kheddar, "A unified approach to integrate unilateral constraints in the stack of tasks," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 670–685, june 2009.

[26] F. Flacco, A. De Luca, and O. Khatib, "Motion control of redundant robots under joint constraints: Saturation in the null space," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 285–292.

[27] L. Saab, O. Ramos, F. Keith, N. Mansard, P. Soueres, and J.-Y. Fourquet, "Dynamic whole-body motion generation under rigid contacts and other unilateral constraints," *Robotics, IEEE Transactions on*, vol. 29, no. 2, pp. 346–362, 2013.

[28] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *The International Journal of Robotics Research*, p. 0278364914521306, 2014.

[29] J. Salini, V. Padois, and P. Bidaud, "Synthesis of complex humanoid whole-body behavior: A focus on sequencing and tasks transitions," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, may 2011, pp. 1283 –1290.

[30] K. E. C. and M. J. M., "Soft constraints and exact penalty functions in model predictive control," in *Proceedings og the UKACC International Conference*, Cambridge, UK, September 2000.

[31] F. Keith, N. Wieber, P.-B.and Mansard, and A. Kheddar, "Analysis of the discontinuities in prioritized tasks-space control under discreet task scheduling operations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 3887–3892.

[32] T. Petrič and L. Žlajpah, "Smooth continuous transition between tasks on a kinematic control level: Obstacle avoidance as a control problem," *Robotics and Autonomous Systems*, vol. 61, no. 9, pp. 948–959, 2013.

[33] N. Mansard, A. Remazeilles, and F. Chaumette, "Continuity of varying-feature-set control laws," *Automatic Control, IEEE Transactions on*, vol. 54, no. 11, pp. 2493–2505, 2009.

[34] J. Lee, N. Mansard, and J. Park, "Intermediate desired value approach for task transition of robots in kinematic control," *Robotics, IEEE Transactions on*, vol. 28, no. 6, pp. 1260–1277, 2012.

[35] A. Dietrich, A. Albu-Schaffer, and G. Hirzinger, "On continuous null space projections for torque-based, hierarchical, multi-objective manipulation," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, May 2012, pp. 2978–2985.

[36] O. Khatib, L. Sentis, J. Park, and J. Warren, "Whole-body dynamic behavior and control of human-like robots," *International Journal of Humanoid Robotics*, vol. 1, no. 1, pp. 29–43, 2004.

[37] C. Samson, M. L. Borgne, and B. Espiau, *Robot contol: the Task Function Approach.* Oxford, United Kingdom: Addison-Wesley Longman Publishing Co., Inc., 1991.

[38] O. Khatib, "Inertial properties in robotic manipulation: An object-level framework," *The International Journal of Robotics Research*, vol. 14, no. 1, pp. 19–36, 1995.

[39] K.-S. Chang and O. Khatib, "Efficient algorithm for extended operational space inertia matrix," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1. IEEE, 1999, pp. 350–355.

[40] B. Siciliano and J.-J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, june 1991, pp. 1211 –1216 vol.2.

[41] P. Baerlocher and R. Boulic, "Task-priority formulations for the kinematic control of highly redundant articulated structures," in *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, vol. 1, 1998, pp. 323–329 vol.1.

[42] H. Sadeghian, L. Villani, M. Keshmiri, and B. Siciliano, "Dynamic multi-priority control in redundant robotic systems," *Robotica*, vol. 31, pp. 1155–1167, 10 2013.

[43] J. Salini, "Dynamic control for the task/posture coordination of humanoids: toward synthesis of complex activities," Ph.D. thesis, Université Pierre et Marie Curie, Paris, France, June 2012.

[44] J. Andersson, J. kesson, and M. Diehl, "Casadi - a symbolic package for automatic differentiation and optimal control," *Recent Advances in Algorithmic Differentiation*, 2012.

[45] O. Kanoun, F. Lamiraux, and P.-B. Wieber, "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 785–792, aug. 2011.

[46] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Companion report," Tech. Rep. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00970816