Generalized hierarchical control

Mingxing Liu, Yang Tan, Vincent Padois

manuscript No.
(will be inserted by the editor)

Generalized Hierarchical Control

Mingxing Liu, Yang Tan, and Vincent Padois

Abstract Multi-objective control systems for complex robots usually have to handle multiple prioritized tasks. Most existing hierarchical control techniques handle either strict task priorities by using null-space projectors or a sequence of quadratic programs; or non strict task priorities by using a weighting strategy. This paper proposes a novel approach to handle both strict and non-strict priorities of an arbitrary number of tasks. It can 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. This projector can be used to perform priority transitions and task insertion or deletion. The control input is computed by solving one quadratic programming problem, where generalized projectors are adopted to maintain a task hierarchy, and equality or inequality constraints can be implemented. The effectiveness of this approach is demonstrated on a simulated robotic manipulator in a dynamic environment.

Keywords Redundant robots · task hierarchy · priority switching · dynamics · torque-based control

1 Introduction

Redundant robots are nowadays expected to perform complex missions involving the simultaneous performance of multiple tasks. 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, controllers for complex robots must be able to handle multiple prioritized tasks and respect various constraints imposed by the robot body and the environment.

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*, which 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*. The solution of these approaches is a compromise among task objectives of different weights. 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.

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. (i) The development of a generalized projector, which can regulate 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. (ii) The development of a generic dynamic hierarchical control framework, which solves a single quadratic program (QP) and

M. Liu · Y. Tan · V. Padois
Sorbonne Universités, UPMC Univ Paris 06, UMR 7222, Institut des Systèmes Intelligents et de Robotique (ISIR), F-75005, Paris, France
CNRS, UMR 7222, ISIR, F-75005, Paris, France
E-mail: {liu, padois}@isir.upmc.fr

uses generalized projectors as a mechanism to account for an arbitrary number of strict and non-strict task priorities. It can achieve desired priority transitions, as well as an elegant way of inserting and deleting tasks among those to be performed. Task hierarchies are handled by the modulation of a priority matrix, without the necessity of modifying the control problem formulation each time the hierarchies change.

2 Related Works

This section reviews some classical hierarchical control frameworks and priority transitions within them.

2.1 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. The idea is based on the use of the limited Jacobian of a lower-priority task, which is projected into the null-space of higher priority tasks by the application of a null-space projector [25]. Such an idea is applied in prioritized inverse kinematics [30, 31], in acceleration based control [14, 15], and in joint torque based control [21, 41, 44]. A generic framework, from which several existing control laws can be derived, is presented in [33]. Projected inverse dynamics schemes are developed for constrained systems in [2, 23], where the dynamics equation is projected into the null-space of the Jacobian of constraint equations. As the limited Jacobians mentioned above could be rank deficient, task priority strategies involving their pseudo-inverses may lead to algorithmic singularities. To overcome the effects of such singularities, a technique based on damped least-squares and extended Jacobian [10] is proposed, which requires to choose the damping factor carefully in order to guarantee good behaviors near singularities.

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 [20], from which repulsive forces are derived to prevent the robot from entering into activation zones of the inequality constraints [20, 32, 35, 42, 43, 46]. However, performing these tasks cannot guarantee that these inequality constraints are actually met. The approach presented in [28] integrates unilateral constraints at any priority level, albeit time consuming. The algorithm introduced in [13, 14] 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 [12, 17, 36, 37]. 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.

Generally, approaches that handling strict hierarchies parameterize the relative priority of one task with respect to another one of a different importance level in a lexicographic way [37]: 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 relative priority of tasks. 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. Compared with a discrete parameterization of strict task priorities, a continuous parameterization of both strict and non-strict task priorities is richer and more informative. Therefore, this work uses a continuous priority parameterization. Moreover, priorities are defined here by pairs of tasks and are encoded by a priority matrix. This choice of priority representation can handle not only a single standard lexicographic hierarchy as HQP does, but also a complex priority network. For example, it can represent two lexicographic hierarchies $1 \triangleright 2 \triangleright 3$[1] and $4 \triangleright 5 \triangleright 6$, with an additional relationship $2 \triangleright 5$, leaving the relationships among all the other pairs of tasks free.

2.2 Approaches for handling a non-strict hierarchy

Non-strict priorities are usually handled by approaches using weighting strategies [1, 5, 7, 26, 40]. 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 guaran-

[1] The notation $i \triangleright j$ indicates that task i has a strict higher priority over task j.

teed by simply adjusting the weights of task objectives, a prioritized control framework is proposed in [27] to ensure the performance of a higher-priority task. However, this approach handles priorities of only two levels. In approaches based on weighting strategies, task priorities can be parameterized continuously. Nonetheless, even though the work in [9] 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 outperforms weighting strategies by permitting priorities to change gradually from a non-strict case to a strict case.

2.3 Task transitions

Recently, different methods have been developed to handle task transition problems. An approach to smooth priority rearrangement between only two levels of tasks is proposed in [19, 34]. A specific inverse operator is proposed in [29] to ensure continuous inverse in the analytical computation of control laws. The approach presented in [24] 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 [8]. However, the design of an activator used by this approach makes it 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 [40]. This method is used in HQP approaches to swap the priorities of tasks coming from two consecutive priority levels [16]. This is achieved by tuning the weights of the tasks, which are merged in the same priority level, to comply with their priorities before and after the transition phase. However this strategy may require a set of swaps before bringing a task to the desired priority level. A novelty of the control framework proposed in this paper is that it allows the simultaneous priority rearrangements for an arbitrary number of pairs of tasks, and it requires only one swap to switch priorities between each pair of tasks at two non-consecutive levels.

3 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

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

where $\mathbf{M}(\boldsymbol{q}) \in \mathbb{R}^{n \times n}$ 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; $\mathbf{J}_c(\boldsymbol{q})^T = [\mathbf{J}_{c,1}(\boldsymbol{q})^T \ \dots \ \mathbf{J}_{c,n_c}(\boldsymbol{q})^T \ \mathbf{S}(\boldsymbol{q}, \dot{\boldsymbol{q}})^T]$ is the transpose of a Jacobian matrix, with $\mathbf{J}_{c,n_\beta}(\boldsymbol{q})$, the Jacobian matrix associated to a contact point β and $\mathbf{S}(\boldsymbol{q}, \dot{\boldsymbol{q}})^T \in \mathbb{R}^{n \times n_a}$, 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 i of a physical frame attached to the robot body can be defined by the following characteristics:

(I) A task variable $\boldsymbol{\xi}_i \in \mathbb{R}^{m_i}$ expressed in terms of some goals to be achieved by the task frame in the task space, such as a desired position or orientation of dimension m_i. The second order derivative of $\boldsymbol{\xi}_i$ can be linearly related to that of $\boldsymbol{q}$

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

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

(II) A local controller $\ddot{\boldsymbol{\xi}}_i^d$, the goal of which is to correct task errors and ensure the convergence of the task variable $\boldsymbol{\xi}_i$ towards its desired trajectory. For task motion control, $\ddot{\boldsymbol{\xi}}_i^d$ can take the form of a proportional-integral-derivative controller with a feed-forward term. For task wrench control, $\ddot{\boldsymbol{\xi}}_i^d$ can take the form of a proportional-integral controller with a feed-forward term. The wrench task can be expressed as a motion task using the inverse of the operational space inertia matrix $\boldsymbol{\Lambda}_i(\boldsymbol{q}) = [\mathbf{J}_i(\boldsymbol{q})\mathbf{M}(\boldsymbol{q})^{-1} \ \mathbf{J}_i(\boldsymbol{q})^T]^{-1}$ [6, 22]

$$\ddot{\boldsymbol{\xi}}_i^d = \boldsymbol{\Lambda}_i(\boldsymbol{q})^{-1}\boldsymbol{w}_i^d \tag{3}$$

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

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

$$\mathbf{A}_i = diag\left(\alpha_{i1}\mathbf{I}_{m_1}, \dots, \alpha_{ij}\mathbf{I}_{m_j}, \dots, \alpha_{in_t}\mathbf{I}_{m_{n_t}}\right) \tag{4}$$

where $\mathbf{A}_i$ is a diagonal matrix, the main diagonal blocks of which are square matrices: $\alpha_{ij}\mathbf{I}_{m_j}$. $\mathbf{I}_{m_j}$ 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 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.

The role of the particular element α_{ii} is given and explained in detail in section 4.2.1.

4 Generalized Projector for Hierarchical Control

The hierarchical control proposed in this paper is based on a new generalized projector, which can precisely regulate how much a task is affected by other tasks. The following part of this subsection first looks at several forms of projectors, then the analysis of which leads to the development of the generalized projector.

4.1 Review of projectors for hierarchical control

Strict priorities can be handled by analytical methods using a null-space projector $\mathbf{N}_j = \mathbf{I} - \mathbf{J}_j^\dagger \mathbf{J}_j$, where $\mathbf{J}_j^\dagger$ is the Moore-Penrose pseudo-inverse of the Jacobian $\mathbf{J}_j$[2]. The projection of a task i into the null-space of another task j can ensure that the lower-priority task i is performed without producing any motion for the higher-priority task j. 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 of all the higher priority tasks [4, 45].

To achieve smooth priority transitions, the null-space projector is replaced by the following matrix in [19, 34]

$$\mathbf{N}_j^{'}(\alpha_{ij}) = \mathbf{I} - \alpha_{ij} \mathbf{J}_j^\dagger \mathbf{J}_j, \tag{5}$$

where a scalar parameter $\alpha_{ij} \in [0,1]$ is used to regulate the priority between two tasks i and j. The greater α_{ij} is, the more task i is 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 priority levels.

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

The following matrix $\mathbf{N}^{''}$ is proposed in [8] for continuous null-space projections

$$\mathbf{N}^{''} = \mathbf{I} - \mathbf{V} \boldsymbol{\Theta} \mathbf{V}^T, \tag{6}$$

where $\mathbf{V} \in \mathbb{R}^{n \times n}$ is the right singular vectors of $\mathbf{J}_j$, the Jacobian of a higher priority task, and $\boldsymbol{\Theta} \in \mathbb{R}^{n \times n}$ is a diagonal activation matrix. The k-th diagonal element of $\boldsymbol{\Theta}$, $\theta_{kk} \in [0,1]$, refers to the k-th column vector in $\mathbf{V}$: when $\theta_{kk} = 1$, the k-th direction in $\mathbf{V}$ is activated in $\mathbf{N}^{''}$; when $0 < \theta_{kk} < 1$, the k-th direction in $\mathbf{V}$ is partially deactivated; when $\theta_{kk} = 0$, the k-th direction in $\mathbf{V}$ is deactivated. As mentioned in [8], for any one-dimensional task j ($\mathbf{J}_j \in \mathbb{R}^{1 \times n}$), the matrix (6) becomes

$$\mathbf{N}_j^{''} = \mathbf{I} - \theta_{11} \frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|} \frac{\mathbf{J}_j}{\|\mathbf{J}_j\|}, \tag{7}$$

where only the first element θ_{11} of $\boldsymbol{\Theta}$ is relevant. $\mathbf{N}_j^{''}$ can be applied to achieve activation or deactivation of task j direction in the projection matrix by the variation of the scalar θ_{11}. When extended to a task (or a set of tasks) of m directions ($\mathbf{J}_j \in \mathbb{R}^{m \times n}$), this method allows one to apply the same transition to all the m directions of $\mathbf{J}_j$, but its application for achieving the separate regulation of each task direction is not easy. This is because each activator θ_{kk} is directly referred to the k-th direction in the right singular vectors of $\mathbf{J}_j$, but not directly referred to a specific direction in $\mathbf{J}_j$.

4.2 Generalized projector

In order to achieve variations of multiple task priorities simultaneously among an arbitrary number of tasks, and to be able to ensure a priority network with both strict and non-strict priorities, an approach to the computation of a generalized projector $\mathbf{P}_i(\mathbf{A}_i) \in \mathbb{R}^{n \times n}$ is developed in this section. Here the subscript i in $\mathbf{P}_i$ indicates that the projector takes into account the priorities of a set of tasks with respect to task i. The dependence of $\mathbf{P}_i$ to $\mathbf{A}_i$ is sometimes omitted hereafter for clarity reasons. Similarly to the form of the matrix $\mathbf{N}^{''}$ in the case of considering a one-dimensional task (7), the form of $\mathbf{P}_i$ is obtained without the necessity of the computation of pseudo-inverse matrices. Moreover, the new projector allows one to regulate the activation of each task directions in a more intuitive way, by regulating the priority matrix $\mathbf{A}_i$ that is more closely related to task directions than the activator $\boldsymbol{\Theta}$ in (6).

First, look at the following matrix, which extends $\mathbf{N}_j^{''}$ defined by (7) from the handling of one task direction to the handling of the directions of n_t tasks

$$\mathbf{N}^{'''} = \mathbf{I} - \sum_{j=1}^{n_t} \alpha_{ij} \frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|} \frac{\mathbf{J}_j}{\|\mathbf{J}_j\|}, \tag{8}$$

where, without the loss of generality, each task dimension is supposed to be 1, and α_{ij} with $j = 1, 2, 3, \ldots, n_t$ parameterizes the priority of each of the n_t tasks with respect to a certain task i. For any task k among the n_t tasks with $\alpha_{ik} = 1$, which means that task k is of the highest priority, the product of $\mathbf{N}^{'''}$ with $\mathbf{J}_k$ leads to

$$\begin{aligned}\mathbf{J}_k\mathbf{N}^{'''} &= \mathbf{J}_k - \mathbf{J}_k\frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|}\frac{\mathbf{J}_j}{\|\mathbf{J}_j\|} - \sum_{j\neq k}\mathbf{J}_k\alpha_{ij}\frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|}\frac{\mathbf{J}_j}{\|\mathbf{J}_j\|} \\ &= -\sum_{j\neq k}\mathbf{J}_k\alpha_{ij}\frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|}\frac{\mathbf{J}_j}{\|\mathbf{J}_j\|}.\end{aligned} \tag{9}$$

In (9), $\mathbf{J}_k\mathbf{N}^{'''} = 0$ if $\mathbf{J}_k\mathbf{J}_j^T = 0$ for each $j \neq k$ and $\alpha_{ij} > 0$. This means that the highest priority of task k may not be satisfied if it is interfered by a lower priority task j, which has a component along task k direction. On the contrary, task priorities can be maintained if such task interferences disappear, or in other words, if all the lower priority task directions become orthogonal to all the higher priority task directions. Based on this observation, the computation of the generalized projector $\mathbf{P}_i$ is divided into three steps.

Step one is a preliminary processing of the matrices $\mathbf{J}$ and $\mathbf{A}_i$, where

$$\mathbf{J} = \left[\mathbf{J}_1^T \ldots \mathbf{J}_j^T \ldots \mathbf{J}_{n_t}^T\right]^T \tag{10}$$

is the augmented Jacobian concatenating the Jacobian matrices of all the n_t tasks. The processing of $\mathbf{J}$ and $\mathbf{A}_i$ is carried out according to the priorities of the n_t tasks with respect to task i. As each row of $\mathbf{J}$ is associated to α_{ij}, the rows of $\mathbf{J}$ can be sorted in descending order with respect to the values of the diagonal elements in $\mathbf{A}_i$. The resulting matrix $\mathbf{J}_{s_i}$ 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 $\mathbf{A}_i$ are sorted accordingly, leading to $\mathbf{A}_i^s$, the diagonal elements of which are organized in descending order starting from the first row.

Step two consists in the computation of a matrix $\mathbf{B}_i(\mathbf{J}_{s_i}) \in \mathbb{R}^{r\times n}$ by using $\mathbf{J}_{s_i}$, where r is the rank of $\mathbf{J}_{s_i}$. The rows of $\mathbf{B}_i(\mathbf{J}_{s_i})$ form an orthonormal basis of the joint space obtained using elementary row transformations on $\mathbf{J}_{s_i}$. Algorithm (1) describes this computation. As in any numerical scheme, tolerances are used here for numerical comparison, such as ϵ in line #11 of Algorithm (1), which is defined as a small positive value. As the use of ϵ may lead to rank jumps in $\mathbf{B}_i$, it is suggested to assign the smallest value greater than zero to ϵ to avoid large variation of $\mathbf{B}_i$.

Step three is to compute the generalized projector, which is given by

$$\mathbf{P}_i(\mathbf{A}_i) = \mathbf{I}_n - \mathbf{B}_i(\mathbf{J}_{s_i})^T\mathbf{A}_i^{r,s}(\mathbf{A}_i, \boldsymbol{origin})\mathbf{B}_i(\mathbf{J}_{s_i}), \tag{11}$$

where $\mathbf{A}_i^{r,s}$ is a diagonal matrix of degree r. The vector $\boldsymbol{origin} \in \mathbb{R}^r$ is a vector of the row indexes of $\mathbf{J}_{s_i}$ selected during the construction of the orthonormal basis $\mathbf{B}_i$. Each of these r rows in $\mathbf{J}_{s_i}$ is linearly independent to all the previously selected ones. The diagonal elements of $\mathbf{A}_i^{r,s}$ are restricted to the r diagonal elements of $\mathbf{A}_i^s$, which correspond to the r rows of $\mathbf{J}_{s_i}$, the row indexes of which belong to $\boldsymbol{origin}$. Algorithm (2) summarizes the construction of the generalized projector.

Note that the interference of lower priority tasks with higher priority tasks, which exists in (8) if two task directions of different priorities are not orthogonal ($\mathbf{J}_k\mathbf{J}_j^T \neq 0$), is avoided in $\mathbf{P}_i$. Indeed, each row in $\mathbf{B}_i$ corresponds to the component of a task direction that is effectively accounted for by the projector $\mathbf{P}_i$. The row sorting in step one ensures that higher priority task directions are accounted for in $\mathbf{B}_i$ prior to any lower priority task direction, and the orthonormalization process in step two ensures that each direction (or row) of $\mathbf{B}_i$ is orthogonal to previous rows associated to all the higher priority task directions.

By varying the value of each α_{ij} in $\mathbf{A}_i$, one can regulate the priority of each task j with respect to task i separately. Indeed, during the execution of task i, the projector $\mathbf{P}_i$ can be configured such that

- for tasks having strict priority over task i, the movement along their task directions is completely forbidden by setting corresponding $\alpha_{i\bullet}$ to 1;
- for tasks over which task i has a strict priority, the movement along their directions is completely allowed by setting corresponding $\alpha_{i\bullet}$ to 0;
- and for tasks with non strict priorities, the movement along their task directions is partially allowed according to the value of their priority parameters. The increase of the values of corresponding $\alpha_{i\bullet} \in (0, 1)$ leads to the increase of the priorities of the associated tasks with respect to task i, and thus stronger restriction of task i movements along their task directions.

4.2.1 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 activates task i gradually. Conversely, increasing α_{ii} continuously from 0 to 1 deactivates the task gradually.

Algorithm 1: Orthonormal basis computation - $GetOrthBasis(\mathbf{J})$

Data: $\mathbf{J}$, ϵ
Result: $\mathbf{B}$, ***origin***, r
begin
  $n \longleftarrow GetNbCol(\mathbf{J})$
  $m \longleftarrow GetNbRow(\mathbf{J})$
  $i \longleftarrow 0$
  for $k \leftarrow 0$ **to** $m-1$ **do**
    if $i \geq n$ **then**
      break
    $\mathbf{B}[i,:] \longleftarrow \mathbf{J}[k,:]$
    for $j \leftarrow 0$ **to** $i-1$ **do**
      $\mathbf{B}[i,:] \longleftarrow \mathbf{B}[i,:] - \left(\mathbf{B}[i,:]\mathbf{B}[j,:]^T\right)\mathbf{B}[j,:]$
    if $norm(B[i,:]) > \epsilon$ **then**
      $\mathbf{B}[i,:] \longleftarrow \mathbf{B}[i,:]/\,norm(\mathbf{B}[i,:])$
      origin$[i] \longleftarrow k$
      $i \longleftarrow i+1$
  $r \longleftarrow i$
  return $\mathbf{B}$, ***origin***, r

Algorithm 2: Generalized projector computation - task i

Data: $\mathbf{A}_i$, $\mathbf{J}$
Result: $\mathbf{P}_i$
begin
  $n \longleftarrow GetNbCol(\mathbf{J})$
  index $\longleftarrow GetRowsIndexDescOrder(\mathbf{A}_i)$
  $\mathbf{A}_i^s \longleftarrow SortRows(\mathbf{A}_i, \boldsymbol{index})$
  $\mathbf{J}_{s_i} \longleftarrow SortRows(\mathbf{J}, \boldsymbol{index})$
  $\mathbf{B}_i$, ***origin***$, r \longleftarrow GetOrthBasis(\mathbf{J}_{s_i})$ ▷`Alg. (1)`
  $\mathbf{A}_i^{r,s} \longleftarrow GetSubDiagMatrix(\mathbf{A}_i^s, \boldsymbol{origin})$
  $\mathbf{P}_i \longleftarrow \mathbf{I}_n - \mathbf{B}_i^T\mathbf{A}_i^{r,s}\mathbf{B}_i$
  return $\mathbf{P}_i$

5 Generalized Hierarchical Control Framework

This paper handles task hierarchies subject to linear constraints. This multi-objective control problem is formulated as a Linear Quadratic Programming (LQP) problem here, where all the task objectives and constraints are solved simultaneously in one LQP. Constraints are formulated in terms of priority consistent joint accelerations by applying generalized projectors.

This section first briefly reviews the LQP control framework that is commonly used by weighting strategies, then explains the implementation of generalized projectors in such a framework to achieve generalized hierarchical control.

5.1 LQP control framework for weighting strategies

When only non-strict task hierarchies are considered, weighting strategies, such as those proposed in [5, 7, 26, 40], can be applied to handle the relative priorities of multiple elementary tasks. In this case, the control problem can be formulated as a LQP problem as

$$\arg\min_{\ddot{\boldsymbol{q}},\boldsymbol{\chi}} \sum_{i=1}^{n_t} \left\| \boldsymbol{f}_i\left(\ddot{\boldsymbol{q}}, \ddot{\boldsymbol{\xi}}_i^d\right)\right\|_{\mathbf{Q}_i}^2 + \left\| \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\chi} \end{bmatrix}\right\|_{\mathbf{Q}_r}^2 \tag{12a}$$

$$\text{subject to} \quad \mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \mathbf{J}_c(\boldsymbol{q})^T\boldsymbol{\chi} \tag{12b}$$

$$\mathbf{G}(\boldsymbol{q},\dot{\boldsymbol{q}}) \begin{pmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\chi} \end{pmatrix} \leq \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{12c}$$

where $\boldsymbol{f}_i\left(\ddot{\boldsymbol{q}}, \ddot{\boldsymbol{\xi}}_i^d\right) = \mathbf{J}_i(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\mathbf{J}}_i(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \ddot{\boldsymbol{\xi}}_i^d$ is the objective function which measures the error of task i. The regulation term minimizes the norm of joint 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 [40]. In (12), $\mathbf{Q}_i = \omega_i\mathbf{I}_{m_i}$ is a weighting matrix to regulate the importance level of task i, with ω_i denoting the weight of task objective i. $\mathbf{Q}_r = \omega_r\mathbf{I}_{n+n_a+3n_c}$ is the weighting matrix of the regularization term, with ω_r denoting the weight value. As the regulation term may increase task error, ω_r is usually very small compared to ω_i. The equation of motion (12b) constitutes an equality constraint to ensure physical realism. The matrix $\mathbf{G}$ and the vector $\boldsymbol{h}$ express some other equality or inequality constraints, such as actuation capabilities (maximum actuator torques and velocities), geometrical limits (joint limits, Cartesian space obstacles), and contact wrenches (contact existence conditions, bounds on the norms of contact wrenches).

By solving (12), the solution of $\ddot{\boldsymbol{q}}$ and $\boldsymbol{\chi}$ can be obtained, from which the solution of joint torques is extracted.

5.2 Generalized hierarchical control using generalized projectors

The control framework based on weighting strategy (12) can qualitatively regulate the relative priorities of tasks by weighting task objectives, but it cannot ensure strict task priorities. The GHC framework proposed here extends framework (12) through the implementation of generalized projectors defined by (11) to handle a priority network with both strict and non-strict task priorities.

Consider the control problem for solving n_t tasks. The operating principle of GHC is summarized by the following LQP problem, which takes into account the desired task priorities parameterized by the priority

matrix $\mathbf{A}_i$.

$$\underset{\ddot{\boldsymbol{q}}',\boldsymbol{\chi}}{\arg\min} \sum_{i=1}^{n_t} \left\| \boldsymbol{f}_i \left(\ddot{\boldsymbol{q}}_i', \ddot{\boldsymbol{\xi}}_i^d \right) \right\|^2 + \left\| \begin{bmatrix} \ddot{\boldsymbol{q}}' \\ \boldsymbol{\chi} \end{bmatrix} \right\|_{\mathbf{Q}_r}^2 \tag{13a}$$

$$\text{subject to} \quad \mathbf{J}_c(\boldsymbol{q})^T \boldsymbol{\chi} = \mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{13b}$$

$$\mathbf{G}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \begin{pmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\chi} \end{pmatrix} \leq \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{13c}$$

$$\ddot{\boldsymbol{q}} = \mathbf{P}\ddot{\boldsymbol{q}}' = \textstyle\sum_{i=1}^{n_t} \mathbf{P}_i(\mathbf{A}_i)\ddot{\boldsymbol{q}}_i' \tag{13d}$$

with $\ddot{\boldsymbol{q}}' = \begin{bmatrix} \ddot{\boldsymbol{q}}_1' \\ \vdots \\ \ddot{\boldsymbol{q}}_{n_t}' \end{bmatrix}$ and $\mathbf{P} = [\mathbf{P}_1(\mathbf{A}_1) \dots \mathbf{P}_{n_t}(\mathbf{A}_{n_t})]$. Each $\ddot{\boldsymbol{q}}_i'$ in (13) is an intermediate joint acceleration variable associated to each task i and $\ddot{\boldsymbol{q}}$ is the overall joint accelerations accounting for the sets of desired task priorities $(\mathbf{A}_1, \dots, \mathbf{A}_{n_t})$. Assuming a perfect model, $\ddot{\boldsymbol{q}}$ is the joint accelerations resulting from the application of the joint torques computed by solving (13).

This optimization problem minimizes the objective function of each task as well as the magnitude of the control input, subject to constraints. Each task objective function is expressed in terms of the intermediate joint acceleration variable $\ddot{\boldsymbol{q}}_i'$. Note that in GHC, task priorities are handled by using the generalized projectors $\mathbf{P}_i$ in (13d) instead of task weights ω_i. Therefore, here the task weighting matrix $\mathbf{Q}_i$ is set to the identity matrix, which is omitted in (13a).

A solution to the equation of motion (13b) can be ensured as long as there exists a highest priority task i such that $\mathbf{P}_i(\mathbf{A}_i) = \mathbf{I}_n$ (with $\mathbf{A}_i$ being the zero matrix), which means that this task is not projected in the nullspace of any other task. Indeed, (13b) can be expressed in terms of intermediate joint accelerations as

$$\mathbf{J}_c(\boldsymbol{q})^T \boldsymbol{\chi} = \mathbf{M}(\boldsymbol{q})\mathbf{P}\ddot{\boldsymbol{q}}' + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}), \tag{14}$$

with $\mathbf{P} = [\mathbf{P}_1(\mathbf{A}_1) \dots \mathbf{P}_{n_t}(\mathbf{A}_{n_t})]$. As the inertia matrix $\mathbf{M}$ is positive definite, a solution to (14), and thus (13b), can be ensured if $\mathbf{P}$ has full row rank. A sufficient condition to ensure this property of $\mathbf{P}$ is that there exists at least one $\mathbf{P}_i$ which equals the identity matrix, and this is the case for the highest priority task in a hierarchy.

Since the constraints have a higher priority than the objectives in LQP, and in (13) the constraints are expressed in terms of the overall joint accelerations $\ddot{\boldsymbol{q}}$, it is ensured that the solution accounting for desired task hierarchies satisfies the constraints. Or in other words, the GHC framework ensures that the constraints, such as the equation of motion (13b) and the other constraints of physical limitations (13c), have a higher priority over task hierarchies. Moreover, this GHC framework can handle strict task hierarchies represented by standard lexicographic orders. The proof is provided in Appendix A.

Another property of GHC is that it is robust to both kinematic and algorithmic singularities. In this framework based on LQP, tasks are expressed in a forward way and most LQP solvers do not require the explicit inversion of Jacobian matrices. Therefore, GHC does not have problems of numerical singularities due to kinematic singularities. Moreover, unlike approaches using the pseudo-inverse of limited Jacobians ($\mathbf{J}_i\mathbf{N}_j$), which requires special treatment for handling algorithmic singularities when the limited Jacobians drop rank [38], GHC does not require the inversion of priority consistent Jacobians. Therefore, the framework does not have to handle such kind of algorithmic singularities.

6 Results

The proposed GHC framework (13) is applied to the control of a 7-DoF KUKA LWR robot. The experiments are conducted in the Arboris-Python simulator [39], which is a rigid multibody dynamics and contacts simulator written in Python. The LQP problem is solved by a QP solver included in CasADi-Python [3], 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 (or position and force) of the end-effector, task 2 for the control of the three dimensional position of the elbow, and task 3 for the control of the 7-DoF posture. The elbow task target is a static target position and the posture task target is a static posture. For each task i, an optimization variable $\ddot{\boldsymbol{q}}_i' \in \mathbb{R}^7$ is defined. A local proportional-derivative controller $\ddot{\boldsymbol{\xi}}_i^d$ is used to ensure the convergence of each task variable towards its target. When a task target is static, $\ddot{\boldsymbol{\xi}}_i^d = k_p\boldsymbol{e}_i + k_d\dot{\boldsymbol{e}}_i$ with $k_p = 30s^{-2}$ and $k_d = 20s^{-1}$. When tracking a desired trajectory $\ddot{\boldsymbol{\xi}}_i^*$, $\ddot{\boldsymbol{\xi}}_i^d = \ddot{\boldsymbol{\xi}}_i^* + k_p\boldsymbol{e}_i + k_d\dot{\boldsymbol{e}}_i$ with $k_p = 100s^{-2}$ and $k_d = 20s^{-1}$. The priority parameter matrices associated with the three tasks are: $\mathbf{A}_i = diag(\alpha_{i1}\mathbf{I}_3, \alpha_{i2}\mathbf{I}_3, \alpha_{i3}\mathbf{I}_7)$ with $i = 1, 2, 3$. The regularization weight ω_r is chosen as 0.01. The following function is used for the smooth variation of α_{ij} (conversely α_{ji}) from 0 to 1 during the transition time period ($[t_1, t_2]$)

$$\begin{aligned} &\alpha_{ij}(t) = 0.5 - 0.5\cos\left(\frac{t - t_1}{t_2 - t_1}\pi\right), \quad t \in [t_1, t_2], \\ &\alpha_{ji}(t) = 1 - \alpha_{ij}(t). \end{aligned} \tag{15}$$

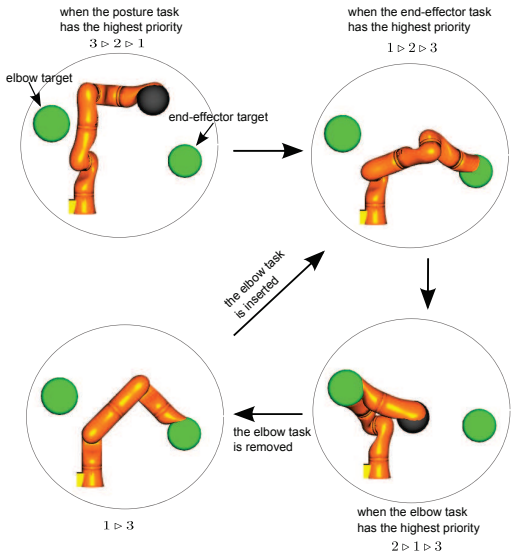

Fig. 1 Experiment of priority switching.

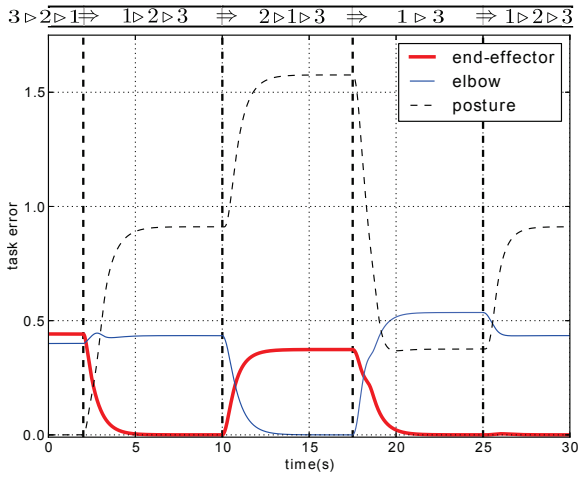

Fig. 2 Task errors using HQP, with fixed task targets. Priority transitions as well as the insertion and deletion of the elbow task are performed. The hierarchy rearrangement is instantaneous.

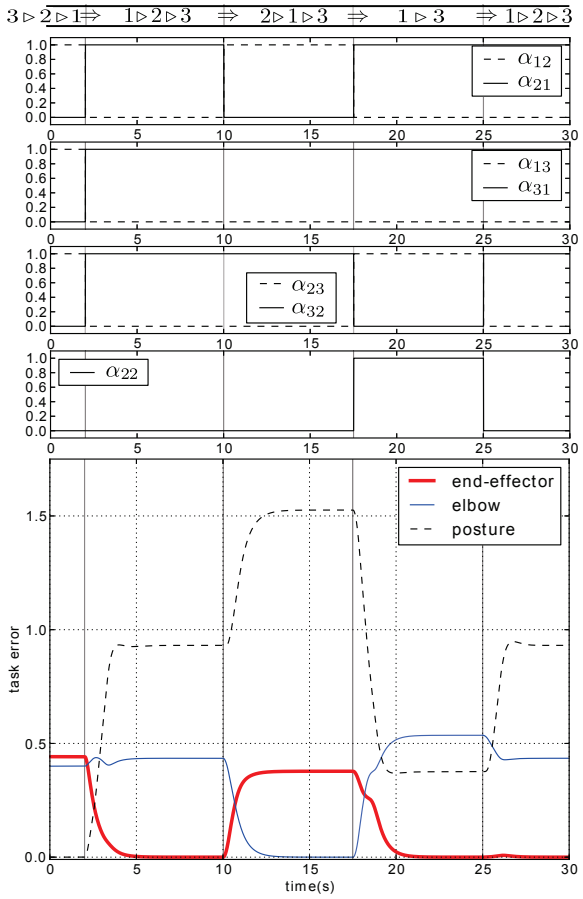

Fig. 3 Evolution of αs (top) and task errors (bottom) using GHC, with fixed task targets. Priority transitions as well as the insertion and deletion of the elbow task are performed. The hierarchy rearrangement duration is 0.005 seconds.

6.1 Hierarchical control with priority transitions, task insertions, and task deletions

In the experiments, task hierarchy is changed four times (see Fig. 1), and the equality constraint (13b) as well as inequality constraints, such as joint velocity and joint torque limits, are imposed. The evolution of the task hierarchy is $3\triangleright 2\triangleright 1 \Rightarrow 1\triangleright 2\triangleright 3 \Rightarrow 2\triangleright 1\triangleright 3 \Rightarrow 1\triangleright 3 \Rightarrow 1\triangleright 2\triangleright 3$. 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. Afterward, 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 with its priority level between those of the end-effector task and the posture task.

The experiment is carried out first using static task targets for steady state error analysis, then using a dynamic end-effector trajectory of a lemniscate shape. Moreover, the performance of GHC is compared with the HQP approach [18].

The results corresponding to the use of static task targets are presented in Fig. 2 to 5. Task errors by using HQP (Fig. 2) as well as those by using GHC with different hierarchy rearrangement durations (Fig. 3 and Fig. 4) are shown. The hierarchy rearrangement duration is 0.005 seconds in Fig. 3 and 2 seconds in Fig. 4. Fig. 5 shows the integration of the absolute values of the resulting joint jerks $\sum_{i=1}^{n}\left(\int_0^t \frac{|\mathrm{d}^3 q_i|}{\mathrm{d}t^3}dt\right)$ by using HQP that performs instantaneous hierarchy rearrangements, as well as by using GHC with faster and slower hierarchy rearrangements. Steady state task errors for each task hierarchy configuration are shown in Table 1, where the results using GHC and HQP are included.

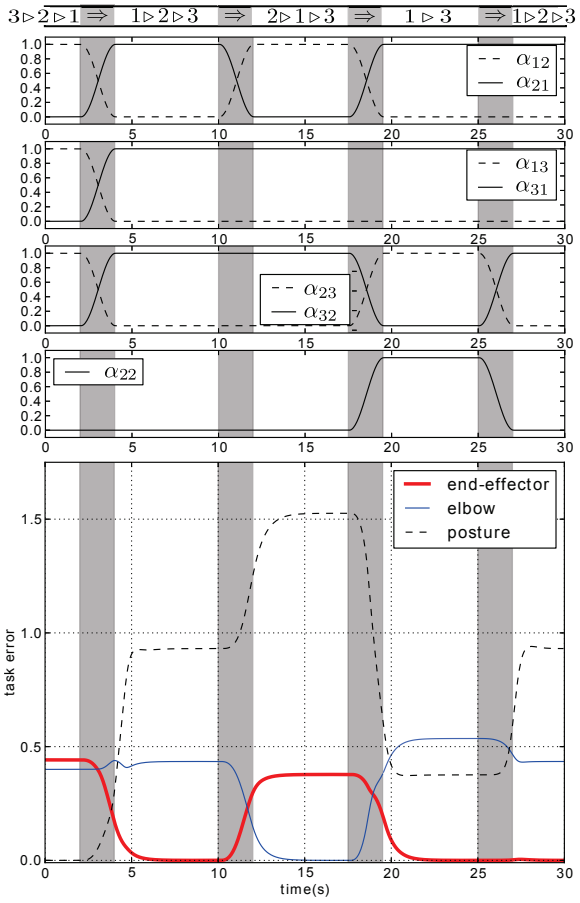

Fig. 4 Evolution of αs (top) and task errors (bottom) using GHC, with fixed task targets. Priority transitions as well as the insertion and deletion of the elbow task are performed. The hierarchy rearrangement duration is 2 seconds.

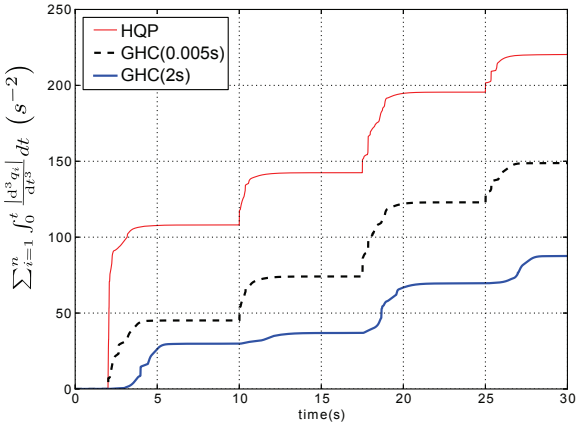

Fig. 5 Integration of the absolute values of joint jerks using GHC and HQP, with fixed task targets. The value is increased each time the task hierarchy is changed. GHC generates less amount of joint jerks by performing slower hierarchy rearrangements; while HQP, which perform instantaneous hierarchy rearrangements generates larger joint jerks.

GHC provides similar results in terms of task errors compared with HQP, as can be observed in Fig. 3 and

Table 1 Steady state task errors for each task hierarchy configuration

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

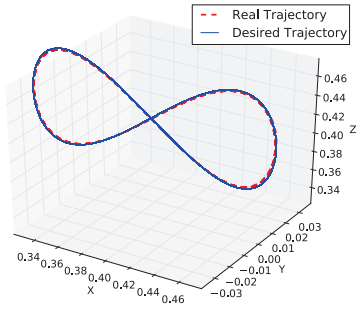

Fig. 6 The desired and the resulting end-effector trajectory provided by GHC, when the end-effector task has the highest priority. The end-effector moves along the lemniscate-shaped trajectory with an orbital period of 2π seconds.

2. The results of task errors in Table 1 show that both GHC and HQP can ensure strict priority. When controlled by either GHC or HQP, errors of the tasks with the highest priority are very small. Moreover, GHC can perform slower and smoother hierarchy rearrangements that require less joint jerks. This can be seen in Fig. 5, which shows that GHC can generate smaller joint jerks than HQP does.

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 2π seconds. The desired and the resulting end-effector trajectory is shown in Fig. 6. The resulting task errors using GHC is presented in Fig. 7. The resulting joint velocities and joint torques are shown in Fig. 8. A video of this experiment that presents the main features of GHC (priority transitions, the insertion and deletion of tasks) is attached to this paper.

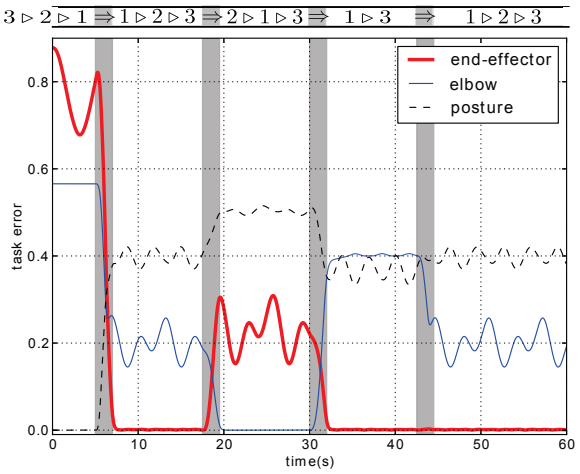

Fig. 7 Task errors using GHC, with the end-effector tracking a lemniscate-shaped trajectory. Desired priority transitions as well as the insertion and deletion of the elbow task are achieved. Strict priorities are maintained.

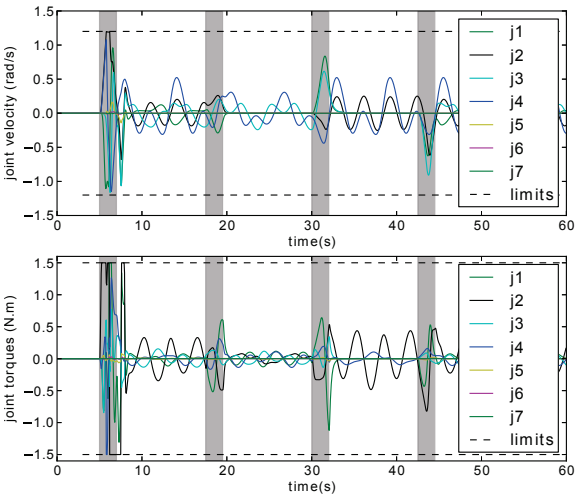

Fig. 8 Evolution of joint velocities and joint torques. The upper and lower bounds of $\dot{\boldsymbol{q}}$ are 1.2 rad/s and -1.2 rad/s, respectively. The upper and lower bounds of $\boldsymbol{\tau}$ are 1.5 $N \cdot m$ and -1.5 $N \cdot m$, respectively. These bounds are voluntarily set low in order to easily illustrate the fact that they are respected.

Fig. 7 shows that when the end-effector task has the highest priority, it can track its desired trajectory precisely. Moreover, Fig. 8 shows that joint velocity and joint torque limits are respected, which demonstrate that GHC can maintain desired task hierarchies while satisfying constraints.

6.2 Hierarchical control with a force task

In this experiment, the end-effector task is to move towards a plane, and then to apply a desired contact force against the plane in the vertical direction (see Fig. 9). Before the establishment of the contact with the plane,

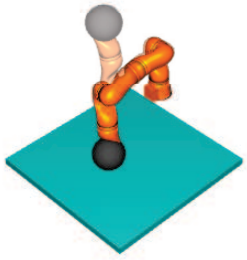

Fig. 9 The target end-effector position is on the plane. The end-effector starts from an initial position, which is above the target position and pointing upwards, then it should move towards the target position, and then apply desired forces to the plane.

the end-effector task (task 1) is a motion task. Once the contact is established, the end-effector task is a composition of a position subtask in the horizontal plane (task 1a) and a force subtask in the vertical direction (task 1b). The force task is transformed into a motion task by applying (3). The evolution of task hierarchy is $2 \triangleright 1 \triangleright 3 \Rightarrow 1 \triangleright 2 \triangleright 3$. At the beginning of this experiment, the elbow task has the highest priority, and the initial position of the end-effector is above its target position but pointing upwards. Then the priorities between the elbow task and the end-effector task switches. The increase of the priority level of the end-effector task allows the end-effector to point downwards, move towards its target position on the surface of the plane, and then push against the plane.

The change of αs, the errors of the elbow position and the end-effector horizontal position, as well as the error of the end-effector contact force are shown in Fig. 10. After the priority switch, the highest priority of the end-effector task allows the controlled frame to achieve its target position and to follow its contact force reference after the contact establishment, except for the impact peak at the moment when the contact is established between the two rigid bodies. This result illustrates the fact that the highest priority of the end-effector task, including both the horizontal position control component and the vertical force control component, is maintained.

6.3 Control with a non-strict hierarchy

In the previous experiments, non-strict priorities are used only in the transition phase. Experiments here handle constant non-strict priorities by using GHC. The elbow task and the end-effector task are considered, with α_{12} being set to a value between 0 and 1. Fig. 11 shows the task errors with respect to different values of α_{12}. It can be seen in this figure that by setting

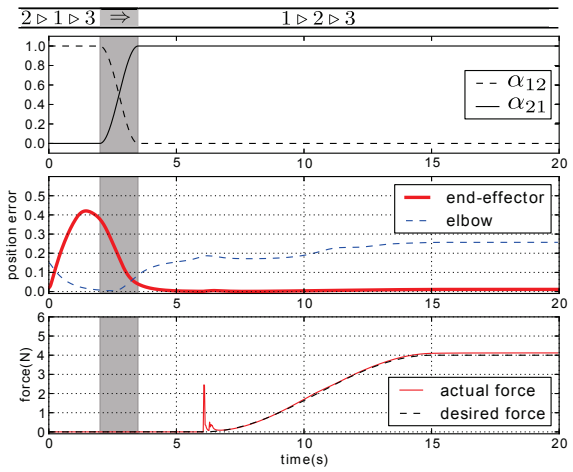

Fig. 10 Results of contact force control. The top figure shows the change of αs. The figure in the middle shows the end-effector position error in the horizontal plane as well as the elbow position error. The bottom figure represents the desired and resulting contact forces between the end-effector and the plane.

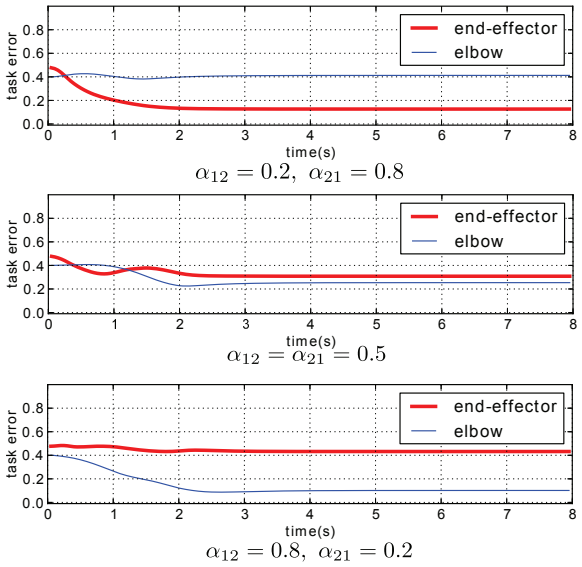

Fig. 11 Errors of the elbow task and the end-effector task with respect to their different relative priorities. Three priority values are applied: $\alpha_{12} = 0.2$, $\alpha_{12} = 0.5$, and $\alpha_{12} = 0.8$.

α_{12} to certain values between 0 and 1, no task is completely satisfied and non-strict hierarchies are achieved. When $\alpha_{12} = 0.5$, the errors of the two tasks show more or less an equal compromise between them. When α_{12} is increased, the performance of the end-effector task is reduced in order to improve the performance of the elbow task. In fact, the increase of α_{12} corresponds to the increase of the elbow task weight with respect to that of the end-effector when using a weighting strategy.

7 Discussion

In this section, the computation cost of GHC is analyzed, which shows that the computation time tends to increase with the number of DoF of the robot and the number of tasks. For a robot of n DoF performing k tasks of different priority levels with a total task dimension of m, the computation cost by using the HQP solver [11] 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 task dimension 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 an intermediate joint acceleration variable $\ddot{\boldsymbol{q}}_i' \in \mathbb{R}^n$ is created for each task i. In this case, one level of QP (13) 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 DoF 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 intermediate joint acceleration variables $\ddot{\boldsymbol{q}}' \in \mathbb{R}^{7n_1}$ and the joint torques $\tau \in \mathbb{R}^7$ needs to be solved for. For a fixed-based humanoid robot iCub with 32 DoF performing n_2 tasks, the number of variables would be $32(n_2 + 1)$. Fig. 12 shows the computation time of using GHC to solve randomly selected hierarchical control problems for the KUKA robot and the iCub robot performing different numbers of tasks. Each control problem consists of the constraint (13b), 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 correspond to a C++ implementation of the controller on a standard Linux PC.

8 Conclusions and Future Works

This paper proposes a generalized hierarchical control approach for handling tasks with both strict and non strict 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

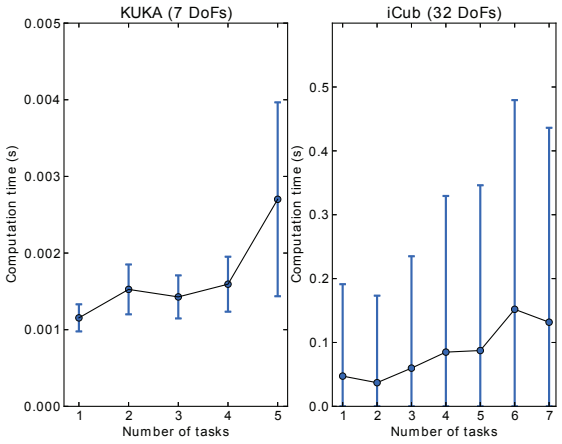

Fig. 12 Mean and standard deviation of the computation time per iteration, when using GHC to solve randomly selected hierarchical control problems for a fixed-based KUKA robot and a fixed-based iCub robot. Each control problem consists of a posture task and a set of 3D Cartesian motion tasks (0 to 4 motion tasks for KUKA and 0 to 6 motion tasks for iCub), subject to the whole-body equilibrium constraint (13b).

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.

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 or a quasi-static controller. The idea is to associate each task with an intermediate task variable in joint space ($\dot{\boldsymbol{q}}'_i$, $\ddot{\boldsymbol{q}}'_i$, $\boldsymbol{\tau}'_i$, 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($\mathbf{P}_i(\mathbf{A}_i)\dot{\boldsymbol{q}}'_i$, $\mathbf{P}_i(\mathbf{A}_i)\ddot{\boldsymbol{q}}'_i$, $\mathbf{P}_i(\mathbf{A}_i)\boldsymbol{\tau}'_i$, 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. Moreover, 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 of great interest.

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

This section proves that the proposed GHC approach (13) 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 $= 2, ..., n_t$. In this case, the generalized projector $\mathbf{P}_i$ for 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 $\mathbf{A}_i$ is an identity matrix. Let each task objective function be $\boldsymbol{f}_i = \mathbf{J}_i\boldsymbol{x}'_i - \boldsymbol{x}^d_i$, with $\boldsymbol{x}'_i$ being a joint space task variable, such as $\dot{\boldsymbol{q}}'_i$, $\ddot{\boldsymbol{q}}'_i$, or $\boldsymbol{\tau}'_i$, etc. Moreover, the global variable $\boldsymbol{x} = \sum_i \mathbf{P}_i\boldsymbol{x}'_i$ should satisfy linear equality or inequality constraints $\mathbf{G}\boldsymbol{x} \leq \boldsymbol{h}$.

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

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(n_t)}}{\arg\min} \sum_{i=1}^{n_t} \left\|\mathbf{J}_i\boldsymbol{x}'_i - \boldsymbol{x}^d_i\right\|^2 \\ &\text{subject to } \mathbf{G}\sum_{i=1}^{n_t}\mathbf{P}_i\boldsymbol{x}'_i \leq \boldsymbol{h} \end{aligned} \tag{16}$$

where $\boldsymbol{x}'_{(n_t)} = \{\boldsymbol{x}'_1, \boldsymbol{x}'_2, \dots \boldsymbol{x}'_{n_t}\}$, and the solution to (16) is denoted as $\boldsymbol{x}^*_{(n_t)} = \{\boldsymbol{x}^*_1, \boldsymbol{x}^*_2, \dots \boldsymbol{x}^*_{n_t}\}$.

When $n_t = 1$, the optimization problem can be written as

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(1)}}{\arg\min} \left\|\mathbf{J}_1\boldsymbol{x}'_{(1)} - \boldsymbol{x}^d_1\right\|^2 \\ &\text{subject to } \mathbf{G}\boldsymbol{x}'_{(1)} \leq \boldsymbol{h}. \end{aligned} \tag{17}$$

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

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

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(k)}}{\arg\min} \sum_{i=1}^{k} \left\|\mathbf{J}_i\boldsymbol{x}'_i - \boldsymbol{x}^d_i\right\|^2 \\ &\text{subject to } \mathbf{G}\sum_{i=1}^{k}\mathbf{P}_i\boldsymbol{x}'_i \leq \boldsymbol{h}. \end{aligned} \tag{18}$$

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

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(k+1)}}{\arg\min} \sum_{i=1}^{k} \left\|\mathbf{J}_i\boldsymbol{x}'_i - \boldsymbol{x}^d_i\right\|^2 + \left\|\mathbf{J}_{k+1}\boldsymbol{x}'_{k+1} - \boldsymbol{x}^d_{k+1}\right\|^2 \\ &\text{subject to } \mathbf{G}\left(\sum_{i=1}^{k}\mathbf{P}_i\boldsymbol{x}'_i + \mathbf{P}_{k+1}\boldsymbol{x}'_{k+1}\right) \leq \boldsymbol{h}. \end{aligned} \tag{19}$$

As $\mathbf{P}_k\mathbf{P}_{k+1} = \mathbf{P}_{k+1}$, the term $\sum_{i=1}^{k}\mathbf{P}_i\boldsymbol{x}'_i + \mathbf{P}_{k+1}\boldsymbol{x}'_{k+1}$ in the constraint in (19) is equivalent to $\sum_{i=1}^{k-1}\mathbf{P}_i\boldsymbol{x}'_i + \mathbf{P}_k\varsigma_k$, with

$$\varsigma_k = \boldsymbol{x}'_k + \mathbf{P}_{k+1}\boldsymbol{x}'_{k+1}. \tag{20}$$

Then problem (19) can be written as

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(k)},\varsigma_k,\boldsymbol{x}_{k+1}}{\arg\min} \sum_{i=1}^{k-1} \left\|\mathbf{J}_i\boldsymbol{x}'_i - \boldsymbol{x}^d_i\right\|^2 + \left\|\mathbf{J}_k\varsigma_k - \boldsymbol{x}^d_k\right\|^2 + \\ &\qquad \left\|\mathbf{J}_{k+1}\boldsymbol{x}'_{k+1} - \boldsymbol{x}^d_{k+1}\right\|^2 \\ &\text{subject to } \mathbf{G}\left(\sum_{i=1}^{k-1}\mathbf{P}_i\boldsymbol{x}'_i + \mathbf{P}_k\varsigma_k\right) \leq \boldsymbol{h} \\ &\qquad \varsigma_k = \boldsymbol{x}'_k + \mathbf{P}_{k+1}\boldsymbol{x}'_{k+1}. \end{aligned} \tag{21}$$

$\boldsymbol{x}'_k$ in (21) is a free variable, and this problem can be separated into two sub-problems. The first sub-problem is

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(k-1)},\varsigma_k}{\arg\min} \sum_{i=1}^{k-1} \left\|\mathbf{J}_i\boldsymbol{x}'_i - \boldsymbol{x}_i^d\right\|^2 + \left\|\mathbf{J}_k\varsigma_k - \boldsymbol{x}_k^d\right\|^2 \\ &\text{subject to } \mathbf{G}\left(\sum_{i=1}^{k-1}\mathbf{P}_i\boldsymbol{x}'_i + \mathbf{P}_k\varsigma_k\right) \leq \boldsymbol{h}. \end{aligned} \tag{22}$$

The optimal solution $\sum_{i=1}^{k-1}\mathbf{P}_i\boldsymbol{x}_i^{*,\prime} + \mathbf{P}_k\varsigma_k^*$ to this problem is equivalent to the one of (18). Indeed, these two solutions have the same effect on task k

$$\mathbf{J}_k\sum_{i=1}^{k}\mathbf{P}_i\boldsymbol{x}_i^{*,\prime} = \mathbf{J}_k\left(\sum_{i=1}^{k-1}\mathbf{P}_i\boldsymbol{x}_i^{*,\prime} + \mathbf{P}_k\varsigma_k^*\right). \tag{23}$$

To prove (23), one needs to notice that $\mathbf{J}_i\mathbf{P}_j = \mathbf{0}$ with $j \geq i$. The second sub-problem is given by

$$\underset{\boldsymbol{x}_{k+1}}{\arg\min}\left\|\mathbf{J}_{k+1}\boldsymbol{x}'_{k+1} - \boldsymbol{x}_{k+1}^d\right\|^2. \tag{24}$$

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 5, 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 $\mathbf{J}_i\mathbf{P}_j = \mathbf{0}$ with $j \geq i$, the final solution is $\sum_{i=1}^{k}\mathbf{P}_i\boldsymbol{x}_i^* + \mathbf{P}_{k+1}\boldsymbol{x}_{k+1}^*$. 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 $S = \sum_{i=1}^{k}\mathbf{P}_i^i\boldsymbol{x}_i^{i,*} + \sum_{i=1}^{k}\mathbf{P}_i^f\boldsymbol{x}_i^{f,*} + \mathbf{P}_{k+1}\boldsymbol{x}_{k+1}^*$, with $\mathbf{P}_i^i$ and $\mathbf{P}_i^f$ the columns in $\mathbf{P}_i$ that correspond to $\boldsymbol{x}_i^{i,*}$ and $\boldsymbol{x}_i^{f,*}$ respectively. The term $\sum_{i=1}^{k}\mathbf{P}_i^f\boldsymbol{x}_i^{f,*}$ 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 $\sum_{i=1}^{k}\mathbf{P}_i^i\boldsymbol{x}_i^{i,*} + \mathbf{P}_{k+1}\boldsymbol{x}_{k+1}^*$, where $\mathbf{P}_i^i\boldsymbol{x}_i^{i,*}$ are used to perform the k previous tasks and $\mathbf{P}_{k+1}\boldsymbol{x}_{k+1}^*$ is used to perform the $(k+1)$-th task in the null-space of all the higher priority tasks.

Acknowledgements We would like to thank the reviewers for their insightful comments on the paper. 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. Abe Y, da Silva M, Popović J (2007) Multiobjective control with frictional contacts. In: Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation, pp 249–258
2. Aghili F (2005) 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 21(5):834–849, DOI 10.1109/TRO.2005.851380
3. Andersson J, kesson J, Diehl M (2012) Casadi - a symbolic package for automatic differentiation and optimal control. Recent Advances in Algorithmic Differentiation
4. Baerlocher P, Boulic R (1998) Task-priority formulations for the kinematic control of highly redundant articulated structures. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, vol 1, pp 323–329 vol.1, DOI 10.1109/IROS.1998.724639
5. Bouyarmane K, Kheddar A (2011) Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configurations. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 4414–4419, DOI 10.1109/IROS.2011.6094483
6. Chang KS, Khatib O (1999) Efficient algorithm for extended operational space inertia matrix. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, vol 1, pp 350–355
7. Collette C, Micaelli A, Andriot C, Lemerle P (2007) Dynamic balance control of humanoids for multiple grasps and non coplanar frictional contacts. In: 7th IEEE-RAS International Conference on Humanoid Robots, pp 81–88
8. Dietrich A, Albu-Schaffer A, Hirzinger G (2012) On continuous null space projections for torque-based, hierarchical, multi-objective manipulation. In: IEEE International Conference on Robotics and Automation (ICRA), pp 2978–2985, DOI 10.1109/ICRA.2012.6224571
9. E-C K, J-M M (2000) Soft constraints and exact penalty functions in model predictive control. In: Proceedings of the UKACC International Conference, Cambridge, UK
10. Egeland O, Sagli J, Spangelo I, Chiaverini S (1991) A damped least-squares solution to redundancy resolution. In: IEEE International Conference on Robotics and Automation, pp 945–950 vol.1, DOI 10.1109/ROBOT.1991.131710
11. Escande A, Mansard N, Wieber PB (2013) Hierarchical quadratic programming: Companion report. Tech. rep., URL http://hal.archives-ouvertes.fr/hal-00970816
12. Escande A, Mansard N, Wieber PB (2014) Hierarchical quadratic programming: Fast online humanoid-robot motion generation. The International Journal of Robotics Research DOI 10.1177/0278364914521306
13. Flacco F, De Luca A, Khatib O (2012) Motion control of redundant robots under joint constraints: Saturation in the null space. In: IEEE International Conference on Robotics and Automation (ICRA), pp 285–292, DOI 10.1109/ICRA.2012.6225376
14. Flacco F, De Luca A, Khatib O (2012) Prioritized multi-task motion control of redundant robots under hard joint constraints. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 3970–3977, DOI 10.1109/IROS.2012.6385619
15. Hsu P, Mauser J, Sastry S (1989) Dynamic control of redundant manipulators. Journal of Robotic Systems 6(2):133–148, DOI 10.1002/rob.4620060203, URL

http://dx.doi.org/10.1002/rob.4620060203
16. Jarquin G, Escande A, Arechavaleta G, Moulard T, Yoshida E, Parra-Vega V (2013) Real-time smooth task transitions for hierarchical inverse kinematics. In: 13th IEEE-RAS International Conference on Humanoid Robots, pp 528–533, DOI 10.1109/HUMANOIDS.2013.7030024
17. Kanoun O, Lamiraux F, Wieber PB, Kanehiro F, Yoshida E, Laumond JP (2009) Prioritizing linear equality and inequality systems: Application to local motion planning for redundant robots. In: IEEE International Conference on Robotics and Automation (2009), pp 2939–2944
18. Kanoun O, Lamiraux F, Wieber PB (2011) Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task. IEEE Transactions on Robotics 27(4):785–792
19. Keith F, Wieber N P-Band Mansard, Kheddar A (2011) 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), pp 3887–3892, DOI 10.1109/IROS.2011.6094706
20. Khatib O (1986) Real-time obstacle avoidance for manipulators and mobile robots. The international journal of robotics research 5(1):90–98
21. Khatib O (1987) A unified approach for motion and force control of robot manipulators: The operational space formulation. IEEE Journal of Robotics and Automation 3(1):43–53
22. Khatib O (1995) Inertial properties in robotic manipulation: An object-level framework. The International Journal of Robotics Research 14(1):19–36
23. Khatib O, Sentis L, Park JH (2008) A unified framework for whole-body humanoid robot control with multiple constraints and contacts. In: European Robotics Symposium 2008, Springer Tracts in Advanced Robotics, vol 44, Springer Berlin / Heidelberg, pp 303–312
24. Lee J, Mansard N, Park J (2012) Intermediate desired value approach for task transition of robots in kinematic control. IEEE Transactions on Robotics 28(6):1260–1277, DOI 10.1109/TRO.2012.2210293
25. Liégeois A (1977) Automatic supervisory control of the configuration and behavior of multibody mechanisms. IEEE Transactions on Systems, Man and Cybernetics 7(12):868–871
26. Liu M, Micaelli A, Evrard P, Escande A, Andriot C (2011) Interactive dynamics and balance of a virtual character during manipulation tasks. In: IEEE International Conference on Robotics and Automation (ICRA), pp 1676–1682
27. Liu M, Micaelli A, Evrard P, Escande A, Andriot C (2012) Interactive virtual humans: A two-level prioritized control framework with wrench bounds. IEEE Transactions on Robotics 28(6):1309–1322, DOI 10.1109/TRO.2012.2208829
28. Mansard N, Khatib O, Kheddar A (2009) A unified approach to integrate unilateral constraints in the stack of tasks. IEEE Transactions on Robotics 25(3):670–685
29. Mansard N, Remazeilles A, Chaumette F (2009) Continuity of varying-feature-set control laws. IEEE Transactions on Automatic Control 54(11):2493–2505, DOI 10.1109/TAC.2009.2031202
30. Mistry M, Nakanishi J, Schaal S (2007) Task space control with prioritization for balance and locomotion. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 331–338, DOI 10.1109/IROS.2007.4399595
31. Mistry M, Nakanishi J, Cheng G, Schaal S (2008) Inverse kinematics with floating base and constraints for full body humanoid robot control. In: 8th IEEE-RAS International Conference on Humanoid Robots, pp 22–27, DOI 10.1109/ICHR.2008.4755926
32. Padois V, Fourquet JY, Chiron P, et al (2007) Kinematic and dynamic model-based control of wheeled mobile manipulators: A unified framework for reactive approaches. Robotica 25(2):157
33. Peters J, Mistry M, Udwadia F, Nakanishi J, Schaal S (2008) A unifying framework for robot control with redundant dofs. Autonomous Robots 24(1):1–12, DOI 10.1007/s10514-007-9051-x, URL http://dx.doi.org/10.1007/s10514-007-9051-x
34. Petrič T, Žlajpah L (2013) Smooth continuous transition between tasks on a kinematic control level: Obstacle avoidance as a control problem. Robotics and Autonomous Systems 61(9):948–959, DOI http://dx.doi.org/10.1016/j.robot.2013.04.019
35. Saab L, Soueres P, Fourquet JY (2009) Coupling manipulation and locomotion tasks for a humanoid robot. In: International Conference on Advances in Computational Tools for Engineering Applications (ACTEA), pp 84–89
36. Saab L, Mansard N, Keith F, Fourquet JY, Soueres P (2011) Generation of dynamic motion for anthropomorphic systems under prioritized equality and inequality constraints. In: IEEE International Conference on Robotics and Automation (ICRA), pp 1091–1096
37. Saab L, Ramos O, Keith F, Mansard N, Soueres P, Fourquet JY (2013) Dynamic whole-body motion generation under rigid contacts and other unilateral constraints. IEEE Transactions on Robotics 29(2):346–362, DOI 10.1109/TRO.2012.2234351
38. Sadeghian H, Villani L, Keshmiri M, Siciliano B (2013) Dynamic multi-priority control in redundant robotic systems. Robotica 31:1155–1167, DOI 10.1017/S0263574713000416
39. Salini J (2013) Arboris-Python: A rigid body dynamics and contacts simulator written in python. https://github.com/salini/arboris-python
40. Salini J, Padois V, Bidaud P (2011) Synthesis of complex humanoid whole-body behavior: A focus on sequencing and tasks transitions. In: IEEE International Conference on Robotics and Automation (ICRA), pp 1283 –1290, DOI 10.1109/ICRA.2011.5980202
41. Sentis L, Khatib O (2004) Prioritized multi-objective dynamics and control of robots in human environments. In: 4th IEEE/RAS International Conference on Humanoid Robots, vol 2, pp 764–780 Vol. 2, DOI 10.1109/ICHR.2004.1442684
42. Sentis L, Khatib O (2004) Task-oriented control of humanoid robots through prioritization. In: IEEE RAS/RSJ International Conference on Humanoid Robots
43. Sentis L, Khatib O (2005) Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. International Journal of Humanoid Robotics 02(04):505–518, DOI 10.1142/S0219843605000594
44. Sentis L, Park J, Khatib O (2010) Compliant control of multi-contact and center of mass behaviors in humanoid robots. IEEE Transactions on Robotics 26(3):483–501, DOI 10.1109/TRO.2010.2043757
45. Siciliano B, Slotine JJ (1991) A general framework for managing multiple tasks in highly redundant robotic systems. In: Fifth International Conference on Advanced Robotics, pp 1211 –1216 vol.2, DOI

10.1109/ICAR.1991.240390
46. Stasse O, Escande A, Mansard N, Miossec S, Evrard P, Kheddar A (2008) Real-time (self)-collision avoidance task on a hrp-2 humanoid robot. In: IEEE International Conference on Robotics and Automation (ICRA), pp 3200–3205

Mingxing Liu is a post-doctoral fellow at the Institut des Systèmes Intelligents et de Robotique (ISIR, UMR CNRS 7222) at Université Pierre et Marie Curie (UPMC), Paris, France. In 2009, she received both her engineering degree from Ecole Centrale Paris, France and her master's degree in Automatic Control from Shanghai Jiao Tong University, China. From 2009 to 2012, she is a PhD student at the Systems and Technologies Integration Laboratory of the French Atomic Energy Commission (CEA-LIST). She received the Ph.D. degree in Robotics in 2013 from UPMC. Her research interests include automatic control, whole-body control for redundant robots such as industrial manipulators and humanoid robots, as well as human-robot interaction.

Yang Tan received the M.S. degree in Mechanical Engineering from Tianjin University, Tianjin, China in June 2012. He is pursuing the Ph.D. degree at the Institute of Intelligent Systems and Robotics (ISIR, UMR CNRS 7222), University of Pierre and Marie Curie, Paris, France. His current research interests include robotics and control.

Vincent Padois is an associated professor of Robotics and Computer Science and a member of the Institut des Systèmes Intelligents et de Robotique (ISIR, UMR CNRS 7222) at Université Pierre et Marie Curie in Paris, France. In 2001, he receives both an engineering degree from the Ecole Nationale d'Ingénieurs de Tarbes (ENIT), France and his master's degree in Automatic Control from the Institut National Polytechnique de Toulouse (INPT), France. From 2001 to 2005, he is a PhD student in Robotics of the ENIT/INPT Laboratoire Genie de Production. In 2006 and 2007, he is a post-doctoral fellow in the Stanford Artificial Intelligence Laboratory and more specifically in the group of Professor O. Khatib. Since 2007, his research activities at ISIR are mainly focused on the automatic design, the modeling and the control of redundant and complex systems such as wheeled mobile manipulators, humanoid robots as well as standard manipulators evolving under constraints in complex environments. He is also involved in research activities that aim at bridging the gap between adaptation and decision making techniques provided by Artificial Intelligence and low-level, reactive control. Since 2011, he holds the "Intervention Robotics" RTE/UPMC chair position.