Hierarchical Inequality Task Specification for Indirect Force Controlled Robots using Quadratic Programming

Ewald Lutscher and Gordon Cheng
Institute for Cognitive Systems (`www.ics.ei.tum.de`)
Technische Universität München

Abstract— **In our previous work we derived a task specification approach for indirect force controlled robots to assign force and positioning tasks in joint and Cartesian space and execute them simultaneously in a hierarchical way. The virtual set points for an underlying joint space indirect force controller have been computed according to the specified tasks, supporting reactive control by generating virtual velocity commands.**

In the present work, the virtual set point generation is extended to inequality tasks by reformulating the problem as a quadratic program. The resulting control layer does not modify the underlying indirect force controller, hence the inherent compliance of the manipulator is preserved.

The new approach is experimentally verified on a 7 degree of freedom manipulator.

I. Motivation

When operating in unstructured environment, compliance is an important requirement for a robotic manipulator. A stable and robust approach to realize compliance is provided by indirect force controllers (*IFC*), where the motion and interaction forces of the physical robot are indirectly controlled by assigning set points to a virtual robot which on his part is coupled with the real robot via a virtual mechanical relationship. This control scheme has also nice stability properties which are independent from the environmental dynamics as long as it is passive. Due to these advantages, IFCs are provided often as the only force control interface for robots forcing application programmers to use this closed architecture.

Even though this control scheme has been under development for decades ([1], [2]), the contributions dealing with IFC set point selection to achieve desired interaction forces and positioning of the robot are very sparse. Conventional trajectory planning approaches are often applied and the IFC is used to compensate for contact uncertainty and unexpected collisions.

In [3], we presented a task specification layer to regulate the positioning and static interaction forces on joint and Cartesian level for IFC controlled robots based on hierarchical nullspace projections. The general task variable, which could either be specified as a Cartesian pose, wrench, joint position or joint torque, could be regulated to a certain desired value. Assigning this desired value to a task variable can also be interpreted as an equality constraint. However, besides avoiding joint limits, there was no way to specify

This work is supported in part within the DFG excellence initiative research cluster Cognition for Technical Systems - CoTeSys (`www.cotesys.org`).

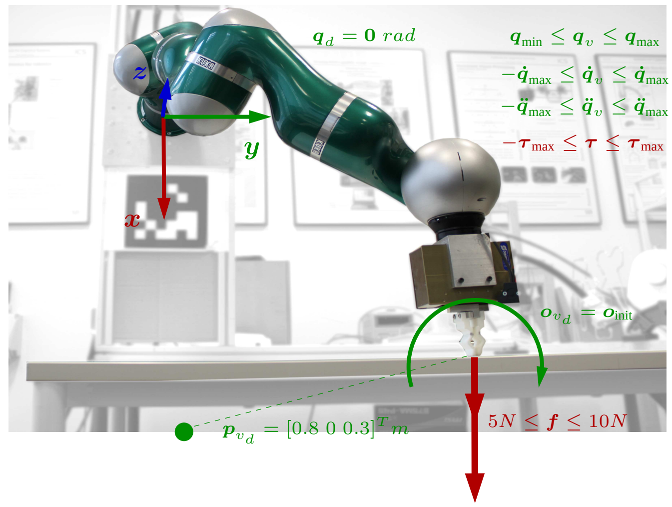

Fig. 1. Experimental setup with root coordinate frame. The task is to reach the point p_{v_d} with constant orientation o_{v_d} while maintaining a contact force between 5N and 10N and keeping the joints as far away from their mechanical limits as possible at the lowest priority level. The inequality tasks with the highest priority are to keep the joints inside their limits and obey the joint velocity, acceleration and torque constraints.

inequality constraints or tasks like for example limiting the applied joint torques/velocities or restricting the end-effector to a certain workspace area. In this paper, we extend the approach from [3] to general inequality tasks.

There exist a vast number of resolving inequality constraints on joint level, e.g. [4], [5], where most of them treat only joint angle limits. Flacco et. al. introduced an algorithm to incorporate joint angle, velocity and acceleration limits and exploit them as good as possible to achieve a Cartesian task by scaling it appropriately [6]. Specific inequality constraints, like collision/singularity avoidance, have been treated in the past via the gradient projection method [7]. A unified but computational expensive approach is presented in [8] where general inequality tasks are treated on every priority level in a stack-of-tasks framework. In recent contributions quadratic programming (*QP*) methods are used to find an optimal solution for the inverse kinematic problem with a given task hierarchy ([9], [10]). The main advantage of the QP approach is that it provides a simple and general formalization of the inverse kinematic problem with inequality constraints. Most of these schemes are defined on the kinematic or force level and to our best knowledge there is no application in the context of indirect force control.

The remainder of this paper is structured as follows. In

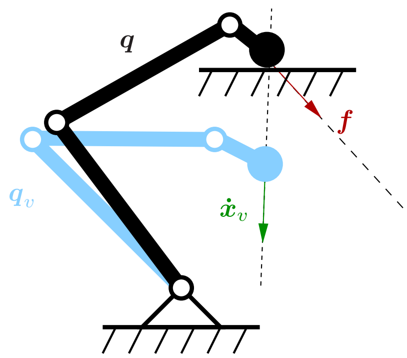

Fig. 2. Motion and interaction forces of the physical manipulator (black) are controlled indirectly by generating set points for the virtual manipulator (blue).

Sec. II some theoretic background and our previous work is recapitulated. The extension to inequality tasks is derived in Sec. III and the experimental results can be found in Sec. IV. Sec. V concludes our work.

II. Theoretical Background

In this section we cover some fundamental theory and basic principles on which our work is based. In Sec. II-A the general concept of indirect force control is briefly summarized, Sec. II-B is a recapitulation of our previous work on task specification for IFC controlled robots and in Sec. II-C the quadratic programming paradigm is treated.

A. Indirect Force Control

An interpretation of IFC schemes is that the motion and interaction forces of a n degree of freedom physical manipulator are controlled indirectly by assigning a joint position set point $\boldsymbol{q}_v \in \mathbb{R}^n$ to a kinematically equivalent virtual manipulator. The applied joint torques are derived from a virtual mechanical relationship, (e.g mechanical impedance, stiffness) between the virtual and the physical manipulator. With $\boldsymbol{q} \in \mathbb{R}^n$ denoting the joint position of the physical manipulator, the positioning difference between the virtual and physical manipulator is related to the static interaction torque $\boldsymbol{\tau} \in \mathbb{R}^n$ via a positive definite $n \times n$ stiffness matrix $\boldsymbol{K}$:

$$\boldsymbol{\tau} = \boldsymbol{K}(\boldsymbol{q_v} - \boldsymbol{q}) \tag{1}$$

The dynamic components of the IFC are neglected, since they firstly, play only a minor role when moving with comparatively low speed and secondly, we can not control them directly without knowledge about the environment. For the continuous case, the desired velocity $\dot{\boldsymbol{q}}_v$ is used to regulate $\boldsymbol{q}_v$. Fig. 2 depicts the basic principle of an IFC.

B. General Task Specification for Indirect Force Controlled Robots

In [3] we generalized force and positioning tasks using a uniform task variable $\boldsymbol{\sigma} \in \mathbb{R}^m$ with the desired value $\boldsymbol{\sigma}_d$. The task could be any quantity which is related to $\boldsymbol{q}_v$ with a $m \times n$ task Jacobian

$$\boldsymbol{A} = \frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{q}_v},$$

TABLE I
Specifications of the four basic task types

type	$\boldsymbol{\sigma}$	$\boldsymbol{A}$
cart. pose	$\boldsymbol{x}_v$	$\boldsymbol{J}_v$
joint position	$\boldsymbol{q}_v$	$\boldsymbol{I}_n$
wrench	$\boldsymbol{h}$	$\boldsymbol{J}^{T+}\boldsymbol{K}$
joint torque	$\boldsymbol{\tau}$	$\boldsymbol{K}$

which is the linearized relation between $\boldsymbol{\sigma}$ and $\boldsymbol{q}_v$, so that

$$\dot{\boldsymbol{\sigma}} = \boldsymbol{A}\dot{\boldsymbol{q}}_v \tag{2}$$

We derived $\boldsymbol{A}$ for the four basic task types

- virtual joint position $\boldsymbol{q}_v$
- virtual Cartesian end effector pose $\boldsymbol{x}_v$
- static joint torque $\boldsymbol{\tau}$
- static end effector wrench $\boldsymbol{h}$,

which can be found in table I, where $\boldsymbol{J}(\boldsymbol{q}) = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{q}}$ denotes the physical manipulators base Jacobian and $\boldsymbol{J}_v = \boldsymbol{J}(\boldsymbol{q}_v)$ the virtual manipulators base Jacobian. Notice again that we consider only the static interaction torques and forces, due to the stiffness relation (1) and regulate only the position of the virtual manipulator. See [3] for details on this. The basic tasks from table I can also be expressed in a certain subspace $\mathbb{S} \subseteq \mathbb{R}^m$ which enlarges the manipulators nullspace with respect to that task. This subspace is characterized by a set of orthonormal vectors, which are the columns of a matrix $\boldsymbol{S}$. The task Jacobian $\boldsymbol{A}$ has to be modified according to

$$\hat{\boldsymbol{A}} = \boldsymbol{S}^T\boldsymbol{A},$$

which is $\boldsymbol{A}$ expressed in $\mathbb{S}$. The $\hat{}$ will be dropped in the rest of the paper for the sake of better readability.

The classical approach for task level control was imposed

$$\dot{\boldsymbol{\sigma}}_d = \boldsymbol{\Lambda}(\boldsymbol{\sigma}_d - \boldsymbol{\sigma}),$$

where $\boldsymbol{\Lambda}$ is usually a diagonal, positive definite $m \times m$ gain matrix that tunes the convergence speed of the task error components to $\boldsymbol{0}$. With (2), the equality task

$$\dot{\boldsymbol{\sigma}}_d = \boldsymbol{A}\dot{\boldsymbol{q}}_v$$

was stated and a hierarchical controller was derived using nullspace projection methods to enforce a strict task hierarchy among a set of k subtasks $[\boldsymbol{\sigma}_1 \dots \boldsymbol{\sigma}_k]$.

C. Quadratic Programming Problem Formulation

The classic QP problem statement is to find a vector $\boldsymbol{x}$, that minimizes a quadratic cost function, subject to linear equality and inequality constraints:

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{a}^T\boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{E}\boldsymbol{x} = \boldsymbol{d}, \end{aligned}$$

where $\boldsymbol{H}, \boldsymbol{C}$ and $\boldsymbol{E}$ are matrices and $\boldsymbol{a}, \boldsymbol{b}$ and $\boldsymbol{d}$ are vectors of appropriate size.

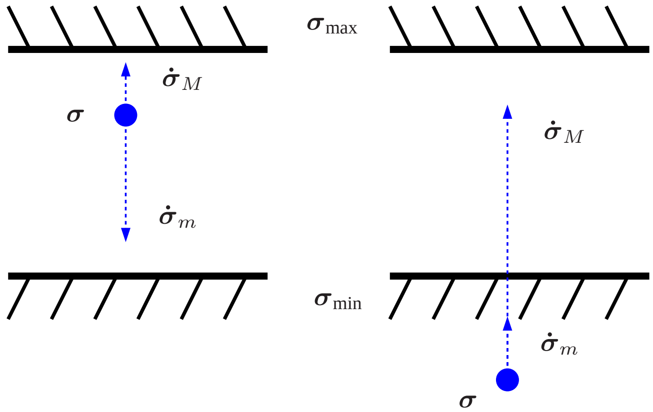

Fig. 3. To keep the task variable $\boldsymbol{\sigma}$ inside its bounds $[\boldsymbol{\sigma}_{\min},\ \boldsymbol{\sigma}_{\max}]$, the velocity $\dot{\boldsymbol{\sigma}}$ is limited to the range $[\dot{\boldsymbol{\sigma}}_m,\ \dot{\boldsymbol{\sigma}}_M]$.

In robot control, we are usually imposing lower and upper bounds on the task variables. Therefor it is easier to state the inequality constraints as box constraints

$$\boldsymbol{b}_m \leq \boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{b}_M,$$

where $\boldsymbol{b}_m$ is the lower and $\boldsymbol{b}_M$ the upper bound on $\boldsymbol{C}\boldsymbol{x}$. By setting $\boldsymbol{b}_m = \boldsymbol{b}_M$, one can also capture equality constraints with this formulation. Also, most QP solvers take lower and upper bounds ($\boldsymbol{x}_m$ and $\boldsymbol{x}_M$) on $\boldsymbol{x}$ directly instead of having to formulate them as inequality constraints. With this, we can state the equivalent problem

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{a}^T\boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{b}_m \leq \boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{b}_M \\ & \boldsymbol{x}_m \leq \boldsymbol{x} \leq \boldsymbol{x}_M \end{aligned} \tag{3}$$

This corresponds also to the format, which is accepted by the QP solver we have used in our implementation [11].

III. Inequality Task Specification

Instead of having one desired value for the task variable $\boldsymbol{\sigma}_d$, we specify lower and upper bounds ($\boldsymbol{\sigma}_{\min}$ and $\boldsymbol{\sigma}_{\max}$) as a desired range for $\boldsymbol{\sigma}$ what can be defined as an inequality task

$$\boldsymbol{\sigma}_{\min} \leq \boldsymbol{\sigma} \leq \boldsymbol{\sigma}_{\max}.$$

As stated in II-C, if we have a certain desired value or equality task $\boldsymbol{\sigma}_d$, we set $\boldsymbol{\sigma}_{\min} = \boldsymbol{\sigma}_{\max} = \boldsymbol{\sigma}_d$.

To keep $\boldsymbol{\sigma}$ inside, respectively make it converge into the range $[\boldsymbol{\sigma}_{\min},\ \boldsymbol{\sigma}_{\max}]$, we apply the method proposed originally in [12] for obstacle avoidance by inducing lower and upper bounds on $\dot{\boldsymbol{\sigma}}$ denoted by

$$\dot{\boldsymbol{\sigma}}_m = \boldsymbol{\Lambda}(\boldsymbol{\sigma}_{\min} - \boldsymbol{\sigma}) \tag{4}$$

$$\dot{\boldsymbol{\sigma}}_M = \boldsymbol{\Lambda}(\boldsymbol{\sigma}_{\max} - \boldsymbol{\sigma}) \tag{5}$$

depending on the distance of $\boldsymbol{\sigma}$ to $\boldsymbol{\sigma}_{\min}$ respectively $\boldsymbol{\sigma}_{\max}$ and the convergence rate $\boldsymbol{\Lambda}$. Figure 3 depicts this approach.

A. QP Problem for one Inequality Task

Suppose we have one inequality task denoted by $\boldsymbol{\sigma}_{0_{\min}}$, $\boldsymbol{\sigma}_{0_{\max}}$, convergence rate $\boldsymbol{\Lambda}_0$ and task Jacobian $\boldsymbol{A}_0 = \frac{\partial \boldsymbol{\sigma}_0}{\partial \boldsymbol{q}_v}$. The lower and upper bounds $\dot{\boldsymbol{\sigma}}_{0_m}$ and $\dot{\boldsymbol{\sigma}}_{0_M}$ for $\dot{\boldsymbol{\sigma}}$ are computed with (4) and (5). Also assume the control input $\dot{\boldsymbol{q}}_v$ is restricted to certain velocity limits $\dot{\boldsymbol{q}}_{v_m}$ and $\dot{\boldsymbol{q}}_{v_M}$. A possible QP problem to find a proper $\dot{\boldsymbol{q}}_v$, which corresponds to (3) could be stated as

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}\boldsymbol{s}^T\boldsymbol{s} \\ \text{s.t.} \quad & \dot{\boldsymbol{\sigma}}_{0_m} \ \leq\ \boldsymbol{A}_0\dot{\boldsymbol{q}}_v - \boldsymbol{s} \ \leq\ \dot{\boldsymbol{\sigma}}_{0_M} \\ & \dot{\boldsymbol{q}}_{v_m} \ \leq\ \dot{\boldsymbol{q}}_v \ \leq\ \dot{\boldsymbol{q}}_{v_M} \end{aligned}$$

with $\boldsymbol{s} \in \mathbb{R}^m$ as a vector of slack variables, which allow violations of the task inequality constraints in case the tasks are unfeasible.

As discussed in [9], the optimization problem can become ill conditioned if some task becomes infeasible with respect to higher priority ones. This can be overcome by balancing the cost of the slack variable with the norm of the resulting solution $\dot{\boldsymbol{q}}_v$ by minimizing

$$\frac{1}{2}\boldsymbol{s}^T\boldsymbol{s} + \frac{1}{2}\rho\dot{\boldsymbol{q}}_v^T\dot{\boldsymbol{q}}_v$$

instead, where $\rho \in \mathbb{R}^+$ is a regularization factor which has to be tuned manually.

To comply with (3) we define a new optimization variable $\boldsymbol{w} = [\dot{\boldsymbol{q}}_v \quad \boldsymbol{s}]^T$ and formulate the according QP problem

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{H}\boldsymbol{w} \\ \text{s.t.} \quad & \dot{\boldsymbol{\sigma}}_{0_m} \ \leq\ [\boldsymbol{A}_0 \quad -\boldsymbol{I}_m]\boldsymbol{w} \ \leq\ \dot{\boldsymbol{\sigma}}_{0_M} \\ & \boldsymbol{w}_m \ \leq\ \boldsymbol{w} \ \leq\ \boldsymbol{w}_M \end{aligned}$$

where $\boldsymbol{I}_m$ is the $m \times m$ identity matrix and

$$\boldsymbol{H} = \begin{bmatrix} \rho\boldsymbol{I}_n & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_m \end{bmatrix}$$

$$\boldsymbol{w}_m = \begin{bmatrix} \dot{\boldsymbol{q}}_{v_m} \\ -\infty \end{bmatrix}$$

$$\boldsymbol{w}_M = \begin{bmatrix} \dot{\boldsymbol{q}}_{v_M} \\ \infty \end{bmatrix}.$$

By solving this problem, we obtain the optimal solution

$$\boldsymbol{w}_0^* = \begin{bmatrix} \dot{\boldsymbol{q}}_{v_0}^* \\ \boldsymbol{s}_0^* \end{bmatrix}.$$

B. QP Problem for a Second Inequality Task

Suppose we have a second task, which should be executed as good as possible without violating the first one. The QP problem for this task is similar to the first one, besides that we have to add another inequality, assuring that the first task is not altered. This inequality is a hard constraint which is not relaxed by the slack variable.

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{H}\boldsymbol{w} \\ \text{s.t.} \quad & \dot{\boldsymbol{\sigma}}_{1_m} \ \leq\ [\boldsymbol{A}_1 \quad -\boldsymbol{I}_m]\boldsymbol{w} \ \leq\ \dot{\boldsymbol{\sigma}}_{1_M} \\ & \dot{\boldsymbol{\sigma}}_{0_m} + \boldsymbol{s}_0^* \ \leq\ \boldsymbol{A}_0\dot{\boldsymbol{q}}_v \ \leq\ \dot{\boldsymbol{\sigma}}_{0_M} + \boldsymbol{s}_0^* \\ & \boldsymbol{w}_m \ \leq\ \boldsymbol{w} \ \leq\ \boldsymbol{w}_M \end{aligned}$$

Again, this problem can be reformulated to comply with (3):

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{H}\boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{b}_{1_m} \ \leq \ \boldsymbol{C}_1\boldsymbol{w} \ \leq \ \boldsymbol{b}_{1_M} \\ & \boldsymbol{w}_m \ \leq \ \boldsymbol{w} \ \leq \ \boldsymbol{w}_M \end{aligned}$$

with

$$\begin{aligned} \boldsymbol{b}_{1_m} &= \begin{bmatrix} \dot{\boldsymbol{\sigma}}_{m_1} \\ \dot{\boldsymbol{\sigma}}_{m_0} + \boldsymbol{s}_0^* \end{bmatrix} \\ \boldsymbol{b}_{1_M} &= \begin{bmatrix} \dot{\boldsymbol{\sigma}}_{M_1} \\ \dot{\boldsymbol{\sigma}}_{M_0} + \boldsymbol{s}_0^* \end{bmatrix} \\ \boldsymbol{C}_1 &= \begin{bmatrix} \boldsymbol{A}_1 & -\boldsymbol{I}_m \\ \boldsymbol{A}_0 & \boldsymbol{0} \end{bmatrix}. \end{aligned}$$

C. General Recursive QP Problem Formulation

Finally, with the initial values

$$\begin{aligned} \bar{\boldsymbol{A}}_0 &= \boldsymbol{0}_{1\times n} \\ \bar{\boldsymbol{b}}_{m_0} &= 0 \\ \bar{\boldsymbol{b}}_{M_0} &= 0 \end{aligned}$$

we can state a recursive formulation for an arbitrary set of k subtasks. For $i = 1 \dots k$

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{H}\boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{b}_{i_m} \ \leq \ \boldsymbol{C}_i\boldsymbol{w} \ \leq \ \boldsymbol{b}_{i_M} \\ & \boldsymbol{w}_m \ \leq \ \boldsymbol{w} \ \leq \ \boldsymbol{w}_M \end{aligned}$$

with

$$\begin{aligned} \boldsymbol{b}_{i_m} &= \begin{bmatrix} \dot{\boldsymbol{\sigma}}_{m_i} \\ \bar{\boldsymbol{b}}_{m_{i-1}} \end{bmatrix} \\ \boldsymbol{b}_{i_M} &= \begin{bmatrix} \dot{\boldsymbol{\sigma}}_{M_i} \\ \bar{\boldsymbol{b}}_{M_{i-1}} \end{bmatrix} \\ \boldsymbol{C}_i &= \begin{bmatrix} \boldsymbol{A}_i & -\boldsymbol{I}_m \\ \bar{\boldsymbol{A}}_{i-1} & \boldsymbol{0} \end{bmatrix} \end{aligned}$$

For the next iteration

$$\begin{aligned} \bar{\boldsymbol{b}}_{i_m} &= \begin{bmatrix} \dot{\boldsymbol{\sigma}}_{m_{i-1}} + \boldsymbol{s}_{i-1}^* \\ \bar{\boldsymbol{b}}_{m_{i-1}} \end{bmatrix} \\ \bar{\boldsymbol{b}}_{i_M} &= \begin{bmatrix} \dot{\boldsymbol{\sigma}}_{M_{i-1}} + \boldsymbol{s}_{i-1}^* \\ \bar{\boldsymbol{b}}_{M_{i-1}} \end{bmatrix} \\ \bar{\boldsymbol{A}}_i &= \begin{bmatrix} \boldsymbol{A}_i \\ \bar{\boldsymbol{A}}_{i-1} \end{bmatrix} \end{aligned}$$

Apparently, having many equality tasks could result in a significant increase in the corresponding slack variables, especially during ill-conditioned cases. However, proper task specification which avoids obviously contradicting tasks should prevent these cases.

D. Capturing Joint Space Limits

The joints of every physical manipulator are usually restricted to certain constraints regarding their angle, velocity, acceleration and torque so that

$$\begin{aligned} \boldsymbol{q}_{\text{min}} \leq \boldsymbol{q}_v \leq \boldsymbol{q}_{\text{max}} \\ -\boldsymbol{v}_{\text{max}} \leq \dot{\boldsymbol{q}}_v \leq \boldsymbol{v}_{\text{max}} \\ -\boldsymbol{a}_{\text{max}} \leq \ddot{\boldsymbol{q}}_v \leq \boldsymbol{a}_{\text{max}} \\ -\boldsymbol{\tau}_{\text{max}} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{\text{max}}. \end{aligned}$$

A conventional approach is to use finite differences, shaping the joint velocity bounds to keep joint position, velocity and acceleration constraints. We adapt this method to add static torque constraints by expressing them as additional joint limits. Using the static relation (1) we can state

$$\begin{aligned} \boldsymbol{\tau}_{\text{max}} &= \boldsymbol{K}(\boldsymbol{q}_{\boldsymbol{v}_{\text{max}}} - \boldsymbol{q}) \\ -\boldsymbol{\tau}_{\text{max}} &= \boldsymbol{K}(\boldsymbol{q}_{\boldsymbol{v}_{\text{min}}} - \boldsymbol{q}) \end{aligned}$$

and solve for the joint limits due to maximum torque:

$$\begin{aligned} \boldsymbol{q}_{\boldsymbol{v}_{\text{max}}} &= \boldsymbol{K}^{-1}\boldsymbol{\tau}_{\text{max}} + \boldsymbol{q} \\ \boldsymbol{q}_{\boldsymbol{v}_{\text{min}}} &= -\boldsymbol{K}^{-1}\boldsymbol{\tau}_{\text{max}} + \boldsymbol{q} \end{aligned}$$

With this the dynamic joint limits can be obtained with

$$\begin{aligned} \hat{\boldsymbol{q}}_{\text{min}} &= \max\{-\boldsymbol{K}^{-1}\boldsymbol{\tau}_{\text{max}} + \boldsymbol{q},\ \boldsymbol{q}_{\text{min}}\} \\ \hat{\boldsymbol{q}}_{\text{max}} &= \min\{\boldsymbol{K}^{-1}\boldsymbol{\tau}_{\text{max}} + \boldsymbol{q},\ \boldsymbol{q}_{\text{max}},\} \end{aligned}$$

where min$\{\bullet\}$ and max$\{\bullet\}$ is the component-wise minimum, respectively maximum of the input vectors. The velocity bounds, observing joint angle, velocity, acceleration and static torque limits are

$$\begin{aligned} \dot{\boldsymbol{q}}_{v_m} &= \max\{\tfrac{\hat{\boldsymbol{q}}_{\text{min}} - \boldsymbol{q}}{T},\ -\boldsymbol{v}_{\text{max}},\ -\sqrt{2\boldsymbol{a}_{\text{max}}(\boldsymbol{q} - \hat{\boldsymbol{q}}_{\text{min}})}\} \\ \dot{\boldsymbol{q}}_{v_M} &= \min\{\tfrac{\hat{\boldsymbol{q}}_{\text{max}} - \boldsymbol{q}}{T},\ \boldsymbol{v}_{\text{max}},\ \sqrt{2\boldsymbol{a}_{\text{max}}(\hat{\boldsymbol{q}}_{\text{max}} - \boldsymbol{q})},\} \end{aligned}$$

where T is the time interval of the discrete controller. See for example [6] for more details. These velocity bounds can be used to bound the optimization variable in the QP, serving as the highest priority joint-level safety bounds.

E. New Task Specification

Our previous task specification is extended by providing upper and lower bounds for the desired task variable. Hence a subtask is defined by

- the task type (or task Jacobian $\boldsymbol{A}$)
- lower and upper bounds $\boldsymbol{\sigma}_{\text{min}}$ and $\boldsymbol{\sigma}_{\text{max}}$
- convergence rate $\boldsymbol{\Lambda}$
- Subspace matrix $\boldsymbol{S}$

IV. Experimental Results

A. Implementation Details and Hardware

The experiments have been carried out on our KUKA LBR-IV lightweight arm. The manipulator was running a joint space impedance controller, which details can be found in [13]. The rate of the discrete controller was $500Hz$ and the stiffness $\boldsymbol{K} = 200\boldsymbol{I}_7 Nm/$rad. The task convergence factors $\boldsymbol{\Lambda}$ and the regularization factor $\rho = 0.01$ where chosen

TABLE II

SET OF SUBTASKS FOR CONSTRAINED POINT TO POINT MOTION

prio	type	$\boldsymbol{\sigma}_{\text{min}}$	$\boldsymbol{\sigma}_{\text{max}}$	$\boldsymbol{\Lambda}$	$\boldsymbol{S}$
1	wrench	$5N$	$10N$	100	$[1\ 0\ 0\ 0\ 0\ 0]^T$
2	cart. pose	$\boldsymbol{o}_{\text{init}}$	$\boldsymbol{o}_{\text{init}}$	$3\boldsymbol{I}_3$	$\begin{bmatrix} \boldsymbol{0}_3 \\ \boldsymbol{I}_3 \end{bmatrix}$
3	cart. pose	$\boldsymbol{p}_{v_d} = \begin{bmatrix} 0.8 \\ 0 \\ 0.3 \end{bmatrix} m$	$\boldsymbol{p}_{v_d} = \begin{bmatrix} 0.8 \\ 0 \\ 0.3 \end{bmatrix} m$	$3\boldsymbol{I}_3$	$\begin{bmatrix} \boldsymbol{I}_3 \\ \boldsymbol{0}_3 \end{bmatrix}$
4	joint position	$\boldsymbol{0}$	$\boldsymbol{0}$	$0.3\boldsymbol{I}_7$	$\boldsymbol{I}_7$

TABLE III

SET OF SUBTASKS FOR CUP HOLDING

prio	type	$\boldsymbol{\sigma}_{\text{min}}$	$\boldsymbol{\sigma}_{\text{max}}$	$\boldsymbol{\Lambda}$	$\boldsymbol{S}$
1	cart. pose	$\boldsymbol{o}_{\text{init}}$	$\boldsymbol{o}_{\text{init}}$	$3\boldsymbol{I}_3$	$\begin{bmatrix} \boldsymbol{0}_3 \\ \boldsymbol{I}_3 \end{bmatrix}$
2	cart. pose	$\boldsymbol{p}_{v_m} = \begin{bmatrix} 0.2 \\ 0.2 \\ 0.4 \end{bmatrix} m$	$\boldsymbol{p}_{v_M} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.7 \end{bmatrix} m$	$3\boldsymbol{I}_3$	$\begin{bmatrix} \boldsymbol{I}_3 \\ \boldsymbol{0}_3 \end{bmatrix}$
3	joint torque	$\boldsymbol{0}$	$\boldsymbol{0}$	$100\boldsymbol{I}_7$	$\boldsymbol{I}_7$

heuristically. The C++ QP library qpOASES [11] was used to carry out the optimization.

The task is to execute a point-to-point motion, where the goal is located inside an obstacle. The experimental setup is depicted in Fig. I. Highest priority is given to the joint level inequalities for joint range, velocity, acceleration and torques with the following symmetric limits

$$\begin{aligned} -120^\circ \leq q_{j_v} &\leq 120^\circ \\ -20^\circ \tfrac{1}{s} \leq \dot{q}_{j_v} &\leq 20^\circ \tfrac{1}{s} \\ -1000^\circ \tfrac{1}{s^2} \leq \ddot{q}_{j_v} &\leq 1000^\circ \tfrac{1}{s^2} \\ -15Nm \leq \tau_j &\leq 15Nm, \end{aligned}$$

with $j = 1 \ldots n$. For a second run the torque limits were lowered to

$$-5Nm \leq \tau_j \leq 5Nm$$

without changing the other parameters.

The rest of the task is specified in table II. The tasks are a force range in x-direction, keeping the orientation $\boldsymbol{o}_v \in \mathbb{R}^3$ of the end effector constant, bring the end effector position $\boldsymbol{p}_v \in \mathbb{R}^3$ to a desired point and keep the joints away from their limits.

Table III shows also the task specification for a constrained cup balancing task. The orientation of the end-effector is kept constant while minimizing the joint torques. In addition the end effector should not leave a certain box in the workspace defined by lower and upper bounds on $\boldsymbol{p}_v$. The video accompanying this paper shows the execution of the described tasks.

B. Results

Figure 4 shows the evolution of the different subtasks. While the highest joint level safety bounds are always obeyed, the other subtasks converge respecting their priority order. Figure 5 shows the effect of the regularization parameter ρ. As the force and the positioning tasks are conflicting, setting $\rho = 0$ results in unstable solutions. Removing the constraint on the x-component of the positioning decouples the conflicting subtasks and results in a stable solution.

An additional potential problem arises if force tasks are included and the robot is driven to singularity. Close to a singular configuration the applied wrench is not correctly computed due to the bad conditioning of the Jacobian. When designing the subtasks it should be avoided to drive the manipulator close to a singular configuration, which also could be added as an additional inequality task.

V. CONCLUSION

We enhanced our previous task specification approach for indirect force controlled robots to support hierarchical inequality tasks. Every subtask is formulated as a quadratic program with inequality constraints, which restrict the possible solutions to a set, which does not affect the higher priority tasks. Enabling inequality tasks enhances the descriptive power of the existing framework significantly as it is now easily possible to assign safety limits for certain tasks or relax task constraints by assigning a desired range instead of a desired value. These task relaxations on their part increase the solution space for lower priority tasks.

The proposed regulation approach does not require modification of the underlying indirect force controller, which makes this method suitable for closed control architectures, which are often provided for commercial robots.

REFERENCES

[1] J. Nevins and D. Whitney, *The Force Vector Assembler Concept*. Springer, 1972.

[2] J. Salisbury, "Active stiffness control of a manipulator in Cartesian coordinates," in *19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, 1980.

[3] E. Lutscher and G. Cheng, "A Practical Approach to Generalized Hierarchical Task Specification for Indirect Force Controlled Robots," in *IEEE/RSJ International Conference on Intelligent Robots and System (IROS)*, 2013.

[4] P. Baerlocher and R. Boulic, "An inverse kinematics architecture enforcing an arbitrary number of strict priority levels," *The Visual Computer*, vol. 20, no. 6, pp. 402–417, Jun. 2004.

[5] T. Chan and R. Dubey, "A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 2, pp. 286–292, 1995.

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

[7] A. Liegeois, "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, 1977.

[8] 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, Jun. 2009.

[9] 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.

[10] A. Escande, N. Mansard, and P. Wieber, "Fast Resolution of Hierarchized Inverse Kinematics with Inequality Constraints," in *Robotics and Automation (ICRA), 2010*, no. 4, 2010, pp. 3733–3738.

[11] H. Ferreau, "qpOASES user's manual," Tech. Rep. March, 2011.

[12] B. Faverjon and P. Tournassoud, "A Local Based Approach for Path Planning of Manipulators With a High Number," pp. 1152–1159, 1987.

[13] A. Albu-Schäffer, C. Ott, and G. Hirzinger, "A unified passivity based control framework for position, torque and impedance control of flexible joint robots," *The International Journal of Robotics Research*, vol. 26, no. 1, pp. 23–39, 2007.

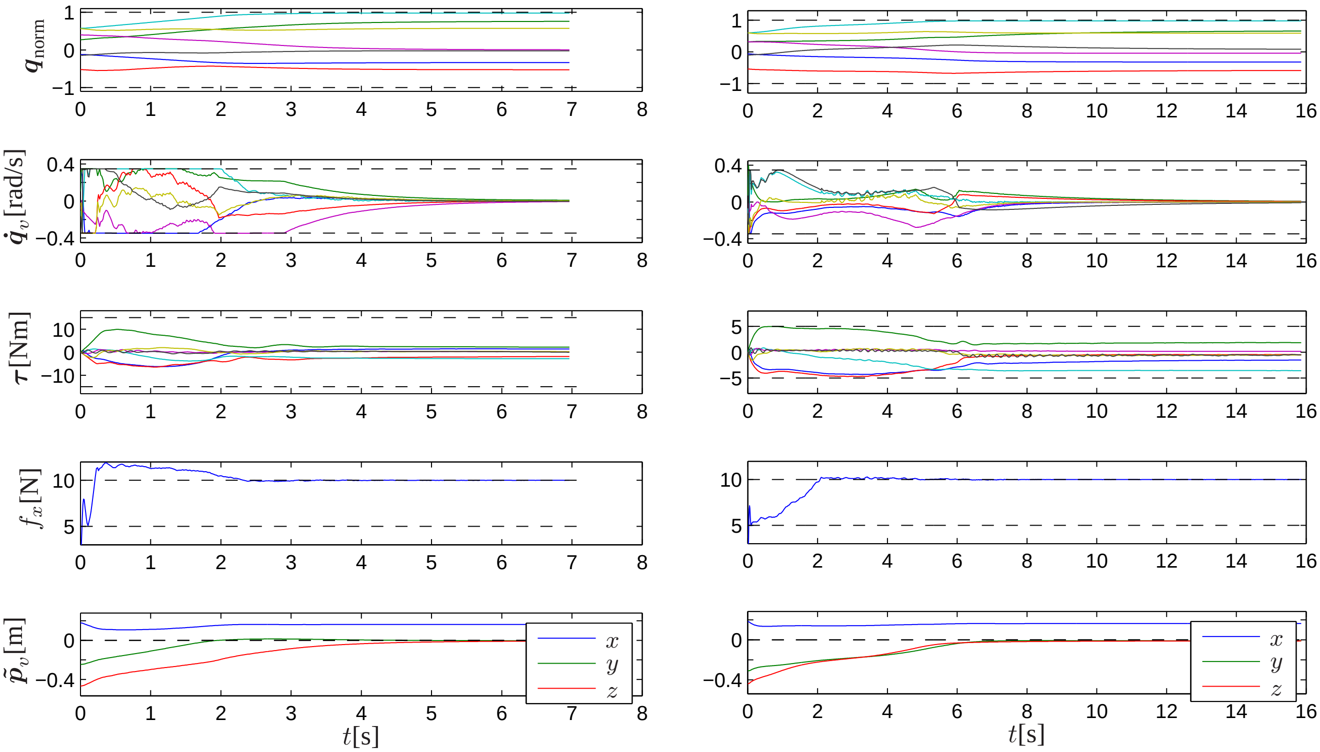

Fig. 4. Evolution of the main subtasks. The dotted lines denote the task bounds, respectively the desired task value. The Cartesian position error $\tilde{\boldsymbol{p}}_v = \boldsymbol{p}_{v_d} - \boldsymbol{p}_v$ and the normalized joint positions $\boldsymbol{q}_{\text{norm}} \in [-1, 1]$ are plotted for better compactness.
Left: the high gain in the positioning task leads quickly to saturation of the joint velocities $\dot{\boldsymbol{q}}_v$ and also to $\boldsymbol{f}_x$ violating its bounds. At approximately $t = 3.5s$, joint 4 (turquoise) reaches its limit and the rest of the task is completed without this joint. Note that the x-component of $\boldsymbol{p}_{v_d}$ does not converge to its goal position, since the higher priority force task is preventing it from penetrating further into the table.
Right: The same task parameters are applied here, besides the torque limits are reduced, stating the new bottleneck for task execution.

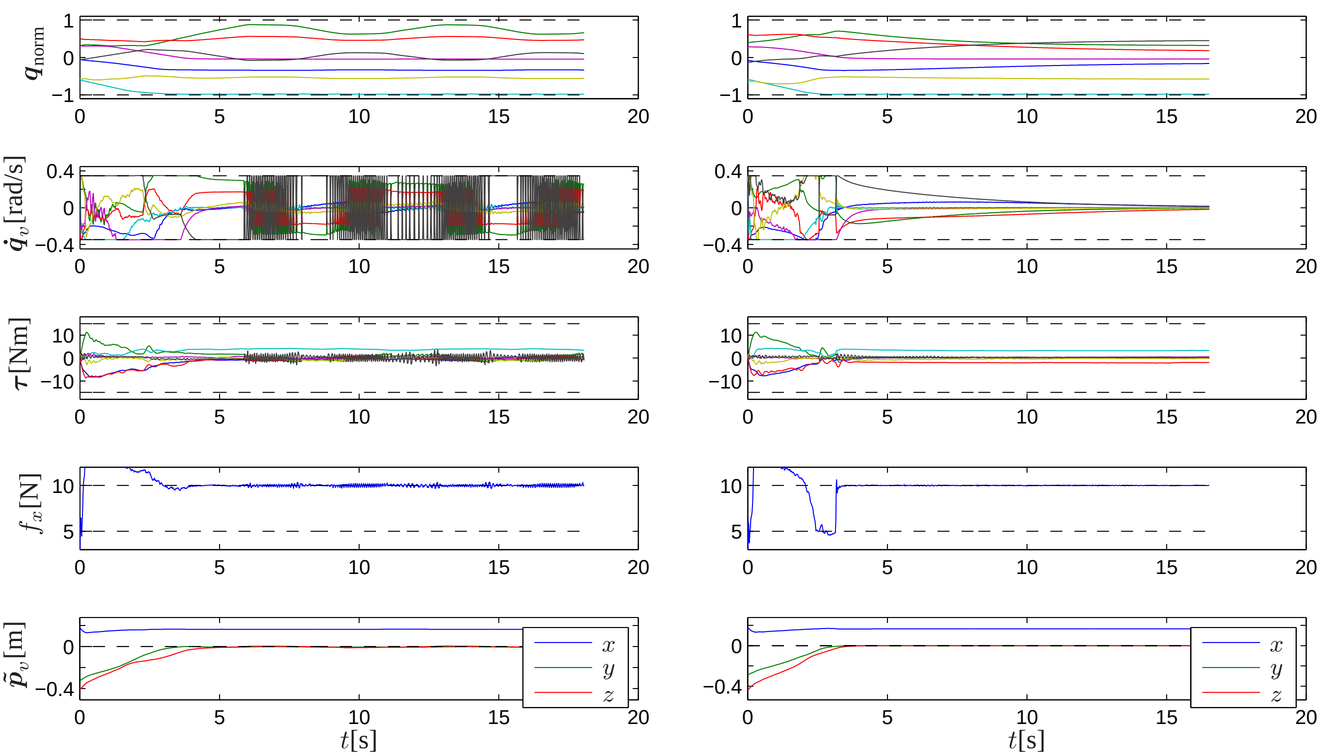

Fig. 5. Constrained point to point motion with regularization $\rho = 0$.
Left: without the regularization the solution can become instable if tasks are conflicting.
Right: when removing the positioning constraint in x-direction the solution is stable.