Parsimonious Kinematic Control of Highly Redundant Robots

Viniçius Mariano Gonçalves, Philippe Fraisse, André Crosnier, Bruno Vilhena Adorno

Parsimonious Kinematic Control of Highly Redundant Robots

Vinicius Mariano Gonçalves[1], Philippe Fraisse[2], André Crosnier[2], Bruno Vilhena Adorno[1]

Abstract— **When a robot is highly redundant in comparison to the task to be executed, current control techniques are not "economic" in the sense that they demand, most of the time unnecessarily, all the joints to move. Such behavior can be undesirable for some applications. In this direction, this work proposes a new control paradigm based on linear programming that intrinsically provides a parsimonious control strategy, that is, one in which few joints move. In addition to a formal stability proof, the paper presents simulation and experimental results on the HOAP-3 humanoid robot. Finally, a comparison is made with a least-square method based on the pseudoinverse of the task Jacobian, showing that the proposed method indeed uses fewer joints than the classic one.**

I. Introduction

The use of redundant robots is on the rise not only in industrial environments [1] but also in human-robot interaction applications, where frequently humanoid robots or mobile manipulators are used. The robot kinematic control strategies are widely based on the pseudoinverse of the task Jacobian ([2], [3], [4]) or, when there are constraints, usually convex quadratic programming solvers are used (see [5], [6], [7]).

Under redundancy, however, these strategies may generate movements that are not very economic in the sense that a huge amount of actuators must be needlessly activated to achieve the desired task. For instance, the traditional pseudoinverse approach generates a trajectory in which *all* joints move, but intuitively, in some cases a task represented by a certain amount of degrees of freedom (DOF) in the task-space can be solved with at most the same number of DOF in the joint space.

This paper is concerned with the development and analysis of a control paradigm that induces "economic" movements in face of highly redundant (overactuated) systems. The idea is to exploit the redundancy of the system to generate control laws that demand as few actuators as possible, which could provide several benefits. For instance, recent research suggests that parsimonious control can potentially provide more natural human-like movements [8], [9], which could improve human-robot interactions. Moreover, parsimonious control also provides easier networked control by reducing the amount of control data exchanged during wireless communication (see [10] and the references therein) and, in addition, it could provide reduced feedback-induced negative effects, such as excitation of non-modeled dynamics. The latter is justified by the fact that in kinematic control the inertial parameters, which depend on the robot configuration, are not taken into consideration. Therefore, the less these configurations change the less the imprecise parameters will play a relevant role in the system and, consequently, the less will be the induced disturbance in the control. In this sense, some recent works, specially in model predictive control, exploit techniques borrowed from compressed sensing to generate control actions with the sparse property (see [11], [12]).

This work has been supported by CAPES, CNPq, and FAPEMIG.

[1]Vinicius Mariano Gonçalves (mariano@cpdee.ufmg.br) and Bruno Vilhena Adorno (adorno@ufmg.br) are with Universidade Federal de Minas Gerais (UFMG), Brazil.

[2]André Crosnier (crosnier@lirmm.fr) and Philippe Fraisse (fraisse@lirmm.fr) are with Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM), France.

A. Compressed sensing

The proposed approach is inspired on the recent developments in compressed sensing in the setting of signal processing (see [13]). In this setting, it is desired to represent a signal as a linear combination of previously-defined basis functions, and the challenge is to represent the signal in the most memory-efficient way possible without losing too much (if any) information. The signal is then sampled in a way that the number of samples is much smaller than the number of basis signals. The values of the reconstructed signal and the original one must match in these samples and therefore the problem of finding the coefficients of the linear combination consists of solving a highly *undetermined* linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$. There are many possible solutions to this equation, but it is of special interest to find one that is *sparse*, or *parsimonious*, which means a low number of non-zero coefficients. The reasoning for that in the signal processing setting is that memory can be saved by storing only the non-zero coefficients. Such sparse decomposition is often possible, as experience shows that usually practical signals have significant energy only in some sparse frequencies along the spectrum. There is a similar intuition in robotics: for highly redundant robots, tasks can often be completed without using all available joints, differently from what happens when conventional approaches are used, where usually all joints are used simultaneously.

The problem of compressed sensing and finding sparse control inputs for trajectory generation/control under the differential inverse kinematics approach have exactly the same mathematical nature, an undetermined system. Indeed, in the latter equations of the form $\mathbf{J}\dot{\mathbf{q}} = \dot{\mathbf{e}}_d$ appear, in which $\mathbf{J}$ is the task Jacobian, $\dot{\mathbf{q}}$ is the target joint velocity (the control input) and $\dot{\mathbf{e}}_d$ is a desired velocity in the task/operational space. When the system is redundant and the matrix $\mathbf{J}$ has a number of columns considerably greater than the number of rows, there is the possibility of finding sparse solutions $\dot{\mathbf{q}}$. In the specific context of robotics, the widely popular choice for solving this problem, which is based on the pseudoinverse of the Jacobian, most often generates non-parsimonious solutions. Although the mathematical structure used in the context of compressed sensing and in the control of redundant robots are essentially the same, the theoretical challenges are different. In the control context, for instance, there is the necessity to prove stability for the closed loop induced by this approach, an issue nonexistent in the signal processing setting.

Finding the *smallest* number of non-zero coefficients for the solution of a linear system $\mathbf{Ax} = \mathbf{b}$ is NP hard (see [14]). Mathematically, the formulation is given by

$$\begin{aligned} \underset{\mathbf{x}}{\text{argmin}} \quad & \|\mathbf{x}\|_0 \\ \text{subject to} \quad & \mathbf{Ax} = \mathbf{b} \end{aligned} \tag{1}$$

in which $\|\cdot\|_0$ is the "norm" that counts the number of non-zero entries.

However, adequate solutions can be found very efficiently if relaxations are done. For instance, using a specific convex positive definite error metric between $\mathbf{Ax}$ and $\mathbf{b}$, namely the 1-norm, the optimization problem can be reduced to a linear program. Linear programs can be solved by using the Simplex algorithm, which has the property of *intrinsically* generating reasonably sparse solutions, even though it may be not the minimum amount of nonzero entries as in Problem 1. This approach has the benefit that linear programs can be solved, in average, very efficiently with the Simplex method [15]. The formulation will be valid even when it is not possible to have $\mathbf{Ax} = \mathbf{b}$ (the system is not redundant), giving in this case the best possible solution in the 1-norm sense. Indeed, even when parsimony is not possible at all, the benefit of reduced computational time should remain.

B. Linear programming for inverse differential kinematics problems

It is not the first time that linear programming is proposed to solve inverse differential kinematic problems although, interestingly, none of the previous works neither made the connection with compressed sensing nor observed the potential of having sparse solutions. Formal guarantee of stability was also absent. Ho et al. [16] proposed a particular case of the general formulation proposed in the present paper, which aims at generating trajectories for human-like 3D models. They note that linear programming can be efficient, in terms of computation effort, to generate trajectories when the system has inequality constraints. Indeed, according to their simulations, in some settings even in the unconstrained case it outperforms the classic pseudoinverse approach by taking much less computational effort, and this is as truer as the number of constraints and degrees of freedom grow. The authors of [17] also propose that a general convex function can be used as a metric, and explicitly mention the $\|\cdot\|_1$ norm, but also do not show simulation results with norms other than the Euclidean one.

C. Contributions

The main contribution of this paper is the formulation of the inverse differential kinematics as a linear program and the associated formal guarantees of Lyapunov stability. It is also shown that parsimonious behavior is obtained when the optimization problem is solved by using the Simplex method, which has low computational cost. Both simulations and experimental results in the HOAP-3 humanoid robot are shown to illustrate the proposed methodology.

II. Linear programming for robot motion generation

For task-solving approaches using differential inverse kinematics and task functions ([18]), control actions in the joint space often follow as a result of optimization problems. These optimizations problems seem to be, mostly, quadratic programming ones (see [2], [3], [4], [5], [6], [7]), for which solutions can be computed explicitly when there are no constraints. To exemplify, let $\mathbf{q}$ be the variable in the joint space and $\mathbf{e}(\mathbf{q}) = \mathbf{0}$ a description of a desired configuration. Hence, the goal is to drive $\mathbf{e}(\mathbf{q}(t))$ to $\mathbf{0}$. Since $\dot{\mathbf{e}}(\mathbf{q}){=}\mathbf{J}(\mathbf{q})\dot{\mathbf{q}}$, setting the target $\dot{\mathbf{e}}_d$ to $-\eta\mathbf{e}(\mathbf{q})$, for $\eta > 0$, aiming for an exponential convergence of the error, one can obtain the target $\dot{\mathbf{q}}(t)$ by solving the following optimization problem

$$\min_{\dot{\mathbf{q}}} \quad \|\mathbf{J}\dot{\mathbf{q}} - \dot{\mathbf{e}}_d\|_2 \quad \Longrightarrow \quad \min_{\dot{\mathbf{q}}} \quad \|\mathbf{J}\dot{\mathbf{q}} + \eta\mathbf{e}\|_2 \tag{2}$$

in which $\mathbf{J} \triangleq \mathbf{J}(\mathbf{q})$ and $\mathbf{e} \triangleq \mathbf{e}(\mathbf{q})$.

The minimal 2-norm solution to the previous optimization problem can be given by using the pseudoinverse as $\dot{\mathbf{q}} = -\eta\mathbf{J}^{+}\mathbf{e}$. However, as mentioned in Section I, this approach frequently implies a solution in which *all* joints move, since $\dot{\mathbf{q}}$ in this case frequently has few, if any, non-zero entries.

In case that some constraints for $\dot{\mathbf{q}}$ (e.g., speed limits or obstacle avoidance) are imposed in the form $\mathbf{W}(\mathbf{q})\dot{\mathbf{q}} \leq \mathbf{w}(\mathbf{q})$, dense solutions may arise in the solution of the problem

$$\begin{aligned} \min_{\dot{\mathbf{q}}} \quad & \|\mathbf{J}\dot{\mathbf{q}} + \eta\mathbf{e}\|_2 \\ \text{subject to} \quad & \mathbf{W}\dot{\mathbf{q}} \leq \mathbf{w}. \end{aligned} \tag{3}$$

in which $\mathbf{W} \triangleq \mathbf{W}(\mathbf{q})$ and $\mathbf{w} \triangleq \mathbf{w}(\mathbf{q})$. Problem 3 has no closed-form solution and the problem, which is a convex quadratic program, needs to be solved with a numerical solver.

If parsimony is sought, an approach that can also potentially generate parsimonious behavior, when possible, will be discussed in the next section.

A. Parsimony

This section shows that the problem of inverse differential kinematics can be posed as a linear program, which in addition of being computationally efficient to solve, also provides parsimony as long as an appropriate solver is used. The problem can be seen as follows

$$\begin{aligned} \min_{\dot{\mathbf{q}}} \quad & \|\mathbf{J}\dot{\mathbf{q}} + \eta\mathbf{e}\|_1 \\ \text{subject to} \quad & \mathbf{W}\dot{\mathbf{q}} \leq \mathbf{w}, \end{aligned} \tag{4}$$

where $\dot{\mathbf{q}} \in \mathbb{R}^m$, $\mathbf{J} \in \mathbb{R}^{k\times m}$, $\mathbf{e} \in \mathbb{R}^k$, $\mathbf{W} \in \mathbb{R}^{s\times m}$ and $\mathbf{w} \in \mathbb{R}^s$.

The formulation in Problem 4 by itself does not guarantee parsimony. However, since it can be transformed into a linear program, the well-known Simplex algorithm can be used to solve it and then, thanks to the intrinsic properties of that algorithm, parsimony appears whenever possible. To explain this behavior, let $\mathbf{g}, \mathbf{c} \in \mathbb{R}^v$, $\mathbf{B} \in \mathbb{R}^{u \times v}$, $\mathbf{b} \in \mathbb{R}^u$, and $c \in \mathbb{R}$. Consider the canonical form of linear programs

$$\begin{aligned} \min_{\mathbf{g}} \quad & \mathbf{c}^T\mathbf{g} + c \\ \text{subject to} \quad & \mathbf{B}\mathbf{g} = \mathbf{b}; \\ & \mathbf{g} \geq \mathbf{0}. \end{aligned} \tag{5}$$

It is important to note that all linear programs can be written in this form. If one uses the *Simplex* algorithm to solve linear programs, at most u entries of the v ones in $\mathbf{g}$ will be non-zero. This is due to how the Simplex algorithm works (see [19]): at each step, the vector $\mathbf{g}$ is split in u *basic* variables and $v-u$ *nonbasic* variables. All the nonbasic variables are set to 0 while the basic ones may or may not be 0. Thus, when possible, linear programming implements parsimony intrinsically when Simplex algorithm is used.

In order to use the Simplex algorithm, Problem 4 must first be transformed into a linear program. In statistics, this problem is known as *constrained least absolute deviation regression* and many different formulations as linear programs have been proposed. According to experiments done in [20], the most efficient formulation, for computational purposes, is the following one:

$$\begin{aligned} \min_{(\dot{\mathbf{q}},\mathbf{y})} \quad & \mathbf{1}^T(2\mathbf{y} - (\mathbf{J}\dot{\mathbf{q}} + \eta\mathbf{e})) \\ \text{subject to} \quad & \mathbf{J}\dot{\mathbf{q}} - \mathbf{y} \leq -\eta\mathbf{e}; \\ & \mathbf{W}\dot{\mathbf{q}} \leq \mathbf{w}; \\ & \mathbf{y} \geq \mathbf{0}, \end{aligned} \tag{6}$$

in which $\mathbf{1}$ is a vector of ones of appropriate dimension.

To write Problem 6 in the canonical form (5), $\dot{\mathbf{q}}$ must be somehow rewritten, since all variables are nonnegative in (5). For handling this, let $\dot{\mathbf{q}}_P$ and $\dot{\mathbf{q}}_N$ be nonnegative variables. One can write without loss of generality that $\dot{\mathbf{q}} = \dot{\mathbf{q}}_P - \dot{\mathbf{q}}_N$.

This decomposition is not only useful to transform Problem 6 into the canonical one, but also to avoid a "flaw" in the formulation of Problem 4: it does not guarantee that $\dot{\mathbf{q}} = \mathbf{0}$ when $\mathbf{e} = \mathbf{0}$ (i.e., the task is achieved). This implies that when the robot achieves the task it may still continue to move its joints, which may be undesirable. Such a concern is nonexistent with the pseudoinverse approach because it implicitly minimizes $\|\dot{\mathbf{q}}\|_2$ and then, clearly, $\dot{\mathbf{q}} = \mathbf{0}$ when $\mathbf{e} = \mathbf{0}$. There are many ways that this problem can be handled, but a simple one, and convenient in the linear programming case, is to choose a positive definite function $\beta(\mathbf{e})$ and impose that $\|\dot{\mathbf{q}}\|_1 \leq \beta(\mathbf{e})$. This way, if $\mathbf{e} = \mathbf{0}$ one has necessarily $\dot{\mathbf{q}} = \mathbf{0}$. This constraint can be implemented very simply as a linear constraint using the previously-introduced decomposition $\dot{\mathbf{q}} = \dot{\mathbf{q}}_P - \dot{\mathbf{q}}_N$. Indeed, clearly $\|\dot{\mathbf{q}}\|_1 = \|\dot{\mathbf{q}}_P - \dot{\mathbf{q}}_N\|_1 \leq \|\dot{\mathbf{q}}_P\|_1 + \|\dot{\mathbf{q}}_N\|_1 = \mathbf{1}^T\dot{\mathbf{q}}_P + \mathbf{1}^T\dot{\mathbf{q}}_N$, because $\dot{\mathbf{q}}_P$ and $\dot{\mathbf{q}}_N$ are nonnegative. Thus, the constraint $\mathbf{1}^T\dot{\mathbf{q}}_P + \mathbf{1}^T\dot{\mathbf{q}}_N \leq \beta(\mathbf{e})$ implies $\|\dot{\mathbf{q}}\|_1 \leq \beta(\mathbf{e})$, as desired.

By introducing slack variables $\mathbf{z}_A$, $\mathbf{z}_B$, z_C, letting $\mathbf{g} = [\dot{\mathbf{q}}_P^T \ \dot{\mathbf{q}}_N^T \ \mathbf{y}^T \ \mathbf{z}_A^T \ \mathbf{z}_B^T \ z_C]^T$, and imposing the additional constraint $\mathbf{1}^T\dot{\mathbf{q}}_P + \mathbf{1}^T\dot{\mathbf{q}}_N \leq \beta(\mathbf{e})$, one can recast Problem 6 as

$$\begin{aligned} \min_{\mathbf{g}} \quad & [-\mathbf{1}^T\mathbf{J} \ \ \mathbf{1}^T\mathbf{J} \ \ 2{\cdot}\mathbf{1}^T \ \ \mathbf{0}^T \ \ \mathbf{0}^T \ \ 0]\mathbf{g} - \eta\mathbf{1}^T\mathbf{e} \\ \text{subject to} \quad & \begin{bmatrix} \mathbf{J} & -\mathbf{J} & -\mathbf{I} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{W} & -\mathbf{W} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{1}^T & \mathbf{1}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} & 1 \end{bmatrix} \mathbf{g} = \begin{bmatrix} -\eta\mathbf{e} \\ \mathbf{w} \\ \beta(\mathbf{e}) \end{bmatrix} \\ & \mathbf{g} \geq \mathbf{0}, \end{aligned} \tag{7}$$

Problem 7 has the canonical form of the linear program (5). In this case, the number of rows of matrix $\mathbf{B} \in \mathbb{R}^{u \times v}$ is $u{=}k{+}s{+}1$, which means that if the Simplex algorithm is used, at most $k{+}s{+}1$ entries of $\mathbf{g}$ will be non-zero and hence at most $k{+}s{+}1$ entries of $(\dot{\mathbf{q}}_P - \dot{\mathbf{q}}_N) = \dot{\mathbf{q}} \in \mathbb{R}^m$ will be non-zero. This implies the sparsity property when $m{>}k{+}s{+}1$.

Note that in the previous analysis, from $u{=}k{+}s{+}1$ DOF that *can possibly be* used to solve the task, $s{+}1$ are used due to inequality constraints, in which s DOF are required by the s inequalities in $\mathbf{W}\dot{\mathbf{q}} \leq \mathbf{w}$ and one DOF is required by the constraint $\mathbf{1}^T\dot{\mathbf{q}}_P + \mathbf{1}^T\dot{\mathbf{q}}_N \leq \beta(\mathbf{e})$. However, it turns out that some of these constraints may be not overly stringent and be fulfilled with strict inequality when the problem is solved. In this case, no DOF for that particular inequality is taken and then the number of nonzero entries may be smaller. Such behavior is observed in simulation and experimental results, as shown in Section III.

It is important to stress that the previous result guarantees that $\dot{\mathbf{q}}(t)$ always has at most $k{+}s{+}1$ non-zero entries, but it does not mean that during the *entire trajectory* the *same* $k{+}s{+}1$ entries of $\dot{\mathbf{q}}(t)$ will be non-zero. In fact, along the trajectory different sets of $k{+}s{+}1$ variables may be non-zero, which means that the solution of (7) guarantees sparsity only locally, not globally. Nevertheless, as the experimental and simulation results of Section III show, this sparsity can be global and, if it is not, it may still be the case that there are some joints that are never used during the whole trajectory, achieving some partial global sparsity.

B. Stability analysis

A natural concern is about the closed-loop stability when the input $\dot{\mathbf{q}}$ is generated by solving the optimization Problem 4 with the additional constraint $\|\dot{\mathbf{q}}\|_1 \leq \beta(\mathbf{e})$, where β is a positive definite function, which is equivalent to

$$\min_{\dot{\mathbf{q}} \in \mathcal{A}(\mathbf{q})} \quad \|\mathbf{J}\dot{\mathbf{q}} + \eta\mathbf{e}\|_1 \tag{8}$$

with

$$\mathcal{A}(\mathbf{q}) = \{\mathbf{p} : \ \mathbf{W}\mathbf{p} \leq \mathbf{w}\} \cap \{\mathbf{p} : \ \|\mathbf{p}\|_1 \leq \beta(\mathbf{e}(\mathbf{q}))\}, \tag{9}$$

in which $\mathbf{W} \triangleq \mathbf{W}(\mathbf{q})$ and $\mathbf{w} \triangleq \mathbf{w}(\mathbf{q})$.

Considering the system as a pure integrator, $\dot{\mathbf{q}}(t){=}\dot{\mathbf{q}}_d(t)$, in which the target velocity $\dot{\mathbf{q}}_d(t)$ is a minimizer of Problem 8, stability analysis can be performed. It is important to note that the target specification is written as $\mathbf{e}(\mathbf{q}(t)){=}\mathbf{0}$. Also, $\dot{\mathbf{q}}(t){=}\dot{\mathbf{q}}_d(t)$ is indeed a dynamical system that depends on $\mathbf{q}$ because when the optimization problem is solved in a given step by finding $\dot{\mathbf{q}}_d(t)$, this value is integrated (or, in the real robot, this velocity is applied to the joints and they move) and therefore $\mathbf{q}$ changes. When $\mathbf{q}$ changes, so does $\mathbf{e}(\mathbf{q})$, $\mathbf{J}(\mathbf{q})$ and the set $\mathcal{A}(\mathbf{q})$ (the latter changes because $\mathbf{W}(\mathbf{q}), \mathbf{w}(\mathbf{q})$ and $\beta(\mathbf{e}(\mathbf{q}))$ change). Hence, the parameters are changed in Problem 8 and it has to be solved again in the next loop.

However, the set of all minimizers $\dot{\mathbf{q}}_d(t)$ of Problem 8 is usually not a singleton (a set with a single member) so the dynamical system generated is not an ordinary differential equation of the form $\dot{\mathbf{q}}(t){=}\mathbf{f}(\mathbf{q}(t))$, but a *differential inclusion* $\dot{\mathbf{q}}(t) \in \mathcal{F}(\mathbf{q}(t))$ since the function $\mathcal{F} : \mathbb{R}^m \mapsto 2^{\mathbb{R}^m}$ is set-valued (it is a *correspondence* between $\mathbb{R}^m$ and $\mathbb{R}^m$). More specifically, in the case of Problem 8,

$$\mathcal{F}(\mathbf{q}) = \underset{\mathbf{p}\in\mathcal{A}(\mathbf{q})}{\operatorname{argmin}} \left\|\mathbf{J}(\mathbf{q})\mathbf{p} + \eta\mathbf{e}(\mathbf{q})\right\|_1 . \tag{10}$$

In order to study stability for this differential inclusion, it is necessary to establish three lemmas.

Lemma 1: Let $\mathbf{e}(\mathbf{q})$ and $\mathbf{J}(\mathbf{q})$ be continuous functions for all $\mathbf{q}$. Let $\mathcal{A}(\mathbf{q})$ be continuous, non-empty, compact-valued and convex-valued for all $\mathbf{q}$. Then, the correspondence $\mathcal{F}(\mathbf{q})$ in (10) is non-empty, convex-valued, compact-valued and upper semicontinuous.

Proof: The function $(\mathbf{q}, \mathbf{p}) \mapsto \left\|\mathbf{J}(\mathbf{q})\mathbf{p} + \eta\mathbf{e}(\mathbf{q})\right\|_1$ is clearly convex in $\mathbf{p}$. Furthermore, due to the assumptions on $\mathbf{J}(\mathbf{q})$ and $\mathbf{e}(\mathbf{q})$ it is also continuous for all $\mathbf{q}$ and $\mathbf{p}$.

Using this fact and the assumptions in $\mathcal{A}(\mathbf{q})$, the proof can be concluded directly from an extended form of Berge's maximum theorem found in [21].[1] ■

Lemma 2: Let ϕ be a convex function, η a scalar, $\mathbf{v}$ a vector and $\mathbf{e}(\mathbf{q})$ a differentiable function with Jacobian $\mathbf{J}(\mathbf{q})$. Let $V(\mathbf{q}) \triangleq \phi(\eta\mathbf{e}(\mathbf{q}))$, and $(\partial\phi)(\mathbf{u})$ be the subgradient of ϕ evaluated at $\mathbf{u}$. Then, there exists a vector $\mathbf{z}_L(\mathbf{q}, \mathbf{v}) \in (\partial\phi)(\eta\mathbf{e}(\mathbf{q}))$ such that

$$\liminf_{\epsilon\to 0+} \frac{V(\mathbf{q}+\epsilon\mathbf{v}) - V(\mathbf{q})}{\epsilon} \leq \eta(\mathbf{J}(\mathbf{q})\mathbf{v})^T\mathbf{z}_L(\mathbf{q}, \mathbf{v}). \tag{11}$$

Proof: It is known that for any convex function ϕ and vectors $\mathbf{r}, \mathbf{s}$ (see [22])

$$\forall \mathbf{z} \in (\partial\phi)(\mathbf{r}) \ , \ \phi(\mathbf{r}) - \phi(\mathbf{s}) \leq (\mathbf{r}-\mathbf{s})^T\mathbf{z}. \tag{12}$$

Consider $\mathbf{r}{=}\eta\mathbf{e}(\mathbf{q}+\epsilon\mathbf{v})$ and $\mathbf{s}{=}\eta\mathbf{e}(\mathbf{q})$ in the previous expression, in order to conclude that

$$\begin{aligned}&\forall \mathbf{z} \in (\partial\phi)(\eta\mathbf{e}(\mathbf{q}+\epsilon\mathbf{v})),\\ &\quad V(\mathbf{q}+\epsilon\mathbf{v}) - V(\mathbf{q}) \leq \eta(\mathbf{e}(\mathbf{q}+\epsilon\mathbf{v}) - \mathbf{e}(\mathbf{q}))^T\mathbf{z},\end{aligned} \tag{13}$$

in which the definition $V(\mathbf{q}) = \phi(\eta\mathbf{e}(\mathbf{q}))$ was used. Take a positive sequence ϵ_n, $n \in \mathbb{Z}^+$ of ϵ's that converges to 0, and a convergent sequence $\mathbf{z}_n \in (\partial\phi)(\eta\mathbf{e}(\mathbf{q}{+}\epsilon_n\mathbf{v}))$.[2] Furthermore, divide both sides of (13) by ϵ_n (the inequality will keep the direction because $\epsilon_n > 0$) and take the $\liminf_{n\to\infty}$, which also preserves the direction of the inequality. Hence

$$\begin{aligned}&\liminf_{n\to\infty} \frac{V(\mathbf{q}+\epsilon_n\mathbf{v}) - V(\mathbf{q})}{\epsilon_n} \leq \\ &\qquad \liminf_{n\to\infty} \eta \underbrace{\left(\frac{\mathbf{e}(\mathbf{q}+\epsilon_n\mathbf{v}) - \mathbf{e}(\mathbf{q})}{\epsilon_n}\right)^T}_{A} \underbrace{\mathbf{z}_n}_{B} .\end{aligned} \tag{14}$$

The limit inferior in the right-hand side of (14) can be replaced by the ordinary limit and distributed between the two terms A and B because both limits of A and B exist ($\mathbf{e}(\mathbf{q})$ is differentiable and the sequence $\mathbf{z}_n$ is convergent).

Using the chain rule for differentiation, the limit of A reduces to $\mathbf{J}(\mathbf{q})\mathbf{v}$ since $\partial\mathbf{e}/\partial\mathbf{q}{=}\mathbf{J}$. In the calculation of the limit of B, one notes that subgradients of convex functions are upper semicontinuous correspondences (see [22]). Since $\mathbf{e}$ is continuous, the correspondence $(\partial\phi)(\eta\mathbf{e}(\mathbf{q}))$ is also upper semicontinuous on $\mathbf{q}$. Therefore, since the sequence $\mathbf{z}_n$ is convergent, $\lim_{n\to\infty}\mathbf{z}_n$ exists and, by the definition of upper semicontinuity of correspondences, this limit belongs to $(\partial\phi)(\eta\mathbf{e}(\mathbf{q}))$ (remember that $\lim_{n\to\infty}\mathbf{q}{+}\epsilon_n\mathbf{v} = \mathbf{q}$). Defining $\mathbf{z}_L(\mathbf{q}, \mathbf{v}) \triangleq \lim_{n\to\infty}\mathbf{z}_n$, the proof is concluded. ■

Lemma 3: If $\phi(\mathbf{u})$ is a convex function and $(\partial\phi)(\mathbf{u})$ is its subgradient evaluated at $\mathbf{u}$, then for any $\mathbf{x}$ and $\mathbf{y}$

$$\forall \mathbf{z} \in (\partial\phi)(\mathbf{y}) \ , \ \mathbf{x}^T\mathbf{z} \leq \phi(\mathbf{x}+\mathbf{y}) - \phi(\mathbf{y}). \tag{15}$$

Proof: Let $\mathbf{r} = \mathbf{y}$ and $\mathbf{s} = \mathbf{x}+\mathbf{y}$ in (12) and multiply both sides of the inequality by -1, and the proof is concluded. ■

The following result, established in [23], is important to establish Lyapunov stability by using non-differentiable functions.

Theorem 1: (see [23]) Let $\mathcal{F}(\mathbf{q})$ be a non-empty, compact-valued, convex-valued upper semicontinuous correspondence. If there exists a function $V(\mathbf{q})$ which is positive definite to a set $\mathcal{S}$ (zero in the set and positive outside it) and locally Lipschitz such that

$$\forall \dot{\mathbf{q}} \in \mathcal{F}(\mathbf{q}) \ , \ \liminf_{\epsilon\to 0+} \frac{V(\mathbf{q}+\epsilon\dot{\mathbf{q}}) - V(\mathbf{q})}{\epsilon} \leq 0 \tag{16}$$

then the differential inclusion $\dot{\mathbf{q}} \in \mathcal{F}(\mathbf{q})$ is Lyapunov stable to $\mathcal{S}$. ■

The closed-loop stability is now proved in the next proposition.

Proposition 1: Consider the following differential inclusion

$$\begin{aligned}&\dot{\mathbf{q}}(t) \in \underset{\mathbf{p}\in\mathcal{B}(\mathbf{q}(t))}{\operatorname{argmin}} \left\|\mathbf{J}(\mathbf{q}(t))\mathbf{p} + \eta\mathbf{e}(\mathbf{q}(t))\right\|_1 ,\\ &\mathbf{q}(0) = \mathbf{q}_0,\end{aligned} \tag{17}$$

[1]The theorem deals with maximum of functions, but an analogous to minima easily follows.

[2]Such convergent sequence always exists. Suppose there is a bounded sequence $\mathbf{z}_n$ (clearly the sequence can be chosen to be bounded since all members of the subgradient are bounded) and it is not convergent. Then Bolzano-Weierstrass theorem guarantees that there exists a convergent subsequence of $\mathbf{z}_n$. This convergent subsequence can be taken along with the respective subsequence of ϵ_n.

in which $\mathbf{e}(\mathbf{q})$ and $\mathbf{J}(\mathbf{q}){=}\partial\mathbf{e}(\mathbf{q})/\partial\mathbf{q}$ are continuous functions, $\mathbf{e}(\mathbf{q})$ is locally Lipschitz, $\eta > 0$, and $\mathcal{B}(\mathbf{q})$ is a continuous, compact-valued, convex-valued and non-empty correspondence with $\mathbf{0} \in \mathcal{B}(\mathbf{q})$ for all $\mathbf{q}$. Then the set $\mathcal{S} \triangleq \{\mathbf{q} : \ \mathbf{e}(\mathbf{q}) = \mathbf{0}\}$ is Lyapunov stable.

Proof: Let $\dot{\mathbf{q}}$ be any particular member of the correspondence in the right side of (17), henceforth denoted by $\mathcal{F}(\mathbf{q})$.

From the definition of argmin, the optimal $\mathbf{p} = \dot{\mathbf{q}}$ from the feasible set $\mathcal{B}$ attains the minimum value of the objective function $H(\mathbf{p}) \triangleq \|\mathbf{J}\mathbf{p} + \eta\mathbf{e}\|_1$ (for the sake of notation, all dependencies on $\mathbf{q}$ will be dropped). In special, the value of $H(\dot{\mathbf{q}})$ is not greater than $H(\mathbf{0})$, since $\mathbf{p} = \dot{\mathbf{q}}$ globally minimizes H and $\mathbf{p} = \mathbf{0}$ is in the feasible set $\mathcal{B}$ by assumption. Consequently,

$$\underbrace{\|\mathbf{J}\dot{\mathbf{q}} + \eta\mathbf{e}\|_1}_{H(\dot{\mathbf{q}})} \leq \underbrace{\|\eta\mathbf{e}\|_1}_{H(\mathbf{0})} \implies \|\mathbf{J}\dot{\mathbf{q}} + \eta\mathbf{e}\|_1 - \|\eta\mathbf{e}\|_1 \leq 0. \quad (18)$$

Consider $\phi(\mathbf{u}) = \|\mathbf{u}\|_1$, which is a convex function. Apply Lemma 3 with $\mathbf{x} = \mathbf{J}\dot{\mathbf{q}}$ and $\mathbf{y} = \eta\mathbf{e}$, together with (18), to conclude that

$$\forall \mathbf{z} \in (\partial\phi)(\eta\mathbf{e}), \ (\mathbf{J}\dot{\mathbf{q}})^T\mathbf{z} \leq \|\mathbf{J}\dot{\mathbf{q}} + \eta\mathbf{e}\|_1 - \|\eta\mathbf{e}\|_1 \leq 0. \quad (19)$$

Since $\eta > 0$, then

$$\forall \mathbf{z} \in (\partial\phi)(\eta\mathbf{e}) \ , \ \eta(\mathbf{J}\dot{\mathbf{q}})^T\mathbf{z} \leq 0. \quad (20)$$

Considering Lemma 2 with $\mathbf{v} = \dot{\mathbf{q}}$, and the fact that $\mathbf{z}_L(\mathbf{q}, \dot{\mathbf{q}}) \in (\partial\phi)(\eta\mathbf{e})$, one can conclude together with (20) that

$$\liminf_{\epsilon \to 0+} \frac{V(\mathbf{q} + \epsilon\dot{\mathbf{q}}) - V(\mathbf{q})}{\epsilon} \leq 0 \quad (21)$$

in which, according to the definition in Lemma 2, $V(\mathbf{q}){=}\phi(\eta\mathbf{e}(\mathbf{q}))$. It is important to note that (21) holds true for any $\dot{\mathbf{q}} \in \mathcal{F}(\mathbf{q})$ because in the beginning of the proof it was assumed that $\dot{\mathbf{q}}$ is arbitrary.

Now, consider $V(\mathbf{q}) = \ \phi(\eta\mathbf{e}(\mathbf{q})) \triangleq \|\eta\mathbf{e}(\mathbf{q})\|_1$ as the Lyapunov function to the set $\mathcal{S} = \{\mathbf{q} : \ \mathbf{e}(\mathbf{q}) = \mathbf{0}\}$. This function is locally Lipschitz because both $\|\cdot\|_1$ and $\mathbf{e}(\mathbf{q})$ are locally Lipschitz functions.

According to Lemma 1, $\mathcal{F}(\mathbf{q})$ is a non-empty, compact-valued, convex-valued and upper semicontinuous correspondence. Therefore, using Theorem 1 together with (21) (remember that it holds true for any $\dot{\mathbf{q}} \in \mathcal{F}(\mathbf{q})$), one concludes that $\mathcal{S}$ is Lyapunov stable.

■

Note that the assumption $\mathbf{0} \in \mathcal{B}(\mathbf{q})$—which implies that the solution $\dot{\mathbf{q}} = \mathbf{0}$ is always admissible—is reasonable: it means that the robot is able to stop moving. The compactness is also reasonable, since the joint velocities always have natural limits and in practice all inequality constraints are non-strict.

Remark 1: Let $\mathcal{B}(\mathbf{q}) = \mathcal{A}(\mathbf{q})$, where $\mathcal{A}(\mathbf{q})$ is given in (9) with β being a positive definite continuous function. If $\mathbf{w}(\mathbf{q}) \geq \mathbf{0}$ for all $\mathbf{q}$, and $\mathbf{W}(\mathbf{q}), \mathbf{w}(\mathbf{q}), \mathbf{J}(\mathbf{q})$ and $\mathbf{e}(\mathbf{q})$ are continuous functions, the latter also being locally Lipschitz, then the assumptions in Proposition 1 hold. Indeed, the set $\mathcal{A}(\mathbf{q})$ is closed, since it is formed by non-strict linear inequalities on $\mathbf{p}$. Furthermore, since the function β is continuous, and hence always finite, the constraint $\|\dot{\mathbf{q}}\|_1 \leq \beta(\mathbf{e}(\mathbf{q}))$ by itself guarantees that the set is bounded. Thus $\mathcal{A}(\mathbf{q})$ is naturally compact. The property of being locally Lipschitz is also verified for the common functions used in kinematics models (sine, cosine, etc...). Consequently, the closed loop induced by solving the linear programming Problem 7, which is equivalent to Problem 8, is Lyapunov stable. ■

C. Computational complexity

The time needed for computing the control action can be crucial in real time applications on robotics, and thus the computational complexity of the proposed approach deserves special attention. Since the main burden of the method lies in solving Problem 7 with Simplex, this algorithm must be better analyzed.

According to [24], variants of the Simplex method take, on average, approximately a number of $\mathcal{O}(u + v)$ steps to solve linear programs as Problem 5, in which u and v are the number of constraints and variables, respectively. Taking into consideration that each step takes $\mathcal{O}(uv)$ operations [19], this leads to the conclusion that the Simplex algorithm has in practice a complexity of $\mathcal{O}(uv^2 + u^2v)$. Since in Problem 7 $u = k + s + 1$ and $v = 2m + 2k + s + 1$,[3] this implies in practice a time complexity of $\mathcal{O}((k + s)(2m + 2k + s)^2 + (k + s)^2(2m + 2k + s))$, which is polynomial.

Since the pseudoinverse approach cannot handle inequality constraints, in order to compare it with LP, let $s = 0$. Furthermore, in this paper only the redundant case is relevant; therefore, considering that the number m of DOFs is considerably greater than the number k of tasks, the practical complexity of LP reduces to $\mathcal{O}(m^2k)$.

In comparison, the pseudoinverse of a matrix $\mathbf{J} \in \mathbb{R}^{k \times m}$ can be also computed on $\mathcal{O}(m^2k)$ (under the consideration that m is much greater than k) using, for instance, a singular value decomposition (see [25]). Therefore, the asymptotic complexity of both approaches is comparable. However, it has been reported in the literature that LP can outperform the pseudoinverse (see [16]), specially when the Jacobian is sparse.

When there are inequality constraints, i.e. $s \neq 0$, algorithms based on pseudoinverse cannot be used and convex quadratic programs as Problem 3 must be solved with numerical solvers. Since convex quadratic programming is a strict superset of linear programming, the former is expected to be, in average, at least as hard to solve as the latter.

III. Experimental and simulation results

A. Simulation results on highly redundant robot

To illustrate the method, consider a serial 8-link planar robot in which each link has unit length and unit mass. The task consists of moving the end-effector to the point $[1 \ 7]^T$ while keeping the x component of the robot's center of mass (COM) in 0. Clearly, the robot is highly redundant in

[3]Recall that $\dot{\mathbf{q}}_P, \dot{\mathbf{q}}_N \in \mathbb{R}^m$, $\mathbf{y}, \mathbf{z}_A \in \mathbb{R}^k$, $\mathbf{z}_B \in \mathbb{R}^s$ and $z_C \in \mathbb{R}$.

TABLE I: Simulation of a serial 8-link planar robot: comparison of the input with respect to four different metrics.

Metric	PINV	LP	PINV/LP
$\int_0^\infty \|\dot{\mathbf{q}}(t)\|_1 dt$	1.644	1.264	1.301
$\sqrt{\int_0^\infty \|\dot{\mathbf{q}}(t)\|_2^2 dt}$	0.503	0.638	0.789
$\int_0^\infty \|\ddot{\mathbf{q}}(t)\|_1 dt$	0.021	0.016	1.329
$\sqrt{\int_0^\infty \|\ddot{\mathbf{q}}(t)\|_2^2 dt}$	0.011	0.011	1.000

comparison with the desired task and there are many possible movements. More specifically, the task Jacobian has 3 rows and 8 columns.

Both pseudoinverse and linear programming (Problem 7) approaches are compared in the solution to this task, with common parameter $\eta = 0.5$, which implies the same convergence rate for both methods. The only constraint in Problem 7 is the limitation of the speed with $\beta(\mathbf{e}) = 5\left\|\mathbf{e}\right\|_1$.

Fig. 1 shows the robot's configurations for both approaches. The initial configuration (cyan) corresponds to the manipulator being entirely in the vertical with the exception of the last joint that is tilted 30 degrees clockwise. Intermediate configurations are shown in intermediate colors (lighter is closer to the initial configuration and darker is closer to the final one). The final configuration is purple.

In the linear programming approach, at any specific time only three joints moved, whereas in the total movement four joints moved (1,4,5 and 8, see Fig. 2). More specifically, there is a switch between joints 4 and 5 at around 70 samples (see Fig 2). In the pseudoinverse approach, all of them moved during the whole trajectory. Table I shows a comparison of both approaches under four different metrics. For each task, the error dynamics in the first and second experiment are *exactly* the same, so the comparison is fair.

Since m=8, k=3 and s=0 (no constraints), the developments in Subsection II-A predict that for the linear programming approach, at a given time at most k+s+1=3+0+1=4

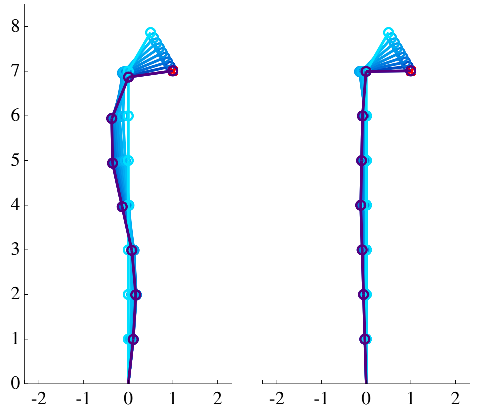


Fig. 1: Comparison of the sequence of configurations with pseudoinverse (*left*) and linear programming (*right*).

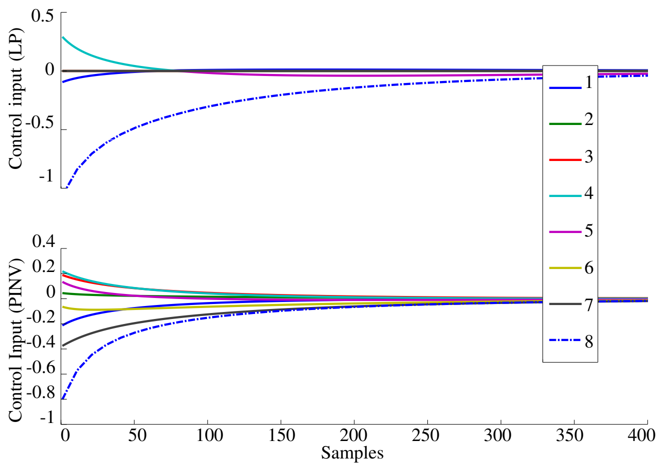


Fig. 2: Comparison of the control inputs $\dot{\mathbf{q}}$ from the pseudoinverse (*bottom*) and linear programming (*top*). The joints are numbered so the first joint is nearest to the base and the last one is nearest to the end-effector (see Fig. 1).

joints should move. In this case, only three joints moved at a given time (see Fig. 2). This is expected if the function $\beta(\mathbf{e})$ is not overly stringent, so the bound is always achieved with strict inequality and no degree of freedom is used to fulfill the constraint in the last row in Problem 7. The function defined for this simulation, $\beta(\mathbf{e}) = 5\left\|\mathbf{e}\right\|_1$, was loose enough for this to happen.

B. Experimental results on humanoid robot HOAP-3

To further illustrate the proposed methodology, an experiment was performed in the HOAP-3 humanoid robot (see Fig. 3).

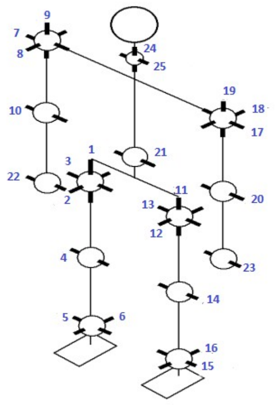


Fig. 3: HOAP-3 humanoid robot (*left*) and convention for numbering the DOF used in this paper (*right*).

In this experiment, initially the robot is standing with the right foot on the ground and the other one raised (see Fig. 4). The position is statically stable, since the (x, y) coordinates of the COM, which is located at $\mathbf{m}_0 = [-0.8cm \ \ -6cm]^T$ with respect to the reference frame of the right foot, is in the support polygon.

The task consists of moving the (x, y) coordinates of the COM to $\mathbf{m}_d = [-3.5cm \ \ -3.6cm]^T$, which is also in the support polygon and defined in a reference frame located at the right foot. The desired task is $\mathbf{e}(\mathbf{q}) = \mathbf{m}(\mathbf{q}) - \mathbf{m}_d = \mathbf{0}$.

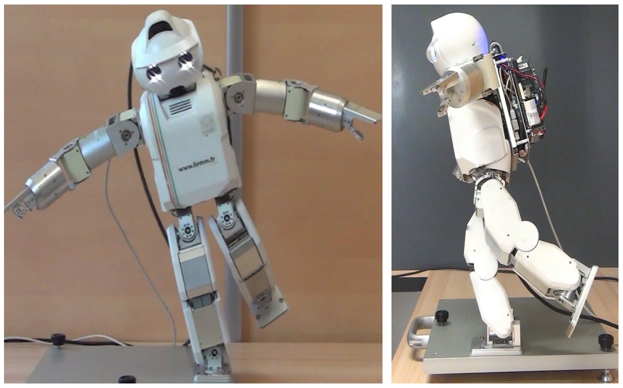


Fig. 4: Initial configuration of the robot for the experiment.

The experiment was performed with two approaches: the classic pseudoinverse of the Jacobian, $\dot{\mathbf{q}} = -\eta \mathbf{J}^+ \mathbf{e}$, and the formulation in Problem 7. In the latter case, the only constraint was defined as $\beta(\mathbf{e}) = 40 \|\mathbf{e}\|_1$. In both approaches, $\eta = 0.1$ to guarantee the same convergence rate.

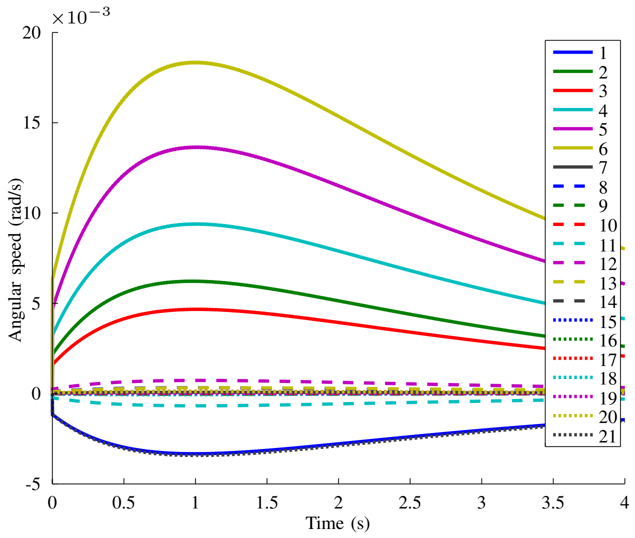
Furthermore, the trajectories of the robot joints were generated offline, both for the pseudoinverse approach and LP, and then executed in the real robot. Therefore, only the trajectory tracking of the joints was executed in closed loop. The implementation of the LP algorithms in the robot and consequently closing the entire loop will remain for a future development.

Fig. 5a and Fig. 5b show the required input $\dot{\mathbf{q}}$ in both scenarios: pseudoinverse (PINV) and linear programming (LP). Since the Jacobian matrix has two rows ($k = 2$) and there are no constraints ($s = 0$) other than the natural one of velocity, it was claimed in Subsection II-A that at most $k+s+1=2+0+1=3$ joints will move in a given moment in the LP approach. Indeed, only two joints moved at all, because the natural velocity constraint was not overly stringent and therefore was never saturated (always achieved with strict inequality) to require a DOF. Furthermore, global sparsity was achieved since at all time the same two joints, namely 5 and 6 (see Fig. 3 for the joint convention), move during the entire trajectory.

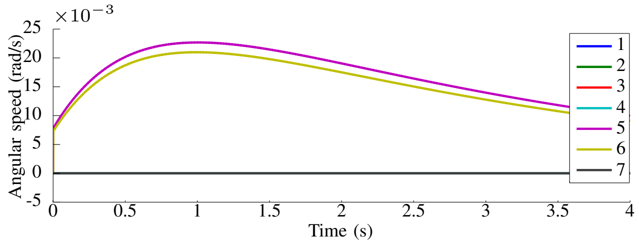
On the other hand, in the pseudoinverse approach *all* joints move, as expected. Some of them moved very little, so if one considers as "effectively moving" only those joints that have a velocity above 5% of the maximum velocity in that given instant, then during the whole time only seven joints effectively moved.

Table II shows the comparison of the demanded input $\dot{\mathbf{q}}$ according to four different metric for both approaches. As before, the error dynamics are exactly the same for both approaches, and therefore the comparison is fair.

It is interesting to note that in the case of LP, the two joints that move (5 and 6, see Fig. 3) are the roll and pitch of the ankle of the right feet, the one on the ground.



(a) PINV approach (joints 22 to 25 were omitted).



(b) LP approach: (joints 8 to 25 were omitted).

Fig. 5: PINV vs. LP: demanded velocities for each joint; omitted joints do not contribute to the motion (see Fig. 3 for the joints convention).

TABLE II: Experiment with HOAP-3: comparison of the input with respect to four different metrics.

Metric	PINV	LP	PINV/LP
$\int_0^\infty \|\dot{\mathbf{q}}(t)\|_1 dt$	0.185	0.131	1.412
$\sqrt{\int_0^\infty \|\dot{\mathbf{q}}(t)\|_2^2 dt}$	0.040	0.067	0.597
$\int_0^\infty \|\ddot{\mathbf{q}}(t)\|_1 dt$	0.101	0.072	1.400
$\sqrt{\int_0^\infty \|\ddot{\mathbf{q}}(t)\|_2^2 dt}$	0.289	0.339	0.852

This agrees with the human movement observed in postural control. Indeed, it has been shown by [26] that in the case of a small disturbance, the postural system will recover balance in the sagittal plane by using mainly the ankle joints.

Finally, it is important to highlight that, although the joints trajectories were generated offline, it is feasible to generate them in real time. In order to support this claim, numerical simulations were done in an Intel i5 2.4 GHz with 4 GB of RAM. Among 500 samples, the pseudoinverse took in average 0.2 milliseconds to be computed, whereas each linear program with Simplex took in average 1.2 milliseconds to be solved. If warm start is used in the Simplex (the solution found in the previous step is used as a feasible basis in the current one), the time in Simplex also reduces to 0.2 milliseconds in average, because in the huge majority of cases the initial basis is primal and dual feasible (thus both feasible and optimal) and only part of the first iteration is necessary in the Simplex method. This happens because the problem has a continuous nature and thus at each step the parameters of the optimization problem change little.

Given that the processor of HOAP-3 is capable of using the pseudoinverse approach in real time with a wide margin (see [27]), the timings presented here imply that it could also be capable of using the LP approach, specially if warm start is implemented.

IV. Conclusion and future developments

This paper proposed a new paradigm to control highly redundant robots. The technique is based on a linear programming formulation, which when solved using the Simplex algorithm intrinsically reduces the number of non-zero components in the control vector. Formal results of stability were presented, and two examples—one in simulation and the other in a real humanoid robot—were implemented to illustrate the methodology.

The natural step for next works is the generalization of the proposed approach to hierarchy of tasks, as in [3], [5], [6], [7], among others. In that case, the 1-norm is attractive because the associated optimization problems—i.e., lexicographical linear programs—can be solved by a very simple variant of the Simplex algorithm, as pointed out in [28]. Furthermore, since the Simplex algorithm is very well understood and efficient, the lexicographical Simplex is also expected to be efficient because it can borrow many advances from its traditional counterpart. However, a theoretical challenge would be to prove stability for the hierarchical case, which would consist of a generalization of Proposition 1.

Finally, it turns out that Proposition 1 can also be generalized to encompass tasks in which it is not only desirable to converge to a set of configurations (e.g., a circle), but also to keep forever circulating in it [29]. This self sustained motion—a limit cycle in the configuration space—could be used to accomplish, for instance, a dance-like motion or even walking [30].

References

[1] C. Fitzgerald, "Developing baxter," in *2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA)*. IEEE, Apr. 2013, pp. 1–6.

[2] A. Liégeois, "Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7, no. 12, pp. 868–871, 1977.

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

[4] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 3, pp. 398–410, Jun. 1997.

[5] L. F. Kanoun, O. and P.-B. Wieber, "Kinematic control of redundant manipulators: generalizing the task priority framework to inequality tasks," *IEEE Trans. on Robotics*, vol. 27, no. 4, pp. 785–792, 2011.

[6] M. I. De Lasa, M. and A. Hertzmann, "Feature-based locomotion controllers." in *ACM SIGGRAPH 10.*, 2010.

[7] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *International Journal of Robotics Research*, vol. 33, no. 7, pp. 1006–1028, 2014.

[8] J.-P. Gauthier, B. Berret, and F. Jean, "A biomechanical inactivation principle," *Proceedings of the Steklov Institute of Mathematics*, vol. 268, no. 1, pp. 93–116, Apr. 2010.

[9] Y. Mehrdad, "Fast Human Movements and Sparse Optimal Control Policies," PhD Dissertation, University of California, San Diego, 2012.

[10] M. Nagahara, D. Quevedo, and D. Nesic, "Maximum Hands-Off Control: A Paradigm of Control Effort Minimization," *IEEE Transactions on Automatic Control*, vol. 11, no. 4, pp. 1–1, 2015.

[11] M. Gallieri and J. Maciejowski, "Lasso mpc: Smart regulation of over-actuated systems," in *American Control Conference (ACC), 2012*, June 2012, pp. 1217–1222.

[12] R. P. Aguilera, R. Delgado, D. Dolz, and J. C. Aguero, "Quadratic mpc with l0-input constraint," in *19th World Congress The International Federation of Automatic Control, 2014*, August 2014.

[13] D. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 5, no. 52, pp. 1289 – 1306, 2006.

[14] S. Muthukrishnan, "Data streams: Algorithms and applications," *Found. Trends Theor. Comput. Sci.*, vol. 1, no. 2, pp. 117–236, Aug. 2005.

[15] R. Shamir, "The efficiency of the simplex method: A survey," *Management Science*, vol. 33, no. 3, pp. pp. 301–334, 1987.

[16] E. S. L. Ho, T. Komura, and R. W. H. Lau, "Computing inverse kinematics with linear programming," in *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, 2005, pp. 163–166.

[17] W. Decre, R. Smits, H. Bruyninckx, and J. De Schutter, "Extending itasc to support inequality constraints and non-instantaneous task specification," in *IEEE International Conference on Robotics and Automation*, May 2009, pp. 964–971.

[18] C. Samson, M. Leborgne, and B. Espiau, "Robot control. the task-function approach." Oxford University Press, 1991.

[19] K. G. Murty, *Linear Programming*. John Wiley & Sons, 1983.

[20] H. Li, "Solve least absolute value regression problems using modified goal programming techniques," *Computers & OR*, vol. 25, no. 12, pp. 1137–1143, 1998.

[21] R. K. Sundaram, *A first course in optimization theory*. Cambridge Univ. Press, 1996.

[22] E. Asplund and R. Rockafellar, "Gradients of convex functions," *Transactions of the American Mathematical Society*, vol. 139, pp. 443–467, 1969.

[23] H. Nakamura, Y. Yamashita, H. Nishitani, and H. Yamamoto, "Discontinuous control of nonholonomic systems using nondifferentiable Lyapunov functions," in *SICE 2003 Annual Conference*, vol. 2, 2003, pp. 1415–1418.

[24] R. J. Vanderbei, *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, 1997.

[25] G. Golub and C. Van Loan, *Matrix Computations*, ser. Johns Hopkins studies in the mathematical sciences. Johns Hopkins University Press.

[26] L. M. Nashner and G. McCollum, "The organization of human postural movements: A formal basis and experimental synthesis," *Behavioral and Brain Sciences*, vol. 8, pp. 135–150, 3 1985.

[27] B. V. Adorno, A. P. L. Bó, P. Fraisse, and P. Poignet, "Towards a cooperative framework for interactive manipulation involving a human and a humanoid," in *IEEE International Conference on Robotics and Automation*. IEEE, May 2011, pp. 3777–3783.

[28] H. Isermann, "Linear lexicographic optimization," *Operations-Research-Spektrum*, vol. 4, no. 4, pp. 223–228, 1982.

[29] V. Goncalves, L. A. Pimenta, C. A. Maia, B. C. O. Dutra, and G. A. S. Pereira, "Vector fields for robot navigation along time-varying curves in n -dimensions," *Robotics, IEEE Transactions on*, vol. 26, no. 4, pp. 647–659, Aug 2010.

[30] D. G. E. Hobbelen and M. Wisse, *Limit Cycle Walking. Humanoid Robots: Human-like Machines*, ser. Systems & control. ITech Education and Publishing, 2007.