Stable-by-Design Kinematic Control Based on Optimization

Vinicius Mariano Gonçalves, Bruno Vilhena Adorno, André Crosnier and Philippe Fraisse

***Abstract*—This paper presents a new kinematic control paradigm for redundant robots based on optimization. The general approach takes into account convex objective functions with inequality constraints and a specific equality constraint resulting from a Lyapunov function, which ensures closed-loop stability by design. Furthermore, we tackle an important particular case by using a convex combination of quadratic and l_1-norm objective functions, making possible for the designer to choose different degrees of sparseness and smoothness in the control inputs. We provide a pseudo-analytical solution to this optimization problem and validate the approach by controlling the center of mass of the humanoid robot HOAP3.**

I. Introduction

MANY practical problems in robot motion control involve some kind of constraints. For instance, a robot manipulator usually has joints limits and its actuators have saturation limits [1]; in robotic surgery the workspace is very constrained and the robot must avoid sensitive areas and respect anatomy-based constraints [2], [3]; in order to keep balance, humanoids must maintain their zero moment point (ZMP) inside the support polygon [4]; and fixed-wing unmanned aerial vehicles must have a minimum forward velocity in order to fly [5].

In order to deal with constraints directly in the motion control law, Liégeois's early work [6] has introduced a task-priority framework that takes into account two hierarchical levels, where the one with higher priority corresponds to the main task (e.g., control of the end-effector pose) and the lowest level is used to impose equality constraints. This way, inequality constraints such as joint limits can be transformed into equality constraints by means of a gradient projection method; however, the constraints are not respected if they come into conflict with the main task [7]. This hierarchical structure has been extended for an arbitrary number of tasks [8] and a scheme for inserting and removing tasks in order to enforce an arbitrary number of secondary equality constraints, called Stack of Tasks, has been proposed [9]. The Stack of Tasks has been extended to take into account inequality constraints [10], but according to Kanoun et al. [11] this formalism has exponential cost in the number of inequatilies.

Other approach to deal explicitly with both equality and inequality constraints is optimization-based control. The great advantages of using such approach are that both equality and inequality constraints are easily included in the problem formulation and it can be implemented in real time.

Owing to these advantages, several works have proposed motion control techniques based on optimization. Faverjon and Tournassoud [12] are one of the pioneers of this technique applied to robotics, and they have proposed quadratic programming to generate the robot control inputs, where inequality constraints are used to implement collision avoidance. Other works have extended quadratic programming to a hierarchical structure [13], [11], [14] and there is also a recent application that aims at optimizing the robot manipulability, based on a quadratic programming formulation [15], in the context of inverse kinematic control for redundant robots.

Instead of using quadratic programming, other works used linear programming to solve the problem of real-time inverse kinematics [16], [17] but usually they are not concerned with closed loop stability and in some cases the optimization problem formulation is not suitable for the application. For instance, in the linear program proposed by Ho et al. [16], the differential kinematics is used as an equality constraint; therefore, the problem is feasible only if the task-space velocities are in the range space of the Jacobian matrix, which is not always possible if the robot is underactuated or in a singular configuration. In addition, Berthet-Rayne et al. [18] use sparse kinematic control laws, based on linear and quadratic programming formulations, to teleoperate a redundant snakelike robot in minimally invasive surgery.

One of the major drawbacks of those optimization-based motion control techniques is that very few are concerned with closed-loop stability, with some notable exceptions. For instance, Al Khudir et al. [19] propose an optimization-based controller and rely on experimental evaluations to claim that the closed-loop system achieves stable and consistent behaviors but it does not present a formal proof of closed-loop stability. Stability, however, refers to an infinite number of scenarios that cannot be tested experimentally and it needs theoretical proof or must be ensured by design. Realizing the importance of formally ensuring closed-loop stability, Escande et al. [14] analyzes the Lyapunov stability of hierarchical systems and Scianca et al. [20] have designed an intrinsically-stable model predictive controller for gait generation. The latter, however, proposes an application-specific constraint, which ensures that the robot center of mass trajectory is stable, and consequently cannot be extended to generic tasks. Gonçalves et. al [21] have proposed an efficient formulation for the kinematic control based on linear programming whose solution is always feasible, there is formal proof of closed-loop stability, but it depends on the solver (i.e., Simplex) to obtain sparse solutions.

On the one hand, the choice of linear programming over quadratic programming in robotics applications is often justi-

Vinicius Mariano Gonçalves (mariano@cpdee.ufmg.br) and Bruno Vilhena Adorno (adorno@ufmg.br) are with the Department of Electrical Engineering of the Federal University of Minas Gerais (UFMG), Brazil.

André Crosnier (crosnier@lirmm.fr) and Philippe Fraisse (fraisse@lirmm.fr) are with LIRMM UMR 5506 CNRS, University of Montpellier, Montpellier, France.

This work was supported by CAPES, CNPq (grant numbers 424011/2016-6 and 303901/2018-7), and FAPEMIG (grant number APQ-02144-18).

fied by computational efficiency or, in some special cases (i.e., when specific solvers are used), because of its sparse solutions. On the other hand, control inputs generated by linear programs tend to be less smooth than the ones generated by quadratic programs.

In order to obtain the best of both worlds, this paper presents a general formulation that takes into account a general positive objective function, which can be specialized to a quadratic convex optimization problem and hence the designer can choose between different degrees of sparseness and, consequently, different degrees of smoothness in the control inputs. Moreover, we propose a special equality constraint on the control inputs, which ensures closed-loop stability by design for general applications.

This paper is organized as follows: Section II presents the problem definition and the overview of the solution, whereas Section III presents a thorough analysis on important aspects such as solution feasibility, properties arising from different choices of objective functions, and Lyapunov constraints. Section IV presents a particular, but very useful, optimization-based controller that enables tuning the control input sparseness and its corresponding pseudo-analytical solution. Section V presents the experimental results and Section VI presents the conclusions and final remarks.

II. Problem definition and overview of the solution

Let $\mathcal{Q} \subseteq \mathbb{R}^n$ be a smooth manifold representing the robot configuration space and $q \in \mathcal{Q}$ be the robot configuration. Assume that the control inputs are the configuration velocities; that is,

$$\dot{q} = u. \tag{1}$$

The goal is to design the control input $u(q)$ to guide q to a subset $\mathcal{Q}_{tg} \subset \mathcal{Q}$, which represents the completeness of a given task (e.g., desired end-effector pose), while considering equality and inequality constraints on the trajectory and ensuring closed-loop stability. Those constraints are defined to ensure, for instance, maximum and minimum configuration velocities, obstacle avoidance, and avoidance of joint limits.

The target set is defined as $\mathcal{Q}_{tg} \triangleq \{q : e(q) = 0\}$, where $e : \mathcal{Q} \to \mathbb{R}^m$ is a differentiable task function [22], which represents an error between desired and current variables. For instance, if $h(q)$ is the current end-effector pose and h_d is the desired end-effector pose, the task function can be defined as $e(q) \triangleq h(q) - h_d$. This way, when $e(q) = 0$ the end-effector pose is at the desired set-point and the set $\mathcal{Q}_{tg}$ represents all configurations that fulfill the task.

A. Overview of the solution

In order to design the control law that fulfills the aforementioned requirements, we first define a differentiable Lyapunov function $V : \mathcal{Q} \to [0, \infty)$ to the set $\mathcal{Q}_{tg}$ that satisfies

$$\xi_m\left(d\left(q, \mathcal{Q}_{tg}\right)\right) \leq V(q) \leq \xi_M\left(d\left(q, \mathcal{Q}_{tg}\right)\right), \tag{2}$$

where $\xi_m, \xi_M : [0, \infty) \to [0, \infty)$ are non-decreasing positive definite functions and $d(x, \mathcal{X}) \triangleq \min_{\overline{x} \in \mathcal{X}} \|x - \overline{x}\|_2$ is the distance function between a point $x \in \mathbb{R}^n$ and a closed subset $\mathcal{X}$ of $\mathbb{R}^n$.

Consider a function $\Phi : \mathbb{R}^n \times \mathcal{Q} \to [0, \infty)$ that quantifies how good a control input is at configuration q according to a relevant metric (e.g., $\|u\|_2^2$ and $\|u\|_1$). The optimal control input u^* is obtained as

$$\begin{aligned} u^* \in \underset{u}{\operatorname{argmin}} \quad & \Phi(u, q) \\ \text{subject to} \quad & A(q)\, u \leq b(q) \\ & \nabla V(q)^T u = -\rho \Psi(q), \end{aligned} \tag{3}$$

where $A : \mathcal{Q} \to \mathbb{R}^{s \times n}$ and $b : \mathcal{Q} \to \mathbb{R}^s$ define the inequality constraints in the control inputs—which can be used to enforce joint limits, bounds on joint velocities, etc.—, $\rho \in [0, 1]$ is a variable used to ensure feasibility of the optimization problem when the Lyapunov and inequality constraints are incompatible, and $\Psi : \mathcal{Q} \to [0, \infty)$ is a positive definite function that defines how fast the system stabilizes. More specifically, $\dot{V}(q) = \nabla V(q)^T u = -\rho \Psi(q) \leq 0$ ensures *stability by design* of the closed loop $\dot{q} = u^*(q)$.

The control law (3) arises some important questions:

1) How do the choice of the function Φ, the Lyapunov function V, and the function Ψ affect the closed-loop behavior?
2) What are the appropriate values for ρ in order to ensure that the solution to this problem is always feasible?
3) How do we choose A and b to enforce relevant constraints?

We address the first two questions in Section III and the third one, although already discussed in the literature, is briefly analyzed in Section III-E from a more theoretical point of view.

The proposed approach aims to solve, simultaneously, three important problems in robotics: inverse kinematics, local motion planning, and stable closed-loop control. *Inverse kinematics* because the configuration q evolves toward the set $\mathcal{Q}_{tg}$, which is not known explicitly, as $e(q)$ approaches 0. *Local motion planning* because we can specify, using the constraints $A(q)u \leq b(q)$, movement constraints such as to perform, for instance, obstacle avoidance. Finally, *stable closed-loop control* because we specify the control inputs that generate the trajectory related to the task.

It is important to note, however, that convergence to $\mathcal{Q}_{tg}$ is not guaranteed with this approach, since it uses only local information of the function V. Therefore, the optimal control input $u^*(q)$ in (3) can guide, eventually, the system to a local minimum of $V(q)$ such that $e(q) \neq 0$ (although we ensure that $\dot{e}(q) = 0$), which can be aggravated by the constraints $A(q)\, u \leq b(q)$. In other words, our approach ensures stability but not necessarily asymptotic stability. Consequently, when this local approach is not able to solve the proposed task, we may need to consider strategies that take into account the global structure of the configuration space and $V(q)$, like motion planning algorithms. However, global motion planning is out of the scope of this paper.

III. Analysis

Given the problem definition in Section II and the proposed solution in Section II-A, we analize the control law (3) in terms of feasibility, relevant objective functions, constraints implementation, and the function Ψ that determines overall convergence behavior.

A. Feasibility

Problem (3) can be unfeasible due to (*i*) the inequality constraint $A(q)u \leq b(q)$, (*ii*) the equality constraint $\nabla V(q)^T u = -\rho\Psi(q)$—which we call Lyapunov specification and specifies the convergence behavior—, (*iii*) or a combination of the two.

In most practical cases, it is reasonable to assume that the set $\{u \in \mathbb{R}^n : A(q)\, u \leq b(q)\}$ is non-empty as long as $q(0)$ does not violate any constraint, as shown in Section III-E, therefore we can rule out (*i*).

To ensure the feasibility of the Lyapunov specification (i.e., finite control inputs), it suffices to impose some conditions on the positive definite function $\Psi(q)$. First, when $\|\nabla V(q)\|_2 = 0$ then $\Psi(q) = 0$; in addition, since $\nabla V(q)^T u = -\rho\Psi(q)$ implies $\|u\|_2 \geq \rho\Psi(q) / \|\nabla V(q)\|_2$, then a necessary condition for a bounded control input u^* is

$$\lim_{q \to q^*} \frac{\rho\Psi(q)}{\|\nabla V(q)\|_2} \neq \infty, \quad \forall q^* \in \{\bar{q} : \nabla V(\bar{q}) = 0\}. \tag{4}$$

For instance,

$$u = -\frac{\rho\Psi(q)}{\|\nabla V(q)\|_2} \frac{\nabla V(q)}{\|\nabla V(q)\|_2} \tag{5}$$

is a feasible solution, which is always bounded, owing to (4). Therefore, we can also rule out (*ii*) by choosing $\Psi(q)$ appropriately.

The final concern is when the constraints are feasible when considered separately but are conflicting between them. Since the Lyapunov constraint $\nabla V(q)^T u = -\rho\Psi(q)$ determines how fast the closed-loop system stabilizes, it may conflict with the system capabilities and the constraints described by $A(q)u \leq b(q)$. One natural idea is to choose an appropriate value for ρ to prevent this conflict. To find the set of all ρ that guarantee compatibility between the Lyapunov specification and the inequality constraint in (3), we solve

$$\begin{aligned} (v^*, \rho^*) \in \underset{v,\rho}{\operatorname{argmax}} \quad & \rho \\ \text{subject to} \quad & A(q)v \leq b(q) \\ & \nabla V(q)^T v = -\rho\Psi(q). \end{aligned} \tag{6}$$

By letting $\rho = -\nabla V(q)^T v / \Psi(q)$, which is always defined owing to (4), and using the fact that $\Psi(q) \geq 0$, (6) is equivalent to

$$\begin{aligned} v^* \in \underset{v}{\operatorname{argmin}} \quad & \nabla V(q)^T v \\ \text{subject to} \quad & A(q)v \leq b(q). \end{aligned} \tag{7}$$

Problems (6) and (7) admit a feasible, but not necessarily optimal, solution $\rho = 0$ whenever $v = 0$ is feasible for $A(q)v \leq b(q)$, that is, $b(q) \geq 0$. This is a not very restrictive assumption for a wide class of robots, as shown in Section III-E.[1] Therefore, we guarantee that Problem (6) is feasible and there exists an optimal solution $\rho^* \geq 0$. If $\rho^* = 0$ then $\dot{V}(q) = \nabla V(q)^T u = 0$, meaning that the system reached a local minimum, which may or may not be the desired global one. This local minimum is not necessarily a local minimum of V, because the inequality constraints also need to be taken into consideration as they can create other local minima that are not minima of V. For instance, if $A(q)u \leq b(q)$ is satisfied only if $u = 0$, for all $q \in \mathcal{Q}$, then the entire set $\mathcal{Q}$ is a local minimum even if $\nabla V(q) \neq 0$.

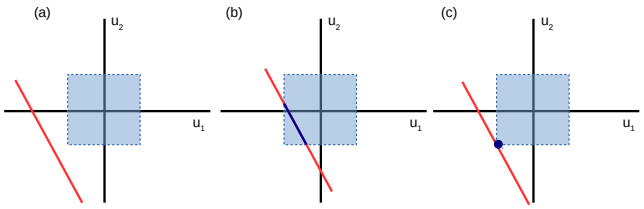


Figure 1: Effect of inequality constraints $Au \leq b$ (blue square) and the Lyapunov constraint $\nabla V^T u = -\rho\Psi$ (red line): (*a*) if $\rho > \rho^*$ then the constraints are conflicting and the problem is unfeasible; (*b*) when $\rho < \rho^*$ the red line is inside the square, therefore the dark blue line is the new feasible set; (*c*) when $\rho = \rho^*$ there is only one feasible solution (the blue point in the corner), except in the very specific case where the red line is parallel to its closest edge on the polygon defined by the constraint $Au \leq b$.

After obtaining the maximal ρ, namely

$$\rho^*(q) = -\frac{\nabla V(q)^T v^*}{\Psi(q)},$$

we choose $\rho = \min\{\rho^*(q), 1\}$ to guarantee feasibility of the control law (3) while respecting the specification $\Psi(q)$ as best as possible. Then we solve (3) with this chosen value ρ to obtain the optimal input u^*.

Two important things must be taken into consideration. First, the variable v^* obtained in (6) is a *dummy velocity* associated with the largest ρ^* possible at that moment. Therefore, if we choose $u = v^*$, the objective function $\Phi(u, q)$ would be completely disregarded and the Lyapunov function would have the *greatest* descent at that moment. In case $\rho^*(q) > 1$, choosing $\rho = \rho^*$ makes the function $\Psi(q)$ ineffective, because (3) may have as feasible set only the single point $u = v^*$, as shown in Fig. 1.

Second, we do not need to solve (6) at each control period. Once we calculate the relaxation variable ρ, we can use it as long as the optimization problem (3) is feasible. If not, then we need to obtain a new relaxation by solving (6).

Remark 1. Alternatively, the equality constraint in (3) can be replaced by the inequality constraint $\nabla V(q)^T u \leq 0$. The latter always ensure feasible closed-loop stability, as it implies $\dot{V}(q) \leq 0$ and $u = 0$ is a feasible solution, but it does not allow the designer to fine tune the closed-loop behavior.

[1] This first analysis assumes the set-point regulation case, but we further extend it to the tracking of time-varying trajectories, as well as for the case where $v = 0$ is not feasible, in Section III-D.

B. The function Φ

The function $\Phi(u, q)$ provides a measure of how good is the control input u at a given configuration q. One interesting choice is the family of l_p-norms[2] of u, that is, $\Phi(u, q) = \|u\|_p$ for a real number $p > 0$. In this case, we penalize large control inputs, but different choices of p induce different behaviors. In a nutshell, as p increases from 0 to ∞, we shift from *sparse* control inputs to *even* control inputs. Sparser control inputs result in more zero entries in the control input vector u while fullfiling the constraints and minimizing the magnitude of u (according to the chosen l_p-norm).[3] This behavior is strongly present when the l_0-norm is used (i.e., $\lim_{p\to 0} \|u\|_p$). More specifically, optimizing $\|u\|_p$ is equivalent to optimizing $\|u\|_p^p$ and

$$\lim_{p\to 0} \|u\|_p^p = \lim_{p\to 0} \sum_i |u_i|^p = \#\{i : u_i \neq 0\},$$

where $\#\mathcal{A}$ denotes the cardinality of set $\mathcal{A}$; therefore, taking into account all constraints, we obtain $\min \lim_{p\to 0} \|u\|_p^p = \min \#\{i : u_i \neq 0\}$. This way, the control input vector u will have as few nonzero entries as possible.

On the other hand, *even* control inputs results in a control input vector u distributed as evenly as possible in magnitude (discarding signs) while keeping the constraints and minimizing the magnitude of u. This behavior is strongly present when the l_∞-norm is used (i.e., $\lim_{p\to\infty} \|u\|_p$). Since

$$\|u\|_\infty = \max_i |u_i|, \tag{8}$$

if the vector coefficients u_i are not evenly distributed in absolute value—for instance, $\exists j$ such that $u_j > u_i$ for all $i \neq j$—the objective function is decreased by reducing u_j and redistributing the remaining value to the other coefficients. Due to the saturating nature of the $\max$ function, the norm will decrease, as long as all constraints are fulfilled. The choice of this norm for solving differential inverse kinematics has already been studied and is called a *minimum effort* approach [23]. In a more general context, solutions to underdetermined linear systems that have maximum possible evenness have also been called *democratic* [24].

Example 2. Consider the optimization problem

$$\begin{aligned} \min_x \quad & \|x\|_p \\ \text{subject to} \quad & \sum_{i=1}^n a_i x_i = b, \end{aligned}$$

where $a_i \neq 0$, $\forall i$ and $|a_i| > |a_j|$ $\forall i < j$. Applying the Karush-Kuhn-Tucker conditions, the analytical solution to this problem is

$$x_i = b \left(\frac{\operatorname{sgn}(a_i)|a_i|^{1/(p-1)}}{\sum_{j=1}^n |a_j|^{p/(p-1)}} \right) \tag{9}$$

when $1 < p < \infty$ (convex case) and $x_1 = b/a_1$ and $x_i = 0$, $\forall i \neq 1$, when $0 < p \leq 1$. In addition, the function $\operatorname{sgn} : \mathbb{R} \to \{-1, 0, 1\}$ is defined for scalars as

$$\operatorname{sgn}(a) = \begin{cases} -1 & a < 0 \\ 0 & a = 0 \\ 1 & a > 0 \end{cases}$$

and extended to vectors by applying it component-wise.

Since both solutions for $p \to 1^+$ and $p \to 1^-$ agree, then for $0 < p \leq 1$ the solution is sparse and the only non-null entry x_i is the one that corresponds to the largest coefficient in absolute value. On the other hand, when $p \to \infty$, the solution is $x = \operatorname{sgn}(a)b/\|a\|_1$; that is, all $|x_i| = |x_j|$, $\forall i, j$, which results in the most even situation possible in terms of the absolute value. □

Norms between l_0 and l_∞ induce these behaviors in varying degrees. The l_0-norm is inconvenient because is nonconvex, and thus yields difficult optimization problems. Indeed, optimizing the l_0-norm, even under linear constraints, is NP-hard [25]. The l_∞-norm, although convex, has the inconvenience of being non-differentiable and cumbersome to handle analytically. Thus, in practice, it is convenient to consider the sparseness induced by the l_1-norm and the evenness induced by the l_2-norm. Section IV presents the analytical solution to an optimization problem composed of a weighted sum of those two norms.

C. The function Ψ

Since the function Ψ in (3) influences the decreasing rate of the Lyapunov function V, it also determines how fast the task is achieved. There are several possibilities for choosing Ψ, as long as the conditions $\|\nabla V(q)\|_2 = 0 \implies \Psi(q) = 0$ and (4) are fulfilled, as shown in Section III-A. Whereas the latter is necessary to generate bounded control inputs, the former ensures that $u = 0$ is a valid control input when the robot reaches a local minimum (i.e., $\nabla V(q) = 0$).

1) Design of Ψ based on desired closed-loop behavior: Suppose we desire a closed-loop behavior given by $\dot{V}(q) = -\eta' V(q)$, where $\eta' \in (0, \infty)$. The goal is to find a function $\Psi(q)$ that enforces this behavior, at least in most regions of the configuration space. Let us choose $\Psi(q) = \eta V(q) \tanh(\kappa \|\nabla V(q)\|_2)$ with $\kappa, \eta \in (0, \infty)$. This function fulfills both requirements as $\|\nabla V(q)\|_2 = 0$ implies $\Psi(q) = \eta V(q) \tanh(0) = 0$ and, letting $\Gamma \triangleq \|\nabla V(q)\|_2$, we obtain[4]

$$\begin{aligned} \lim_{\Gamma\to 0} \frac{\rho\Psi(q)}{\Gamma} &= \lim_{\Gamma\to 0} \frac{\rho\eta V(q) \tanh(\kappa\Gamma)}{\Gamma} \\ &= \lim_{\Gamma\to 0} \rho\eta V(q)\kappa \operatorname{sech}^2(\kappa\Gamma) \\ &= \rho\eta V(q)\kappa < \infty. \end{aligned}$$

By choosing an appropriate value κ, when far from local minima $\tanh(\kappa\|\nabla V(q)\|_2) \to 1$, hence $\Psi(q) \to \eta V(q)$. Thus, let $\eta' \triangleq \rho\eta$ to obtain

$$\dot{V}(q) = \nabla V(q)^T u = -\rho\Psi(q) = -\eta' V(q),$$

[2]The l_p-norm $\|x\|_p$ of a vector x, with $0 < p < \infty$, is given by $\|x\|_p = \left(\sum_i |x_i|^p\right)^{1/p}$.

[3]This is related to the parsimonious behavior in kinematic control discussed in our previous work [21]. That work required the use of the simplex algorithm to achieve sparse behavior, whereas the method presented in this paper is independent of the algorithm used to solve the optimization problem.

[4]We use L'Hôpital's rule and the fact that $\frac{d}{d\Gamma} \tanh(\kappa\Gamma) = \kappa \operatorname{sech}^2(\kappa\Gamma)$.

which means that the system will converge at exponential rate.

2) Design of Ψ based on Φ to enforce actuation limits: We can use Ψ to enforce actuation limits (i.e., bounded configuration velocities) by means of its explicit relationship with the objective function Φ. For example, let us consider the objective function

$$\Phi(u) = \gamma\|u\|_1 + (1-\gamma)\frac{1}{2}\|u\|_2^2,$$

which, as shown in Section IV, has very interesting properties. Since (5) is a feasible solution, which we denote as u_{feas}, and u^* is the optimal one, then

$$\gamma\|u^*\|_1 + \frac{(1-\gamma)}{2}\|u^*\|_2^2 \leq \gamma\|u_{\text{feas}}\|_1 + \frac{(1-\gamma)}{2}\|u_{\text{feas}}\|_2^2 \quad (10)$$

as the objective function is nondecreasing. Substituting (5) in (10), we obtain

$$\gamma\|u^*\|_1 + \frac{(1-\gamma)}{2}\|u^*\|_2^2 \leq \gamma\frac{\|\nabla V(q)\|_1}{\|\nabla V(q)\|_2^2}\rho\Psi(q) + \frac{(1-\gamma)}{2}\frac{\rho^2\Psi(q)^2}{\|\nabla V(q)\|_2^2}. \quad (11)$$

Since $\|x\|_\infty \leq \|x\|_1$ and $\|x\|_\infty \leq \|x\|_2$, $\forall x$, then

$$\gamma\|u^*\|_\infty + \frac{(1-\gamma)}{2}\|u^*\|_\infty^2 \leq \gamma\|u^*\|_1 + \frac{(1-\gamma)}{2}\|u^*\|_2^2 \quad (12)$$

In addition, $\|x\|_1 \leq \sqrt{n}\|x\|_2$, $\forall x$,[5] in which $n \in \mathbb{N}$ is the size of x, and $y \leq ny$, $\forall y \in [0, \infty)$; thus, by using the right-hand term in (11), we obtain

$$\gamma\frac{\|\nabla V(q)\|_1}{\|\nabla V(q)\|_2^2}\rho\Psi(q) + \frac{(1-\gamma)}{2}\frac{\rho^2\Psi(q)^2}{\|\nabla V(q)\|_2^2} \leq \gamma\sqrt{n}\frac{\rho\Psi(q)}{\|\nabla V(q)\|_2} + n\frac{(1-\gamma)}{2}\frac{\rho^2\Psi(q)^2}{\|\nabla V(q)\|_2^2}. \quad (13)$$

Using (11), (12) and (13), yields

$$\gamma\|u^*\|_\infty + \frac{(1-\gamma)}{2}\|u^*\|_\infty^2 \leq \gamma\sqrt{n}\frac{\rho\Psi(q)}{\|\nabla V(q)\|_2} + n\frac{(1-\gamma)}{2}\frac{\rho^2\Psi(q)^2}{\|\nabla V(q)\|_2^2}.$$

Let $F(x) \triangleq \gamma x + (1/2)(1-\gamma)x^2$, thus

$$F(\|u^*\|_\infty) \leq F\left(\frac{\sqrt{n}\rho\Psi(q)}{\|\nabla V(q)\|_2}\right). \quad (14)$$

As the function F is increasing in the interval $[0, \infty)$, the previous inequality reduces to

$$\|u^*\|_\infty \leq \frac{\sqrt{n}\rho\Psi(q)}{\|\nabla V(q)\|_2}. \quad (15)$$

Therefore, let us choose

$$\Psi(q) \triangleq \frac{u_{\max}}{\sqrt{n}}\|\nabla V(q)\|_2 R(q), \quad (16)$$

where $u_{\max}$ is the desired upper bound for the configuration velocity and $R(q) : \mathcal{Q} \to [0, 1]$. Thus

$$\|u^*\|_\infty \leq u_{\max}\rho R(q) \leq u_{\max}, \quad (17)$$

which implies that the optimal control input u^* has all its entries bounded by $u_{\max}$.

D. Handling non-null velocity inputs

In Section III-A, we assumed that the control input $u = 0$ must belong to the set of admissible solutions to ease the feasibility analysis, but this assumption is not strictly necessary to ensure the feasibility of Problem 3 and can be relaxed, under mild considerations, to handle time-varying trajectories or persistent motions.

1) Time-varying constraints: To handle time-varying tasks, which are encoded in $V(q, t)$, it is necessary to add a feedforward term in the Lyapunov constraint in Problem 3. More specifically, since we want to induce stability by design, we enforce the constraint $\dot{V}(q(t), t) = -\rho\Psi(q(t), t)$, which yields[6]

$$\nabla_q V(q, t)^T u + \frac{\partial V}{\partial t}(q, t) = -\rho\Psi(q, t), \quad (18)$$

where $\partial V(q, t)/\partial t$ is the feedforward term.

In addition, to guarantee that a bounded solution always exists, it is necessary that

$$\lim_{(q,t)\to(q^*,t^*)} \frac{\left|\rho\Psi(q, t) + \frac{\partial V}{\partial t}(q, t)\right|}{\|\nabla_q V(q, t)\|_2} \neq \infty, \quad \forall (q^*, t^*) \in \Omega_S, \quad (19)$$

where $\Omega_S = \{(\bar{q}, \bar{t}) : \nabla_q V(\bar{q}, \bar{t}) = 0\}$. From Condition 4, $|\rho\Psi(q, t)|/\|\nabla_q V(q, t)\|_2$ is required to be bounded for all $(q^*, t^*) \in \Omega_S$; therefore, by using the triangle inequality, Condition 19 is satisfied if

$$\lim_{(q,t)\to(q^*,t^*)} \frac{\left|\frac{\partial V}{\partial t}(q, t)\right|}{\|\nabla_q V(q, t)\|_2} \neq \infty. \quad (20)$$

2) Persistent motion: Suppose we want to make the system converge to a one-dimensional curve $\mathcal{C}$ in the configuration space $\mathbb{R}^n$ and circulate it in a given direction. This curve can be defined by $n-1$ level surfaces of differentiable scalar functions, $\alpha_i : \mathbb{R}^n \to \mathbb{R}$, as [26]

$$\mathcal{C} \triangleq \{q \in \mathbb{R}^n \ : \ \alpha_i(q) = 0, \ i = 1, 2, \ldots, n-1\}, \quad (21)$$

where the gradients $\nabla_q \alpha_i(q)$ are linearly independent for all q. If we consider the vector $T(q) \in \mathbb{R}^n$ in the tangent space of the curve $\mathcal{C}$ at point q, such that $\|T(q)\|_2 \neq 0$ when $q \in \mathcal{C}$, then the system of $n-1$ linearly-independent equations

$$\nabla_q \alpha_i(q)^T T(q) = 0, \ \forall i \in \{1, \ldots, n-1\} \quad (22)$$

have infinite non-null solutions for $T(q)$, but if we impose the additional constraint $\|T(q)\|_2 = 1$, then there are only

[5] Recall that, using the Cauchy–Schwarz inequality, $\|x\|_1 = \sum_{i=1}^n |x_i| = \sum_{i=1}^n |x_i| \cdot 1 \leq \left(\sum_{i=1}^n |x_i|^2\right)^{1/2}\left(\sum_{i=1}^n |1|^2\right)^{1/2} = \|x\|_2\sqrt{n}$.

[6] The constraint (18) can be written as $\nabla_q V(q, t)^T u = -\rho\Psi(q, t) - \frac{\partial V}{\partial t}(q, t)$. Therefore, the resulting problem is a QP12. See Problem 27 in Section IV.

two possible non-null solutions, one being the opposite of the other. In the particular case that q lies on $\mathcal{C}$, both solutions are normalized *tangent vectors* on $\mathcal{C}$ that allow the traversal of the curve, where $T(q)$ and $-T(q)$ enable the circulation in opposite directions.

Consider a solution $T(q)$ to (22), where $\|T(q)\|_2 = 1$, which can be made continuous on q because the gradients $\nabla_q \alpha_i(q)$ are always linearly independent. Let $\phi, V_\alpha : \mathbb{R}^{n-1} \to [0, \infty)$ be two positive definite functions without local minimas and consider the control law

$$\begin{aligned} u^* \in \underset{u}{\operatorname{argmin}} \quad & \phi(\nabla_q \alpha_1(q)^T u, \ldots, \nabla_q \alpha_{n-1}(q)^T u) \\ \text{subject to} \quad & T(q)^T u = \zeta(q) \\ & \nabla_q V(q)^T u = -\rho \Psi(q), \end{aligned} \tag{23}$$

which is a particular case of (3), where $V(q) = V_\alpha(\alpha_1(q), \alpha_2(q), \ldots, \alpha_{n-1}(q))$ is a Lyapunov function and $\zeta(q) \in (0, \infty)$. This control input u^* guarantees that convergence to the curve is achieved and, once on the curve, the configuration $q(t)$ will circulate the curve in a direction given by $T(q)$. Indeed, by construction, $T(q)$ and

$$\nabla_q V(q) = \sum_{i=1}^{n-1} \frac{\partial V}{\partial \alpha_i} \nabla_q \alpha_i(q)$$

are always orthogonal, so the optimization problem is always feasible. The Lyapunov constraint guarantees convergence to $\mathcal{C}$ because the gradients are always linearly independent and V_α has no local minima. In addition, once on the curve, the objective function guarantees that the optimal solution is $u^*(q) = \zeta(q) T(q)$, which is the solution to the first equality, because in that situation only the first constraint in (23) is active, as on the curve both $\nabla_q V(q)$ and $\Psi(q)$ are null. Therefore, the configuration $q(t)$ will move along the curve in the direction given by $T(q)$.

E. Implementing constraints

The control law (3) takes into account the general inequality constraint

$$A(q)\, u \leq b(q), \tag{24}$$

where $\dot{q} = u$ is the control input. Therefore, all constraints must be linear in the control inputs. Furthermore, to always ensure feasible solutions when the task consists in set-point regulation, as discussed in Section III-A, we must enforce $b(q) \geq 0$, $\forall q$, and $u = 0$ must be a feasible solution (i.e., the robot can stop). In case $u = 0$ is not a feasible solution, such as when tracking a time-varying trajectory, the feedforward term must satisfy (20) whereas when executing persistent motions, the component enforcing the minimum velocity must be orthogonal to ∇V.

To briefly exemplify how to represent useful constraints as (24), we consider three types of constraints: 1) maximum and minimum control inputs (configuration velocities); 2) constraints of the form $h(q) \leq 0$, which can be used to enforce obstacle avoidance [27], joint limits, and static balance constraints (e.g., to ensure that the zero moment point is inside the support polygon in humanoid robots); and 3) Pfaffian nonholonomic constraints of the form $C(q)\dot{q} = 0$ [28].

In the first case, we can write $\dot{q} \leq \omega_u$ and $-\dot{q} \leq \omega_l$ where $\omega_u, \omega_l \in (0, \infty)^n$ are the velocities upper and lower limits, respectively. Therefore, $A(q) \triangleq \operatorname{blkdiag}(I_n, -I_n)$ is a block diagonal matrix, where I_n is the $n \times n$ identity matrix, and $b(q) \triangleq \begin{bmatrix} \omega_u^T & \omega_l^T \end{bmatrix}^T \geq 0$.

In the second case, first we define a differentiable function $h(q) : \mathbb{R}^n \to \mathbb{R}$ such that $h(q) \leq 0$ represents feasible robot configurations whereas $h(q) > 0$ represent forbidden configurations [27]. We then transform the nonlinear configuration constraint $h(q) \leq 0$ in a linear constraint (in the control inputs) by using

$$\frac{\partial h}{\partial q}(q)\dot{q} \leq -\eta h(q), \tag{25}$$

for a positive scalar $\eta > 0$ [12], [11]. Since (25) is equivalent to $\dot{h} + \eta h \leq 0$, then if $h(q(0)) \leq 0$ (i.e., the robot initial configuration is outside the forbidden configuration space), by Gronwall's Lemma[7] we have $h(q(t)) \leq e^{-\eta t} h(q(0)) \leq 0$ for all $t \geq 0$ (i.e., the configuration always stays outside the forbidden configuration space). If $\mathcal{Q}_{\text{free}}$ is the space of all non-forbidden configurations, then any dynamical system that implements (25) on $\dot{q}$ has $\mathcal{Q}_{\text{free}}$ as a positive invariant set. In addition, we must ensure that $\mathcal{Q}_{tg} \cap \mathcal{Q}_{\text{free}} \neq \emptyset$, otherwise it is impossible to achieve the task. If we have k constraints such as (25), then

$$A(q) \triangleq \begin{bmatrix} \partial h_1 / \partial q \\ \vdots \\ \partial h_k / \partial q \end{bmatrix}$$

and $b(q) \triangleq \begin{bmatrix} -\eta_1 h_1(q) & \cdots & -\eta_k h_k(q) \end{bmatrix}^T$, with $\eta_i \in (0, \infty)$, $\forall i \in \{1, \ldots, k\}$. Since

$$q(0) \in \mathcal{Q}_{\text{free}} \implies h_i(q) \leq 0, \forall t, i \implies -\eta_i h_i(q) \geq 0, \forall i,$$

then $b(q) \geq 0$, for all $q(t)$.

In the third case, we can write $C(q)\dot{q} = 0$ as $0 \leq C(q)\dot{q} \leq 0$; therefore, $\begin{bmatrix} C(q)^T & -C(q)^T \end{bmatrix} \dot{q} \leq 0$, where $b(q) \triangleq 0$, for all q.

IV. The canonical QP12

Since (3) is a very general formulation that, in general, does not have an analytical solution, we propose the control law[8]

$$\begin{aligned} u^* \in \underset{u}{\operatorname{argmin}} \quad & \gamma \|u\|_1 + (1 - \gamma) \frac{1}{2} \|u\|_2^2 \\ \text{subject to} \quad & \nabla V(q)^T u = -\rho \Psi(q), \end{aligned} \tag{26}$$

with $0 \leq \gamma \leq 1$. In addition to being stable by design, as discussed in Section II-A, and penalizing large values of the control input u, the control law (26) can be solved very efficiently in a semi-analytical manner and provides a trade-off between sparse and even behaviors by properly choosing γ.

[7] Gronwall's Lemma [29] establishes that the differential inequality $\dot{\Omega}(t) + \alpha\Omega(t) \leq 0$ implies the inequality $\Omega(t) \leq e^{-\alpha t}\Omega(0)$.

[8] Since the objective function is strictly convex for $\gamma \in [0, 1)$ and the constraint is convex, it has only one solution; therefore, argmin returns a singleton set.

More specifically, $\gamma \to 1 \implies \Phi(u) \to \|u\|_1$, which induces sparseness, whereas $\gamma \to 0 \implies \Phi(u) \to (1/2)\|u\|_2^2$, which induces evenness, as shown in Example 2; anything in between produces a mixed behavior.

In order to perform mathematical analysis on (26), we first introduce a mathematically convenient canonical form in order to establish the theoretical results.

Definition 3. The canonical QP12 is given by the quadratic convex optimization problem

$$\begin{aligned} \min_{x} \quad & \gamma\|x\|_1 + (1-\gamma)\frac{1}{2}\|x\|_2^2 \\ \text{subject to} \quad & a^T x = b, \end{aligned} \tag{27}$$

where $0 \leq \gamma < 1$, $b \in (0, \infty)$ and the m coefficients a_i of a obey $a_1 \geq a_2 \geq \cdots \geq a_m \geq 0$ with $a_1 > 0$.

Since $a \neq 0$, the constraint $a^T x = b$ is always fulfilled; therefore, the canonical QP12 is always feasible. Furthermore, it can be characterized in an almost analytic manner, as shown next in Proposition 6, and a single algorithm can be used to find its solution. But first we introduce two lemmas, which will help in the proof of Proposition 6. The first one, Lemma 4, shows that any solution to a canonical QP12 has only nonnegative coefficients and the second one, Lemma 5, shows that it always exists a solution that is as sparse as possible.

Lemma 4. *Any solution x to a canonical QP12 optimization problem is such that $x \geq 0$.*

Proof: The proof follows by contradiction. Suppose we have the optimal x and $\exists x_i < 0$. There are two possible cases:

1) If $a_i = 0$ then x_i can be taken to 0 without violating the constraint $a^T x = b$ while improving the objective function. Thus, x is not optimal, which is a contradiction;
2) If $a_i > 0$ then $\exists j$ such that $x_j > 0$ and $a_j x_j > 0$, because otherwise it would be impossible to have $a^T x = b$ with $b > 0$. Therefore, if x_i is increased to zero then x_j can be decreased accordingly, which will improve the objective function. Thus, x is not optimal, which is a contradiction.

■

Lemma 5. *Let $\mathcal{S}(x) \triangleq \{i : x_i \neq 0\}$ and $\overline{\mathcal{S}}(x) \triangleq \{1, \ldots, m\}$, where $x \in \mathbb{R}^m$, then there must exist a solution x^* of the canonical QP12 such that $\mathcal{S}(x^*) = \{1, 2, \ldots, M\}$ for some $M \leq m$.*

Proof: Consider a solution x such that $\mathcal{S}(x) \neq \{1, 2, \ldots, M\}$, because otherwise the result is trivially proved by letting $x^* = x$. We will construct, from x, an equivalent solution x^* in which $\mathcal{S}(x^*) = \{1, 2, \ldots, M\}$ for some $M \leq m$.

Since $\mathcal{S}(x) \neq \{1, 2, \ldots, M\}$, there must exist $r, s \in \overline{\mathcal{S}}(x)$, with $r < s$, such that $x_r = 0$ and $x_s \neq 0$. Assuming that $a_r = a_s$, we can choose $x_r^* = x_s$, $x_s^* = x_r$, and $x_i^* = x_i$ for $i \notin \{r, s\}$. Applying these changes until there are no such indexes r, s, we obtain a solution x^* such that $\mathcal{S}(x^*) = \{1, 2, .., M\}$.

We now show by contradiction that the assumption $a_r = a_s$ holds. Let us suppose that $a_s \neq a_r$ such that $a_s < a_r$ as the indexes are ordered in a non-increasing order. A solution x^* is constructed from the optimal solution x such that $x_i^* = x_i$ for $i \notin \{r, s\}$, $x_r^* = a_s x_s / a_r$, and $x_s^* = 0$. Note that $a^T x^* = b$.

Rewriting the objective function in (27), we obtain $\phi(x) = \sum_{i=1}^n \bar{\phi}(x_i)$, where $\bar{\phi}(x_i) = \gamma|x_i| + (1-\gamma)(1/2)x_i^2$ with $0 \leq \gamma < 1$. Since $x_i \geq 0$ according to Lemma 4 and $\bar{\phi}(x_i)$ is a strictly increasing function in the interval $x_i \in [0, \infty)$, then $\phi(x)$ is also strictly increasing in the valid interval of x. However, since $x_i^* = x_i$ for $i \notin \{r, s\}$ and $a_s < a_r$, then $x_s^* = 0 \leq x_r$ and $x_r^* = a_s x_s / a_r < x_s$, hence the objective function will strictly decrease with this new solution x^*. Thus x is not an optimal solution, which is a contradiction. Therefore, $a_r = a_s$. ■

We can then prove the following result.

Proposition 6. *When $\gamma \neq 1$, a solution to the canonical QP12 optimization problem is*

$$x_i = \begin{cases} \frac{(\lambda_M a_i - \gamma)}{1-\gamma} & \text{if } i \in \{1, \ldots, M\} \\ 0 & \text{otherwise}, \end{cases} \tag{28}$$

where $M \leq m$ and

$$\lambda_M = \left(\sum_{i=1}^{M} a_i^2\right)^{-1} \left((1-\gamma)b + \gamma \sum_{i=1}^{M} a_i\right) \tag{29}$$

with[9]

$$a_M \lambda_M > \gamma \geq a_{M+1} \lambda_M. \tag{30}$$

Proof: Since the optimization problem is convex and the constraint is linear in x, the Karush-Kuhn-Tucker (KKT) conditions for non-differentiable functions [30], which are necessary and sufficient for optimality to this problem, are[10]

$$0 \in \gamma \overline{\text{sgn}}(x) + (1-\gamma)x - \lambda a, \tag{31}$$

$$b = a^T x. \tag{32}$$

where λ is the Lagrange multiplier and the set-valued function $\overline{\text{sgn}} : \mathbb{R} \rightrightarrows \{-1, [-1, 1], 1\}$ is defined as

$$\overline{\text{sgn}}(a) = \begin{cases} \{-1\} & a < 0 \\ [-1, 1] & a = 0 \\ \{1\} & a > 0 \end{cases}$$

and extended to vectors by applying it component-wise.

From Lemma 4 and Lemma 5, there exists a solution $x \geq 0$ and a natural number $M \leq m$ such that $\mathcal{S}(x) = \{1, \ldots, M\}$. By definition, $\forall i \in \mathcal{S}(x)$, $x_i \neq 0$; therefore, $\forall i \in \mathcal{S}(x)$, $x_i > 0$ and (31) reduces to

$$x_i = \frac{\lambda_M a_i - \gamma}{1-\gamma}, \tag{33}$$

[9]The rightmost condition does not need to be verified if $M = m$.

[10]Only the stationarity and primal feasibility conditions are needed for our particular problem, namely $0 \in \partial\Phi(x) + \lambda \partial g(x)$ and $g(x) = 0$, where λ is the Lagrange multiplier, $\partial\Phi(x)$ and $\partial g(x)$ are the subdifferentials of the objective and constraint functions, respectively, and $+$ is the Minkowski sum.

in which we write $\lambda \triangleq \lambda_M$ to emphasize the dependence of λ on M.
Since $\forall i \notin \mathcal{S}(x)$, $x_i = 0$, then

$$\lambda_M a_i > \gamma, \forall i \in \mathcal{S}(x) \tag{34}$$

and, by (31),

$$\lambda_M a_i \in [-\gamma, \gamma], \forall i \notin \mathcal{S}(x), \tag{35}$$

which can be written equivalently as

$$|\lambda_M| a_i = |\lambda_M a_i| \leq \gamma, \forall i \notin \mathcal{S}(x). \tag{36}$$

Using (32) and (33), we obtain

$$\lambda_M = \left(\sum_{i=1}^{M} a_i^2\right)^{-1} \left((1-\gamma)b + \gamma \sum_{i=1}^{M} a_i\right), \tag{37}$$

which is nonnegative. In that case, (36) reduces to

$$\lambda_M a_i \leq \gamma, i \notin \mathcal{S}(x). \tag{38}$$

Since $a_M = \min\{a_1, a_2, \ldots, a_M\}$ and $a_{M+1} = \max\{a_{M+1}, \ldots, a_m\}$, then (34) reduces to $\lambda_M a_M > \gamma$ and (38) reduces to $\lambda_M a_{M+1} \leq \gamma$. Therefore, $\lambda_M a_M > \gamma \geq \lambda_M a_{M+1}$, which concludes the proof. ∎

Remark 7. When $\gamma = 1$ (i.e., sparsest behavior), the solution (28) is not defined and we obtain the optimal solution to (27) by taking $M = 1$ and, since $\max\{a_1, \ldots, a_m\} = a_1 > 0$, $x_1 = b/a_1$ and $x_i = 0$ for all $i \in \{2, \ldots, m\}$. On the other hand, when $\gamma = 0$ (i.e. most uniformly distributed behavior), the solution can be obtained with $M = m$ such that $\lambda_M = b/\|a\|^2$ and thus $x = ba/\|a\|_2^2 = \left(a^T\right)^+ b$, where $\left(a^T\right)^+$ is the right pseudoinverse of a^T.

The solution described in Proposition 6 requires an appropriate value for M that satisfies (30). Although M depends on the parameter γ, there is no analytical expression to find it, but it can be obtained by using an iterative procedure in $\mathcal{O}(m)$ time. For that, we begin from $M = 1$ and compute the value of λ_M incrementally to prevent unnecessary computations, as shown in Algorithm 1. More specifically, in line 4 we initialize λ_M considering $M = 1$, and from line 7 to line 10 the value of λ_M is updated by reusing its previous value and computing only the remaining terms inside the summations in (29). Those updates are done until condition (30) is fulfilled, which in the worst case happens when $M = m$.

In order to solve Problem (26), we first convert it to the canonical QP12 form, which can be done in $\mathcal{O}(m \log m)$ time. Since $\Psi(q) = 0 \iff \nabla V(q) = 0$ implies the trivial solution $u = 0$ when $\nabla V(q) = 0$, we need to analyze only the case where $\nabla V(q) \neq 0$.

Since the equality $\nabla V(q)^T u = -\rho \Psi(q)$ in (26) must be satisfied, then

$$b \triangleq \rho \Psi(q) = -\sum_{i=1}^{m} v_i u_i,$$

where $\begin{bmatrix} v_1 & \cdots & v_m \end{bmatrix} \triangleq \nabla V(q)^T$. Let $\hat{u}_i \triangleq -\text{sgn}(v_i) u_i$ and $\hat{a}_i \triangleq |v_i|$ for all i, sort $\hat{a}$ in a decreasing ordering to obtain

Algorithm 1: Pseudo-code for solving the canonical QP12.

Data: $a \in [0, \infty)^m$ with $a_1 > 0$, $b \in (0, \infty)$, $\gamma \in [0, 1)$
Result: $x \in [0, \infty)^m$

1 $x \leftarrow 0$
2 $\alpha \leftarrow (1-\gamma)b + \gamma a_1$
3 $\beta \leftarrow a_1^2$
4 $\lambda \leftarrow \alpha/\beta$
5 $M \leftarrow 1$
6 **while** $M + 1 \leq m$ *and* ($a_M \lambda \leq \gamma$ *or* $\gamma < a_{M+1}\lambda$) **do**
7 $\quad M \leftarrow M + 1$
8 $\quad \alpha \leftarrow \alpha + \gamma a_M$
9 $\quad \beta \leftarrow \beta + a_M^2$
10 $\quad \lambda \leftarrow \alpha/\beta$
11 **end**
12 $x_{1:M} \leftarrow (\lambda a_{1:M} - \gamma)/(1-\gamma)$ /* The coefficients x_1 to x_M of x are updated */

a and then construct a vector $z \triangleq \begin{bmatrix} z_1 & \cdots & z_m \end{bmatrix}$ such that $a_i = \hat{a}_{z_i}$; that is, z_i contains the unique index of the coefficient in $\hat{a}$ that corresponds to a_i. Since $b > 0$ and $a \geq 0$ (and with coefficients in a decreasing ordering), the canonical QP12 can be used to find the solution $x \geq 0$ such that $x_i = \hat{u}_{z_i}$.

To verify that x is indeed a solution to the original problem, first we show that it respects the constraint:

$$\begin{aligned} a^T x &= \sum_{i=1}^{m} \hat{a}_{z_i} \hat{u}_{z_i} \\ &= -\sum_{i=1}^{m} |v_{z_i}| \text{sgn}(v_{z_i}) u_{z_i} = -\sum_{i=1}^{m} v_{z_i} u_{z_i} = b, \end{aligned}$$

where the last equality holds because $z_i \neq z_j$ for all $i \neq j$ and $z_i \in \{1, \ldots, m\}$. Second, since x is just a reordering of the input u with signal changes in some of its coefficients, the objective function does not change as it is sign invariant; that is, $\Phi(x) = \Phi(u)$. The procedure to find the solution to (26) is summarized in Algorithm 2.

The sorting in line 6 of Algorithm 2 can be done in $\mathcal{O}(m \log m)$ time with a Quicksort algorithm, whereas the problem in canonical form in line 2 can be solved using $\mathcal{O}(m)$ operations. The mapping from the variable x to the variable u, by reordering the vector and making the necessary sign changes, is also done in $\mathcal{O}(m)$ operations.

However, owing to the continuous nature of the control problem, as q changes from one control period to another the vector $\nabla V(q)$ usually changes very little. Therefore, the same ordering of the previous control input is almost always applicable to the next one. As a consequence, the ordering phase—which has the largest complexity— can be skipped by just checking if the previous ordering is also applicable to the current problem. Since this warm-start strategy can be done in $\mathcal{O}(m)$ time, the solution to (26) can be solved almost always using $\mathcal{O}(m)$ operations.

Algorithm 2: Pseudo-code for solving the QP12.

Data: $\nabla V(q) \in \mathbb{R}^m$, $\rho\Psi(q) \in (0,\infty)$, $\gamma \in [0,1)$
Result: Control input $u \in \mathbb{R}^m$

$b \leftarrow \rho\Psi(q)$
$v \leftarrow \nabla V(q)$
for $i \leftarrow 1$ *to* m **do**
  $\hat{a}_i \triangleq |v_i|$
end
$a \leftarrow \texttt{Sort}(\hat{a})$
$z \leftarrow \begin{bmatrix} z_1 & \cdots & z_m \end{bmatrix}$ such that $a_i = \hat{a}_{z_i}$
$x \leftarrow \texttt{CanonicalQP12}(a, b, \gamma)$
for $i \leftarrow 1$ *to* m **do**
  $j \leftarrow z_i$
  $u_j \leftarrow -x_i \text{sgn}(v_j)$
end
$u \leftarrow \begin{bmatrix} u_1 & \cdots & u_m \end{bmatrix}^T$

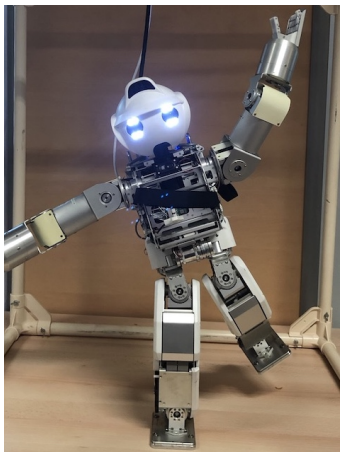


Figure 2: Fujitsu HOAP-3 humanoid robot

V. Experimental results

In order to evaluate the control law (26), which is described in Section IV, an experiment was performed with the Fujitsu HOAP-3 humanoid robot, shown in Fig. 2, which has 21 revolute joints controlled in position. The goal was to control the robot Center of Mass (CoM), denoted by $m(q)$ and defined with respect to the right foot, while the pose of the right foot was kept constant on the ground.

Given the initial joint configuration q_0, shown in Fig. 2, the task consists in moving the initial statically stable Center of Mass position to a constant desired statically stable position, given respectively by

$$m(q_0) = \begin{bmatrix} -0.0085 & 0.0279 & 0.2869 \end{bmatrix}^T \text{m},$$
$$m_d = \begin{bmatrix} 0.01 & 0.025 & 0.2869 \end{bmatrix}^T \text{m}.$$

The desired task function is given by $e_m(q) = m(q) - m_d$, where $m(q)$ is the CoM current position associated with the joint configuration q, and the task differential kinematics is expressed by $\dot{e}_m = J_m(q)\dot{q}$, where $J_m(q) = \partial m/\partial q$ and $\dot{m}_d = 0$, $\forall t$.

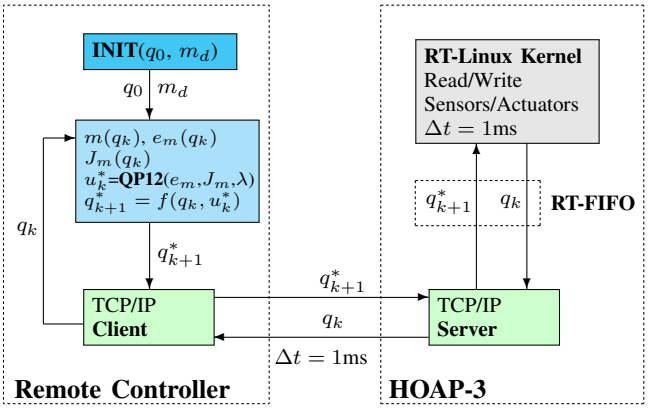


Figure 3: Control Architecture of the experimental setup.

A. Experimental setup

The HOAP-3 robot is equipped with a 1 GHz Pentium III processor using RT-Linux Operating System running at 1 ms sampling period. Since this processor is not powerful enough to solve the CoM controller using the 21 DOF at this sampling period, we have developed a remote controller with a TCP/IP connection using the Julia language v1.0 [31]. The remote controller was implemented on a laptop equipped with a 3.1 GHz Intel Core i7 processor (7920HQ) running Mac OS 10.14.

The total duration time required to compute one sample on the remote controller is in average 0.5 ms.[11] The data communication between the remote controller and the embedded RT-Linux controller on HOAP-3 is carried out by using TCP/IP communication. The TCP/IP server uses the RT-Scheduler through the RT-FIFO to send the current joint position value q_k, read from the robot onboard sensors, to the TCP/IP client on the remote controller. Then, 1 ms after receiving q_k the TCP/IP client sends the control vector q_{k+1}^* back, which synchronizes the remote controller with the real-time task at 1 ms sampling period.

B. Benchmark

In order to assess the performance of QP12 (with $\gamma = 0.5$) in terms of computational time, we compared it with two other algorithms by using the BenchmarkTools package available on Julia. The first algorithm (PINV) is based on the pseudoinverse of the Jacobian matrix [32] and the other one is based on OSQP, which is an operator splitting solver for quadratic programs [33]. Using those three algorithms, we used the `@benchmark` function from the BenchmarkTools package to calculate 10,000 samples of the CoM inverse kinematics for the configuration shown in Fig. 2, whose corresponding center of mass is $m(q_0)$. The results are summarized in Table I,

[11] We solve the QP12 optimization problem in a semi-analytical manner by using Algorithm 2 at each sample time k in the control loop. The optimal solution $u^*(k)$ is then used to compute the desired joint positions of the robot by using first-order Euler integration, that is, $q^*(k+1) = q(k) + u^*(k)\,\Delta t$, where $q(k)$ is the joint position of the robot measured at time k and Δt is the sampling period.

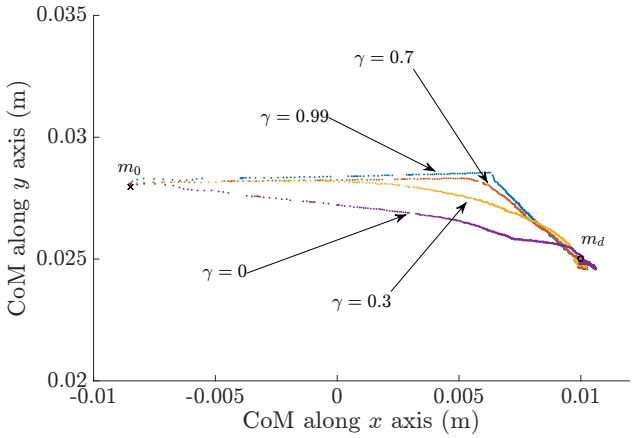


Figure 4: Center of Mass trajectory projected onto the support polygon (right foot) along the x and y axes from $m_0 \triangleq m(q_0)$ to m_d with QP12.

showing that QP12 outperforms both PINV and OSQP in terms of minimum, mean, and maximum times.

	QP12	**PINV**	**QP** (OSQP)
Minimum Time (μs)	**1.259**	7.713	29.64
Mean Time (μs)	**1.358**	12.04	32.3
Maximum Time (μs)	**7.575**	35.15	153.24

Table I: Benchmark of QP12 versus PINV and QP

When considering the equality constraint defined by (16), QP12 takes in average 1.358μs for solving the same inverse kinematics problem. In addition, the computational time to calculate the CoM forward kinematics $m(q)$ is in average 160.95μs, and the computational time to calculate the corresponding Jacobian matrix is in average 109.85μs. However, both the forward kinematics and Jacobian calculations are identical for all algorithms.

C. Analysis of QP12

First, we consider the Lyapunov function $V(q) = \|e_m(q)\|_2^2/2$; therefore,

$$\nabla V(q) = J_m(q)^T e_m(q). \tag{39}$$

The canonical QP12 in (26) is expressed as

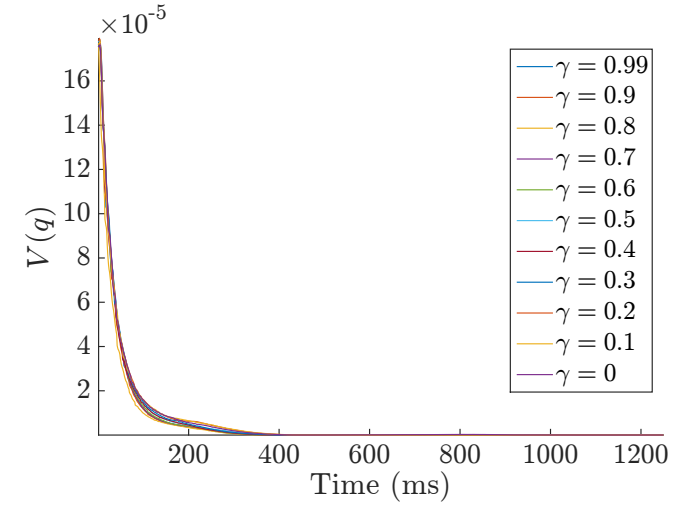
$$\begin{aligned} u^* \in \underset{u}{\operatorname{argmin}} \quad & \gamma\|u\|_1 + (1-\gamma)\frac{1}{2}\|u\|_2^2 \\ \text{subject to} \quad & e_m(q)^T J_m(q) u + \rho\Psi(q) = 0. \end{aligned} \tag{40}$$

Second, in order to take into account input saturation (i.e., joint velocity constraints), we use (16) with (39), and

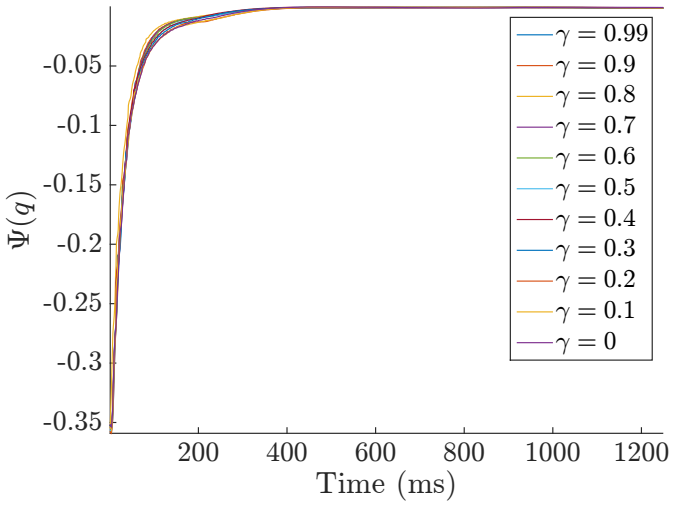
$$R(q) = \frac{2}{\pi}\arctan\left(\beta\|e_m(q)\|_2\right), \tag{41}$$

where $u_{\max}$ is the desired maximum velocity in the joint space and $\beta \in (0, \infty)$.

We have assessed the QP12 semi-analytical algorithm with $\gamma \in \{0, 0.3, 0.7, 0.99\}$, as shown in Fig. 4. When $\gamma = 0$, the length of the CoM trajectory along the x axis, shown in Fig. 4,



(a)



(b)

Figure 5: QP12 with $\beta = 46$, $u_{\max} = 0.6\,\text{rad/s}$, $0 \leq \gamma \leq 0.99$: (*a*) Lyapunov function $V(q)$; (*b*) $\Psi(q)$ function.

within the support polygon is about $\Delta\text{CoM}_x = m_{dx} - m_{0x} = 1.85\text{cm}$. Based on this trajectory, we have performed 11 trials with QP12 to study the effect of different values of γ on the stability and the optimal control vector u^*.

1) Effect of γ on the Lyapunov function dynamics: For all cases, the constraint in (40) ensures the closed-loop stability, as predicted by the theory, because the constraint guarantees that $\dot{V}(q) \leq -\rho\Psi(q) \leq 0$. Furthermore, since $\mathcal{Q}_{tg} \subset \mathcal{Q}_{\text{free}}$, $\mathcal{Q}_{tg}$ is asymptotically stable, as shown in Fig. 5a. Since the convergence rate is determined exclusively by the equality constraint in (40), the Lyapunov function dynamics is very similar for all values of γ, as expected.

The $\Psi(q)$ function used in the equality constraint in (40) has a strong correlation with $V(q)$ and reinforce the idea that Ψ influences the decrease of V.

2) Effect of γ on the control input sparseness: As shown in Section III-B, the parameter γ defines the density of the control vector; therefore, the control vector is densest with $\gamma = 0.0$, as the solution is equivalent to the one based on

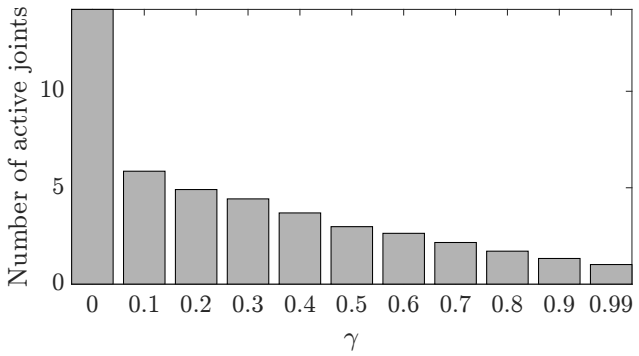


Figure 6: QP12: Average number of active joints along the motion with $0 \leq \gamma \leq 0.99$ and $u_{\max} = 0.6\,\mathrm{rad/s}$.

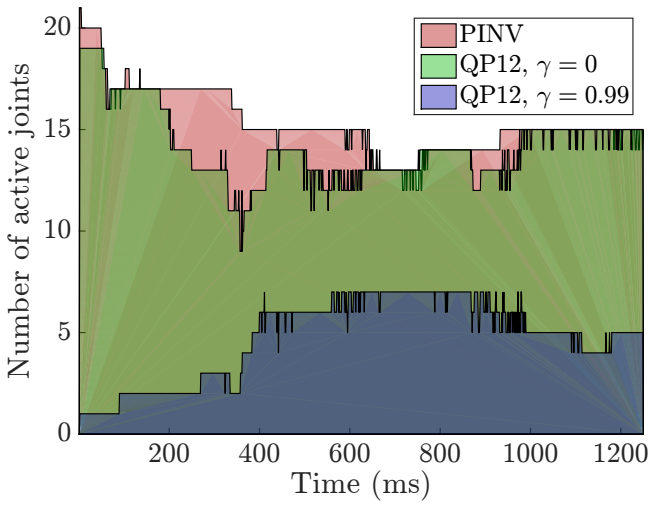


Figure 7: Number of active joints vs. time with QP12 ($\gamma = 0$ and $\gamma = 0.99$) and pseudo-inverse (PINV).

the Jacobian matrix pseudoinverse, and sparse with $\gamma = 0.99$. Fig. 6 shows the average number of active joints during the whole motion for each value of γ. The number of active joints decreases almost linearly for $\gamma \in [0.1, 0.99]$. When compared with the controller based on the pseudoinverse, QP12 with $\gamma = 0.1$ provides a solution that uses, in average, 6 actuators instead of 18 actuators for the pseudo-inverse. The number of active joints, when $\gamma = 0.1$, is higher than the sparsest solution for this problem, which theoretically requires only two active joints, but the sparsest solution can be obtained when $\gamma = 0.99$.

Fig. 7 shows the number of active joints along the motion for QP12 (for $\gamma = 0$, which corresponds to a dense solution, and $\gamma = 0.99$, which corresponds to a sparse one) and PINV. When $\gamma = 0$ the behaviors of QP12 and PINV are very similar as both provide a dense control vector u^*. On the contrary, $\gamma = 0.99$ provides a sparse control vector u^* with just one or two active joints, which are in this case the pitch and roll joints of the right ankle.

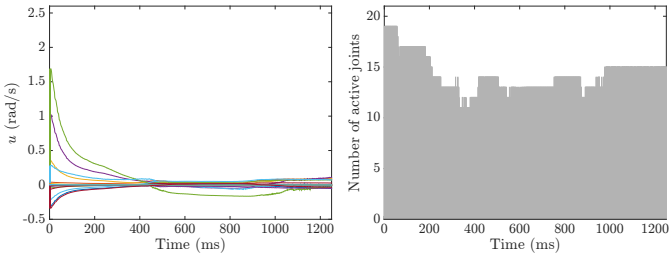
Last, Fig. 8 presents side-by-side the effect of γ on the smoothness and sparseness of the control inputs: the greater the value of γ, the more sparse and abrupt are the control inputs. Conversely, the smaller the value of γ, the more dense and smooth are the control inputs. A joint is considered inactive if its velocity is below a threshold of $0.001\mathrm{rad/s}$. In order to prevent chattering when the robot stabilizes, which may cause vibrations in the robot structure, we used an adaptive value for γ, namely $\gamma = (2\gamma_{\mathrm{far}}/\pi)\arctan(100\|\nabla V(q)\|)$, such that *only very close* to a stable region $\gamma \to 0$ but $\gamma \approx \gamma_{\mathrm{far}}$ elsewhere. Therefore, since the system achieves the stable region aproximately after 400 ms, more joints are active after that moment because γ tends to be much smaller than γ_{far}.

VI. Conclusion

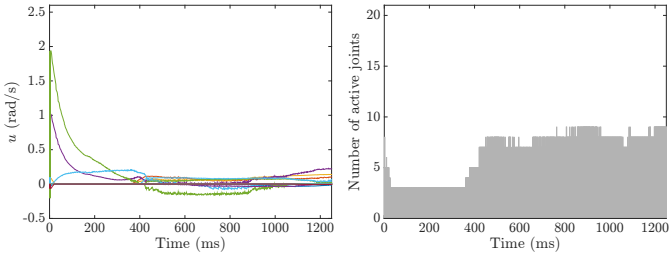
The last couple of decades have seen an increasing demand for complex robots, such as humanoids and mobile manipulators, which are highly redundant with respect to most tasks. A popular way of dealing with such complexity is to use convex programming subject to constraints, more particularly quadratic programming, thanks to its computational efficiency. However, proving closed-loop stability is usually hard for arbitrary objective functions, therefore one of our main contributions is an optimization-based control law that ensures closed-loop stability by design, which can be achieved by imposing suitable constraints based on Lyapunov function derivatives, namely Lyapunov constraints. This property is very important as we guarantee that the closed-loop system is stable for any objective function without having to provide additional mathematical proofs. In addition, we have proposed a pseudo-analytical solution to solve an objective function composed of a convex combination of quadratic and l_1-norm objective functions (QP12), which provides useful properties in kinematic control, such as the control vector sparseness, by using only one parameter. The QP12 algorithm is suitable for real-time applications and outperforms the compared state-of-the-art quadratic optimization solvers in term of computational time. In addition to providing guidelines on how to design suitable Lyapunov constraints with different convergence profiles, we also presented a simple way to impose limits on joint velocities by properly defining those Lyapunov constraints. It is important to highlight that although QP12 has a pseudo-analytic solution, it admits only one specific equality constraint. Despite this limitation, QP12 is a particular case of our much more general formulation, which is stable by design and includes both equality and inequality constraints, but at the expense of using a convex optimization solver.

We validated our method by controlling the center of mass of the humanoid robot HOAP3, in real-time. The results showed that, by properly adjusting just one parameter, our controller yielded a solution as dense as the one provided by the classic kinematic controller based on the Jacobian pseudoinverse ($\gamma = 0$). On the other hand, when $\gamma = 0.99$, the controller yielded the sparsest solution possible for this particular task.

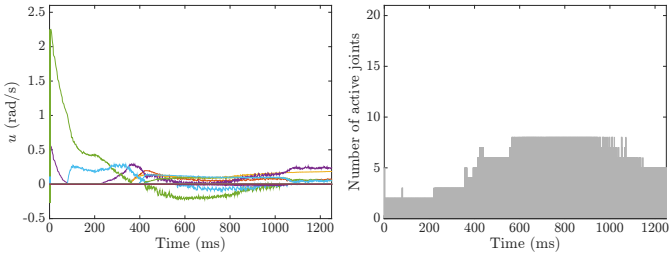
These properties could be very useful for controlling redundant robots in different ways. For instance, in physical human-robot interaction, it could be convenient for the human operator to interact with a robot that performs sparse motions



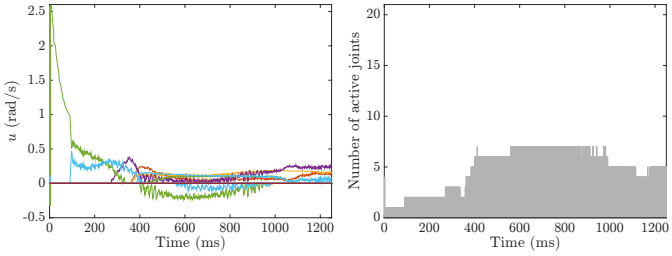
(a) $\gamma_{\text{far}} = 0$



(b) $\gamma_{\text{far}} = 0.3$



(c) $\gamma_{\text{far}} = 0.7$



(d) $\gamma_{\text{far}} = 0.99$

Figure 8: The effect of $\gamma = (2\gamma_{\text{far}}/\pi)\arctan(100\|\nabla V(q)\|)$ on the smoothness and sparseness of the control inputs: the greater the value of γ, the more sparse and abrupt are the control inputs. On the *left*, the control inputs; on the *right*, the number of active joints for different values of γ. When *very close* to a stable region, $\gamma \to 0$, otherwise $\gamma \approx \gamma_{\text{far}}$.

because dense motions may be harder to predict. On the other hand, industrial applications usually require a trade-off between sparse and dense solutions or dense solutions only.

Even though we have proposed an optimization-based controller, we have not explicitly tackled multiple objectives in this paper. More specifically, differently from usual approaches, our objective function is not directly related to the task. Instead, it quantifies how good is a control input at a particular configuration according to a relevant metric. Therefore, since the information related to the task must be encoded in the Lyapunov constraint, multiple objectives could be encoded as a weighted Lyapunov function.

Finally, although information regarding robot dynamics is not explicitly considered in our current formulation, some dynamics information can be used by enforcing appropriate constraints. However, when dealing with severe dynamics, usually torque control inputs must be considered instead of velocity control inputs, which is one of our main assumptions. Hence, our method in the present state would not be the most appropriate for those cases. Nonetheless, with appropriate modifications, the overall idea of stability by design can be applied to dynamics and this topic will be explored in future works.

REFERENCES

[1] A. Colome and C. Torras, "Closed-Loop Inverse Kinematics for Redundant Robots: Comparative Assessment and Two Enhancements," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 2, pp. 944–955, apr 2015.

[2] J. Funda, R. Taylor, B. Eldridge, S. Gomory, and K. Gruben, "Constrained Cartesian motion control for teleoperated surgical robots," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 3, pp. 453–465, jun 1996.

[3] M. Li, M. Ishii, and R. H. Taylor, "Spatial Motion Constraints Using Virtual Fixtures Generated by Anatomy," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 4–19, feb 2007.

[4] M. VUKOBRATOVIĆ and B. BOROVAC, "ZERO-MOMENT POINT - THIRTY FIVE YEARS OF ITS LIFE," *International Journal of Humanoid Robotics*, vol. 01, no. 01, pp. 157–173, mar 2004.

[5] J. L. G. Olavo, G. D. Thums, T. A. Jesus, L. C. de Araujo Pimenta, L. A. B. Torres, and R. M. Palhares, "Robust Guidance Strategy for Target Circulation by Controlled UAV," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 3, pp. 1415–1431, jun 2018.

[6] 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.

[7] 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, apr 1995.

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

[9] N. Mansard and F. Chaumette, "Task Sequencing for High-Level Sensor-Based Control," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 60–72, feb 2007.

[10] 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.

[11] 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.

[12] B. Faverjon and P. Tournassoud, "A local based approach for path planning of manipulators with a high number of degrees of freedom," in *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, vol. 4. Institute of Electrical and Electronics Engineers, 1987, pp. 1152–1159.

[13] M. de Lasa, I. Mordatch, and A. Hertzmann, "Feature-based locomotion controllers," *ACM Transactions on Graphics*, vol. 29, no. 4, p. 1, jul 2010.

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

[15] K. Dufour and W. Suleiman, "On integrating manipulability index into inverse kinematics solver," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, sep 2017, pp. 6967–6972.

[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 - VRST '05*. New York, New York, USA: ACM Press, 2005, p. 163.

[17] Z. K. Kingston, N. T. Dantam, and L. E. Kavraki, "Kinematically constrained workspace control via linear optimization," *IEEE-RAS International Conference on Humanoid Robots*, vol. 2015-Decem, pp. 758–764, 2015.

[18] P. Berthet-Rayne, K. Leibrandt, G. Gras, P. Fraisse, A. Crosnier, and G.-Z. Yang, "Inverse Kinematics Control Methods for Redundant Snake-like Robot Teleoperation During Minimally Invasive Surgery," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2501–2508, jul 2018.

[19] K. Al Khudir, G. Halvorsen, L. Lanari, and A. De Luca, "Stable Torque Optimization for Redundant Robots Using a Short Preview," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2046–2053, apr 2019.

[20] N. Scianca, M. Cognetti, D. De Simone, L. Lanari, and G. Oriolo, "Intrinsically stable MPC for humanoid gait generation," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, nov 2016, pp. 601–606.

[21] V. M. Gonçalves, P. Fraisse, A. Crosnier, and B. V. Adorno, "Parsimonious Kinematic Control of Highly Redundant Robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 65–72, jan 2016.

[22] C. Samson, M. L. Borgne, and B. Espiau, *Robot control: The Task Function Approach*. Oxford University Press, 1991.

[23] A. Deo and I. Walker, "Minimum effort inverse kinematics for redundant manipulators," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 5, pp. 767–775, 1997.

[24] C. Studer, T. Goldstein, W. Yin, and R. G. Baraniuk, "Democratic Representations," 2014.

[25] B. K. Natarajan, "Sparse Approximate Solutions to Linear Systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, apr 1995.

[26] V. M. Gonçalves, L. C. 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," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 647–659, aug 2010.

[27] M. M. Marinho, B. V. Adorno, K. Harada, and M. Mitsuishi, "Active Constraints Using Vector Field Inequalities for Surgical Robots," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2018, pp. 5364–5371.

[28] J. J. Quiroz-Omaña and B. V. Adorno, "Whole-Body Kinematic Control of Nonholonomic Mobile Manipulators Using Linear Programming," *Journal of Intelligent & Robotic Systems*, vol. 91, no. 2, pp. 263–278, aug 2018.

[29] T. H. Gronwall, "Note on the Derivatives with Respect to a Parameter of the Solutions of a System of Differential Equations," *The Annals of Mathematics*, vol. 20, no. 4, p. 292, jul 1919.

[30] A. Ruszczyński, "Nonlinear Optimization," in *Nonlinear Optimization*. Princeton University Press, 2006, p. 464.

[31] J. Bezanson, S. Karpinski, V. B. Shah, and A. Edelman, "Julia: A Fast Dynamic Language for Technical Computing," Tech. Rep., sep 2012.

[32] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*, ser. Advanced Textbooks in Control and Signal Processing. London: Springer-Verlag London, 2009.

[33] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An Operator Splitting Solver for Quadratic Programs," nov 2017.