Joint Position and Velocity Bounds in Discrete-Time Acceleration/Torque Control of Robot Manipulators

Andrea del Prete

Joint Position and Velocity Bounds in Discrete-Time Acceleration/Torque Control of Robot Manipulators

Andrea Del Prete[1], *Member, IEEE*

***Abstract*—This paper deals with the problem of controlling a robotic system whose joints have bounded position, velocity and acceleration/torque. Assuming a discrete-time acceleration control, we compute tight bounds on the current joint accelerations that ensure the existence of a feasible trajectory in the future. Despite the clear practical importance of this issue, no complete and exact solution has been proposed yet, and all existing control architectures rely on hand-tuned heuristics. We also extend this methodology to torque-controlled robots, for which joint accelerations are only indirectly bounded by the torque limits. Numerical simulations are presented to validate the proposed method, which is computationally efficient and hence suitable for high-frequency control.**

***Index Terms*—Motion Control, Robot Safety, Optimization and Optimal Control**

I. Introduction

ONE of the main challenges in the control of robotic systems is the fact that they are highly constrained. Robot joints typically have bounded positions. Moreover, actuators have bounded velocity and acceleration. Velocity and acceleration may be not directly bounded, but their limits may be the consequence of other bounds, such as on the motor current, or the gear-box torque. This paper focuses on the control of joints with constant bounds on position, velocity and acceleration. Even though velocity and acceleration bounds may be not constant, they can be often well approximated by constant values [1], [2], [3], [4], [5], [6], [7].

First, let us introduce the notation that will be used throughout the paper:

- $\wedge$ and $\vee$ denote the logical quantifiers AND and OR
- $t \in \mathbb{R}^+$ denotes time
- $i \in \mathbb{N}$ denotes discrete time steps
- δt is the time-step duration of the controller
- $q(t), \dot{q}(t), \ddot{q}(t) \in \mathbb{R}$ are the joint position, velocity and acceleration at time t
- $q_i \triangleq q(i\delta t), \quad \dot{q}_i \triangleq \dot{q}(i\delta t), \quad \ddot{q}_i \triangleq \ddot{q}(i\delta t)$
- q^{min}, q^{max} are the joint position boundaries
- $\dot{q}^{max}$, $\ddot{q}^{max}$ are the maximum velocity and acceleration

Manuscript received: April, 20, 2017; Accepted July, 30, 2017.

This paper was recommended for publication by Editor Nikos Tsagarakis upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the European Research Council through the Actanthrope project (ERC Grant Agreement 340050). The author would like to thank Nicolas Mansard and Steve Tonneau for the help and the useful discussions.

[1]Andrea Del Prete is with the CNRS, LAAS, 7 avenue du colonel Roche, Univ. de Toulouse, LAAS, F-31400 Toulouse, France. e-mail: adelpret@laas.fr

Digital Object Identifier (DOI): see top of this page.

Assuming a constant acceleration throughout each time step i, the future position q_{i+1} and velocity $\dot{q}_{i+1}$ are functions of the current acceleration $\ddot{q}_i$:

$$\begin{aligned} q_{i+1} &= q_i + \delta t\, \dot{q}_i + \frac{1}{2}\, \delta t^2 \ddot{q}_i \\ \dot{q}_{i+1} &= \dot{q}_i + \delta t\, \ddot{q}_i \end{aligned} \tag{1}$$

A naive approach to bound accelerations is to compute the maximum and minimum $\ddot{q}_i$ such that q_{i+1} and $\dot{q}_{i+1}$ are within their bounds:

$$\begin{aligned} \ddot{q}_i &\leq \min\left(\quad \ddot{q}^{max}, \frac{1}{\delta t}(\quad \dot{q}^{max} - \dot{q}_i), \frac{2}{\delta t^2}(q^{max} - q_i - \delta t\, \dot{q}_i) \right) \\ \ddot{q}_i &\geq \max\left(-\ddot{q}^{max}, \frac{1}{\delta t}(-\dot{q}^{max} - \dot{q}_i), \frac{2}{\delta t^2}(q^{min} - q_i - \delta t\, \dot{q}_i) \right) \end{aligned} \tag{2}$$

This approach is unsatisfactory because the resulting bounds may be incompatible [1]. Several improvements have been proposed, but none of them have completely solved the problem.

A common approach [1], [2] is to use a larger value of δt in (2), which results (most of the time) in stricter bounds. This helps reducing the acceleration when approaching a bound, but does not guarantee constraint compatibility.

Another approach [3] is to bound velocity with a hand-tuned linear function of the distance to the position limit. Reducing the velocity as you get close to a position bound is surely a sensible idea, but this method does not explicitly account for acceleration limits.

Other inverse-dynamics control frameworks [8], [9] simply do not model joint position-velocity bounds, but rather rely on the design of reference trajectories that ensure their satisfaction. This approach cannot guarantee that the controller reaction to a disturbance will not lead to violating a bound.

Control barrier functions provide a general framework for handling constraints with arbitrary relative degree [10], [11]. However, these methods do not deal with constraint conflicts, which are the key issue when considering bounds on position and acceleration [5].

To the best of our knowledge, Decre et al. [6] have been the first ones trying to provide formal guarantees of satisfaction of position, velocity and acceleration bounds. They computed the future position trajectory assuming maximum deceleration, and imposed the satisfaction of the position bound for all the future states. Their method does not require any hand tuning, but it has two critical issues. First, they assumed constant velocity throughout the time step, so they do not really bound the acceleration, but a pseudo-acceleration, defined as $(\dot{q}_{i+1} - \dot{q}_i)/\delta t$. Second, their method may lead to

conflicts between velocity and acceleration limits when getting close to the position bounds. This second issue was later addressed by Rubrecht et al. [5], [7]. However, they introduced some conservatism in the solution, which they suggest to be beneficial in case of measurement errors. We believe that the robustness issue should be tackled explicitly if necessary, and not through some arbitrary conservatism. Moreover, also this work dealt with pseudo-acceleration bounds.

Other relevant results have been proposed in the field of online trajectory generation. Kroeger et al. [4] presented online trajectory-generation algorithms that can guarantee velocity, acceleration and jerk bounds. While these algorithms are fast enough to run at 1 kHz, they treat a different problem from the one treated in this paper. First, they do not consider position limits, while we do. Second, they compute minimum time joint-space trajectories, while we compute the minimum and maximum accelerations that ensure the possibility to satisfy the bounds in the long term. This means that, for instance, motion can be generated according to a task-space criterion, such as following a trajectory with the end-effector, while exploiting the manipulator redundancy to satisfy the robot constraints.

The two main contributions of our method are:

- it is exact, i.e. not conservative—contrary to [5]
- it assumes constant acceleration between consecutive time steps, which allows bounding the real acceleration—rather than a pseudo-acceleration [5], [6].

Section II formally states the problem and Section III presents an efficient algorithm to solve it. Section IV discusses a way to extend this result to torque control. Section V validates the approach through numerical simulations. Finally, Section VI draws the conclusions.

II. Problem Statement

Let us consider a robot whose joints have limited position, velocity and acceleration. For one joint, we define the set of all feasible states $\mathcal{F}$ as:

$$\mathcal{F} = \{(q, \dot{q}) \in \mathbb{R}^2 : q^{min} \leq q \leq q^{max}, |\dot{q}| \leq \dot{q}^{max}\} \tag{3}$$

Moreover, let us assume that our control inputs are the joint accelerations, which are also bounded: $|\ddot{q}| \leq \ddot{q}^{max}$. At each time step, we want to know the maximum and minimum accelerations such that position and velocity limits can still be satisfied in the future. This can be formulated as an infinite-horizon optimal control problem:

$$\begin{aligned} \ddot{q}_0^{max} = \underset{\ddot{q}_0, \ddot{q}_1, \dots}{\text{maximize}} \quad & \ddot{q}_0 \\ \text{subject to} \quad & (q(t), \dot{q}(t)) \in \mathcal{F} \qquad \forall t > 0 \\ & |\ddot{q}_i| \leq \ddot{q}^{max} \qquad \forall i \geq 0 \\ & (q(0), \dot{q}(0)) \quad \text{fixed} \end{aligned} \tag{4}$$

Similarly, we can define $\ddot{q}_0^{min}$ by minimizing $\ddot{q}_0$ rather than maximizing it. Since acceleration can change only at discrete time step, position and velocity between two successive time steps are:

$$\begin{aligned} q(i\delta t + t) &= q_i + t\,\dot{q}_i + \frac{t^2}{2}\ddot{q}_i \qquad \forall i \geq 0, t \in [0, \delta t] \\ \dot{q}(i\delta t + t) &= \dot{q}_i + t\,\ddot{q}_i \qquad \forall i \geq 0, t \in [0, \delta t] \end{aligned} \tag{5}$$

The joint limits have to be satisfied not only at the discrete time steps, but in continuous time, so the problem has an (uncountable) infinite number of constraints. Because of the infinite number of variables and constraints we cannot solve this problem in this form. The main contribution of this paper is an efficient algorithm to compute the exact solution of this problem.

III. Problem Solution

A. Viability

We recall now the concept of viability, which we will exploit to reformulate problem (4) in a simpler form. A state is *viable* [12] if starting from that state there exists a sequence of control inputs that allows for the satisfaction of all the constraints in the future. Formally, we can define the viability kernel $\mathcal{V}$ gathering all viable states as:

$$\begin{aligned} (q(0), \dot{q}(0)) \in \mathcal{V} \quad \Leftrightarrow \quad \exists (\ddot{q}_i)_{i=0}^{\infty} : (q(t), \dot{q}(t)) \in \mathcal{F} \quad \forall t \geq 0, \\ |\ddot{q}_i| \leq \ddot{q}^{max} \quad \forall i \geq 0 \end{aligned} \tag{6}$$

The interest of introducing the viability kernel $\mathcal{V}$ is that ensuring the existence of a feasible future trajectory (i.e. our original problem) is clearly equivalent to ensuring that the next state belongs to $\mathcal{V}$. However, this definition of $\mathcal{V}$ does not immediately provide practical utility: verifying its membership amounts to finding an infinite sequence of accelerations that results in a feasible trajectory, which is too computationally demanding. In the following we derive an equivalent definition of $\mathcal{V}$ that allows us to check membership easily. Thanks to this, we reformulate the hard problem of satisfying the position-velocity-acceleration limits as the simple problem of ensuring that the next state is viable.

1) Continuous-Time Control: Let us start by assuming that we can change $\ddot{q}$ at any instant (i.e. it can vary continuously). This results in a set of viable states $\mathcal{V}^c$, which is a superset of $\mathcal{V}$, namely $\mathcal{V} \subseteq \mathcal{V}^c$.

It is obvious that $\mathcal{V}^c$ is a subset of $\mathcal{F}$ because all viable states are also feasible, but not all feasible states are viable. If q is approaching one of its bounds with a large velocity, then the acceleration capabilities may not be sufficient to stop before the bound. For a given initial position q_0, we can find the maximum initial velocity $\dot{q}_M^{\mathcal{V}}$ that allows us to satisfy the position limits in the future:

$$\begin{aligned} \dot{q}_M^{\mathcal{V}} = \underset{\dot{q}_0, \ddot{q}(t)}{\text{maximize}} \quad & \dot{q}_0 \\ \text{subject to} \quad & (q(t), \dot{q}(t)) \in \mathcal{F} \qquad \forall t \geq 0 \\ & |\ddot{q}(t)| \leq \ddot{q}^{max} \qquad \forall t \geq 0 \\ & q(0) = q_0, \quad \dot{q}(0) = \dot{q}_0 \end{aligned} \tag{7}$$

The solution of this problem is rather intuitive: the maximum initial velocity is such that, if we constantly apply the maximum deceleration, we end up exactly at q^{max} with zero velocity. Then to solve the problem we can:

1) write the position trajectory for constant acceleration $\ddot{q}(t) = -\ddot{q}^{max}$,
2) compute the time at which the velocity of this trajectory is zero $t^0 = \dot{q}_0 / \ddot{q}^{max}$,

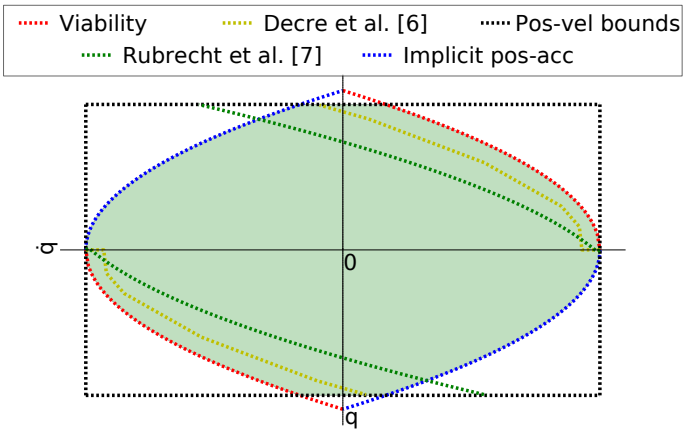


Fig. 1: State-space plot showing the viability constraints (as computed in this paper, in [5] and in [6]), the position and velocity bounds and the constraints implicitly imposed by the position and acceleration bounds for $q^{max} = -q^{min} = 1$ rad, $\dot{q}^{max} = 5$ rad/s, $\ddot{q}^{max} = 15$ rad/s^2.

3) compute the initial velocity such that $q(t^0) = q^{max}$

Following these steps we find:

$$\dot{q}^{\mathcal{V}}_M(q) = \sqrt{2\ddot{q}^{max}(q^{max} - q_0)} \tag{8}$$

Similarly, the minimum initial velocity to ensure viability is:

$$\dot{q}^{\mathcal{V}}_m(q) = -\sqrt{2\ddot{q}^{max}(q_0 - q^{min})} \tag{9}$$

We can conclude that the set of viable states is:

$$\mathcal{V}^c = \{(q, \dot{q}) : (q, \dot{q}) \in \mathcal{F},\ \dot{q}^{\mathcal{V}}_m(q) \leq \dot{q} \leq \dot{q}^{\mathcal{V}}_M(q)\} \tag{10}$$

This definition of $\mathcal{V}^c$ allows us to check easily the viability of a state by just verifying three inequalities.

Fig. 1 shows how $\mathcal{V}^c$ and $\mathcal{F}$ may look like in a state-space plot. The same plot also depicts the joint velocity limits computed using the methods presented in [5] and in [6]. The method proposed by [6] is conservative in the context of acceleration control because it was derived assuming velocity control. Moreover, we can see a discontinuity when approaching the position bounds. The second method [5] fixed the issue of the discontinuity, but it is conservative (even assuming velocity control). Finally, Fig. 1 also shows the state constraints implicitly imposed by the position-acceleration bounds, e.g. starting from q^{min} with zero velocity and applying maximum acceleration we find the maximum velocity we can reach for each state.

2) Discrete-Time Control: If the control can only change at discrete time steps, we have to pay attention to what happens after we reached the position boundary (e.g. q^{max}) with zero velocity. In case of continuous-time control we can switch immediately to $\ddot{q} = 0$ after reaching q^{max}, so we can be sure that no constraint violation will occur. In case of discrete-time control instead we need to maintain $\ddot{q} = -\ddot{q}^{max}$ until the end of the time step in which we reach q^{max}. In theory this may lead us to violating the lower position or velocity bound. However, in practice, this is extremely rare, as we argue in the following.

Let us assume the worst case, namely we reach the state $(q^{max}, 0)$ while applying maximum deceleration a moment after the beginning of the time step. In this case we need to maintain $\ddot{q} = -\ddot{q}^{max}$ for the remaining of the time step (which is approximately δt). At the end of the time step the state will then be $q = q^{max} - 0.5\delta t^2\ddot{q}^{max}$, $\dot{q} = -\delta t\,\ddot{q}^{max} \triangleq \dot{q}^{discr}$. At that point we can bring the joint to a stop by applying $\ddot{q} = \ddot{q}^{max}$ for one time step, which would lead us to $q = q^{max} - \delta t^2\ddot{q}^{max} \triangleq q^{discr}$. In case q^{discr} and $\dot{q}^{discr}$ are within their bounds, then we have nothing to worry about and $\mathcal{V} = \mathcal{V}^c$. The conditions to verify are then:

$$q^{max} - q^{min} \geq \delta t^2\ddot{q}^{max}, \qquad \dot{q}^{max} \geq \delta t\,\ddot{q}^{max} \tag{11}$$

Typically these conditions are verified because δt is much smaller than 1. For instance, even assuming a value of δt as large as 0.1 s, to violate (11) we would need either $\dot{q}^{max}$ to be more than 10 times smaller than $\ddot{q}^{max}$, or $(q^{max} - q^{min})$ to be at least 100 times smaller than $\ddot{q}^{max}$. Even if this is the case, we can safely set $\ddot{q}^{max}$ to be smaller than its real value so as to satisfy (11).

When (11) is not verified the definition of $\mathcal{V}$ is much more complex than that of $\mathcal{V}^c$. To avoid this mathematical complexity of limited practical utility, in the remaining of the paper we will assume that (11) is verified, and so $\mathcal{V} = \mathcal{V}^c$.

B. Reformulating (4) *in Terms of Viability*

Now that we derived a simple definition of $\mathcal{V}$, we can exploit it to reformulate problem (4). Given the current state $(q, \dot{q}) \in \mathcal{V}$, we need to find the maximum (or minimum) value of $\ddot{q}$ such that: i) the next state $(q(\delta t), \dot{q}(\delta t))$ belongs to $\mathcal{V}$, and ii) the whole trajectory leading to the next state belongs to $\mathcal{F}$. Note that the first condition alone is not sufficient: the trajectory between two viable states may violate a constraint. Viability only ensures the existence of a feasible future trajectory, meaning that, even starting from a viable state, you can end up violating a constraint. In mathematical terms we reformulate (4) as:

$$\begin{aligned} \ddot{q}^{max}_0 = \underset{\ddot{q}}{\text{maximize}} \quad & \ddot{q} \\ \text{subject to} \quad & q(0) = q, \quad \dot{q}(0) = \dot{q} \\ & (q(t), \dot{q}(t)) \in \mathcal{F} \qquad \forall t \in [0, \delta t] \\ & \dot{q}^{\mathcal{V}}_m(q(\delta t)) \leq \dot{q}(\delta t) \leq \dot{q}^{\mathcal{V}}_M(q(\delta t)) \\ & |\ddot{q}| \leq \ddot{q}^{max} \end{aligned} \tag{12}$$

This problem is much simpler than its original version because: i) it has a single variable (rather than an infinite sequence of variables), and ii) its constraints concern only the trajectory in $[0, \delta t]$ (rather than in $[0, \infty]$). However, even problem (12) is hard to solve because its constraints are i) infinitely many, and ii) nonlinear. In the next three subsections we reformulate the inequality constraints of problem (12); each one will give us a lower and an upper bound on $\ddot{q}$, which we will combine in Section III-F.

C. Position Inequalities

The position trajectory is a second-order polynomial, which we want to remain bounded within the position limits:

$$q^{min} \leq q + t\dot{q} + 0.5t^2\ddot{q} \leq q^{max} \qquad \forall t \in [0, \delta t] \tag{13}$$

The classical approach in the literature is to ensure that these inequalities are satisfied only for $t = \delta t$. If the velocity does not change sign in the current time step, this is actually enough to guarantee the satisfaction of these constraints over the whole interval $[0, \delta t]$. However, in case of a change in the sign of the velocity, the constraints may be satisfied for $t = \delta t$, but violated for some value of t between 0 and δt. Let us focus first on the upper bound. To reduce the infinite number of constraints to a finite number we rewrite the upper-bound constraint in (13) as:

$$f(\ddot{q}) \leq q^{max}, \tag{14}$$

where:

$$f(\ddot{q}) = \underset{t \in [0, \delta t]}{\text{maximize}} \, [q + t\dot{q} + 0.5t^2\ddot{q}] \tag{15}$$

To find the time at which the position trajectory reaches its extremum we set its derivative to zero:

$$\dot{q} + t\ddot{q} = 0 \quad \rightarrow \quad t^{ext} \triangleq -\dot{q}/\ddot{q} \tag{16}$$

The extremum is a maximum and it is reached in the interior of the time step if and only if:

$$\dot{q} > 0, \qquad \ddot{q} \leq -\dot{q}/\delta t \triangleq \ddot{q}_1^M \tag{17}$$

If these conditions are satisfied the maximum position is:

$$f(\ddot{q}) = q + t^{ext}\dot{q} + 0.5(t^{ext})^2\ddot{q} = q - \dot{q}^2/(2\ddot{q}) \tag{18}$$

Constraint (14) then becomes:

$$\ddot{q} \leq \frac{-\dot{q}^2}{2(q^{max} - q)} \triangleq \ddot{q}_2^M \tag{19}$$

If instead the conditions (17) are not satisfied then the maximum position is reached at the boundaries of the time step, so we only need to ensure that the constraint is satisfied for $t = \delta t$, which gives us:

$$\ddot{q} \leq \frac{2}{\delta t^2}(q^{max} - q - \delta t\, \dot{q}) \triangleq \ddot{q}_3^M \tag{20}$$

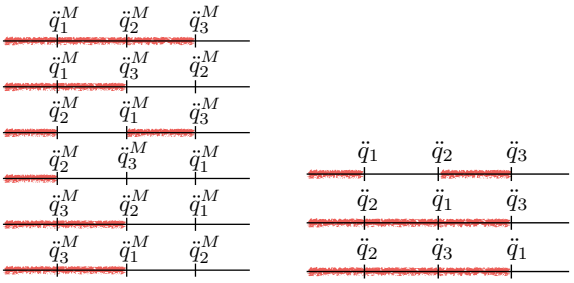
Now we can put all of this together. If $\dot{q} \leq 0$ the acceleration upper bound is simply $\ddot{q}_3^M$. If $\dot{q} > 0$ things are more complicated instead: if (17) is satisfied then the upper bound is $\ddot{q}_2^M$, otherwise the upper bound is $\ddot{q}_3^M$. We may then rewrite (14) as:

$$(\ddot{q} \leq \ddot{q}_1^M \wedge \ddot{q} \leq \ddot{q}_2^M) \quad \vee \quad (\ddot{q} > \ddot{q}_1^M \wedge \ddot{q} \leq \ddot{q}_3^M), \tag{21}$$

or equivalently:

$$\ddot{q} \leq \min(\ddot{q}_1^M, \ddot{q}_2^M) \quad \vee \quad \ddot{q}_1^M < \ddot{q} \leq \ddot{q}_3^M \tag{22}$$

Depending on the values of $\ddot{q}_1^M, \ddot{q}_2^M$ and $\ddot{q}_3^M$, one among them is the *real* acceleration upper bound. Fig. 2a depicts the six possible scenarios. The third case results in a disconnected set of feasible accelerations, which is impossible because $\mathcal{V}$ is connected. When $\ddot{q}_3^M > \ddot{q}_1^M$ (i.e. the first two cases) then $\ddot{q}_3^M$ is the upper bound. In the other three cases the upper bound is the minimum between $\ddot{q}_1^M$ and $\ddot{q}_2^M$.



(a) The six possible orderings of $\ddot{q}_1^M, \ddot{q}_2^M$ and $\ddot{q}_3^M$. The region in red represents the feasible acceleration region according to (22).

(b) The three possible orderings of $\ddot{q}_1, \ddot{q}_2$ and $\ddot{q}_3$. The region in red represents the feasible acceleration region according to (26).

Fig. 2: Possible orderings of the variables used in our algorithm.

Algorithm 1 accBoundsFromPosLimits

Require: $q, \dot{q}, q^{min}, q^{max}, \delta t$
$\ddot{q}_1^M \leftarrow -\dot{q}/\delta t$
$\ddot{q}_2^M \leftarrow -\dot{q}^2/(2(q^{max} - q))$
$\ddot{q}_3^M \leftarrow 2(q^{max} - q - \delta t\, \dot{q})/(\delta t^2)$
$\ddot{q}_2^m \leftarrow \dot{q}^2/(2(q - q^{min}))$
$\ddot{q}_3^m \leftarrow 2(q^{min} - q - \delta t\, \dot{q})/(\delta t^2)$
if $\dot{q} \geq 0$ **then**
  $\ddot{q}^{LB} \leftarrow \ddot{q}_3^m$
  if $\ddot{q}_3^M > \ddot{q}_1^M$ **then**
    $\ddot{q}^{UB} \leftarrow \ddot{q}_3^M$
  else
    $\ddot{q}^{UB} \leftarrow \min(\ddot{q}_1^M, \ddot{q}_2^M)$
else
  $\ddot{q}^{UB} \leftarrow \ddot{q}_3^M$
  if $\ddot{q}_3^m < \ddot{q}_1^M$ **then**
    $\ddot{q}^{LB} \leftarrow \ddot{q}_3^m$
  else
    $\ddot{q}^{LB} \leftarrow \max(\ddot{q}_1^M, \ddot{q}_2^m)$
return $\{\ \ddot{q}^{LB}, \ddot{q}^{UB}\ \}$

Repeating the same analysis for the lower bound we get the computation summarized by Algorithm 1.

D. Velocity Inequalities

Since the velocity trajectory is a line we just need to ensure that the velocity bounds be satisfied for $t = \delta t$:

$$\frac{1}{\delta t}(-\dot{q}^{max} - \dot{q}) \leq \ddot{q} \leq \frac{1}{\delta t}(\dot{q}^{max} - \dot{q}) \tag{23}$$

E. Viability Inequalities

Let us consider the upper bound of the viability inequality:

$$\begin{aligned} \dot{q}(\delta t) &\leq \sqrt{2\ddot{q}^{max}(q^{max} - q(\delta t))} \\ \dot{q} + \delta t\ddot{q} &\leq \sqrt{2\ddot{q}^{max}(q^{max} - q - \delta t\, \dot{q} - 0.5\delta t^2\ddot{q})} \end{aligned} \tag{24}$$

This constraint is clearly nonlinear in $\ddot{q}$, but we can derive an algorithm to reformulate it as a simple upper bound. Since

Algorithm 2 accBoundsFromViability

Require: $q, \dot{q}, q^{min}, q^{max}, \ddot{q}^{max}, \delta t$
$a \leftarrow \delta t^2$
$b \leftarrow \delta t(2\dot{q} + \ddot{q}^{max}\delta t)$
$c \leftarrow \dot{q}^2 - 2\ddot{q}^{max}(q^{max} - q - \delta t\,\dot{q})$
$\ddot{q}_1 \leftarrow -\dot{q}/\delta t$
$\Delta \leftarrow b^2 - 4ac$
if $\Delta \geq 0$ **then**
$\ddot{q}^{UB} \leftarrow \max(\ddot{q}_1, (-b + \sqrt{\Delta})/(2a))$
else
$\ddot{q}^{UB} \leftarrow \ddot{q}_1$
$b \leftarrow 2\delta t\,\dot{q} - \ddot{q}^{max}\delta t^2$
$c \leftarrow \dot{q}^2 - 2\ddot{q}^{max}(q + \delta t\,\dot{q} - q^{min})$
$\Delta \leftarrow b^2 - 4ac$
if $\Delta \geq 0$ **then**
$\ddot{q}^{LB} \leftarrow \min(\ddot{q}_1, (-b - \sqrt{\Delta})/(2a))$
else
$\ddot{q}^{LB} \leftarrow \ddot{q}_1$
return $\{\ \ddot{q}^{LB}, \ddot{q}^{UB}\ \}$

the right-hand side (RHS) of (24) is always positive, if the left-hand side (LHS) is negative then (24) is satisfied:

$$\dot{q} + \delta t\,\ddot{q} \leq 0 \quad \Leftrightarrow \quad \ddot{q} \leq -\frac{\dot{q}}{\delta t} \triangleq \ddot{q}_1 \tag{25}$$

If instead the LHS is positive we can take the square of both sides:

$$\begin{aligned} &(\dot{q} + \delta t\ddot{q})^2 \leq 2\ddot{q}^{max}(q^{max} - q - \delta t\,\dot{q} - 0.5\delta t^2\ddot{q})) \\ &a\ddot{q}^2 + b\ddot{q} + c \leq 0, \end{aligned} \tag{26}$$

where:

$$\begin{aligned} a &= \delta t^2 \\ b &= \delta t\,(2\dot{q} + \delta t\,\ddot{q}^{max}) \\ c &= \dot{q}^2 - 2\ddot{q}^{max}(q^{max} - q - \delta t\,\dot{q}) \end{aligned} \tag{27}$$

If $\Delta = b^2 - 4ac \geq 0$ this parabola is equal to zero in two points $\ddot{q}_2$ and $\ddot{q}_3$ (which coincide if $\Delta = 0$):

$$\ddot{q}_2 \triangleq \frac{-b - \sqrt{\Delta}}{2a}, \qquad \ddot{q}_3 \triangleq \frac{-b + \sqrt{\Delta}}{2a} \tag{28}$$

Since $a > 0$ by definition, the inequality (26) is satisfied for $\ddot{q}_2 \leq \ddot{q} \leq \ddot{q}_3$. If instead $\Delta < 0$ then the parabola is always positive and there exists no value of $\ddot{q}$ that satisfies (26). Putting it all together we can rewrite (24) as:

$$(\ddot{q} \leq \ddot{q}_1) \quad \vee \quad [(\ddot{q} > \ddot{q}_1) \wedge (\Delta \geq 0) \wedge (\ddot{q}_2 \leq \ddot{q} \leq \ddot{q}_3)] \tag{29}$$

Similarly to what we did for the position inequalities in Section III-C, we can analyse the different scenarios depending on the values of $\ddot{q}_1, \ddot{q}_2$ and $\ddot{q}_3$ (see Fig. 2b). Here we have only three possible scenarios because by definition $\ddot{q}_2 \leq \ddot{q}_3$. The third case results in a disconnected set of feasible accelerations, which is impossible because $\mathcal{V}$ is connected. In the other cases the bound is given by the maximum between $\ddot{q}_1$ and $\ddot{q}_3$. Repeating the same analysis for the lower bound, we get the computation summarized by Algorithm 2.

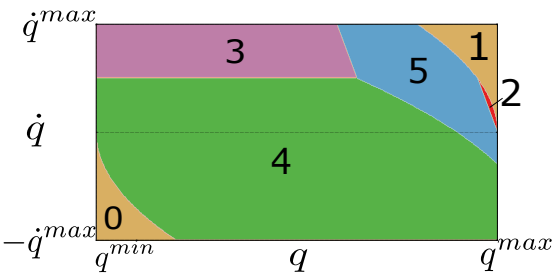


Fig. 3: Feasible state space for $q^{max} = -q^{min} = 0.5$ rad, $\dot{q}^{max} = 2$ rad/s, $\ddot{q}^{max} = 10$ rad/s^2, and $\delta t = 0.1$s. Region 0 represents the unreachable space, while region 1 represents the space that is not viable. In each of the other four regions, a different acceleration upper bound dominates the others. In region 2 it is the one coming from the position inequality (13). In region 3 it is the one coming from the velocity inequality (23). In region 4 it is the acceleration upper bound $\ddot{q}^{max}$. In region 5 it is the one coming from the viability inequality (24).

Algorithm 3 Compute Joint Acceleration Bounds

Require: $q, \dot{q}, q^{min}, q^{max}, \dot{q}^{max}, \ddot{q}^{max}, \delta t$
$\ddot{q}^{UB} \leftarrow [0, 0, 0, \ddot{q}^{max}]$
$\ddot{q}^{LB} \leftarrow [0, 0, 0, -\ddot{q}^{max}]$
$(\ddot{q}^{LB}[0], \ddot{q}^{UB}[0]) \leftarrow$ accBoundsFromPosLimits(...)
$\ddot{q}^{LB}[1] \leftarrow (-\dot{q}^{max} - \dot{q})/\delta t$
$\ddot{q}^{UB}[1] \leftarrow (\dot{q}^{max} - \dot{q})/\delta t$
$(\ddot{q}^{LB}[2], \ddot{q}^{UB}[2]) \leftarrow$ accBoundsFromViability(...)
return $\{\max(\ddot{q}^{LB}), \min(\ddot{q}^{UB})\}$

F. Final Algorithm to Solve (4)

Finally, to compute the solution of our original problem (4), we just need to compute all the upper and lower bounds of $\ddot{q}$, and then select the minimum among the upper bounds and the maximum among the lower bounds. This procedure is summarized by Algorithm 3.

Fig. 3 shows the feasible state space divided into six different regions. Apart from the non-viable region and the non-reachable region, in each of the other four regions there is a different active constraint in (12). Note that the size of these regions depends on the values of the time step and the position, velocity and acceleration bounds. In some cases certain regions may even disappear.

IV. Torque Control

During the last 15 years robots are switching from velocity/acceleration control to torque control [13], [14]. This section presents one way to apply the methodology proposed in the previous section to a torque-controlled robot. In order to do so we have to face two issues.

First, so far we have assumed a constant acceleration during the time step, while now we have a constant torque. Considering an n-degree-of-freedom manipulator, the relationship between torques and accelerations is given by:

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}), \tag{30}$$

Algorithm 4 Estimate Acceleration Bounds From Torque Bounds

Require: $\mathbf{q}^{min}, \mathbf{q}^{max}, \dot{\mathbf{q}}^{max}, \boldsymbol{\tau}^{max}, N$
  $\ddot{\mathbf{q}}^{UB} \leftarrow [\infty, \dots, \infty]$
  $\ddot{\mathbf{q}}^{LB} \leftarrow -[\infty, \dots, \infty]$
  for $i = 1 \rightarrow N$ **do**
    $\mathbf{q} \leftarrow \text{random}(\mathbf{q}^{min}, \mathbf{q}^{max})$
    $\dot{\mathbf{q}} \leftarrow \text{random}(-\dot{\mathbf{q}}^{max}, \dot{\mathbf{q}}^{max})$
    $M \leftarrow \text{computeMassMatrix}(\mathbf{q})$
    $\mathbf{h} \leftarrow \text{computeBiasForces}(\mathbf{q}, \dot{\mathbf{q}})$
    for $j = 1 \rightarrow \text{size}(\mathbf{q})$ **do**
      $\ddot{\mathbf{q}}^{max} \leftarrow (\boldsymbol{\tau}^{max}[j] - \mathbf{h}[j])/M[j,j]$
      $\ddot{\mathbf{q}}^{min} \leftarrow (-\boldsymbol{\tau}^{max}[j] - \mathbf{h}[j])/M[j,j]$
      $\ddot{\mathbf{q}}^{UB}[j] \leftarrow \min(\ddot{\mathbf{q}}^{UB}[j], \ddot{\mathbf{q}}^{max})$
      $\ddot{\mathbf{q}}^{LB}[j] \leftarrow \max(\ddot{\mathbf{q}}^{LB}[j], \ddot{\mathbf{q}}^{min})$
  return $\{\ddot{q}^{LB}, \ddot{q}^{UB}\}$

where $\boldsymbol{\tau} \in \mathbb{R}^n$ are the joint torques, $\ddot{\mathbf{q}} \in \mathbb{R}^n$ are the joint accelerations, $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n \times n}$ is the inertia matrix, $\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^n$ are the bias forces. According to (30) a constant torque does not imply a constant acceleration because $\mathbf{q}$ and $\dot{\mathbf{q}}$ change during the time step. However, since time steps are typically short (i.e. between 1 and 10 ms), $\mathbf{h}$ and $\mathbf{M}$ do not change much between the beginning and the end of the time step. Hence we can assume that accelerations are approximately constant throughout the time step.

The second issue is that joint accelerations are not directly bounded, but they are indirectly bounded by the torque limits τ^{max}. This means that the acceleration bounds depend on the system state. Trajectory generation under torque constraints requires solving a complex optimization problem [15], which we cannot afford inside a control loop. We suggest a conservative solution, that is to estimate offline *worst-case* acceleration bounds and use them in our algorithm.

Algorithm 4 shows a sampling-based procedure to estimate these bounds: i) we generate a large number N of random states, ii) for each state we compute the upper/lower acceleration bound at each joint, assuming that the other joint accelerations are zero, iii) for each joint we select the worst-case bounds. This procedure is not completely conservative because it computes the acceleration bounds for a certain joint assuming that the other joint accelerations are zero—while this is unlikely the case in reality. However, typically only one joint at a time will have its state on the boundaries of the viable set $\mathcal{V}$. When this happens, only this joint requires maximum acceleration to stop before its position limit, and the tracking at the other joints can be temporarily degraded to accommodate this need. However, if needed, a more conservative procedure to estimate worst-case acceleration bounds from torque bounds can be used.

V. Simulations

In this section we validate the proposed method through numerical simulations with the 7-degree-of-freedom arm of the Baxter robot[1]. In both our tests we compare the naive approach to bound accelerations (2) with the approach proposed in this paper. Our first test focuses on joint-space acceleration control, while in our second test the robot joints are torque controlled and the task is executed by the end-effector in Cartesian-space. All the kinematics and dynamics quantities have been computed with the Pinocchio library [16].

A. Joint-Space Acceleration Control

In this test we use the following joint-space acceleration controller:

$$\begin{aligned} \underset{\ddot{\mathbf{q}}}{\text{minimize}} \quad & ||\ddot{\mathbf{q}}^d - \ddot{\mathbf{q}}||^2 \\ \text{subject to} \quad & \ddot{\mathbf{q}}^{lb} \leq \ddot{\mathbf{q}} \leq \ddot{\mathbf{q}}^{ub} \end{aligned} \tag{31}$$

The desired joint accelerations are computed as:

$$\ddot{\mathbf{q}}^d = k_p(\mathbf{q}^r - \mathbf{q}) - k_d\dot{\mathbf{q}}, \tag{32}$$

where $k_p, k_d \in \mathbb{R}^+$ are the proportional and derivative gains. The values used for the simulations are $k_p = 1000$, $k_d = 2\sqrt{k_p}$, $\delta t = 0.01$ s. We carried out several simulations using different methods to compute the acceleration bounds $\ddot{\mathbf{q}}^{lb}, \ddot{\mathbf{q}}^{ub}$ and verified which ones were successful in preventing constraint violations. To incite the hitting of a joint bound we set the reference position $\mathbf{q}^r$ for the first joint outside its valid range (i.e. 0.1 radians above its upper bound). The results are summarized by Fig. 4.

1) Naive Acceleration Bounds: Using the naive approach (2), the first joint violated the upper position bound, as shown in Fig. 4a. Fig. 4b shows that using a larger value of δt for computing the acceleration bounds [1], [2] helps mitigating the issue, but it does not completely solve it. The impact velocity is reduced, but the constraint is still violated.

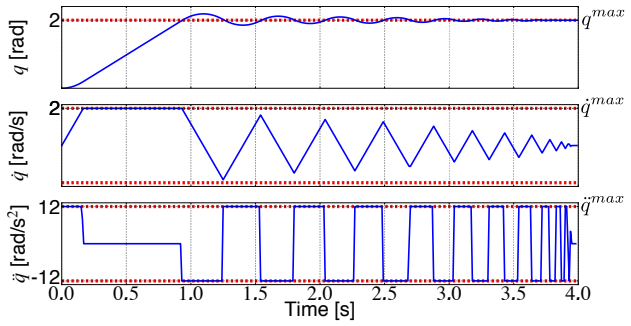
2) Viability-Based Acceleration Bounds: Using our algorithm, position and velocity bounds are reached multiple times, but never violated, as shown by Fig. 4c. The system never reached the steady state $(q^{max}, 0)$—as it would if the controller were continuous time. The joint velocity oscillates around zero, while its acceleration is discontinuous. Even if unpleasant on a real system, this behavior is *sound*, because we did not bound jerk. We can easily get rid of these discontinuities by using larger values of δt for computing the acceleration bounds, as shown in Fig. 4d.

B. Cartesian-Space Torque Control

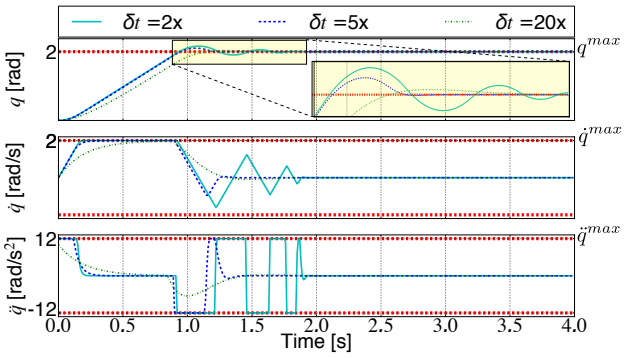
In this simulation we show the importance of correctly bounding the joint accelerations during task-space control. We used a task-space inverse dynamics controller [17] to reach a desired configuration with the end-effector of Baxter's arm. At the same time, a lower-priority joint-space task is used to stabilize the null space of the end-effector. The controller also guarantees the torque and acceleration limits. At each control loop we computed the desired joint torques by solving the following Quadratic Program:

$$\begin{aligned} \underset{\boldsymbol{\tau}, \ddot{\mathbf{q}}}{\text{minimize}} \quad & ||\dot{\mathbf{v}}^d - \mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}||^2 + w||\ddot{\mathbf{q}}^p - \ddot{\mathbf{q}}||^2 \\ \text{subject to} \quad & \boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) \\ & \ddot{\mathbf{q}}^{lb} \leq \ddot{\mathbf{q}} \leq \ddot{\mathbf{q}}^{ub}, \qquad |\boldsymbol{\tau}| \leq \boldsymbol{\tau}^{max}, \end{aligned} \tag{33}$$

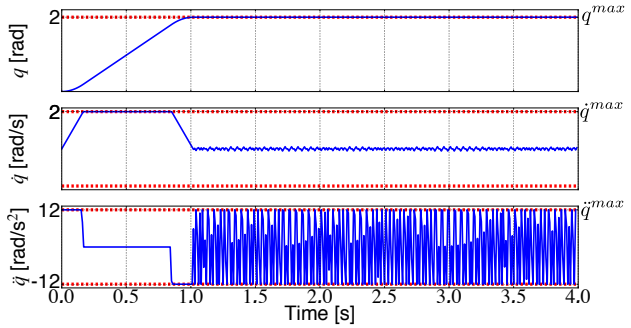
[1] http://www.rethinkrobotics.com/baxter/



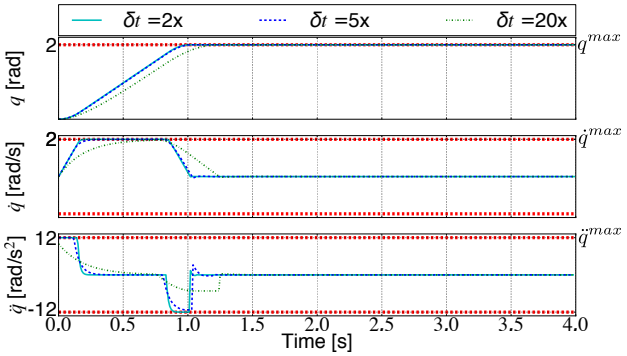
(a) Acceleration bounds computed with the naive approach (2) and the same δt of the controller.



(b) Acceleration bounds computed with the naive approach (2) and larger values of δt than the controller.

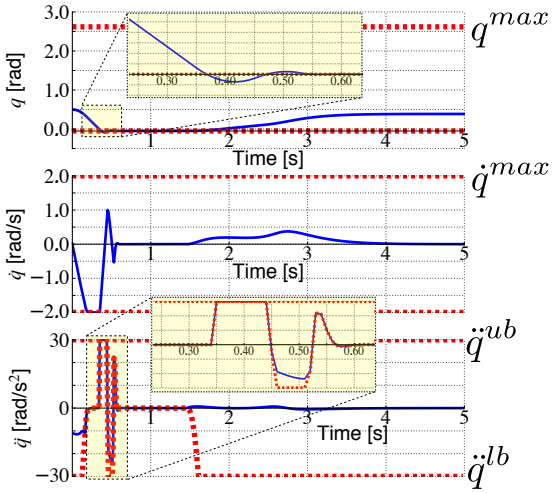


(c) Acceleration bounds computed with the approach proposed in this paper and the same δt of the controller. Even if these acceleration discontinuities may be undesirable on a real system, this behavior is *sound* because we did not limit the jerk of our trajectory. Using a larger value of δt in the acceleration bound computation makes them disappear.

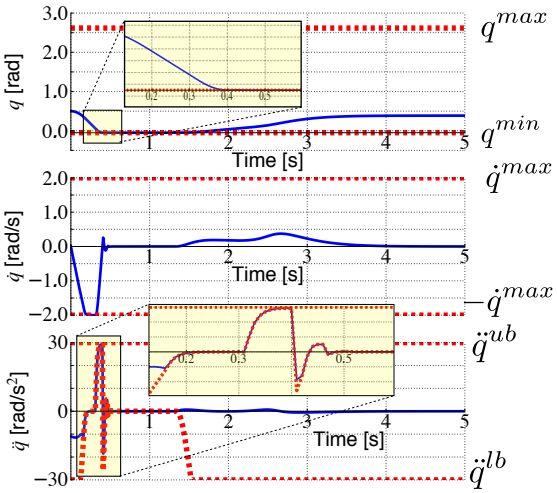


(d) Acceleration bounds computed with the approach proposed in this paper and larger values of δt than the controller.

Fig. 4: Trajectories of the first joint of the Baxter arm obtained by using different techniques to compute the acceleration bounds. For this simulation we had $\delta t = 0.01$ s, $q^{max} = 2$ rad, $\dot{q}^{max} = 2$ rad/s, $\ddot{q}^{max} = 12$ rad/s^2.



(a) Using the naive approach the joint reaches its lower position bound at 0.36 s with a velocity of -1.4 rad/s.



(b) Using the proposed method position, velocity and acceleration bounds are satisfied throughout the whole trajectory.

Fig. 5: Trajectory of the elbow joint using either the naive approach or our approach to compute the joint acceleration bounds. In both cases we used a larger δt ($2\times$) than the controller to avoid discontinuities.

where $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{6\times n}$ is the end-effector Jacobian, and $w \in \mathbb{R}$ is the postural-task weight. The desired end-effector acceleration and joint accelerations are computed as:

$$\begin{aligned} \dot{\mathbf{v}}^d &= k_p \log(\mathbf{H}(\mathbf{q})^{-1}\mathbf{H}^d) - k_d \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \\ \ddot{\mathbf{q}}^p &= k_p(\mathbf{q}^r - \mathbf{q}) - k_d \dot{\mathbf{q}}, \end{aligned} \tag{34}$$

where $k_p, k_d \in \mathbb{R}^+$ are the proportional and derivative gains, and $\mathbf{H}, \mathbf{H}^d \in SE(3)$ are respectively the measured and desired rigid transformation from world to end-effector frame. The values used for this simulations are $k_p = 10$, $k_d = 2\sqrt{k_p}$, $\delta t = 0.01$ s, $w = 10^{-3}$.

We selected the initial and final pose of the end-effector so that the elbow joint moves towards its lower bound. We carried out several simulations using different methods to compute the acceleration bounds $\ddot{\mathbf{q}}^{lb}, \ddot{\mathbf{q}}^{ub}$ and checked for violations of joint limits. The acceleration bounds were estimated offline from the torque bounds using Algorithm 4 with $N = 10^6$ (about 5 minutes of computation time).

1) Naive Acceleration Bounds: In our first simulation we computed $\ddot{\mathbf{q}}^{lb}$ and $\ddot{\mathbf{q}}^{ub}$ using the naive method (2). We computed the acceleration bounds with a value of δt twice as large as the controller to avoid discontinuities in the joint accelerations. The robot managed to reach the desired pose with its end-effector, but the elbow joint reached its lower position bound with a large velocity of 1.4 rad/s (as shown by Fig. 5a). On a real robot this could have seriously damaged the system. We also tried using a much larger value of δt ($20\times$). While this helped reducing the impact velocity of the elbow joint to 0.3 rad/s, it did not prevent the impact.

2) Viability-Based Acceleration Bounds: We then computed $\ddot{\mathbf{q}}^{lb}$ and $\ddot{\mathbf{q}}^{ub}$ using the method proposed in this paper (i.e. Algorithm 3). Similarly to the previous test, we used a larger δt ($2\times$) to compute the acceleration bounds. Also in this case the end-effector reached the desired pose, but no constraint was violated: Fig. 5b shows that the elbow joint safely reached its lower bound with zero velocity.

VI. Conclusions

This paper discussed the control of robots whose joints have bounded positions, velocities and accelerations. Despite the well-known difficulty of satisfying position, velocity and acceleration bounds [6], [7], [3], [1], no exact solution had been proposed yet. We found the maximum and minimum current joint accelerations that guarantee the existence of a feasible future trajectory. This problem can be formulated as an untractable infinite-horizon optimal control problem. We reformulated it using the concept of viability and derived an efficient algorithm to solve it. We also proposed a pragmatic way to apply this technique to torque-controlled robots, whose accelerations are only indirectly bounded by the torque limits. Simulations on the Baxter arm showed the interest of using our approach to guarantee the satisfaction of the joint bounds.

Despite its clear utility, the proposed algorithm has several limitations, which should be addressed in future work.

- The algorithm assumes constant bounds. Even though we proposed an approach to deal with state-dependent bounds, this approach is conservative and further research is necessary to fully exploit the robot capabilities.
- An exact knowledge of the state and the model of the robot was assumed, hence the computed bounds are not robust to uncertainties.
- We did not deal with the case of physical interaction with the environment, in which unknown external forces may lead to constraint violations. Moreover, rigid contacts constrain the robot motion, coupling all the joints, thus the proposed method would not be directly applicable.
- We did not account for jerk limits, which would be certainly useful in practice, even though we expect them to substantially increase the algorithm complexity [4].

References

[1] K. C. Park, P. H. Chang, and S. H. Kim, "The Enhanced Compact QP Mathod for Redundant Manipulators Using Practical Inequality Constraints," in *IEEE International Conference on Robotics and Automation (ICRA)*, 1998.

[2] L. Saab, O. E. Ramos, N. Mansard, P. Soueres, and J.-y. Fourquet, "Dynamic Whole-Body Motion Generation under Rigid Contacts and other Unilateral Constraints," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 346–362, 2013.

[3] F. Kanehiro, M. Morisawa, W. Suleiman, K. Kaneko, and E. Yoshida, "Integrating geometric constraints into reactive leg motion generation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[4] T. Kröger and F. M. Wahl, "On-Line trajectory generation in robotic systems. Basic concepts for instananeous reactions to unforseen (sensor) events," *IEEE Transaction on Robotics*, vol. 26, no. 1, pp. 94–111, 2010.

[5] S. Rubrecht, V. Padois, P. Bidaud, and M. De Broissia, "Constraints Compliant Control : constraints compatibility and the displaced configuration approach," in *IEEE/RSJ Intenational Conference on Intelligent Robots and Systems (IROS)*, 2010.

[6] W. Decré, 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 (ICRA)*, 2009.

[7] S. Rubrecht, V. Padois, P. Bidaud, M. Broissia, and M. Da Silva Simoes, "Motion safety and constraints compatibility for multibody robots," *Autonomous Robots*, vol. 32, no. 3, pp. 333–349, 2012.

[8] P. M. Wensing and D. E. Orin, "Generation of Dynamic Humanoid Behaviors Through Task-Space Control with Conic Optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[9] T. Koolen, J. Smith, G. Thomas, S. Bertrand, et al., "Summary of Team IHMC ' s Virtual Robotics Challenge Entry," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2013.

[10] Q. Nguyen and K. Sreenath, "Exponential Control Barrier Functions for enforcing high relative-degree safety-critical constraints," in *American Control Conference*, 2016.

[11] M. Rauscher, M. Kimmel, and S. Hirche, "Constrained Robot Control Using Control Barrier Functions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[12] J.-P. Aubin., *Viability Theory*. Birkhauser, 1990.

[13] G. Hirzinger and A. Albu-Schäffer, "On a new generation of torque controlled light-weight robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2001.

[14] A. Herzog, N. Rotella, S. Mason, F. Grimminger, S. Schaal, and L. Righetti, "Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 473–491, 2016.

[15] J. E. Bobrow, S. Dubowsky, and J. S. Gibson, "Time-Optimal Control of Robotic Manipulators Along Specified Paths," *The International Journal of Robotics Research*, vol. 4, no. 3, pp. 3–17, 1985.

[16] N. Mansard, "http://stack-of-tasks.github.io/pinocchio," 2016.

[17] A. Del Prete, F. Nori, G. Metta, and L. Natale, "Prioritized Motion-Force Control of Constrained Fully-Actuated Robots: "Task Space Inverse Dynamics"," *Robotics and Autonomous Systems*, vol. 63, pp. 150–157, 2015.