Real-time path-tracking control of robotic manipulators with bounded torques and torque-derivatives

Corrado GUARINO LO BIANCO* and Oscar GERELLI

Abstract—Minimum-time path-tracking control of robotic manipulators assumes a relevant role in industrial applications where efficiency is an issue. On the other hand, minimizing the traveling time leads to an increment of the mechanical solicitations: the actuators dynamic limits can be easily exceeded. For this reasons, kinematic and/or dynamic constraints are normally taken into account when planning optimal trajectories through off-line algorithms. Nevertheless this precaution, dynamic limits can be easily violated during actual operations due to model uncertainties and the action of the feedback controller. In order to fulfill with certainty the given constraints, planned trajectories are typically online scaled by means of dynamic filters. Normally, this is done by only considering torque constraints. On the contrary, in this paper, the trajectory is online scaled by also taking into account the existence of bounds on the torque derivatives. Indeed, torque derivatives have a direct impact both on the mechanical solicitations and on the tracking accuracy. A new nonlinear filter is proposed for the optimal trajectory scaling. Its effectiveness is verified by means of simulations.

Index Terms—Path tracking, online trajectory scaling, torque bounded control, torque-derivative bounded control.

I. INTRODUCTION

Robotic manipulators are commonly used in many industrial applications which need the generation of sophisticated motions, usually along a specified paths. The manipulator end effector is expected to perfectly track reference paths even when high speed movements are required. Indeed, in most industrial applications the production rate is limited by the robot performances rather than the process constraints and this motivates the research of minimum-time controls. Unfortunately, this requirement collides with the physical limits of the mechanical structures which must be necessarily taken into account.

Classically, time-optimal trajectories are off-line planned by means of optimization algorithms which consider the kinematic and/or dynamic constraints. Two different approaches can be observed. In the first one, given some via points, the path and the control inputs are determined as a whole by the optimization algorithm [1]–[4]. In the second one, the path-velocity decomposition method is used [5], so that the solution is obtained by computing an optimal longitudinal velocity profile to be used to move along an assigned path [6].

* Corresponding author
The authors are with the Dip. di Ing. dell’Informazione, University of Parma, I-43100 Parma, Italy, Tel. +39 0521 905725, Fax. +39 0521 905723, Email: {guarino,gerelli}@ce.unipr.it

Independently from the used approach and according to the Pontryagin Maximum Principle, when minimum-time solutions are considered there is at least one joint that, at any time, is working on the boundary of the feasible region: any external disturbance or unmodelled behavior cannot be compensated, so that the manipulator may deviate from the desired path. Several approaches have been proposed to address this issue. In [7] it has been introduced for the first time the idea, further developed in [8], of the admissible path velocity and the admissible regions in the phase plane in order to take care of the torque bounds, while practical applications of this method have been proposed in [9] and [10], where an online time-scaling control is devised to compensate for unmodeled effects. To this aim, [11] proposed a perturbation control via a linear-quadratic-Gaussian solution and in [12] a theory has been introduced in order to adapt classical time-optimal control methods for tracking paths with a predefined tolerance.

This paper reconsiders the method proposed in [10] for the online trajectory scaling by introducing bounds on the torque derivatives. Indeed, it is well known that high torques derivatives solicit the manipulators unmodelled dynamics, thus decreasing the controller effectiveness, and cause high mechanical stresses which, in turn, can induce wear and tear in the robot actuators or gears. Moreover, owing to mechanical and electrical limits, real actuators can only generate limited torque variations, so that path tracking is lost with certainty every time torque derivatives exceed the given limits.

In the paper it will be shown how, given a path and due to the added constraints, the admissible region becomes a volume in the $(s,\dot{s},\ddot{s})$-space. Dynamic constraints are managed by online converting torque and torque-derivative bounds into equivalent bounds on the longitudinal acceleration and jerk. To this purpose, a recently devised iterative algorithm, which efficiently evaluates the manipulators high order dynamics [13], has been used. The desired trajectory scaling is then obtained by using a novel discrete-time filter which modifies the longitudinal velocity profile such to fulfills the given bounds. Owing to its efficiency, the algorithm is suited to be online implemented.

The paper is organized as follows. The trajectory planning problem is discussed in §II. Then, in §III, the manipulator controller is introduced together with the method adopted for the online evaluation of the bounds on the longitudinal acceleration and jerk. The online trajectory scaling filter is
described in §IV, while simulation results are proposed in §V. Finally, in §VI some concluding remarks are drawn.

Notation: Vectors are indicated by means of lower case bold character (e.g. \( \mathbf{q} \)), while matrices are indicated by capital bold characters (e.g. \( \mathbf{H} \)). The set of positive reals be denoted by \( \mathbb{R}^+ \). Given \( \mathbf{A} \in \mathbb{R}^{n \times n} \) and \( \mathbf{B} \in \mathbb{R}^{p \times q} \), \( \mathbf{C} = \mathbf{A} \otimes \mathbf{B} \in \mathbb{R}^{mp \times nq} \) denotes the Kronecker product. \( \mathbf{I}_q \in \mathbb{R}^{q \times q} \) denotes the identity matrix of order \( q \).

II. PROBLEM FORMULATION

The control strategy proposed in this paper is based on the so called path-velocity decomposition [5]: a robot trajectory is obtained by first planning a path to be followed and, then, by generating a velocity profile to move along such path.

Paths can be indifferentely planned in the task space or in the joint space. In fact, it is always possible to convert a task space path into a joint space one by means of an appropriate use of the manipulator jacobian matrix \( \mathbf{J}(\mathbf{q}) \) [14]. In case of non-redundant manipulators, the conversion is straightforward since the jacobian matrix has maximum rank, so that task space and joint space planning can be considered strictly equivalent. For these reasons and without any loss of generality, let us define a parametric curve in the joint space by means of a vector function \( \mathbf{f}(s) \) defined as follows

\[
\mathbf{f} : [0,s_f] \rightarrow \mathbb{R}^n
\]

\[
s \rightarrow \mathbf{q}_d := \mathbf{f}(s) .
\]

(1)

where \( s \) is the curvilinear coordinate along the path, defined as the distance from the beginning of the curve, \( s_f \) is the curve length, and \( n \) is the number of independent joints. In the same way it is possible to define a monotonically increasing time law to be used to move along the curve \n
\[
s : [0,t_f] \rightarrow [0,s_f]
\]

(2)

\[
t \rightarrow s_d := s(t) .
\]

where \( t_f \) is the total travelling time. Necessarily, \( s_d(t) > 0 \). The overall robot trajectory is obtained by combining (1) and (2); \( \mathbf{q}_d := \mathbf{f}(s_d(t)) \). By taking into account the chain differentiation rule, it is possible to evaluate the trajectory time derivatives

\[
\dot{\mathbf{q}}_d = f'(s)\mathbf{s} ,
\]

\[
\ddot{\mathbf{q}}_d = f''(s)\mathbf{s}s + f'(s)\mathbf{s}^2 ,
\]

\[
\dddot{\mathbf{q}}_d = f'''(s)\mathbf{s}s^3 + 3f''(s)\mathbf{s}s^2 + f'(s)\mathbf{s}s .
\]

(3)

(4)

(5)

where superscript \( \cdot \) indicates a differentiation with respect to \( s \), e.g., \( f'(s) = \frac{df}{ds} \), while, as usual, dots indicate time derivatives, e.g., \( \dot{s} = \frac{ds}{dt} \).

Trajectory \( \mathbf{q}_d := \mathbf{f}(s(t)) \) can be planned in several ways depending on the robot task. Typically it is obtained by minimizing the travelling time subject to some given kinematic and/or dynamic constraints. The most commonly adopted dynamic constraint is represented by maximum admissible torques: any planned trajectory can be followed only if all joint torques \( \tau_i \) are below given limits

\[
\mathbf{z}_i \leq \tau_i \leq \mathbf{z}_i , \quad i = 1,2,\ldots,n
\]

(6)

where \( \mathbf{z}_i \) and \( \mathbf{z}_i \) represent the lower and upper bounds on the \( i \)-th joint torque.

In this paper, the problem is deepened by also bounding joint torque derivatives \( \dot{\tau}_i \)

\[
\dot{\mathbf{z}}_i \leq \dot{\tau}_i \leq \dot{\mathbf{z}}_i , \quad i = 1,2,\ldots,n
\]

(7)

where \( \dot{\mathbf{z}}_i \) and \( \dot{\mathbf{z}}_i \) represent the lower and upper bounds on the \( i \)-th joint torque derivative.

The following problem immediately arises: given a trajectory \( \mathbf{q}_d := \mathbf{f}(s(t)) \), is it possible to verify its feasibility with respect to (6) and (7)? In order to answer to this question let us consider the rigid body dynamics of a serial-link manipulator

\[
\tau = \mathbf{H}(\mathbf{q}) \ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{F}(\mathbf{q})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q})
\]

(8)

where \( \mathbf{q} \in \mathbb{R}^n \) is the vector of the joint displacements, \( \tau \in \mathbb{R}^n \) is the vector of the applied torques, \( \mathbf{H}(\mathbf{q}) \in \mathbb{R}^{n \times n} \) is the symmetric positive-definite inertia matrix, \( \mathbf{c}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^n \) is the vector of Coriolis and centripetal forces, \( \mathbf{F}(\mathbf{q}) \in \mathbb{R}^{n \times n} \) describes the viscous friction and \( \mathbf{g}(\mathbf{q}) \in \mathbb{R}^n \) is the vector of gravitational forces. In order to verify the feasibility of the trajectory with respect to (7), an analytical representation of \( \tau \) is required. It can be obtained from (8) by means of the following differentiation rule (see also [4])

\[
\frac{d}{dt} \left[ \mathbf{A}(\mathbf{x}(t)) \right] = \frac{\partial \mathbf{A}(\mathbf{x}(t))}{\partial \mathbf{x}} \left( \mathbf{I}_q \otimes \dot{\mathbf{x}}(t) \right)
\]

where \( \mathbf{A}(\mathbf{x}(t)) \in \mathbb{R}^{p \times q} \) and \( \mathbf{x}(t) := [x_1(t) \ldots x_m(t)]^T \in \mathbb{R}^m \). By differentiating (8) with respect to time, it is possible to obtain

\[
\dot{\tau} = \frac{\partial \mathbf{H}(\mathbf{q})}{\partial \mathbf{q}} (\mathbf{I}_n \otimes \dot{\mathbf{q}}) + \mathbf{H}(\mathbf{q})\dddot{\mathbf{q}} + \frac{\partial \mathbf{c}(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}} \dot{\mathbf{q}} + \frac{\partial \mathbf{c}(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}} \ddot{\mathbf{q}} + \frac{\partial \mathbf{F}(\mathbf{q})}{\partial \mathbf{q}} (\mathbf{I}_n \otimes \dot{\mathbf{q}}) + \mathbf{F}(\mathbf{q})\dot{\mathbf{q}} + \frac{\partial \mathbf{g}(\mathbf{q})}{\partial \mathbf{q}} \dot{\mathbf{q}}
\]

which can be synthetically rewritten as

\[
\dot{\tau} = \mathbf{H}(\mathbf{q}) \dddot{\mathbf{q}} + \mathbf{G}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{D}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}
\]

(9)

by defining

\[
\mathbf{D}(\mathbf{q},\dot{\mathbf{q}}) := \frac{\partial \mathbf{c}(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}} \dot{\mathbf{q}} + \frac{\partial \mathbf{F}(\mathbf{q})}{\partial \mathbf{q}} (\mathbf{I}_n \otimes \dot{\mathbf{q}}) + \frac{\partial \mathbf{g}(\mathbf{q})}{\partial \mathbf{q}} \dot{\mathbf{q}}
\]

\[
\mathbf{G}(\mathbf{q},\dot{\mathbf{q}}) := \frac{\partial \mathbf{H}(\mathbf{q})}{\partial \mathbf{q}} (\mathbf{I}_n \otimes \dot{\mathbf{q}}) + \frac{\partial \mathbf{c}(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}} \dot{\mathbf{q}} + \mathbf{F}(\mathbf{q})
\]

where \( \mathbf{D}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^{n \times n} \) and \( \mathbf{G}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^{n \times n} \). Due to (1)–(5), equations (8)–(9) can be rewritten as functions of the curvilinear coordinate \( s \) and its derivatives

\[
\tau(s,\dot{s},\ddot{s}) = \mathbf{b}_1(s,\dot{s})\dot{s} + \mathbf{b}_2(s,\dot{s})
\]

(10)

\[
\dot{\tau}(s,\dot{s},\dddot{s}) = \mathbf{c}_1(s,\dddot{s}) + \mathbf{c}_2(s,\dot{s},\dddot{s})
\]

(11)
The resulting trajectory is feasible if and only if triplet $s = (s, \dot{s}, \ddot{s})$ belongs to $A$. This can mainly happen for two reasons. In the first scenario, for example, it is very common to plan time optimal trajectories by solving the following problem: given a curve $q = f(s)$ in the joint space, find the optimal profile $s(t), t \in [0, t_f]$, which minimizes the traveling time $t_f$ subject to the n-link robot dynamics described by (8) and the actuator constraints (6) and (7). Time optimal trajectories are normally off-line evaluated by means of optimization algorithms, such those described in [4], [15]. The result is a trajectory which has bang-bang characteristics, that is, there is always at least one robot joint working at its dynamic limits. This corresponds to a point $(s(t), \dot{s}(t), \ddot{s}(t))$ which is constantly moving along the boundary surfaces of region $A$. Due to model uncertainties or external disturbances the point could abandon the feasible area, so that trajectory tracking is lost.

Another common situation which causes the lost of tracking arises when the trajectory is programmed by an operator. Normally, dynamic constraints are not considered in the planning phase, thus the resulting trajectory could be infeasible.

When path tracking is a priority, online trajectory scaling algorithms are able to overcome these issues by slowing down reference trajectories. Generally a two level controller is used: an inner torque controller, designed with standard techniques to achieve the best transient responses, and an outer trajectory time-scaling controller, which slows down the reference velocity along the curve, in order to fulfill the dynamic constraints. In this paper, time-scaling is managed by a discrete time filter controlled with a nonlinear dynamic feedback. A sketch of the proposed control strategy is given in Fig. 2. The manipulator is driven by a standard torque feedforward controller with position and velocity feedback, whose output signals are saturated both in amplitude and slew rate. The same controller evaluates, depending on the current status of motion, appropriate acceleration and jerk bounds in order to fulfill the dynamic constraints on the maximum torque and torque derivative. The trajectory scaling filter modifies the reference velocity in order to satisfy such bounds. Controller and bound evaluation problem are analyzed in §III, while §IV is devoted to the trajectory scaling filter.

### III. Controller parametrization and online bounds evaluation

Any standard torque controller can be parametrized with respect to the curvilinear coordinate $s$ by means of (1)–(5). Examples of such parametrizations are given in [10]. In this

![Proposed trajectory control scheme](image-url)
paper, a feedforward controller with position and velocity feedback is used. As known, its outputs are evaluated according to
\[
\tau(q_d, \dot{q}_d, \ddot{q}_d) = H(q_d) \dot{q}_d + c(q_d, \dot{q}_d) + F(q_d) \ddot{q}_d + g(q_d) + K_p \epsilon + K_i \dot{\epsilon}
\]  
(17)
where $K_p, K_i \in (\mathbb{R}^+)^n$ are the controller gain vectors and $c := q - q_d$, $\dot{\epsilon} := \dot{q} - \dot{q}_d$ are, respectively, the trajectory tracking error vector and its derivative. By considering (1)–(4), controller (17) can be easily converted into
\[
\tau(s, \dot{s}, \ddot{s}) = b_1(s) \ddot{s} + b_2(s, \dot{s})
\]  
(18)
where $b_1(s)$ is defined according to (12), while
\[
b_2(s, \dot{s}) := b_2(s, \dot{s}) + K_p e + K_i \dot{e}
\]  
(19)
with $b_2(s, \dot{s})$ defined according to (13).

The two vectors $b_1(s) := [b_{1,1} b_{1,2} \cdots b_{1,n}]^T$ and $b_2(s, \dot{s}) := [b_{2,1} b_{2,2} \cdots b_{2,n}]^T$ are evaluated at each iteration of the control algorithm on the basis of the reference position along the path $s(t)$ and the tracking error $e$. Typically, an efficient iterative Newton-Euler algorithm recently proposed in [13]. The additional computational burden needed for the evaluation of $c_1(s)$ and $c_2(s, \dot{s}, \ddot{s})$ is comparable with the computational burden required for the evaluation of $b_1(s)$ and $b_2(s, \dot{s})$, the resulting overall procedure is suitable for an online implementation. In order to satisfy (7) it is necessary to impose
\[
\ddot{s} \leq c_{1,i} \dddot{s} + c_{2,i} \ddot{s} \leq \dddot{s}_i, \quad i = 1, 2, \ldots, n.
\]  
(25)
Evidently, $\dddot{s}$ is feasible if $\dddot{s} \in \bigcup_{i=1}^n [\dddot{s}_i, \dddot{s}_i]$, with
\[
\dddot{s}_i = \left\{ \begin{array}{ll}
\dddot{s}_i-c_{1,i} \dddot{s}_i \ddot{s}_i & \text{if } c_{1,i} > 0 \\
\dddot{s}_i-c_{1,i} \dddot{s}_i & \text{if } c_{1,i} < 0 \end{array} \right.
\]
and $\dddot{s}_i = \left\{ \begin{array}{ll}
\dddot{s}_i-c_{1,i} \dddot{s}_i & \text{if } c_{1,i} > 0 \\
\dddot{s}_i-c_{1,i} \dddot{s}_i & \text{if } c_{1,i} < 0 \end{array} \right.$
(26)
or, equivalently, if $\dddot{s} \in [U^-, U^+]$ where
\[
U^+ := \min_{i=1}^n \{ \dddot{s}_i \}, \quad U^- := \max_{i=1}^n \{ \dddot{s}_i \}.
\]  
(27)
Also in this case, $U^+ > U^-$ only if triplet $(s, \dot{s}, \dddot{s})$ is feasible, otherwise there does not exist any solution which fulfills the torque derivative constraints.

It is worth noticing that $S^+, S^-, U^+$, and $U^-$ are evaluated by not only considering the manipulator dynamics, but also the feedback controller dynamics. This means that feasible volume $\mathcal{A}$ is online reshaped in order to take into account any action of the feedback controller and, accordingly, to generate appropriate bounds for the admissible accelerations and jerks.

IV. ONLINE TRAJECTORY SCALING

The bounds on the longitudinal jerk and acceleration, evaluated according to the procedure proposed into the previous section, are used to online scale the robot trajectory. To this purpose, the online trajectory scaling filter shown in Fig. 3 has been developed. Starting from a generic trajectory $s(t)$, which is possibly infeasible, the filter generates a new trajectory $\tilde{s}(t)$ which fulfills the dynamic bounds.

The time scaling filter is composed by two key elements. The first is the trajectory generator, which returns the desired
velocity \( \dot{s}_d(s) \) and acceleration \( \ddot{s}_d(s) \) for each position \( s \) along the curve. As previously anticipated, \( \dot{s}_d(t) \) could be infeasible, i.e., it could not satisfy the constraints on the maximum longitudinal acceleration and jerk. The nonlinear dynamic filter, which represents the second element of the scheme, is used to dynamically scale the velocity in order to fulfill the desired bounds. Its kernel is based on a modified version of the nonlinear trajectory smoother proposed in [17]. Practically, given a desired reference signal \( \dot{s}_d \), the filter is able to track it “at best” by taking into account the given constraints. “At best” means that the filter output \( \dot{s}(t) \) exactly coincides with \( \dot{s}_d(t) \) only if the assigned bounds are fulfilled, i.e., if \( \dot{s}_d \in [S^-, S^+] \) and \( \dot{s}_d \in [U^-, U^+] \), while tracking is voluntarily lost every time such bounds are violated. In this case a new velocity profile \( \dot{s} \), which satisfies the given constraints, is generated. The dynamic filter is designed such that \( \dot{s} \) robustly converges in minimum time toward \( \dot{s}_d \) as soon as \( \dot{s}_d \) newly fulfills the dynamic constraints. The output of the filter variable-structure controller is evaluated according to the following discrete time algorithm

\[
\dot{u}_n := -U \text{sat}(\sigma_n) \tag{28}
\]

\[
U := \begin{cases} 
  -U^- & \text{if } (\sigma_n > -1) \text{ or } (\rho < 0) \\
  U^+ & \text{if } (\sigma_n \geq 1) \text{ or } (\rho \geq 0) \\
  U^+ & \text{if } (\sigma_n < 1) \text{ or } (\rho < 0) 
\end{cases} \tag{29}
\]

\[
\sigma_n := \begin{cases} 
  \dot{z}_n + \frac{(n-1)}{2} \text{sgn}(\zeta) & \text{if } \zeta_n < \zeta_M^+ \\
  \dot{z}_n - \frac{(n-1)}{2} \text{sgn}(\zeta) & \text{if } \zeta_n > \zeta_M^-
\end{cases} \tag{30}
\]

where

\[
\rho := \frac{y_n}{T} + \frac{\dot{y}_n}{2} \tag{31}
\]

\[
\zeta_n := \begin{cases} 
  -\frac{\rho}{T U^-} & \text{if } \rho < 0 \\
  \frac{\rho}{T U^+} & \text{if } \rho > 0
\end{cases} \tag{32}
\]

\[
\dot{\zeta}_n := \begin{cases} 
  -\frac{\dot{y}_n}{T U^-} & \text{if } \rho < 0 \\
  \frac{\dot{y}_n}{T U^+} & \text{if } \rho > 0
\end{cases} \tag{33}
\]

\[
m := \text{Int} \left[ 1 + \sqrt{1 + 8|\zeta_n|} \right] \tag{34}
\]

\[
\zeta_M^+ := -\left[ \zeta_M^+ \right] \left[ \frac{\dot{z}_M^+ - \frac{\dot{\zeta}_M}{2} - \frac{1}{2}}{2} \right], \quad \zeta_M^- := -\left[ \zeta_M^- \right] \left[ \frac{\dot{z}_M^- - \frac{\dot{\zeta}_M}{2} - \frac{1}{2}}{2} \right] \tag{35}
\]

\[
\dot{\zeta}_M^+ := -\left[ \dot{z}_M^+ \right] \left[ \frac{\dot{z}_M^+ - \frac{\dot{\zeta}_M}{2} - \frac{1}{2}}{2} \right], \quad \dot{\zeta}_M^- := -\left[ \dot{z}_M^- \right] \left[ \frac{\dot{z}_M^- - \frac{\dot{\zeta}_M}{2} - \frac{1}{2}}{2} \right] \tag{36}
\]

and where \( \dot{y} := \dot{s} - \dot{s}_d \) represents the tracking error with respect to \( \dot{s}_d \), while \( \ddot{y} := \ddot{s} - \ddot{s}_d \) is its first derivative. Finally, \( T \) is the filter sampling time.

The new filter (28)–(36), differently from that proposed in [17], admits asymmetric bounds \( U^+ \) and \( U^- \) on \( \dot{s} \). Owing to the lack of space, the stability proof and the convergence properties of the filter are omitted. They can be demonstrated by means of techniques similar to those proposed in [17].

### V. Results

The effectiveness of the proposed two-level trajectory controller has been evaluated by simulating the control along a given trajectory of the two link planar robot. The manipulator kinematic and dynamic parameters are defined according to Table I and II respectively.

The manipulator path is an ellipsoid represented by means of a curve in the joint space \( f(s) := [f_1(s) f_2(s)]^T \) defined as follows

\[
\begin{cases} 
  f_1(s) := 0.4(1 - \cos(s)) \\
  f_2(s) := 0.8\sin(s)
\end{cases} \tag{37}
\]

The following nominal velocity profile has been assumed

\[
\dot{s}_d(s) = \begin{cases} 
  -K_1(s - a)^2 + K_2, & 0 \leq s \leq a \\
  4a, & a \leq s \leq b \\
  -K_3(s - c)^2 + K_4, & 2b \leq s \leq c \\
  3a, & c \leq s \\
  \text{otherwise}
\end{cases} \tag{38}
\]

In order to evaluate the controller behavior, the velocity profile has been designed such to be too demanding with respect to the dynamic constraints. For example, it can be immediately verified that \( \dot{s}_d(s) \) is discontinuous at \( s = b \), so that, in that point, the acceleration is infinite. Obviously such profile could be tracked only if an infinite torque is available.

Two different scenarios have been simulated. In the first, the manipulator model has been supposed to be exactly known. In the second, the robustness of the proposed approach has been tested by considering a partial and slightly perturbed knowledge of the model. In both cases, torques and torque derivatives are constrained between the following

<table>
<thead>
<tr>
<th>Link ( d_j ) (m)</th>
<th>( \theta_i ) (rad)</th>
<th>( a_j ) (m)</th>
<th>( \alpha_i ) (rad)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( d_1 )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>( d_2 )</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Link ( M ) (Kg)</th>
<th>Center of gravity ( (m) )</th>
<th>Inertia ( (Kg.m^2) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23.90</td>
<td>0.090</td>
</tr>
<tr>
<td>2</td>
<td>3.88</td>
<td>0.048</td>
</tr>
</tbody>
</table>
bounds: \(|\tau_i| \leq 30\) (\(N\)) and \(|\ddot{\tau}_i| \leq 350\) (\(N\,s^{-1}\)), \(i = 1,2\). For all the examples the controller feedback gains are equal to 
\(K_p = [200\,200]^T\) and \(K_v = [60\,60]^T\).

A. Perfect knowledge of the manipulator model

In the first example the manipulator model is supposed to be completely known, so that it is possible to implement control law (17). The control improvements due to the time scaling filter are highlighted by considering three different situations:

- **Case 1** - the time scaling filter is disabled and reference velocity (37) and acceleration (38) are directly used to drive the manipulator controller (red dotted lines);
- **Case 2** - the time scaling filter is activated but only torque constraints (6) are taken into account (green dashed lines);
- **Case 3** - the time scaling filter is fully activated in order to simultaneously fulfill (6) and (7) (blue continuous lines).

Since the desired velocity and acceleration profiles are not feasible, i.e., the nominal trajectory does not always belong to region \(A\), path tracking is evidently lost if the time scaling strategy is not used. This conclusion is immediately confirmed by Fig. 5, where the robot path is plotted by means of a red dotted line. The controller performances improve if the time scaling filter is activated. In a first time only torque constraints (6) are taken into account, thus mimicking the situation considered by Dahl and Nielsen in [10]. Torque derivative constraints are not considered in this phase. As expected, tracking errors drastically reduce as can be evinced from Fig. 4. More precisely, errors detected for **Case 2** are almost one order of magnitude smaller than those obtained for **Case 1**. As shown in Fig. 4a, this result is obtained by online modifying the velocity reference signal. Fig. 4 also demonstrates that the situation can further improve when the time scaling filter proposed in this paper is used, i.e., when the torque constraints (6) and the torque derivative constraints (7) are simultaneously considered. In particular, blue continuous curves show that maximum tracking errors reduce of almost two orders of magnitude with respect to **Case 1**.

The paths obtained for the three cases are shown in Fig. 5: path tracking neatly improves when torque bounds are taken into account, but it is almost perfect when also torque derivative constraints are considered.

Fig. 6 refers to **Case 3**. More precisely, Fig. 6a and Fig. 6b respectively show the real time evaluated bounds on \(\dot{s}\) and \(\ddot{s}\) (dashed red lines) compared with the actual manipulator longitudinal accelerations and jerks (continuous blue lines): constraints are evidently active in several points along the path. Finally, Fig. 6c and 6d, respectively show the controller output torques and torque derivatives: dynamic constraints are always fulfilled despite any interference of the feedback actions through the two gain vectors \(K_p\) and \(K_v\).

B. Approximate knowledge of the manipulator model

The controller robustness has been analyzed in presence of a partial and wrong knowledge of the robot model. Indeed, the identification of the robot parameters, especially for manipulators with many degree of freedom, can be quite a demanding task. For this reason, the following simplified control law has been assumed

\[
\tau(s,\dot{s},\ddot{s}) = \hat{H}(\mathbf{f}(s))\mathbf{f}'(s)\ddot{s} + K_p^T e + K_v^T \mathbf{e} \tag{39}
\]

where \(\hat{H}(\cdot)\) denotes an estimated inertia matrix obtained by perturbing the coefficients of the nominal \(H(\cdot)\) by the 5%.

The resulting path tracking errors are shown in Fig. 7 compared with those obtained when the manipulator model is perfectly known: only minor differences can be detected.
by considering the presence of model uncertainties. The proposed control strategy is a nonlinear time scaling filter which dynamically scales the longitudinal reference velocity of the end effector. The kernel of the control strategy is a torque-limited path following by online algorithms for the efficient evaluation of the high order manipulator dynamics, the proposed control strategy is suited to be used in online applications. Simulation results have shown that the proposed approach allows a good path tracking even when reference velocities and accelerations are not physically feasible. The algorithm robustness has been tested by considering the presence of model uncertainties.

VI. CONCLUSIONS

The paper has addressed a manipulator trajectory tracking problem subject to torque and torque derivative constraints. The proposed control algorithm extends the results obtained by Dahl and Nielsen by considering the existence of dynamic constraints on the torque derivatives. The kernel of the control strategy is a nonlinear time scaling filter which dynamically scales the longitudinal reference velocity of the end effector. Due to the use of recently devised iterative algorithms for the efficient evaluation of the high order manipulator dynamics, the proposed control strategy is suited to be used in online applications. Simulation results have shown that the proposed approach allows a good path tracking even when reference velocities and accelerations are not physically feasible. The algorithm robustness has been tested by considering the presence of model uncertainties.

REFERENCES