Evaluation of generalized force derivatives by means of a recursive Newton-Euler approach

Corrado GUARINO LO BIANCO, Member, IEEE

***Abstract*—An accurate estimation of the dynamics efforts acting on a robot manipulator represents an important issue for both the analysis of its behavior and the synthesis of appropriate controllers. This paper proposes an iterative algorithm, based on the Newton-Euler approach, for the efficient evaluation of the manipulators high order kinematics and dynamics. In particular, the algorithm computes velocities, accelerations, and jerks of each link, while new dynamic equations are devised in order to evaluate the first derivative of generalized forces. Owing to its moderate computational burden, the algorithm is suited to be used in online applications.**

***Index Terms*—Kinematics, dynamics, jerk, force derivative, Newton-Euler, rigid-body manipulators.**

I. Introduction

Manipulators joint forces and torques, i.e., the Generalized Forces (GFs), are naturally bounded due to the actuators physical limits. For this reason, the generalized force bounding problem has been widely investigated in the past. Several solutions have been proposed, which can be roughly divided into two categories: offline and online approaches. Offline approaches are normally based on algorithms for the optimal trajectory planning which consider the existence of kinematic and/or dynamic constraints. For example, in [1] a minimum-time movement along an assigned path was planned by accounting for constraints on joint velocities and torques. However, when an optimal trajectory is used, there is at least one joint which is constantly working at the maximum of its kinematic or dynamic capabilities: due to model uncertainties the control could easily be lost. Several online algorithms have been developed in the past for handling this problem. The scheme originally proposed in [2] is often cited as an example: dynamic constraints are satisfied by means of an online method which automatically and appropriately scales assigned trajectories.

Similarly, also Generalized Force Derivatives (GFDs) are physically bounded. For example, in case of electric actuators, the available supply voltage necessarily limits the GFs variation rate. Unfortunately, closed form expressions for the evaluation of GFDs are normally complex and time expensive. As a consequence, bounds on GFDs are often neglected or indirectly considered by limiting jerks. For example in [3], again in the context of offline optimal trajectory planning, kinematic limits on joint velocities, accelerations, and jerks were converted into constraints for the resulting minimum-time optimization problem, while in [4] optimality was achieved by means of an efficient online algorithm. In other approaches, still not considering GFDs, kinematic and dynamic constraints have been simultaneously considered. For example, works [5]–[7] deal with minimum-time trajectory planning problems which take into account constraints on jerks and torques.

The simultaneous existence of bounds on both GFs and GFDs was first explicitly mentioned in [8]. The two authors proposed a dynamic programming approach, which, unfortunately, was not sufficiently investigated in the paper. Later, the topic was also considered in [9] for a discretized robot. In that paper, numerically evaluated generalized force derivatives were used. The necessity of considering a discrete-time model, and consequently a discrete-time problem, was justified as follows: "...discrete computation methods are essential as the highly nonlinear equations of motion for the manipulator dynamics are too complex for real-time computation within a sampling interval". Evidently, the approach proposed in [9] only solves an approximate problem. Generalized force derivatives have been recently considered in [10] where a Lagrangian based approach was used for the optimal offline trajectory planning along assigned paths. Depending on the complexity of the considered manipulator, the computational burden introduced by Lagrangian based approaches can easily become unacceptable: the sampling time could be violated in case of online methods, or the convergence of offline optimization algorithms could require an excessive time.

This paper proposes a technique for the exact and efficient evaluation of GFDs. In particular, high order kinematics and dynamics are evaluated by means of an algorithm obtained by adding new equations to the standard Newton-Euler recursive algorithm originally proposed in [11]. The advantage of this choice is double. First, existing iterative algorithms can easily be modified in order to implement the new functionalities and, secondly, the number of operations required for the evaluation of the high order dynamics is moderate, since it still linearly depends on the number of joints. Precursors of the algorithm here proposed, see e.g. [12], were used in [13] for the optimal offline trajectory planning, and in [14] for the online trajectory scaling. In both cases constraints on GFs and on GFDs were considered.

The paper is organized as follows. The notation used along the paper is presented in §2, together with an overview of the basic rules used to manipulate vectors and rotation matrices. The iterative algorithm is proposed in §3, while §4 is devoted to demonstrating its equations. Some computational considerations are drawn in §5, where a Stanford manipulator is used to validate the algorithm. Final conclusions are reported in §6.

II. Notation

This section introduces the notation used along the paper. Frames have been assigned to an N link manipulator according to the modified Denavit-Hartenberg procedure [15]. Using a well assessed convention, let us define (see Fig. 1)

- α_i angle between the $\hat{\mathbf{z}}_i$ and the $\hat{\mathbf{z}}_{i+1}$ axes measured in the righthand sense about $\hat{\mathbf{x}}_i$;
- a_i distance between the $\hat{\mathbf{z}}_i$ and the $\hat{\mathbf{z}}_{i+1}$ axes, measured along $\hat{\mathbf{x}}_i$;
- θ_{i+1} angle between the $\hat{\mathbf{x}}_i$ and the $\hat{\mathbf{x}}_{i+1}$ axes, measured in the righthand sense about $\hat{\mathbf{z}}_{i+1}$;
- d_{i+1} distance between the $\hat{\mathbf{x}}_i$ and the $\hat{\mathbf{x}}_{i+1}$ axes, measured along $\hat{\mathbf{z}}_{i+1}$.

The orientation of a generic frame $i+1$ with respect to frame i can be expressed by means of the following rotation matrix

$$ {}_{i+1}^{i}\mathbf{R} = {}^{i+1}_{i}\mathbf{R}^T := \begin{bmatrix} \mathrm{c}\theta_{i+1} & -\mathrm{s}\theta_{i+1} & 0 \\ \mathrm{s}\theta_{i+1}\,\mathrm{c}\alpha_i & \mathrm{c}\theta_{i+1}\,\mathrm{c}\alpha_i & -\mathrm{s}\alpha_i \\ \mathrm{s}\theta_{i+1}\,\mathrm{s}\alpha_i & \mathrm{c}\theta_{i+1}\,\mathrm{s}\alpha_i & \mathrm{c}\alpha_i \end{bmatrix}, $$

while the position of frame $i+1$ with respect to frame i, described with respect to frame i, is given by

$$ {}^{i}\mathbf{p}_{i+1,i} := \begin{bmatrix} a_i \\ -\mathrm{s}\alpha_i\, d_{i+1} \\ \mathrm{c}\alpha_i\, d_{i+1} \end{bmatrix}. $$

A compact notation has been adopted for ${}_{i+1}^{i}\mathbf{R}$ and ${}^{i}\mathbf{p}_{i+1,i}$, so that $\mathrm{c}\theta_i := \cos\theta_i$, $\mathrm{s}\theta_i := \sin\theta_i$ and so on.

The author is with the Dip. di Ingegneria dell'Informazione, University of Parma, Parco Area delle Scienze 181/A, I-43100 Parma, Italy (email:guarino@ce.unipr.it).

This work was partially supported by AER-TECH Lab from Regione Emilia-Romagna, Italy.

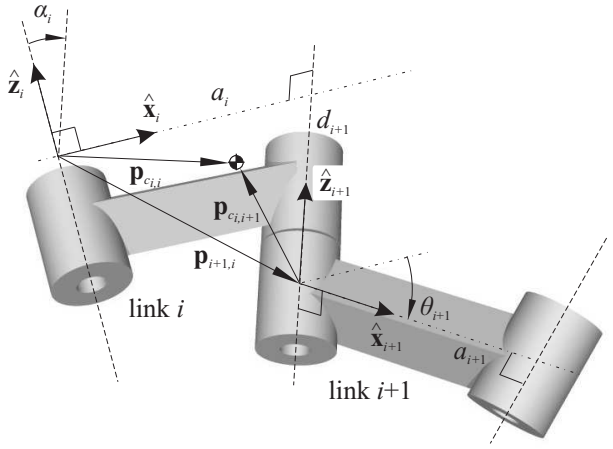

Fig. 1. Modified Denavit-Hartenberg frames.

Conventionally, ${}^i\mathbf{v}_{j,k}$ represents the velocity between frame j and frame k, described with respect to frame i. As known, the same vector can be described with respect to a different frame by means of a rotation matrix, e.g. ${}^h\mathbf{v}_{j,k} = {}^h_i\mathbf{R}\,{}^i\mathbf{v}_{j,k}$. To shorten the representation, inertial frame 0 is normally not indicated, thus $\mathbf{v}_i$ is equivalent to ${}^0\mathbf{v}_{i,0}$ and ${}^i\mathbf{v}_i$ is equivalent to ${}^i\mathbf{v}_{i,0} = {}^i_0\mathbf{R}\,{}^0\mathbf{v}_{i,0}$. The same convention is assumed for accelerations and jerks.

According to the differentiation rules of rotation matrices, it is possible to write

$$ {}^0_i\dot{\mathbf{R}} = \mathbf{S}(\boldsymbol{\omega}_i)\,{}^0_i\mathbf{R}\,, \tag{1} $$

where $\mathbf{S}(\boldsymbol{\omega}_i)$ is a skew-symmetric matrix based on the components of angular velocity $\boldsymbol{\omega}_i := [\omega_x\ \omega_y\ \omega_z]^T$ and defined as follows

$$ \mathbf{S}(\boldsymbol{\omega}_i) := \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}\,. $$

Properties of skew-symmetric matrices can be found in [16].

Symbol ${}^i\hat{\mathbf{z}}_{i+1}$ is used to indicate the $\hat{\mathbf{z}}$ unit vector of frame $i+1$, described with respect to frame i. It can be obtained from ${}_{i+1}{}^i\mathbf{R}$ according to the following expression

$$ {}^i\hat{\mathbf{z}}_{i+1} = {}_{i+1}{}^i\mathbf{R}\,\hat{\mathbf{z}}_0\,, $$

where $\hat{\mathbf{z}}_0$ is a constant unit vector defined as $\hat{\mathbf{z}}_0 := [0\ 0\ 1]^T$.

Analogously, unit vector $\hat{\mathbf{z}}_{i+1}$ can be represented as

$$ \hat{\mathbf{z}}_{i+1} = {}_{i+1}{}^0\mathbf{R}\,\hat{\mathbf{z}}_0\,, $$

so that, by virtue of (1) and the properties of $\mathbf{S}(\cdot)$, its derivative can be expressed as

$$ \dot{\hat{\mathbf{z}}}_{i+1} = {}_{i+1}{}^0\dot{\mathbf{R}}\,\hat{\mathbf{z}}_0 = \mathbf{S}(\boldsymbol{\omega}_{i+1})\,{}_{i+1}{}^0\mathbf{R}\,\hat{\mathbf{z}}_0 = \mathbf{S}(\boldsymbol{\omega}_{i+1})\,\hat{\mathbf{z}}_{i+1} = \boldsymbol{\omega}_{i+1}\times\hat{\mathbf{z}}_{i+1}\,. \tag{2} $$

III. A recursive Newton-Euler algorithm for the evaluation of generalized force derivatives

The recursive algorithm proposed in the following efficiently evaluates the joint GFDs for rigid, open chain manipulators. It returns, as usual, the solution of the inverse dynamic problem

$$ \boldsymbol{\tau} = \boldsymbol{\tau}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\,, $$

but it also evaluates, and this is a novelty, GFDs

$$ \dot{\boldsymbol{\tau}} = \dot{\boldsymbol{\tau}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \dddot{\mathbf{q}})\,. $$

To this purpose, the efficient Newton-Euler approach originally proposed by Luh, Walker and Paul [11] has been modified by adding new expressions. For each link, a forward recursion returns linear and angular velocities, accelerations, and jerks, then a backward recursion gives the solution of the inverse dynamic problem by evaluating for each link GFs and, furthermore, GFDs.

The algorithm is proposed in the following, while its equations are derived in the next section. Frames are assigned according to the modified Denavit-Hartenberg method [15]. An equivalent algorithm based on the standard Denavit-Hartenberg convention has been proposed in [12].

A. Forward recursion

The following recursive algorithm evaluates link velocities, accelerations, and jerks ($i = 0, 1, \ldots, N-1$)

$$ {}^{i+1}\boldsymbol{\omega}_{i+1} = {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\omega}_i + \dot{\theta}_{i+1}\,\hat{\mathbf{z}}_0 \tag{3} $$

$$ {}^{i+1}\boldsymbol{\gamma}_{i+1} = {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\gamma}_i + {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\omega}_i\times\dot{\theta}_{i+1}\,\hat{\mathbf{z}}_0 + \ddot{\theta}_{i+1}\,\hat{\mathbf{z}}_0 \tag{4} $$

$$ \begin{aligned} {}^{i+1}\boldsymbol{\iota}_{i+1} = {}& {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\iota}_i + {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\gamma}_i\times\dot{\theta}_{i+1}\,\hat{\mathbf{z}}_0 + \dddot{\theta}_{i+1}\,\hat{\mathbf{z}}_0 \\ & + {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\omega}_i\times\left(2\,\ddot{\theta}_{i+1}\,\hat{\mathbf{z}}_0 + {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\omega}_i\times\dot{\theta}_{i+1}\,\hat{\mathbf{z}}_0\right) \end{aligned} \tag{5} $$

$$ \begin{aligned} {}^{i+1}\mathbf{a}_{i+1} = {}& {}^{i+1}_i\mathbf{R}\left[{}^i\mathbf{a}_i + {}^i\boldsymbol{\gamma}_i\times{}^i\mathbf{p}_{i+1,i} + {}^i\boldsymbol{\omega}_i\times\left({}^i\boldsymbol{\omega}_i\times{}^i\mathbf{p}_{i+1,i}\right)\right] \\ & + 2\,{}^{i+1}\boldsymbol{\omega}_{i+1}\times\dot{d}_{i+1}\,\hat{\mathbf{z}}_0 + \ddot{d}_{i+1}\,\hat{\mathbf{z}}_0 \end{aligned} \tag{6} $$

$$ {}^i\mathbf{a}_{c_i} = {}^i\mathbf{a}_i + {}^i\boldsymbol{\gamma}_i\times{}^i\mathbf{p}_{c_i,i} + {}^i\boldsymbol{\omega}_i\times\left({}^i\boldsymbol{\omega}_i\times{}^i\mathbf{p}_{c_i,i}\right) \tag{7} $$

$$ \begin{aligned} {}^{i+1}\mathbf{j}_{i+1} = {}& {}^{i+1}_i\mathbf{R}\left\{{}^i\mathbf{j}_i + {}^i\boldsymbol{\iota}_i\times{}^i\mathbf{p}_{i+1,i} + 2\,{}^i\boldsymbol{\gamma}_i\times({}^i\boldsymbol{\omega}_i\times{}^i\mathbf{p}_{i+1,i})\right. \\ & \left. + {}^i\boldsymbol{\omega}_i\times[{}^i\boldsymbol{\gamma}_i\times{}^i\mathbf{p}_{i+1,i} + {}^i\boldsymbol{\omega}_i\times({}^i\boldsymbol{\omega}_i\times{}^i\mathbf{p}_{i+1,i})]\right\} \\ & + \dddot{d}_{i+1}\hat{\mathbf{z}}_0 + {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\gamma}_i\times 3\,\ddot{d}_{i+1}\hat{\mathbf{z}}_0 \\ & + {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\omega}_i\times(3\,\ddot{d}_{i+1}\hat{\mathbf{z}}_0 + {}^{i+1}_i\mathbf{R}\,{}^i\boldsymbol{\omega}_i\times 3\,\dot{d}_{i+1}\hat{\mathbf{z}}_0) \end{aligned} \tag{8} $$

$$ \begin{aligned} {}^i\mathbf{j}_{c_i} = {}& {}^i\mathbf{j}_i + {}^i\boldsymbol{\iota}_i\times{}^i\mathbf{p}_{c_i,i} + 2\,{}^i\boldsymbol{\gamma}_i\times\left({}^i\boldsymbol{\omega}_i\times{}^i\mathbf{p}_{c_i,i}\right) \\ & + {}^i\boldsymbol{\omega}_i\times\left[{}^i\boldsymbol{\gamma}_i\times{}^i\mathbf{p}_{c_i,i} + {}^i\boldsymbol{\omega}_i\times\left({}^i\boldsymbol{\omega}_i\times{}^i\mathbf{p}_{c_i,i}\right)\right]\,. \end{aligned} \tag{9} $$

where (all vectors are described with respect to frame i)

- ${}^i\boldsymbol{\omega}_i$ angular velocity of frame i;
- ${}^i\boldsymbol{\gamma}_i$ angular acceleration of frame i;
- ${}^i\boldsymbol{\iota}_i$ angular jerk of frame i;
- ${}^i\mathbf{a}_i$ linear acceleration of frame i;
- ${}^i\mathbf{a}_{c_i}$ linear acceleration of the link i mass centre;
- ${}^i\mathbf{j}_i$ linear jerk of frame i;
- ${}^i\mathbf{j}_{c_i}$ linear jerk of the link i mass centre;
- ${}^i\mathbf{p}_{c_i,i}$ position of the link i mass centre with respect to frame i;

while $\dot{\theta}_{i+1}$, $\ddot{\theta}_{i+1}$, $\dddot{\theta}_{i+1}$, $\dot{d}_{i+1}$, $\ddot{d}_{i+1}$, and $\dddot{d}_{i+1}$ represent the first, second, and third time derivatives of the joint variables. It is worth noting that $\dot{\theta}_{i+1} = \ddot{\theta}_{i+1} = \dddot{\theta}_{i+1} = 0$ if joint $i+1$ is prismatic, while $\dot{d}_{i+1} = \ddot{d}_{i+1} = \dddot{d}_{i+1} = 0$ if joint $i+1$ is revolute.

B. Backward recursion

Starting from the manipulator kinematics, evaluated by means of (3)–(9), and from the knowledge of the external efforts acting on the last link, i.e., ${}^{N+1}\mathbf{f}_{N+1}$, ${}^{N+1}\mathbf{n}_{N+1}$, ${}^{N+1}\dot{\mathbf{f}}_{N+1}$, ${}^{N+1}\dot{\mathbf{n}}_{N+1}$, it is possible to determine the joint GFs and GFDs. The following recursive backward algorithm is proposed ($i = N, N-1, \ldots, 1$)

$$ {}^i\mathbf{F}_i = m_i\,{}^i\mathbf{a}_{c_i} \tag{10} $$

$$ {}^i\mathbf{N}_i = {}^i\mathbf{I}_i\,{}^i\boldsymbol{\gamma}_i + {}^i\boldsymbol{\omega}_i\times{}^i\mathbf{I}_i\,{}^i\boldsymbol{\omega}_i \tag{11} $$

$$ {}^i\dot{\mathbf{F}}_i = m_i\,{}^i\mathbf{j}_{c_i} \tag{12} $$

$$ \begin{aligned} {}^i\dot{\mathbf{N}}_i = {}& {}^i\boldsymbol{\omega}_i\times{}^i\mathbf{I}_i\,{}^i\boldsymbol{\gamma}_i + {}^i\mathbf{I}_i[{}^i\boldsymbol{\gamma}_i\times{}^i\boldsymbol{\omega}_i] + {}^i\mathbf{I}_i\,{}^i\boldsymbol{\iota}_i \\ & + {}^i\boldsymbol{\gamma}_i\times{}^i\mathbf{I}_i\,{}^i\boldsymbol{\omega}_i + {}^i\boldsymbol{\omega}_i\times{}^i\mathbf{N}_i \end{aligned} \tag{13} $$

$$ {}^i\mathbf{f}_i = {}_{i+1}{}^i\mathbf{R}\,{}^{i+1}\mathbf{f}_{i+1} + {}^i\mathbf{F}_i \tag{14} $$

$$ \begin{aligned} {}^i\mathbf{n}_i = {}& {}_{i+1}{}^i\mathbf{R}\,{}^{i+1}\mathbf{n}_{i+1} + {}^i\mathbf{p}_{c_i,i}\times{}^i\mathbf{F}_i \\ & + {}^i\mathbf{p}_{i+1,i}\times{}_{i+1}{}^i\mathbf{R}\,{}^{i+1}\mathbf{f}_{i+1} + {}^i\mathbf{N}_i \end{aligned} \tag{15} $$

$$
\begin{aligned}
{}^i\dot{\mathbf{f}}_i &= {}_{i+1}^{i}\mathbf{R}\,{}^{i+1}\dot{\mathbf{f}}_{i+1} + {}^i\dot{\mathbf{F}}_i && (16)\\
{}^i\dot{\mathbf{n}}_i &= {}_{i+1}^{i}\mathbf{R}\,{}^{i+1}\dot{\mathbf{n}}_{i+1} + ({}^i\omega_i \times {}^i\mathbf{p}_{c_i,i}) \times {}^i\mathbf{F}_i + {}^i\mathbf{p}_{c_i,i} \times {}^i\dot{\mathbf{F}}_i \\
&\quad + ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i+1,i} + \dot{d}_{i+1}\,{}^i\hat{\mathbf{z}}_{i+1}) \times {}_{i+1}^{i}\mathbf{R}\,{}^{i+1}\mathbf{f}_{i+1} \\
&\quad + {}^i\mathbf{p}_{i+1,i} \times {}_{i+1}^{i}\mathbf{R}\,{}^{i+1}\dot{\mathbf{f}}_{i+1} + {}^i\dot{\mathbf{N}}_i && (17)\\
\tau_i &= \begin{cases} {}^i\mathbf{n}_i^T\,\hat{\mathbf{z}}_0 \text{ if joint } i \text{ is revolute} & (18)\\ {}^i\mathbf{f}_i^T\,\hat{\mathbf{z}}_0 \text{ if joint } i \text{ is prismatic} & (19)\end{cases}\\
\dot{\tau}_i &= \begin{cases} \left[{}^i\dot{\mathbf{n}}_i + {}^i\mathbf{n}_i \times {}^i\boldsymbol{\omega}_i\right]^T \hat{\mathbf{z}}_0 \text{ if joint } i \text{ is revolute} & (20)\\ \left[{}^i\dot{\mathbf{f}}_i + {}^i\mathbf{f}_i \times {}^i\boldsymbol{\omega}_i\right]^T \hat{\mathbf{z}}_0 \text{ if joint } i \text{ is prismatic} & (21)\end{cases}
\end{aligned}
$$

where (all vectors are described with respect to frame i)

- m_i mass of link i;
- ${}^i\mathbf{F}_i$ total force acting on the link i mass centre;
- ${}^i\mathbf{N}_i$ total torque acting on the link i mass centre;
- ${}^i\mathbf{f}_i$ total force exerted on link i by link $i-1$;
- ${}^i\mathbf{n}_i$ total torque exerted on link i by link $i-1$;
- τ_i generalized force exerted on joint i;
- ${}^i\mathbf{I}_i$ inertia tensor of link i about a frame placed at the mass centre and fixed to the body.

The accuracy of (18)–(21) could be improved by considering friction and motor inertia. The interested reader can refer to [16] for a discussion on the consequences that both phenomena have on generalized joint forces. By opportunely differentiating the generalized force components corresponding to friction and motor inertia, it is easily possible to devise the further terms to be added to (20) and (21) in order to obtain more accurate expressions. For the sake of simplicity, both effects have been neglected in this work.

IV. Synthesis of the recursive equations

It is known that the kinematic equations of an N degrees of freedom, rigid-link manipulator can be written in recursive form according the scheme originally proposed in [17] and in [18], i.e., the kinematic status of each link can be evaluated by means of a recursive algorithm according to the following equations

$$
\begin{aligned}
\boldsymbol{\omega}_{i+1} &= \boldsymbol{\omega}_i + \dot{\theta}_{i+1}\,\hat{\mathbf{z}}_{i+1}\,, && (22)\\
\mathbf{v}_{i+1} &= \mathbf{v}_i + \boldsymbol{\omega}_i \times \mathbf{p}_{i+1,i} + \dot{d}_{i+1}\,\hat{\mathbf{z}}_{i+1}\,, && (23)\\
\boldsymbol{\gamma}_{i+1} &= \boldsymbol{\gamma}_i + \ddot{\theta}_{i+1}\,\hat{\mathbf{z}}_{i+1} + \dot{\theta}_{i+1}\,\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i+1}\,, && (24)\\
\mathbf{a}_{i+1} &= \mathbf{a}_i + \boldsymbol{\gamma}_i \times \mathbf{p}_{i+1,i} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i+1,i}) \\
&\quad + \ddot{d}_{i+1}\hat{\mathbf{z}}_{i+1} + 2\,\dot{d}_{i+1}\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i+1}\,. && (25)
\end{aligned}
$$

In [11], the efficiency was improved by describing velocities and accelerations of each link directly with respect to the link frame. The result was a smart recursive algorithm given by (3), (4), (6), (7), (10), (11), (14), (15), (18), and (19).

In order to evaluate the generalized force derivatives, new expressions have been added to the algorithm proposed in [11]. The following subsections are devoted to demonstrating the new equations (5), (8), (9), (12), (13), (16), (17), (20), and (21).

Any effort has been spent in order to optimize the performances. For the same efficiency reasons pointed out in [11], link variables are always described with respect to the link frames and equations are arranged such to minimize the number of products between vectors and matrices as well as the number of crossproducts, since both operations are computationally expensive.

A. Recursive formulation of angular jerk

It is well known [15] that the angular acceleration of frame i with respect to inertial frame 0 can be expressed by means of (24). By differentiating (24) with respect to time and also considering (2), it is possible to express the angular jerk of frame $i+1$ as follows

$$
\begin{aligned}
\boldsymbol{\iota}_{i+1} &= \boldsymbol{\iota}_i + \dddot{\theta}_{i+1}\,\hat{\mathbf{z}}_{i+1} + \ddot{\theta}_{i+1}\,(\boldsymbol{\omega}_{i+1} \times \hat{\mathbf{z}}_{i+1}) + \ddot{\theta}_{i+1}\,\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i+1} \\
&\quad + \dot{\theta}_{i+1}\,\boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i+1} + \dot{\theta}_{i+1}\,\boldsymbol{\omega}_i \times (\boldsymbol{\omega}_{i+1} \times \hat{\mathbf{z}}_{i+1})\,. && (26)
\end{aligned}
$$

By taking into account (22) it is possible write

$$
\boldsymbol{\omega}_{i+1} \times \hat{\mathbf{z}}_{i+1} = (\boldsymbol{\omega}_i + \dot{\theta}_{i+1}\,\hat{\mathbf{z}}_{i+1}) \times \hat{\mathbf{z}}_{i+1} = \boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i+1} \qquad (27)
$$

since, as known, $\hat{\mathbf{z}}_{i+1} \times \hat{\mathbf{z}}_{i+1} = 0$. Consequently, (26) can be reorganized as follows

$$
\begin{aligned}
\boldsymbol{\iota}_{i+1} &= \boldsymbol{\iota}_i + \dddot{\theta}_{i+1}\,\hat{\mathbf{z}}_{i+1} + 2\ddot{\theta}_{i+1}\,(\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i+1}) \\
&\quad + \dot{\theta}_{i+1}\,\boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i+1} + \dot{\theta}_{i+1}\,\boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i+1})\,.
\end{aligned}
$$

By noting that ${}^{i+1}\hat{\mathbf{z}}_{i+1} = \hat{\mathbf{z}}_0$, angular jerk can be described with respect to frame $i+1$ according to the following equation

$$
\begin{aligned}
{}^{i+1}\boldsymbol{\iota}_{i+1} &= {}_{i}^{i+1}\mathbf{R}\,{}^i\boldsymbol{\iota}_i + \dddot{\theta}_{i+1}\,\hat{\mathbf{z}}_0 + 2\ddot{\theta}_{i+1}\,({}_{i}^{i+1}\mathbf{R}\,{}^i\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_0) \\
&\quad + \dot{\theta}_{i+1}\,{}_{i}^{i+1}\mathbf{R}\,{}^i\boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_0 \\
&\quad + \dot{\theta}_{i+1}\,{}_{i}^{i+1}\mathbf{R}\,{}^i\boldsymbol{\omega}_i \times ({}_{i}^{i+1}\mathbf{R}\,{}^i\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_0)\,. && (28)
\end{aligned}
$$

Equation (5) is finally inferred by rearranging (28).

It is worth noticing that (5) can be indifferently used for prismatic or revolute joints. In case of prismatic joints evidently $\dot{\theta}_{i+1} = \ddot{\theta}_{i+1} = \dddot{\theta}_{i+1} = 0$.

B. Recursive formulation of linear jerk

The linear jerk of frame $i+1$ can be obtained by differentiating (25)

$$
\begin{aligned}
\mathbf{j}_{i+1} &= \mathbf{j}_i + \boldsymbol{\iota}_i \times \mathbf{p}_{i+1,i} + \boldsymbol{\gamma}_i \times \dot{\mathbf{p}}_{i+1,i} + \boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i+1,i}) \\
&\quad + \boldsymbol{\omega}_i \times (\boldsymbol{\gamma}_i \times \mathbf{p}_{i+1,i}) + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \dot{\mathbf{p}}_{i+1,i}) \\
&\quad + \dddot{d}_{i+1}\hat{\mathbf{z}}_{i+1} + \ddot{d}_{i+1}\dot{\hat{\mathbf{z}}}_{i+1} + 2\,\ddot{d}_{i+1}\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i+1} \\
&\quad + 2\,\dot{d}_{i+1}\boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i+1} + 2\,\dot{d}_{i+1}\boldsymbol{\omega}_i \times \dot{\hat{\mathbf{z}}}_{i+1}\,.
\end{aligned}
$$

Since $\mathbf{p}_{i+1,i} = \mathbf{p}_{i+1} - \mathbf{p}_i$, it is possible to write $\dot{\mathbf{p}}_{i+1,i} = \dot{\mathbf{p}}_{i+1} - \dot{\mathbf{p}}_i = \mathbf{v}_{i+1} - \mathbf{v}_i$ and to conclude that, owing to (23),

$$
\dot{\mathbf{p}}_{i+1,i} = \boldsymbol{\omega}_i \times \mathbf{p}_{i+1,i} + \dot{d}_{i+1}\,\hat{\mathbf{z}}_{i+1}\,. \qquad (29)
$$

Thus, taking into account (2) and (27), after a few algebraic manipulations, jerk can be expressed as follows

$$
\begin{aligned}
\mathbf{j}_{i+1} &= \mathbf{j}_i + \boldsymbol{\iota}_i \times \mathbf{p}_{i+1,i} + 2\boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i+1,i}) \\
&\quad + \boldsymbol{\omega}_i \times [\boldsymbol{\gamma}_i \times \mathbf{p}_{i+1,i} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i+1,i})] \\
&\quad + \dddot{d}_{i+1}\hat{\mathbf{z}}_{i+1} + 3\,\ddot{d}_{i+1}\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i+1} + 3\,\dot{d}_{i+1}\boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i+1} \\
&\quad + 3\,\dot{d}_{i+1}\boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i+1})\,.
\end{aligned}
$$

Also in this case, the number of involved mathematical operations reduces if jerk is described with respect to frame $i+1$

$$
\begin{aligned}
{}^{i+1}\mathbf{j}_{i+1} &= {}_{i}^{i+1}\mathbf{R}\left\{{}^i\mathbf{j}_i + {}^i\boldsymbol{\iota}_i \times {}^i\mathbf{p}_{i+1,i} + 2\,{}^i\boldsymbol{\gamma}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i+1,i}) \right. \\
&\quad \left. + {}^i\boldsymbol{\omega}_i \times [{}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{i+1,i} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i+1,i})]\right\} \\
&\quad + \dddot{d}_{i+1}\hat{\mathbf{z}}_0 + 3\,{}_{i}^{i+1}\mathbf{R}\,{}^i\boldsymbol{\omega}_i \times \ddot{d}_{i+1}\hat{\mathbf{z}}_0 + 3\,{}_{i}^{i+1}\mathbf{R}\,{}^i\boldsymbol{\gamma}_i \times \dot{d}_{i+1}\hat{\mathbf{z}}_0 \\
&\quad + 3\,{}_{i}^{i+1}\mathbf{R}\,{}^i\boldsymbol{\omega}_i \times ({}_{i}^{i+1}\mathbf{R}\,{}^i\boldsymbol{\omega}_i \times \dot{d}_{i+1}\hat{\mathbf{z}}_0)\,.
\end{aligned}
$$

Equation (8) is obtained straightforward. The same equation can also be used for revolute joints assumed that $\dot{d}_{i+1} = \ddot{d}_{i+1} = \dddot{d}_{i+1} = 0$.

The linear jerk of the mass centre can be written in an iterative fashion starting from the knowledge of the linear jerk of frame i. The expression to be used is still (8), but, now, the role of frame $i+1$ is played by the centre of gravity frame c_i. By considering this substitution and remembering that frames i and c_i are fixed to the

same arm so that we can assume $\dot{d}_{c_i} = \ddot{d}_{c_i} = \dddot{d}_{c_i} = 0$, it is possible to write

$$ {}^{c_i}\mathbf{j}_{c_i} = {}^{c_i}_{i}\mathbf{R} \left\{ {}^{i}\mathbf{j}_i + {}^{i}\boldsymbol{\iota}_i \times {}^{i}\mathbf{p}_{c_i,i} + 2\,{}^{i}\boldsymbol{\gamma}_i \times ({}^{i}\boldsymbol{\omega}_i \times {}^{i}\mathbf{p}_{c_i,i}) + {}^{i}\boldsymbol{\omega}_i \times [{}^{i}\boldsymbol{\gamma}_i \times {}^{i}\mathbf{p}_{c_i,i} + {}^{i}\boldsymbol{\omega}_i \times ({}^{i}\boldsymbol{\omega}_i \times {}^{i}\mathbf{p}_{c_i,i})] \right\} . \quad (30) $$

Equation (9) is obtained by describing (30) with respect to frame i.

C. Recursive formulation of force and torque derivatives acting on the links mass centre

Equation (12) is immediately obtained by differentiating Newton equation $\mathbf{F}_i = m_i\,\mathbf{a}_{c_i}$ and describing the result with respect to frame i.

Similarly, the torque resultant can be expressed by means of the Euler equation

$$ \mathbf{N}_i = \frac{d}{dt}(\mathbf{I}_i\,\boldsymbol{\omega}_i) = \mathbf{I}_i\,\boldsymbol{\gamma}_i + \boldsymbol{\omega}_i \times \mathbf{I}_i\,\boldsymbol{\omega}_i . \quad (31) $$

where $\mathbf{I}_i$ is the inertia tensor evaluated with respect to a frame placed at the link gravity centre and parallel to frame 0. Evidently, $\mathbf{I}_i$ is not constant. Now consider the following transformation rule [16]

$$ \mathbf{I}_i = {}^{0}_{i}\mathbf{R}\,{}^{i}\mathbf{I}_i\,{}^{i}_{0}\mathbf{R} = {}^{0}_{i}\mathbf{R}\,{}^{i}\mathbf{I}_i\,{}^{0}_{i}\mathbf{R}^T , \quad (32) $$

where ${}^{i}\mathbf{I}_i$ is the inertia tensor of link i evaluated with respect to a frame placed at the link gravity centre and fixed to the link body. Tensor ${}^{i}\mathbf{I}_i$, being integral with link i, is constant. The time derivative of $\mathbf{I}_i$ can be obtained from (32)

$$ \dot{\mathbf{I}}_i = \mathbf{S}(\boldsymbol{\omega}_i)\,\mathbf{I}_i - \mathbf{I}_i\,\mathbf{S}(\boldsymbol{\omega}_i) . \quad (33) $$

Let us differentiate (31)

$$ \dot{\mathbf{N}}_i = \dot{\mathbf{I}}_i\,\boldsymbol{\gamma}_i + \mathbf{I}_i\,\boldsymbol{\iota}_i + \boldsymbol{\gamma}_i \times \mathbf{I}_i\,\boldsymbol{\omega}_i + \boldsymbol{\omega}_i \times \frac{d}{dt}(\mathbf{I}_i\,\boldsymbol{\omega}_i) . \quad (34) $$

By virtue of (31) and (33), (34) becomes

$$ \dot{\mathbf{N}}_i = \boldsymbol{\omega}_i \times \mathbf{I}_i\,\boldsymbol{\gamma}_i - \mathbf{I}_i[\boldsymbol{\omega}_i \times \boldsymbol{\gamma}_i] + \mathbf{I}_i\,\boldsymbol{\iota}_i + \boldsymbol{\gamma}_i \times \mathbf{I}_i\,\boldsymbol{\omega}_i + \boldsymbol{\omega}_i \times \mathbf{N}_i . $$

Again, for efficiency reasons, it is better to describe $\dot{\mathbf{N}}_i$ with respect to frame i

$$ {}^{i}\dot{\mathbf{N}}_i = {}^{i}_{0}\mathbf{R}\,\boldsymbol{\omega}_i \times {}^{i}_{0}\mathbf{R}\,\mathbf{I}_i\,\boldsymbol{\gamma}_i - {}^{i}_{0}\mathbf{R}\,\mathbf{I}_i[\boldsymbol{\omega}_i \times \boldsymbol{\gamma}_i] + {}^{i}_{0}\mathbf{R}\,\mathbf{I}_i\,\boldsymbol{\iota}_i + {}^{i}_{0}\mathbf{R}\,\boldsymbol{\gamma}_i \times {}^{i}_{0}\mathbf{R}\,\mathbf{I}_i\,\boldsymbol{\omega}_i + {}^{i}_{0}\mathbf{R}\,\boldsymbol{\omega}_i \times {}^{i}_{0}\mathbf{R}\,\mathbf{N}_i . $$

Owing to (32), it is possible to write

$$ {}^{i}_{0}\mathbf{R}\,\mathbf{I}_i = {}^{i}\mathbf{I}_i\,{}^{i}_{0}\mathbf{R} , $$

so that ${}^{i}\dot{\mathbf{N}}_i$ can be written as follows

$$ {}^{i}\dot{\mathbf{N}}_i = {}^{i}_{0}\mathbf{R}\,\boldsymbol{\omega}_i \times {}^{i}\mathbf{I}_i\,{}^{i}_{0}\mathbf{R}\,\boldsymbol{\gamma}_i - {}^{i}\mathbf{I}_i\,{}^{i}_{0}\mathbf{R}[\boldsymbol{\omega}_i \times \boldsymbol{\gamma}_i] + {}^{i}\mathbf{I}_i\,{}^{i}_{0}\mathbf{R}\,\boldsymbol{\iota}_i + {}^{i}_{0}\mathbf{R}\,\boldsymbol{\gamma}_i \times {}^{i}\mathbf{I}_i\,{}^{i}_{0}\mathbf{R}\,\boldsymbol{\omega}_i + {}^{i}_{0}\mathbf{R}\,\boldsymbol{\omega}_i \times {}^{i}_{0}\mathbf{R}\,\mathbf{N}_i , $$

and, in turn, as (13).

D. Recursive formulation of the joints generalized force derivatives

It is known that, for a rigid manipulator, the force resultant is given by the sum of the joint forces, i.e., $\mathbf{F}_i = \mathbf{f}_i - \mathbf{f}_{i+1}$. Equation (16) is immediately obtained by differentiating this equation and describing it with respect to frame i.

Similarly, the torque balance, evaluated with respect to the centre of gravity, can be expressed as

$$ \mathbf{n}_i - \mathbf{n}_{i+1} + \mathbf{f}_i \times \mathbf{p}_{c_i,i} - \mathbf{f}_{i+1} \times \mathbf{p}_{c_i,i+1} = \mathbf{N}_i . \quad (35) $$

In order to simplify the subsequent differentiation operation, (35) can be reorganized as follows (see also Fig. 1)

$$ \mathbf{n}_i = \mathbf{n}_{i+1} + {}^{0}_{i}\mathbf{R}\,{}^{i}\mathbf{p}_{c_i,i} \times \mathbf{F}_i + \mathbf{p}_{i+1,i} \times \mathbf{f}_{i+1} + \mathbf{N}_i . $$

The time derivative of $\mathbf{n}_i$ can be evaluated by taking into account that $\dot{\mathbf{p}}_{i+1,i}$ is given by (29), while ${}^{i}\mathbf{p}_{c_i,i}$ is constant

$$ \dot{\mathbf{n}}_i = \dot{\mathbf{n}}_{i+1} + (\omega_i \times {}^{0}_{i}\mathbf{R}\,{}^{i}\mathbf{p}_{c_i,i}) \times \mathbf{F}_i + {}^{0}_{i}\mathbf{R}\,{}^{i}\mathbf{p}_{c_i,i} \times \dot{\mathbf{F}}_i + (\boldsymbol{\omega}_i \times \mathbf{p}_{i+1,i} + \dot{d}_{i+1}\,\hat{\mathbf{z}}_{i+1}) \times \mathbf{f}_{i+1} + \mathbf{p}_{i+1,i} \times \dot{\mathbf{f}}_{i+1} + \dot{\mathbf{N}}_i . $$

Finally, (17) is obtained by describing $\dot{\mathbf{n}}_i$ with respect to frame i.

The last step of the recursive algorithm evaluates the derivative of the joint GFs. First consider a revolute joint. Scalar products are not influenced by the reference frame used to describe its operands. Thus, (18) can be equivalently written as follows

$$ \tau_i = {}^{i}\mathbf{n}_i^T\,\hat{\mathbf{z}}_0 = {}^{i}\mathbf{n}_i^T\,{}^{i}\hat{\mathbf{z}}_i = \mathbf{n}_i^T\,\hat{\mathbf{z}}_i . $$

By differentiating with respect to time and taking into account (2), it follows that

$$ \dot{\tau}_i = \dot{\mathbf{n}}_i^T\,\hat{\mathbf{z}}_i + \mathbf{n}_i^T\,\dot{\hat{\mathbf{z}}}_i = \dot{\mathbf{n}}_i^T\,\hat{\mathbf{z}}_i + \mathbf{n}_i^T\,(\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_i) . $$

Now, express all the operands with respect to frame i

$$ \dot{\tau}_i = {}^{i}\dot{\mathbf{n}}_i^T\,{}^{i}\hat{\mathbf{z}}_i + {}^{i}\mathbf{n}_i^T\,({}^{i}\boldsymbol{\omega}_i \times {}^{i}\hat{\mathbf{z}}_i) = {}^{i}\dot{\mathbf{n}}_i^T\,\hat{\mathbf{z}}_0 + {}^{i}\mathbf{n}_i^T\,({}^{i}\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_0) . $$

The second term is a triple scalar product: its result does not change if the scalar and the cross product operators are exchanged

$$ \dot{\tau}_i = {}^{i}\dot{\mathbf{n}}_i^T\,\hat{\mathbf{z}}_0 + ({}^{i}\mathbf{n}_i \times {}^{i}\boldsymbol{\omega}_i)^T\hat{\mathbf{z}}_0 . $$

Equation (20) is finally obtained by collecting $\hat{\mathbf{z}}_0$.

In case of prismatic joints, motion depends on the joint forces. Equation (21) is obtained by differentiating (19) and considering the same algebraic manipulations proposed for revolute joints. Passages are omitted for brevity.

V. Computational considerations

The algorithm efficiency can be assessed on the basis of the number of involved mathematical operations. As usual, the computational burden is estimated by considering a generic manipulator with revolute joints under the hypothesis of an immobile base frame and taking into account the existence of a gravitational field. In order to figure out the worst situation, inertia tensors are supposed to be full matrices and external forces are applied to the last link.

Many terms of the iterative algorithm are common to several equations. For example, ${}^{i}_{i+1}\mathbf{R}\,{}^{i+1}\mathbf{f}_{i+1}$ appears three times in the backward recursion. By evaluating multiply repeated terms only a single time (e.g., ${}^{i+1}_{i}\mathbf{R}\,{}^{i}\boldsymbol{\omega}_i$, ${}^{i+1}_{i}\mathbf{R}\,{}^{i}\boldsymbol{\gamma}_i$, etc..), it is possible to greatly reduce the number of required mathematical operations: the current implementation evaluates GFs and GFDs by means of $309N$-162 multiplications and $268N$-138 additions. The new iterative algorithm, like any classic Newton-Euler approach, involves a number of operations which linearly depends on the number of joints. The computational load approximately doubles with respect to classic algorithms which only evaluate GFs, but evaluation times are still compatible with real-time control loops, as it will be shown in the following.

It is interesting to observe that the analogous algorithm proposed in [12], based on the standard Denavit-Hartenberg frame allocation, evaluates generalized force derivatives by means of $327N$-81 multiplications and $274N$-72 additions. This confirms the common idea that algorithms based on the modified Denavit-Hartenberg method are slightly more efficient.

The computational load can further be reduced by specializing (3)–(21) for specific applications. For example, the number of involved operations dramatically decreases when diagonal inertia tensors are considered or when the algorithm is optimized for a specific robot by means of symbolic techniques similar to those proposed in [19], [20].

The iterative algorithm (3)–(21) has been tested by considering a Stanford manipulator (2R)P(3R) whose kinematic and dynamic parameters are reported in Table I and II respectively [21]. The

TABLE I
KINEMATIC PARAMETERS OF A STANFORD MANIPULATOR

i	a_i	α_i	d_{i+1}	θ_{i+1}
0	0	0	0.412	θ_1
1	0	$-\pi/2$	0.154	θ_2
2	0	$\pi/2$	d_3	$-\pi/2$
3	0	0	0	θ_4
4	0	$-\pi/2$	0	θ_5
5	0	$\pi/2$	0.263	θ_6

TABLE II
DYNAMIC PARAMETERS OF A STANFORD MANIPULATOR.

Link	Mass	x (m)	y (m)	z (m)	I_{xx} $(Kg.m^2)$	I_{yy} $(Kg.m^2)$	I_{zz} $(Kg.m^2)$
1	9.29	0	-0.1105	-0.0175	0.276	0.071	0.255
2	5.01	0	0	-0.1054	0.108	0.100	0.018
3	4.25	0	0	-0.6447	2.51	2.51	0.006
4	1.08	0	-0.0054	-0.0092	0.002	0.001	0.001
5	0.63	0	-0.0566	0	0.003	0.0004	0.003
6	0.51	0	0	0.1554	0.013	0.013	0.0003

manipulator was chosen since it has both revolute and prismatic joints: in this way, all possible combinations of (3)–(21) can be checked. The aim is to verify the approach correctness and the algorithm efficiency.

A point-to-point movement has been planned in the joint space by means of quintic order splines which satisfy the boundary conditions of Table III. The total travelling time is $t_f = 2$ s. Initial and final velocities, accelerations, and jerks are equal to zero. The gravitational field acceleration is equal to 9.81 ms^{-2}.

The algorithm correctness has been verified by comparing GFDs evaluated by means of (3)–(21) with those obtained by numerically differentiating τ_i according to the following equation

$$\dot{\tilde{\tau}}_i(t_k) = \frac{\tau_i(t_k) - \tau_i(t_{k-1})}{T_s} \ , \tag{36}$$

where T_s is a discrete sampling time and $t_k := k\,T_s$, $k \in \mathbb{N}$. In particular, the chosen performance index is the maximum of $|\Delta\dot{\tau}_i(t_k)| = \left|\dot{\tau}_i(t_k) - \dot{\tilde{\tau}}_i(t_k)\right|$, evaluated for $t_k \in [0, \ t_f]$, i.e., the maximum difference between $\dot{\tau}_i$ evaluated by means of (3)–(21) and the corresponding $\dot{\tilde{\tau}}_i$ evaluated by means of (36). Necessarily, numerical differentiation only returns a rough estimation of actual derivatives: the accuracy depends on the assumed sampling time. For example, for the first joint $\max_{t_k}\{|\Delta\dot{\tau}_1(t_k)|\} = 2.462\text{e-}2$ Nms^{-1} when T_s=1.0e-4 s while it increases of one order of magnitude when T_s=1.0e-3 s. Another order of magnitude is lost if T_s=1.0e-2 s. Similar figures hold for all the joints. This result highlights a good agreement between numerically and analytically evaluated derivatives, thus confirming the correctness of (3)–(21), but, at the same time, it points out the relevance of using exact expressions for the evaluation of $\dot{\tau}_i$ in order to avoid any dependence of the accuracy on T_s.

The computational burden has been verified by means of a Pentium Pc, 1400 MHz, OS Windows XP SP2, under the Matlab environment. For the considered Stanford manipulator, τ_i and $\dot{\tau}_i$ are calculated on average in 5.255e-5 s, while classic Newton-Euler algorithm evaluates the sole τ_i in 2.399e-5 s: nevertheless the Matlab environment is not optimized for speed, computational times are absolutely compatible with those required by online applications.

VI. CONCLUSIONS

A recursive algorithm, able to evaluate generalized force derivatives for a robotic manipulator, has been proposed in the paper. It has been explicitly developed for online applications which require accuracy and a moderate computational burden. Accuracy is gained by using iterative closed form expressions instead of approximate numerical differentiation methods. Any effort has been spent to minimize the number of involved mathematical operations which linearly depend on the number of joints. The number of required operations can further be reduced if the algorithm is optimized for a specific manipulator. An important advantage of the proposed method is that it is an extension of analogous algorithms currently used to solve the manipulators inverse dynamics: the new functionalities can be obtained by simply updating existing procedures.

TABLE III
INITIAL AND FINAL VALUES OF THE JOINT VARIABLES USED FOR THE TRAJECTORY PLANNING.

Joint Variable	θ_1	θ_2	d_3	θ_4	θ_5	θ_6
Initial Value	$\pi/2$	0	0	$\pi/2$	$\pi/4$	$\pi/2$
Final Value	$\pi/4$	$\pi/2$	2	$\pi/4$	$\pi/2$	0

REFERENCES

[1] L. Žlajpah, "On time optimal path control of manipulators with bounded joint velocities and torques," in *Procs. of the 1996 Int. Conf. on Robotics and Automation*, Minneapolis, Minnesota, April 1996, pp. 1572–1577.

[2] O. Dahl and L. Nielsen, "Torque-limited path following by on-line trajectory time scaling," *IEEE Trans. on Robotics and Automation*, vol. 6, no. 5, pp. 554–561, 1990.

[3] C.-S. Lin, P.-R. Chang, and J. Luh, "Formulation and optimization of cubic polynomial joint trajectories for industrial robots," *IEEE Trans. Automatic Control*, vol. AC-28, no. 12, pp. 1066–1074, 1983.

[4] S. Macfarlane and E. A. Croft, "Jerk-bounded manipulator trajectory planning: design for real-time applications," *IEEE Trans. on Robotics and Automation*, vol. 19, no. 1, pp. 42–52, 2003.

[5] A. De Luca, L. Lanari, and G. Oriolo, "A sensitivity approach to optimal spline robot trajectories," *Automatica*, vol. 27, no. 3, pp. 535–539, 1991.

[6] C. Guarino Lo Bianco and A. Piazzi, "A genetic/interval approach to optimal trajectory planning of industrial robots under torque constraints," in *Procs. of the 1999 European Control Conference*, Karlsruhe, Germany, Sept. 1999.

[7] ——, "A semi-infinite optimization approach to optimal trajectory planning of mechanical manipulators," in *Semi-Infinite Programming: Recent Advances*. Kluwer, 2001, ch. 13, pp. 271–297.

[8] K. G. Shin and N. D. McKay, "A dynamic programming approach to trajectory planning of robotic manipulators," *IEEE Trans. on Automatic Control*, vol. AC-31, no. 6, pp. 491–500, June 1986.

[9] H. H. Tan and R. B. Potts, "Minimum time trajectory planner for the discrete dynamic robot model with dynamic constraints," *IEEE J. of Robotics and Automation*, vol. 4, no. 2, pp. 174–185, April 1988.

[10] D. Constantinescu and E. A. Croft, "Smooth and time-optimal trajectory planning for industrial manipulators along specified paths," *J. of Robotic Systems*, vol. 17, no. 5, pp. 233–249, 2000.

[11] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, "On-line computational scheme for mechanical manipulators," *ASME J. of Dynamic Systems, Measurement, and Control*, vol. 102, pp. 69–76, 1980.

[12] C. Guarino Lo Bianco and E. Fantini, "A recursive newton-euler approach for the evaluation of generalized forces derivatives," in *IEEE 12th Int. Conf. on Methods and Models in Automation and Robotics*, Miedzyzdroje, Poland, August 2006, pp. 739–744.

[13] C. Guarino Lo Bianco and A. Piazzi, "Minimum-time trajectory planning of mechanical manipulators under dynamic constraints," *Int. J. of Control*, vol. 75, no. 13, pp. 967–980, 2002.

[14] O. Gerelli and C. Guarino Lo Bianco, "Real-time path-tracking control of robotic manipulators with bounded torques and torque-derivatives," in *2008 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS 2008*, Nice, France, Sept. 2008, pp. 532–537.

[15] W. Khalil and J.-F. Kleinfinger, "A new geometric notation for open and closed-loop robots," in *Proc. of the IEEE Conf. on Robotics and Automation*, San Francisco, CA, 1986, pp. 1174–1180.

[16] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*, ser. Advanced Textbooks in Control and Signal Processing. Berlin, Germany: Springer-Verlag, 2000.

[17] W. W. Hooker and G. Margulies, "The dynamical attitude equations for an n body satellite," *J. of Astronomical Sciences*, vol. 12, no. 4, pp. 123–128, 1965.

[18] D. E. Orin, R. B. McGhee, M. Vukobratović, and G. Hatoch, "Kinematic and kinetic analysis of open-chain linkages utilizing newton-euler methods," *Mathematical Biosciences*, vol. 43, no. 1/2, pp. 107–130, February 1979.

[19] W. Khalil and J.-F. Kleinfinger, "Minimum operations and minimum parameters of the dynamic model of tree structure robots," *IEEE J. of Robotics and Automation*, vol. 3, no. 6, pp. 517–526, 1987.
[20] M. Renaud, "Quasi-minimal computation of the dynamic model of a robot manipulator utilizing the newton-euler formalism and the notion of augmented body," in *IEEE Int. Conf. on Robotics and Automation*, Railegh, USA, March-April 1987, pp. 1677–1682.
[21] R. Paul, *Robot manipulators: mathematics, programming and control*. M.I.T. Press, 1981.