On the closed form computation of the dynamic matrices and their differentiations

Gianluca Garofalo, Christian Ott and Alin Albu-Schäffer

***Abstract*— In this paper we review and extend some classic results on rigid body dynamics, in order to give a symbolic expression of the different derivatives of the matrices of the dynamic model of a general tree-structured robot. In what follows the matrices are differentiated with respect to time, state and dynamic parameters. Obviously from the derivatives of the single matrices it is possible to recover the derivatives of the direct and inverse dynamic functions and classic results like the regressor matrix. Moreover an iterative algorithm is sketched which allows to compute all these derivatives as well as the kinematics and dynamics of the robot.**

I. Introduction

Because of the steady increase of the complexity of robotic systems and their simulation and control, the dynamic equations of motion of the robot have been analysed since decades. Both through the Lagrangian and the Newton-Euler approach these equations were derived and used by many authors with different purposes and from different points of view. Either for simulation or control both the direct and inverse dynamic problem have been considered [1]. Among the first works on inverse dynamics for serial chain robots there are those of Uicker [2] and Stepanenko and Vukobratovic [3] who formulated the recursive Newton-Euler algorithm. In order to obtain a more efficient computation, later Orin et al. [4] reformulated their work. Hollerbach [5] showed that also the Lagrangian formulation could provide an equally efficient algorithm and finally Silver [6] provided the equivalence of the two methods. Using also the results of Vereshchagin [7], for both the inverse and the direct dynamic problem algorithms are available with $O(n)$ complexity. Other authors investigated the possibility of using more elegant and efficient tools to write the dynamics. Important are the works of Featherstone [8], Rodriguez [9] and Park et al. [10], where spatial operator algebra and Lie groups are respectively used. On the other hand the necessity of solving identification problems brought Atkeson et al. [11], Khosla and Kanade [12], as well as Kawasaki and Nishimura [13] to introduce the regressor matrix. Later Gautier and Khalil [14] focused on the determination of the minimum set of inertial parameters to reduce the computational cost and simplify the identification. The linearity in the dynamic parameter suggested the development of adaptive controllers like the first one in [15] by Craig et al. In order to avoid acceleration feedback, alternative algorithms were developed by Hsu et al. [16] and Middleton and Goodwin [17], but it was especially thanks to the introduction of a novel regressor matrix that Slotine and Li [18] proved the global asymptotic stability of their controller completely avoiding any information on the current acceleration. An excellent algorithm to compute such regressor matrix was presented in [19] by Yuan and Yuan.

More recently the introduction of flexible joints has risen the need of computing the derivatives with respect to time of all the matrices of the dynamic model [20], [21], [22]. The solution of other problems and analysis requires, instead, to differentiate the matrices with respect to the state. The controllability analysis of underactuated manipulators [23] motivated Müller [24] to provide an efficient factorisation for the inverse of the inertia matrix, in order to compute its partial derivatives. On the other hand for optimisation problems the derivative with respect to the state of the direct and inverse dynamic function were provided in [25] and [26]. The linearisation of the dynamics is also useful in state estimation problems and in general whenever the use of the extended Kalman filter [27] is required.

From the discussion it follows that the problem of differentiating the direct and/or inverse dynamic functions is a well known and analysed problem, however a summary and generalisation in which the derivative of each matrix of the dynamic model is provided with respect to time, state and dynamic parameters, it is still not present. This is the main contribution of our paper. To this end we first present in the next section the formulation of the dynamics for a system like the one sketched in Fig. 1, based on the works [6], [28], [29]. In this way we will provide the symbolic expression of each matrix of the dynamic equation, which in the following sections will be differentiated with respect to either time, or state or dynamic parameters. Finally we will sketch an iterative algorithm, totally analogue to the outward recursion of the Newton-Euler algorithm, which will provide a simple way to compute the derivatives. It is worth to notice that, as it will be clear from the following analysis, the computation of all the matrices and their derivatives is mainly solved after the computation of the direct kinematics and differential kinematics.

For the reader not familiar with twists and wrenches we have collected in the appendix the expression of the matrices used in the rest of the paper. For the twist coordinates we have chosen the convention with the linear part for the top three rows and the angular part for the bottom ones [28]. All the other matrices are defined accordingly. Nevertheless, throughout the paper we avoid to write the explicit expression of the matrices, so the reader used to the opposite convention will have no problem to follow the equations.

G. Garofalo, C. Ott and A. Albu-Schäffer are with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Wessling, Germany. gianluca.garofalo@dlr.de

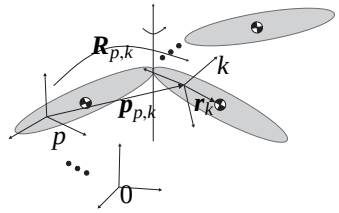

Fig. 1. The mechanical systems considered in the paper are general branched connections of rigid bodies.

II. The Newton-Euler equation in body coordinates

Let us first consider a frame attached to a rigid body (which will be indicated with the index k) in movement with respect to a spatial frame (which will be indicated with the index 0). Using the body twist $\boldsymbol{\nu}_k$ and the body wrench $\boldsymbol{f}_k$, the equation of motion can be written as

$$\frac{d}{dt}\left(\mathrm{Ad}_{0,k}^{-T}\,\boldsymbol{\Lambda}_k\boldsymbol{\nu}_k\right) = \mathrm{Ad}_{0,k}^{-T}\,\boldsymbol{f}_k\ , \tag{1}$$

where $\boldsymbol{\Lambda}_k$ is the constant body inertia matrix and $\mathrm{Ad}_{0,k}$ is the adjoint matrix which uses an element of the Lie group (the homogeneous transformation matrix from the inertial to the body frame) as a linear mapping on the Lie algebra [28]. Computing the time derivative in (1) we obtain

$$\boldsymbol{\Lambda}_k\dot{\boldsymbol{\nu}}_k - \mathrm{adj}_{0,k}^{T}\,\boldsymbol{\Lambda}_k\boldsymbol{\nu}_k = \boldsymbol{f}_k\ , \tag{2}$$

where $\dot{\mathrm{Ad}}_{0,k}^{-1} = -\,\mathrm{adj}_{0,k}\,\mathrm{Ad}_{0,k}^{-1}$, being $\mathrm{adj}_{0,k}$ the Lie bracket matrix which uses an element of the Lie algebra (the body twist $\boldsymbol{\nu}_k$) as a linear mapping on the Lie algebra itself [28]. Finally using the property $\mathrm{adj}_{0,k}\,\boldsymbol{\nu}_k = \mathbf{0}$ we can add the term $\boldsymbol{\Lambda}_k\,\mathrm{adj}_{0,k}\,\boldsymbol{\nu}_k$ to (2) without changing the equation, so that it can be rewritten as

$$\boldsymbol{\Lambda}_k\dot{\boldsymbol{\nu}}_k + \boldsymbol{\Psi}_k\boldsymbol{\nu}_k = \boldsymbol{f}_k\ , \tag{3}$$

where $\boldsymbol{\Psi}_k = \left(\boldsymbol{\Lambda}_k\,\mathrm{adj}_{0,k} - \mathrm{adj}_{0,k}^{T}\,\boldsymbol{\Lambda}_k\right)$.

Remark 1: Since $\boldsymbol{\Psi}_k$ is skew symmetric ($\boldsymbol{\Psi}_k = -\boldsymbol{\Psi}_k^T$) and $\boldsymbol{\Lambda}_k$ is constant, the property $\dot{\boldsymbol{\Lambda}}_k = \boldsymbol{\Psi}_k + \boldsymbol{\Psi}_k^T$ is satisfied.

Remark 2: Due to the important property $\mathrm{adj}_{0,k}\,\boldsymbol{\nu}_k = \mathbf{0}$ we can conclude that $\dot{\boldsymbol{\sigma}}_k = \mathrm{Ad}_{0,k}\,\dot{\boldsymbol{\nu}}_k$, where $\boldsymbol{\sigma}_k$ is the spatial twist.

Let us assume now to have N bodies. The N equations of motions in the form of (3) can be written stacking the twists in $\boldsymbol{\nu} = \mathrm{col}\,(\boldsymbol{\nu}_k)$, the wrenches in $\boldsymbol{f} = \mathrm{col}\,(\boldsymbol{f}_k)$ and using the block diagonal matrices $\boldsymbol{\Lambda} = \mathrm{blkdiag}\,(\boldsymbol{\Lambda}_k)$ and $\boldsymbol{\Psi} = \mathrm{blkdiag}\,(\boldsymbol{\Psi}_k)$, where $k = 1, ..., N$. If the bodies are constrained to each other, then it is possible to project each of the equations in the form of (3) in the space orthogonal to the constraint reaction forces. Using a minimal set of coordinates $\boldsymbol{q}$ to identify the configuration and assuming a mapping in the form $\boldsymbol{\nu}_k = \boldsymbol{J}_k\dot{\boldsymbol{q}}$, then this is possible through $\boldsymbol{J}_k^T$, so that

$$\boldsymbol{J}^T\left[\boldsymbol{\Lambda}\boldsymbol{J}\ddot{\boldsymbol{q}} + \left(\boldsymbol{\Psi}\boldsymbol{J} + \boldsymbol{\Lambda}\dot{\boldsymbol{J}}\right)\dot{\boldsymbol{q}}\right] = \boldsymbol{\tau} - \boldsymbol{J}^T\boldsymbol{\Lambda}\boldsymbol{\gamma}\ , \tag{4}$$

where $\boldsymbol{J} = \mathrm{col}\,(\boldsymbol{J}_k)$ and each $\boldsymbol{f}_k = \boldsymbol{f}_{k_c} + \boldsymbol{f}_{k_\gamma} + \boldsymbol{f}_{k_\tau}$ has been split in

- constraint reaction forces (which disappear after the projection, i.e $\boldsymbol{J}_k^T\boldsymbol{f}_{k_c} = \mathbf{0}$),
- the weight (which is linear in $\boldsymbol{\gamma} = \mathrm{col}\,(\boldsymbol{\gamma}_k)$, being $\boldsymbol{\gamma}_k$ the body gravitational acceleration, i.e $\boldsymbol{f}_{k_\gamma} = -\boldsymbol{\Lambda}_k\boldsymbol{\gamma}_k$),
- all the other external forces (which after the projection correspond to $\boldsymbol{\tau}$, i.e $\boldsymbol{J}_k^T\boldsymbol{f}_{k_\tau} = \boldsymbol{\tau}$).

Using Remark 2 we can write $\boldsymbol{\gamma}_k$ as the product of a configuration dependent part and a constant part, i.e. $\boldsymbol{\gamma}_k = \mathrm{Ad}_{0,k}^{-1}\,\boldsymbol{\gamma}_0$, where $\boldsymbol{\gamma}_0$ is the constant spatial gravitational acceleration[1].

Equation (4) can be written in a more compact way as

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{g} = \boldsymbol{\tau}\ , \tag{5}$$

where

$$\boldsymbol{M} = \sum_k \boldsymbol{J}_k^T\boldsymbol{\Lambda}_k\boldsymbol{J}_k\ , \tag{6a}$$

$$\boldsymbol{C} = \sum_k \boldsymbol{J}_k^T\left[\left(\boldsymbol{\Lambda}_k\,\mathrm{adj}_{0,k} - \mathrm{adj}_{0,k}^{T}\,\boldsymbol{\Lambda}_k\right)\boldsymbol{J}_k + \boldsymbol{\Lambda}_k\dot{\boldsymbol{J}}_k\right]\ , \tag{6b}$$

$$\boldsymbol{g} = \sum_k \boldsymbol{J}_k^T\boldsymbol{\Lambda}_k\,\mathrm{Ad}_{0,k}^{-1}\,\boldsymbol{\gamma}_0\ , \tag{6c}$$

$k = 1, ..., N$. While $\boldsymbol{\Lambda}_k$ is constant, each of the matrices $\boldsymbol{J}_k$, $\mathrm{Ad}_{0,k}^{-1}$ and $\mathrm{adj}_{0,k}$ are state dependent and are the only quantities that must be computed to use the formulas in (6).

Remark 3: The vector $\boldsymbol{g}$ is the mapping of the weight wrench $m\boldsymbol{\gamma}_0$ in the configuration space, where m is the total mass. Since the weight wrench can always be thought to be applied in the total centre of mass (CoM), it is clear that the transpose of the matrix multiplying $\boldsymbol{\gamma}_0$ in (6c) is related to $\boldsymbol{J}_{CoM}$, which maps the velocity $\dot{\boldsymbol{q}}$ into the CoM velocity. Writing $\boldsymbol{g}$ in terms of stacking matrices, we derive

$$\boldsymbol{J}_{CoM} = \frac{1}{m}\boldsymbol{P}^T\boldsymbol{\Lambda}\boldsymbol{J}\ , \tag{7}$$

where $\boldsymbol{P}$ is the matrix obtained taking only the columns of $\mathrm{col}\left(\mathrm{Ad}_{0,k}^{-1}\right)$, $k = 1, ..., N$ which multiply the linear part of $\boldsymbol{\gamma}_0$.

III. Parameters differentiation

In order to compute the derivative of (6) with the respect to the dynamic parameters, we will take advantage of the linearity of the matrices in the parameters themselves.

Proposition 1: If $\boldsymbol{\upsilon}_k$ and $\boldsymbol{\omega}_k$ are the linear and angular part of the twist $\boldsymbol{\nu}_k$ respectively, then the momentum $\boldsymbol{\Lambda}_k\boldsymbol{\nu}_k$ can be written as

$$\boldsymbol{\Lambda}_k\boldsymbol{\nu}_k = \boldsymbol{A}(\boldsymbol{\nu}_k)\boldsymbol{\pi}_k\ , \tag{8}$$

where

$$\boldsymbol{\pi}_k = \begin{bmatrix} m_k & m_k\boldsymbol{r}_k^T & I_{xx_k} & I_{yy_k} & I_{zz_k} & I_{xy_k} & I_{xz_k} & I_{yz_k}\end{bmatrix}^T\ , \tag{9}$$

and $\boldsymbol{r}_k$ is the position of the centre of mass of the $k-th$ body with respect to its own frame (see Fig. 1). Moreover, the rows of $\boldsymbol{A}(\boldsymbol{\nu}_k) \in \mathbb{R}^{6\times 10}$ for the linear momentum are

$$\boldsymbol{A}_\upsilon = \begin{bmatrix}\boldsymbol{\upsilon}_k & \tilde{\boldsymbol{\omega}}_k & \boldsymbol{O}_{3\times 6}\end{bmatrix} \in \mathbb{R}^{3\times 10}\ , \tag{10}$$

and the ones for the angular momentum are

$$\boldsymbol{A}_\omega = \begin{bmatrix}\boldsymbol{0} & -\tilde{\boldsymbol{\upsilon}}_k & \tilde{\tilde{\boldsymbol{\omega}}}_k\end{bmatrix} \in \mathbb{R}^{3\times 10}\ , \tag{11}$$

[1]The angular part of $\boldsymbol{\gamma}_0$ is always zero, while assuming for example that the third axis is the vertical one, then the linear part will be $\begin{bmatrix}0 & 0 & g\end{bmatrix}^T$, where g is the gravitational acceleration constant.

where $\boldsymbol{O}_{3\times6} \in \mathbb{R}^{3\times6}$ is a matrix of zeros, $\tilde{\boldsymbol{a}}$ is the skew symmetric matrix such that $\tilde{\boldsymbol{a}}\boldsymbol{b} = \boldsymbol{a}\times\boldsymbol{b}$ and

$$\tilde{\tilde{\boldsymbol{a}}} = \begin{bmatrix} a_x & 0 & 0 & a_y & a_z & 0 \\ 0 & a_y & 0 & a_x & 0 & a_z \\ 0 & 0 & a_z & 0 & a_x & a_y \end{bmatrix} , \tag{12}$$

Proposition 2 (Parameters differentiation): Given the expression of the dynamic matrices in (6) and the factorisation presented in Proposition 1, then the derivatives of (6) with respect to the dynamic parameters are

$$\frac{\partial \boldsymbol{M}^j}{\partial \boldsymbol{\pi}_{k_h}} = \boldsymbol{J}_k^T \boldsymbol{A}(\boldsymbol{J}_k^j)^h , \tag{13a}$$

$$\frac{\partial \boldsymbol{C}^j}{\partial \boldsymbol{\pi}_{k_h}} = \boldsymbol{J}_k^T \left(\boldsymbol{A}(\mathrm{adj}_{0,k}\, \boldsymbol{J}_k^j + \dot{\boldsymbol{J}}_k^j)^h - \mathrm{adj}_{0,k}^T\, \boldsymbol{A}(\boldsymbol{J}_k^j)^h \right) , \tag{13b}$$

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\pi}_{k_h}} = \boldsymbol{J}_k^T \boldsymbol{A}(\mathrm{Ad}_{0,k}^{-1}\, \boldsymbol{\gamma}_0)^h , \tag{13c}$$

where, for example, with the superscript h we indicate the $h-th$ column of the corresponding matrix.

Proof: Let us just consider the computation for the inertia matrix, since the same can be repeated for the others. From (6) we can write

$$\boldsymbol{M}^j = \sum_k \boldsymbol{J}_k^T \boldsymbol{\Lambda}_k \boldsymbol{J}_k^j . \tag{14}$$

Applying the factorisation in Proposition 1 we can write $\boldsymbol{\Lambda}_k \boldsymbol{J}_k^j = \boldsymbol{A}(\boldsymbol{J}_k^j)\boldsymbol{\pi}_k$. Differentiating the expression with respect to $\boldsymbol{\pi}_{k_h}$, we prove the proposition. ∎

Remark 4: From (6) and Proposition 1 is also possible to easily compute the Slotine-Li regressor $\boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \dot{\boldsymbol{q}}_r, \ddot{\boldsymbol{q}}_r)$. Using $\boldsymbol{\pi} = \mathrm{col}(\boldsymbol{\pi}_k)$, the identity $\boldsymbol{M}\ddot{\boldsymbol{q}}_r + \boldsymbol{C}\dot{\boldsymbol{q}}_r + \boldsymbol{g} = \boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \dot{\boldsymbol{q}}_r, \ddot{\boldsymbol{q}}_r)\boldsymbol{\pi}$ holds, where

$$\boldsymbol{Y}\boldsymbol{\pi} = \sum_k \boldsymbol{J}_k^T \left(\boldsymbol{A}(\alpha_k) - \mathrm{adj}_{0,k}^T\, \boldsymbol{A}(\boldsymbol{J}_k \dot{\boldsymbol{q}}_r) \right) \boldsymbol{\pi}_k , \tag{15}$$

$\boldsymbol{\alpha}_k = \boldsymbol{J}_k \ddot{\boldsymbol{q}}_r + \mathrm{adj}_{0,k}\, \boldsymbol{J}_k \dot{\boldsymbol{q}}_r + \dot{\boldsymbol{J}}_k \dot{\boldsymbol{q}}_r + \boldsymbol{\gamma}_k$ and $k = 1, ..., N$. Rewriting (15) in terms of stacking matrices we obtain the expression of the regressor

$$\boldsymbol{Y} = \boldsymbol{J}^T\ \mathrm{blkdiag}\left(\boldsymbol{A}(\alpha_k) - \mathrm{adj}_{0,k}^T\, \boldsymbol{A}(\boldsymbol{J}_k \dot{\boldsymbol{q}}_r) \right) , \tag{16}$$

$k = 1, ..., N$. Given the Slotine-Li regressor, the classic regressor can be obtained replacing $\dot{\boldsymbol{q}}_r = \dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}_r = \ddot{\boldsymbol{q}}$.

IV. State differentiation

Although quite involved, the key point for the computation of the derivatives with respect to the state is to find a convenient factorisation for $\mathrm{adj}_{0,k}$; similarly to what we did in Sec. III for the momentum. From (6), we can conclude that the derivatives of $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{g}$ can be computed if the derivative of $\boldsymbol{J}_k$, $\dot{\boldsymbol{J}}_k$, $\mathrm{Ad}_{0,k}^{-1}$ and $\mathrm{adj}_{0,k}$ are available. The computation of $\boldsymbol{J}_k$ and all its necessary derivatives is treated in Sec. VI, so here we focus on the other two matrices.

In this section (with an abuse of notation) we will use $\mathrm{blkdiag}(\boldsymbol{a})$ to indicate a block diagonal matrix with the element $\boldsymbol{a}$ repeated six times on the diagonal.

Proposition 3: The matrix $\mathrm{adj}_{0,k}$ can be factorised as the product of a constant matrix[2] $\boldsymbol{W} \in \mathbb{R}^{6\times36}$ and a block diagonal matrix $\boldsymbol{V}_k \in \mathbb{R}^{36\times6}$

$$\mathrm{adj}_{0,k} = \boldsymbol{W}\boldsymbol{V}_k , \tag{17}$$

where $\boldsymbol{V}_k = \mathrm{blkdiag}(\boldsymbol{\nu}_k)$ and $\boldsymbol{W}$ selects the necessary entries from the twist $\boldsymbol{\nu}_k$ to produce either $\tilde{\boldsymbol{v}}_k$ or $\tilde{\boldsymbol{\omega}}_k$.

Proposition 4: Given the factorisation of $\mathrm{adj}_{0,k}$ in Proposition 3 and the identity $\boldsymbol{J}_k\dot{\boldsymbol{q}} = \sum_h \boldsymbol{J}_k^h \dot{\boldsymbol{q}}_h$, it follows that

$$\frac{\partial\, \mathrm{Ad}_{0,k}^{-1}}{\partial \boldsymbol{q}_h} = -\,\mathrm{adj}_{\boldsymbol{J}_k^h}\, \mathrm{Ad}_{0,k}^{-1} , \tag{18}$$

$$\frac{\partial\, \mathrm{adj}_{0,k}}{\partial \dot{\boldsymbol{q}}_h} = \mathrm{adj}_{\boldsymbol{J}_k^h} , \tag{19}$$

where $\mathrm{adj}_{\boldsymbol{J}_k^h}$ is the matrix computed using $\boldsymbol{J}_k^h \in \mathbb{R}^6$, instead of $\boldsymbol{\nu}_k \in \mathbb{R}^6$, while

$$\frac{\partial\, \mathrm{adj}_{0,k}}{\partial \boldsymbol{q}_h} = \mathrm{adj}_{\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h}\dot{\boldsymbol{q}}} , \tag{20}$$

where in $\mathrm{adj}_{\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h}\dot{\boldsymbol{q}}}$ this time $\boldsymbol{\nu}_k \in \mathbb{R}^6$ has been replaced by $\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h}\dot{\boldsymbol{q}} \in \mathbb{R}^6$.

Proof: Let us start considering

$$\frac{\partial\, \mathrm{Ad}_{0,k}^{-1}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} = \dot{\mathrm{Ad}}_{0,k}^{-1} = -\,\mathrm{adj}_{0,k}\, \mathrm{Ad}_{0,k}^{-1} , \tag{21}$$

where, using Proposition 3, we can rewrite $\mathrm{adj}_{0,k}$ as

$$\mathrm{adj}_{0,k} = \boldsymbol{W}\ \mathrm{blkdiag}\left(\boldsymbol{J}_k^h \dot{\boldsymbol{q}}_h\right) = \sum_h \boldsymbol{W}\ \mathrm{blkdiag}\left(\boldsymbol{J}_k^h\right) \dot{\boldsymbol{q}}_h . \tag{22}$$

Substituting (22) in (21), we obtain an equality that, being true $\forall \dot{\boldsymbol{q}} \in \mathbb{R}^n$, leads us to

$$\frac{\partial\, \mathrm{Ad}_{0,k}^{-1}}{\partial \boldsymbol{q}_h} = -\boldsymbol{W}\ \mathrm{blkdiag}\left(\boldsymbol{J}_k^h\right) \mathrm{Ad}_{0,k}^{-1} , \tag{23}$$

which is equivalent to (18). On the other hand the differentiation of (22) with respect to $\dot{\boldsymbol{q}}_h$ results in (19), while the one with respect to $\boldsymbol{q}_h$ results in (20). ∎

Proposition 5 (State differentiation): Given the expression of the dynamic matrices in (6) and the factorisation presented in Proposition 4, then the derivatives of (6) with respect to the state are

$$\frac{\partial \boldsymbol{M}}{\partial \boldsymbol{q}_h} = \sum_k \frac{\partial \boldsymbol{J}_k^T}{\partial \boldsymbol{q}_h}\boldsymbol{\Lambda}_k\boldsymbol{J}_k + \boldsymbol{J}_k^T\boldsymbol{\Lambda}_k\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h} , \tag{24a}$$

$$\begin{aligned}\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{q}_h} = &\sum_k \frac{\partial \boldsymbol{J}_k^T}{\partial \boldsymbol{q}_h}\left(\boldsymbol{\Psi}_k\boldsymbol{J}_k + \boldsymbol{\Lambda}_k\dot{\boldsymbol{J}}_k\right) + \boldsymbol{J}_k^T\boldsymbol{\Lambda}_k\frac{\partial \dot{\boldsymbol{J}}_k}{\partial \boldsymbol{q}_h} \\ &+ \boldsymbol{J}_k^T\left[\left(\boldsymbol{\Lambda}_k\,\mathrm{adj}_{\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h}\dot{\boldsymbol{q}}} - \mathrm{adj}_{\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h}\dot{\boldsymbol{q}}}^T\,\boldsymbol{\Lambda}_k\right)\boldsymbol{J}_k + \boldsymbol{\Psi}_k\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h}\right] ,\end{aligned} \tag{24b}$$

$$\frac{\partial \boldsymbol{C}}{\partial \dot{\boldsymbol{q}}_h} = \sum_k \boldsymbol{J}_k^T\left[\left(\boldsymbol{\Lambda}_k\,\mathrm{adj}_{\boldsymbol{J}_k^h} - \mathrm{adj}_{\boldsymbol{J}_k^h}^T\,\boldsymbol{\Lambda}_k\right)\boldsymbol{J}_k + \boldsymbol{\Lambda}_k\frac{\partial \dot{\boldsymbol{J}}_k}{\partial \dot{\boldsymbol{q}}_h}\right] , \tag{24c}$$

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}_h} = \sum_k \left(\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h}^T\boldsymbol{\Lambda}_k - \boldsymbol{J}_k^T\boldsymbol{\Lambda}_k\,\mathrm{adj}_{\boldsymbol{J}_k^h}\right)\mathrm{Ad}_{0,k}^{-1}\boldsymbol{\gamma}_0 , \tag{24d}$$

[2] In the appendix we give the expression of $\boldsymbol{W}$.

where $k = 1, ..., N$ and $\boldsymbol{C}$ is the only matrix which depends on both $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$.

V. Time differentiation

Obviously the derivatives of (6) with respect to time could be computed using the chain rule and (24). Nevertheless, if the derivatives with respect to the state are not required for the considered application, it is more efficient to directly differentiate (6) with respect to time. This is quite clear considering that the derivative of a matrix with respect to a vector is an order three tensor, while the derivative with respect to a scalar is also a matrix.

Proposition 6 (Time differentiation): Given the expression of the dynamic matrices in (6), then the derivatives of (6) with respect to the state are

$$\dot{\boldsymbol{M}} = \sum_k \dot{\boldsymbol{J}}_k^T \boldsymbol{\Lambda}_k \boldsymbol{J}_k + \boldsymbol{J}_k^T \boldsymbol{\Lambda}_k \dot{\boldsymbol{J}}_k \ , \tag{25a}$$

$$\begin{aligned} \dot{\boldsymbol{C}} = &\sum_k \dot{\boldsymbol{J}}_k^T \left(\boldsymbol{\Psi}_k \boldsymbol{J}_k + \boldsymbol{\Lambda}_k \dot{\boldsymbol{J}}_k \right) + \boldsymbol{J}_k^T \boldsymbol{\Lambda}_k \ddot{\boldsymbol{J}}_k \\ &+ \boldsymbol{J}_k^T \left[\left(\boldsymbol{\Lambda}_k \dot{\mathrm{adj}}_{0,k} - \dot{\mathrm{adj}}_{0,k}^T \boldsymbol{\Lambda}_k \right) \boldsymbol{J}_k + \boldsymbol{\Psi}_k \dot{\boldsymbol{J}}_k \right] \ , \end{aligned} \tag{25b}$$

$$\dot{\boldsymbol{g}} = \sum_k \left(\dot{\boldsymbol{J}}_k^T \boldsymbol{\Lambda}_k - \boldsymbol{J}_k^T \boldsymbol{\Lambda}_k \, \mathrm{adj}_{0,k} \right) \mathrm{Ad}_{0,k}^{-1} \boldsymbol{\gamma}_0 \ , \tag{25c}$$

where $k = 1, ..., N$.

From (25) it is clear that the only additional necessary matrices are: $\ddot{\boldsymbol{J}}_k$ and $\dot{\mathrm{adj}}_{0,k}$. The latter can be easily computed using $\dot{\boldsymbol{\nu}}_k = \boldsymbol{J}_k \ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_k \dot{\boldsymbol{q}}_k$, instead of $\boldsymbol{\nu}_k$, as can be recognised taking the derivative of each entry of $\mathrm{adj}_{0,k}$. The computation of $\ddot{\boldsymbol{J}}_k$ is treated in Sec. VI.

Remark 5: Notice that once again the passivity property is satisfied. In fact, since each $\boldsymbol{J}_k^T \left(\boldsymbol{\Lambda}_k \, \mathrm{adj}_{0,k} - \mathrm{adj}_{0,k}^T \boldsymbol{\Lambda}_k \right) \boldsymbol{J}_k$ is a skew symmetric matrix, then

$$\boldsymbol{C} + \boldsymbol{C}^T = \sum_k \dot{\boldsymbol{J}}_k^T \boldsymbol{\Lambda}_k \boldsymbol{J}_k + \boldsymbol{J}_k^T \boldsymbol{\Lambda}_k \dot{\boldsymbol{J}}_k = \dot{\boldsymbol{M}} \ ,$$

$k = 1, ..., N$.

Regarding high order derivatives, we can conclude saying that each additional differentiation of the dynamic matrices with respect to time, requires the propagation of an additional derivative of $\dot{\boldsymbol{J}}_k$ according to the algorithm in Sec. VI and the knowledge of an additional time derivative of $\dot{\boldsymbol{q}} \in \mathbb{R}^n$.

VI. Iterative Algorithm

Here we will show how to compute $\boldsymbol{J}_k$ and its derivatives, which, as seen in the previous sections, are fundamental for the computation of all the matrices and their derivatives.

As already mentioned, the computation of the Jacobian will be carried out with an iterative procedure which will propagate the matrices from the root to the leaves of the tree structured robot. This is completely equivalent to the outward recursion of the Newton-Euler algorithm and, like the latter, it is based on the coordinate transformation for the velocities, which can be formulated as

$$\boldsymbol{\nu}_k = \mathrm{Ad}_{p,k}^{-1} \boldsymbol{\nu}_p + \boldsymbol{\xi}_k \ , \tag{26}$$

where both p and k are used to indicate a frame attached to a rigid body, while $\boldsymbol{\xi}_k$ gives the relative velocity between these two bodies. In the following we use k for the current link and p for its parent[3]. Writing (26) in terms of Jacobians, we obtain an equality that, being true $\forall \dot{\boldsymbol{q}} \in \mathbb{R}^n$, leads us to

$$\boldsymbol{J}_k = \mathrm{Ad}_{p,k}^{-1} \boldsymbol{J}_p + \boldsymbol{\Xi}_k \ , \tag{27}$$

where $\boldsymbol{\Xi}_k$ is a constant matrix completely determined by the type of interconnection between the bodies[4], such that $\boldsymbol{\xi}_k = \boldsymbol{\Xi}_k \dot{\boldsymbol{q}}$. This concept, as discussed in [30], allows the joints to be completely general, with any number of degrees of freedom up to and including six[5]. Differentiating (27) with respect to time, we have

$$\dot{\boldsymbol{J}}_k = \mathrm{Ad}_{p,k}^{-1} \dot{\boldsymbol{J}}_p - \mathrm{adj}_{p,k} \, \mathrm{Ad}_{p,k}^{-1} \boldsymbol{J}_p \ , \tag{28}$$

$$\begin{aligned} \ddot{\boldsymbol{J}}_k = &\mathrm{Ad}_{p,k}^{-1} \ddot{\boldsymbol{J}}_p - \mathrm{adj}_{p,k} \, \mathrm{Ad}_{p,k}^{-1} \dot{\boldsymbol{J}}_p \\ &- \dot{\mathrm{adj}}_{p,k} \, \mathrm{Ad}_{p,k}^{-1} \boldsymbol{J}_p - \mathrm{adj}_{p,k} \dot{\boldsymbol{J}}_k \ , \end{aligned} \tag{29}$$

which is used for the computation of (6), (13), (24), (25). Moreover, because of the property

$$\frac{\partial \dot{\boldsymbol{J}}_k}{\partial \dot{\boldsymbol{q}}} = \frac{\partial}{\partial \dot{\boldsymbol{q}}} \left(\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} \right) = \frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}} \ , \tag{30}$$

the needed derivatives with respect to the state are

$$\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h} = \mathrm{Ad}_{p,k}^{-1} \frac{\partial \boldsymbol{J}_p}{\partial \boldsymbol{q}_h} - \mathrm{adj}_{\boldsymbol{\Xi}_k^h} \, \mathrm{Ad}_{p,k}^{-1} \boldsymbol{J}_p \ , \tag{31}$$

$$\frac{\partial \dot{\boldsymbol{J}}_k}{\partial \boldsymbol{q}_h} = \mathrm{Ad}_{p,k}^{-1} \frac{\partial \dot{\boldsymbol{J}}_p}{\partial \boldsymbol{q}_h} - \mathrm{adj}_{\boldsymbol{\Xi}_k^h} \, \mathrm{Ad}_{p,k}^{-1} \dot{\boldsymbol{J}}_p - \mathrm{adj}_{p,k} \frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}_h} \ , \tag{32}$$

which is used for the computation of (24).

The formulas derived so far can be collected in:

Algorithm Iterative computation of the kinematics

Given $\boldsymbol{q}$ and its derivatives, for each body

- Compute the homogeneous transformation matrix $\boldsymbol{T}_{p,k}$ (for example through the product of exponentials formula)
- Compute $\mathrm{Ad}_{p,k}^{-1}$ (through $\boldsymbol{T}_{p,k}$)
- Compute $\mathrm{adj}_{p,k}$ and $\mathrm{adj}_{\boldsymbol{\Xi}_k^h}$ (through $\boldsymbol{\xi}_k$ and $\boldsymbol{\Xi}_k$)
- Propagate the Jacobian and its derivatives (eq. (27) - (32))
- Propagate $\mathrm{Ad}_{0,k}^{-1}$ ($\mathrm{Ad}_{0,k}^{-1} = \mathrm{Ad}_{p,k}^{-1} \mathrm{Ad}_{0,p}^{-1}$)
- Compute $\mathrm{adj}_{0,k}$ (through $\boldsymbol{\nu}_k = \boldsymbol{J}_k \dot{\boldsymbol{q}}$)

which provides the informations for

1) dynamic matrices (6),
2) CoM Jacobian (7),
3) Slotine-Li regressor (16),
4) Parameters derivatives (13),
5) State derivatives (24),
6) Time derivatives (25).

Notice that, by definition, the parent of a frame attached to a root body of the structure is the spatial frame. This

[3] If the bodies are numbered according to a so called regular numbering scheme [8], it is guaranteed that during the iteration the parent's quantities are computed before those of its children.

[4] Notice how in (27) we need only "local" informations, meaning that $\mathrm{Ad}_{p,k}^{-1}$ and $\boldsymbol{\Xi}_k$ are given by the connection between the link and its parent.

[5] This is particularly important to model floating base robots.

means that for these bodies $\boldsymbol{J}_p$ and $\dot{\boldsymbol{J}}_p$ are matrices of zeros, while $\mathrm{Ad}^{-1}_{p,0} = \mathrm{Ad}^{-1}_{0,0}$ is the identity. These are actually the informations which trigger the propagation.

Before concluding the discussion of the necessary computations in the algorithm, two points are worth to be clarified.

Remark 6: The last two steps are only necessary for the computation of $\boldsymbol{g}$ and $\boldsymbol{C}$. In particular since $\mathrm{Ad}^{-1}_{0,k}$ is only used to compute $\boldsymbol{\gamma}_k$, whose angular part is always zero, the only information needed from $\mathrm{Ad}^{-1}_{0,k}$ is $\boldsymbol{R}^{-1}_{0,k}$ (see the expression of the adjoint in the appendix). This suggest to replace the propagation of $\mathrm{Ad}^{-1}_{0,k}$ with the more efficient $\boldsymbol{R}^T_{0,k} = \boldsymbol{R}^T_{p,k}\boldsymbol{R}^T_{0,p}$.

Remark 7: The matrix $\boldsymbol{\Xi}_k$ is clearly constant for revolute and prismatic joints. For example, in case of a serial chain robot with revolute joints, only the $k-th$ column of $\boldsymbol{\Xi}_k$ is non zero. Its linear part is given by $-\boldsymbol{e}_k \times \boldsymbol{c}_k$, while the angular part is given by $\boldsymbol{e}_k$, where $\boldsymbol{c}_k$ is a point on the axis of the screw whose direction is given by the unit vector $\boldsymbol{e}_k$ while $\boldsymbol{q}_k$ gives the amount of rotation around the axis (the screw coordinate of the rotational motion [28]). Unfortunately using $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ as input for the algorithm, $\boldsymbol{\Xi}_k$ will not be constant for other types of joints (e.g. universal joints). Nevertheless, choosing a different velocity input $\boldsymbol{w}$ it is still possible to use a constant matrix $\boldsymbol{\Xi}_k$. A clear example is the case of a free joint[6], used when modelling free floating base systems. In this case if $\boldsymbol{w}$ itself contains $\boldsymbol{\xi}_k$, then $\boldsymbol{\Xi}_k$ is a selection matrix, such that $\boldsymbol{\xi}_k = \boldsymbol{\Xi}_k \boldsymbol{w}$. Because of this choice $\dot{\boldsymbol{q}} = \boldsymbol{\Gamma}(\boldsymbol{q})\boldsymbol{w}$, where $\boldsymbol{\Gamma}(\boldsymbol{q})$ is responsible for the mapping from angular velocity to rate of change of Euler angles, which can introduce representation singularity [1]. Moreover $\boldsymbol{\Gamma}(\boldsymbol{q})$ will appear when applying the chain rule in (21) and (30). For example (30) becomes

$$\frac{\partial \dot{\boldsymbol{J}}_k}{\partial \boldsymbol{w}} = \frac{\partial}{\partial \boldsymbol{w}}\left(\frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}}\boldsymbol{\Gamma}(\boldsymbol{q})\boldsymbol{w}\right) = \frac{\partial \boldsymbol{J}_k}{\partial \boldsymbol{q}}\boldsymbol{\Gamma}(\boldsymbol{q}) \ . \tag{33}$$

VII. Comparisons and practical applications

The implementation aspects, the evaluation of the complexity of the algorithm sketched in Sec. VI and the comparisons with different approaches are beyond the scope of this paper. The same argumentations that motivated Ploen in [29] to provide the single matrices, rather than the solution of multiple particular inverse dynamic problems, apply also in our case. Even the regressor could be computed solving multiple inverse dynamic problems, but an explicit analytic form enables inspection of the physical properties of the manipulator, such as the determination of the minimum set of inertial parameters. The reader interested in alternatives to the algorithm presented in Sec. VI and to the comparisons and performances is then referred to the works of Orin and Schrader [31], Park [32] and Jain [33], where these topics are extensively treated.

As practical applications, we have implemented the computation of the matrices in (6) for the control algorithms of the robot shown in Fig. 2. TORO is a torque controlled humanoid robot developed at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). It has 25

[6]A free joint is a joint which allows the motion along and around all the three directions.

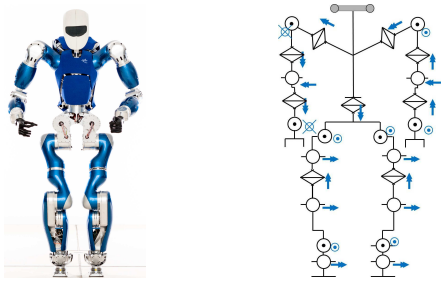

Fig. 2. TORO (TOrque controlled RObot) and its kinematics.

revolute joints, each equipped with position and torque sensors. Using a computer with an Intel ® Xeon ® Processor W3530 (2.80 *GHz*, 4 physical cores), 100 runs of the algorithm take an average of 0.244 *ms* each. For the derivatives, we compared the time necessary to compute the matrices in (6) at two different instants and the one necessary for an extended version of the code, where even the matrices in (25) are provided as output. The average ratio between the two approaches is 0.71 in favour of the latter. The improvement would be even significantly higher when the differentiation with respect to the state is required. In this case each matrix must be computed twice for each state variable and scales then with the number of joints.

As future application we also plan to use (16) and (24) for identification, optimisation and adaptive algorithms.

VIII. Conclusion

In this paper we have considered the dynamic equations of a system of interconnected rigid bodies. The focus was essentially on the computation of the matrices of the dynamic equations and their derivatives, even though more than this can be easily computed once a clear formulation of the dynamic is available (for example the Slotine-Li regressor and the CoM Jacobian). We started with a formulation of Newton's third law in terms of twists and wrenches to obtain closed form expressions for the matrices of a branched connection of rigid bodies. This results in the possibility of computing the derivatives in a straightforward way. Depending on the application (control of elastic joint robots, optimisation, identification, etc.), different types of differentiation are needed (time, state or dynamic parameters). A Newton-Euler like outward recursion for the propagation of the Jacobian matrices, together with the formulas and remarks in the paper, allows the reader to easily implement a library where all the necessary informations for simulation and control can be computed. Although the complexity was not analysed, the computation is expected to be quite efficient since it is based on just the first step of the Newton-Euler algorithm.

Appendix

Here we give the expression of the matrices used throughout the paper, in case $\boldsymbol{\nu}_k = \begin{bmatrix}\boldsymbol{v}_k^T & \boldsymbol{\omega}_k^T\end{bmatrix}^T$.

Given the rotation matrix $\boldsymbol{R}_{p,k} \in SO(3)$ and the position vector $\boldsymbol{p}_{p,k} \in \mathbb{R}^3$ as in Fig. 1, the corresponding homogeneous transformation matrix, adjoint and Lie bracket matrices are

$$\boldsymbol{T}_{p,k} = \begin{bmatrix} \boldsymbol{R}_{p,k} & \boldsymbol{p}_{p,k} \\ \boldsymbol{0}^T & 1 \end{bmatrix},$$

$$\mathrm{Ad}_{p,k} = \begin{bmatrix} \boldsymbol{R}_{p,k} & \tilde{\boldsymbol{p}}_{p,k}\boldsymbol{R}_{p,k} \\ \boldsymbol{O}_{3\times3} & \boldsymbol{R}_{p,k} \end{bmatrix}, \ \mathrm{adj}_{p,k} = \begin{bmatrix} \tilde{\boldsymbol{\omega}}_k & \tilde{\boldsymbol{v}}_k \\ \boldsymbol{O}_{3\times3} & \tilde{\boldsymbol{\omega}}_k \end{bmatrix}.$$

According to this convention

$$\boldsymbol{\Lambda}_k = \begin{bmatrix} m_k\boldsymbol{E}_3 & -m_k\tilde{\boldsymbol{r}}_k \\ m_k\tilde{\boldsymbol{r}}_k & \boldsymbol{I}_k - m_k\tilde{\boldsymbol{r}}_k^2 \end{bmatrix},$$

$$\boldsymbol{A}(\boldsymbol{\nu}_k) = \begin{bmatrix} \boldsymbol{\nu}_k & \tilde{\boldsymbol{\omega}}_k & \boldsymbol{O}_{3\times6} \\ \boldsymbol{0} & -\tilde{\boldsymbol{v}}_k & \tilde{\tilde{\boldsymbol{\omega}}}_k \end{bmatrix}, \ \boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_\omega & \boldsymbol{W}_v \\ \boldsymbol{O}_{3\times18} & \boldsymbol{W}_\omega \end{bmatrix},$$

$$\boldsymbol{W}_\omega = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{e}_z & -\boldsymbol{e}_y \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ -\boldsymbol{e}_z & \boldsymbol{0} & \boldsymbol{e}_x \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{e}_y & -\boldsymbol{e}_x & \boldsymbol{0} \end{bmatrix}^T \quad \boldsymbol{W}_v = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{e}_z & -\boldsymbol{e}_y \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ -\boldsymbol{e}_z & \boldsymbol{0} & \boldsymbol{e}_x \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{e}_y & -\boldsymbol{e}_x & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}^T,$$

where $\boldsymbol{I}_k$ is the inertia tensor with respect to a frame oriented as the body frame and with the origin in the CoM of the rigid body, $\boldsymbol{0} \in \mathbb{R}^3$ and $\boldsymbol{e}_x$, $\boldsymbol{e}_y$, $\boldsymbol{e}_z$ are such that $\begin{bmatrix} \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z \end{bmatrix} = \boldsymbol{E}_3$, being $\boldsymbol{E}_3 \in \mathbb{R}^{3\times3}$ the identity matrix.

Acknowledgment

The first author wants to thank Vincenzo Lippiello and Fabio Ruggiero for the discussions during the preparation of his master thesis, in which he started focusing on the dynamic equations of the robot.

This research is partly supported by the Initiative and Networking Fund of the Helmholtz Association through a Helmholtz Young Investigators Group (Grant no. VH-NG-808).

References

[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer Publishing Company, Incorporated, 2008.

[2] J. J. Uicker, "On the dynamic analysis of spatial linkages using 4x4 matrices," Ph.D. dissertation, Northwestern University, 1965.

[3] Y. Stepanenko and M. Vukobratovic, "Dynamics of articulated open-chain active mechanisms," *Mathematical Biosciences*, vol. 28, pp. 137–170, 1976.

[4] D. E. Orin, R. McGhee, M. Vukobratovi, and G. Hartoch, "Kinematic and kinetic analysis of open-chain linkage utilizing newton-euler methods," *Mathematical Biosciences*, vol. 43, pp. 107–130, 1979.

[5] J. M. Hollerbach, "A recursive Lagrangian formulation of maniputator dynamics and a comparative study of dynamics formulation complexity," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 10, pp. 730–736, 1980.

[6] W. M. Silver, "On the equivalence of Lagrangian and Newton-Euler dynamics for manipulators," *Int. Journal of Robotics Research*, vol. 1, pp. 60–70, 1982.

[7] A. F. Vereshchagin, "Computer simulation of the dynamics of complicated mechanisms of robot manipulators," *Engineering Cybernetics*, vol. 6, pp. 65–70, 1974.

[8] R. Featherstone, *Robot Dynamics Algorithms*. Boston: Kluwer Academic Publishers, 1987.

[9] G. Rodriguez, "Kalman filtering, smoothing and recursive robot arm forward and inverse dynamics," *IEEE Journal Robotics and Automation*, vol. 3, pp. 624–639, 1987.

[10] F. C. Park, J. E. Bobrow, and S. R. Ploen, "A Lie group formulation of robot dynamics," *Int. Journal of Robotics Research*, vol. 14, pp. 609–618, 1995.

[11] C. G. Atkeson, C. H. An, and J. M. Hollerbach, "Estimation of inertial parameters of manipulator loads and links," *Int. Journal of Robotics Research*, vol. 5, pp. 101–119, 1986.

[12] P. K. Khosla and T. Kanade, "Parameter identification of robot dynamics," in *IEEE Conf. on Decision and Control*, 1985, pp. 1754–1760.

[13] H. Kawasaki and K. Nishimura, "Terminal-link parameter estimation of robotic manipulators," *IEEE Journal Robotics and Automation*, vol. 4, pp. 485–490, Oct. 1988.

[14] M. Gautier and W. Khalil, "Direct calculation of minimum set of inertial parameters of serial robots," *IEEE Trans. Robotics and Automation*, vol. 6, p. 368373, June 1990.

[15] J. J. Craig, P. Hsu, and S. S. Sastry, "Adaptive control of mechanical manipulators," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, San Francisco, USA, Apr. 1986, pp. 190–195.

[16] P. Hsu, M. Bodson, S. S. Sastry, and B. E. Paden, "Adaptive identification and control for manipulators without using joint accelerations," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Raleigh, USA, Mar. 1987, pp. 1210–1215.

[17] R. H. Middleton and G. C. Goodwin, "Adaptive computed torque control for rigid link manipulators," in *IEEE Conf. on Decision and Control*, Athens, Greece, Dec. 1986, pp. 68–73.

[18] J.-J. E. Slotine and W. Li, "On the adaptive control of robot manipulators," *Int. Journal of Robotics Research*, vol. 6, pp. 49–59, 1987.

[19] J. Yuan and B. Yuan, "Recursive computation of the slotine-li regressor," in *American Control Conference (ACC)*, Washington D.C., USA, June 1995, pp. 2327–2331.

[20] M. W. Spong, "Modeling and control of elastic joint robots," *Journal of Dynamic Systems, Measurement and Control*, vol. 109, pp. 310–319, 1987.

[21] A. De Luca and F. Flacco, "A PD-type regulator with exact gravity cancellation for robots with flexible joints," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 2011, pp. 317–323.

[22] C. Ott, *Cartesian Impedance Control of Redundant and Flexible-Joint Robots*, ser. Springer Tracts in Advanced Robotics. Berlin: Springer-Verlag, 2008.

[23] M. Reyhanoglu, A. van der Schaft, H. McClamroch, and I. V. Kolmanovsky, "Dynamics and control of a class of underactuated mechanical systems," *IEEE Trans. on Automatic Control*, vol. 44, no. 9, pp. 1663–1671, 1999.

[24] A. Müller, "Partial derivatives of the inverse mass matrix of multibody systems via its factorization," *IEEE Trans. on Robotics*, vol. 23, no. 1, pp. 164–168, 2007.

[25] W. Suleiman, E. Yoshida, J.-P. Laumond, and A. Monin, "Optimizing humanoid motions using recursive dynamics and Lie groups," in *International Conference on Information and Communication Technologies: from Theory to Applications (ICTTA)*, Damascus, Syria, Apr. 2008, pp. 1–6.

[26] G. A. Sohl and J. E. Bobrow, "A recursive multibody dynamics and sensitivity algorithm for branched kinematic chains," *the ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 123, pp. 391–399, 2001.

[27] A. Gelb, *Applied optimal estimation*. Cambridge, MA: MIT Press, 1974.

[28] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[29] S. R. Ploen, "Geometric algorithms for the dynamics and control of multibody systems," Ph.D. dissertation, University of California Irvine, 1997.

[30] R. E. Roberson and R. Schwertassek, *Dynamics of Multibody Systems*. Berlin: Springer-Verlag, 1987.

[31] D. E. Orin and W. W. Schrader, "Efficient computation of the jacobian for robot manipulators," *Int. Journal of Robotics Research*, vol. 3, no. 4, pp. 66–75, Dec. 1984.

[32] F. C. Park, "Computational aspects of the product-of-exponentials formula for robot kinematics," *IEEE Trans. on Automatic Control*, vol. 39, pp. 643–647, Mar. 1994.

[33] A. Jain, "Unified formulation of dynamics for serial rigid multibody systems," *Journal of Guidance, Control and Dynamics*, vol. 14, pp. 531–542, 1991.