Dynamic Modeling of Robots Using Newton-Euler Formulation

Wisama Khalil

Ecole Centrale de Nantes, IRCCyN UMR CNRS 6597, 1 Rue de la Noë, 44321, Nantes, France
Wisama.khalil@irccyn.ec-nantes.fr

Abstract. This paper presents the use of recursive Newton-Euler formulation to model different types of robots. The main advantages of this technique are the facility of implementation and obtaining models with reduced number of operations. The following structures are treated: rigid tree structure robots, closed loop robots, parallel robots, robots with lumped elasticity, robots with moving base and wheeled robots.

Keywords: Dynamic modeling, Newton-Euler, Recursive calculation, Tree structure, Parallel robots, Flexible joints, Mobile robots.

1 Introduction

The dynamic modeling of robots is an important topic for the design, simulation, and control of robots. Different techniques have been proposed and used by the robotics community. In this paper we show that the use of Newton-Euler recursive technique is easy to develop and implement. The proposed algorithm can be extended to many types of structures. In section 2 we recall the method used to describe the kinematics of the structure, and then in section 3 we present the inverse and direct dynamic models of tree structure rigid robots. The following sections present the generalization to the other systems.

2 Description of the Robots

The structures of robots will be described using Khalil and Kleinfinger notations [1]. This method can take into account tree structures and closed loop robots. Its use facilitates the calculation of minimum number of inertial parameters [2]…[4], which reduce the number of operations of the dynamic models and are needed in dynamic identification and adaptive control laws.

2.1 Geometric Description of Tree Structure Robots

A tree structure robot is composed of $n+1$ links and n joints. Link 0 is the base and link n is a terminal link. The joints are revolute or pris-matic, rigid or elastic. The links are numbered consecutively from the base, to the terminal links. Joint j connects

J.A. Cetto et al. (Eds.): Informatics in Control, Automation and Robotics, LNEE 89, pp. 3–20.
springerlink.com

link j to link $a(j)$, where a(j) denotes the link antecedent to link j. A frame R_i is attached to each link i such that (Fig. 1):

- z_i is along the axis of joint i;
- x_i is along the common normal between z_i and one of the succeeding joint axes.

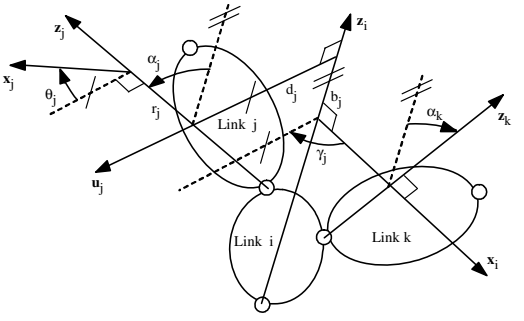

Fig. 1. Geometric parameters for frame j

The transformation matrix ${}^{i}h_{j}$, defining frame R_j with respect to (wrt) frame R_i is obtained as a function of six geometric parameters $(\gamma_j, b_j, \alpha_j, d_j, \theta_j, r_j)$ such that:

$$ {}^{i}h_{j} = Rot(z,\gamma_j)\, Tran(z,b_j)\, Rot(x,\alpha_j)\, Tran(x,d_j)\, Rot(z,\theta_j)\, Tran(z,r_j) $$

After developing, this matrix can be written as follows:

$$ {}^{i}h_{j} = \begin{bmatrix} {}^{i}R_j & {}^{i}P_j \\ 0_{1x3} & 0 \end{bmatrix} = \begin{bmatrix} C\gamma C\theta - S\gamma C\alpha S\theta & -C\gamma S\theta - S\gamma C\alpha C\theta & S\gamma S\alpha & dC\gamma + rS\gamma S\alpha \\ S\gamma C\theta + C\gamma C\alpha S\theta & -S\gamma S\theta + C\gamma C\alpha C\theta & -C\gamma S\alpha & dS\gamma - rC\gamma S\alpha \\ S\alpha S\theta & S\alpha C\theta & C\alpha & rC\alpha + bj \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) $$

Where ${}^{i}R_j$ defines the (3×3) rotation matrix and ${}^{i}P_j$ defines the (3×1) vector defining the position of the origin of frame j with respect to frame i.

If x_i is along the common normal between z_i and z_j, both γ_j and b_j will be zero.

The type of the joint is identified by σ_j where $\sigma_j = 0$ if j is revolute, $\sigma_j = 1$ if j is prismatic, and $\bar{\sigma}_j = 1 - \sigma_j$. The joint variable, denoted as q_j, is θ_j if j is revolute and r_j if j is prismatic.

A serial structure is a special case of a tree structure where $a(j) = j-1,\ \gamma_j = 0,\ b_j = 0$ for all $j = 1,..,n$.

2.2 Description of Closed Loop Structure

The system is composed of L joints and $n+1$ links, where link 0 is the fixed base and $L > n$. The number of independent closed loops is equal to $B = L - n$.

The joints are either active (motorized) or passive. The number of active joints is denoted *N*. The location of all the links can be determined as a function of the active variables. The geometric parameters of the mechanism can be determined as follows:

a) Construct an equivalent tree structure having n joints by virtually cutting each closed chain at one of its passive joints. Define the geometric parameters of the tree structure as given in section 2.1.

b) Number the cut joints $k= n+1,\ldots,L$,

c) For each cut joint define two frames on one of the links connected by this joint.

Assuming a cut joint k, and that the links connected by it are link i and link j, the frames are defined as follows (Fig. 2):

- frame R_k is fixed on link j such that $a(k)=i$, the axis z_k is along the axis of joint k, and x_k is along the common normal between z_k and z_j. The matrix ${}^{i}h_k$ is determined using the parameters $(\gamma_k,\ b_k, \alpha_k, d_k, \theta_k, r_k)$.

- frame R_{k+B} is aligned with R_k, but $a(k+B)= j$. The matrix ${}^{j}h_{k+B}$ is constant.

The joint variables are denoted as:

$$q = \begin{bmatrix} q_{tr} \\ q_c \end{bmatrix}, q_{tr} = \begin{bmatrix} q_a \\ q_p \end{bmatrix} \tag{2}$$

- q_{tr} tree structure joint variables, q_c the cut joint variables,
- q_a, q_p active and passive joint variables of the tree structure.

Since R_k and R_{k+B} are aligned, the geometric constraint equations for each loop, can be written as:

$$ {}^{k+B}h_j \ \ldots \ {}^{i}h_k \ = \ I_4 \tag{3}$$

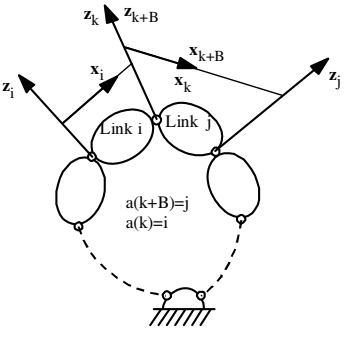

Fig. 2. Frames around a cut joint k

The kinematic constraint equations are given by:

$$\begin{aligned} {}^{0}V_k &= {}^{0}V_{k+B} \\ \mathrm{J}_k \dot{q}_{b1} &= \mathrm{J}_{k+B} \dot{q}_{b2} \end{aligned} \tag{4}$$

Where V_k defines the (6×1) kinematic screw vector of frame k, given by:

$$V_k = \begin{bmatrix} v_k^T & \omega_k^T \end{bmatrix}^T \tag{5}$$

v_k linear velocity of the origin of frame $\mathrm{R_k}$, ω_k angular velocity of frame k;
J_k the kinematic Jacobian matrix of frame k;
$\dot{q}_{b1}, \dot{q}_{b2}$ joint velocities through the two branches of the loop.

3 Dynamic Modeling of Tree Structure Robots

3.1 Introduction

The most common methods used to calculate the dynamic models are the Lagrange equations and the Newton Euler Equations [5],…[7]. The Lagrange model is given as:

$$\Gamma = A(q)\ddot{q} + H(q,\dot{q}) \tag{6}$$

where A is the inertia matrix of the robot and H is the Coriolis, centrifugal and gravity torques.

Calculating Γ in terms of $(\mathrm{q},\dot{\mathrm{q}},\ddot{\mathrm{q}})$ is known as the inverse dynamic problem, and calculating $\ddot{q}$ in terms of $(q,\dot{q},\Gamma)$ is known as the direct dynamic model. The inverse dynamic model is obtained from (6), whereas the direct model is calculated as:

$$\ddot{q} = (A)^{-1}(\Gamma - H) \tag{7}$$

The recursive Newton-Euler algorithms have been shown to be the most efficient technique to model rigid robots [8]…[11]. In [12] a recursive Lagrange algorithm is presented but without achieving better performances than that of Newton-Euler.

The Newton-Euler equations giving the total forces and moments on a link j about the origin of frame j are written as:

$$\mathrm{j}\Phi_{\mathrm{j}} = {}^{j}J_j \, {}^{j}\dot{V}_j + \begin{bmatrix} {}^{j}\omega_j \times \left({}^{j}\omega_j \times {}^{j}MS_j \right) \\ {}^{j}\omega_j \times \left({}^{j}I_{oj} \, {}^{j}\omega_j \right) \end{bmatrix} \tag{8}$$

where

ω_{j} angular velocity of link j; $\dot{V}_j$ contains the linear acceleration of the origin of frame j, denoted $\dot{v}_j$, and the angular acceleration $\dot{\omega}_{\mathrm{j}}$.

Φ_{j} total wrench (forces and moments) on link j at origin of frame R_j;

$J_j\ (6\times6)$ inertia matrix of link j, which is given as:

$$ {}^{j}J_j = \begin{bmatrix} M_j I_3 & -{}^{j}M\hat{S}_j \\ {}^{j}M\hat{S}_j & {}^{j}I_{oj} \end{bmatrix} \tag{9} $$

Where M_j, MS_j and I_{oj} are the standard inertial parameters of link j. They are respectively, the mass, the first moments, and the inertia matrix at the origin.

3.2 Calculation of the Inverse Dynamics Using Recursive NE Algorithm

The algorithm consists of two recursive computations: forward and backward [8]. The forward equations, from 1 to n, compute the velocities, accelerations and the dynamic wrench of links. The backward equations from n to 1, provide the joint torques. The algorithm will be denoted by:

$$ \Gamma = \mathrm{NE}(q, \dot{q}, \ddot{q}, F_e) \tag{10} $$

Where F_e is the external wrench exerted by the links of the robot on the environment. The forward equations for $j = 1,...,n$ are as follows:

$$ {}^{j}\omega_j = {}^{j}R_i\, {}^{i}\omega_i + \bar{\sigma}_j\, {}^{j}z_j\, \dot{q}_j \tag{11} $$

$$ {}^{j}\dot{V}_j = {}^{j}T_i\, {}^{i}\dot{V}_i + \ddot{q}_j\, {}^{j}a_j + \begin{bmatrix} {}^{j}R_i\left[{}^{i}\omega_i \times \left({}^{i}\omega_i \times {}^{i}P_j \right) \right] + 2\sigma_j ({}^{j}\omega_i \times \dot{q}_j\, {}^{j}z_j) \\ \bar{\sigma}_j\, {}^{j}\omega_i \times \dot{q}_j\, {}^{j}z_j \end{bmatrix} \tag{12} $$

$$ {}^{j}\Phi_j = {}^{j}J_j\, {}^{j}\dot{V}_j + \begin{bmatrix} {}^{j}\omega_j \times \left({}^{j}\omega_j \times {}^{j}MS_j \right) \\ {}^{j}\omega_j \times \left({}^{j}I_{oj}\, {}^{j}\omega_j \right) \end{bmatrix} \tag{13} $$

Where

- $i = a(j), {}^{j}z_j = [0\ 0\ 1]^T$, ${}^{j}a_j = [0\ 0\ \sigma_j\ 0\ 0\ \bar{\sigma}_j]^T$, $\hat{w}$ defines the skew matrix of vector product associated to the (3×1) vector w such that $w \times v = \hat{w}\, v$.
- ${}^{j}T_i$ the (6×6) screw transformation matrix:

$$ {}^{j}T_i = \begin{bmatrix} {}^{j}R_i & {}^{j}\hat{P}_i\, {}^{j}R_i \\ 0_{3x3} & {}^{j}R_i \end{bmatrix} \tag{14} $$

These equations are initialized by ${}^{0}\omega_0 = 0, {}^{0}\dot{\omega}_0 = 0$. The gravity effect on all the links is taken into account by putting $\dot{v}_0 = -g$.

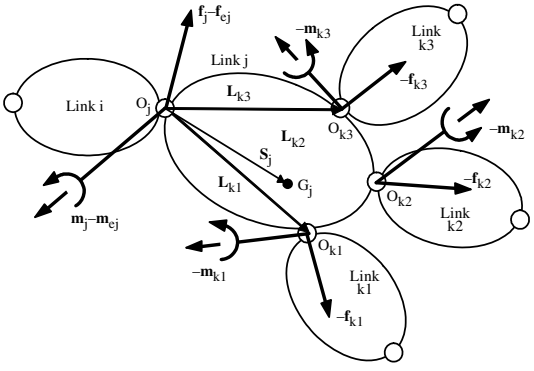

Fig. 3. Forces and moments acting on link j

The backward recursive equations are deduced from the total forces and moments on link j around its origin (Fig. 3). They can be calculated for $j = n,...,1$

$$
\begin{aligned}
{}^{j}F_j &= {}^{j}\Phi_j + \sum_k {}^{k}T_j^{T}\,{}^{k}F_k + {}^{j}F_{ej} \\
\Gamma_j &= {}^{j}F_j^{T}\,{}^{j}a_j + I_{aj}\,\ddot{q}_j + F_{sj}\,\mathrm{sign}(\dot{q}_j) + F_{vj}\,\dot{q}_j
\end{aligned}
\tag{15}
$$

Where $a(k) = j$, F_j is the reaction wrench (forces and moments) of link i on link j, F_{ej} is the wrench (force and moment) exerted by link j on the environment, Ia_j is the gear and rotor inertia of actuator j and F_{sj} , F_{vj} are the coulomb and viscous friction parameters.

The computational cost of this algorithm is linear in the number of degrees of freedom of the robot. To reduce the number of operations of the calculation of this model the base inertial parameters and the customized symbolic calculation can be used [9]…[11].

3.3 Computation of the Direct Dynamic Model

Two methods based on Newton-Euler methods can be used to obtain the dynamic model: the first is proposed by Walker and Orin [13], it is based on calculating the A and H matrices, defined in (6), using Newton-Euler inverse dynamic model and to calculate the joint accelerations by (7); the second method is based on a recursive algorithm [14] [16].

3.3.1 Using the Inverse Dynamic Model to Calculate the Direct Dynamic Model

From (6) and (10) we deduce that $H(q,\dot{q})$ is equal to Γ *if* $\ddot{q} = 0$, and that the ith column of $A(q)$ is equal to $\mathbf{\Gamma}$ if: $\ddot{q} = u_i, \dot{q} = 0, g = 0, F_{ej} = 0$ **,** where u_i is the (nx1) unit vector whose ith element is equal to 1, and the other elements are equal to zero.

3.3.2 Recursive NE Computation of the Direct Dynamic Model

Using (13) and (15) the equilibrium equations of link j can be written as:

$$ {}^{\mathrm{j}}J_{\mathrm{j}}\,{}^{\mathrm{j}}\dot{V}_{\mathrm{j}} = {}^{\mathrm{j}}F_{\mathrm{j}} + {}^{\mathrm{j}}\beta_{\mathrm{j}} - \sum_{k} {}^{\mathrm{k}}T_{\mathrm{j}}^{\mathrm{T}}\,{}^{\mathrm{k}}F_{\mathrm{k}} \tag{16} $$

where k denotes the links articulated on link j such that $a(k)=j$, and

$$ {}^{j}\beta_{j} = -{}^{j}F_{ej} - \begin{bmatrix} {}^{j}\omega_{j}\times\left({}^{j}\omega_{j}\times {}^{j}MS_{j}\right) \\ {}^{j}\omega_{j}\times\left({}^{j}Io_{j}\,{}^{j}\omega_{j}\right) \end{bmatrix} \tag{17} $$

The joint accelerations are obtained as a result of three recursive computations:

i) first forward computations for j = 1, …, n: in this step, we compute the screw transformation matrices ${}^{j}T_{i}$, the link angular velocities ${}^{j}\omega_{j}$ and ${}^{j}\gamma_{j}, {}^{j}\beta_{j}$ vectors, which appear in the link accelerations equation (12), and the link equilibrium equation (17);

$$ {}^{j}\gamma_{j} = \begin{bmatrix} {}^{j}R_{i}\left[{}^{i}\omega_{i}\times\left({}^{i}\omega_{i}\times {}^{i}P_{j}\right)\right] + 2\sigma_{j}({}^{j}\omega_{i}\times\dot{q}_{j}\,{}^{j}z_{j}) \\ \bar{\sigma}_{j}\,{}^{j}\omega_{i}\times\dot{q}_{j}\,{}^{j}z_{j} \end{bmatrix} \tag{18} $$

ii) backward recursive computation [6]: in this step we calculate the elements ${}^{j}H_{j}$, ${}^{j}J_{j}$, ${}^{j}\beta_{j}$, ${}^{j}K_{j}$, ${}^{j}\alpha_{j}$ which express $\ddot{q}_{j}$ and ${}^{i}F_{j}$ in terms of ${}^{i}\dot{V}_{i}$ in the third recursive equations. Thus for j = n, ... ,1 compute:

$$ \begin{aligned} H_{j} &= ({}^{j}a_{j}\,{}^{j}J_{j}^{*}\,{}^{j}a_{j} + Ia_{j}) \\ {}^{j}K_{j} &= {}^{j}J_{j}^{*} - {}^{j}J_{j}^{*}\,{}^{j}a_{j}\,H_{j}^{-1}\,{}^{j}a_{j}^{T}\,{}^{j}J_{j}^{*} \\ {}^{j}\alpha_{j} &= {}^{j}K_{j}\,{}^{j}\gamma_{j} + {}^{j}J_{j}^{*}\,{}^{j}a_{j}\,H_{j}^{-1}(\tau_{j} + {}^{j}a_{j}^{T}\,{}^{j}\beta_{j}^{*}) - {}^{j}\beta_{j}^{*} \end{aligned} \tag{19} $$

If $a(j)\neq 0$ calculate also:

$$ \begin{aligned} {}^{i}\beta_{i}^{*} &= {}^{i}\beta_{i}^{*} - {}^{j}T_{i}^{T}\,{}^{j}\alpha_{j} \\ {}^{i}J_{i}^{*} &= {}^{i}J_{i}^{*} + {}^{j}T_{i}^{T}\,{}^{j}K_{j}\,{}^{j}T_{i} \end{aligned} \tag{20} $$

These equations are initialized by: ${}^{j}J_{j}^{*} = {}^{j}J_{j}$, ${}^{j}\beta_{j}^{*} = {}^{j}\beta_{j}$ for $j=1,\ldots,n$

iii) third recursive equations. Since $({}^{0}\dot{v}_{0} = -g, {}^{0}\dot{\omega}_{0} = 0)$, the third recursive equations give ${}^{j}\dot{V}_{j}$ and ${}^{j}F_{j}$ (if needed) for j = 1… n as follows:

$$
\begin{aligned}
&\ddot{q}_j = H_j^{-1}[-{}^j a_j \, {}^j J_j^* ({}^j T_i \, {}^i \dot{V}_i + {}^j \gamma_j) + \tau_j + {}^j a_j^T \, {}^j \beta_j^*] \\
&{}^j F_j = {}^j K_j \, {}^j T_i \, {}^i \dot{V}_i + {}^j \alpha_j \\
&{}^j \dot{V}_j = {}^j T_i \, {}^i \dot{V}_i + {}^j a_j \, \ddot{q}_j + {}^j \gamma_j \\
&\text{where } \tau_j = \Gamma_j - F_{sj} \, sign(\dot{q}_j) - F_{vj} \, \dot{q}_j
\end{aligned}
\tag{21}
$$

4 Inverse Dynamic Modeling of Closed Loop Robots

The computation of the inverse dynamic model of closed loop robots can be obtained by first calculating the inverse dynamic model of the equivalent tree structure robot, in which the joint variables satisfy the constraints of the loop. Then the closed loop torques of the active joints Γ_c are obtained by solving the following equation:

$$
\begin{bmatrix} \Gamma_c \\ 0 \end{bmatrix} = \Gamma_{\mathrm{tr}}(q_{tr}, \dot{q}_{tr}, \ddot{q}_{tr}) + \mathrm{W}^{\mathrm{T}} \lambda \tag{22}
$$

Where λ is the vector of Lagrange multipliers and $W \dot{q}_{tr} = 0$ is the kinematics constraint equations between the velocities of the passive and active joints of the tree structure. The matrix W can be obtained from (4).

After developing we obtain:

$$
\Gamma_c = G^{\mathrm{T}} \Gamma_{\mathrm{tr}}(q_{tr}, \dot{q}_{tr}, \ddot{q}_{tr}) = \Gamma_a + \frac{\partial \dot{q}_p}{\partial \dot{q}_a} \Gamma_p \tag{23}
$$

where:

$$
G = \frac{\partial q_{tr}}{\partial q_a} = \frac{\partial \dot{q}_{tr}}{\partial \dot{q}_a} \tag{24}
$$

Γ_a and Γ_p are the actuated and passive joint torques of the tree structure.

From equation (23) we see that the active joints of the closed loop structure is calculated by projecting the tree structure torques Γ_{tr} on the motorized joints using the transpose of the Jacobian matrix of the tree structure variables (or velocities) in terms of the active joint variables (or velocities) [6],[17].

There is no recursive method to obtain the direct dynamic model of closed loop robots. It can be computed using the inverse dynamic model by a procedure similar to that given in section (3.3.1) in order to obtain the matrices A_c and H_c such that:

$$
\Gamma_c = A_c(q_{tr}) \ddot{q}_a + H_c(q_{tr}, \dot{q}_{tr}) \tag{25}
$$

5 Inverse Dynamic Modeling of Parallel Robots

A parallel robot is a complex multi-body system having several closed loops. It is composed of a moving platform connected to a fixed base by m parallel legs. The

dynamic model can be obtained as described in the previous section, but in this section we present a method that takes into account the parallel structure. To simplify the notations we present the case of parallel robots with six degrees of freedom. Examples concerning reduced mobility robots are given in [18].

The inverse dynamic model gives the forces and torques of motorized joints as a function of the desired trajectory of the mobile platform.

Decomposing the robot into two subsystems: the platform and the legs. The dynamics of the platform is calculated by Newton-Euler equation as a function of the Cartesian variables (location, velocity and acceleration of the platform), whereas the dynamics of each leg i is calculated as a function of its joint variables $(q_i, \dot{q}_i, \ddot{q}_i)$ for $i=1,\ldots,m$. The active joint torques are obtained by projecting these two dynamics on the active joint axes.

To project the dynamics of the platform on the active joint space we multiply it by the transpose of the robot Jacobian matrix, and to project the leg dynamics on the active joint space we use the Jacobian matrix between these two spaces. Thus the dynamic model of the parallel structure is given by the following equation:

$$\Gamma = \mathrm{J}_P^T(\boldsymbol{\Phi}_P + F_{ep}) + \sum_{i=1}^{m} \left(\frac{\partial \dot{q}_i}{\partial \dot{q}_a} \right)^T \Gamma_i \tag{26}$$

Where

$\boldsymbol{\Phi}_P$ is the wrench required to achieve the motion of the platform, calculated by Newton- Euler equation (13) with the platform inertial parameters.

F_{ep} external forces on the platform.

J_P is the (6×6) kinematic Jacobian matrix of the robot, which gives the platform screw V_P (translational and angular velocities) as a function of the active joint velocities:

$$V_P = \mathrm{J}_P \dot{q}_a \tag{27}$$

Γ_i is the inverse dynamic model of leg i, it is a function of $(q_i, \dot{q}_i, \ddot{q}_i)$, which can be obtained in terms of the platform location, velocity and acceleration, using the inverse kinematic models of the legs [19]. We note that q_i does not include the passive joint variables connecting the legs to the platform.

The calculation of J_p is obtained by inverting $\mathrm{J}_\mathrm{p}^{-1}$, which is easy to obtain for most parallel structures. The detailed calculation of $\partial \dot{q}_i / \partial \dot{q}_a$ is given in [18].

6 Inverse Dynamic Modeling of Robots with Elastic Joints

A tree structure robots with lumped elastic joints can be described using the method presented in section 2. Each joint could be elastic or rigid.

6.1 Lagrange Dynamic Form

The general form of the dynamic model of a system with flexible joints is the same as (6), it can be partitioned as follows:

$$\Gamma = \begin{bmatrix} \Gamma_r \\ \Gamma_f \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & A_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_r \\ \ddot{q}_f \end{bmatrix} + \begin{bmatrix} H_r(q,\dot{q}) \\ H_f(q,\dot{q}) \end{bmatrix} \tag{28}$$

Where $(q,\dot{q},\ddot{q})$ are the (n×1) vectors of positions, velocities, and accelerations of rigid and elastic joints;

$H(q,\dot{q})$ is the (n×1) vector of Coriolis, centrifugal and gravity forces,

$A(q)$ is the (n×n) inertia matrix of the system,

Γ_r is the vector of rigid joint torques, Γ_f is the vector of elastic joint torques.

If joint j is flexible:

$$\Gamma_j = -\Delta q_j \, k_j = -(q_j - q_{rj}) \, k_j \tag{29}$$

where k_j is the stiffness of the elastic joint, Δq_j is the elastic deformation, q_{rj} is the reference joint position corresponding to zero elastic deformation.

The inverse dynamic model calculates the input torques and the elastic accelerations as a function of the joint positions, velocities and rigid joint accelerations. The definition of the direct model definition is similar to that of the rigid robots; it gives the joint accelerations in terms of the input torques.

Eq. (28) can be used to calculate the inverse model, we have first to calculate the elastic accelerations from the second row, and then we calculate the rigid joint torques from the first row.

6.2 Direct Dynamics of Systems with Flexible Joints Using Recursive NE

The direct dynamic model of system with flexible joints can be calculated using the recursive direct dynamic model algorithm of rigid joints presented in section (3.3) after putting $\Gamma_j = - \Delta q_j \, k_j$ for the elastic joints. We note that the recursive algorithms of sections 6.2 and 6.3 can be used in case of passive joints without actuators by putting the torque of these joints equal to 0.

6.3 Inverse Dynamics of Systems with Flexible Joints Using Recursive NE

The recursive inverse dynamic algorithm of rigid links cannot be used for system with flexible joints since the accelerations of the flexible joints are unknown. But it can be used to obtain the A and H matrices as explained in section (3.3.1), then we can proceed as explained in section 6.1 for the calculation of $\ddot{q}_f$ and Γ_r.

To solve this problem we use three recursive steps [20].

i) The first forward step is the same as that of the direct dynamic model (section 3.3).

ii) The second backward recursive equations calculate the matrices giving the elastic accelerations $\ddot{q}_e$ and ${}^{j}F_j$ in terms of ${}^{i}\dot{V}_i$. These matrices can be calculated for $j = n...1$ as follows:

- if joint j is elastic:

$$\begin{aligned} H_j &= {}^{j}a_j \, {}^{j}J_j^* \, {}^{j}a_j \\ {}^{j}K_j &= {}^{j}J_j^* - {}^{j}J_j^* \, {}^{j}a_j \, H_j^{-1} \, {}^{j}a_j^T \, {}^{j}J_j^* \\ {}^{j}\alpha_j &= {}^{j}K_j \, {}^{j}\gamma_j + {}^{j}J_j^* \, {}^{j}a_j \, H_j^{-1} (-k_j \Delta q_j + {}^{j}a_j^T \, {}^{j}\beta_j^*) - {}^{j}\beta_j^* \end{aligned} \tag{30}$$

- If joint j is rigid:

$$\begin{aligned} {}^{j}K_j &= {}^{j}J_j^* \\ {}^{j}\alpha_j &= {}^{j}K_j ({}^{j}\gamma_j + {}^{j}a_j \, \ddot{q}_j) - {}^{j}\beta_j^* \end{aligned} \tag{31}$$

if $a(j) \neq 0$, calculate:

$$\begin{aligned} {}^{i}\beta_i^* &= {}^{i}\beta_i^* - {}^{j}T_i^T \, {}^{j}\alpha_j \\ {}^{i}J_i^* &= {}^{i}J_i^* + {}^{j}T_i^T \, {}^{j}K_j \, {}^{j}T_i \end{aligned} \tag{32}$$

The previous equations are initialized by: ${}^{j}J_j^* = {}^{j}J_j$, ${}^{j}\beta_j^* = {}^{j}\beta_j$.

The third recursive equations (for $j = 1,...,n$) calculate $\ddot{q}_j$ for the elastic joints and the joint torques for the rigid joints using the following equations:

$${}^{j}F_j = {}^{j}K_j \, {}^{j}T_i \, {}^{i}\dot{V}_i + {}^{j}\alpha_j \tag{33}$$

- if j is elastic:

$$\begin{aligned} \ddot{q}_j &= H_j^{-1} [-{}^{j}a_j \, {}^{j}J_j^* ({}^{j}T_i \, {}^{i}\dot{V}_i + {}^{j}\gamma_j) - \text{kj} \, \Delta \, \text{qj} + {}^{j}a_j^T \, {}^{j}\beta_j^*] \\ {}^{j}\dot{V}_j &= {}^{j}T_i \, {}^{i}\dot{V}_i + {}^{j}a_j \, \ddot{q}_j + {}^{j}\gamma_j \end{aligned} \tag{34}$$

- if j is rigid

$$\Gamma_j = {}^{\text{j}}F_{\text{j}}^T \, {}^{\text{j}}a_{\text{j}} + I_{aj} \, \ddot{q}_j + F_{sj} \, sign(\dot{q}_j) + F_{vj} \, \dot{q}_j \tag{35}$$

7 Dynamic Modeling of Robots with a Mobile Base

This section treats mobile robots which are composed of a tree structure with a moving base. It includes a big number of systems such as: cars, mobile manipulators, walking robots, Humanoid robots, eel like robots [21], snakes like robots, flying

robots, spatial vehicles, etc. The difference between all of these systems will be in the calculation of the interaction forces with the environment. In the previous sections the acceleration of the base is known as equal to zero, whereas in the case of moving base the acceleration of the base must be determined in both direct and inverse dynamic models. The inverse dynamic model, which is used in general in the control problems, can be used in simulation too when the objective is to study the evolution of the base giving joint positions, velocities and accelerations of the other joints. The direct dynamic model can be used in simulation when the joint torques are specified.

We use the same notations of section 2 to describe the structure. The base fixed frame R_0 is defined wrt the world fixed frame R_w by the transformation matrix ${}^{w}h_0$. This matrix is supposed known at t = 0, it will be updated by integrating the base acceleration. The velocity and acceleration of the base are represented by the (6x1) vectors V_0 and $\dot{V}_0$ respectively.

7.1 General Form of the Dynamic Model

The dynamic model of a robot with a moving base can be represented by the following relation:

$$\begin{bmatrix} 0_{6x1} \\ \Gamma \end{bmatrix} = A \begin{bmatrix} {}^{0}\dot{V}_0 \\ \ddot{q} \end{bmatrix} + H = \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & A_{22} \end{bmatrix} \begin{bmatrix} {}^{0}\dot{V}_0 \\ \ddot{q} \end{bmatrix} + \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} \tag{36}$$

Γ $(n \times 1)$ vector of joint torques, q $(n \times 1)$ vector of joint positions,

A is the $(6+n) \times (6+n)$ inertia matrix of the robot,

H is the $(n+6) \times 1$ vector representing the Coriolis, centrifugal, gravity and external forces effect on the robot.

The inverse dynamic model gives the joint torques and the base acceleration in terms of the desired trajectory (position, velocity and acceleration) of the articulated system (links 1 to n) and the base position and velocity. At first, the inverse dynamic model is solved by using the first row of (36) to obtain the base acceleration:

$${}^{0}\dot{V}_0 = -(A_{11})^{-1} (H_1 + A_{12}\ddot{q}) \tag{37}$$

Then the second row of (36), can be used to find the joint torques:

$$\Gamma = A_{12}^T \, {}^{0}\dot{V}_0 + A_{22}\, \ddot{q} + H_2 \tag{38}$$

The direct dynamic model gives the joint accelerations and the base acceleration in terms of the position and velocity of the base and the articulated system and the joint input torques. Thus using (36), the direct dynamic model is solved as follows:

$$\begin{bmatrix} \dot{V}_0 \\ \ddot{q} \end{bmatrix} = A^{-1} \begin{bmatrix} -H_1 \\ \Gamma - H_2 \end{bmatrix} \tag{39}$$

The calculation of A and H can be done by Lagrange method. They can also be calculated using the inverse dynamic model of tree structure of section (3.2) and using

the procedure of section (3.3.1). The mobile base can be taken into account as follows:

- the initial values of velocities and accelerations are calculated from those of the base as follows: ${}^0V_0 = {}^0V_b$, ${}^0\dot{v}_0 = {}^0\dot{v}_0 - g$, ${}^0\dot{\omega}_0 = {}^0\dot{\omega}_b$, where the subscript b indicates the base.
- The backward recursive equations must continue to $j = 0$, where this new iteration will obtain the 6 equations of Newton-Euler equations of the base.

Solving the inverse and direct dynamic problems using A and H is very time consuming for systems with big number of degrees of freedom (as the eel like robot). Therefore, we propose here to use a recursive method, which is easy to programme, and its computational complexity is linear in the number of degrees of freedom.

7.2 Recursive NE Calculation of the Inverse Dynamic Model

The inverse dynamic algorithm in this case consists of three recursive equations (a forward, then a backward, then a forward) [21].

i) Forward recursive calculation:

In this step we calculate the screw transformation matrices, the link rotational velocities, and the elements of the accelerations and external wrenches on the links which are independent of the acceleration of the robot base ($\dot{v}_0, \dot{\omega}_0$). Thus we calculate for $j = 1,...,n$ the following: ${}^{j}T_{i}$, ${}^{j}\omega_j$ as given in section (3.2). We calculate also ${}^{j}\beta_{j}, {}^{j}\gamma_{j}$ using (17) and (18) they represent the elements of total forces and acceleration which are independent of the base acceleration. We define also:

$$ {}^{j}\zeta_{j} = {}^{j}\gamma_{j} + \ddot{q}_j \, {}^{j}a_j \tag{40} $$

ii) Backward recursive equations:

In this step we obtain the base acceleration using the inertial parameters of the composite link *0*, where the composite link j consists of the links j, j+1, …, n.

We note that (17), giving the equilibrium equation of link j, can be rewritten as:

$$ {}^{j}F_{j} = {}^{j}J_{j} \, {}^{j}\dot{V}_{j} - {}^{j}\beta_{j} + \sum_{k} {}^{k}T_{j}^{T} \, {}^{k}F_{k} \tag{41} $$

Applying the Newton-Euler equations on the composite link j, we obtain:

$$ {}^{j}F_j = {}^{j}J_j \, {}^{j}\dot{V}_j - {}^{j}\beta_j + \sum_{s(j)} {}^{s(j)}T_j^T \left({}^{s(j)}J_{s(j)} \, {}^{s(j)}\dot{V}_{s(j)} - {}^{s(j)}\beta_{s(j)} \right) \tag{42} $$

Where s(j) means all the links succeeding joint j, that is to say joining j to the terminal links.

Substituting for ${}^{s(j)}\dot{V}_{s(j)}$ in terms of ${}^{j}\dot{V}_{j}$ using (12), we obtain:

$$ {}^{s(j)}\dot{V}_{s(j)} = {}^{s(j)}T_j \, {}^{j}\dot{V}_j + \sum_r {}^{s(j)}T_r \, {}^{r}\zeta_r \tag{43} $$

Where r denotes all links between j and s(j).

Thus we obtain:

$$ {}^{j}F_j = {}^{j}J_j^c \, {}^{j}\dot{V}_j - {}^{j}\beta_j^c \tag{44} $$

with:

$$ \begin{aligned} {}^{j}J_j^c &= {}^{j}J_j^c + \sum_k {}^{k}T_j^T \, {}^{k}J_k^c \, {}^{k}T_j \\ {}^{j}\beta_j^c &= {}^{j}\beta_j^c - \sum_k {}^{k}T_j^T \, {}^{k}\beta_k^c + {}^{k}T_j^T \, {}^{k}J_k^c \, {}^{k}\zeta_k \end{aligned} \tag{45} $$

${}^{j}J_j^c$ is the inertial matrix of the composite link j.

For j = 0, and since ${}^{0}F_0$ is equal to zero, we obtain using (44):

$$ {}^{0}\dot{V}_0 = \left({}^{0}J_0^c \right)^{-1} {}^{0}\beta_0^c \tag{46} $$

To conclude, the recursive equations of this step consist of initializing ${}^{j}J_j^c = {}^{j}J_j$, ${}^{j}\beta_j^c = {}^{j}\beta_j$ and then calculating (45) *for j = n,…, 0.* At the end ${}^{0}\dot{V}_0$ is calculated by (46).

iii) Forward recursive equations:

After calculating ${}^{0}\dot{V}_0$, the wrench ${}^{j}F_j$ and the joint torques are obtained using equations (12) and (45) for j= 1,…, n as:

$$ \begin{aligned} {}^{j}\dot{V}_j &= {}^{j}T_i \, {}^{i}\dot{V}_i + {}^{j}\zeta_j \\ {}^{j}F_j &= {}^{j}J_j^c \, {}^{j}\dot{V}_j - {}^{j}\beta_j^c \end{aligned} \tag{47} $$

The joint torque is calculated by projecting ${}^{j}F_j$ on the joint axis, and by taking into account the friction and the actuators inertia:

$$ \Gamma_j = {}^{j}F_j^T \, {}^{j}a_j + I_{aj} \, \ddot{q}_j + F_{sj} \, sign(\dot{q}_j) + F_{vj} \, \dot{q}_j \tag{48} $$

It is to be noted that the inverse dynamic model algorithm can be used in the dynamic simulation of the mobile robot when the objective is to study the effect of the joint motions on the base. In this case the joint positions, velocities and accelerations trajectories are given. At each sampling time the acceleration of the base will be integrated to provide the angular and linear velocities for the next sampling time.

7.3 Recursive Direct Dynamic Model

The direct dynamic model consists of three recursive calculations forward, backward and forward:

i) Forward recursive equations:

We calculate for *j = 1, ..., n* the link rotational velocities using (11) and the terms ${}^{j}\gamma_{j}$ and ${}^{j}\beta_{j}$ of Cartesian accelerations and equilibrium equations of the links that are independent of the accelerations of the base and the joints.

ii) Backward recursive equations:

In this second step, we initialize ${}^{j}J_{j}^{*} = {}^{j}J_{j}$, ${}^{j}\beta_{j}^{*} = {}^{j}\beta_{j}$ and then we calculate for j = n,…,1 the following elements, which permit to calculate ${}^{j}F_{j}$ and $\ddot{q}_j$ in terms of ${}^{i}\dot{V}_i$ and will be used in the third recursive equations:

$$
\begin{aligned}
H_j &= {}^{j}a_j^T \, {}^{j}J_j^* \, {}^{j}a_j + Ia_j \\
{}^{j}K_j &= {}^{j}J_j^* - {}^{j}J_j^* \, {}^{j}a_j H_j^{-1} \, {}^{j}a_j^T \, {}^{j}J_j^* \\
{}^{i}J_i^* &= {}^{i}J_i^* + {}^{j}T_i^T \, {}^{j}K_j \, {}^{j}T_i \\
{}^{j}\alpha_j &= {}^{j}K_j \, {}^{j}\gamma_j + {}^{j}J_j^* \, {}^{j}a_j H_j^{-1} \left(\Gamma_j + {}^{j}a_j^T \, {}^{j}\beta_j^* \right) - {}^{j}\beta_j^* \\
{}^{i}\beta_i^* &= {}^{i}\beta_i^* - {}^{j}T_i^T \, {}^{j}\alpha_j
\end{aligned}
\tag{49}
$$

iii) Forward recursive equations:

At first, the base acceleration is calculated by the following relation:

$$
{}^{0}\dot{V}_0 = \left({}^{0}J_0^* \right)^{-1} {}^{0}\beta_0^* \tag{50}
$$

$\ddot{q}_j$ and ${}^{j}F_{j}$ (if desired) are calculated for *j=1,…,n* using the following equations:

$$
\begin{aligned}
\ddot{q}_j &= H_j^{-1} \left[- {}^{j}a_j^T \, {}^{j}J_j^* \left({}^{j}T_i \, {}^{i}\dot{V}_i + {}^{j}\gamma_j \right) + \tau_j + {}^{j}a_j^T \, {}^{j}\beta_j^* \right] \\
{}^{j}F_j &= {}^{j}K_j \, {}^{j}T_i \, {}^{i}\dot{V}_i + {}^{j}\alpha_j
\end{aligned}
\tag{51}
$$

where: $\tau_j = \Gamma_j - F_{sj} \; sign(\dot{q}_j) \; - \; F_{vj} \; \dot{q}_j$

$$
{}^{j}\dot{V}_j = {}^{j}T_i \, {}^{i}\dot{V}_i + {}^{j}a_j \ddot{q}_j + {}^{j}\gamma_j \tag{52}
$$

8 Dynamic Modeling of Wheeled Mobile Robots

The robot is composed of a cart and classical wheels. The wheels could be fixed, steering or casters. Each of these wheels has a rotational variable φ_j. The steering and castor wheels have also an orientation variable β_j. The vector of configuration variables is composed of the wheel variables φ_j, β_j and the posture variables ζ giving the position and orientation of the cart. To simplify the presentation we consider that the motion of the cart lies in a plane thus the posture is composed of the x,y position coordinates and the orientation is the angle around the z axis. For more details on these robots the reader can consult [22]. These mobile robots contain non-holonomic constraint equations between the configuration velocities and have some joints which are not motorized. The constraint equations can be written as:

$$W\dot{q} = 0, and\ \dot{q} = S\dot{q}_m \tag{53}$$

Where $q = \begin{bmatrix} \zeta^T & \beta^T & \varphi^T \end{bmatrix}^T$, q_m is the motorized joint variables among β and φ.

The difference with respect to the previous sections is that the coupling between the configuration variables is kinematically defined.

In this case we apply the procedure of closed loop structure of section 4 such that:

$$\begin{bmatrix} \Gamma_m \\ 0 \end{bmatrix} = \Gamma(q,\dot{q},\ddot{q}) + \mathrm{W}^{\mathrm{T}}\lambda$$
$$\Gamma_m = \left(\frac{\partial \dot{q}}{\partial \dot{q}_m} \right)^T \Gamma = S^T \Gamma \tag{54}$$

Γ is the torques corresponding to the configuration variables without taking into account the constraints, it can be calculated using the method of section 3.2, Γ_m correspond to the real motor torques.

9 Conclusions

This paper presents the inverse and direct dynamic modeling of different types of robots. The dynamic models are developed using the recursive Newton-Euler formalism. The inverse model provides the torque of the joint and the acceleration of the free degrees of freedom such as the elastic joints, or the acceleration of the base in case of mobile base. The direct model provides the joint acceleration of the joints including those of the free degrees of freedom.

These algorithms constitute the generalization of the algorithms of articulated rigid manipulators to the other cases.

The proposed methods have been applied on more complicated systems such as: flexible link robots [23], Micro Continuous system [24] and Hybrid structure, where the robot is composed of parallel modules, which are connected in series [25].

References

1. Khalil, W., Kleinfinger, J.-F.: A new geometric notation for open and closed-loop robots. In: Proc. IEEE Int. Conf. on Robotics and Automation, San Francisco, pp. 1174–1180. IEEE Press, New York (1986)
2. Gautier, M., Khalil, W.: Direct calculation of minimum set of inertial parameters of serial robots. IEEE Trans. on Robotics and Automation RA-6(3), 368–373 (1990)
3. Khalil, W., Bennis, F.: Comments on Direct Calculation of Minimum Set of Inertial Parameters of Serial Robots. IEEE Trans. on Rob. & Automation RA-10(1), 78–79 (1994)
4. Khalil, W., Bennis, F.: Symbolic calculation of the base inertial parameters of closed-loop robots. International Journal of Robotics Research 14(2), 112–128 (1995)
5. Craig, J.J.: Introduction to robotics: mechanics and control. Addison Wesley Publishing Company, Reading (1986)
6. Khalil, W., Dombre, E.: Modeling identification and control of robots. Hermes, Penton-Sciences, London (2002)
7. Angeles, J.: Fundamentals of Robotic Mechanical Systems, 2nd edn. Springer, New York (2002)
8. Luh, J.Y.S., Walker, M.W., Paul, R.C.P.: On-line computational scheme for mechanical manipulators. Trans. of ASME, J. of Dynamic Systems, Measurement, and Control 102(2), 69–76 (1980)
9. Khosla, P.K.: Real-time control and identification of direct drive manipulators. Ph. D. Thesis, Carnegie Mellon (1986)
10. Khalil, W., Kleinfinger, J.-F.: Minimum operations and minimum parameters of the dynamic model of tree structure robots. IEEE J. of Robotics and Automation RA-3(6), 517–526 (1987)
11. Khalil, W., Creusot, D.: SYMORO+: a system for the symbolic modelling of robots. Robotica 15, 153–161 (1997)
12. Hollerbach, J.M.: An iterative lagrangian formulation of manipulators dynamics and a comparative study of dynamics formulation complexity. IEEE Trans. on Systems, Man, and Cybernetics SMC-10(11), 730–736
13. Walker, M.W., Orin, D.E.: Efficient dynamic computer simulation of robotics mechanism. Trans. of ASME, J. of Dynamic Systems, Measurement, and Control 104, 205–211 (1982)
14. Armstrong, W.W.: Recursive solution to the equation of motion of an N-links manipulator. In: Proc. 5th World Congress on Theory of Machines and Mechanisms, pp. 1343–1346
15. Brandl, H., Johanni, R., Otter, M.: A very efficient algorithm for the simulation of robots and multibody systems without inversion of the mass matrix. In: Proc. IFAC Symp. on Theory of Robots, Vienne, pp. 365–370 (1986)
16. Featherstone, R.: The calculation of robot dynamics using articulated-body inertias. The Int. J. of Robotics Research 2(3), 87–101 (1983)
17. Nakamura, Y., Ghodoussi, M.: Dynamic computation of closed-link robot mechanism with nonredundant and redundant actuators. IEEE Trans. Robotics and Automation (5), 294–302 (1989)
18. Khalil, W., Ibrahim, O.: General solution for the Dynamic modeling of parallel robots. Journal of Intelligent and Robotic Systems 49, 19–37 (2007)
19. Khalil, W., Guegan, S.: Inverse and Direct Dynamic Modeling of Gough-Stewart Robots. IEEE Transactions on Robotics and Automation 20(4), 754–762 (2004)
20. Khalil, W., Gautier, M.: Modeling of mechanical systems with lumped elasticity. In: Proc. IEEE Int. Conf. on Robotics and Automation, San Francisco, pp. 3965–3970 (2000)

21. Khalil, W., Gallot, G., Boyer, F.: Dynamic Modeling and Simulation of a 3-D Serial Eel-Like Robot. IEEE Transactions on Systems, Man and Cybernetics, Part C: Application and reviews 37(6), 1259–1268 (2007)
22. Campion, G., Chung, W.: Wheeled Robots. In: Siciliano, B., Khatib, O. (eds.) Handbook of Robotics, pp. 391–410. Springer, Heidelberg (2008)
23. Boyer, F., Khalil, W.: An efficient calculation of flexible manipulator inverse dynamic. Int. Journal of Robotics Research 17(3), 282–293 (1998)
24. Boyer, F., Porez, M., Khalil, W.: Macro-continuous torque algorithm for a three-dimensional eel-like robot. IEEE Robotics Transaction 22(4), 763–775 (2006)
25. Ibrahim, O., Khalil, W.: Inverse and direct dynamic models of Hybride robots. Mechanism and Machine Theory 45(4), 627–640 (2010)