DYNAMIC MODELING OF ROBOTS USING RECURSIVE NEWTON-EULER TECHNIQUES

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

Keywords: Dynamic modelling, Newton-Euler, recursive calculation, tree structure, parallel robots, flexible joints, mobile robots.

Abstract: This paper present the use of recursive Newton-Euler to model different robotics systems. The main advantages of this technique are the facility of implementation by numerical or symbolical programming and providing models with reduced number of operations. In this paper the inverse and direct dynamic models of different robotics systems will be presented. At first we start by rigid tree structure robots, then these algorithms will be generalized for closed loop robots, parallel robots, and robots with lumped elasticity. At the end the case of robots with moving base will be treated.

1 INTRODUCTION

The dynamic modelling 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 for different robotics systems is easy to develop and programme. The proposed algorithm can be extended to many types of structures; serial, tree structure, closed, parallel, with a fixed base or with moving platform. The same technique can be used for robots with lumped elasticity or flexible links.
In section 2 we will recall the method used to describe the kinematics of the structure, and then in section 3 we present the inverse and the direct dynamic modeling of tree structure rigid robots which are considered as the base methods. The following sections present the generalization to the other systems.

2 DESCRIPTION OF THE KINEMATICS OF ROBOTS

The geometry of the structures will be described using the Modified Denavit and Hartenberg method as proposed in (Khalil and Kleinfinger, 1986). This method can take into account tree structures and closed loop robots. Its use facilitates the calculation of the base inertial parameters of robots (Gautier and Khalil, 1988, Khalil W., Bennis F., 1994, Khalil and Bennis, 1995).

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 either revolute or prismatic, rigid or elastic. The links are numbered consecutively from the base, link 0, to the terminal links. Joint j connects 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 (Figure 1):

- $\mathbf{z}_i$ is along the axis of joint i;
- $\mathbf{x}_i$ is taken along the common normal between $\mathbf{z}_i$ and one of the succeeding joint axes, which are fixed on link i.

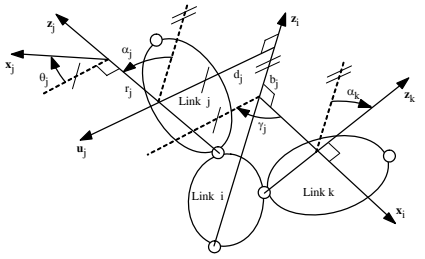

Figure 1: Geometric parameters for a link i.

In general the homogeneous transformation matrix ${}^{i}\mathbf{T}_j$, which defines the frame R_j relative to frame R_i is obtained as a function of six geometric parameters (γ_j , b_j, α_j, d_j, θ_j, r_j). Thus ${}^{i}\mathbf{T}_j$ is obtained as:

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

After developing, this matrix can be partitioned as follows:

$$ {}^{i}\mathbf{T}_j = \begin{bmatrix} {}^{i}\mathbf{R}_j & {}^{i}\mathbf{P}_j \\ \mathbf{0}_{1x3} & 0 \end{bmatrix} \quad (1) $$

Where **R** defines the (3×3) rotation matrix and P defines the (3×1) vector defining the position of the origin of frame j with respect to frame i.

If $\mathbf{x}_i$ is along the common normal between $\mathbf{z}_i$ and $\mathbf{z}_j$, the parameters γ_j and b_j will be equal to zero.

The joint variable of joint j is denoted by:

$$ q_j = \bar{\sigma}_j \theta_j + \sigma_j r_j $$

where $\sigma_j = 0$ if joint j is revolute, $\sigma_j = 1$ if joint j is prismatic, and $\bar{\sigma}_j = 1 - \sigma$. We set $\sigma_j = 2$ to define a frame R_j fixed with respect to frame a(j). In this case, q_j and $\bar{\sigma}_j$ are not defined.

The serial structure is a special case of a tree structure where a(j)=j-1, $\gamma_j = 0$, and $b_j = 0$ 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 position and orientation of all the links can be determined as a function of the active joint variables.

To determine the geometric parameters of a mechanism with closed chains, we proceed 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) For each cut joint define two supplementary frames on one of the links connected by this joint. Assuming that a cut joint is numbered k (where k=n+1,…, L) and that the links connected by joint k are numbered i and j (where i and j <n) the frames will be defined as follows (Figure 2):

- frame R_k is defined fixed on link j such that a(k)=i, the axis $\mathbf{z}_k$ is along the axis of joint k, and $\mathbf{x}_k$ is along the common normal between $\mathbf{z}_k$ and $\mathbf{z}_j$. The matrix ${}^{i}\mathbf{T}_k$ will be determined using the general parameters γ_k, b_k, α_k, d_k, θ_k, r_k.
- frame R_{k+B} is aligned with R_k that is to say it is fixed on link j, but a(k+B)= j. The geometric parameters defining R_{k+B} are constant, we note that r_{k+B} and θ_{k+B} are zero since $\mathbf{x}_{k+B}$ is normal to $\mathbf{z}_j$.

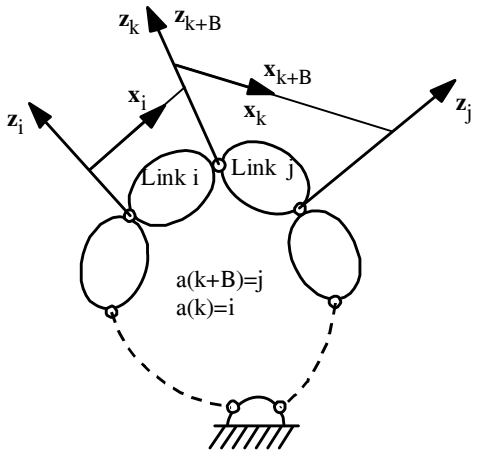

Figure 2: Frames of a cut joint k.

The joint variables are denotes as:

$$ \mathbf{q} = \begin{bmatrix} \mathbf{q}_{tr} \\ \mathbf{q}_c \end{bmatrix}, \mathbf{q}_{tr} = \begin{bmatrix} \mathbf{q}_a \\ \mathbf{q}_p \end{bmatrix} \quad (2) $$

- $\mathbf{q}_{tr}$ vector containing the tree structure joint variables;
- $\mathbf{q}_a$ vector containing the N active joint variables;
- $\mathbf{q}_p$ vector containing the p=n−N passive joint variables of the equivalent tree structure;
- $\mathbf{q}_c$ vector containing the B variables of the cut joints.

Only the N active variables $\mathbf{q}_a$ are independent. Since R_k and R_{k+B} are aligned, the geometric constraint equations for each loop, which can be used to calculate the passive joint variables in terms of the active joint variables, can be written as:

$$^{k+B}\mathbf{T}_j \ldots {}^{i}\mathbf{T}_k = \mathbf{I}_4 \quad (3)$$

The kinematic constraint equations are obtained by using the fact that the screw of frame k is equal to that of frame k+B:

$$\begin{aligned} {}^{0}\mathbb{V}_k &= {}^{0}\mathbb{V}_{k+B} \\ \mathbf{J}_k\,\dot{\mathbf{q}}_{\mathbf{b1}} &= \mathbf{J}_{k+B}\dot{\mathbf{q}}_{\mathbf{b2}} \end{aligned} \quad (4)$$

$\mathbb{V}_j$ (6×1) kinematic screw vector of frame j, given by:

$$\mathbb{V}_j = \begin{bmatrix} \mathbf{V}_j^{\,T} & \boldsymbol{\omega}_j^{\,T} \end{bmatrix}^T \quad (5)$$

$\mathbf{V}_j$ linear velocity of the origin of frame R_j,

$\boldsymbol{\omega}_j$ angular velocity of frame j,

$\dot{\mathbf{q}}_{\mathbf{b1}}$ joint velocities from the base to frame k, through branch 1,

$\dot{\mathbf{q}}_{\mathbf{b2}}$ joint velocities from the base to frame k through branch 2.

3 DYNAMIC MODELING OF TREE STRUCTURE ROBOTS

3.1 Introduction

The most common in use methods to calculate the dynamic models are the Lagrange equations and the Newton Euler Equations (Craig 1986, Khalil and Dombre 2002, Angeles 2006).

The Lagrange equation is given as:

$$\boldsymbol{\Gamma} = \frac{\mathrm{d}}{\mathrm{dt}}\left[\frac{\partial L}{\partial \dot{\mathbf{q}}}\right]^T - \left[\frac{\partial L}{\partial \mathbf{q}}\right]^T \quad (6)$$

where Γ is the joint torques and forces, L is the Lagrangian of the robot defined as the difference between the kinetic energy E and the potential energy U of the system : L = E – U. After developing we obtain:

$$\boldsymbol{\Gamma} = \mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{H}(\mathbf{q},\dot{\mathbf{q}}) \quad (7)$$

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

Solving the previous equation to find Γ in terms of $(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})$ is known as the inverse dynamic problem, and solving it to obtain $\ddot{\mathbf{q}}$ in terms of $(\mathbf{q},\dot{\mathbf{q}},\boldsymbol{\Gamma})$ is known as the direct dynamic model. The inverse dynamic model is obtained by substituting $(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})$ into (7), whereas the direct model needs to inverse the inertia matrix.

$$\ddot{\mathbf{q}} = -\left(\mathbf{A}\right)^{-1}\left(\boldsymbol{\Gamma} - \mathbf{H}\right) \quad (8)$$

The calculation of the Lagrange equations for systems with big number of degrees of freedom using developed symbolic methods is time consuming, and the obtained model will need more time to execute with respect to that of recursive methods. The recursive Newton-Euler algorithms have been shown to be an excellent tool to model rigid robots (Khalil and Kleinfinger, 1987, Khalil and Creusot, 1987, Khosla, 1987). In (Hollerbach, 1980) an efficient recursive Lagrange algorithm is presented but without achieving better performances than that of Newton-Euler.

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

$$^{j}\mathbb{F}_j = {}^{j}\mathbb{J}_j\,{}^{j}\dot{\mathbb{V}}_j + \begin{bmatrix} {}^{j}\boldsymbol{\omega}_j \times \left({}^{j}\boldsymbol{\omega}_j \times {}^{j}\mathbf{MS}_j\right) \\ {}^{j}\boldsymbol{\omega}_j \times \left({}^{j}\mathbf{J}_j\,{}^{j}\boldsymbol{\omega}_j\right) \end{bmatrix} \quad (9)$$

where

$$^{j}\mathbb{F}_j = \begin{bmatrix} {}^{j}\mathbf{F}_j \\ {}^{j}\mathbf{M}_j \end{bmatrix} \quad (10)$$

$\boldsymbol{\omega}_j$ the angular velocity of link j;

$\dot{\mathbf{V}}_j$ the linear acceleration of the origin of frame j;

$\mathbb{F}_j$ total external wrench on link j;

$\mathbf{F}_j$ total external forces on link j;

$\mathbf{M}_j$ total external moments on link j about O_j;

$\mathbb{J}_j$ (6×6) inertia matrix of link j:

$$^{j}\mathbb{J}_j = \begin{bmatrix} \mathrm{M}_j\mathbf{I}_3 & -{}^{j}\hat{\mathbf{MS}}_j \\ {}^{j}\hat{\mathbf{MS}}_j & {}^{j}\mathbf{J}_j \end{bmatrix} \quad (11)$$

Where M_j, $\mathbf{MS_j}$ and $\mathbf{J_j}$ are the standard inertial parameters of link j. They are respectively, the mass, the first moments, and the inertia matrix about the origin.

3.2 Calculation of the Inverse dynamics using recursive NE algorithm

The algorithm consists of two recursive computations (Luh, Walker and Paul, 1980): forward recursion and backward recursion. The forward equations, from link 1 to link n, compute the link velocities and accelerations and consequently the dynamic wrench on each link. The backward equations, from link n to the base, provide the reaction wrenches on the links and consequently the joint torques.

This method gives the joint torques in terms of the joint positions, velocities and accelerations

without explicitly computing the matrices **A** and **H**. That is to say the algorithm will be denoted by:

$$\mathbf{\Gamma} = \mathrm{NE}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{f}_e, \mathbf{m}_e) \quad (12)$$

Where $\mathbf{f}_e$ and $\mathbf{m}_e$ are the external forces and moments of the links of the robot on the environment.

The forward recursive equations are based on the following equations (Khalil and Dombre 2002):

$${}^j\mathbb{V}_j = {}^j\mathbb{T}_i \, {}^i\mathbb{V}_i + \dot{q}_j \, {}^j\mathbb{a}_j \quad (13)$$

$${}^j\dot{\mathbb{V}}_j = {}^j\mathbb{T}_i \, {}^i\dot{\mathbb{V}}_i + {}^j\boldsymbol{\gamma}_j + \ddot{q}_j \, {}^j\mathbb{a}_j \quad (14)$$

$${}^j\boldsymbol{\gamma}_j = \begin{bmatrix} {}^j\mathbf{R}_i \left[{}^i\boldsymbol{\omega}_i \times \left({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{P}_j \right) \right] + 2\sigma_j ({}^j\boldsymbol{\omega}_i \times \dot{q}_j \, {}^j\mathbf{a}_j) \\ \bar{\sigma}_j \, {}^j\boldsymbol{\omega}_i \times \dot{q}_j \, {}^j\mathbf{a}_j \end{bmatrix} \quad (15)$$

where

${}^j\mathbb{a}_j$ is the (6x1) column matrix given as :

$${}^j\mathbb{a}_j = [0 \;\; 0 \;\; \sigma_j \;\; 0 \;\; 0 \;\; \bar{\sigma}_j]^T \quad (16)$$

${}^j\mathbb{T}_i$ and the screw transformation matrix is:

$${}^j\mathbb{T}_i = \begin{bmatrix} {}^j\mathbf{R}_i & -{}^j\mathbf{R}_i \, {}^i\hat{\mathbf{P}}_j \\ \mathbf{0}_{3x3} & {}^j\mathbf{R}_i \end{bmatrix} \quad (17)$$

The forward algorithm is given for j=1,…, n, with i = a(j), as follows:

$${}^j\boldsymbol{\omega}_i = {}^j\mathbf{R}_i \, {}^i\boldsymbol{\omega}_i \quad (18)$$

$${}^j\boldsymbol{\omega}_j = {}^j\boldsymbol{\omega}_i + \bar{\sigma}_j \, \dot{q}_j \, {}^j\mathbf{a}_j \quad (19)$$

$${}^j\dot{\boldsymbol{\omega}}_j = {}^j\mathbf{R}_i \, {}^i\dot{\boldsymbol{\omega}}_i + \bar{\sigma}_j \, (\ddot{q}_j \, {}^j\mathbf{a}_j + {}^j\boldsymbol{\omega}_i \times \dot{q}_j \, {}^j\mathbf{a}_j) \quad (20)$$

$${}^j\dot{\mathbf{V}}_j = {}^j\mathbf{R}_i \, ({}^i\dot{\mathbf{V}}_i + {}^i\mathbf{U}_i \, {}^i\mathbf{P}_j) + \sigma_j \, (\ddot{q}_j \, {}^j\mathbf{a}_j + 2 \, {}^j\boldsymbol{\omega}_i \times \dot{q}_j \, {}^j\mathbf{a}_j) \quad (21)$$

$${}^j\mathbf{F}_j = M_j \, {}^j\dot{\mathbf{V}}_j + {}^j\mathbf{U}_j \, {}^j\mathbf{MS}_j \quad (22)$$

$${}^j\mathbf{M}_j = {}^j\mathbf{J}_j \, {}^j\dot{\boldsymbol{\omega}}_j + {}^j\boldsymbol{\omega}_j \times ({}^j\mathbf{J}_j \, {}^j\boldsymbol{\omega}_j) + {}^j\mathbf{MS}_j \times {}^j\dot{\mathbf{V}}_j \quad (23)$$

with

$${}^j\mathbf{U}_j = {}^j\hat{\dot{\boldsymbol{\omega}}}_j + {}^j\hat{\boldsymbol{\omega}}_j \, {}^j\hat{\boldsymbol{\omega}}_j$$

and where $\mathbf{a}_j$ is the unit vector along the $\mathbf{z}_j$ axis which is the axis of joint j.

The matrix $\hat{\mathbf{w}}$ defines the 3×3 vector product matrix associated to the (3×1) vector $\mathbf{W}$ such that:

$$\hat{\mathbf{w}} = \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix} \quad (24)$$

$$\mathbf{w} \times \mathbf{v} = \hat{\mathbf{w}} \, \mathbf{v}$$

These equations are initialized by $\boldsymbol{\omega}_0 = \mathbf{0}$, $\dot{\boldsymbol{\omega}}_0 = \mathbf{0}$, $\dot{\mathbf{V}}_0 = -\mathbf{g}$, ${}^0\mathbf{U}_0 = 0$, with $\mathbf{g}$ is the acceleration of gravity.

Initialising the linear acceleration $\dot{\mathbf{V}}_0$ by $-\mathbf{g}$ will take automatically the effect of gravity forces on all the links of the structure.

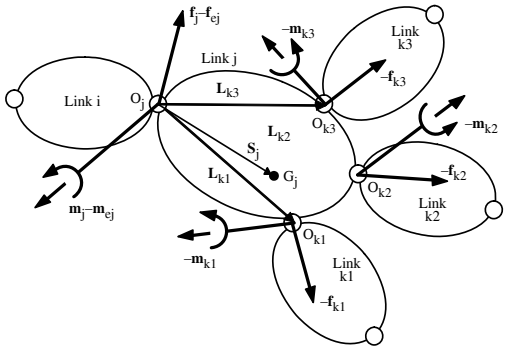

Figure 3: Forces and moments acting on a link j

The backward recursive equations are deduced from the resultant forces and moments on link j around the origin of link j (Figure 3).

$${}^j\mathbb{f}_j = {}^j\mathbb{F}_j + \sum_k {}^k\mathbb{T}_j^T \, {}^k\mathbb{f}_k + {}^j\mathbb{f}_{ej} \quad (25)$$

Where a(k)=j,

The backward equations can be calculated for j=n,…,1:

$${}^j\mathbf{f}_j = {}^j\mathbf{F}_j + {}^j\mathbf{f}_{ej} \quad (26)$$

$${}^j\mathbf{m}_j = {}^j\mathbf{M}_j + {}^j\mathbf{m}_{ej} \quad (27)$$

$${}^i\mathbf{f}_j = {}^i\mathbf{R}_j \, {}^j\mathbf{f}_j \quad (28)$$

$${}^i\mathbf{f}_{ei} = {}^i\mathbf{f}_{ei} + {}^i\mathbf{f}_j \quad (29)$$

$${}^i\mathbf{m}_{ei} = {}^i\mathbf{m}_{ei} + {}^i\mathbf{R}_j \, {}^j\mathbf{m}_j + {}^i\mathbf{P}_j \times {}^i\mathbf{f}_j \quad (30)$$

$$\Gamma_j = (\sigma_j \, {}^j\mathbf{f}_j + \bar{\sigma}_j \, {}^j\mathbf{m}_j)^T {}^j\mathbf{a}_j + Ia_j \, \ddot{q}_j + F_{sj} \, \mathrm{sign}(\dot{q}_j) + F_{vj} \, \dot{q}_j \quad (31)$$

Where:

$\mathbf{f}_j$ and $\mathbf{m}_j$ are the reaction forces and moments of link a(j) on link j respectively, Ia_j is the inertia of the rotor and transmission gears of the motor of joint j, F_{sj} and F_{vj} are the coulomb and viscous friction parameters respectively, ${}^j\mathbf{f}_{ej}$ and ${}^j\mathbf{m}_{ej}$ are the external forces and moments of link j on the environment.

This algorithm is easy to program numerically or symbolically. The computational cost is linear with 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 can be used instead of the standard inertial parameters and the

technique of customized symbolic method can be applied (Khosla 1986 , Khalil and Kleinfinger 1987, Khalil and Creusot 1997).

3.3 Computation of the direct dynamic model

The computation of the direct dynamic model is employed to carry out simulations for the purpose of testing the robot performances and studying the control laws. During simulation, the dynamic equations are solved for the joint accelerations given the input torques and the state of the robot (joint positions and velocities). Through integration of the joint accelerations, the robot trajectory is then determined.

The direct dynamic model can be obtained from Lagrange equation (8) as follows:

$$\ddot{\mathbf{q}} = \mathbf{A}^{-1} [\boldsymbol{\Gamma} - \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}})]$$

Two methods based on Newton-Euler methods can be used to obtain the dynamic model: the first is based on calculating the **A** and **H** matrices using Newton-Euler inverse dynamic model in order to calculate the joint accelerations by (8); the second method is based on a recursive Newton-Euler algorithm that does not explicitly calculate the matrix **A** and has a computational cost that varies linearly with the number of degrees of freedom of the robot. For tree structure robots, the second method is more efficient, but the first method can be used for closed loop robots and other complicated systems. That is why we will present both methods.

3.3.1 Using the inverse dynamic model to calculate the direct dynamic model

In this method the matrices $\mathbf{H}(\mathbf{q},\dot{\mathbf{q}})$ and $\mathbf{A}(\mathbf{q})$ are calculated using the inverse model by giving special values for the joint accelerations, joint velocities, external forces, friction, gravity (Walker and Orin 1982).

By comparing equations (7) and (12) we deduce that $\mathbf{H}(\mathbf{q}, \dot{\mathbf{q}})$ is equal to $\boldsymbol{\Gamma}$ if $\ddot{\mathbf{q}} = \mathbf{0}$, and that the i^{th} column of **A** is equal to $\boldsymbol{\Gamma}$ if:

$$\ddot{\mathbf{q}} = \mathbf{u}_i, \dot{\mathbf{q}} = \mathbf{0}, \mathbf{g} = \mathbf{0}, \mathbf{f}_{ej} = \mathbf{0}, \mathbf{m}_{ej} = \mathbf{0}$$

where $\mathbf{u}_i$ is the (nx1) unit vector whose i^{th} element is equal to 1, and the other elements are zeros. Iterating the procedure for i = 1,…, n leads to the construction of the entire inertia matrix.

To reduce the computational complexity of this algorithm, we can make use of the base inertial parameters and the customized symbolic techniques. Moreover, we can take advantage of the fact that the inertia matrix **A** is symmetric.

3.3.2 Recursive NE computation of the direct dynamic model

This method is based on the recursive Newton-Euler equations and does not use explicitly the inertia matrix of the robot (Armstrong 1979, (Featherstone 1983, (Brandl, Johanni and Otter, 1986).

Using (9) and (25) the equilibrium equations of link j can be written as:

$${}^{j}\mathbb{J}_j \, {}^{j}\dot{\mathbb{V}}_j = {}^{j}\mathbb{f}_j + {}^{j}\boldsymbol{\beta}_j - \sum_{k} {}^{k}\mathbb{T}_j^{T} \, {}^{k}\mathbb{f}_k \qquad (32)$$

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

$${}^{j}\boldsymbol{\beta}_j = -{}^{j}\mathbb{f}_{ej} - \begin{bmatrix} {}^{j}\boldsymbol{\omega}_j \times \left({}^{j}\boldsymbol{\omega}_j \times {}^{j}\mathbf{MS}_j \right) \\ {}^{j}\boldsymbol{\omega}_j \times \left({}^{j}\mathbf{J}_j \, {}^{j}\boldsymbol{\omega}_j \right) \end{bmatrix} \qquad (33)$$

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 ${}^{j}\mathbb{T}_i$, the link angular velocities ${}^{j}\boldsymbol{\omega}_j$ as well as ${}^{j}\boldsymbol{\gamma}_j$ and ${}^{j}\boldsymbol{\beta}_j$ vectors, which appear in the link accelerations and the link wrenches equations respectively when $\ddot{\mathbf{q}} = \mathbf{0}$;

$${}^{j}\boldsymbol{\gamma}_j = \begin{bmatrix} {}^{j}\mathbf{R}_i \left[{}^{i}\boldsymbol{\omega}_i \times \left({}^{i}\boldsymbol{\omega}_i \times {}^{i}\mathbf{P}_j \right) \right] + 2\sigma_j ({}^{j}\boldsymbol{\omega}_i \times \dot{\mathbf{q}}_j \, {}^{j}\mathbf{a}_j) \\ \bar{\sigma}_j \, {}^{j}\boldsymbol{\omega}_i \times \dot{\mathbf{q}}_j \, {}^{j}\mathbf{a}_j \end{bmatrix} \qquad (34)$$

$${}^{j}\boldsymbol{\beta}_j = -{}^{j}\mathbb{f}_{ej} - \begin{bmatrix} {}^{j}\boldsymbol{\omega}_j \times \left({}^{j}\boldsymbol{\omega}_j \times {}^{j}\mathbf{MS}_j \right) \\ {}^{j}\boldsymbol{\omega}_j \times \left({}^{j}\mathbf{J}_j \, {}^{j}\boldsymbol{\omega}_j \right) \end{bmatrix} \qquad (35)$$

ii) backward recursive computation: in this step we calculate the elements H_j, ${}^{j}\mathbb{J}_j^{*}$, ${}^{j}\boldsymbol{\beta}_j^{*}$, ${}^{j}\mathbb{K}_j$, ${}^{j}\boldsymbol{\alpha}_j$ which express $\ddot{q}_j$ and ${}^{j}\mathbb{f}_j^{*}$ in terms of ${}^{i}\dot{\mathbb{V}}_i$ in the third recursive equations. These equations are demonstrated in the following sub-section.

For j = n ... 1, compute:

$$H_j = ({}^{j}\mathbb{a}_j^{T} \, {}^{j}\mathbb{J}_j^{*} \, {}^{j}\mathbb{a}_j + Ia_j) \qquad (36)$$

$$ {}^{j}\mathbb{K}_j = {}^{j}\mathbb{J}_j^* - {}^{j}\mathbb{J}_j^* \, {}^{j}\mathbb{a}_j \, H_j^{-1} \, {}^{j}\mathbb{a}_j^T \, {}^{j}\mathbb{J}_j^* \quad (37) $$

$$ {}^{j}\boldsymbol{\alpha}_j = {}^{j}\mathbb{K}_j \, {}^{j}\boldsymbol{\gamma}_j + {}^{j}\mathbb{J}_j^* \, {}^{j}\mathbb{a}_j \, H_j^{-1} \, (\tau_j + {}^{j}\mathbb{a}_j^T \, {}^{j}\boldsymbol{\beta}_j^*) - {}^{j}\boldsymbol{\beta}_j^* \quad (38) $$

If a(j) ≠ 0, calculate also:

$$ {}^{i}\boldsymbol{\beta}_i^* = {}^{i}\boldsymbol{\beta}_i^* - {}^{j}\mathbb{T}_i^T \, {}^{j}\boldsymbol{\alpha}_j \quad (39) $$

$$ {}^{i}\mathbb{J}_i^* = {}^{i}\mathbb{J}_i^* + {}^{j}\mathbb{T}_i^T \, {}^{j}\mathbb{K}_j \, {}^{j}\mathbb{T}_i \quad (40) $$

These equations are initialized by ${}^{j}\mathbb{J}_j^* = {}^{j}\mathbb{J}_j$ and ${}^{j}\boldsymbol{\beta}_j^* = {}^{j}\boldsymbol{\beta}_j$.

iii) second forward recursive computations. Since the acceleration of the base is known ($\dot{\mathbf{V}}_0 = -\mathbf{g}$, $\dot{\boldsymbol{\omega}}_0 = \mathbf{0}$ for fixed base), the third recursive computation gives $\ddot{q}_j$ and ${}^{j}\mathbb{f}_j$ (if needed) for j = 1… n. as follows:

$$ \ddot{q}_j = H_j^{-1} \, [- {}^{j}\mathbb{a}_j^T \, {}^{j}\mathbb{J}_j^* \, ({}^{j}\mathbb{T}_i \, {}^{i}\dot{\mathbb{V}}_i + {}^{j}\boldsymbol{\gamma}_j) + \tau_j + {}^{j}\mathbb{a}_j^T \, {}^{j}\boldsymbol{\beta}_j^*] \quad (41) $$

$$ {}^{j}\mathbb{f}_j = \begin{bmatrix} {}^{j}\mathbf{f}_j \\ {}^{j}\mathbf{m}_j \end{bmatrix} = {}^{j}\mathbb{K}_j \, {}^{j}\mathbb{T}_i \, {}^{i}\dot{\mathbb{V}}_i + {}^{j}\boldsymbol{\alpha}_j \quad (42) $$

$$ {}^{j}\dot{\mathbb{V}}_j = {}^{j}\mathbb{T}_i \, {}^{i}\dot{\mathbb{V}}_i + {}^{j}\mathbb{a}_j \, \ddot{q}_j + {}^{j}\boldsymbol{\gamma}_j \quad (43) $$

where

$$ \tau_j = \Gamma_j - F_{sj} \, \mathrm{sign}(\dot{q}_j) - F_{vj} \, \dot{q}_j \quad (44) $$

Calculation of the elements of the backward recursive equations

To simplify the notations, we consider the case of a serial structure of n joints. Expressing the acceleration of link n in terms of the acceleration of link n-1, and since ${}^{n+1}\mathbb{f}_{n+1} = \mathbf{0}$, we obtain:

$$ {}^{n}\mathbb{J}_n \, ({}^{n}\mathbb{T}_{n-1} \, {}^{n-1}\dot{\mathbb{V}}_{n-1} + \ddot{q}_n \, {}^{n}\mathbb{a}_n + {}^{n}\boldsymbol{\gamma}_n) = {}^{n}\mathbb{f}_n + {}^{n}\boldsymbol{\beta}_n \quad (45) $$

Since:

$$ {}^{j}\mathbb{a}_j^T \, {}^{j}\mathbb{f}_j = \tau_j - Ia_j \, \ddot{q}_j $$

$$ \tau_j = \Gamma_j - F_{sj} \, \mathrm{sign}(\dot{q}_j) - F_{vj} \, \dot{q}_j $$

We obtain the joint acceleration of joint n:

$$ \ddot{q}_n = H_n^{-1} \, (- {}^{n}\mathbb{a}_n^T \, {}^{n}\mathbb{J}_n \, ({}^{n}\mathbb{T}_{n-1} \, {}^{n-1}\dot{\mathbb{V}}_{n-1} + {}^{n}\boldsymbol{\gamma}_n) + \tau_n + {}^{n}\mathbb{a}_n^T \, {}^{n}\boldsymbol{\beta}_n) \quad (46) $$

where H_n is a scalar given as:

$$ H_n = ({}^{n}\mathbb{a}_n^T \, {}^{n}\mathbb{J}_n \, {}^{n}\mathbb{a}_n + Ia_n) \quad (47) $$

Substituting for $\ddot{q}_n$ from (46) and (45), we obtain the dynamic wrench ${}^{n}\mathbb{f}_n$ as:

$$ {}^{n}\mathbb{f}_n = \begin{bmatrix} {}^{n}\mathbf{f}_n \\ {}^{n}\mathbf{m}_n \end{bmatrix} = {}^{n}\mathbb{K}_n \, {}^{n}\mathbb{T}_{n-1} \, {}^{n-1}\dot{\mathbb{V}}_{n-1} + {}^{n}\boldsymbol{\alpha}_n \quad (48) $$

where:

$$ {}^{n}\mathbb{K}_n = {}^{n}\mathbb{J}_n - {}^{n}\mathbb{J}_n \, {}^{n}\mathbb{a}_n \, H_n^{-1} \, {}^{n}\mathbb{a}_n^T \, {}^{n}\mathbb{J}_n \quad (49) $$

$$ {}^{n}\boldsymbol{\alpha}_n = {}^{n}\mathbb{K}_n \, {}^{n}\boldsymbol{\gamma}_n + {}^{n}\mathbb{J}_n \, {}^{n}\mathbb{a}_n \, H_n^{-1} \, (\tau_n + {}^{n}\mathbb{a}_n^T \, {}^{n}\boldsymbol{\beta}_n) - {}^{n}\boldsymbol{\beta}_n \quad (50) $$

We now have $\ddot{q}_n$ and ${}^{n}\mathbb{f}_n$ in terms of ${}^{n-1}\dot{\mathbb{V}}_{n-1}$. Iterating the procedure for j = n – 1, we obtain:

$$ {}^{n-1}\mathbb{J}_{n-1} \, {}^{n-1}\dot{\mathbb{V}}_{n-1} = {}^{n-1}\mathbb{f}_{n-1} + {}^{n}\mathbb{T}_{n-1}^T \, {}^{n}\mathbb{f}_n + {}^{n-1}\boldsymbol{\beta}_{n-1} \quad (51) $$

which can be rewritten as:

$$ {}^{n-1}\mathbb{J}_{n-1}^* \, ({}^{n-1}\mathbb{T}_{n-2} \, {}^{n-2}\dot{\mathbb{V}}_{n-2} + \ddot{q}_{n-1} \, {}^{n-1}\mathbb{a}_{n-1} + {}^{n-1}\boldsymbol{\gamma}_{n-1}) = {}^{n-1}\mathbb{f}_{n-1} + {}^{n-1}\boldsymbol{\beta}_{n-1}^* \quad (52) $$

where:

$$ {}^{n-1}\mathbb{J}_{n-1}^* = {}^{n-1}\mathbb{J}_{n-1} + {}^{n}\mathbb{T}_{n-1}^T \, {}^{n}\mathbb{K}_n \, {}^{n}\mathbb{T}_{n-1} \quad (53) $$

$$ {}^{n-1}\boldsymbol{\beta}_{n-1}^* = {}^{n-1}\boldsymbol{\beta}_{n-1} - {}^{n}\mathbb{T}_{n-1}^T \, {}^{n}\boldsymbol{\alpha}_n \quad (54) $$

Equation (52) has the same form as (45). Thus, we can express $\ddot{q}_{n-1}$ and ${}^{n-1}\mathbb{f}_{n-1}$ in terms of ${}^{n-2}\dot{\mathbb{V}}_{n-2}$. Iterating this procedure for j = n – 2, …, 1, we obtain $\ddot{q}_j$ and ${}^{j}\mathbb{f}_j$ in terms of ${}^{j-1}\dot{\mathbb{V}}_{j-1}$ for j = n – 1, ..., 1 as given by equations (41) and (43) which represent the general case.

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 $\boldsymbol{\Gamma}_c$ are obtained by projecting the tree structure torques $\boldsymbol{\Gamma}_{tr}$ 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).

$$ \boldsymbol{\Gamma}_c = \mathbf{G}^T \, \boldsymbol{\Gamma}_{tr}(\mathbf{q}_{tr}, \dot{\mathbf{q}}_{tr}, \ddot{\mathbf{q}}_{tr}) \quad (55) $$

where:

$$ \mathbf{G} = \frac{\partial \mathbf{q}_{tr}}{\partial \mathbf{q}_a} = \frac{\partial \dot{\mathbf{q}}_{tr}}{\partial \dot{\mathbf{q}}_a} \quad (56) $$

It can be written also as:

$$\boldsymbol{\Gamma}_c = \boldsymbol{\Gamma}_a + \frac{\partial \dot{\mathbf{q}}_p}{\partial \dot{\mathbf{q}}_a} \boldsymbol{\Gamma}_p \quad (57)$$

Where:

$\boldsymbol{\Gamma}_a$ and $\boldsymbol{\Gamma}_p$ are the torque of actuated and passive joints of the tree structure.

The kinematics Jacobian matrix can be obtained from (4) representing the kinematics closed loop constraints.

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 $\mathbf{A}_c$ and $\mathbf{H}_c$ of the following relation:

$$\boldsymbol{\Gamma}_c = \mathbf{A}_c(\mathbf{q}_{tr})\ddot{\mathbf{q}}_a + \mathbf{H}_c(\mathbf{q}_{tr}, \dot{\mathbf{q}}_{tr}) \quad (58)$$

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 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 will present her the case of parallel robots with six degrees of freedom. Examples concerning reduced mobility robots are given in (Khalil and Ibrahim 2007).

The robot is composed of a fixed base and a mobile platform. They are connected using m parallel legs.

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

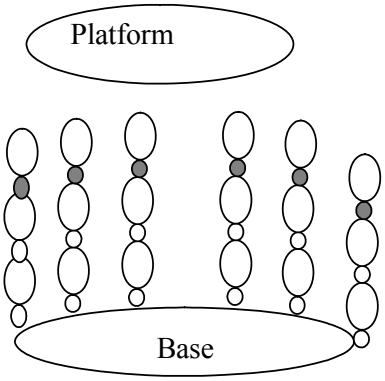

Figure 4: Parallel structure after separating the platform

To obtain the dynamic models of parallel robots, we exploit their structural characteristics by decomposing the system into two subsystems: the platform and the legs.

The dynamics of the platform is calculated as a function of the Cartesian variables (spatial Cartesian position, velocity and acceleration of the platform), whereas the dynamics of the legs are calculated as a function of the joint variables of the legs $(\mathbf{q}_i, \dot{\mathbf{q}}_i, \ddot{\mathbf{q}}_i)$ for i=1,…,m. The active joint torques are obtained by the sum of these dynamics and projecting them 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, which gives the platform screw $\mathbb{V}_p$ in terms of the motorized joint velocities $\dot{\mathbf{q}}_a$, and to project the leg dynamics on the active joint space we use the Jacobian between these two spaces. Thus the dynamic model of the parallel structure is given by the following equation:

$$\boldsymbol{\Gamma} = \mathbf{J}_P^T \mathbb{F}_P + \sum_{i=1}^{m} \left(\frac{\partial \dot{\mathbf{q}}_i}{\partial \dot{\mathbf{q}}_a} \right)^T \boldsymbol{\Gamma}_i \quad (59)$$

where

$\mathbb{F}_P$ is the total forces and moments on the platform,

$\mathbf{J}_P$ is the (6×n) kinematics Jacobian matrix of the robot, which gives the platform velocity $\mathbf{V}_P$ (translational and angular) as a function of the active joint velocities:

$$\mathbb{V}_P = \mathbf{J}_P \dot{\mathbf{q}}_a \quad (60)$$

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

In this section we suppose n=6, thus $\mathbf{J}_P$ is (6×6) matrix.

The calculation of $\mathbf{J}_p$ is obtained by inverting $\mathbf{J}_p^{-1}$, which is easy to obtain for most parallel structures.

$\mathbb{F}_P$ is calculated by the Newton-Euler equation (9).

The calculation of $\partial \dot{\mathbf{q}}_i / \partial \dot{\mathbf{q}}_a$ is carried out by the following relation, which exploits the parallel structure of the robot:

$$\frac{\partial \dot{\mathbf{q}}_i}{\partial \dot{\mathbf{q}}_a} = \frac{\partial \dot{\mathbf{q}}_i}{\partial \mathbf{v}_i} \frac{\partial \mathbf{v}_i}{\partial \mathbb{V}_P} \frac{\partial \mathbb{V}_P}{\partial \dot{\mathbf{q}}_a} \quad (61)$$

with: $\mathbf{v_i}$ is the Cartesian velocity transferred from leg i to the platform.

We can rewrite (61) as:

$$\frac{\partial \dot{\mathbf{q}}_i}{\partial \dot{\mathbf{q}}_a} = \mathbf{J}_i^{-1}\,\mathbf{J}_{vi}\,\mathbf{J}_r \qquad (62)$$

$\mathbf{J_i}$ is the kinematic Jacobian matrix of leg i such that:

$$\mathbf{v_i} = \mathbf{J_i}\,\dot{\mathbf{q}}_i \qquad (63)$$

$\mathbf{J_{vi}}$ gives $\mathbf{v_i}$ as a function of $\mathbb{V}_P$:

$$\mathbf{v_i} = \mathbf{J_{vi}}\,\mathbb{V}_P \qquad (64)$$

For the Gough-Stewart platform (where the mobile platform is connected to the legs using spherical joints), we obtain:

$$\mathbf{J_{vi}} = \frac{\partial \mathbf{v_i}}{\partial \mathbb{V}_P} = \begin{bmatrix} \mathbf{I}_3 & -\hat{\mathbf{P}}_i \end{bmatrix} \qquad (65)$$

Where P_i is the vector between the origin of the platform frame and the centre of the spherical joint linking the platform with leg i.

Finally the inverse dynamic model of the robot is given by the following form:

$$\mathbf{\Gamma} = \mathbf{J_p^T}\left[\mathbb{F}_P + \sum_{i=1}^{m} \mathbf{J_{vi}^T}\,\mathbf{J_i^{-T}}\mathbf{\Gamma}_i\right] \qquad (66)$$

We note that the term between the brackets in (66) represents the dynamic model of the robot expressed in the Cartesian space of the platform frame (Khalil and Guegain, 2002).

6. INVERSE DYNAMIC MODELING OF ROBOTS WITH ELASTIC JOINTS

In this section we tree structure robots with lumped elasticity or flexible joints. The system can be described using Modified Denavit and Hartenberg method presented in section 2. Each joint could be either elastic or rigid (Khalil and Gautier, 2000).

6.1 Lagrange dynamic form

The general form of the dynamic model of a system with flexible joints has the same form as (7). It can be rewritten as:

$$\mathbf{\Gamma} = \mathbf{A(q)}\,\ddot{\mathbf{q}} + \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}}) \qquad (67)$$

It can be partitioned as follows:

$$\Gamma = \begin{bmatrix} \Gamma_r \\ \Gamma_f \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{12}^T & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_r \\ \ddot{\mathbf{q}}_f \end{bmatrix} + \begin{bmatrix} \mathbf{H}_r \\ \mathbf{H}_f \end{bmatrix} \qquad (68)$$

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

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

$\mathbf{A}(\mathbf{q})$ is the (n×n) inertia matrix of the system,

$\mathbf{\Gamma}_r$ is the vector of rigid joint torques,

$\mathbf{\Gamma}_r$ is the vector of elastic joint torques.

If joint j is flexible:

$$\Gamma_j = -\,\Delta q_j\, K_j \qquad (69)$$

where K_j is the stiffness of the elastic joint,

$$\Delta q_j = q_j - q_{0j} \qquad (70)$$

q_{0j} is the joint position corresponding to zero elasticity force.

In the case of a system with elasticity, the direct dynamic model has the same outputs as in the case of rigid bodies; it gives the joint accelerations as a function of the joint torques and of the system state variables ($\mathbf{q}$, $\dot{\mathbf{q}}$). It can be calculated using (68) by calculation the inverse of $\mathbf{A}$.

In the case of a system with elasticity, the inverse dynamic model calculates the input torques and the elastic accelerations as a function of the joint positions, velocities and rigid joint accelerations. It is to be noted that the accelerations of the elastic variables cannot be specified independently. Using (68) to calculate the inverse model, we have first to calculate the acceleration elastic accelerations from the second row:

$$\Gamma_f = \begin{bmatrix} \mathbf{A}_{12}^T & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_r \\ \ddot{\mathbf{q}}_f \end{bmatrix} + \mathbf{H}_f \qquad (71)$$

then we can 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.

Remark: We note that in case of rigid non motorized joint, the same algorithm can be used after putting $\Gamma_j = 0$.

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. On the contrary 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 6.1 for the calculation of $\ddot{\mathbf{q}}_f$ and $\mathbf{\Gamma}_r$.

We propose here a recursive algorithm to solve this problem (Khalil and Gautier 2000). This algorithm consists of three recursive steps.

i) The first forward iteration is exactly 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 $\ddot{q}_j$ and ${}^j\mathbb{f}_j$ as a function of ${}^{a(j)}\dot{\mathbb{V}}_{a(j)}$. These matrices can be defined using a similar procedure as in section (3.3). They can be calculated for j =n, ...,1, as follows:

- If joint j is elastic:

$$H_j = {}^j\mathbb{a}_j^T\, {}^j\mathbb{J}_j^*\, {}^j\mathbb{a}_j \tag{72}$$

$${}^j\mathbb{K}_j = {}^j\mathbb{J}_j^* - {}^j\mathbb{J}_j^*\, {}^j\mathbb{a}_j\, H_j^{-1}\, {}^j\mathbb{a}_j^T\, {}^j\mathbb{J}_j^* \tag{73}$$

$${}^j\boldsymbol{\alpha}_j = {}^j\mathbb{K}_j\, {}^j\boldsymbol{\gamma}_j + {}^j\mathbb{J}_j^*\, {}^j\mathbb{a}_j\, H_j^{-1}\, (-K_j\, \Delta q_j + {}^j\mathbb{a}_j^T\, {}^j\boldsymbol{\beta}_j^*) - {}^j\boldsymbol{\beta}_j^* \tag{74}$$

- If joint j is rigid:

$${}^j\mathbb{K}_j = {}^j\mathbb{J}_j^* \tag{75}$$

$${}^j\boldsymbol{\alpha}_j = {}^j\mathbb{K}_j\, {}^j\boldsymbol{\gamma}_j + {}^j\mathbb{J}_j^*\, {}^j\mathbb{a}_j\, \ddot{q}_j - {}^j\boldsymbol{\beta}_j^* \tag{76}$$

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

$${}^i\boldsymbol{\beta}_i^* = {}^i\boldsymbol{\beta}_i^* - {}^j\mathbb{T}_i^T\, {}^j\boldsymbol{\alpha}_j \tag{77}$$

$${}^i\mathbb{J}_i^* = {}^i\mathbb{J}_i^* + {}^j\mathbb{T}_i^T\, {}^j\mathbb{K}_j\, {}^j\mathbb{T}_i \tag{78}$$

The previous equations are initialized by:

${}^j\mathbb{J}_j^* = {}^j\mathbb{J}_j$, and ${}^j\boldsymbol{\beta}_j^* = {}^j\boldsymbol{\beta}_j$.

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 equation:

$${}^j\mathbb{f}_j = \begin{bmatrix} {}^j\mathbf{f}_j \\ {}^j\mathbf{m}_j \end{bmatrix} = {}^j\mathbb{K}_j\, {}^j\mathbb{T}_i\, {}^i\dot{\mathbb{V}}_i + {}^j\boldsymbol{\alpha}_j \tag{79}$$

- if j is elastic:

$$\ddot{q}_j = H_j^{-1}\,[-\,{}^j\mathbb{a}_j^T\, {}^j\mathbb{J}_j^*\, ({}^j\mathbb{T}_i\, {}^i\dot{\mathbb{V}}_i + {}^j\boldsymbol{\gamma}_j) - K_j\, \Delta q_j + {}^j\mathbb{a}_j^T\, {}^j\boldsymbol{\beta}_j^*] \tag{80}$$

$${}^j\dot{\mathbb{V}}_j = {}^j\mathbb{T}_i\, {}^i\dot{\mathbb{V}}_i + {}^j\mathbb{a}_j\, \ddot{q}_j + {}^j\boldsymbol{\gamma}_j \tag{81}$$

- if j is rigid

$$\mathbf{\Gamma}_j = (\sigma_j\, {}^j\mathbf{f}_j + \bar{\sigma}_j\, {}^j\mathbf{m}_j)^T\, {}^j\mathbf{a}_j + Ia_j\, \ddot{q}_j \tag{82}$$

7. DYNAMIC MODELING OF ROBOTS WITH MOVING BASE

The structure treated in this section includes a big number of systems such as: cars, mobile robots, mobile manipulators, walking robots, Humanoid robots, eel like robots (Khalil W., G. Gallot G., Boyer F., 2007), snakes like robots, flying robots, spatial vehicle, 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 base is fixed thus the acceleration of the base is equal to zero, whereas in the case of a mobile base system the acceleration of the base must be determined in both direct and inverse dynamic models. The proposed recursive dynamic models are easy to implement and calculate using numerical calculation. 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 ${}^w\mathbf{T}_0$. 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 $\mathbb{V}_0$ and $\dot{\mathbb{V}}_0$ respectively.

The Cartesian velocities and accelerations of the links are calculated using the recursive equations (13)-(17).

7.1 General form of the dynamic models

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

$$\begin{bmatrix} \mathbf{0}_{6x1} \\ \Gamma \end{bmatrix} = \mathbf{A} \begin{bmatrix} {}^0\dot{\mathbb{V}}_0 \\ \ddot{\mathbf{q}} \end{bmatrix} + \mathbf{H} \qquad (83)$$

$\mathbf{\Gamma}$ $(n \times 1)$ vector of joint torques,

$\mathbf{q}$ $(n \times 1)$ vector of joint positions,

$\mathbf{A}$ is the $(6+n)\times(6+n)$ inertia matrix of the robot, it can be partitioned as follows:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{12}^T & \mathbf{A}_{22} \end{bmatrix} \qquad (84)$$

$\mathbf{A}_{11}$ is the (6×6) inertia matrix of the composed link 0, which is composed of the inertia of all the links referred to frame R_0 (the base).

$\mathbf{A}_{22}$ is the $(n \times n)$ inertia matrix of the other links when the head is fixed,

$\mathbf{A}_{12}$ is the $(6\times n)$ coupled inertia matrix of the joints and the base. It reflects the effect of the joint accelerations on the base motion, and the dual effect of base accelerations on the joint motions.

$\mathbf{H}$ is the $(n+6)\times 1$ vector representing the Coriolis, centrifugal, gravity and external forces effect on the robot. Its elements are functions of the base and joint velocities and the external forces. This vector can be partitioned as follows:

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \end{bmatrix} \qquad (85)$$

where:

$\mathbf{H}_1$ the Coriolis, centrifugal, gravity and external forces on the base.

$\mathbf{H}_2$ the Coriolis, centrifugal, gravity and external forces on the links 1,…,n.

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. Using equation (83) and (84), the inverse dynamic model is solved by using the first row of equation (83) to obtain the base acceleration:

$${}^0\dot{\mathbb{V}}_0 = -\left(\mathbf{A}_{11}\right)^{-1}\left(\mathbf{H}_1 + \mathbf{A}_{12}\ddot{\mathbf{q}}\right) \qquad (86)$$

Then the second row of (83), can be used to find the joint torques:

$$\Gamma = \mathbf{A}_{12}^T\,\dot{\mathbb{V}}_0 + \mathbf{A}_{22}\,\ddot{\mathbf{q}} + \mathbf{H}_2 \qquad (87)$$

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 (83), the direct dynamic model is solved as follows:

$$\begin{bmatrix} \dot{\mathbb{V}}_0 \\ \ddot{\mathbf{q}} \end{bmatrix} = \mathbf{A}^{-1}\begin{bmatrix} -\mathbf{H}_1 \\ \Gamma - \mathbf{H}_2 \end{bmatrix} \qquad (88)$$

The calculation of $\mathbf{A}$ and $\mathbf{H}$ 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 base can be taken into account by either of the following methods:

- The velocity and acceleration of the base will be the initial conditions $\dot{\mathbb{V}}_0$ and $\boldsymbol{\omega}_0$ for the forward recursive calculation. The backward recursive calculation must continue to j=0, where this new iteration will obtain the 6 equations of Newton-Euler equations of the base.
- We can assign link 1 to be the base, and suppose that link 0 is a virtual link whose inertial parameters are equal to zero but has the velocity and acceleration of the base. This can be done by putting σ_2=2. The six equations of the base will be those of $f_1 = 0$;

Solving the inverse and direct dynamic problems using $\mathbf{A}$ and $\mathbf{H}$ may be 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 wrt the number of degrees of freedom.

The recursive Newton-Euler algorithm is based on the kinematic equations presented in section 3.

7.2 Recursive NE calculation of the inverse dynamic model of robots with mobile base

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

i) Forward recursive calculation:

In this step we calculate the screw transformation matrices, link velocities, and the elements of the accelerations and external wrenches on the links, which are independent of the acceleration of the robot base ($\dot{\mathbf{V}}_0, \dot{\boldsymbol{\omega}}_0$). Thus we calculate for j=1,…,n: ${}^j\mathbb{T}_i$, ${}^j\mathbb{V}_j$ and ${}^j\boldsymbol{\gamma}_j$ using equations (13)-(17). We calculate also ${}^j\boldsymbol{\beta}_j$ representing the elements of the Newton-Euler

equations, which are independent of the base acceleration in equations (14) and (15) such that:

$$ {}^{j}\boldsymbol{\zeta}_j = {}^{j}\boldsymbol{\gamma}_j + \ddot{q}_j \, {}^{j}\mathbf{a}_j \tag{89} $$

$$ {}^{j}\boldsymbol{\beta}_j = -{}^{j}\mathbf{f}_{ej} - \begin{bmatrix} {}^{j}\boldsymbol{\omega}_j \times \left({}^{j}\boldsymbol{\omega}_j \times {}^{j}\mathbf{MS}_j \right) \\ {}^{j}\boldsymbol{\omega}_j \times \left({}^{j}\mathbf{J}_j \, {}^{j}\boldsymbol{\omega}_j \right) \end{bmatrix} \tag{90} $$

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 (32), giving the equilibrium equation of link j, can be rewritten using (90) as:

$$ {}^{j}\mathbf{f}_j = {}^{j}\mathbb{J}_j \, {}^{j}\dot{\mathbb{V}}_j - {}^{j}\boldsymbol{\beta}_j + \sum_{k} {}^{k}\mathbb{T}_j^{T} \, {}^{k}\mathbf{f}_k \tag{91} $$

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

$$ {}^{j}\mathbf{f}_j = {}^{j}\mathbb{J}_j \, {}^{j}\dot{\mathbb{V}}_j - {}^{j}\boldsymbol{\beta}_j + \sum_{s(j)} {}^{s(j)}\mathbb{T}_j^{T} \left({}^{s(j)}\mathbb{J}_{s(k)} \, {}^{s(j)}\dot{\mathbb{V}}_{s(j)} - {}^{s(j)}\boldsymbol{\beta}_{s(j)} \right) \tag{92} $$

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

Substituting for ${}^{s(j)}\dot{\mathbb{V}}_{s(j)}$ in terms of ${}^{j}\dot{\mathbb{V}}_j$ using (14), we obtain:

$$ {}^{s(j)}\dot{\mathbb{V}}_{s(j)} = {}^{s(j)}\mathbb{T}_j \, {}^{j}\dot{\mathbb{V}}_j + \sum_{r} {}^{s(j)}\mathbb{T}_r \, {}^{r}\boldsymbol{\zeta}_r \tag{93} $$

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

From (92), we obtain:

$$ {}^{j}\mathbf{f}_j = {}^{j}\mathbb{J}_j^{c} \, {}^{j}\dot{\mathbb{V}}_j - {}^{j}\boldsymbol{\beta}_j^{c} \tag{94} $$

with:

$$ {}^{j}\mathbb{J}_j^{c} = {}^{j}\mathbb{J}_j^{c} + \sum_{k} {}^{k}\mathbb{T}_j^{T} \, {}^{k}\mathbb{J}_k^{c} \, {}^{k}\mathbb{T}_j \tag{95} $$

$$ {}^{j}\boldsymbol{\beta}_j^{c} = {}^{j}\boldsymbol{\beta}_j^{c} - \sum_{k} {}^{k}\mathbb{T}_j^{T} \, {}^{k}\boldsymbol{\beta}_k^{c} + {}^{k}\mathbb{T}_j^{T} \, {}^{k}\mathbb{J}_k^{c} \, {}^{k}\boldsymbol{\zeta}_k \tag{96} $$

${}^{j}\mathbb{J}_j^{c}$ is the inertial matrix of the composite link j.

For j = 0, and supposing ${}^{0}\mathbf{f}_0$ is equal to zero, we obtain using (94):

$$ {}^{0}\dot{\mathbb{V}}_0 = \left({}^{0}\mathbb{J}_0^{c} \right)^{-1} {}^{0}\boldsymbol{\beta}_0^{c} \tag{97} $$

To conclude, the recursive equations of this step consist of initialising ${}^{n}\mathbb{J}_n^{c} = {}^{n}\mathbb{J}_n$, ${}^{n}\boldsymbol{\beta}_n^{c} = {}^{n}\boldsymbol{\beta}_n$ and then calculating (95)-(96) for j = n,…, 0. At the end ${}^{0}\dot{\mathbb{V}}_0$ is calculated by (97).

Comparing (97) with (68) we can deduce that $\mathbf{A}_{11}$ is equal to ${}^{0}\mathbb{J}_0^{c}$, whereas ${}^{0}\boldsymbol{\beta}_0^{c}$ is equal to $(\mathbf{H}_1 + \mathbf{A}_{12}\ddot{\mathbf{q}})$.

iii) Forward recursive equations:

After calculating ${}^{0}\dot{\mathbb{V}}_0$, the wrench ${}^{j}\mathbf{f}_j$ and the joint torques are obtained using equations (6) and (22) for j= 1,…, n as:

$$ {}^{j}\dot{\mathbb{V}}_j = {}^{j}\mathbb{T}_i \, {}^{i}\dot{\mathbb{V}}_i + {}^{j}\boldsymbol{\zeta}_j \tag{98} $$

$$ {}^{j}\mathbf{f}_j = \begin{bmatrix} {}^{j}\mathbf{f}_j \\ {}^{j}\mathbf{m}_j \end{bmatrix} = {}^{j}\mathbb{J}_j^{c} \, {}^{j}\dot{\mathbb{V}}_j - {}^{j}\boldsymbol{\beta}_j^{c} \tag{99} $$

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

$$ \Gamma_j = {}^{j}\mathbf{f}_j^{T} \, {}^{j}\mathbf{a}_j + F_{sj} \, \mathrm{sign}\left(\dot{q}_j \right) + F_{vj} \, \dot{q}_j + I_{aj} \, \ddot{q}_j \tag{100} $$

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 in the same order as those of the inverse dynamic model (forward, backward and forward):

i) Forward recursive equations:

We calculate the link Cartesian velocities using (13) and the terms of Cartesian accelerations and equilibrium equations of the links that are independent of the accelerations of the base and of the joints. We calculate the following recursive equations for j = 1, ..., n:

$$ {}^{j}\boldsymbol{\gamma}_{j} = \begin{bmatrix} {}^{j}\mathbf{R}_{i}\left[{}^{i}\boldsymbol{\omega}_{i} \times \left({}^{i}\boldsymbol{\omega}_{i} \times {}^{i}\mathbf{P}_{j} \right) \right] + 2\sigma_{j}({}^{j}\boldsymbol{\omega}_{i} \times \dot{q}_{j}\, {}^{j}\mathbf{a}_{j}) \\ \bar{\sigma}_{j}\, {}^{j}\boldsymbol{\omega}_{i} \times \dot{q}_{j}\, {}^{j}\mathbf{a}_{j} \end{bmatrix} \quad (101) $$

$$ {}^{j}\boldsymbol{\beta}_{j} = -{}^{j}f_{ej} - \begin{bmatrix} {}^{j}\boldsymbol{\omega}_{j} \times \left({}^{j}\boldsymbol{\omega}_{j} \times {}^{j}\mathbf{MS}_{j} \right) \\ {}^{j}\boldsymbol{\omega}_{j} \times \left({}^{j}\mathbf{J}_{j}\, {}^{j}\boldsymbol{\omega}_{j} \right) \end{bmatrix} \quad (102) $$

ii) Backward recursive equations:

In this second step, we first initialise ${}^{n}\mathbb{J}^{*}_{n} = {}^{n}\mathbb{J}_{n}$, ${}^{n}\boldsymbol{\beta}^{*}_{n} = {}^{n}\boldsymbol{\beta}_{n}$ 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 ${}^{i}\dot{\mathbb{V}}_{i}$ and will be used in the third recursive equations (these matrices can be obtained using a similar procedure as for the direct dynamic model of rigid links):

$$ H_{j} = {}^{j}a_{j}^{T}\, {}^{j}\mathbb{J}^{*}_{j}\, {}^{j}a_{j} + Ia_{j} \quad (103) $$

$$ {}^{j}\mathbb{K}_{j} = {}^{j}\mathbb{J}^{*}_{j} - {}^{j}\mathbb{J}^{*}_{j}\, {}^{j}a_{j} H_{j}^{-1}\, {}^{j}a_{j}^{T}\, {}^{j}\mathbb{J}^{*}_{j} \quad (104) $$

$$ {}^{i}\mathbb{J}^{*}_{i} = {}^{i}\mathbb{J}_{i} + {}^{j}\mathbb{T}_{i}^{T}\, {}^{j}\mathbb{K}_{j}\, {}^{j}\mathbb{T}_{i} \quad (105) $$

$$ \tau_{j} = \Gamma_{j} - F_{sj}\, \mathrm{sign}(\dot{q}_{j}) - F_{vj}\, \dot{q}_{j} \quad (106) $$

$$ {}^{j}\boldsymbol{\alpha}_{j} = {}^{j}\mathbb{K}_{j}\, {}^{j}\boldsymbol{\gamma}_{j} + {}^{j}\mathbb{J}^{*}_{j}\, {}^{j}a_{j} H_{j}^{-1} \left(\tau_{j} + {}^{j}a_{j}^{T}\, {}^{j}\boldsymbol{\beta}^{*}_{j} \right) - {}^{j}\boldsymbol{\beta}^{*}_{j} \quad (107) $$

$$ {}^{i}\boldsymbol{\beta}^{*}_{i} = {}^{i}\boldsymbol{\beta}_{i} - {}^{j}\mathbb{T}_{i}^{T}\, {}^{j}\boldsymbol{\alpha}_{j} \quad (108) $$

iii) Forward recursive equations:

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

$$ {}^{0}\dot{\mathbb{V}}_{0} = \left({}^{0}\mathbb{J}^{*}_{0} \right)^{-1}\, {}^{0}\boldsymbol{\beta}^{*}_{0} \quad (109) $$

We note that ${}^{0}\boldsymbol{\beta}^{*}_{0}$ is a function of $\boldsymbol{\tau}$, whereas ${}^{0}\boldsymbol{\beta}^{c}_{0}$ (used in the inverse model) is a function of $\ddot{\mathbf{q}}$.

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

$$ \ddot{q}_{j} = H_{j}^{-1} \left[-{}^{j}a_{j}^{T}\, {}^{j}\mathbb{J}^{*}_{j} \left({}^{j}\dot{\mathbb{V}}_{j-1} + {}^{j}\boldsymbol{\gamma}_{j} \right) + \tau_{j} + {}^{j}a_{j}^{T}\, {}^{j}\boldsymbol{\beta}^{*}_{j} \right] \quad (110) $$

$$ {}^{j}f_{j} = {}^{j}\mathbb{K}_{j}\, {}^{j}\mathbb{T}_{i}\, {}^{i}\dot{\mathbb{V}}_{i} + {}^{j}\alpha_{j} \quad (111) $$

where:

$$ {}^{j}\dot{\mathbb{V}}_{j} = {}^{j}\dot{\mathbb{V}}_{j-1} + {}^{j}a_{j}\ddot{q}_{j} + {}^{j}\boldsymbol{\gamma}_{j} \quad (112) $$

8. CONCLUSIONS

This paper presents the inverse and direct dynamic modeling of different robotics systems. 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 manipulators to the other cases.

The proposed methods have been applied on more complicated systems such as:

- flexible link robots (Boyer and Khalil, 1998),
- Micro continuous system (Boyer, Porez and Khalil, 2006),
- hybrid structure, where the robot is composed of parallel modules, which are connected in serie, (Ibrahim, Khalil 2010).

REFERENCES

Angeles J. 2002. *Fundamentals of Robotic Mechanical Systems*. Second edition, Springer-Verlag, New York.

Armstrong W.W., 1979. Recursive solution to the equation of motion of an N-links manipulator. In *Proc. 5th World Congress on Theory of Machines and Mechanisms*, p. 1343-1346.

Boyer, F., Khalil,W, 1998. An efficient calculation of flexible manipulator inverse dynamic. In, *Int. Journal of Robotics Research,* vol. 17, No.3, pp.282-293

Boyer, F., Porez M., Khalil, W. 2006. Macro-continuous torque algorithm for a three-dimensional eel-like robot. In *IEEE Robotics transaction*, vol.22, No.4, 2006, pp.763-775.

Brandl H., Johanni R., Otter M., .1986. 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, p. 365-370.

Craig J.J., 1986. *Introduction to robotics: mechanics and control*. Addison Wesley Publishing Company, Reading.

Featherstone R., 1983. The calculation of robot dynamics using articulated-body inertias. In the *Int. J. of Robotics Research*, Vol. 2(3), p. 87-101.Gautier Gautier,M., Khalil W. 1990. Direct calculation of minimum set of inertial parameters of serial robots. In *IEEE Trans. on Robotics and Automation*, Vol. RA-6(3), p. 368-373.

Ibrahim, O., Khalil, W, 2010. Inverse and direct dynamic models of Hybride robots. In *Mechanism and machine theory,* Volume 45, Issue 4, p. 627-640.

Luh J.Y.S., Walker M.W., Paul R.C.P.,1980. On-line computational scheme for mechanical manipulators. In *Trans. of ASME, J. of Dynamic Systems, Measurement, and Control*, Vol. 102(2) p. 69-76.

Khalil W., Kleinfinger J.-F., 1986. A new geometric notation for open and closed-loop robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, San Francisco, p. 1174-1180.

Khalil W., Kleinfinger J.-F., 1987. Minimum operations and minimum parameters of the dynamic model of tree structure robots. In *IEEE J. of Robotics and Automation*, Vol. RA-3(6), p. 517-526.

Khalil W., Bennis F., 1994. Comments on Direct Calculation of Minimum Set of Inertial Parameters of Serial Robots. In *IEEE Trans. on Rob.& Automation*, Vol. RA-10(1), p. 78-79.

Khalil W., Creusot D., 1997. SYMORO+: a system for the symbolic modelling of robots. In *Robotica*, Vol. 15, p. 153-161.

Khalil W., Gautier M., 2000. Modeling of mechanical systems with lumped elasticity", In *Proc. IEEE Int. Conf. on Robotics and Automation*, San Francisco, p. 3965-3970.

Khalil, W., Dombre, E. 2002. *Modeling identification and control of robots*. Hermes, Penton-Sciences, London.

Khalil W. and Guegan S.,2004. Inverse and Direct Dynamic Modeling of Gough-Stewart Robots. In *IEEE Transactions on Robotics and Automation*, 20(4), p. 754-762.

Khalil W., G. Gallot G., Boyer F., 2007. Dynamic Modeling and Simulation of a 3-D Serial Eel-Like Robot. In *IEEE Transactions on Systems, Man and Cybernetics, Part C: Application and reviews, Vol. 37, N° 6.*

Khalil W., Ibrahim O., 2007**.** General solution for the Dynamic modeling of parallel robots. In *Journal of Intelligent and Robotic Systems*, Vol.49, pp.19-37.

Khosla P.K., 1986. Real-time control and identification of direct drive manipulators. Ph. D. Thesis, Carnegie Mellon.

Walker M.W., Orin D.E., 1982. Efficient dynamic computer simulation of robotics mechanism. In *Trans. of ASME, J. of Dynamic Systems, Measurement, and Control*, Vol. 104, p. 205-211.