Chapter 9

Dynamic modeling of serial robots

9.1. Introduction

The *inverse dynamic model* provides the joint torques and forces in terms of the joint positions, velocities and accelerations. It is described by:

$$\Gamma = \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbb{f}_e) \qquad [9.1]$$

with:

- Γ: vector of joint torques or forces, depending on whether the joint is revolute or prismatic respectively. In the sequel, we will only write *joint torques*;
- $\mathbf{q}$: vector of joint positions;
- $\dot{\mathbf{q}}$: vector of joint velocities;
- $\ddot{\mathbf{q}}$: vector of joint accelerations;
- $\mathbb{f}_e$: vector of forces and moments exerted by the robot on the environment.

Equation [9.1] is an inverse dynamic model because it defines the system input as a function of the output variables. It is often called the *dynamic model*.

The *direct dynamic model* describes the joint accelerations in terms of the joint positions, velocities and torques. It is represented by the relation:

$$\ddot{\mathbf{q}} = \mathbf{g}(\mathbf{q}, \dot{\mathbf{q}}, \Gamma, \mathbb{f}_e) \qquad [9.2]$$

The dynamic model of robots plays an important role in their design and operation. For robot design, the inverse dynamic model can be used to select the actuators [Chedmail 90b], [Potkonjak 86], while the direct dynamic model is

employed to carry out simulations (§ 9.7) for the purpose of testing the performance of the robot and to study the relative merits of possible control schemes. Regarding robot operations, the inverse dynamic model is used to compute the actuator torques, which are needed to achieve a desired motion (Chapter 14). It is also used to identify the dynamic parameters that are necessary for both control and simulation applications (Chapter 12).

Several approaches have been proposed to model the dynamics of robots [Renaud 75], [Coiffet 81], [Vukobratovic 82]. The most frequently employed in robotics are the Lagrange formulation [Uicker 69], [Khalil 76], [Renaud 80a], [Hollerbach 80], [Paul 81], [Megahed 84], [Renaud 85] and the Newton-Euler formulation [Hooker 65], [Armstrong 79], [Luh 80b], [Orin 79], [Khalil 85a], [Khosla 86], [Khalil 87b], [Renaud 87].

In this chapter, we present the dynamic modeling of serial robots using these two formulations. The problem of calculating a minimum set of inertial parameters will be covered in detail. We will focus our study on the minimization of the number of operations of the dynamic model in view of its real time computation for control purposes. Lastly, the computation of the direct dynamic model will be addressed. In Chapter 10, these results will be generalized for tree structured and closed chain robots.

9.2. Notations

The main notations used in this chapter are compiled below:

- $\mathbf{a}_j$ unit vector along axis $\mathbf{z}_j$;
- $\mathbf{F}_j$ external forces on link j;
- $\mathbf{f}_j$ force exerted on link j by link j – 1;
- $\mathbf{f}_{ej}$ force exerted by link j on the environment;
- F_{cj} parameter of Coulomb friction acting at joint j;
- F_{vj} parameter of viscous friction acting at joint j;
- $\mathbf{g}$ gravitational acceleration;
- G_j center-of-mass of link j;
- $\mathbf{I}_{Gj}$ inertia tensor of link j about G_j and with respect to a frame parallel to frame R_j;
- Ia_j moment of inertia of the rotor and the transmission system of actuator j referred to the joint side;
- ${}^j\mathbf{J}_j$ inertia tensor of link j with respect to frame R_j. It is described by:

$$ {}^{j}\mathbf{J}_j = \begin{bmatrix} \int(y^2+z^2)dm & -\int xydm & -\int xzdm \\ -\int xydm & \int(x^2+z^2)dm & -\int yzdm \\ -\int xzdm & -\int yzdm & \int(x^2+y^2)dm \end{bmatrix} = \begin{bmatrix} XX_j & XY_j & XZ_j \\ XY_j & YY_j & YZ_j \\ XZ_j & YZ_j & ZZ_j \end{bmatrix} \qquad [9.3] $$

$\mathbb{J}_j$ (6x6) spatial inertia matrix of link j (relation [9.21]);
$\mathbf{L}_j$ position vector between O_{j-1} and O_j;
M_j mass of link j;
$\mathbf{MS}_j$ first moments of link j with respect to frame R_j, equal to $M_j\ \mathbf{S}_j$. The components of ${}^{j}\mathbf{MS}_j$ are denoted by $[\ MX_j\ \ MY_j\ \ MZ_j\ ]^T$;
$\mathbf{M}_{Gj}$ moment of external forces on link j about G_j;
$\mathbf{M}_j$ moment of external forces on link j about O_j;
$\mathbf{m}_j$ moment about O_j exerted on link j by link j – 1;
$\mathbf{m}_{ej}$ moment about O_j exerted by link j on the environment;
$\mathbf{S}_j$ vector of the center-of-mass coordinates of link j. It is equal to $\mathbf{O_jG_j}$;
$\mathbf{V}_j$ linear velocity of O_j;
$\mathbb{V}_j$ (6x1) kinematic screw vector of link j, formed by the components of $\mathbf{V}_j$ and $\boldsymbol{\omega}_j$;
$\dot{\mathbf{V}}_j$ linear acceleration of O_j;
$\mathbf{V}_{Gj}$ linear velocity of the center-of-mass of link j;
$\dot{\mathbf{V}}_{Gj}$ linear acceleration of the center-of-mass of link j;
$\boldsymbol{\omega}_j$ angular velocity of link j;
$\dot{\boldsymbol{\omega}}_j$ angular acceleration of link j.

9.3. Lagrange formulation

9.3.1. *Introduction*

The dynamic model of a robot with several degrees of freedom represents a complicated system. The Newton-Euler method developed in § 9.5 presents an efficient and systematic approach to solving this problem. In this section, we develop a simple Lagrange method to present the general form of the dynamic model of robots and to get an insight into its properties. Firstly, we consider an ideal system without friction or elasticity, exerting neither forces nor moments on the environment. These phenomena will be covered in § 9.3.4 through 9.3.8.

The Lagrange formulation describes the behavior of a dynamic system in terms of work and energy stored in the system. The Lagrange equations are commonly written in the form:

$$\Gamma_i = \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} \qquad \text{for } i = 1, \ldots, n \qquad [9.4a]$$

where L is the Lagrangian of the robot defined as the difference between the kinetic energy E and the potential energy U of the system:

$$L = E - U \qquad [9.4b]$$

9.3.2. *General form of the dynamic equations*

The kinetic energy of the system is a quadratic function in the joint velocities such that:

$$E = \frac{1}{2}\dot{\mathbf{q}}^T \mathbf{A}\, \dot{\mathbf{q}} \qquad [9.5]$$

where **A** is the (nxn) symmetric and positive definite *inertia matrix* of the robot. Its elements are functions of the joint positions. The (i, j) element of **A** is denoted by A_{ij}.

Since the potential energy is a function of the joint positions, equations [9.4] and [9.5] lead to:

$$\Gamma = \mathbf{A}(\mathbf{q})\, \ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\, \dot{\mathbf{q}} + \mathbf{Q}(\mathbf{q}) \qquad [9.6]$$

where:

- $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\, \dot{\mathbf{q}}$ is the (nx1) vector of Coriolis and centrifugal torques, such that:
  $\mathbf{C}\, \dot{\mathbf{q}} = \dot{\mathbf{A}}\, \dot{\mathbf{q}} - \frac{\partial E}{\partial \mathbf{q}}$
- $\mathbf{Q} = [Q_1 \ \ldots \ Q_n]^T$ is the vector of gravity torques.

Consequently, the dynamic model of a robot is described by n coupled and nonlinear second order differential equations.

There exist several forms for the vector $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\, \dot{\mathbf{q}}$. Using the *Christoffell symbols* $c_{i,jk}$, the (i, j) element of the matrix **C** can be written as:

$$\begin{cases} C_{ij} = \sum_{k=1}^{n} c_{i,jk}\,\dot{q}_k \\ c_{i,jk} = \frac{1}{2}\left[\frac{\partial A_{ij}}{\partial q_k} + \frac{\partial A_{ik}}{\partial q_j} - \frac{\partial A_{jk}}{\partial q_i}\right] \end{cases} \qquad [9.7]$$

The Q_i element of the vector **Q** is calculated according to:

$$Q_i = \frac{\partial U}{\partial q_i} \qquad [9.8]$$

The elements of **A**, **C** and **Q** are functions of the geometric and inertial parameters of the robot.

9.3.3. *Computation of the elements of A, C and Q*

To compute the elements of **A**, **C** and **Q**, we begin by symbolically computing the expressions of the kinetic and potential energies of all the links of the robot. Then, we proceed as follows:

- the element A_{ii} is equal to the coefficient of $(\dot{q}_i^2/2)$ in the expression of the kinetic energy, while A_{ij}, for $i \neq j$, is equal to the coefficient of $\dot{q}_i\,\dot{q}_j$;
- the elements of **C** are obtained from equation [9.7];
- the elements of **Q** are obtained from equation [9.8].

9.3.3.1. *Computation of the kinetic energy*

The kinetic energy of the robot is given as:

$$E = \sum_{j=1}^{n} E_j \qquad [9.9]$$

where E_j denotes the kinetic energy of link j, which can be computed by:

$$E_j = \frac{1}{2}\left(\omega_j^T\, I_{Gj}\, \omega_j + M_j\, V_{Gj}^T\, V_{Gj}\right) \qquad [9.10]$$

Since the velocity of the center-of-mass can be expressed as (Figure 9.1):

$$V_{Gj} = V_j + \omega_j \times S_j \qquad [9.11]$$

and since:

$$\mathbf{J}_j = \mathbf{I}_{Gj} - M_j \hat{\mathbf{S}}_j \hat{\mathbf{S}}_j \qquad [9.12]$$

equation [9.10] becomes:

$$E_j = \frac{1}{2}[\omega_j^T \mathbf{J}_j \omega_j + M_j \mathbf{V}_j^T \mathbf{V}_j + 2\,\mathbf{MS}_j^T (\mathbf{V}_j \times \omega_j)] \qquad [9.13]$$

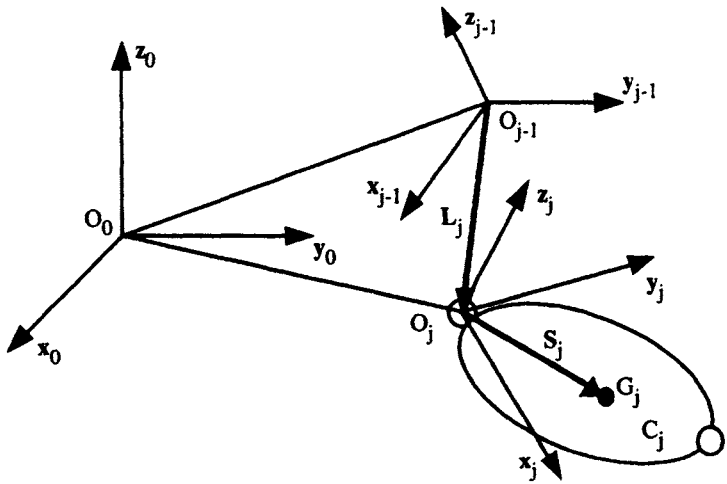

Figure 9.1. *Composition of velocities*

Equation [9.10] is not linear in the coordinates of the vector $\mathbf{S}_j$. On the contrary, equation [9.13] is linear in the elements of M_j, $\mathbf{MS}_j$ and $\mathbf{J}_j$, which we call the *standard inertial parameters*. The linear and angular velocities $\mathbf{V}_j$ and ω_j are computed using the following recursive equations (Figure 9.1):

$$\omega_j = \omega_{j-1} + \bar{\sigma}_j \dot{q}_j \mathbf{a}_j \qquad [9.14]$$

$$\mathbf{V}_j = \mathbf{V}_{j-1} + \omega_{j-1} \times \mathbf{L}_j + \sigma_j \dot{\mathbf{q}}_j \mathbf{a}_j \qquad [9.15]$$

If the base of the robot is fixed, the previous equations are initialized by $\mathbf{V}_0 = \mathbf{0}$ and $\omega_0 = \mathbf{0}$.

All the elements appearing in equation [9.13] must be expressed in the same frame. The most efficient way is to express them relative to frame R_j. Therefore, equations [9.13], [9.14] and [9.15] are rewritten as:

$$E_j = \frac{1}{2}\left[{}^j\omega_j{}^T\,{}^jJ_j\,{}^j\omega_j + M_j\,{}^jV_j{}^T\,{}^jV_j + 2\,{}^jMS_j{}^T\,({}^jV_j \times {}^j\omega_j)\right] \quad [9.16]$$

$${}^j\omega_j = {}^jA_{j-1}\,{}^{j-1}\omega_{j-1} + \bar{\sigma}_j\,\dot{q}_j\,{}^ja_j = {}^j\omega_{j-1} + \bar{\sigma}_j\,\dot{q}_j\,{}^ja_j \quad [9.17]$$

$${}^jV_j = {}^jA_{j-1}\,({}^{j-1}V_{j-1} + {}^{j-1}\omega_{j-1} \times {}^{j-1}P_j) + \sigma_j\,\dot{q}_j\,{}^ja_j \quad [9.18]$$

The parameters ${}^jJ_j$ and ${}^jMS_j$ are constants.

Using the spatial notation, the kinetic energy can be written in the following compact form:

$$E_j = \frac{1}{2}\,{}^j\mathbb{V}_j^T\,{}^j\mathbb{J}_j\,{}^j\mathbb{V}_j \quad [9.19]$$

where:

$${}^j\mathbb{V}_j = \begin{bmatrix} {}^jV_j \\ {}^j\omega_j \end{bmatrix} \quad [9.20]$$

$${}^j\mathbb{J}_j = \begin{bmatrix} M_jI_3 & -{}^j\hat{MS}_j \\ {}^j\hat{MS}_j & {}^jJ_j \end{bmatrix} \quad [9.21]$$

The recursive velocity relations [9.17] and [9.18] can be combined as follows:

$${}^j\mathbb{V}_j = {}^j\mathbb{T}_{j-1}\,{}^{j-1}\mathbb{V}_{j-1} + \dot{q}_j\,{}^j\mathbb{a}_j \quad [9.22]$$

where ${}^j\mathbb{T}_{j-1}$ is the (6x6) screw transformation matrix defined in [2.46] as:

$${}^j\mathbb{T}_{j-1} = \begin{bmatrix} {}^jA_{j-1} & -{}^jA_{j-1}\,{}^{j-1}\hat{P}_j \\ 0_3 & {}^jA_{j-1} \end{bmatrix} = \begin{bmatrix} {}^jA_{j-1} & {}^j\hat{P}_{j-1}\,{}^jA_{j-1} \\ 0_3 & {}^jA_{j-1} \end{bmatrix} \quad [9.23a]$$

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

$${}^j\mathbb{a}_j = \begin{bmatrix} \sigma_j\,{}^ja_j \\ \bar{\sigma}_j\,{}^ja_j \end{bmatrix} \quad [9.23b]$$

9.3.3.2. *Computation of the potential energy*

The potential energy is given by:

$$U = \sum_{j=1}^{n} U_j = \sum_{j=1}^{n} -M_j \, \mathbf{g}^T (\mathbf{L}_{0,j} + \mathbf{S}_j) \qquad [9.24]$$

where $\mathbf{L}_{0,j}$ is the position vector from the origin O_0 to O_j. Projecting the vectors appearing in [9.24] into frame R_0, we obtain:

$$U_j = -M_j \, {}^0\mathbf{g}^T ({}^0\mathbf{P}_j + {}^0\mathbf{A}_j \, {}^j\mathbf{S}_j) \qquad [9.25a]$$

an expression that can be rewritten linearly in M_j and the elements of ${}^j\mathbf{MS}_j$ as:

$$U_j = -{}^0\mathbf{g}^T (M_j \, {}^0\mathbf{P}_j + {}^0\mathbf{A}_j \, {}^j\mathbf{MS}_j) = -\begin{bmatrix} {}^0\mathbf{g}^T & 0 \end{bmatrix} {}^0\mathbf{T}_j \begin{bmatrix} {}^j\mathbf{MS}_j \\ M_j \end{bmatrix} \qquad [9.25b]$$

Since the kinetic and potential energies are linear in the elements of ${}^j\mathbf{J}_j$, ${}^j\mathbf{MS}_j$, M_j, we deduce that the dynamic model is also linear in these parameters.

9.3.3.3. *Dynamic model properties*

In this section, we summarize some important properties of the dynamic model of robots:

a) the inertia matrix $\mathbf{A}$ is symmetric and positive definite;

b) the energy of link j is a function of $(q_1, \ldots, q_j)$ and $(\dot{q}_1, \ldots, \dot{q}_j)$;

c) the element A_{ij} is a function of $q_{k+1}, \ldots, q_n$, with $k = \min(i, j)$, and of the inertial parameters of links r, ..., n, with $r = \max(i, j)$;

d) from property b and equation [9.4], we deduce that Γ_i is a function of the inertial parameters of links i, ..., n;

e) the matrix $[\frac{d}{dt}\mathbf{A} - 2\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})]$ is skew-symmetric for the choice of the matrix $\mathbf{C}$ given by equation [9.7] [Koditschek 84], [Arimoto 84]. This property is used in Chapter 14 for the stability analysis of certain control schemes;

f) the inverse dynamic model is linear in the elements of the standard inertial parameters M_j, ${}^j\mathbf{MS}_j$ and ${}^j\mathbf{J}_j$. This property is exploited to identify the dynamic parameters (inertial and friction parameters), to reduce the computation burden of the dynamic model, and to develop adaptive control schemes;

g) there exist some positive real numbers a_1, ..., a_7 such that for any values of $\mathbf{q}$ and $\dot{\mathbf{q}}$ we have [Samson 87]:

$$\| \mathbf{A}(\mathbf{q}) \| \leq a_1 + a_2 \| \mathbf{q} \| + a_3 \| \mathbf{q} \|^2$$

$$\| \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \| \leq \| \dot{\mathbf{q}} \| (a_4 + a_5 \| \mathbf{q} \|)$$

$$\| \mathbf{Q} \| \leq a_6 + a_7 \| \mathbf{q} \|$$

where $\|*\|$ indicates a matrix or vector norm. If the robot has only revolute joints, these relations become:

$$\| \mathbf{A}(\mathbf{q}) \| \leq a_1$$

$$\| \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \| \leq a_4 \| \dot{\mathbf{q}} \|$$

$$\| \mathbf{Q} \| \leq a_6$$

h) a robot is a passive system which dissipates energy. This property is related to property e).

9.3.4. *Considering friction*

Friction plays a dominant role in limiting the quality of robot performance. Non-compensated friction produces static error, delay, and limit cycle behavior [Canudas de Wit 90]. Many works have been devoted to studying friction torque in the joint and transmission systems. Various friction models have been proposed in the literature [Dahl 77], [Canudas de Wit 89], [Armstrong 88], [Armstrong 91], [Armstrong 94]. In general, three kinds of frictions are noted: Coulomb friction, static friction, and viscous friction.

The model based on Coulomb friction assumes a constant friction component that is independent of the magnitude of the velocity. The static friction is the torque necessary to initiate motion from rest. It is often greater than the Coulomb friction (Figure 9.2a). The viscous friction is generally represented as being proportional to the velocity, but experimental studies [Armstrong 88] have pointed out the Stribeck phenomenon that arises from the use of fluid lubrication. It results in decreasing friction with increasing velocity at low velocity, then the friction becomes proportional to velocity (Figure 9.2b). A general friction model describing these components is given by:

$$\Gamma_{fi} = F_{ci} \operatorname{sign}(\dot{q}_i) + F_{vi} \dot{q}_i + (F_{sti} - F_{ci}) \operatorname{sign}(\dot{q}_i) e^{-|\dot{q}_i|B_i} \qquad [9.26]$$

In this expression, Γ_{fi} denotes the friction torque of joint i, F_{ci} and F_{vi} indicate the Coulomb and viscous friction parameters respectively. The static torque is equal to $F_{sti} \operatorname{sign}(\dot{q}_i)$.

The most often employed model is composed of Coulomb friction together with viscous friction (Figure 9.2c). Therefore, the friction torque at joint i is written as:

$$\Gamma_{fi} = F_{ci}\,\mathrm{sign}(\dot{q}_i) + F_{vi}\,\dot{q}_i \qquad [9.27]$$

To take into account the friction in the dynamic model of a robot, we add the vector Γ_f on the right side of equation [9.6] such that:

$$\Gamma_f = \mathbf{diag}(\dot{\mathbf{q}})\mathbf{F}_v + \mathbf{diag}[\mathrm{sign}(\dot{\mathbf{q}})]\mathbf{F}_c \qquad [9.28]$$

where:

- $\mathbf{F}_v = [\ F_{v1}\ \cdots\ F_{vn}\ ]^T$;
- $\mathbf{F}_c = [\ F_{c1}\ \cdots\ F_{cn}\ ]^T$;
- $\mathbf{diag}(\dot{\mathbf{q}})$ is the diagonal matrix whose elements are the components of $\dot{\mathbf{q}}$.

This friction model can be approximated by a piecewise linear model as shown in Figure 9.2d.

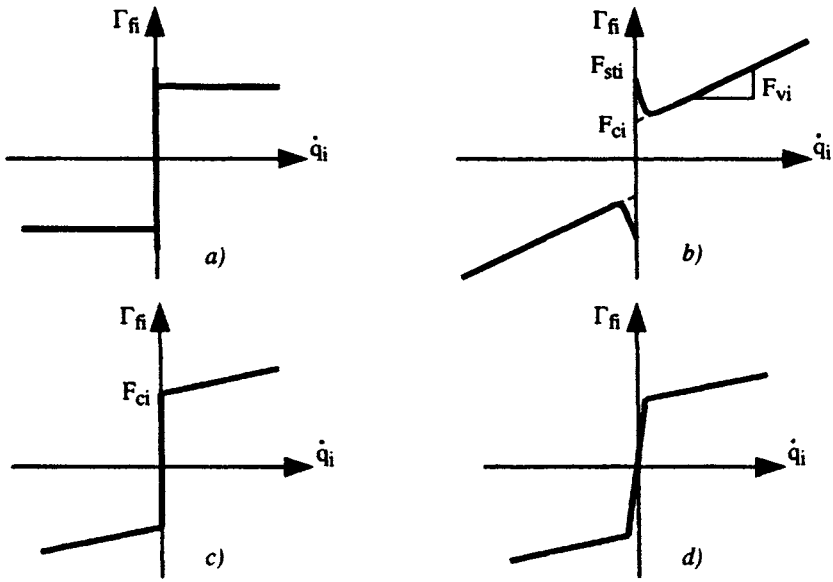

Figure 9.2. *Friction models*

9.3.5. *Considering the rotor inertia of actuators*

The kinetic energy of the rotor (and transmission system) of actuator j, is given by the expression $\frac{1}{2}\ Ia_j\ \dot{q}_j^2$. The inertial parameter Ia_j denotes the equivalent inertia referred to the joint velocity. It is given by:

$$Ia_j = N_j^2\ J_{mj} \qquad [9.29]$$

where J_{mj} is the moment of inertia of the rotor and transmissions of actuator j, N_j is the transmission ratio of joint j axis, equal to $\dot{q}_{mj} / \dot{q}_j$ where $\dot{q}_{mj}$ denotes the rotor velocity of actuator j. In the case of a prismatic joint, Ia_j is an equivalent mass.

In order to consider the rotor inertia in the dynamic model of the robot, we add the inertia (or mass) Ia_j to the A_{jj} element of the matrix **A**.

Note that such modeling neglects the gyroscopic effects of the rotors, which take place when the actuator is fixed on a moving link. However, this approximation is justified for high gear transmission ratios. For more accurate modeling of the rotors the reader is referred to [Llibre 83], [Chedmail 86], [Murphy 93], [Sciavicco 94].

9.3.6. *Considering the forces and moments exerted by the end-effector on the environment*

We have seen in § 5.9 that the joint torque vector Γ_e necessary to exert a given wrench $\mathbb{f}_{en}$ on the environment is obtained using the basic static equation:

$$\Gamma_e = J_n^T\ \mathbb{f}_{en} \qquad [9.30]$$

Thus, we have to add the vector Γ_e on the right side of equation [9.6].

9.3.7. *Relation between joint torques and actuator torques*

In general, the joint variables are not equal to the motor variables because of the existence of transmission systems or because of couplings between the actuator variables. The relation between joint torques and actuator torques can be obtained using the principle of virtual work. Let the relationship between the infinitesimal joint displacement $\mathbf{dq}$ and the infinitesimal actuator variable $\mathbf{dq_m}$ be given by:

$$\mathbf{dq} = \mathbf{J_{qm}}\ \mathbf{dq_m} \qquad [9.31]$$

where $\mathbf{J}_{qm}$ is the Jacobian of $\mathbf{q}$ with respect to $\mathbf{q}_m$, equal to $\frac{\partial \mathbf{q}}{\partial \mathbf{q}_m}$.

The virtual work can be written as:

$$\Gamma^T d\mathbf{q}^* = \tau_m^T d\mathbf{q}_m^* \qquad [9.32]$$

where τ_m is the actuator torque vector and the superscript (*) indicates virtual displacements.

Combining equations [9.31] and [9.32] yields:

$$\tau_m = \mathbf{J}_{q_m}^T \Gamma \qquad [9.33]$$

9.3.8. *Modeling of robots with elastic joints*

The presence of joint flexibility is a common feature of many current industrial robots. The joint elasticity may arise from several sources, such as elasticity in the gears, transmission belts, harmonic drives, etc. It follows that a time-varying displacement is introduced between the position of the driving actuator and the joint position. The joint elasticity is modeled as a linear torsional spring for revolute joints and a linear spring for prismatic joints [Khalil 78], [Spong 87]. Thus, the dynamic model requires twice the number of generalized coordinates to completely characterize the configuration of the links and the rotors of the actuators. Let $\mathbf{q}_M$ denote the (nx1) vector of rotor positions as reflected through the gear ratios (Figure 9.3). Consequently, the vector of joint deformations is given by $(\mathbf{q} - \mathbf{q}_M)$. Note that the direct geometric model is only a function of the joint variables $\mathbf{q}$.

The potential energy of the springs is given by:

$$U_k = \frac{1}{2} (\mathbf{q} - \mathbf{q}_M)^T \mathbf{k} (\mathbf{q} - \mathbf{q}_M) \qquad [9.34]$$

where $\mathbf{k}$ is the (nxn) definite positive joint stiffness matrix.

The dynamic equations are obtained using the Lagrange equation, i.e.:

$$\mathbf{A}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{Q} + \mathbf{k}(\mathbf{q} - \mathbf{q}_M) = \mathbf{0} \qquad [9.35a]$$

$$\mathbf{I}_a \ddot{\mathbf{q}}_M + \mathbf{diag}(\dot{\mathbf{q}}_M)\,\mathbf{F}_{vm} + \mathbf{diag}(\mathrm{sign}(\dot{\mathbf{q}}_M))\,\mathbf{F}_{cm} - \mathbf{k}(\mathbf{q} - \mathbf{q}_M) = \Gamma \qquad [9.35b]$$

where $\mathbf{I_a}$ is the (nxn) diagonal matrix containing the inertia of the rotors, $\mathbf{F_{vm}}$ and $\mathbf{F_{cm}}$ are the (nx1) vectors containing the viscous and Coulomb parameters of the actuators and transmissions referred to the joint side. The joint friction terms can easily be included in equation [9.35a].

A general and systematic method to model systems with lumped elasticity is presented in [Khalil 00a].

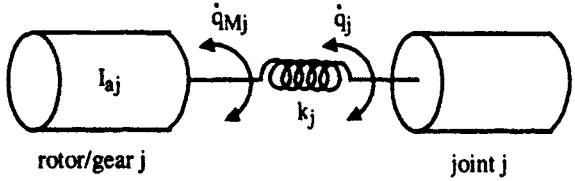

Figure 9.3. *Modeling of joint flexibility*

• **Example 9.1.** Computation of the elements of the matrices **A** and **Q** for the first three links of the Stäubli RX-90 robot whose geometric parameters are given in Example 3.1.

i) computation of the angular velocities (equation [9.17])

$$^0\omega_0 = \mathbf{0}$$

$$^1\omega_1 = [0 \quad 0 \quad \dot{q}_1]^T$$

$$^2\omega_2 = {}^2A_1\,{}^1\omega_1 + \dot{q}_2\,{}^2a_2$$

$$= \begin{bmatrix} C2 & 0 & S2 \\ -S2 & 0 & C2 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{q}_1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{q}_2 \end{bmatrix} = [S2\,\dot{q}_1 \quad C2\,\dot{q}_1 \quad \dot{q}_2]^T$$

$$^3\omega_3 = {}^3A_2\,{}^2\omega_2 + \dot{q}_3\,{}^3a_3$$

$$= \begin{bmatrix} C3 & S3 & 0 \\ -S3 & C3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S2\,\dot{q}_1 \\ C2\,\dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{q}_3 \end{bmatrix} = [S23\,\dot{q}_1 \quad C23\,\dot{q}_1 \quad \dot{q}_1 + \dot{q}_2]^T$$

ii) computation of the linear velocities (equation [9.18])

$$^0v_0 = \mathbf{0}$$

$$^1V_1 = 0$$
$$^1V_2 = {}^1V_1 + {}^1\omega_1 \times {}^1P_2 = 0$$
$$^2V_2 = 0$$
$$^2V_3 = {}^2\omega_2 \times {}^2P_3 = [0 \quad D3\,\dot{q}_2 \quad -C2D3\,\dot{q}_1]^T$$
$$^3V_3 = {}^3A_2\,{}^2V_3 = [S3\,D3\,\dot{q}_2 \quad C3\,D3\,\dot{q}_2 \quad -C2D3\,\dot{q}_1]^T$$

iii) computation of the inertia matrix **A**. With the following general inertial parameters:

$$^jMS_j = [MX_j \quad MY_j \quad MZ_j]^T$$

$$^jJ_j = \begin{bmatrix} XX_j & XY_j & XZ_j \\ XY_j & YY_j & YZ_j \\ XZ_j & YZ_j & ZZ_j \end{bmatrix}, I_a = \begin{bmatrix} I_{a1} & 0 & 0 \\ 0 & I_{a2} & 0 \\ 0 & 0 & I_{a3} \end{bmatrix}$$

we obtain the elements of the robot inertia matrix as:

$$A_{11} = Ia_1 + ZZ_1 + SS2\,XX_2 + 2CS2\,XY_2 + CC2\,YY_2 + SS23\,XX_3 + 2CS23\,XY_3 + CC23\,YY_3 + 2C2\,C23\,D3\,MX_3 - 2C2\,S23\,D3\,MY_3 + CC2\,D3^2\,M_3$$
$$A_{12} = S2\,XZ_2 + C2\,YZ_2 + S23\,XZ_3 + C23\,YZ_3 - S2\,D3\,MZ_3$$
$$A_{13} = S23\,XZ_3 + C23\,YZ_3$$
$$A_{22} = Ia_2 + ZZ_2 + ZZ_3 + 2C3\,D3\,MX_3 - 2S3\,D3\,MY_3 + D3^2\,M_3$$
$$A_{23} = ZZ_3 + C3\,D3\,MX_3 - S3\,D3\,MY_3$$
$$A_{33} = Ia_3 + ZZ_3$$

where $SSj = (\sin\theta_j)^2$, $CCj = (\cos\theta_j)^2$ and $CSj = \cos\theta_j \sin\theta_j$. The elements of the matrix **C** can be computed by equation [9.7];

iv) computation of the gravity forces. Assuming that gravitational acceleration is given as $^0\mathbf{g} = [0 \quad 0 \quad G3]^T$, and using equation [9.25], the potential energy is obtained as:

$$U = -G3\,(MZ_1 + S2MX_2 + C2MY_2 + S23MX_3 + C23MY_3 + D3S2M_3)$$

Using equation [9.8] gives:

$$Q_1 = 0$$
$$Q_2 = -G3\,(C2\,MX_2 - S2\,MY_2 + C23\,MX_3 - S23\,MY_3 + D3\,C2\,M_3)$$

$$Q_3 = -G3\,(C23\,MX_3 - S23\,MY_3)$$

9.4. Determination of the base inertial parameters

In this section, we study the concept of *base inertial parameters* or *identifiable parameters*. We develop a straightforward closed-form method to determine them. These parameters constitute the minimum set of inertial parameters that are needed to compute the dynamic model of a robot [Mayeda 90]. The use of the base inertial parameters in a customized Newton-Euler algorithm reduces its computational cost [Khalil 86c], [Khalil 87b]. The determination of the base parameters is also essential for the identification of the dynamic parameters (Chapter 12), since they constitute the only identifiable parameters. The base inertial parameters can be deduced from the standard parameters by eliminating those that have no effect on the dynamic model and by grouping some others.

9.4.1. *Computation of the base parameters using the dynamic model*

Since the kinetic energy and the potential energy are linear in the standard inertial parameters, we deduce that the dynamic model is also linear in these parameters. It can be written as:

$$\Gamma = \sum_{j=1}^{N_p} \mathbf{D}^j K_j = \mathbf{D}\,\mathbf{K} \qquad [9.36]$$

where:

- $\mathbf{D}$: $(n \times N_p)$ matrix, which is a function of $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ and the geometric parameters;
- $\mathbf{K}$: $(N_p \times 1)$ vector of the standard inertial parameters of the links representing for each link a mass, three elements for the first moments, six elements for the inertia tensor, and one element for the rotor inertia ($N_p = 11n$);
- $\mathbf{D}^j$: j^{th} column of $\mathbf{D}$.

From equation [9.36], we deduce that:

a) a parameter K_j has no effect on the dynamic model if:

$$\mathbf{D}^j = \mathbf{0} \qquad [9.37]$$

Consequently, we can set $K_j = 0$ in equation [9.36] without changing the value of Γ, which means that K_j can be eliminated;

b) a parameter K_j can be grouped with some other parameters K_{j1}, ..., K_{jr}, if the $\mathbf{D}^j$ column is linearly dependent of $\mathbf{D}^{j1}$, ..., $\mathbf{D}^{jr}$ such that:

$$\mathbf{D}^j = t_{j1}\,\mathbf{D}^{j1} + \ldots + t_{jr}\,\mathbf{D}^{jr} \qquad [9.38]$$

where all t_{jk} are constants.

In that case, the column $\mathbf{D}^j$ and the parameter K_j can be eliminated while the parameters K_{j1}, ..., K_{jr}, will be replaced by KR_{j1}, ..., KR_{jr}, where $KR_{jp} = K_{jp} + t_{jp}\,K_j$, for p = 1, ..., r. This operation will be repeated until the elimination of all the parameters with dependent columns. At the end, we obtain the minimum inertial parameter vector $\mathbf{K_B}$.

The selection of the parameters to be eliminated is not unique. We choose to eliminate those with the highest subscript in **K**. The search for dependent columns of **D** starts with the last column and moves backwards toward the first one. The link parameters are arranged in **K** such that we first place the parameters of link 1, and lastly, those of link n. The parameters of link j, defined by the vector $\mathbf{K}_j$, are given in the following order: XX_j, XY_j, XZ_j, YY_j, YZ_j, ZZ_j, MX_j, MY_j, MZ_j, M_j, Ia_j.

In summary, the calculation of $\mathbf{K_B}$ is based on the study of the linear dependence of the columns of the matrix **D**. Assuming b to be the rank of the matrix **D**, the determination of the base parameters can be illustrated in a compact and global form by writing equation [9.36] as:

$$\Gamma = [\,\mathbf{D1}\;\;\mathbf{D2}\,]\begin{bmatrix}\mathbf{K1}\\ \mathbf{K2}\end{bmatrix} \qquad [9.39]$$

where:

- **D1** represents the first b independent columns of **D**;
- **D2** represents the dependent columns of **D** such that $\mathbf{D2} = \mathbf{D1}\beta$, where β is a constant matrix.

We deduce that the parameters **K2** can be grouped with **K1** as follows:

$$\Gamma = \mathbf{D1}\,[\mathbf{K1} + \beta\,\mathbf{K2}] = \mathbf{D1}\,\mathbf{K_B} \qquad [9.40]$$

In Appendix 5, we present a general numerical method for determining the base inertial parameters [Gautier 91]. This numerical approach is based on the use of the QR decomposition. It can be used to determine the base parameters of closed chain robots and the identifiable geometric parameters of robots (Chapter 11).

9.4.2. *Determination of the base parameters using the energy model*

The use of the dynamic model to compute the base inertial parameters is tedious and error prone, owing to the complicated expressions of $\mathbf{D}^j$. In this section, we present a straightforward closed-form method for determining the base parameters. The demonstration of this method is based on the energy formulation, but the algorithm itself consists of simple rules, which do not need to calculate the energy expressions [Gautier 90b], [Khalil 94a].

Since the total energy of link j is linear in the inertial parameters, it can be written as:

$$H_j = E_j + U_j = \mathbf{h}_j \mathbf{K}_j = (\mathbf{e}_j + \mathbf{u}_j)\,{}^j\mathbf{K}_j \qquad [9.41]$$

$${}^j\mathbf{K}_j = [\ XX_j\ \ XY_j\ \ XZ_j\ \ YY_j\ \ YZ_j\ \ ZZ_j\ \ MX_j\ \ MY_j\ \ MZ_j\ \ M_j\ ]^T \qquad [9.42]$$

$$\mathbf{h}_j = [\ h_{XXj}\ \ h_{XYj}\ \ h_{XZj}\ \ h_{YYj}\ \ h_{YZj}\ \ h_{ZZj}\ \ h_{MXj}\ \ h_{MYj}\ \ h_{MZj}\ \ h_{Mj}\ ] \qquad [9.43]$$

with:

- $\mathbf{K}_j$: (10x1) vector of the standard inertial parameters of link j (the parameter I_{aj} will be considered in § 9.4.2.5);
- $\mathbf{h}_j$: (1x10) row vector of the total energy functions of link j;
- $\mathbf{e}_j$: (1x10) row vector of the kinetic energy functions of link j;
- $\mathbf{u}_j$: (1x10) row vector of the potential energy functions of link j.

The elements of $\mathbf{h}_j$ are obtained from equations [9.13] and [9.25] as:

$$\begin{cases} h_{XXj} = \frac{1}{2}\omega_{1,j}\,\omega_{1,j} \\ h_{XYj} = \omega_{1,j}\,\omega_{2,j} \\ h_{XZj} = \omega_{1,j}\,\omega_{3,j} \\ h_{YYj} = \frac{1}{2}\omega_{2,j}\,\omega_{2,j} \\ h_{YZj} = \omega_{2,j}\,\omega_{3,j} \\ h_{ZZj} = \frac{1}{2}\omega_{3,j}\,\omega_{3,j} \\ h_{MXj} = \omega_{3,j}\,V_{2,j} - \omega_{2,j}\,V_{3,j} - {}^0\mathbf{g}^T\,{}^0\mathbf{s}_j \\ h_{MYj} = \omega_{1,j}\,V_{3,j} - \omega_{3,j}\,V_{1,j} - {}^0\mathbf{g}^T\,{}^0\mathbf{n}_j \\ h_{MZj} = \omega_{2,j}\,V_{1,j} - \omega_{1,j}\,V_{2,j} - {}^0\mathbf{g}^T\,{}^0\mathbf{a}_j \\ h_{Mj} = \frac{1}{2}\,{}^jV_j^T\,{}^jV_j - {}^0\mathbf{g}^T\,{}^0\mathbf{P}_j \end{cases} \qquad [9.44]$$

where ${}^j\omega_j = [\omega_{1,j} \quad \omega_{2,j} \quad \omega_{3,j}]^T$ and ${}^jV_j = [V_{1,j} \quad V_{2,j} \quad V_{3,j}]^T$.

From equations [9.13] and [9.25], we deduce the following recursive relationship between the energy functions of link j and link j – 1 (Appendix 6):

$$\mathbf{h}_j = \mathbf{h}_{j-1}\, {}^{j-1}\lambda_j + \dot{q}_j \eta_j \qquad [9.45]$$

where ${}^{j-1}\lambda_j$ is a (10x10) matrix whose elements are given in Table 9.1. It is a function of the geometric parameters of frame R_j [Gautier 90a]. The matrix ${}^{j-1}\lambda_j$ represents the transformation matrix of the inertial parameters of a link from frame R_j into frame R_{j-1} such that:

$${}^{j-1}K_j = {}^{j-1}\lambda_j\, {}^jK_j \qquad [9.46]$$

The row vector η_j is written as:

$$\eta_j = \bar{\sigma}_j [0 \quad 0 \quad \omega_{1,j} \quad 0 \quad \omega_{2,j} \quad (\omega_{3,j} - \frac{1}{2}\dot{q}_j) \quad V_{2,j} \quad -V_{1,j} \quad 0 \quad 0]$$
$$+ \sigma_j [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad -\omega_{2,j} \quad \omega_{1,j} \quad 0 \quad (V_{3,j} - \frac{1}{2}\dot{q}_j)] \qquad [9.47]$$

9.4.2.1. *Determination of the parameters having no effect on the dynamic model*

From equation [9.37], we deduce that an inertial parameter K_j has no effect on the dynamic model if the corresponding energy function h_j is constant;

h_j constant → K_j has no effect on the dynamic model [9.48]

Referring to the velocity equations [9.17] and [9.18] and to the $\mathbf{h}_j$ functions [9.44], we can obtain general rules to determine the parameters that have no effect on the dynamic model [Khalil 94a]. Let us assume that r_1 is the first revolute joint and r_2 is the subsequent revolute joint whose axis is not parallel to the axis of joint r_1 (Figure 9.4). The parameters that have no effect on the dynamic model belong to the links 1, ..., r_2, owing to the restricted motion of these links. The rules allowing us to determine them are given in the general algorithm presented in § 9.4.2.4.

Table 9.1. *Expression of the elements of matrix* $^{j-1}\lambda_j$

CCθ	−2CSθ	0	SSθ	0	0	0	0	2r	r^2
CSθCα	(CCθ−SSθ)Cα	−CθSα	−CSθCα	SθSα	0	−dSθCα+rCθSα	−dCθCα−rSθSα	dSα	drSα
CSθSα	(CCθ−SSθ)Sα	CθCα	−CSθSα	−SθCα	0	−dSθSα−rCθCα	−dCθSα+rSθCα	−dCα	−drCα
SSθCCα	2CSθCCα	−2SθCSα	CCθCCα	−2CθCSα	SSα	2(dCθ+rSθCSα)	2(−dSθ+rCθCSα)	2rCCα	d^2+r^2CCα
SSθCSα	2CSθCSα	Sθ(CCα−SSα)	CCθCSα	Cθ(CCα−SSα)	−CSα	rSθ(SSα−CCα)	rCθ(SSα−CCα)	2rCSα	r^2CSα
SSθSSα	2CSθSSα	2SθCSα	CCθSSα	2CθCSα	CCα	2(dCθ−rSθCSα)	2(−dSθ−rCθCSα)	2rSSα	d^2+r^2SSα
0	0	0	0	0	0	Cθ	−Sθ	0	d
0	0	0	0	0	0	SθCα	CθCα	−Sα	−rSα
0	0	0	0	0	0	SθSα	CθSα	Cα	rCα
0	0	0	0	0	0	0	0	0	1

The subscript j of the geometric parameters has been omitted.

The following notations are used: $CS^* = cos(*)\,sin(*)$, $CC^* = cos^2(*)$, $SS^* = sin^2(*)$

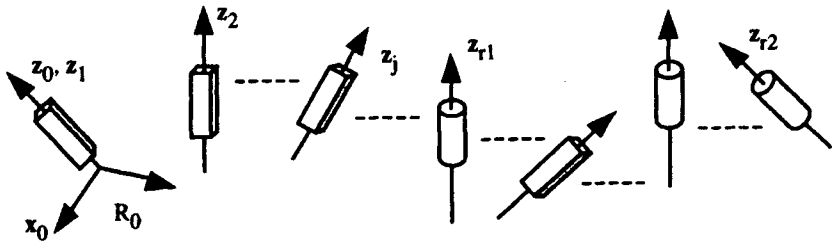

Figure 9.4. *Definition of joints r_1 and r_2*

9.4.2.2. *General grouping relations*

From condition [9.38], a parameter K_j is grouped with the parameters K_{jp} for $p = 1, \ldots, r$ if the corresponding energy function h_j can be written as:

$$h_j = \sum_{p=1}^{r} t_{jp} h_{jp} + \text{constant} \qquad [9.49]$$

The grouping relations are the same as in § 9.4.1, that is to say if equation [9.49] holds, then the parameter K_j can be grouped with the parameters K_{jp} for $p = 1, \ldots, r$ using the relation $KR_{jp} = K_{jp} + t_{jp} K_j$.

Using the recursive relation [9.45] between the energy functions $\mathbf{h}_j$ and $\mathbf{h}_{j-1}$, we can find general energy functions that satisfy equation [9.49]. These functions depend on the type of joint.

Let us first consider the case where joint j is revolute. The following three relations always hold:

$$h_{XXj} + h_{YYj} = \mathbf{h}_{j-1} ({}^{j-1}\boldsymbol{\lambda}_j^1 + {}^{j-1}\boldsymbol{\lambda}_j^4) \qquad [9.50a]$$

$$h_{MZj} = \mathbf{h}_{j-1}\, {}^{j-1}\boldsymbol{\lambda}_j^9 \qquad [9.50b]$$

$$h_{Mj} = \mathbf{h}_{j-1}\, {}^{j-1}\boldsymbol{\lambda}_j^{10} \qquad [9.50c]$$

where ${}^{j-1}\boldsymbol{\lambda}_j^k$ denotes the k^{th} column of the matrix ${}^{j-1}\boldsymbol{\lambda}j$.

Consequently, three inertial parameters can be grouped with the others. The choice of these parameters is not unique. By choosing to group the parameters YY_j, MZ_j and M_j we obtain:

$$XXR_j = XX_j - YY_j \qquad [9.51]$$

$$\mathbf{KR}_{j-1} = \mathbf{K}_{j-1} + YY_j ({}^{j-1}\lambda_j^1 + {}^{j-1}\lambda_j^4) + MZ_j \, {}^{j-1}\lambda_j^9 + M_j \, {}^{j-1}\lambda_j^{10} \qquad [9.52]$$

Substituting for ${}^{j-1}\lambda_j^1, {}^{j-1}\lambda_j^4, {}^{1}\lambda_j^9, {}^{j-1}\lambda_j^{10}$ from Table 9.1 into equations [9.51] and [9.52] gives the following theorem:

Theorem 9.1. If joint j is revolute, the parameters YY_j, MZ_j and M_j can be grouped with the parameters of link j and link j – 1. The resulting grouped parameters are:

$$\begin{aligned}
XXR_j &= XX_j - YY_j \\
XXR_{j-1} &= XX_{j-1} + YY_j + 2\, r_j\, MZ_j + r_j^2\, M_j \\
XYR_{j-1} &= XY_{j-1} + d_j\, S\alpha_j\, MZ_j + d_j\, r_j\, S\alpha_j\, M_j \\
XZR_{j-1} &= XZ_{j-1} - d_j\, C\alpha_j\, MZ_j - d_j\, r_j\, C\alpha_j\, M_j \\
YYR_{j-1} &= YY_{j-1} + CC\alpha_j\, YY_j + 2\, r_j\, CC\alpha_j\, MZ_j + (d_j^2 + r_j^2\, CC\alpha_j)\, M_j \\
YZR_{j-1} &= YZ_{j-1} + CS\alpha_j\, YY_j + 2\, r_j\, CS\alpha_j\, MZ_j + r_j^2\, CS\alpha_j\, M_j \\
ZZR_{j-1} &= ZZ_{j-1} + SS\alpha_j\, YY_j + 2\, r_j SS\alpha_j\, MZ_j + (d_j^2 + r_j^2\, SS\alpha_j)\, M_j \\
MXR_{j-1} &= MX_{j-1} + d_j\, M_j \\
MYR_{j-1} &= MY_{j-1} - S\alpha_j\, MZ_j - r_j\, S\alpha_j\, M_j \\
MZR_{j-1} &= MZ_{j-1} + C\alpha_j\, MZ_j + r_j\, C\alpha_j\, M_j \\
MR_{j-1} &= M_{j-1} + M_j
\end{aligned} \qquad [9.53]$$

where $SS(*) = S(*)\,S(*)$, $CC(*) = C(*)\,C(*)$ and $CS(*) = C(*)\,S(*)$.

On the other hand, if joint j is prismatic, the following six relations always hold:

$$h_{XXj} = \mathbf{h}_{j-1} \, {}^{j-1}\lambda_j^1, \; h_{XYj} = \mathbf{h}_{j-1} \, {}^{j-1}\lambda_j^2, \; \ldots, \; h_{ZZj} = \mathbf{h}_{j-1} \, {}^{j-1}\lambda_j^6 \qquad [9.54]$$

Therefore, six parameters can be grouped. Choosing to group the parameters of link j with those of link j – 1 yields:

$$\mathbf{KR}_{j-1} = \mathbf{K}_{j-1} + {}^{j-1}\lambda_j^1 \, XX_j + {}^{j-1}\lambda_j^2 \, XY_j + \ldots + {}^{j-1}\lambda_j^6 \, ZZ_j \qquad [9.55]$$

Expanding equation [9.55] gives the following theorem:

Theorem 9.2. If joint j is prismatic, the parameters of the inertia tensor of link j can be grouped with those of link j – 1. The grouping relations are:

$$\begin{aligned}
XXR_{j-1} &= XX_{j-1} + CC\theta_j\, XX_j - 2\, CS\theta_j\, XY_j + SS\theta_j\, YY_j \\
XYR_{j-1} &= XY_{j-1} + CS\theta_j\, C\alpha_j\, XX_j + (CC\theta_j - SS\theta_j)\, C\alpha_j\, XY_j - C\theta_j\, S\alpha_j\, XZ_j \\
&\quad - CS\theta_j\, C\alpha_j\, YY_j + S\theta_j\, S\alpha_j\, YZ_j
\end{aligned}$$

$$
\begin{aligned}
XZR_{j-1} &= XZ_{j-1} + CS\theta_j\, S\alpha_j\, XX_j + (CC\theta_j - SS\theta_j)\, S\alpha_j\, XY_j + C\theta_j\, C\alpha_j\, XZ_j \\
&\quad - CS\theta_j\, S\alpha_j\, YY_j - S\theta_j\, C\alpha_j\, YZ_j \\
YYR_{j-1} &= YY_{j-1} + SS\theta_j\, CC\alpha_j\, XX_j + 2CS\theta_j\, CC\alpha_j\, XY_j - 2S\theta_j\, CS\alpha_j\, XZ_j \\
&\quad + CC\theta_j\, CC\alpha_j\, YY_j - 2C\theta_j\, CS\alpha_j\, YZ_j + SS\alpha_j\, ZZ_j \qquad [9.56] \\
YZR_{j-1} &= YZ_{j-1} + SS\theta_j\, CS\alpha_j\, XX_j + 2CS\theta_j\, CS\alpha_j\, XY_j + S\theta_j\, (CC\alpha_j - SS\alpha_j)\, XZ_j \\
&\quad + CC\theta_j\, CS\alpha_j\, YY_j + C\theta_j\, (CC\alpha_j - SS\alpha_j)\, YZ_j - CS\alpha_j\, ZZ_j \\
ZZR_{j-1} &= ZZ_{j-1} + SS\theta_j\, SS\alpha_j\, XX_j + 2CS\theta_j\, SS\alpha_j\, XY_j + 2S\theta_j\, CS\alpha_j\, XZ_j \\
&\quad + CC\theta_j\, SS\alpha_j\, YY_j + 2C\theta_j\, CS\alpha_j\, YZ_j + CC\alpha_j\, ZZ_j
\end{aligned}
$$

Equation [9.56] can be rewritten under the following matrix form:

$$
{}^{j-1}\mathbf{JR}_{j-1} = {}^{j-1}\mathbf{J}_{j-1} + {}^{j-1}\mathbf{A}_j\, {}^{j}\mathbf{J}_j\, {}^{j}\mathbf{A}_{j-1} \qquad [9.57]
$$

where ${}^{j}\mathbf{J}_j$ is the inertia tensor of link j, and ${}^{j-1}\mathbf{A}_j$ is the (3x3) orientation matrix of frame R_j relative to frame R_{j-1}.

Relation [9.57] can be also obtained by calculating the sum of the rotational kinetic energy of link j – 1 and link j in terms of the angular velocity of link j – 1, and by noting that when joint j is prismatic the angular velocity of link j is equal to the angular velocity of link j – 1.

9.4.2.3. *Particular grouped parameters*

Equations [9.53] and [9.56] allow us to compute most of the grouped inertial parameters of any serial robot. Additional grouping of inertial parameters concerns the parameters of the prismatic links between link r_1 and link r_2 (joints r_1 and r_2 are defined in § 9.4.2.1). The following two cases are considered [Khalil 94a]:

i) if the axis of the prismatic joint j, $r_1 < j < r_2$, is not parallel to the r_1 axis, then the functions h_{MXj}, h_{MYj} and h_{MZj} satisfy the relation:

$$
{}^{j}a_{xr1}\, h_{MXj} + {}^{j}a_{yr1}\, h_{MYj} + {}^{j}a_{zr1}\, h_{MZj} = \text{constant} \qquad [9.58]
$$

where ${}^{j}\mathbf{a}_{r1} = [{}^{j}a_{xr1} \quad {}^{j}a_{yr1} \quad {}^{j}a_{zr1}]^T$ is the unit vector along the $\mathbf{z}_{r1}$ axis referred to frame R_j.

Therefore, depending on the particular values of the components of ${}^{j}\mathbf{a}_{r1}$, one parameter can be eliminated or grouped as shown in Table 9.2.

Table 9.2. *Grouped parameters if $r_1 < j < r_2$, $\sigma_j = 1$ and j is not parallel to the r_1 axis*

Condition	Grouping or elimination
${}^{j}a_{zr1} \neq 0$	$MXR_j = MX_j - \frac{{}^{j}a_{xr1}}{{}^{j}a_{zr1}} MZ_j$ $MYR_j = MY_j - \frac{{}^{j}a_{yr1}}{{}^{j}a_{zr1}} MZ_j$
${}^{j}a_{zr1} = 0, {}^{j}a_{xr1}\, {}^{j}a_{yr1} \neq 0$	$MXR_j = MX_j - \frac{{}^{j}a_{xr1}}{{}^{j}a_{yr1}} MY_j$
${}^{j}a_{zr1} = 0, {}^{j}a_{xr1} = 0$	$MY_j = 0$ (has no effect)
${}^{j}a_{zr1} = 0, {}^{j}a_{yr1} = 0$	$MX_j = 0$ (has no effect)

ii) if the axis of the prismatic joint j, $r_1 < j < r_2$, is parallel to the r_1 axis, then the following relation is obtained:

$$\mathbf{h}_{MSj}^{T} = {}^{j}\mathbf{A}_{j-1}\, \mathbf{h}_{MSj-1}^{T} + \begin{bmatrix} 2\, d_j C\theta_j\, h_{ZZi} & -2 d_j S\theta_j\, h_{ZZi} & 0 \end{bmatrix}^{T} \qquad [9.59]$$

where i denotes the nearest revolute joint from j back to the base, $i \geq r_1$, and:

$$\mathbf{h}_{MSj} = \begin{bmatrix} h_{MXj} & h_{MYj} & h_{MZj} \end{bmatrix}$$

Therefore, we deduce that the parameter MZ_j has no effect on the dynamic model and that the parameters MX_j and MY_j can be grouped using the relations:

$$\begin{aligned} MXR_{j-1} &= MX_{j-1} + C\theta_j\, MX_j - S\theta_j\, MY_j \\ MYR_{j-1} &= MY_{j-1} + S\theta_j\, C\alpha_j\, MX_j + C\theta_j\, C\alpha_j\, MY_j \\ MZR_{j-1} &= MZ_{j-1} + S\theta_j\, S\alpha_j\, MX_j + C\theta_j\, S\alpha_j\, MY_j \\ ZZR_i &= ZZ_i + 2\, d_j\, C\theta_j\, MX_j - 2\, d_j\, S\theta_j\, MY_j \end{aligned} \qquad [9.60]$$

9.4.2.4. *Practical determination of the base parameters*

The following algorithm can be used to determine all the parameters that can be eliminated or grouped. The remaining parameters constitute the set of base inertial parameters of the links. The grouped relations make use of closed-form symbolic expressions.

For $j = n, ..., 1$:

1) use the general grouping relations [9.53] or [9.56] to group:
   a) YY_j, MZ_j and M_j if joint j is revolute ($\sigma_j = 0$);
   b) XX_j, XY_j, XZ_j, YY_j, YZ_j and ZZ_j if joint j is prismatic ($\sigma_j = 1$);
2) if joint j is prismatic and $\mathbf{a}_j$ is parallel to $\mathbf{a}_{r1}$, for $r_1 < j < r_2$, then eliminate MZ_j and group MX_j and MY_j using relation [9.60];
3) if joint j is prismatic and $\mathbf{a}_j$ is not parallel to $\mathbf{a}_{r1}$, for $r_1 < j < r_2$, then group or eliminate one of the parameters MX_j, MY_j, MZ_j using Table 9.2;
4) if joint j is revolute, for $r_1 \leq j < r_2$, then eliminate XX_j, XY_j, XZ_j and YZ_j. Note that the axis of this joint is parallel to the axis of joint r_1, and that the parameter YY_j has been eliminated by rule 1;
5) if joint j is revolute, for $r_1 \leq j < r_2$, and the $\mathbf{a}_j$ axis is along $\mathbf{a}_{r1}$, and if $\mathbf{a}_{r1}$ is parallel to $\mathbf{a}_i$ and to gravity $\mathbf{g}$, for all $i < j$, then eliminate the parameters MX_j, MY_j. Note that MZ_j is eliminated by rule 1;
6) if joint j is prismatic and $j < r_1$, then eliminate the parameters MX_j, MY_j, MZ_j.

From this algorithm, we deduce that the number of minimum inertial parameters of the links for a general serial robot (without considering the inertia of the rotors) is given by:

$$b_m \leq 7\, n_r + 4\, n_p - 3 - \bar{\sigma}_1 - 2\, n_{g0} \qquad [9.61]$$

with:

- n_r: number of revolute joints = $\sum \bar{\sigma}_j$;
- n_p: number of prismatic joints = $\sum \sigma_j$;
- $n_{g0} = 1$ if the first joint is revolute and parallel to gravity, $n_{g0} = 0$ otherwise.

This equation gives in most cases the exact number of base inertial parameters.

9.4.2.5. *Considering the inertia of rotors*

Considering the rotor inertial parameter Ia_j, the number of standard inertial parameters per link becomes equal to 11. The corresponding components in the matrices $\mathbf{h}_j$ and $\mathbf{K}_j$ are given as:

$$h_{11,j} = \frac{1}{2}\dot{q}_j^2 \qquad [9.62]$$

$$K_{11,j} = Ia_j \qquad [9.63]$$

To verify if a parameter Ia_j can be grouped with other parameters, let us look for the existence of a linear relationship between $h_{11,j}$ and the other energy functions $\{h_k\}$. Such a relationship always holds for $j = r_1$, and exists under some conditions for $j = r_2$, and for the first prismatic joint denoted as p_1:

i) for joint r_1, we deduce that:

$$h_{6,r1} = \frac{1}{2}\dot{q}_{r1}^2 \qquad [9.64]$$

Consequently, the parameter Ia_{r1} can be grouped with ZZ_{r1} using the relation:

$$ZZR_{r1} = ZZ_{r1} + Ia_{r1} \qquad [9.65]$$

ii) if the axis of joint r_2 is orthogonal to that of r_1, we obtain:

$$h_{6,r2} = \frac{1}{2}\dot{q}_{r2}^2 \qquad [9.66]$$

Thus, the parameter Ia_{r2} can be grouped with ZZ_{r2} using the relation:

$$ZZR_{r2} = ZZ_{r2} + Ia_{r2} \qquad [9.67]$$

iii) if the axis of the first prismatic joint p_1 is orthogonal to gravity, and if $p_1 = 1$ or its axis is aligned with the revolute axes preceding it (such that M_1 has no effect on the gravity force of joints $1, \ldots, p_1$), then we can group Ia_{p1} with M_{p1}:

$$MR_{p1} = M_{p1} + Ia_{p1} \qquad [9.68]$$

• **Example 9.2.** Find the base inertial parameters of the Stäubli RX-90 robot. For this robot, $r_1 = 1$ and $r_2 = 2$. Since all the joints are revolute, the use of equation [9.53] for $j = n, \ldots, 1$ gives all the grouped parameters:

link 6:

$$XXR_6 = XX_6 - YY_6$$
$$XXR_5 = XX_5 + YY_6$$
$$ZZR_5 = ZZ_5 + YY_6$$
$$MYR_5 = MY_5 + MZ_6$$
$$MR_5 = M_5 + M_6$$

The minimum inertial parameters of link 6 are: XXR_6, XY_6, XZ_6, YZ_6, ZZ_6, MX_6 and MY_6.

link 5:

$$XXR_5 = XX_5 + YY_6 - YY_5$$
$$XXR_4 = XX_4 + YY_5$$
$$ZZR_4 = ZZ_4 + YY_5$$
$$MYR_4 = MY_4 - MZ_5$$
$$MR_4 = M_4 + MR_5 = M_4 + M_5 + M_6$$

The minimum parameters of link 5 are: XXR_5, XY_5, XZ_5, YZ_5, ZZR_5, MX_5 and MYR_5.

link4:

$$XXR_4 = XX_4 + YY_5 - YY_4$$
$$XXR_3 = XX_3 + YY_4 + 2\,RL4\,MZ_4 + RL4^2\,(M_4 + M_5 + M_6)$$
$$ZZR_3 = ZZ_3 + YY_4 + 2\,RL4\,MZ_4 + RL4^2\,(M_4 + M_5 + M_6)$$
$$MYR_3 = MY_3 + MZ_4 + RL4\,(M_4 + M_5 + M_6)$$
$$MR_3 = M_3 + M_4 + M_5 + M_6$$

The minimum parameters of link 4 are: XXR_4, XY_4, XZ_4, YZ_4, ZZR_4, MX_4 and MYR_4.

link 3:

$$XXR_3 = XX_3 + YY_4 + 2\,RL4\,MZ_4 + RL4^2\,(M_4 + M_5 + M_6) - YY_3$$
$$XXR_2 = XX_2 + YY_3$$
$$XZR_2 = XZ_2 - D3\,MZ_3$$
$$YYR_2 = YY_2 + D3^2\,(M_3 + M_4 + M_5 + M_6) + YY_3$$
$$ZZR_2 = ZZ_2 + D3^2\,(M_3 + M_4 + M_5 + M_6)$$
$$MXR_2 = MX_2 + D3\,(M_3 + M_4 + M_5 + M_6)$$
$$MZR_2 = MZ_2 + MZ_3$$
$$MR_2 = M_2 + M_3 + M_4 + M_5 + M_6$$

The minimum parameters of link 3 are: XXR_3, XY_3, XZ_3, YZ_3, ZZR_3, MX_3 and MYR_3.

link 2:

$$XXR_2 = XX_2 - YY_2 - D3^2\,(M_3 + M_4 + M_5 + M_6)$$
$$ZZR_1 = ZZ_1 + YY_2 + D3^2\,(M_3 + M_4 + M_5 + M_6) + YY_3$$

The minimum parameters of link 2 are: XXR_2, XY_2, XZR_2, YZ_2, ZZR_2, MXR_2 and MY_2.

link 1: from § 9.4.2.4 (rules 4 and 5), we deduce that the parameters XX_1, XY_1, XZ_1, YY_1, YZ_1, MX_1, MY_1, MZ_1 and M_1 have no effect on the dynamic model. Note that MZ_2 and M_2 have no effect because they are grouped with parameters having no effect. The only parameter of link 1 is ZZR_1.

Rotor inertia. From § 9.4.2.5, we can group the parameter Ia_1 with ZZ_1 and the parameter Ia_2 with ZZ_2:

$$ZZR_1 = ZZ_1 + Ia_1 + YY_2 + D3^2 (M_3 + M_4 + M_5 + M_6) + YY_3$$
$$ZZR_2 = ZZ_2 + Ia_2 + D3^2 (M_3 + M_4 + M_5 + M_6)$$

The final result can be summarized as follows:

- the parameters that have no effect on the dynamic model are: XX_1, XY_1, XZ_1, YY_1, YZ_1, MX_1, MY_1, MZ_1, M_1, MZ_2 and M_2;
- the parameters that are grouped are: Ia_1, YY_2, Ia_2, YY_3, MZ_3, M_3, YY_4, MZ_4, M_4,YY_5, MZ_5, M_5, YY_6, MZ_6 and M_6;
- the grouping equations are:

$$ZZR_1 = ZZ_1 + Ia_1 + YY_2 + D3^2 (M_3 + M_4 + M_5 + M_6) + YY_3$$
$$XXR_2 = XX_2 - YY_2 - D3^2 (M_3 + M_4 + M_5 + M_6)$$
$$XZR_2 = XZ_2 - D3\, MZ_3$$
$$ZZR_2 = ZZ_2 + Ia_2 + D3^2 (M_3 + M_4 + M_5 + M_6)$$
$$MXR_2 = MX_2 + D3 (M_3 + M_4 + M_5 + M_6)$$
$$XXR_3 = XX_3 - YY_3 + YY_4 + 2\,RL4\,MZ_4 + RL4^2 (M_4 + M_5 + M_6)$$
$$ZZR_3 = ZZ_3 + YY_4 + 2RL4\,MZ_4 + RL4^2 (M_4 + M_5 + M_6)$$
$$MYR_3 = MY_3 + MZ_4 + RL4 (M_4 + M_5 + M_6)$$
$$XXR_4 = XX_4 + YY_5 - YY_4$$
$$ZZR_4 = ZZ_4 + YY_5$$
$$MYR_4 = MY_4 - MZ_5$$
$$XXR_5 = XX_5 + YY_6 - YY_5$$
$$ZZR_5 = ZZ_5 + YY_6$$
$$MYR_5 = MY_5 + MZ_6$$
$$XXR_6 = XX_6 - YY_6$$

Table 9.3 gives the 40 base inertial parameters of the Stäubli RX-90 robot.

• **Example 9.3.** Let us assume that the inertia tensors ${}^j\mathbf{J}_j$, for j = 1, ..., 6, are diagonal and that the first moments of the links are given by:

$${}^1\mathbf{MS}_1 = [\,0 \;\; 0 \;\; MZ1\,]^T$$
$${}^2\mathbf{MS}_2 = [\,MX2 \;\; MY2 \;\; 0\,]^T$$

$^3\mathbf{MS}_3 = [\,0 \quad MY3 \quad 0\,]^T$
$^4\mathbf{MS}_4 = [\,0 \quad 0 \quad MZ4\,]^T$
$^5\mathbf{MS}_5 = [\,0 \quad MY5 \quad 0\,]^T$
$^6\mathbf{MS}_6 = [\,0 \quad 0 \quad MZ6\,]^T$

The corresponding 19 base inertial parameters are given in Table 9.4. They are derived using the general grouping relations after eliminating the parameters whose values are zero.

Table 9.3. *Base inertial parameters of the Stäubli RX-90 robot*

j	XX_j	XY_j	XZ_j	YY_j	YZ_j	ZZ_j	MX_j	MY_j	MZ_j	M_j	Ia_j
1	0	0	0	0	0	ZZR_1	0	0	0	0	0
2	XXR_2	XY_2	XZR_2	0	YZ_2	ZZR_2	MXR_2	MY_2	0	0	0
3	XXR_3	XY_3	XZ_3	0	YZ_3	ZZR_3	MX_3	MYR_3	0	0	Ia_3
4	XXR_4	XY_4	XZ_4	0	YZ_4	ZZR_4	MX_4	MYR_4	0	0	Ia_4
5	XXR_5	XY_5	XZ_5	0	YZ_5	ZZR_5	MX_5	MYR_5	0	0	Ia_5
6	XXR_6	XY_6	XZ_6	0	YZ_6	ZZ_6	MX_6	MY_6	0	0	Ia_6

Table 9.4. *Simplified base inertial parameters for the Stäubli RX-90 robot*

j	XX_j	XY_j	XZ_j	YY_j	YZ_j	ZZ_j	MX_j	MY_j	MZ_j	M_j	Ia_j
1	0	0	0	0	0	ZZR_1	0	0	0	0	0
2	XXR_2	0	0	0	0	ZZR_2	MXR_2	MY_2	0	0	0
3	XXR_3	0	0	0	0	ZZR_3	0	MYR_3	0	0	Ia_3
4	XXR_4	0	0	0	0	ZZR_4	0	0	0	0	Ia_4
5	XXR_5	0	0	0	0	ZZR_5	0	MYR_5	0	0	Ia_5
6	XXR_6	0	0	0	0	ZZ_6	0	0	0	0	Ia_6

9.5. Newton-Euler formulation

9.5.1. *Introduction*

The Newton-Euler equations describing the forces and moments (wrench) acting on the center-of-mass of link j are given as:

$$\mathbf{F}_j = M_j \dot{\mathbf{V}}_{Gj} \qquad [9.69]$$

$$\mathbf{M}_{Gj} = \mathbf{I}_{Gj} \dot{\omega}_j + \omega_j \times (\mathbf{I}_{Gj} \omega_j) \qquad [9.70]$$

The Newton-Euler algorithm of Luh, Walker and Paul [Luh 80b], which is considered as one of the most efficient algorithms for real time computation of the inverse dynamic model, consists of two recursive computations: forward recursion and backward recursion. The forward recursion, from the base to the terminal link, computes the link velocities and accelerations and consequently the dynamic wrench on each link. The backward recursion, from the terminal link to the base, provides the reaction wrenches on the links and consequently the joint torques.

This method gives the joint torques as defined by equation [9.1] without explicitly computing the matrices **A**, **C** and **Q**. The model obtained is not linear in the inertial parameters because it makes use of M_j, $\mathbf{S}_j$ and $\mathbf{I}_{Gj}$.

9.5.2. *Newton-Euler inverse dynamics linear in the inertial parameters*

In this section, we develop a Newton-Euler algorithm based on the double recursive computations of Luh et al. [Luh 80b], but which uses as inertial parameters the elements of $\mathbf{J}_j$, $\mathbf{MS}_j$ and M_j [Khalil 87b], [Khosla 86]. The dynamic wrench on link j is calculated on O_j and not on the center of gravity G_j. Therefore, the resulting model is linear in the dynamic parameters. This reformulation allows us to compute the dynamic model in terms of the base inertial parameters and to use it for the identification of the dynamic parameters.

The Newton-Euler equations giving the forces and moments of link j at the origin of frame R_j are given as:

$$\mathbf{F}_j = M_j \dot{\mathbf{V}}_j + \dot{\omega}_j \times \mathbf{MS}_j + \omega_j \times (\omega_j \times \mathbf{MS}_j) \qquad [9.71]$$

$$\mathbf{M}_j = \mathbf{J}_j \dot{\omega}_j + \omega_j \times (\mathbf{J}_j \omega_j) + \mathbf{MS}_j \times \dot{\mathbf{V}}_j \qquad [9.72]$$

Using the spatial notation, we can write these equations as:

$$\mathbb{F}_j = \mathbb{J}_j \dot{\mathbb{V}}_j + \begin{bmatrix} \omega_j \times (\omega_j \times MS_j) \\ \omega_j \times (J_j \omega_j) \end{bmatrix} \quad [9.73]$$

where $\mathbb{F}_j = \begin{bmatrix} F_j \\ M_j \end{bmatrix}$, $\dot{\mathbb{V}} = \begin{bmatrix} \dot{V} \\ \dot{\omega} \end{bmatrix}$, $\mathbb{J}_j$ is the spatial inertia matrix (equation [9.21]).

i) forward recursive computation: to compute F_j and M_j, for j = 1, ..., n, using equations [9.71] and [9.72], we need ω_j, $\dot{\omega}_j$ and $\dot{V}_j$. The velocities are given by the recursive equations [9.14] and [9.15] rewritten hereafter as:

$$\omega_j = \omega_{j-1} + \bar{\sigma}_j \dot{q}_j a_j \quad [9.74]$$

$$V_j = V_{j-1} + \omega_{j-1} \times L_j + \sigma_j \dot{q}_j a_j \quad [9.75]$$

Differentiating equations [9.74] and [9.75] with respect to time gives:

$$\dot{\omega}_j = \dot{\omega}_{j-1} + \bar{\sigma}_j (\ddot{q}_j a_j + \omega_{j-1} \times \dot{q}_j a_j) \quad [9.76]$$

$$\dot{V}_j = \dot{V}_{j-1} + \dot{\omega}_{j-1} \times L_j + \omega_{j-1} \times (\omega_{j-1} \times L_j) + \sigma_j (\ddot{q}_j a_j + 2 \omega_{j-1} \times \dot{q}_j a_j) \quad [9.77]$$

The initial conditions for a robot with a fixed base are $\omega_0 = 0$, $\dot{\omega}_0 = 0$ and $\dot{V}_0 = 0$;

ii) backward recursive computation: this is based on writing for each link j, for j = n,...,1, the Newton-Euler equations at the origin O_j, as follows (Figure 9.5):

$$F_j = f_j - f_{j+1} + M_j g - f_{ej} \quad [9.78]$$

$$M_j = m_j - m_{j+1} - L_{j+1} \times f_{j+1} + S_j \times M_j g - m_{ej} \quad [9.79]$$

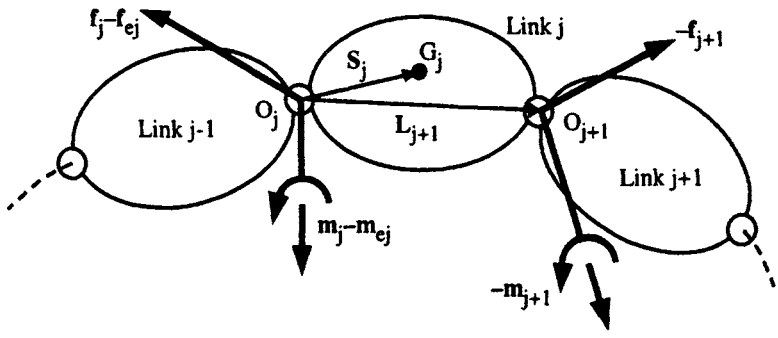

Figure 9.5. *Forces and moments on link j*

We note that $\mathbf{f}_{ej}$ and $\mathbf{m}_{ej}$, which represent the force and moment exerted by link j on the environment, may include contributions from springs, dampers, contact with the environment, etc. Their values are assumed to be known, or at least to be calculated from known quantities.

We can cancel the gravity terms from equations [9.78] and [9.79] and take into account their effects by setting up the initial linear acceleration such that:

$$\dot{\mathbf{V}}_0 = -\mathbf{g} \qquad [9.80]$$

Thus, using equations [9.78] and [9.79], we obtain:

$$\mathbf{f}_j = \mathbf{F}_j + \mathbf{f}_{j+1} + \mathbf{f}_{ej} \qquad [9.81]$$

$$\mathbf{m}_j = \mathbf{M}_j + \mathbf{m}_{j+1} + \mathbf{L}_{j+1} \times \mathbf{f}_{j+1} + \mathbf{m}_{ej} \qquad [9.82]$$

This backward recursive algorithm is initialized by $\mathbf{f}_{n+1} = \mathbf{0}$ and $\mathbf{m}_{n+1} = \mathbf{0}$.

Finally, the joint torque Γ_j can be obtained by projecting $\mathbf{f}_j$ or $\mathbf{m}_j$ on the joint axis, depending whether the joint is prismatic or revolute respectively. We can also consider the friction forces and the rotor inertia as shown in the Lagrange method:

$$\Gamma_j = (\sigma_j \mathbf{f}_j + \bar{\sigma}_j \mathbf{m}_j)^T \mathbf{a}_j + F_{cj}\,\mathrm{sign}(\dot{q}_j) + F_{vj}\,\dot{q}_j + Ia_j\,\ddot{q}_j \qquad [9.83]$$

From equations [9.81], [9.82] and [9.83], we deduce that Γ_j is a function of the inertial parameters of links j, j+1, ..., n. This property has been outlined in § 9.3.3.3.

9.5.3. *Practical form of the Newton-Euler algorithm*

Since $\mathbf{J}_j$ and $\mathbf{MS}_j$ are constants when referred to their own link coordinates, the Newton-Euler algorithm can be efficiently computed by referring the velocities, accelerations, forces and moments to the local link coordinate system [Luh 80b]. The forward recursive equations become, for j = 1, ..., n:

$${}^j\omega_{j-1} = {}^jA_{j-1}\,{}^{j-1}\omega_{j-1} \qquad [9.84]$$

$${}^j\omega_j = {}^j\omega_{j-1} + \bar{\sigma}_j\,\dot{q}_j\,{}^ja_j \qquad [9.85]$$

$${}^j\dot{\omega}_j = {}^jA_{j-1}\,{}^{j-1}\dot{\omega}_{j-1} + \bar{\sigma}_j\,(\ddot{q}_j\,{}^ja_j + {}^j\omega_{j-1} \times \dot{q}_j\,{}^ja_j) \qquad [9.86]$$

$${}^j\dot{V}_j = {}^jA_{j-1}\,({}^{j-1}\dot{V}_{j-1} + {}^{j-1}U_{j-1}\,{}^{j-1}P_j) + \sigma_j\,(\ddot{q}_j\,{}^ja_j + 2\,{}^j\omega_{j-1} \times \dot{q}_j\,{}^ja_j) \qquad [9.87]$$

$${}^jF_j = M_j\,{}^j\dot{V}_j + {}^jU_j\,{}^jMS_j \qquad [9.88]$$

$${}^jM_j = {}^jJ_j\,{}^j\dot{\omega}_j + {}^j\omega_j \times ({}^jJ_j\,{}^j\omega_j) + {}^jMS_j \times {}^j\dot{V}_j \qquad [9.89]$$

$$ {}^{j}U_j = {}^{j}\hat{\dot{\omega}}_j + {}^{j}\hat{\omega}_j \, {}^{j}\hat{\omega}_j \qquad [9.90] $$

For a stationary base, the initial conditions are $\omega_0 = 0$, $\dot{\omega}_0 = 0$ and $\dot{V}_0 = -g$.

The use of ${}^{j}U_j$ saves 21n multiplications and 6n additions in the computation of the inverse dynamic model of a general robot [Kleinfinger 86a].
It is to be noted that ${}^{j}\dot{V}_j$ means ${}^{j}A_0 {}^{0}\dot{V}_j$, and not the time derivative of ${}^{j}V_j$, since ${}^{j}\dot{V}_j = \frac{d}{dt} {}^{j}V_j + {}^{j}\omega_j \times {}^{j}V_j$. On the contrary, ${}^{j}\dot{\omega}_j$ is equal to the time derivative of ${}^{j}\omega_j$.

The backward recursive equations, for j = n, ..., 1, are:

$$ {}^{j}f_j = {}^{j}F_j + {}^{j}f_{j+1} + {}^{j}f_{ej} \qquad [9.91] $$

$$ {}^{j-1}f_j = {}^{j-1}A_j \, {}^{j}f_j \qquad [9.92] $$

$$ {}^{j}m_j = {}^{j}M_j + {}^{j}A_{j+1} \, {}^{j+1}m_{j+1} + {}^{j}P_{j+1} \times {}^{j}f_{j+1} + {}^{j}m_{ej} \qquad [9.93] $$

$$ \Gamma_j = (\sigma_j \, {}^{j}f_j + \bar{\sigma}_j \, {}^{j}m_j)^T \, {}^{j}a_j + F_{sj} \, \mathrm{sign}(\dot{q}_j) + F_{vj} \, \dot{q}_j + Ia_j \, \ddot{q}_j \qquad [9.94] $$

The previous algorithm can be numerically programmed for a general serial robot. Its computational complexity is $O(n)$, which means that the number of operations is linear in the number of degrees of freedom. However, as we will see in the next section, the use of the base inertial parameters in a customized symbolic algorithm considerably reduces the number of operations of the dynamic model.

9.6. Real time computation of the inverse dynamic model

9.6.1. *Introduction*

Much work has been focused on the problem of computational efficiency of the inverse dynamic model of robots in order to realize real time dynamic control. This objective is now recognized as being attained (Table 9.5).

Before describing our method, which is based on a customized symbolic Newton-Euler formulation linear in the inertial parameters [Khalil 87b], [Kleinfinger 86a], we review the main approaches presented in the literature.

The Lagrangian formulation of Uicker and Kahn [Uicker 69], [Kahn 69] served as a standard robot dynamics formulation for over a decade. In this form, the complexity of the equations precluded the real time computation of the inverse dynamic model. For example, for a six degree-of-freedom robot, this formulation requires 66271 multiplications and 51548 additions [Hollerbach 80]. This led researchers to investigate either simplification or tabulation approaches to achieve real time implementation.

The first approach towards simplification is to neglect the Coriolis and centrifugal terms with the assumption that they are not significant except at high speeds [Paul 72], [Bejczy 74]. Unfortunately, this belief is not valid: firstly, Luh et al. [Luh 80b] have shown that such simplifications leads to large errors not only in

the value of the computed joint torques but also in its sign; later, Hollerbach [Hollerbach 84a] has shown that the velocity terms have the same significance relative to the acceleration terms whatever the velocity.

An alternative simplification approach has been proposed by Bejczy [Bejczy 79]. This approach is based on an exhaustive term-by-term analysis of the elements A_{ij}, $C_{i,jk}$ and Q_i so that the most significant terms are only retained. A similar procedure has been utilized by Armstrong et al. [Armstrong 86] who also proposed computing the elements A_{ij}, $C_{i,jk}$ and Q_i with a low frequency rate with respect to that of the servo rate. Using such an analysis for a six degree-of-freedom robot becomes very laborious and tedious.

In the tabulation approach, some terms of the dynamic equations are precomputed and tabulated. The combination of a look-up table with reduced analytical computations renders them feasible in real time. Two methods based on a trade-off between memory space and computation time have been investigated by Raibert [Raibert 77]. In the first method *SSM (State Space Model)*, the table was carried out as a function of the joint positions and velocities ($\mathbf{q}$ and $\dot{\mathbf{q}}$), but the required memory was too big to consider in real applications at that time. In the *Configuration Space Method (CSM)*, the table is computed as a function of the joint positions. Another technique, proposed by Aldon [Aldon 82] consists of tabulating the elements A_{ij} and Q_i and of calculating the elements $C_{i,jk}$ on-line in terms of the A_{ij} elements. This method considerably reduces the required memory but increases the number of on-line operations, which becomes proportional to n^3. We note that the tabulated elements are functions of the load inertial parameters, which means making a table for each load.

Luh et al. [Luh 80a] proposed to determine the inverse dynamic model using a Newton-Euler formulation (§ 9.5). The complexity of this method is $O(n)$. This method has proved the importance of the recursive computations and the organization of the different steps of the dynamic algorithm. Therefore, other researchers tried to improve the existing Lagrange formulations by introducing recursive computations. For example, Hollerbach [Hollerbach 80] proposed a new recursive Lagrange formulation with complexity $O(n)$, whereas Megahed [Megahed 84] developed a new recursive computational procedure for the Lagrange method of Uicker and Kahn. However, these methods are less efficient than the Newton-Euler formulation of Luh et al. [Luh 80a].

More recently, researchers investigated alternative formulations [Kane 83], [Vukobratovic 85], [Renaud 85], [Kazerounian 86], but the recursive Newton-Euler proved to be computationally more efficient.

The most efficient models proposed until now are based on a customized symbolic Newton-Euler formulation that takes into account the particularity of the geometric and inertial parameters of each robot [Kanade 84], [Khalil 85a], [Khalil 87b], [Renaud 87]. Moreover, the use of the base inertial parameters in this algorithm reduces the computational cost by about 25%. We note that the number of operations for this method is even less than that of the tabulated CSM method.

Table 9.5. *Computational cost of the inverse dynamic modeling methods*

Method	Robot	General case		2RP(3R) Stanford		R(2P)(3R) TH8		6R Stäubli RX-90	
	Operations	n ddl	n=6	General	Simplified*	General	Simplified*	General	Simplified*
Raibert 78**	Mult.	n^3+2n^2	288	288	288	288	288	288	288
	Add.	3^n+n^2	252	252	252	252	252	252	252
Luh 80b	Mult.	137n-22	800	800	800	800	800	800	800
	Add.	101n-11	595	595	595	595	595	595	595
Hollerbach 80**	Mult.	412n-277	2195	2195	2195	2195	2195	2195	2195
	Add.	320n-201	1719	1719	1719	1719	1719	1719	1719
Kane 83**	Mult.	?	?	646	?	?	?	?	?
	Add.	?	?	394	?	?	?	?	?
Vukobratovic 85**	Mult.	?	?	?	>372	?	>193	?	?
	Add.	?	?	?	>167	?	>80	?	?
Renaud 85**	Mult.	?	?	?	?	?	?	?	368
	Add.	?	?	?	?	?	?	?	271
Presented method Khalil [87b]	Mult.	92n-127	425	240	142	175	104	253	159
	Add.	81n-117	369	227	99	162	72	238	113

? the number of operations is not given.

* the matrix J_j is diagonal and two components of the first moments are zero.

** forces and moments exerted by the end-effector on the environment are not considered.

> the given number of operations corresponds to the computation of the elements of A, C_{ijk} and Q.

Before closing this section, it is worth noting the formidable technological progress in the field of computer processors, to the point that the dynamic model can be calculated at control rate with standard personal computers (Chapter 14).

9.6.2. *Customization of the Newton-Euler formulation*

The recursive Newton-Euler formulation of robot dynamics has become a standard algorithm for real time control and simulation (§ 9.5.3). To increase the efficiency of the Newton-Euler algorithm, we generate a customized symbolic model for each specific robot and utilize the base dynamic parameters. To obtain this model, we expand the recursive equations to transform them into scalar equations without incorporating loop computations. The elements of a vector or a matrix containing at least one mathematical operation are replaced by an intermediate variable. This variable is written in the output file, which contains the customized model [Kanade 84], [Khalil 85a]. The elements that do not contain any operation are not modified. We propagate the obtained vectors and matrices in the subsequent equations. Consequently, customizing eliminates multiplications by one (and minus one) and zero, and additions with zero. A good choice of the intermediate variables allows us to avoid redundant computations. In the end, the dynamic model is obtained as a set of intermediate variables. Those that have no effect on the desired output, the joint torques in the case of inverse dynamics, can be eliminated by scanning the intermediate variables from the end to the beginning.

The customization technique allows us to reduce the computational load for a general robot, but this reduction is larger when carried out for a specific robot [Kleinfinger 86a]. The computational efficiency in customization is obtained at the cost of a software symbolic iterative structure [Khalil 97] and a relatively increased program memory requirement.

• **Example 9.4.** To illustrate how to generate a customized symbolic model, we develop in this example the computation of the link angular velocities ${}^j\omega_j$ for the Stäubli RX-90 robot. The computation of the orientation matrices ${}^{j-1}A_j$ (Example 3.3) generates the 12 sinus and cosinus intermediate variables:

$$\begin{aligned} Sj &= \sin(q_j) \\ Cj &= \cos(q_j) \qquad \text{for } j = 1, \ldots, 6 \end{aligned}$$

The computation of the angular velocities for $j = 1, \ldots, 6$ is given as:

$${}^1\omega_1 = \begin{bmatrix} 0 \\ 0 \\ QP1 \end{bmatrix}$$

Computation of ${}^1\omega_1$ does not generate any intermediate variable.

$$ {}^2\omega_1 = \begin{bmatrix} S2*QP1 \\ C2*QP1 \\ 0 \end{bmatrix} = \begin{bmatrix} WI12 \\ WI22 \\ 0 \end{bmatrix} $$

Computation of ${}^2\omega_2$ generates the following intermediate variables:

$$ \begin{aligned} WI12 &= S2*QP1 \\ WI22 &= C2*QP1 \end{aligned} $$

In the following, the vector ${}^2\omega_2$ is set as:

$$ {}^2\omega_2 = \begin{bmatrix} WI12 \\ WI22 \\ QP2 \end{bmatrix} $$

Continuing the recursive computation leads to:

$$ {}^3\omega_2 = \begin{bmatrix} C3*WI12 + S3*WI22 \\ -S3*WI12 + C3*WI22 \\ QP2 \end{bmatrix} = \begin{bmatrix} WI13 \\ WI23 \\ QP2 \end{bmatrix} $$

$$ {}^3\omega_3 = \begin{bmatrix} WI13 \\ WI23 \\ QP2 + QP3 \end{bmatrix} = \begin{bmatrix} WI13 \\ WI23 \\ W33 \end{bmatrix} $$

$$ {}^4\omega_3 = \begin{bmatrix} C4*WI13 - S4*W33 \\ -S4*WI12 - C4*W33 \\ WI23 \end{bmatrix} = \begin{bmatrix} WI14 \\ WI24 \\ WI23 \end{bmatrix} $$

$$ {}^4\omega_4 = \begin{bmatrix} WI14 \\ WI24 \\ WI23 + QP4 \end{bmatrix} = \begin{bmatrix} WI14 \\ WI24 \\ W34 \end{bmatrix} $$

$$ {}^5\omega_4 = \begin{bmatrix} C5*WI14 + S5*W34 \\ -S5*WI14 + C5*W34 \\ -WI24 \end{bmatrix} = \begin{bmatrix} WI15 \\ WI25 \\ -WI24 \end{bmatrix} $$

$$ {}^5\omega_5 = \begin{bmatrix} WI15 \\ WI25 \\ -WI24 + QP5 \end{bmatrix} = \begin{bmatrix} WI15 \\ WI25 \\ W35 \end{bmatrix} $$

$$^6\omega_5 = \begin{bmatrix} C6*WI15 - S6*W35 \\ -S6*WI15 - C6*W35 \\ WI25 \end{bmatrix} = \begin{bmatrix} WI16 \\ WI26 \\ WI25 \end{bmatrix}$$

$$^6\omega_6 = \begin{bmatrix} WI16 \\ WI26 \\ WI25 + QP6 \end{bmatrix} = \begin{bmatrix} WI16 \\ WI26 \\ W36 \end{bmatrix}$$

Finally, the computation of $^j\omega_j$, for j = 1, ..., 6, requires the following intermediate variables: WI12, WI22, WI13, WI23, W33, WI14, WI24, W34, WI15, WI25, W35, WI16, WI26 and W36, in addition to the variables Sj and Cj for j = 2, ..., 6. The variables S1 and C1 can be eliminated because they have no effect on the angular velocities.

9.6.3. *Utilization of the base inertial parameters*

It is obvious that the use of the base inertial parameters in a customized Newton-Euler formulation that is linear in the inertial parameters will reduce the number of operations because the parameters that have no effect on the model or have been grouped are set equal to zero. Practically, the number of operations of the inverse dynamic model when using the base inertial parameters for a general n revolute degree-of-freedom robot is 92n – 127 multiplications and 81n – 117 additions (n > 2), which gives 425 multiplications and 369 additions for n = 6. By general robot, we mean:

- the geometric parameters r_1, d_1, α_1 and r_n are zero (this assumption holds for any robot);
- the other geometric parameters, all the inertial parameters, and the forces and moments exerted by the terminal link on the environment can have an arbitrary real value;
- the friction forces are assumed to be negligible, otherwise, with a Coulomb and viscous friction model, we add n multiplications, 2n additions, and n sign functions.

Table 9.6. shows the computational complexity of the inverse dynamic model for the Stäubli RX-90 robot using the customized Newton-Euler formulation. In Appendix 7, we give the dynamic model of the Stäubli RX-90 robot when using the base inertial parameters of Table 9.4, which takes into account the symmetry of the links. The corresponding computational cost is 160 multiplications and 113 additions.

Table 9.6. *Computational complexity of the inverse dynamic model for the Stäubli RX-90 robot*

	Inertial parameters	Multiplications	Additions
General inertial parameters	Standard parameters	294	283
	Base parameters	253	238
Simplified inertial parameters	Standard parameters	202	153
	Base parameters	160	113

9.7. 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 synthesis of the control laws. During simulation, the dynamic equations are solved for the joint accelerations given the input torques and the current state of the robot (joint positions and velocities). Through integration of the joint accelerations, the robot trajectory is then determined. Although the simulation may be carried out off-line, it is interesting to have an efficient direct dynamic model to reduce the simulation time. In this section, we consider two methods: the first is based on using a specialized Newton-Euler inverse dynamic model and needs to compute the inverse of the inertia matrix **A** of the robot; the second method is based on a recursive Newton-Euler algorithm that does not explicitly use the matrix **A** and has a computational cost that varies linearly with the number of degrees of freedom of the robot.

9.7.1. *Using the inverse dynamic model to solve the direct dynamic problem*

From equation [9.6], we can express the direct dynamic problem as the solution of the joint accelerations from the following equation:

$$\mathbf{A}\,\ddot{\mathbf{q}} = [\Gamma - \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}})] \qquad [9.95]$$

where **H** contains the Coriolis, centrifugal, gravity, friction and external torques:

$$\mathbf{H}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\,\dot{\mathbf{q}} + \mathbf{Q} + \mathbf{diag}(\dot{\mathbf{q}})\mathbf{F}_v + \mathbf{diag}[\mathrm{sign}(\dot{\mathbf{q}})]\mathbf{F}_c + \mathbf{J}^T \mathbb{f}_{en}$$

Although in practice we do not explicitly calculate the inverse of the matrix **A**, the solution of equation [9.95] is generally denoted by:

$$\ddot{\mathbf{q}} = \mathbf{A}^{-1} [\Gamma - \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}})] \qquad [9.96]$$

The computation of the direct dynamics can be broken down into three steps: the calculation of $\mathbf{H}(\mathbf{q}, \dot{\mathbf{q}})$, the calculation of $\mathbf{A}$, and the solution of the linear equation [9.95] for $\ddot{\mathbf{q}}$.

The computational complexity of the first step is minimized by the use of a specialized version of the inverse dynamics algorithm in which the desired joint accelerations are zero [Walker 82]. By comparing equations [9.1] and [9.95], we deduce that $\mathbf{H}(\mathbf{q}, \dot{\mathbf{q}})$ is equal to Γ if $\ddot{\mathbf{q}} = \mathbf{0}$.

The inertia matrix can also be calculated one column at a time, using Newton-Euler inverse dynamic model [Walker 82]. From relation [9.95], we deduce that the i^{th} column of $\mathbf{A}$ is equal to Γ if:

$$\ddot{\mathbf{q}} = \mathbf{u}_i, \dot{\mathbf{q}} = \mathbf{0}, \mathbf{g} = \mathbf{0}, \mathbf{F}_c = \mathbf{0} \ (\mathbf{f}_{ej} = \mathbf{0}, \mathbf{m}_{ej} = \mathbf{0} \quad \text{for } j = 1,\ldots, n) \qquad [9.97]$$

where $\mathbf{u}_i$ is an (nx1) unit vector with 1 in the i^{th} row and zeros elsewhere. 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 $\mathbf{A}$ is symmetric. A more efficient procedure for computing the inertia matrix using the concept of composite links is given in Appendix 8. Alternative efficient approaches for computing the inertia matrix based on the Lagrange formulation are proposed in [Megahed 82], [Renaud 85].

NOTE.– The nonlinear state equation of a robot follows from relation [9.95] as:

$$\begin{bmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{q}} \\ \mathbf{A}^{-1} [\Gamma - \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}})] \end{bmatrix} \qquad [9.98]$$

and the output equation is written as:

$$\mathbf{y} = \mathbf{q} \text{ or } \mathbf{y} = \mathbf{X}(\mathbf{q}) \qquad [9.99]$$

In this formulation, the state variables are given by $[\mathbf{q}^T \ \dot{\mathbf{q}}^T]^T$, the equation $\mathbf{y} = \mathbf{q}$ gives the output vector in the joint space, and $\mathbf{y} = \mathbf{X}(\mathbf{q})$ denotes the coordinates of the end-effector frame in the task space.

9.7.2. *Recursive 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 79], [Featherstone 83b], [Brandl 86]. In this section, we utilize the compact spatial notation, which is also called *screw notation*. Consequently, by combining equations [9.87] and [9.86], which give ${}^j\dot{\mathbf{V}}_j$ and ${}^j\dot{\boldsymbol{\omega}}_j$, we obtain:

$$ {}^j\dot{\mathbb{V}}_j = {}^j\mathbb{T}_{j-1}\,{}^{j-1}\dot{\mathbb{V}}_{j-1} + \ddot{q}_j\,{}^j\mathbb{a}_j + {}^j\gamma_j \qquad [9.100] $$

where ${}^j\mathbb{a}_j$ is defined by equation [9.23b], and:

$$ {}^j\gamma_j = \begin{bmatrix} {}^jA_{j-1}[{}^{j-1}\omega_{j-1} \times ({}^{j-1}\omega_{j-1} \times {}^{j-1}P_j)] + 2\sigma_j ({}^j\omega_{j-1} \times \dot{q}_j\,{}^ja_j) \\ \bar{\sigma}_j\,{}^j\omega_{j-1} \times \dot{q}_j\,{}^ja_j \end{bmatrix} \qquad [9.101] $$

Equations [9.88], [9.89], [9.91] and [9.93], which represent the equilibrium equations of link j, can be combined as:

$$ {}^j\mathbb{J}_j\,{}^j\dot{\mathbb{V}}_j = {}^j\mathbb{f}_j - {}^{j+1}\mathbb{T}_j^T\,{}^{j+1}\mathbb{f}_{j+1} + {}^j\beta_j \qquad [9.102] $$

where:

$$ {}^j\beta_j = -{}^j\mathbb{f}_{ej} - \begin{bmatrix} {}^j\omega_j \times ({}^j\omega_j \times {}^jMS_j) \\ {}^j\omega_j \times ({}^jJ_j\,{}^j\omega_j) \end{bmatrix} \qquad [9.103] $$

In equation [9.102], we use equation [2.63] to transform the dynamic wrench from frame R_{j+1} to frame R_j.

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

i) first forward recursive computations for j = 1, ..., n: in this step, we compute the screw transformation matrices ${}^j\mathbb{T}_{j-1}$, the link angular velocities ${}^j\omega_j$ as well as ${}^j\gamma_j$ and ${}^j\beta_j$ vectors, which represent the link accelerations and the link wrenches respectively when $\ddot{q} = 0$;

ii) backward recursive computations for j = n, ..., 1: we compute the vectors and matrices needed to express the joint acceleration $\ddot{q}_j$ and the wrench ${}^j\mathbb{f}_j$ in terms of ${}^{j-1}\dot{\mathbb{V}}_{j-1}$. To illustrate the equations required, we detail the case when $j = n$ and $j = n - 1$. By combining equations [9.100] and [9.102] for $j = n$, and since ${}^{n+1}\mathbb{f}_{n+1} = \mathbf{0}$, we obtain:

$$ {}^{n}\mathbb{J}_n \left({}^{n}\mathbb{T}_{n-1}\, {}^{n-1}\dot{\mathbb{V}}_{n-1} + \ddot{q}_n\, {}^{n}\mathbb{a}_n + {}^{n}\gamma_n \right) = {}^{n}\mathbb{f}_n + {}^{n}\beta_n \qquad [9.104] $$

Since:

$$ {}^{j}\mathbb{a}_j^{T}\, {}^{j}\mathbb{f}_j = \tau_j - Ia_j\, \ddot{q}_j \qquad [9.105] $$

$$ \tau_j = \Gamma_j - F_{sj}\, \mathrm{sign}(\dot{q}_j) - F_{vj}\, \dot{q}_j \qquad [9.106] $$

multiplying equation [9.104] by ${}^{n}\mathbb{a}_n^{T}$ and using equation [9.105], we deduce the joint acceleration of joint n:

$$ \ddot{q}_n = H_n^{-1} \left(- {}^{n}\mathbb{a}_n^{T}\, {}^{n}\mathbb{J}_n \left({}^{n}\mathbb{T}_{n-1}\, {}^{n-1}\dot{\mathbb{V}}_{n-1} + {}^{n}\gamma_n \right) + \tau_n + {}^{n}\mathbb{a}_n^{T}\, {}^{n}\beta_n \right) \qquad [9.107] $$

where H_n is a scalar given as:

$$ H_n = \left({}^{n}\mathbb{a}_n^{T}\, {}^{n}\mathbb{J}_n\, {}^{n}\mathbb{a}_n + Ia_n \right) \qquad [9.108] $$

Substituting for $\ddot{q}_n$ from equation [9.107] into equation [9.104], we obtain the dynamic wrench ${}^{n}\mathbb{f}_n$ as:

$$ {}^{n}\mathbb{f}_n = \begin{bmatrix} {}^{n}f_n \\ {}^{n}m_n \end{bmatrix} = {}^{n}\mathbb{K}_n\, {}^{n}\mathbb{T}_{n-1}\, {}^{n-1}\dot{\mathbb{V}}_{n-1} + {}^{n}\alpha_n \qquad [9.109] $$

where:

$$ {}^{n}\mathbb{K}_n = {}^{n}\mathbb{J}_n - {}^{n}\mathbb{J}_n\, {}^{n}\mathbb{a}_n\, H_n^{-1}\, {}^{n}\mathbb{a}_n^{T}\, {}^{n}\mathbb{J}_n \qquad [9.110] $$

$$ {}^{n}\alpha_n = {}^{n}\mathbb{K}_n\, {}^{n}\gamma_n + {}^{n}\mathbb{J}_n\, {}^{n}\mathbb{a}_n\, H_n^{-1} \left(\tau_n + {}^{n}\mathbb{a}_n^{T}\, {}^{n}\beta_n \right) - {}^{n}\beta_n \qquad [9.111] $$

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, from equation [9.102]:

$$ {}^{n-1}\mathbb{J}_{n-1}\, {}^{n-1}\dot{\mathbb{V}}_{n-1} = {}^{n-1}\mathbb{f}_{n-1} + {}^{n}\mathbb{T}_{n-1}^{T}\, {}^{n}\mathbb{f}_n + {}^{n-1}\beta_{n-1} \qquad [9.112] $$

which can be rewritten using equation [9.100] as:

$$ {}^{n-1}\mathbb{J}^{*}_{n-1} \left({}^{n-1}\mathbb{T}_{n-2}\, {}^{n-2}\dot{\mathbb{V}}_{n-2} + \ddot{q}_{n-1}\, {}^{n-1}\mathbb{a}_{n-1} + {}^{n-1}\gamma_{n-1} \right) = {}^{n-1}\mathbb{f}_{n-1} + {}^{n-1}\beta^{*}_{n-1} \qquad [9.113] $$

where:

$$^{n-1}\mathbb{J}^*_{n-1} = {}^{n-1}\mathbb{J}_{n-1} + {}^{n}\mathbb{T}^T_{n-1}\,{}^{n}\mathbb{K}_n\,{}^{n}\mathbb{T}_{n-1} \qquad [9.114]$$

$$^{n-1}\boldsymbol{\beta}^*_{n-1} = {}^{n-1}\boldsymbol{\beta}_{n-1} - {}^{n}\mathbb{T}^T_{n-1}\,{}^{n}\boldsymbol{\alpha}_n \qquad [9.115]$$

Equation [9.113] has the same form as equation [9.104]. Consequently, 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 $j = n-2, \ldots, 1$, we obtain $\ddot{q}_j$ and $^{j}\mathbb{f}_j$ in terms of $^{j-1}\dot{\mathbb{V}}_{j-1}$ for $j = n-1, \ldots, 1$. Since $^{0}\dot{\mathbb{V}}_0$ is composed of the linear and angular accelerations of the base that are assumed to be known ($\dot{\mathbf{V}}_0 = -\mathbf{g}$, $\dot{\boldsymbol{\omega}}_0 = \mathbf{0}$), the third recursive computation allows us to compute $\ddot{q}_j$ and $^{j}\mathbb{f}_j$ for $j = 1, \ldots, n$. These backward recursive equations are summarized as follows:

For $j = n, \ldots, 1$, compute:

$$H_j = ({}^{j}\mathbb{a}_j^T\,{}^{j}\mathbb{J}^*_j\,{}^{j}\mathbb{a}_j + Ia_j) \qquad [9.116]$$

$${}^{j}\mathbb{K}_j = {}^{j}\mathbb{J}^*_j - {}^{j}\mathbb{J}^*_j\,{}^{j}\mathbb{a}_j\,H_j^{-1}\,{}^{j}\mathbb{a}_j^T\,{}^{j}\mathbb{J}^*_j \qquad [9.117]$$

$${}^{j}\boldsymbol{\alpha}_j = {}^{j}\mathbb{K}_j\,{}^{j}\boldsymbol{\gamma}_j + {}^{j}\mathbb{J}^*_j\,{}^{j}\mathbb{a}_j\,H_j^{-1}\,(\tau_j + {}^{j}\mathbb{a}_j^T\,{}^{j}\boldsymbol{\beta}^*_j) - {}^{j}\boldsymbol{\beta}^*_j \qquad [9.118]$$

$$^{j-1}\boldsymbol{\beta}^*_{j-1} = {}^{j-1}\boldsymbol{\beta}_{j-1} - {}^{j}\mathbb{T}^T_{j-1}\,{}^{j}\boldsymbol{\alpha}_j \qquad [9.119]$$

$$^{j-1}\mathbb{J}^*_{j-1} = {}^{j-1}\mathbb{J}_{j-1} + {}^{j}\mathbb{T}^T_{j-1}\,{}^{j}\mathbb{K}_j\,{}^{j}\mathbb{T}_{j-1} \qquad [9.120]$$

Note that these equations are initialized by $^{j}\mathbb{J}^*_j = {}^{j}\mathbb{J}_j$ and that equations [9.119] and [9.120] are not calculated for $j = 1$;

iii) second forward recursive computations for $j = 1, \ldots, n$. The joint acceleration $\ddot{q}_j$ and the dynamic wrench $^{j}\mathbb{f}_j$ (if needed) are then obtained from the following equations (see equations [9.107] and [9.109]):

$${}^{j}\dot{\mathbb{V}}_{j-1} = {}^{j}\mathbb{T}_{j-1}\,{}^{j-1}\dot{\mathbb{V}}_{j-1} \qquad [9.121]$$

$$\ddot{q}_j = H_j^{-1}\,[-{}^{j}\mathbb{a}_j^T\,{}^{j}\mathbb{J}^*_j\,({}^{j}\dot{\mathbb{V}}_{j-1} + {}^{j}\boldsymbol{\gamma}_j) + \tau_j + {}^{j}\mathbb{a}_j^T\,{}^{j}\boldsymbol{\beta}^*_j] \qquad [9.122]$$

$${}^{j}\mathbb{f}_j = \begin{bmatrix} {}^{j}\mathbf{f}_j \\ {}^{j}\mathbf{m}_j \end{bmatrix} = {}^{j}\mathbb{K}_j\,{}^{j}\dot{\mathbb{V}}_{j-1} + {}^{j}\boldsymbol{\alpha}_j \qquad [9.123]$$

$${}^{j}\dot{\mathbb{V}}_j = {}^{j}\dot{\mathbb{V}}_{j-1} + {}^{j}\mathbb{a}_j\,\ddot{q}_j + {}^{j}\boldsymbol{\gamma}_j \qquad [9.124]$$

NOTES.–

- to reduce the number of operations of this algorithm, we can make use of the base inertial parameters and the customized symbolic technique. Thereby, the number of operations of the direct dynamic model for the Stäubli RX-90 robot is 889 multiplications and 653 additions [Khalil 97]. In the case of the use of simplified inertial parameters (Table 9.4), the computational cost becomes 637 multiplications and 423 additions;
- the computational complexity of this method is $O(n)$, while the method requiring the inverse of the robot inertia matrix is of complexity $O(n^3)$;
- from the numerical point of view, this method is more stable than the method requiring the inverse of the robot inertia matrix [Cloutier 95].

9.8. Conclusion

In this chapter, we have presented the dynamics of serial robots using Lagrange and Newton-Euler formulations that are linear in the inertial parameters. The Lagrange formulation allowed us to study the characteristics and properties of the dynamic model of robots, while the Newton-Euler was shown to be the most efficient for real time implementation. We have illustrated that the base inertial parameters can be determined using simple closed-form rules without calculating neither the dynamic model nor the energy functions. In order to increase the efficiency of the Newton-Euler algorithms, we have proposed the use of the base inertial parameters in a customized symbolic programming algorithm. The problem of computing the direct dynamic model for simulating the dynamics of robots has been treated using two methods; the first is based on the Newton-Euler inverse dynamic algorithm, while the second is based on another Newton-Euler algorithm that does not require the computation of the robot inertia matrix.

In the next chapter, we extend these results to tree structured and closed loop robots. Note that the inverse and direct dynamic algorithms using recursive Newton-Euler equations have been generalized to flexible robots [Boyer 98] and to systems with lumped elasticities [Khalil 00a].