Decomposition and Cross-Product-Based Method for Computing the Dynamic Equation of Robots

Regular Paper

Ching-Long Shih¹, Wen-Yo Lee²,* and Chia-Pin Wu¹

1 National Taiwan University of Science and Technology, Department of Electrical Engineering, Taiwan
2 LungHwa University of Science and Technology, Department of Information Networking Engineering, Taiwan
* Corresponding author E-mail: tristanwylee@mail.lhu.edu.tw

Received 05 May 2012; Accepted 29 May 2012

© 2012 Shih et al.; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

ARTICLE

Abstract This paper aims to demonstrate a clear relationship between Lagrange equations and Newton-Euler equations regarding computational methods for robot dynamics, from which we derive a systematic method for using either symbolic or on-line numerical computations. Based on the decomposition approach and cross-product operation, a computing method for robot dynamics can be easily developed. The advantages of this computing framework are that: it can be used for both symbolic and on-line numeric computation purposes, and it can also be applied to biped systems, as well as some simple closed-chain robot systems.

Keywords Newton-Euler equation, Lagrange formulation, robotic dynamic equation, symbolic computation, biped robot

1. Introduction

When looking at robotics research, dynamic modelling is basically a form of artwork. The complexity of a dynamic model concerns the degrees of freedom (DOFs) of a machine or a robot and the dynamic model of a low DOFs robot can be easily developed, but dynamic models for a high DOFs robot are difficult to derive by hand. The drawbacks to creating the models by hand are: difficulty of maintenance, complexity of proofreading the dynamic model and inflexibility. The biped robot is the target of a complex dynamic system study [1] that also reveals a demand for a computer-aided systematic derivation tools for robotics.

Using an auto-deriving tool to obtain a dynamic model of a robot is very economical. Tošić [2] proposed the benefits of a symbolic simulation of an engineering system, while Cetinkunt [3] proposed a symbolic modelling of the dynamics of robotic manipulators on the numerical tool REDUCE. Since the coding of REDUCE is based on C and Fortran, modifying the system model by simply changing the symbol is rather difficult. Vibet demonstrated the symbolic tool FORM [4], which automatically generates the dynamic equations of the manipulator. The MACSYMA tool was then introduced to mechanical
engineering education [5], providing the steps for determining a system dynamics model, but it does not support online simulations that integrate both system models and control rules.

Numerous contributions in both algorithms and their computational efficiency have been made in the field of robot dynamics [6-9]. The need for an online computing [6] method is motivated by simulation studies of the dynamics of a robotic system with large DOFs. While computation efficiency continues to be crucial for the simulation and control of increasingly complex mechanisms operating at higher speeds, other aspects of the dynamics problem are also vital. Algorithms should be formulated under a simple framework to enable ease of development, modification, extension and understanding.

This paper aims to demonstrate a clear relationship between Lagrange equations and Newton-Euler equations regarding the computational methods for robot dynamics, from which we derive a systematic method using either symbolic or on-line numerical computations. First, this study reviews the Lagrange formulation, as well as its symbolic generation method, which is suitable for studying a low-DOF robotic system. Second, based on the decomposition approach and cross-product operation, a computing framework can be easily developed to calculate the inertia matrix, the Coriolis and centrifugal matrix, and the gravity force vector of robot dynamics equations. The advantages of this computing method are that: it can be used for both symbolic and the on-line numeric computation purposes, and it also can be applied to a closed-chain robot system. We present several robotic system examples to demonstrate the versatility of the proposed method.

2. Decomposition method

Assume that a robot system has a generalized coordinate with \( n \) DOFs and let \( q = [q_1 \ q_2 \ \cdots \ q_n]^T \). The equation of robot motion [9] can be written as

\[
D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = u ,
\]  

where the inertia matrix \( D \) is an \( n \times n \) positive-definite symmetric matrix, \( C \) is an \( n \times n \) Coriolis and centrifugal matrix, and \( G \) is an \( n \times 1 \) gravity force vector. The control input \( u = [u_1 \ u_2 \ \cdots \ u_n]^T \) is the vector of joint input forces or torques, and \( u_i \equiv 0 \) if the \( i \)-th joint is not actuated. Significantly, \( C \) is not unique, and \( S = D - 2C \) is a skew-symmetric matrix caused by the null energy property.

2.1 Lagrange formulation approach

The total kinetic energy \( K \) of a robot consists of the rotational kinetic energy \( K_{\text{rot}} \) and the translational kinetic energy \( K_{\text{tr}} \) of each link. The total potential energy \( P \) consists of the potential energy \( P_i \) of each link; and therefore,

\[
K = \sum_{i=1}^{n} K_{\text{rot}} + \sum_{i=1}^{n} K_{\text{tr}},
\]

and

\[
P = \sum_{i=1}^{n} P_i.
\]

The kinetic energies \( K_{\text{rot}} \) and \( K_{\text{tr}} \), and potential energy \( P_i \) of each link can be calculated as stated below. Let \( m_i \) be the mass of link \( i \) and \( I_i \) be the inertia of link \( i \) expressed in the body-attached frame. Let \( R_i \) be the rotational matrix, \( \omega_i \) be the angular velocity and \( p_{ci} \) be the CoM (Centre of Mass) of link \( i \). The angular velocity of link \( i \) can thus be expressed as

\[
\omega_i = J_{\text{rot}} \dot{q}_i ,
\]

where \( J_{\text{rot}} = \frac{\partial \omega_i}{\partial q} \) is the angular Jacobian matrix of link \( i \), and the rotational kinetic energy of link \( i \) can be obtained through

\[
K_{\text{rot}} = \frac{1}{2} \omega_i^T (R_i I_i R_i^T) \omega_i = \frac{1}{2} \dot{q}_i^T (J_{\text{rot}}^T R_i I_i R_i J_{\text{rot}}) \dot{q}_i .
\]

The potential energy of link \( i \) can be obtained by

\[
P_i = m_i g p_{ci} e_3 ,
\]

where \( g \) is the acceleration of gravity and \( e_3 = [0 \ 0 \ 1]^T \). The velocity of the CoM of link \( i \) can be computed by

\[
\dot{p}_{ci} = J_{\text{tr}} \dot{q}_i ,
\]

where \( J_{\text{tr}} = \frac{\partial p_{ci}}{\partial q} \) is the linear Jacobian matrix of link \( i \), and the translational energy of link \( i \) is via

\[
K_{\text{tr}} = \frac{1}{2} m_i \dot{p}_{ci}^T \dot{p}_{ci} = \frac{1}{2} \dot{q}_i^T (m_i J_{\text{tr}}^T J_{\text{tr}}) \dot{q}_i
\]
From the Lagrange formulation, we have

\[
D_{ai} = \frac{\partial}{\partial \dot{q}} \left( \frac{\partial K_{ai}}{\partial q} \right)
\]

\[
C_{ai} = \frac{\partial}{\partial \dot{q}} \left( \frac{\partial K_{ai}}{\partial q} \right) - \frac{1}{2} \frac{\partial}{\partial \dot{q}} \left( D_{ai} \dot{q} \right)
\]

and

\[
D_{vi} = \frac{\partial}{\partial \dot{q}} \left( \frac{\partial K_{vi}}{\partial q} \right)
\]

\[
C_{vi} = \frac{\partial}{\partial \dot{q}} \left( \frac{\partial K_{vi}}{\partial q} \right) - \frac{1}{2} \frac{\partial}{\partial \dot{q}} \left( D_{vi} \dot{q} \right)
\]

\[
G_i = \frac{\partial P_i}{\partial \dot{q}}.
\]

The \( D \), \( C \) and \( G \) matrices of the robot’s equations of motion are then

\[
D = \sum_{i=1}^{n} D_{ai} + \sum_{i=1}^{n} D_{vi} \tag{9}
\]

\[
C = \sum_{i=1}^{n} C_{ai} + \sum_{i=1}^{n} C_{vi} \tag{10}
\]

and

\[
G = \sum_{i=1}^{n} G_i. \tag{11}
\]

2.2 Newton-Euler formulation approach

The decomposition of the robot’s equations of motion can also be conducted using the Newton-Euler approach. The angular acceleration of link \( i \) is expressed as

\[
\ddot{\omega}_i = J_{ai} \ddot{q} + \dot{J}_{ai} \dot{q}, \tag{12}
\]

where \( \dot{J}_{ai} = \frac{\partial}{\partial \dot{q}} \left( J_{ai} \dot{q} \right) \) is the differentiation of the angular Jacobian matrix \( J_{ai} \). The Euler equation of link \( i \) is

\[
R_i I_i R_i^T \ddot{\omega}_i + \omega_i \times \left( R_i I_i R_i^T \right) \omega_i = \tau_i, \tag{13}
\]

and hence,

\[
R_i I_i R_i^T \left( J_{ai} \ddot{q} + \dot{J}_{ai} \dot{q} \right) + \omega_i \times \left( R_i I_i R_i^T \right) \omega_i = \tau_i.
\]

Since \( J_{ai}^T \tau_i = u \), we now have the equation

\[
J_{ai}^T R_i I_i R_i^T J_{ai} \ddot{q} + J_{ai}^T R_i I_i R_i^T \dot{J}_{ai} \dot{q} + \dot{J}_{ai}^T \omega_i \times \left( R_i I_i R_i^T \right) J_{ai} \dot{q} = \dot{J}_{ai}^T \tau_i = \dot{u},
\]

which is the same as

\[
D_{ai} \ddot{q} + C_{ai} \dot{q} = u.
\]

Therefore, \( D_{ai} \) and \( C_{ai} \) can be obtained respectively using:

\[
D_{ai} = J_{ai}^T \left( R_i I_i R_i^T \right) J_{ai} \tag{14}
\]

and

\[
C_{ai} = J_{ai}^T \left( R_i I_i R_i^T \right) \dot{J}_{ai} + J_{ai}^T \left[ \omega_i \times \left( R_i I_i R_i^T \right) \right] J_{ai}, \tag{15}
\]

where \( \omega_i \times \) is the skew-symmetric matrix representation of the cross-product operation.

The acceleration of CoM of link \( i \) is

\[
\ddot{p}_i = J_{vi} \ddot{q} + \dot{J}_{vi} \dot{q}, \tag{16}
\]

where \( \dot{J}_{vi} = \frac{\partial}{\partial \dot{q}} \left( J_{vi} \dot{q} \right) \) is the differentiation of the linear Jacobian matrix \( J_{vi} \). The Newton equation of the CoM of link \( i \) is

\[
m_i \ddot{p}_i = m_i g e_3 + f_i, \tag{17}
\]

and hence,

\[
m_i \left( J_{vi} \ddot{q} + \dot{J}_{vi} \dot{q} \right) + m_i g e_3 = f_i.
\]

Since \( J_{vi}^T f_i = u \), we formulate the equation

\[
m_i J_{vi}^T \ddot{q} + m_i J_{vi}^T \dot{J}_{vi} \dot{q} + m_i g J_{vi}^T e_3 = J_{vi}^T f_i = u,
\]

which is also the same as

\[
D_{vi} \ddot{q} + C_{vi} \dot{q} + G_e = u.
\]
Therefore, the $D_{vi}$, $C_{vi}$ and $G_i$ matrices can be obtained by

$$D_{vi} = m_i J^T_{vi} J_{vi}$$

(18)

and

$$C_{vi} = m_i J^T_{vi} J_{vi}$$

(19)

and

$$G_i = \frac{\partial P}{\partial q_i}.$$  

(20)

### 2.3 Extended coordinate system

The Cartesian coordinates of the base point, $[x_0 \ y_0 \ z_0]^T$, are frequently added to form an extended set of generalized coordinates for the robot systems. Let $q_i = [x_0 \ y_0 \ z_0 \ q_1 \ \cdots \ q_n]^T$ be the extended coordinate system and next define the total mass $M = \sum_{i=1}^n m_i$, Jacobian matrix $J_{\text{Com}} = \frac{1}{M} \sum_{i=1}^n m_i J_{vi}$, and its derivative $\dot{J}_{\text{Com}} = \frac{1}{M} \sum_{i=1}^n m_i \dot{J}_{vi}$. The robot’s equations of motion in this extended coordinate system can thus be expressed as

$$\dot{D}_i(q_i)\ddot{q}_i + \dot{C}_i(q_i, \dot{q}_i)\dot{q}_i + G_i(q_i) = B_i u,$$  

(21)

where

$$D_i = \begin{bmatrix} M I_{3x3} & M J_{\text{Com}} \\ M J^T_{\text{Com}} & D \end{bmatrix},$$

$$C_i = \begin{bmatrix} 0_{3x3} & MJ_{\text{Com}} \\ 0_{n\times3} & C \end{bmatrix},$$

$$G_i = \begin{bmatrix} M e_i \\ G \end{bmatrix},$$

and

$$B_i = \begin{bmatrix} 0_{3x3} \\ I_{n\times1} \end{bmatrix}.$$

### 3. Procedures for computing the dynamic equations of a serial-link robot

Let $m_i$ be the mass of link $i$, $I_i$ be the inertia of link $i$ expressed in the body-attached frame and then $q = [q_1 \ q_2 \ \cdots \ q_n]^T$ be the joint coordinate vector, where $n$ is the number of joints. For simplicity, it is assumed that the relative motion of link $i$ with respect to link (i-1) is either the rotational (revolute joint) or the translational (prismatic joint) motion with one of the basic axes in the local body frame from the set $\{ \pm x$-axis, $\pm y$-axis, and $\pm z$-axis $\}$ and is denoted by symbol $\rho_i$, where

$$\rho_i \in \{ +Rx, -Rx, +Ry, -Ry, +Rz, -Rz \}.$$

Define three basic rotational matrices: $R_i(q_i)$, $R_j(q_j)$, and $R_k(q_k)$, as shown below

$$R_i(q_i) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos q_i & -\sin q_i \\ 0 & \sin q_i & \cos q_i \end{bmatrix},$$

and

$$R_j(q_j) = \begin{bmatrix} \cos q_j & 0 & \sin q_j \\ 0 & 1 & 0 \\ -\sin q_j & 0 & \cos q_j \end{bmatrix},$$

and

$$R_k(q_k) = \begin{bmatrix} \cos q_k & -\sin q_k & 0 \\ \sin q_k & \cos q_k & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The procedures for computing the dynamic equation of a serial-link robot are thus as follows:

**Step 1:** Computing the rotational matrix of link $i$, $R_i$, $i=1,2,\cdots,n$

Let $R_0 = I_{3x3}$, and

$$R_i = \begin{cases} R_{i-1}R_i(\pm q_i), & \rho_i = \pm Rx \\ R_{i-1}R_i(\pm q_i), & \rho_i = \pm Ry \\ R_{i-1}R_i(\pm q_i), & \rho_i = \pm Rz \end{cases}, \quad i=1,2,\cdots,n.$$  

(22)

Let $a_i$ and $t_i$ be the vectors of the corresponding rotational and translational axes of link $i$ with respect to link (i-1), and then

$$a_i = \begin{cases} \pm R_{i-1}(\pm q_i), & \rho_i = \pm Rx \\ \pm R_{i-1}(\pm q_i), & \rho_i = \pm Ry \end{cases}, \quad i=1,2,\cdots,n,$$  

(23)

and

$$t_i = \begin{cases} \pm R_{i-1}(\pm q_i), & \rho_i = \pm Tx \\ \pm R_{i-1}(\pm q_i), & \rho_i = \pm Ty \end{cases}, \quad i=1,2,\cdots,n.$$  

(24)
where $R(:,k), k = 1, 2, 3$ represents the k-th column of the matrix $R$.

Step 2: Computing the angular Jacobian matrix of link $i$, $\omega_i = J_{ai} \dot{q}_i, i = 1, 2, \cdots, n$

Let $\omega_0 = 0_{3 \times 1}$, and

$$\omega_i = \omega_{i-1} + a_i \dot{q}_j, \ i = 1, 2, \cdots, n.$$ \hspace{1cm} (25)

The angular Jacobian matrix of link $i$, $J_{ai}$, is

$$J_{ai}(:,j) = \begin{cases} a_j & j \leq i \\ 0_{3 \times 1} & j > i \end{cases}, \ i, j = 1, 2, \cdots, n.$$ \hspace{1cm} (26)

where $J_{ai}(:,j)$ represents the j-th column of matrix $J_{ai}$ and the total rotational kinetic energy $K_\omega$ is obtained from

$$K_\omega = \sum_{i=1}^{n} \omega_i^T R_i I_i \omega_i = \frac{1}{2} \dot{\theta}^T \left( \sum_{i=1}^{n} J_{ai}^T R_i I_i J_{ai} \right) \dot{\theta}.$$ \hspace{1cm} (27)

Matrix $D_{ai}$ is formed by

$$D_{ai} = J_{ai}^T R_i I_i R_i^T J_{ai}, \ i = 1, 2, \cdots, n.$$ \hspace{1cm} (28)

Step 3: Differentiation of the angular Jacobian matrices

The angular acceleration of link $i$ is obtained as

$$\ddot{\omega}_i = J_{ai} \ddot{q}_i + J_{ai} \dot{q}_i, \ i = 1, 2, \cdots, n,$$

where $\ddot{J}_{ai} = \frac{\partial}{\partial \dot{q}} \left( J_{ai} \dot{q}_i \right) = \sum_{j=1}^{n} \frac{\partial J_{ai}}{\partial \dot{q}_j} \dot{q}_j$ is found to be

$$\ddot{J}_{ai}(:,j) = \begin{cases} \sum_{k=1}^{j-1} a_k \times a_j \dot{q}_k & j \leq i, \ i, j = 1, 2, \cdots, n \\ 0_{3 \times 1} & j > i \end{cases}.$$ \hspace{1cm} (29)

Matrix $C_{ai}$ is set up by

$$C_{ai} = J_{ai}^T R_i I_i R_i^T \ddot{J}_{ai} + J_{ai}^T \left[ \omega_i \times J_{ai} \right] R_i I_i R_i^T J_{ai},$$

$$i = 1, 2, \cdots, n.$$ \hspace{1cm} (30)

Step 4: Computing the linear Jacobian matrix of the CoM of link $i$, $\dot{p}_i = J_{vi} \dot{q}_i, \ i = 1, 2, \cdots, n$

First, compute the joint position and CoM position of link $i$, $p_i$ and $\frac{\partial p_i}{\partial \dot{q}}$, $i = 1, 2, \cdots, n$. The linear Jacobian matrix of the CoM of link $i$, $J_{vi} = \frac{\partial p_i}{\partial \dot{q}}$, is then found as

$$J_{vi}(:,j) = \begin{cases} a_j \times (p_{ci} - p_j) & \text{revolute joint} \\ \delta_{ij} & \text{prismatic joint} \end{cases}, \ i, j = 1, 2, \cdots, n.$$ \hspace{1cm} (31)

The total translational kinetic energy $K_v$ and the total potential energy $P$ are obtained as

$$K_v = \sum_{i=1}^{n} K_{vi} = \frac{1}{2} \sum_{i=1}^{n} m_i \dot{p}_i^T \dot{p}_i$$

and

$$P = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} m_i g p_i^T e_i.$$ \hspace{1cm} (32)

The matrices $C_{vi}$ and $G_i$ are set up as

$$C_{vi} = m_i J_{vi}^T J_{vi},$$

and

$$G_i = m_i g J_{vi}^T e_i, \ i = 1, 2, \cdots, n.$$ \hspace{1cm} (33)

Step 5: Differentiation of the linear Jacobian matrices

The linear acceleration of the CoM of link $i$ is obtained as

$$\ddot{p}_i = J_{vi} \ddot{q}_i + \frac{\partial}{\partial \dot{q}} \left( J_{vi} \dot{q}_i \right) \ddot{q}_i = J_{vi} \ddot{q}_i + \dot{J}_{vi} \dot{q}_i$$

where $\dot{J}_{vi} = \frac{\partial}{\partial \dot{q}} \left( J_{vi} \dot{q}_i \right) = \sum_{j=1}^{n} \frac{\partial J_{vi}}{\partial \dot{q}_j} \dot{q}_j$ is found to be

$$\dot{J}_{vi}(:,j) = \begin{cases} \sum_{k=1}^{j-1} a_k \times J_{vi}(:,j) \dot{q}_k & j \leq i \\ \sum_{k=1}^{j-1} a_k \times J_{vi}(:,j) \dot{q}_k & j > i \end{cases}, \ i, j = 1, 2, \cdots, n.$$ \hspace{1cm} (34)
The matrix $C_{vi}$ is equal to

$$C_{vi} = m_i J_v^T J_v, \quad i = 1, 2, \cdots, n.$$  

Step 6: Putting it all together

The $D$, $C$, and $G$ matrices of the robot's equations of motion are

$$D = \sum_{i=1}^{n} D_{ai} + \sum_{i=1}^{n} D_{ri} = \sum_{i=1}^{n} J_v^T R_i J_v + \sum_{i=1}^{n} m_i J_v^T J_v, \quad i = 1, 2, \cdots, n, \quad (33)$$

$$C = \sum_{i=1}^{n} C_{ai} + \sum_{i=1}^{n} C_{ri} = \sum_{i=1}^{n} \left( J_v^T R_i J_v \right) + \sum_{i=1}^{n} \left[ \omega_i \times \right] R_i J_v^T J_v, \quad (34)$$

and

$$G = \sum_{i=1}^{n} G_i = \sum_{i=1}^{n} m_i g J_v^T e_1. \quad (35)$$

4. Illustrated examples

4.1 Example 1: a 3 DOFs spherical robot

Figure 1 shows an example of a simple 3-DOF spherical robot system with both revolute and prismatic joints, and generalized coordinate $q = [q_1, q_2, r]^T$, and is presented to verify the proposed computing method. The system parameters of the spherical robot are shown in Table 1 and the control input vector $u = [u_1, u_2, u_3]^T$. For a link with two joints, a virtual link with zero mass and/or zero inertia can be inserted for easy programming. The proposed method and the Lagrange method are shown to obtain the same equations of motion as follows

$$D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = u,$$

where

$$D = \begin{bmatrix} I_z + c_2^2 (I_x - I_z + Mr^2) & 0 & 0 \\ 0 & I_y + Mr^2 & 0 \\ 0 & 0 & M \end{bmatrix},$$

$$C = \begin{bmatrix} c_2 \left( Mr c_2 \hat{r} - (Mr^2 + I_x - I_z) s_2 \hat{q}_2 \right) \\ (Mr^2 + I_x - I_z) c_2 s_2 \hat{q}_1 \\ -Mr c_2^2 \hat{q}_1 \end{bmatrix},$$

and

$$G = \begin{bmatrix} -M g r s_c c_2 \\ -M g r c_2 s_c \\ M g c_2 s_c \end{bmatrix},$$

where $c_i = \cos(q_i)$, $s_i = \sin(q_i)$, and $i = 1, 2$.

![Diagram of a 3-DOF spherical robot system](image1)

**Figure 1.** Example of a 3-DOF spherical robot system.

| Joint | Link mass $m_i$ | Link length $r_i$ | Link CoM $d_i$ | Link inertia $I_i$ |
|-------|----------------|------------------|---------------|------------------|
| 1     | $+Rx$ 0       | [0 0 0]^T        | [0 0 0]^T     | zero inertia     |
| 2     | $+Ry$ 0       | [0 0 0]^T        | [0 0 0]^T     | $diag\{I_x, I_y, I_z\}$ |
| 3     | $+Tz$ $M$     | [0 0 $r_z$]^T   | [0 0 $r_z$]^T| zero inertia     |

**Table 1.** System parameters of the 3-DOF spherical robot system.

4.2 Example 2: a 3-DOF robot system

Figure 2 presents an example of a 3-DOF robotic pendulum system with the generalized coordinate $q = [q_1, q_2, q_3]^T$. Table 2 lists the system parameters of this example and the control input vector $u = [u_1, u_2, u_3]^T$. The proposed method and the traditional Lagrange method are shown to obtain the same equations of motion,

$$D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = u.$$  

![Diagram of a 3-DOF robotic pendulum system](image2)

**Figure 2.** Example of a 3-DOF robotic pendulum system.
Table 2. System parameters of a 3-DOF robotic pendulum system.

| $i$ | Joint $\rho_i$ | Link mass $m_i$ | Link length $r_i$ | Link CoM $d_i$ | Link inertia $I_i$ |
|-----|----------------|----------------|----------------|----------------|-----------------|
| 0   | base           | $M_0$          | $0\ 0\ 0$     | $0\ 0\ 0$     | zero inertia    |
| 1   | +Rx            | $M_1$          | $0\ L_1$      | $0\ L_1/2$    | zero inertia    |
| 2   | +Ry            | $M_2$          | $0\ 0\ 0$     | $0\ L_2$      | zero inertia    |
| 3   | +Ry            | $M_3$          | $0\ 0\ 0$     | $0\ 0\ 0$     | zero inertia    |

Table 3. System parameters of a biped robot.

| $i$ | Joint $\rho_i$ | Link mass $m_i$ | Link length $r_i$ | Link CoM $d_i$ | Link inertia $I_i$ |
|-----|----------------|----------------|----------------|----------------|-----------------|
| 0   | base           | $M_0$          | $0\ 0\ 0$     | $0\ 0\ 0$     | zero inertia    |
| 1   | +Rx            | $M_1$          | $0\ L_1$      | $0\ L_1/2$    | zero inertia    |
| 2   | +Ry            | $M_2$          | $0\ 0\ 0$     | $0\ L_2$      | zero inertia    |
| 3   | +Ry            | $M_3$          | $0\ 0\ 0$     | $0\ 0\ 0$     | zero inertia    |
| 4   | +Rx            | $M_4$          | $0\ 0\ 0$     | $0\ 0\ 0$     | zero inertia    |
| 5   | +Ry            | $M_5$          | $0\ 0\ 0$     | $0\ 0\ 0$     | zero inertia    |

4.3 Example 3: a 10-DOF biped robot

This example considers a 10-DOF fully actuated 3D biped with feet, as shown in Figure 3. The 3D biped robot consists of 7 links: a torso and two legs and two feet. Each leg has a knee and an equal length for the shin and thigh. The 10 actuators are installed as follows: two at the hip, one at the knee and two at the ankle. Table 3 lists the system parameters of this example.

Let $n=10$, the generalized coordinate $q=[q_1, q_2, \ldots, q_{10}]^T$, and the control input vector $u=[u_1, \ldots, u_{10}]^T$. In the single-support phase, the biped's dynamic equation can be obtained as

$$D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = u,$$

where

$$D = \sum_{i=1}^{n} m_i J_{wi}^T J_{vi},$$

$$C = \sum_{i=1}^{n} m_i J_{wi}^T \dot{J}_{vi},$$

and

$$G = \sum_{i=1}^{n} m_i g J_{wi}^T e_3.$$

Aside from the equations of motion, significant amounts of important data for the analysis of the biped's walking ability are also computed with minimal additional effort. For example, the biped ground reaction force $F = [F_x, F_y, F_z]^T$ and the zero moment point (ZMP) during the single-support phase $[x_{ZMP}, y_{ZMP}]^T$ can be set up as:

$$F_x = \sum_{i=0}^{n} m_i \ddot{x}_i,$$

$$F_y = \sum_{i=0}^{n} m_i \ddot{y}_i,$$

$$F_z = \sum_{i=0}^{n} m_i (g + \ddot{z}_i),$$

and

$$x_{ZMP} = \frac{1}{\sum_{i=0}^{n} m_i (g + \ddot{z}_i)} \left[ \sum_{i=0}^{n} m_i x_i (g + \ddot{z}_i) - \sum_{i=0}^{n} m_i \dot{z}_i \ddot{x}_i - \sum_{i=0}^{n} \tau_i \right],$$

$$y_{ZMP} = \frac{1}{\sum_{i=0}^{n} m_i (g + \ddot{z}_i)} \left[ \sum_{i=0}^{n} m_i y_i (g + \ddot{z}_i) - \sum_{i=0}^{n} m_i \dot{z}_i \ddot{y}_i + \sum_{i=0}^{n} \tau_i \right].$$
where \(g\) is the gravity acceleration, 
\[ p_i = [x_i, y_i, z_i]^T \]
is the centre of mass of the \(i\)-th link, and 
\[ \tau_i = [(\tau_{i})_x, (\tau_{i})_y, (\tau_{i})_z]^T \]
is the moment of the \(i\)-th link. For authentication, the 10-DOF biped robot online simulation work has been implemented on MATLAB. After symbolic calculation, we transfer the dynamic equations into C-Mex files to improve the efficiency of the biped robot walking simulation.

4.4 Example 4: a simple 2-DOF closed-chain robot

We can also apply the proposed method to calculate the equations of motion for simple closed-chain robotic systems. Figure 4 shows an example of a 2-DOF four-bar linkage robot system, where Link 1 and Link 4 are parallel links and their absolute angle \(\theta_1\) is controlled by the input torque \(u_1\). Similarly, Link 2 and Link 3 are parallel links and their absolute angle \(\theta_2\) is controlled by the input torque \(u_2\). The four-bar robot system can be treated as 2 serial-chain robot systems. The first serial-chain robot consists of Link 1 and Link 2 with generalized coordinates

\[
q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \theta_2 - \theta_1 \end{bmatrix} = A_1 \begin{bmatrix} \theta_1^{\text{L}} \\ \theta_2^{\text{L}} \end{bmatrix},
\]

where
\[
A_1 = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}.
\]

Similarly, the second serial-chain robot consists of Link 4 and Link 3 with generalized coordinates

\[
q = \begin{bmatrix} q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} \theta_3 \\ \theta_4 - \theta_3 \end{bmatrix} = A_2 \begin{bmatrix} \theta_3^{\text{L}} \\ \theta_4^{\text{L}} \end{bmatrix},
\]

where
\[
A_2 = \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix}.
\]

For the first robot system, the system matrices \(D_1, C_1\) and \(G_1\) can be calculated using the above method. For the second robot system, the matrices \(D_2, C_2\) and \(G_2\) can be similarly obtained. The equations of motion for the four-bar linkage can thus be determined as:

\[
\left( A_1^T D_1 A_1 + A_2^T D_2 A_2 \right) \ddot{\dot{\theta}} + \left( A_1^T C_1 A_1 + A_2^T C_2 A_2 \right) \dot{\dot{\theta}} + \left( A_1^T G_1 + A_2^T G_2 \right) = u
\]

and hence

\[
D = A_1^T D_1 A_1 + A_2^T D_2 A_2 =
\begin{bmatrix}
\frac{1}{2}(m_1 + 2m_2)L_1^2 & \frac{1}{2}(m_1 + m_2)L_1L_2 \cos(\theta_1 - \theta_2) \\
\frac{1}{2}(m_1 + m_2)L_1L_2 \cos(\theta_1 - \theta_2) & \frac{1}{2}(m_2 + 2m_1)L_2^2
\end{bmatrix}
\]

\[
C = A_1^T C_1 A_1 + A_2^T C_2 A_2 =
\begin{bmatrix}
0 & \frac{1}{2}(m_1 + m_2)L_1L_2 \sin(\theta_1 - \theta_2) \dot{\theta}_2 \\
\frac{1}{2}(m_1 + m_2)L_1L_2 \sin(\theta_1 - \theta_2) \dot{\theta}_1 & 0
\end{bmatrix}
\]

and

\[
G = A_1^T G_1 + A_2^T G_2 =
\begin{bmatrix}
g(m_1 + m_2)L_1 \cos \theta_1 \\ g(m_1 + m_2)L_2 \cos \theta_2
\end{bmatrix}
\]

Figure 3. Biped coordinate system when the left foot is the stance foot.

Figure 4. An example of a 2-DOF four-bar linkage robot system.
5. Conclusion

This paper has developed a decomposition and cross-product-based method to compute robot dynamics, offering several examples that demonstrate the versatility of the proposed method. The focus of this paper is first to establish a clear relationship between Lagrange equations and Newton-Euler equations, and second to develop an extremely intuitive and systematic method to compute the inertia matrix, the Coriolis and centrifugal matrix, and the gravity force vector of robot dynamics equations.

6. Acknowledgements

The work of C.L. Shih is supported by the Taiwan National Science Council (NSC) under Grant NSC 100-2221-E-011-020.

7. References

[1] Hurmuzlua Y, Genotb F, Brogliatoc B (2004) Modeling, Stability and Control of Biped Robots—a General Framework. Automatica 40: 1647-1664.
[2] Tošic D.V, Lutovac M.D (2008) Symbolic Simulation of Engineering Systems. 4th European Conference on Circuits and Systems for Communications. pp. 59-68.
[3] Cetinkunt S, Ittoop B (1992) Computer-automated Symbolic Modeling of Dynamics of Robotic Manipulators with Flexible Links. IEEE Trans. on Robotics and Automation. 8: 94-105.
[4] Vibet C (1995) Symbolic Modeling of Robot Kinematics and Dynamics. Robotics and Autonomous Systems. 14: 301-314.
[5] Vira N, Tunstel E (1992) Use of Symbolic Computation in Robotics Education. IEEE Trans. on Education. 35: 18-30.
[6] Luh J, Walker M, Paul R (1980) On-line Computational Scheme for Mechanical Manipulators. ASME Journal of Dynamic Systems, Measurement, and Control. 102: 69-76.
[7] Hollerbach J.M (1980) A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complex. IEEE Trans. on Systems, Man, and Cybernetics. 11: 730-736.
[8] Featherstone R, Orin D (2000) Robot Dynamics: Equations and Algorithms. IEEE Inter. Conf. on Robotics and Automation. pp. 826-834.
[9] Spong M, Hutchinson S, Vidyasagar M (2006) Robot Modeling and Control. John Wiley & Sons, Inc.