Computer Algebra for Real-Time Dynamics of Robots with Large Numbers of Joints

Ramutis Bansevicius[1], Algimantas Cepulkauskas[2], Regina Kulvietiene[2], and Genadijus Kulvietis[2]

[1] Kaunas University of Technology, Donelaicio 73, Kaunas 3006, Lithuania
bansevicius@cr.ktu.lt

[2] Vilnius Gediminas Technical University, Sauletekio 11, Vilnius 2040, Lithuania
{algimantas.cepulkauskas,regina.kulvietiene,
genadijus.kulvietis}@gama.vtu.lt

Abstract. This paper deals with the description of a theoretical background of systematic computer algebra methods for analyzing the real-time dynamics of robots with a large numbers of joints. Many numerical methods based on different principles of mechanics were developed to obtain the equations that model the dynamic behavior of robots. In this paper, the efficiency of computer algebra application was compared with the most popular methods of forming the dynamic equations of robots in real time. To this end, the computer algebra system VIBRAN was used. A real-time dynamic model in closed form of the robots with large numbers of joints has been developed, using the computer algebra technique with the following automatic program code generation.

1 Introduction

The application of general control theory to complex mechanical systems, such as robots, aircrafts, etc., represents an extremely difficult problem because of prominent nonlinearity and complexity of mathematical models of these systems. If industrial robots have large numbers of joints, the application of such a theory and development of new control algorithms are unavoidable in order to achieve a high positioning speed and accuracy. In on-line control, the calculation of model equations must be repeated very often, preferably at the sampling frequency that is no lower than 50Hz. However, the problem of forming the dynamic equations of robots in real time by means of today's computers is rather difficult and complex. It appears necessary to develop computer methods of mathematical modelling for at least two reasons. One of them is that it is impossible to immediately choose the most convenient configuration when designing robots. The term configuration should be interpreted as the structure (i.e., kinematic scheme) and parameters (i.e., dimensions, masses, etc.). Thus, it is necessary to analyze a number of different robot configurations and choose the one, most appropriate to the future purpose of the device. Knowing how complex a task it is to write a mathematical model by hand, the need for an algorithm that would enable a computer to perform the task seems quite logical. The other reason is the need in multiple applications for real-time control of robots. The development of computer methods, such that perform real-time calculations of robot dynamics, is a direct contribution to the synthesis of control algorithms for practical purposes. Particularly this problem is much more complex for the robots with a large number of joints [7], [12], [13].

In the last three decades, numerous investigators have used different principles of dynamics in order to obtain the equations that model the dynamic behavior of robot arms. The first formulations to be developed were based on a closed form representation of the equations, and the Lagrange-Euler (L-E) equations were preferentially used for this purpose. These formulations were found to be inefficient due to the high number of algebraic operations involved. A solution to this problem was found with the use of relationships present in the dynamic equations. The Newton-Euler (N-E) equations were found to be the most appropriate dynamic principle for this type of formulation and they have been used to develop the most efficient formulations known so far. Other formulations, based on the Kane equations, have yielded algorithms whose computational complexity is similar to that found in formulations based on the N-E equations. The use of dynamic principles different from those employed in the formulations based on L-E, N-E or Kane equations was minor and, furthermore, has produced formulations of high computational complexity. Currently it is believed that the use of diverse dynamic principles will lead to similar formulations of equivalent computational complexity. This has been partially proved by applying the appropriate relationships to the L-E equations in order to obtain an equivalent formulation to that given by the N-E equations, although a greater effort is required in order to reach the final equations [14]. It is for this reason that most of the formulations that produce efficient algorithms have been developed from the N-E equations. Featherstone and Orin [6] make a detailed review of these methods and algorithms derived.

The Gibbs-Appell (G-A) equations are one of the principles that has been used the least for solving the dynamic problem of manipulating robots. The simple form of these equations deal with mechanical systems subjected to holonomic and non-holonomic type of constraints is also emphasized in the specialized technical literature. Surprisingly, a bibliographical review of the literature on this area reveals a limited use of the G-A equations in modern dynamics. A few years ago, the supposed relationship of the G-A equations and Kane's dynamic equations caused a great number of works and comments on the matter [14]. In the field of robotics, Popov proposed a method, later developed by Vukobratovic [14], in which the G-A equations were used to develop a closed form representation of high computational complexity. This method was used by Desoyer and Lugner [11], [14] to solve, by means of the recursive formulation $O(n^2)$ (n is the number of the degree-of-freedom), an inverse dynamic problem, using the Jacobian matrix of the manipulator, with the view of avoiding the explicit development of partial derivatives. Another approach was suggested by Vereshchagin [14] who proposed manipulator motion equations from Gauss' principle and Gibbs' function. This approach was used by Rudas and Toth [11] to solve the inverse dynamic problem of robots. Recently, Mata et al. [10] have presented a formulation of order *O(n)* that solves the inverse dynamic problem and establishes recursive relations that involve a reduced number of algebraic operations. The algorithms that model the dynamic behavior of manipulators are divided into two types: algorithms that solve the inverse dynamic problem and those that give a solution to the forward dynamic problem. In the former, the forces exerted by the actuators are obtained algebraically for certain configurations of the manipulator (position, velocity and acceleration). On the other hand, the forward dynamic problem computes the acceleration of joints of the manipulator once the forces, exerted by the actuators, are put. This problem is part of the process that must be followed in order to simulate the dynamic behavior of the manipulator. This process is completed after it has calculated the velocity and position of the joints by means of the process of numerical integration in which the acceleration of the joints and the initial configuration are data input to the problem.

The first efficient recursive algorithm for solving the inverse dynamic problem was proposed by Luh et al. [9]. This algorithm, based on the N-E equations, has been improved repeatedly in the course of years [2], [6]. Other authors have developed efficient recursive algorithms to solve the inverse dynamic problem, based on other principles of dynamics. As examples of these, we have the work of Hollerbach [14] that uses the L-E equations; and those of Kane and Levinson [14], and Angeles et al. [1], which use the Kane equations. The complexity of the above mentioned numerical algorithms will be compared with computer algebra realization. Some efforts to apply symbolic calculations in the dynamics of robots were made [11], [14], but due to tremendous final closed form equations these efforts were unsuccessful.

Simulations by means of numerical methods are powerful tools for investigations in mechanics but they do have drawbacks, e.g., finite precision, errors generated when evaluating expressions. The computerized symbolic manipulation is a very attractive means to reliably perform analytic calculations even with complex formulas and expressions. But frequently a semi-analytic approach, combining the features of analytical and numerical computations, is the most desirable synthesis. This allows the analytic work to be pushed further before numerical computations start.

For numerical-symbolic computation of the real-time dynamics of robots with large numbers of joints the computer algebra system VIBRAN [5], [8] was used [11]. The computer algebra system VIBRAN is a FORTRAN preprocessor for analytical computation with polynomials, rational functions and trigonometric series. Special VIBRAN's procedure can generate an optimized FORTRAN code from the obtained analytical expressions, which can be directly used in the programs for a further numerical analysis.

2 Real-Time Dynamics of Robot

The real-time dynamic model of a robot was constructed using the Uicker-Kahn method [11], [14], based on the L-E equations, that is very convenient for computer algebra implementation [3], [11]. This method enables the calculation of all the matrices of the dynamic robot model: the inertial matrix, the matrix of Coriolis and centrifugal effects and the gravity vector. The dynamic equations of an n-degree-of-freedom manipulator, derived using this method, are of the following form:

$$P_i = \sum_{j=i}^{n} \left\{ \sum_{k=1}^{j} \left[tr\left(\frac{\partial W_j}{\partial q_i} J_j \frac{\partial W_j^T}{\partial q_k} \right) \right] \ddot{q}_k + \right. \\ \left. + \sum_{k=1}^{j} \sum_{l=1}^{j} \left[tr\left(\frac{\partial W_j}{\partial q_i} J_j \frac{\partial^2 W_j^T}{\partial q_k \partial q_l} \right) \dot{q}_k \dot{q}_l \right] - m_j \overrightarrow{g}^T \frac{\partial W_j}{\partial q_i} \tilde{r}_{j0} \right\}, \quad (1)$$

where P_i is a driving torque acting at the i-th joint; q_i is a generalized joint coordinate corresponding to the i-th degree of freedom; W_i is the transformation matrix between the i-th local coordinate system and the reference system; J_i is the inertia matrix of the i-th link with the respect to local coordinate system; m_i is the mass of the link i; $\tilde{r}_{i0}$ is the distance vector between the center of mass of the link i and the origin of the reference coordinate system, expressed in the local coordinate system of the i-th link; $\overrightarrow{g}$ is the gravity vector.

The matrix W_i may be expressed as

$$W_i = A_0^1 A_1^2 ... A_{i-1}^i,$$

where A_{k-1}^k is a (4×4) transformation matrix between two local coordinate systems.

Equation (1) may be expressed in the matrix form

$$P = H(q)\ddot{q} + \dot{q}^T C(q)\dot{q} + g(q), \quad (2)$$

where P is the vector of driving torques; $H(q)$ is the inertial matrix of the system; $C(q)$ is the $n \times n \times n$ matrix of Coriolis and centrifugal effects; $g(q)$ is the vector of gravity effects.

Fig. 1 illustrates a flexible robot with a large number of joints [3], [4]. The robot consists of cylindrical piezoceramic transducers and spheres. Here the resonant oscillations of every piezoelectric transducer are controlled by a microprocessor, switching on and off the high-frequency and high-voltage signal from the signal generator. The phase and duration of every pulse, applied to the electrodes of transducers, are synchronized with the rotation of an unbalanced rotor, mounted in the gripper of the robot.

The external torque vector, placed in the gripper and rotating in the plane perpendicular to the gripper direction, is expressed in the form

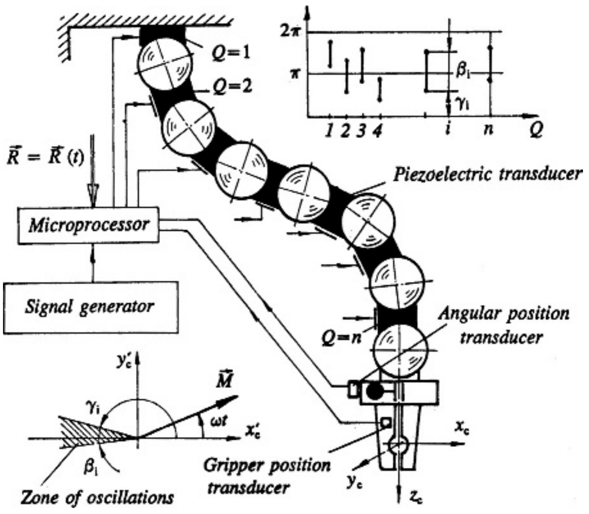

Fig. 1. The scheme of a robot with a large number of joints

$$\vec{F} = \left\{ \begin{array}{c} F_x = m_0 r \omega^2 \cos(\omega t) \\ F_y = m_0 r \omega^2 \sin(\omega t) \\ 0 \end{array} \right\}, \tag{3}$$

where m_0 is the mass of unbalance; r is a radius; ω is the angular velocity.

The recursive algorithm consists of two steps for each local coordinate. Therefore, the first step is the calculation of active forces and the second one is the definition of active torques. This algorithm may be expressed in the form

$$\begin{aligned} &\vec{F}_i = \tilde{A}_i^{i+1} \vec{F}_{i+1} \\ &\vec{M}_i = \tilde{A}_i^{i+1} \vec{M}_{i+1} + \vec{h}_{i,i-1} \times \vec{F}_{i+1}, \end{aligned} \tag{4}$$

where $\vec{F}_n = \vec{F}$, see formula (3). Expressions (4) are calculated starting from

$$i = n - 1 \quad to \quad i = 1.$$

The generalized torque for the i-th joint may be obtained in the form

$$Q_i = \vec{M}_i \vec{z}_{i0}, \tag{5}$$

where $\vec{z}_{i0}$ is the unit vector of the respective axis.

3 Computer Algebra Implementation

In the algorithm for automatic generation of the analytical model, it will be assumed that the parameters of a robot (length, mass, inertia, etc.) are known and will be treated as constants. Joint coordinates as well as their derivatives will be treated as independent variables, i.e., as symbols. Using the computer algebra technique, the Uicker–Kahn method is very convenient, because it enables us to obtain the equations of motion in closed form and may be applied in solving either the direct or the inverse problem of dynamics.

Fig. 2 illustrates a fragment of the VIBRAN program that implements the Uicker–Kahn method. In this program the sparse matrix technology was used to achieve the best performance. To have a possibility to compare various results and algorithms, only two joints of the proposed robot will be considered.

```
POLINOM A(16),B(20),C(20)
RACIONAL D,E,U
INTEGER*2 NA(18),NB(22),NC(22)
DATA G/0.,0.,-9.80621,0./
.....
100 RSMP(U,E,D,N)
ADDA(U,D)
100 RSMP(U,E,D,N)
```

Fig. 2. A fragment of the VIBRAN program

This program calculates all the elements of matrices *H(q), C(q), g(q)*. These matrices were calculated for the discussed flexible robot with the 6-th-degree-of-freedom. The kinematic parameters of this robot in Denavit–Hartenberg's notation [3], [11], [14] are presented in the table below.

N	q_i	α_i	a_i	d_i
1	q_1	0	0	0
2	q_2	$90°$	0	0
3	q_3	0	0.04	0
4	q_4	$-90°$	0	0
5	q_5	$-90°$	0	0
6	q_6	0	0	0.04

For simplicity, a substitution was made to avoid numerical trigonometric calculation of the function

$$S_i = \sin q_i, \quad C_i = \cos q_i.$$

The fragment of analytical calculations of flexible robot matrices performed by the VIBRAN program is presented in Fig. 3.

In total 153 elements were calculated and about 15% of them were equal to zero.

A special VIBRAN procedure [5] , [8] generates two FORTRAN subroutines from the obtained analytical expressions of robot matrices. The code of the first generated subroutine contains a dictionary of monomials included into the expressions of robot's matrices. This dictionary of monomials is sorted in ascending order of monomial multiindices to reduce the number of floating point multiplications. The code of the second generated subroutine contains the calculation of common members included in all the expressions and all the elements of robot's matrices. The generated subroutines can be immediately compiled and used for real-time operation, simulation or control synthesis.

```
H11  = .8326E-4+.1296E-3*C3**2-.9964E-4*C3**2*C4**2*C5**+.9964E
   -4*C3*S3*S4*C4*C5**2-
.................
G3 = -.113752E-6*S5*C4*C3+.113752E-6*S5*S4*S3+.14121E-5*C4*S3*S6
...........
G6 = .14121E-5*S3*S4*C6-.14121E-5*C4*C3*C6-.14121E-5*S3*C4*C5*S6
   -.14121E-5*C3*S4*C5*S6
```

Fig. 3. Analytical expressions of robot's matrices

The number of floating point product operations required to construct the dynamic model by the Uicker–Kahn method numerically depends on n^4 (n is the number of the degree-of-freedom) and, by contrast, the recursive methods based on the N-E or G-A equations have a linear dependency on the number of the degree-of-freedom. Some differences appear using the computer algebra technique. The Uicker–Kahn method produces closed-form differential equations and only recursive equations can be obtained from other well-known algorithms which means that only the numerical implementation is possible and this method suits only for inverse dynamics. The code presented in Fig. **??** contains only 371 floating point product. The computational complexity of the proposed approach is comparable with that of the most efficient algorithms known so far, as shown in the table below.

Authors	Principle	Products ($n+6$)	Number of operations
Luh et al. [9]	N-E	$150n-48$	852
Angeles et al. [1]	Kane	$105n-109$	521
Balafoutis and Patel [2]	N-E	$93n-69$	489
Mata et al. [10]	G-A	$96n-101$	475
This work	L-E	Closed form	371

Generalized torques were calculated in the same manner . These torques are needed to complete the control scheme of the robot. Another VIBRAN program calculates the acting forces and torques, using formula (4), and generalized torques, using formula (5).

4 Conclusions

The proposed mixed numerical-analytical implementation of the Uicker–Kahn method drastically reduces the number of floating point operations, particularly for robots with a large number of joints. The use of the computer algebra technique enables us to obtain the equations of motion in closed form. It can be applied in solving both the direct and the inverse problem of dynamics as well as in real-time dynamics modelling for intelligent control scheme realization.

References

1. Angeles, J., Ma, O., Rojas, A.: An algorithm for the inverse dynamics of n-axis general manipulators using Kane's equations. Comp. Math. Appl. 17 (12) (1989) 1545–1561.
2. Balafoutis, C.A., Patel, R.V.: Dynamic Analysis of Robot Manipulators: A Cartesian Tensor Approach, Kluwer Academic Press, Boston (1991).
3. Barauskas, R.; Bansevicius; R. Kulvietis; G. Ragulskis K.. 1988. Vibromotors for Precision Microrobots. Hemisphere Publishing Corp., USA.
4. Bansevicius R., Parkin R., Jebb, A., Knight, J.: Piezomechanics as a Sub-System of Mechatronics: Present State of the Art, Problems, Future Developments. IEEE Transactions on Industrial Electronics, vol. 43, (1) (1996) 23–30.
5. Cepulkauskas, A., Kulvietiene, R., Kulvietis G.: Computer Algebra for Analyzing the Vibrations of Nonlinear Structures. Lecture Notes in Computer Science, Vol. 2657. Springer-Verlag, Berlin Heidelberg New York (2003) 747–753.
6. Featherstone, R., .Orin, D. E.: Robot dynamics: equations and algorithms. Proceedings of the 2000 IEEE International Conference on Robotics and Automation, San Francisco (2000) 826–834.
7. Knani J.: Dynamic modelling of flexible robotic mechanisms and adaptive robust control of trajectory computer simulation. Applied Mathematical Modelling , Vol. 26. (12) (2002) 1113–1124.
8. Kulvietiene, R., Kulvietis, G.: Analytical Computation Using Microcomputers. LUSTI, Vilnius (1989).
9. Luh, J.Y.S., Walker, M.W., Paul, R.P.: On-line computational scheme for mechanical manipulators. J. Dyn. Syst. Meas. Control 102 (1980).
10. Mata, V., Provenzano, S., Valero, F., Cuadrado, J., I.: Serial-robot dynamics algorithms for moderately large numbers of joints. Mechanism and Machine Theory, 37 (2002) 739–755.
11. Rovetta, A., Kulvietis, G.: Lo sviluppo di software per il controllo dinamico di robot industriali. Dipartimento di Meccanica, Politecnico di Milano, Milano (1986).
12. Surdhar, J., S., White, A., S.: A parallel fuzzy-controlled flexible manipulator using optical tip feedback. Robotics and Computer-Integrated Manufacturing, Vol. 19 (3) (2003) 273–282.
13. Tso, S., K., Yang, T., W., Xu, W., L., Sun, Z., Q.: Vibration control for a flexible-link robot arm with deflection feedback. International Journal of Nonlinear Mechanics, 38 (2003) 51–62.
14. Vukobratovic, K., M., Kircanski M., N.: Real-time Dynamics of Manipulation Robots, Springer-Verlag, Berlin Heidelberg New York (1985).