Symbolic Calculation of the Generalized Inertia Matrix of Robots with a Large Number of Joints

Ramutis Bansevicius¹, Algimantas Čepulkauskas², Regina Kulvietienė²,
and Genadijus Kulvietis²

¹ Kaunas University of Technology,
Donelaičio 73, Kaunas 3006, Lithuania
`bansevicius@cr.ktu.lt`

² Vilnius Gediminas Technical University,
Saulėtekio 11, Vilnius 2040, Lithuania
{`algimantas-cepulkauskas`, `regina-kulvietiene`,
`genadijus-kulvietis`}@gama.vtu.lt

Abstract. The aim of this paper is to simplify numerical simulation of robots with a large number of joints. Many numerical methods based on different principles of mechanics are 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 generalized inertia matrix of robots. To this end, the computer algebra system was used. Expressions for the generalized inertia matrix of the robots with a large number of joints have been derived, using the computer algebra technique with the following automatic program code generation. As shown in the paper, such an application could drastically reduce the number of floating point product operations, needed for efficient numerical simulation of robots.

1 Introduction

Manipulator and robot systems possess several specific qualities in both a mechanical and a control sense. In the mechanical sense, a feature specific to manipulation robots is that all the degrees of freedom are “active”, i.e., powered by their own actuators, in contrast to conventional mechanisms in which motion is produced primarily by the so-called kinematics degrees of freedom. Another specific quality of such a mechanism is their variable structure, ranging from open to closed configurations, from one to another kind of boundary conditions. A further feature typical of spatial mechanisms is redundancy reflected in the excess of the degrees of freedom for producing certain functional movements of robots and manipulators. From the control standpoint, robot and manipulator systems represent redundant, multivariable, essentially nonlinear automatic control systems [13]. A manipulation robot is also an example of a dynamically coupled system, and the control task itself is a dynamic task [14].

The methods that model the dynamic behavior of manipulators are divided into two types: methods that solve the inverse dynamic problem and those that

give the solution to the direct 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 direct dynamic problem computes the acceleration of joints of the manipulator once the forces exerted by the actuators are given. This problem is part of the process that must be followed to perform the simulation of the dynamic behavior of the manipulator. This process is completed once are 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. So, the methods may be divided with respect to the laws of mechanics on the basis of which motion equations are formed. Taking this as a criterion, one may distinguish methods based on Lagrange–Euler’s (L-E), Newton–Euler’s (N-E), Gibbs–Appell’s (G-A) and other equations. The property of whether the method permits the solution of the direct or inverse problem of dynamics may represent another criterion. The direct problem of dynamics refers to determining the motion of the robot for known driving forces (torques), and the inverse problem of dynamics to determining driving forces for the known motion. Clearly, the methods allowing both problems of dynamics to be solved are of particular importance. The number of floating-point multiplications (divisions) / additions (subtractions) required to form a model is the most important criterion to compare the methods. This criterion is also important from the point of view of their on-line applicability.

The algorithms developed to solve the direct dynamic problem use, regardless of the dynamics principle from which they are derived, one of the following approaches [2], [9]:

- calculation of the acceleration of the joints by means of the method proposed and solution of a system of simultaneous equations;
- recursive calculation of the acceleration of the joints, propagating motion and constraint forces throughout the mechanism.

The algorithms derived from the methods that use the first approach require the calculation of the generalized inertia matrix and the bias vector [2]. The generalized inertia matrix is also used in advanced control schemes, as well as in parameter estimation procedures. For this reason its calculation, by means of simple and efficient procedures, is also beneficial to other fields, not only to motion simulation of mechanical systems. The generalized inertia matrix can be obtained through the Hessian of kinetic energy of the mechanical system with respect to generalized velocities; however, the most computationally efficient algorithms are not based on this procedure. The best known method that follows this first approach was proposed by Walker and Orin [15] who have developed (using N-E equations) the method of a composed rigid body, in which the generalized inertia matrix is obtained recursively with a complexity $O(n^2)$. Angeles and Ma [1] have proposed another method that follows this approach, based on the calculation of the natural orthogonal complement of the manipulator kine-

matics constraint equations with a complexity $O(n^3)$, using Kane's equations to obtain the bias vector.

On the other hand, algorithms derived from the methods that use the second approach usually have a complexity $O(n)$. These algorithms do not obtain the generalized inertia matrix, and for this reasons their application is limited to system motion simulations. The best known method among those that use the second approach is the articulated body method developed by Featherstone [6]. The number of required algebraic operations is lower to those needed in the composed rigid body method, but only for the systems that contain nine or more bodies. In [12], Saha has symbolically performed the Gaussian elimination to obtain a decomposition of the generalized inertia matrix. As an application of this decomposition, he proposed an $O(n)$ direct dynamic algorithm with a computational complexity very similar to that of [6].

The complexity of the numerical algorithms mentioned above for forming the generalized inertia matrix will be compared with computer algebra realization. The computer algebra technique application in the formation of the generalized inertia matrix of robots is very attractive, because it allows the analytic work to be pushed before the numerical integration of the system of nonlinear differential equations starts. This approach has been successfully applied to the inverse dynamic problem of the robot [3].

2 Algorithm for Calculating the Generalized Inertia Matrix

The algorithm for calculating the generalized inertia matrix has been constructed using the Uicker–Kahn method [12], based on the L-E equations, that is very convenient for computer algebra implementation [5]. The same approach was used to solve the inverse dynamic problem [3], but formation of the generalized inertia matrix must be considered more carefully, because the matrix must be recalculated at every step of numerical integration time of the robot dynamic model [14], [15]. The equations of the direct dynamic problem formulated by Vucobratovic [14] are contained in the following matrix expression:

$$H(\vec{q})\ddot{\vec{q}} = \vec{P} - \vec{C}(\vec{q}, \dot{\vec{q}}) + \vec{G}(\vec{q}), \quad (1)$$

where $H(\vec{q})$ is the generalized inertia matrix; $\vec{q}$, $\dot{\vec{q}}$, $\ddot{\vec{q}}$ are generalized coordinates, velocity, and acceleration of the robot, respectively; $\vec{P}$ is the generalized force vector; $\vec{C}(\vec{q}, \dot{\vec{q}})$ is the vector of Coriolis and centrifugal effects; $\vec{G}(\vec{q})$ is the vector of gravity effect. The bias vector ($\vec{C}(\vec{q}, \dot{\vec{q}}) + \vec{G}(\vec{q})$) could be calculated separately, using the computer algebra approach for the inverse dynamic problem, presented in the previous work [13].

The elements of the generalized inertia matrix, according to the Uicker–Kahn method, could be expressed in the following form [2], [11]:

$$H_{ij} = \sum_{k=1}^j \left[\text{trace} \left(\frac{\partial W_j}{\partial q_i} J_j \frac{\partial W_k}{\partial q_k} \right) \right], \quad (2)$$

where H_{ij} are the elements of the generalized inertia matrix; q_i is a generalized coordinate of the i -th joint; J_j is the inertia matrix of the j -th link with respect to the local coordinate system.

The transformation matrix W_i between the i -th local coordinate system and the reference system can be expressed as

$$W_i = A_0^1 A_1^2 \dots A_{i-1}^i, \quad (3)$$

where A_{k-1}^k is a (4×4) – homogenous transformation matrix between two local coordinate systems, and it is of the form:

$$A_{k-1}^k = \begin{bmatrix} \tilde{A}_{k-1}^k & \vec{b}_{k,k-1} \\ O & I \end{bmatrix}, \quad (4)$$

where $\tilde{A}_{k-1}^k$, $\vec{b}_{k,k-1}$ are rotation and transition transformations between two local coordinates; O and I mean zero and unit matrices, respectively. Transformation matrices are of the shape [15]:

$$\tilde{A}_{k-1}^k = \begin{bmatrix} \cos q_k & -\cos \alpha_k \sin q_k & \sin \alpha_k \sin q_k \\ \sin q_k & \cos \alpha_k \cos q_k & -\sin \alpha_k \cos q_k \\ 0 & \sin \alpha_k & \cos \alpha_k \end{bmatrix}, \quad (5)$$

$$\vec{b}_{k,k-1} = \begin{bmatrix} a_k \\ d_k \sin \alpha_k \\ d_k \cos \alpha_k \end{bmatrix}, \quad (6)$$

where α_k, a_k, d_k are kinematic parameters of the joint k .

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

The flexible robot with a large number of joints is shown schematically in Figure 1. The robot is composed of cylindrical piezoceramic transducers and spheres, made from passive material, in this case, from steel [4], [5]. The contact force between the spheres and cylindrical piezoceramic transducers is maintained with the aid of permanent magnets. Here the resonant oscillations of each piezoelectric transducer are controlled by a microprocessor that switches 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. High-frequency resonant mechanical oscillations of ultrasonic frequency cause motions (rotations) in all directions and, at the contact zone, they turn to continuous motion.

The external torque vector, appearing in the gripper and rotating on the plane perpendicular to the gripper direction, is calculated by the computer algebra approach described in [3]. Dynamic simulation of this kind of flexible robots is a very complicated problem, because there are two types of motions – continuous and vibration [7], [13].

3 Computer Algebra Implementation

The methods using N-E or G-A equations are in principle complex because of the complexity to eliminate the constraints by forces and moments. Moreover, they do not directly show the algebraic values of the forces and moments due to the action of actuators. The L-E equations provide an opportunity of direct regarding the equations as functions of the system control inputs [13]. However, the inherent unsuitability of applying L-E equations lies in the need to calculate the partial derivatives (see formula (1), which is not a trivial numerical procedure, but very convenient for computer algebra techniques.

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 equations of motion in closed form and may be applied to solving either the direct or the inverse problem of dynamics.

The Uicker–Kahn method was implemented using VIBRAN [8]. The sparse matrix technology was used in this program to achieve the best performance. In order to compare the various results and algorithms, only two joints (6-degrees-of-freedom) of the proposed robot are considered.

All the elements of the generalized inertia matrix H_{ij} are calculated in the program making use of formula (2). The elements of the generalized inertia matrix have been computed for the discussed flexible robot with 6-degrees-of-freedom. Table 1 contains the kinematic parameters of this robot in Denavit–Hartenberg's notation [5], [10], [14].

Table 1.

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

In order to avoid the numerical computation of the trigonometric function a substitution was applied $S_i = \sin q_i$, $C_i = \cos q_i$.

The fragment of analytical calculation performed for the generalized inertia matrix of the flexible robot by the VIBRAN program, illustrated in Figure 2 (the expression of the coefficients should be generated automatically).

Twenty one element (only the symmetric part of the generalized inertia matrix) have been calculated in total and some of them are equal to zero. A special VIBRAN procedure generates two FORTRAN subroutines from the obtained analytical expressions of the generalized inertia matrix [3], [8]. The code of the first generated subroutine contains a dictionary of monomials included into the

```
SUBROUTINE robo01(A,O)
IMPLICIT REAL(A-Z)
DIMENSION A(1),O(1)
S1 =A(1)
C1 =A(2)
.....
C6 =A(12)
O(61)=S5*C4
O(63)=S6*S4
O(66)=C5*S4
O(228)=C6*C5
O(62)=O(228)*C4
END
SUBROUTINE robo(A,B)
DIMENSION A(1),B(1),O( 230)
CALL robo01(A,O)
Y1=+.1296E-3*O(1)
Y2=-.9964E-4*O(2)
.....
B(1)=+.8326E-4+Y1+Y2+Y3+Y4+Y5+Y6+Y7+Y8+Y9+Y10+Y11
*+Y12+Y13+Y14+Y15+Y16+Y17+Y18+Y19+Y20+Y21+Y22+Y23
*+Y24+Y25+Y26+Y27+Y28+Y29+Y30+Y31+Y32+Y33+Y34+Y35
.....
END
```

Fig. 2. Fragment of the code of two subroutines for numerical simulation

expressions of robot's matrices. This dictionary of monomials is sorted in ascending order of the monomial multiindices to reduce the number of floating point multiplications. The second code of the generated subroutine contains calculations of common members included in all the expressions and all the elements of robot's matrices. The generated subroutines can be immediately compiled and directly used for numerical simulation.

The number of floating point product operations, required to form the generalized inertia matrix of the robot by the Uicker-Kahn method, numerically depends on n^4 (n – number of degrees-of-freedom) and, vice versa, the recursive methods based on N-E or G-A equations mainly depend on the number of degrees-of-freedom. When using the computer algebra technique, there emerge some differences. By virtue of the Uicker-Kahn method the expressions for the elements of the generalized inertia matrix are found in closed form, meanwhile, other well-known algorithms yield only recursive equations. This fact indicates that only the numerical implementation is possible and therefore this method is suitable for the direct dynamics problem only. The code presented in Figure 2 contains only 144 floating point products and 186 sums. The computational complexity of the proposed approach is comparable with that of the most efficient algorithms known so far, as shown in Table 2.

Table 2.

Authors	Principle	Products ($n = 6$)	Sums ($n = 6$)
Walker and Orin [9]	N-E	$12n^2 + !56n! - !27(741)$	$7n^2 + !67n! - !56(598)$
Angeles and Ma [1]	N-E	$n^3 + !17n^2 - !21n! + !8(710)$	$n^3 + !14n^2 - !16n! - !5(629)$
Mata et al. [11]	G-A	$11.5n^2 + !19.5n! - !49(482)$	$8.5n^2 + !31.5n! - !69(426)$
This work	L-E	144	186

Some remarks could be made to explain these results. First of all, computer algebra systems work very efficiently with a large number of short expressions, which enables an effective simplification of these expressions during analytical computation. It appears that a lot of numerical methods are developed especially to avoid numerical differentiation and most of them are recursive, which is inconvenient for analytical computation. However, the calculation of derivatives is a very simple procedure for computer algebra systems.

4 Conclusions

The expressions for the generalized inertia matrix of the robots with a large number of joints have been obtained using the Uicker-Kahn method, based on Lagrange-Euler's equations, and realized by the computer algebra technique. The computational complexity of the proposed approach is comparable with that of the most efficient algorithms known so far.

The proposed analytical implementation of Uicker-Kahn's method drastically reduces the number of floating point operations, particularly for the robots with

a large number of joints. This approach performs the efficient simulation of dynamic behavior of the robot.

References

1. Angeles, J., Ma, O.: Dynamic simulation of n -axis serial robotic manipulators using a natural orthogonal complement, *Int. J. Rob. Res.* 7 (5) (1988) 32–37
2. Balafoutis, C.A., Patel, R.V.: *Dynamic Analysis of Robot Manipulators: A Cartesian Tensor Approach*, Kluwer Academic Press, Boston (1991)
3. Bansevicius, R., Cepulkauskas, A., Kulvietiene, R., Kulvietis, G.: *Computer Algebra for Real-Time Dynamics of Robots with Large Number of Joints*. Lecture Notes in Computer Science, Vol. 3039. Springer-Verlag, Berlin Heidelberg New York (2004) 278–285
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. Barauskas, R., Bansevicius, R., Kulvietis, G., Ragulskis, K.: *Vibromotors for Precision Microrobots*. Hemisphere Publishing Corp., USA (1988).
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. Mata, V., Provenzano, S., Valero, F., Cuadrado, J., I.: An $O(n)$ algorithm for solving the inverse dynamic problem in robots by using the Gibbs–Appell formulation, *Proceedings of Tenth World Congress on Theory of Machines and Mechanisms*, Oulu, Finland, (1999), 1208–1215
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. Saha, S.K.: A decomposition of the manipulator inertia matrix, *IEEE Trans. Rob. Autom.* 13 (2) (1997) 301–304
13. 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
14. Vucobratovic, K., M., Kircanski M., N.: *Real-time Dynamics of Manipulation Robots*, Springer-Verlag, Berlin Heidelberg New York (1985)
15. Walker, M.W., Orin, D.E.: Efficient dynamic computer simulation of robotic mechanisms, *J. Dyn. Syst. Meas. Control* 104 (1982) 205–211