623. Computer algebra for solving dynamics problems of piezoelectric robots with large number of joints

G. Kulvietis[2], R. Bansevičius[1], A. Čepulkauskas[2], R. Kulvietienė[2]
[1]Kaunas University of Technology
Kęstučio str. 27, LT-44312, Kaunas, Lithuania
E-mail: *bansevicius@cr.ktu.lt*
[2]Vilnius Gediminas Technical University,
Sauletekio str.11, LT-2040, Vilnius, Lithuania
E-mail: *{Algimantas_Cepulkauskas, Regina_Kulvietiene, Genadijus_Kulvietis}@gama.vtu.lt*

(Received 24 April 2011; accepted 15 May 2011)

Abstract. The application of general control theory to complex mechanical systems represents an extremely difficult problem. If industrial piezoelectric robots have large number of joints, development of new control algorithms is unavoidable in order to achieve high positioning accuracy. 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, a computer algebra system VIBRAN was used. Expressions for the generalized inertia matrix of the robots have been derived by means of the computer algebra technique with the following automatic program code generation. As shown in the paper, such application could drastically reduce the number of floating point product operations that are required for efficient numerical simulation of piezoelectric robots.

Keywords: Computer algebra; piezoelectric robots; real-time dynamics; numerical-symbolic computation.

1. Introduction

If industrial piezoelectric robots have large number of joints, the application of such a theory and development of new control algorithms are unavoidable in order to achieve high positioning speed and accuracy. In on-line control, the calculation of model equations must be repeated very often, preferably at sampling frequency that is not lower than 50 Hz. It appears to be necessary to develop computer methods of mathematical modeling for at least two reasons. One of them is that it is impossible to immediately choose the most convenient configuration when designing robots. Thus, it is necessary to analyze a number of different robot configurations and select the most appropriate to the future purpose of the device.

Knowing how complex is a task of writing 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 [8, 14, 15].

Manipulator and robot systems possess several specific qualities in both mechanical and control sense. From the mechanics point of view, a feature that is 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. With respect to control, robot and manipulator systems represent redundant,

multivariable, essentially nonlinear automatic control systems [14]. A manipulation robot is also an example of a dynamically coupled system, and the control task itself is a dynamic task [16].

The methods that model the dynamic behavior of manipulators are divided in 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 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, 11]:

- 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 [17] 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 kinematics 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 [7]. 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 [13], 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 [7].

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 was successfully applied to the inverse dynamic problem of the robot [4].

The first efficient recursive algorithm for the solution of the inverse dynamic problem was proposed by Luh et al. [10]. This algorithm, based on the N-E equations, has been improved repeatedly in the course of years [2, 7]. 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 [16] that uses the L-E equations; and those of Kane and Levinson [16], and Angeles et al. [1], which use Kane's equations. The complexity of the above mentioned numerical algorithms will be compared with computer algebra realization. Some efforts to apply symbolic calculations for dynamics of robot were done [12, 16], however 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 with even complex formulas and expressions. But frequently a semi-analytical 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 numeric-symbolic computation of the real-time dynamics of piezoelectric robots with large number of joints computer algebra system VIBRAN [6, 9] was used [12]. 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 optimized FORTRAN code from obtained analytical expressions, which can be directly used in the programs for the further numerical analysis.

2. Real-time dynamics of robot

The real-time dynamic model of a robot was constructed using Uicker-Kahn's method [12, 16], based on L-E equations, that is very convenient for computer algebra implementation [5, 12]. This method enables the calculation of all the matrices of 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, have the following form:

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

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 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 reference coordinates system, expressed in the local coordinate system of the i-th link; $\vec{g}$ is the gravity vector.

Equation (1) may be expressed in the matrix form

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

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

The flexible piezoelectric robot with a large number of joints is shown schematically in Fig. 1. The robot is composed of cylindrical piezoceramic transducers and spheres, made from passive material, in this case, from steel [3, 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.

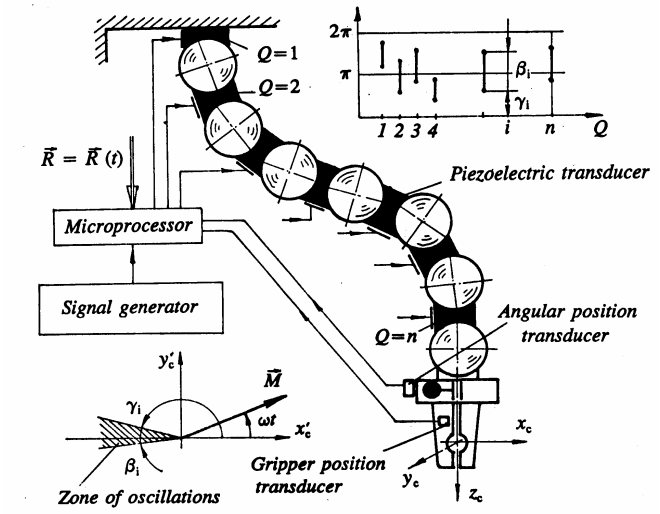

Fig. 1. The scheme of the flexible piezoelectric robot with a large number of joints

The external torque vector placed in the gripper and rotating in the plane perpendicular to the gripper direction expressed in the form [5]:

$$\vec{F} = \begin{Bmatrix} F_x = m_0 r\omega^2 \cos(\omega t) \\ F_y = m_0 r\omega^2 \sin(\omega t) \\ O \end{Bmatrix} \tag{3}$$

where m_0 is the mass of unbalance; r is the 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 the active torques. This algorithm may be expressed in the form:

$$\vec{F}_i = \tilde{A}_i^{i+1}\vec{F}_{i+1}$$
$$\vec{M}_i = \tilde{A}_i^{i+1}\vec{M}_{i+1} + \vec{h}_{i,i-1} \times \vec{F}_{i+1} \quad (4)$$

where $\vec{F}_n = \vec{F}$, see formula (3). Expressions (4) are calculated starting from $i = n-1$ to $i = 1$.

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

$$Q_i = \vec{M}_i \vec{z}_{i0} \quad (5)$$

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

3. Algorithm for calculating the generalized inertia matrix

The algorithm for calculating the generalized inertia matrix has been constructed using the Uicker-Kahn method [13], 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 [4,18], 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 [16, 17]. The equations of the direct dynamic problem formulated by Vucobratovic [16] are contained in the following matrix expression:

$$H(\vec{q})\ddot{\vec{q}} = \vec{P} - \vec{C}(\vec{q},\dot{\vec{q}}) + \vec{G}(\vec{q}), \quad (6)$$

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 [4].

The elements of the generalized inertia matrix, according to Uicker-Kahn method, could be expressed in the form [2, 12]:

$$H_{ij} = \sum_{k=1}^{j}\left[trace\left(\frac{\partial W_j}{\partial q_i} J_j \frac{\partial W_k}{\partial q_k} \right)\right], \quad (7)$$

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:

$$W_i = A_0^1 A_1^2 ... A_{i-1}^i, \quad (8)$$

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

$$A_{k-1}^k = \begin{bmatrix} \tilde{A}_{k-1}^k & \vec{b}_{k,k-1} \\ O & I \end{bmatrix}, \quad (9)$$

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 [17]:

© VIBROENGINEERING. JOURNAL OF VIBROENGINEERING. JUNE 2011. VOLUME 13, ISSUE 2. ISSN 1392-8716

$$\tilde{A}_{k-1}^{k} = \begin{bmatrix} \cos q_k & -\cos\alpha_k \sin q_k & \sin\alpha_k \sin q_k \\ \sin q_k & \cos\alpha_k \cos q_k & -\sin\alpha_k \cos q_k \\ 0 & \sin\alpha_k & \cos\alpha_k \end{bmatrix}, \tag{10}$$

$$\vec{b}_{k,k-1} = \begin{bmatrix} a_k \\ d_k \sin\alpha_k \\ d_k \cos\alpha_k \end{bmatrix} \tag{11}$$

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

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 [4]. Dynamic simulation of this kind of flexible robots is a very complicated problem, because there are two types of motions – continuous and vibration [8, 14].

4. Computer algebra implementation

In the algorithm for automatic generation of the analytical model, it will be assumed that the parameters of a robot (lengths, masses, inertias, 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 computer algebra technique the Uicker-Kahn's method is very convenient, because enables to obtain equations of motion in a closed form and may be applied in solving either the direct or the inverse problem of dynamics.

Fig. 2 illustrates fragment of 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./
.....
RFND(C,NC,J,K,NF3)
RTRN(C,NC)
RMLT(C,NC,B,NB,C,NC,D,E,I)
ADDA(U,D)
100 RSMP(U,E,D,N)
```

Fig. 2. A fragment of the VIBRAN program

This program calculates all elements of matrices $P = H(q)\ddot{q} + \dot{q}^T C(q)\dot{q} + g(q)$. These matrices were calculated for the discussed flexible robot with the 6-degree-of-freedom. The kinematic parameters of this robot in Denavit-Hartenberg's notation [5, 12, 16] are presented in the Table 1.

Table 1. Kinematic parameters of the robot

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:

$S_i = \sin q_i$

$C_i = \cos q_i$

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.

```
 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*C3*C6*C5+.14121E-5*C3*S4*S6-.14121E-5*S4*S3*C6*C5+.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 matrices

A special VIBRAN procedure [6, 9] 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 expressions of robot's matrices. This dictionary of monomials is sorted in ascending order of monomials multi-indices to reduce the number of floating point multiplications. The code of second generated subroutine contains the calculation of common members included in all 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.

The number of floating point product operations required to construct the dynamic model by Uicker-Kahn method numerically depends on n^4 (n is the number of degrees-of-freedom) and, by contrast, the recursive methods based on N-E or G-A equations have a linear dependency on the number of the degrees-of-freedom. Some differences appear using the computer algebra technique. Uicker-Kahn method produces closed-form differential equations and only recursive equations can be obtained from other well-known algorithms which mean that only the numerical implementation is possible and this method suits only for inverse dynamics. The computational complexity of the proposed approach is comparable with that of the most efficient algorithms that are known, as shown in Table 2.

Table 2. Computational complexity of algorithms

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 required 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).

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 Fig. 3 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 3.

Table 3. Computational complexity of algorithms

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.

5. Conclusions

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

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 means of computer algebra technique. The computational complexity of the proposed approach is comparable with that of the most efficient algorithms known so far.

Acknowledgement

This work has been supported by Research Council of Lithuania, Project No. MIP-122/2010.

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] 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.

[4] Bansevicius R., Čepulkauskas 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.

[5] Barauskas R., Bansevicius R., Kulvietis G., Ragulskis K. Vibromotors for Precision Microrobots. Hemisphere Publishing Corp., USA (1988).

[6] Čepulkauskas 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.

[7] 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.

[8] 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.

[9] Kulvietiene R., Kulvietis G. Analytical Computation Using Microcomputers. LUSTI, Vilnius (1989)

[10] Luh J. Y. S., Walker M. W., Paul R. P. On-line computational scheme for mechanical manipulators. J. Dyn. Syst. Meas. Control 102 (1980).

[11] 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.

[12] Rovetta A., Kulvietis G. Lo sviluppo di software per il controllo dinamico di robot industriali. Dipartimento di Meccanica, Politecnico di Milano, Milano (1986).

[13] Saha S. K. A decomposition of the manipulator inertia matrix, IEEE Trans. Rob. Autom. 13 (2) (1997) 301-304.

[14] 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.

[15] Tso S. K., Yang T. W., Xu, W. L., Sun Z. Q. Vibration control for a flexible-link robot arm with deflection feeback. International Journal of Non-Linear Mechanics, 38 (2003) 51-62.

[16] Vucobratovic K. M., Kircanski M. N. Real-time Dynamics of Manipulation Robots, Springer-Verlag, Berlin Heidelberg New York (1985).

[17] Walker M. W., Orin D. E. Efficient dynamic computer simulation of robotic mechanisms, J. Dyn. Syst. Meas. Control 104 (1982) 205-211.

[18] Rahmuone M., Osmont D. Classic finite elements for simulation of piezoelectric smart structures. – Mechanika. – Kaunas: Technologija, 2010, No6 (86), p.50-58.