Robotica (2006) volume 24, pp. 251–256. © 2006 Cambridge University Press
doi:10.1017/S0263574705002456 Printed in the United Kingdom

# A recursive approach for the analysis of snake robots using Kane's equations

H. Tavakoli Nia, H. N. Pishkenari and A. Meghdari*

*Center of Excellence in Design, Robotics and Automation (CEDRA), School of Mechanical Engineering, Sharif University of Technology, Tehran (Iran)*

(Received in Final Form: May 18, 2005)

**SUMMARY**

This paper presents a recursive approach for solving kinematic and dynamic problems in snake-like robots using Kane's equations. An $n$-link model with $n$-nonholonomic constraints is used as the snake robot model in our analysis. The proposed algorithm which is used to derive kinematic and dynamic equations recursively, enhances the computational efficiency of our analysis. Using this method we can determine the number of additions and multiplications as a function of $n$. The proposed method is compared with the Lagrange and Newton-Euler's method in three different aspects: Number of operations, CPU time and error in the computational procedures.

KEYWORDS: Snake robot; Nonholonomic constraints; Kane's equations; Computational efficiency.

## I. INTRODUCTION

This paper discusses the equations of motions of the snake-like articulated robots. Such a robot has attracted attention of many researchers for its capability of multiple functions, such as grasping and locomotion, by varying its shape. Particularly, the mechanism of locomotion is quite different from that of other mobile robots; that is to say, the robot has no driving wheel. Several kinds of snake-like robots have been proposed by many reseachers.[1–4] The Prautsch[3] research deals with an articulated snake-like robot with an actuator in each joint, and a passive wheel in the middle of each link. It is assumed that the wheel does not sideslip.

A dynamic analysis of the mechanism has considerable importance in the field of snake robots. The snake robot can be considered as a serial manipulator with $n$ links that has $n$ constraints. The constraint equations are the no sideslip condition of each wheel in the direction perpendicular to the corresponding link. These constraints are of the nonholonomic type. In the dynamic analysis of such mechanisms, one approach is to derive the equations of motion for the unconstraint system, and then to apply the constraint equations, as it is done by Prautsch and Mita.[3] Hence in our approach, the dynamic analysis of the unconstraint system is reduced to a dynamic analysis of a serial manipulator.

In deriving the equations of motion for a serial manipulator the main task is the derivation of the generalized mass matrix and bias vector. There are several methods for the dynamic analysis of a serial robot. The best-known method was proposed by Walker and Orin.[5] Using the Newton-Euler method, Walker and Orin developed the method of a rigid body, for which the generalized mass matrix is obtained recursively. Angeles and Ma[6] proposed another method that follows this approach, which is based on the calculation of the natural orthogonal complement of the manipulator kinematic constraint equation. In the above methods, in order to apply the nonholonomic constraints, either the degree-of-freedom of the system should be increased or the Lagrange multipliers should be introduced. Hence, the equations describing the motion of the system are increased in number and, as a result, the computational complexity of the system increases considerably. Hence an alternate formulation should be investigated, which would lead to a simpler algorithm in the dynamic analysis of the constrained manipulators. This is the motivation behind this study reported here. In this regard, Kane's equations are investigated first in the realm of snake robots.

Kane and Levinson[7] proposed a method for forward and inverse dynamic of robotic manipulators, which was first introduced by Kane for general nonholonomic mechanical systems. Kane's method implements the concept of generalized speeds as a way to represent motion, similar to what the concept of generalized coordinates does for the configuration. This implementation allows one to focus on the motion aspects of dynamic systems rather than on the configuration. Therefore, it provides a suitable framework for treating nonholonomic constraints. Generalized speeds provide the formulation process with a desired flexibility because they can be chosen to satisfy the needs and interests of the designer, as it is presented by Baruh.[8]

In this paper we intend to derive a form of Kane's equations of motion which is beneficial when we need to write recursive relations. This form of Kane's equations is novel and very beneficial for establishing closed form equations. Our discussions are organized around the following steps:

**Step 1:** An $n$-link model of snake-like robots is constructed.

**Step 2:** An effective form of Kane's equations of motion is demonstrated and the final equations are derived recursively.

**Step3 :** Lagrange and Newton's formulations are presented.

**Step 4:** The effectiveness of different approaches is compared in various aspects.

* Corresponding author. E-mail: meghdari@sharif.edu

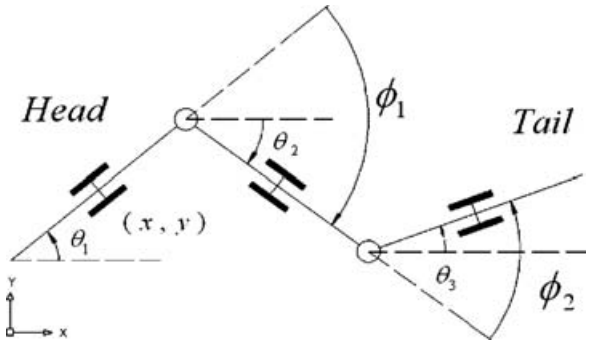


Fig. 1. An n-link snake robot model.

## II. A SNAKE-LIKE ROBOT MODEL

In this research, we use an $n$-link model as a model of the snake like robot (see Figure 1). The center of mass (CM) of each link is considered to be in the middle of the link. Each link has a passive wheel, which does not have sideslip, so the CM motion of each link is restricted to be parallel to the link direction, which means there is a nonholonomic constraint in each link.

In our model, $x$ and $y$ denote the position of the center of mass of the first link, $\theta_i$ $(1 \le i \le n)$ is orientation of the $i$-th link and $\varphi_i$ $(1 \le \mathrm{i} \le n-1)$ is the relative angle of the $(i+1)$-th link to the $i$-th link.

The mass, length and moment of inertia of the $i$-th link are $m_i$, $L_i$ and $I_i$, respectively.

## III. KANE'S EQUATIONS OF MOTION

To begin the formulation of equations of motion, one chooses a set of parameters and variables to describe a certain mass distribution, kinematic and kinetic quantities of the system under consideration. Choosing parameters and variables sounds deceptively simple, but the decisions made here exert a strong influence on the eventual form and numerical efficiency of the equations of motion.

If $x$, $y$ and $\theta_i$ $(1 \le i \le n)$ are identified at any instant, the system configuration is known, so we select the generalized coordinates as follows:

$$q_i = \theta_i (1 \le i \le n) \tag{1a}$$

$$q_{n+1} = x \tag{1b}$$

$$q_{n+2} = y. \tag{1c}$$

After choosing generalized coordinates, generalized speeds are chosen as follows:

$$u_1 = V_1 \tag{2a}$$

$$u_2 = \dot{\theta}_1 \tag{2b}$$

where $V_1$ is the speed of the first link which is parallel to this link. In this stage we intend to obtain $\dot{q}_i$ $(1 \le i \le n+2)$ as a function of $q_i$ $(1 \le i \le n+2)$ and $u_j$ $(1 \le j \le 2)$. Since $q_{n+1}$ and $q_{n+2}$ have not appeared in the equations of motion (these coordinates are called ignorable coordinates), we can obtain $\dot{q}_i$ versus $q_i$ $(1 \le i \le n)$ and $u_j$ $(1 \le j \le 2)$.

The goal is to determine $\dot{q}_i$ $(1 \le i \le n+2)$ with a recursive approach, without using a constraint matrix which increases operations in derivation. We have:

$$\begin{aligned}\mathbf{V_{i+1}} = \mathbf{V_i} + \boldsymbol{\omega_i} \times h_i (\cos\theta_i \hat{\mathbf{I}} + \sin\theta_i \hat{\mathbf{J}}) \\ + \boldsymbol{\omega_{i+1}} \times h_{i+1} (\cos\theta_{i+1} \hat{\mathbf{I}} + \sin\theta_{i+1} \hat{\mathbf{J}})\end{aligned} \tag{3}$$

where $\mathbf{V_i}$ is the velocity of $i$-th link, $\boldsymbol{\omega_i}$ is the angular velocity of the $i$-th link and $h_i = \frac{L_i}{2}$.

We can exert nonholonomic constraints as:

$$\mathbf{V}_\mathrm{i} = V_i (\cos\theta_i \hat{\mathbf{I}} + \sin\theta_i \hat{\mathbf{J}}) \tag{4}$$

$$\boldsymbol{\omega_i} = \omega_i \hat{\mathbf{K}}. \tag{5}$$

If we substitute equations (4) and (5) into equations (3) we have:

$$V_{i+1} = V_i C_i + \omega_i h_i S_i \tag{6}$$

$$\omega_{i+1} = (V_i S_i - \omega_i h_i C_i)/h_{i+1} \tag{7}$$

where $C_i$ denotes $\cos\varphi_i$ and $S_i$ denotes $\sin\varphi_i$.

Thus we have obtained a recursive form of $V$ and $\omega$ which can be used to determine the velocity and angular velocity of each link as a function of $u_1$, $u_2$ and $\varphi_i$ $(1 \le i \le n-1)$. The computational complexity of calculating $V$, the velocity vector, that is its components are $V_i$, is as follows:

$$M = 3n - 3$$

$$A = n - 1.$$

It should be noted that in this paper the computational complexity is measured in terms of numbers of multiplications (M) and additions (A).

For the computational complexity of calculating $\boldsymbol{\omega}$, the angular velocity vector, that its components are $\boldsymbol{\omega_i}$, we have:

$$M = 4n - 4$$

$$A = n - 1.$$

At this stage we introduce a novel form of Kane's equations of motion for this particular problem as follows (in general, some alterations should be created):

$$\mathbf{M}\dot{\mathbf{u}} + \mathbf{N}\boldsymbol{\omega} = \mathbf{F} \tag{8}$$

where $\dot{\mathbf{u}}$ and $\boldsymbol{\omega}$ are the generalized acceleration vector and the angular velocity vector, respectively. $\mathbf{M}$, $\mathbf{N}$ and $\mathbf{F}$ are mass matrix, bias matrix and the generalized force vector, respectively, that are defined as follows:

$$\mathbf{M}(i,j) = \sum_{k=1}^{n} \left( m_k \frac{\partial V_k}{\partial u_i} \frac{\partial V_k}{\partial u_j} + I_k \frac{\partial \omega_k}{\partial u_i} \frac{\partial \omega_k}{\partial u_j} \right) \quad 1 \le i, j \le 2 \tag{9}$$

$$\mathbf{N}(i,j) = \sum_{k=1}^{n} \left( m_k \frac{\partial V_k}{\partial u_i} \frac{\partial V_k}{\partial q_j} + I_k \frac{\partial \omega_k}{\partial u_i} \frac{\partial \omega_k}{\partial q_j} \right) \quad 1 \le i, j \le n \tag{10}$$

$$\mathbf{F}(i) = \sum_{k=1}^{n-1} T_k \left( \frac{\partial \omega_{k+1}}{\partial u_i} - \frac{\partial \omega_k}{\partial u_i} \right) \quad 1 \le i \le 2. \tag{11}$$

In equation (11), $T_k$ denotes the exerted torque to $k$-th joint. Since $x$ and $y$ have not appeared in equation (10), we can neglect $x$ and $y$ when deriving N.

As can be seen, the partial derivatives of $V$ and $\omega$ with respect to $u$ and $q$ have appeared in equations (9) to (11). So there is the necessity to compute these terms efficiently. In what follows, we propose recursive relations for these terms. For the derivatives with respect to $u$ we have:

$$\frac{\partial V_1}{\partial u_1} = 1, \quad \frac{\partial \omega_1}{\partial u_2} = 1 \tag{12}$$

$$\frac{\partial V_{i+1}}{\partial u_j} = \frac{\partial V_i}{\partial u_j} C_i + \frac{\partial \omega_i}{\partial u_j} S_i h_i \tag{13}$$

$$\frac{\partial \omega_{i+1}}{\partial u_j} = \left( \frac{\partial V_i}{\partial u_j} S_i - \frac{\partial \omega_i}{\partial u_j} C_i h_i \right) \Big/ h_{i+1}. \tag{14}$$

Using equations (12) to (14), the computational complexity of calculating all the partial derivatives of $\frac{\partial V_i}{\partial u_j}$ is:

$$M = 6n - 6$$
$$A = 2n - 2.$$

And the computational complexity of calculating the partial derivatives of $\frac{\partial \omega_i}{\partial u_j}$ is:

$$M = 8n - 8$$
$$A = 2n - 2.$$

But for the derivatives with respect to $q$, we initially set all the derivatives equal to zero and then we proceed as follows:

For $i = j - 1$ and $j > 1$

$$\frac{\partial V_{i+1}}{\partial q_j} = -V_i S_i + \omega_i C_i h_i \tag{15}$$

$$\frac{\partial \omega_{i+1}}{\partial q_j} = (V_i C_i + \omega_i S_i h_i)/ h_{i+1}. \tag{16}$$

For $i = j$:

$$\frac{\partial V_{i+1}}{\partial q_j} = \frac{\partial V_i}{\partial q_j} C_i + \frac{\partial \omega_i}{\partial q_j} S_i h_i + V_i S_i - \omega_i C_i h_i \tag{17}$$

$$\frac{\partial \omega_{i+1}}{\partial q_j} = \left( \frac{\partial V_i}{\partial q_j} S_i - \frac{\partial \omega_i}{\partial q_j} C_i h_i - V_i C_i - \omega_i S_i h_i \right) \Big/ h_{i+1}. \tag{18}$$

For $i > j$:

$$\frac{\partial V_{i+1}}{\partial q_j} = \frac{\partial V_i}{\partial q_j} C_i + \frac{\partial \omega_i}{\partial q_j} S_i h_i \tag{19}$$

$$\frac{\partial \omega_{i+1}}{\partial q_j} = \left( \frac{\partial V_i}{\partial q_j} S_i - \frac{\partial \omega_i}{\partial q_j} C_i h_i \right) \Big/ h_{i+1} \tag{20}$$

The calculation of above relations is simplified if we consider the following relations:

$$\frac{\partial V_i}{\partial q_j} = 0, \quad \frac{\partial \omega_i}{\partial q_j} = 0. \tag{21}$$

Using equations (15)–(21), the computational complexity of calculating all the partial derivatives of $\frac{\partial V_i}{\partial q_j}$ is:

$$M = \tfrac{3}{2}n^2 + \tfrac{9}{2}n - 9$$
$$A = \tfrac{1}{2}n^2 + \tfrac{5}{2}n - 5.$$

And the computational complexity of calculating all the partial derivatives of $\frac{\partial \omega_i}{\partial q_j}$ is as follows:

$$M = 2n^2 + 6n - 12$$
$$A = \tfrac{1}{2}n^2 + \tfrac{5}{2}n - 5.$$

Now having computed all the required partial derivatives, it is possible to calculate the main terms $\mathbf{F}$, $\mathbf{N}$ and $\mathbf{M}$ that are appeared in equation (8). The required number of additions and multiplications for calculating $\mathbf{F}$, $\mathbf{N}$ and $\mathbf{M}$ are as follows:

$$M = 4n^2 + 14n - 6$$
$$A = 2n^2 + 10n - 13.$$

Having computed $\mathbf{F}$, $\mathbf{N}$ and $\mathbf{M}$, the linear equations (8) should be solved for $\dot{\mathbf{u}}$. The numbers of required operations for deriving explicit expression for $\dot{\mathbf{u}}$ are as follow:

$$M = 6$$
$$A = 3.$$

The total numbers of additions and multiplications in Kane's method are as follow:

$$M = 3n^2 + 24n - 27$$
$$A = \tfrac{15}{2}n^2 + \tfrac{97}{2}n - 21.$$

## IV. LAGRANGE'S METHOD

As mentioned earlier, the dynamics of a snake robot is equivalent to an $n$-link that is subjected to $n$ velocity constraints. In this section, the equations of motion for an unconstrained $n$-link system, which moves on the ground, will be derived. As it was mentioned earlier, we only show the brief results for the Lagrange's method.

Using equations (1) for the definition of generalized coordinates that are the same with the Kane's method, the kinetic energy is given by:

$$K = \frac{1}{2} \sum_{i=1}^{n} \left( m_i \mathbf{V_i} \cdot \mathbf{V_i} + I_i \dot{\theta}_i^2 \right) \tag{22}$$

where $\mathbf{V_i}$ is determined in equation (3). As there is no variation in potential energy, the equations of motion can

be written as:

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} - Q_i = 0\,(i = 1, \ldots, n+2). \quad (23)$$

The above equation is equivalent to:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{B}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{Q} = \mathbf{0} \quad (24)$$

where $\mathbf{M}(\mathbf{q})$ is the generalized inertia matrix and $\mathbf{B}(\mathbf{q},\dot{\mathbf{q}})$ is called bias vector. The generalized inertia matrix can be calculated by using the following equation:

$$\mathbf{M}(i, j) = \frac{\partial^2 K}{\partial \dot{q}_i \partial \dot{q}_j} \quad (25)$$

where $K$ is introduced in equation (22). Equation (25) is the base of the recursive algorithm that is used to obtain the generalized inertia matrix. The computational complexity of this algorithm is as follows:

$$M = \tfrac{5}{3}n^3 + \tfrac{5}{2}n^2 - \tfrac{19}{6}n - 17$$
$$A = \tfrac{1}{6}n^3 + \tfrac{3}{2}n^2 + \tfrac{4}{3}n - 5.$$

In the next step the bias vector will be derived. The bias vector is calculated by using the algorithm proposed by Walker and Orin.[5] Letting $\ddot{\mathbf{q}} = \mathbf{0}$ in equation (24), it can be seen that the bias vector will be equal to the generalized forces. Eliminating the terms related to $\ddot{\mathbf{q}}$, the centrifugal terms would remain. In other words, it can be said that the work done by these centrifugal forces equals to the work done by generalized forces. So, employing a recursive algorithm, this vector will be obtained with the following computational complexity:

$$M = 10n - 4$$
$$A = 6n - 4.$$

In order to apply the velocity constraints to the system, the following equation should be used:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{B}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{A}(\mathbf{q})\boldsymbol{\lambda} = \mathbf{Q} \quad (26)$$

where $\mathbf{A}(\mathbf{q})$ is the Jacobian matrix and $\boldsymbol{\lambda}$ are the Lagrange's multipliers. The matrix $\mathbf{A}$ has a simple form and it can be calculated with the following computational complexity:

$$M = \tfrac{3}{2}n^2 - \tfrac{3}{2}n$$
$$A = \tfrac{1}{2}n^2 - \tfrac{1}{2}n.$$

Equation (26) forms a system of differential-algebraic equations. One of the well-known methods for solving such systems is the augmented method discussed by Ginsberg.[9] In this method the system of equations is written in the following form:

$$\begin{bmatrix} \mathbf{M} & -\mathbf{A}^{\mathbf{T}} \\ \mathbf{A}^{\mathbf{T}} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} \\ \dot{\mathbf{A}}\dot{\mathbf{q}} - \dot{\mathbf{b}} \end{Bmatrix} \quad (27)$$

where $\dot{\mathbf{A}}$ is the derivative of the constraint matrix with respect to time. Since the constraints are independent of time, $\dot{\mathbf{b}}$ vanishes. The matrix $\dot{\mathbf{A}}$ also has a simple form. The computational complexity of calculating this matrix is as follows:

$$M = \tfrac{3}{2}n^2 - \tfrac{1}{2}n$$
$$A = n^2 - n.$$

In the next step the linear equations (27) should be solved for the vector $\ddot{\mathbf{q}}$. For this task the method of Triangular Decomposition used by Nobel.[10] The computational complexity is as follows:

$$M = \tfrac{4}{3}n^3 + 10n^2 + \tfrac{44}{3}n + 6$$
$$A = \tfrac{4}{3}n^3 + 8n^2 + \tfrac{29}{3}n + 3.$$

The total numbers of addition and multiplication in this method are:

$$M = 3n^3 + \tfrac{31}{2}n^2 + \tfrac{39}{2}n - 11$$
$$A = \tfrac{3}{2}n^3 + 11n^2 + \tfrac{31}{2}n - 2.$$

## V. NEWTON-EULER'S METHOD

In order to compare the proposed method with another known method, the formulation that was introduced by Walker and Orin[5] is chosen. This method is one of the best methods that have used Newton-Euler equations to derive the equations of motion for an $n$-link system. The computational complexity of each step is as follows:

Computing the generalized inertia matrix:

$$M = 12n^2 + 56n - 27$$
$$A = 7n^2 + 67n - 53.$$

Computing the bias vector

$$M = 137n - 22$$
$$A = 101n - 11.$$

Computing the Jacobian matrix (This part is similar to what is done in Lagrange's method):

$$M = \tfrac{3}{2}n^2 - \tfrac{3}{2}n$$
$$A = \tfrac{1}{2}n^2 - \tfrac{1}{2}n.$$

Computing the derivative of Jacobian matrix (This part is also similar to what is done in Lagrange's method):

$$M = \tfrac{3}{2}n^2 - \tfrac{1}{2}n$$
$$A = n^2 - n.$$

Table I. Number of operations in each method.

| Method | General $n$ | | $n=3$ | | $n=12$ | | $n=20$ | |
|---|---|---|---|---|---|---|---|---|
| | A | M | A | M | A | M | A | M |
| Kane | $\frac{15}{2}n^2+\frac{97}{2}n-21$ | $3n^2+24n-27$ | 72 | 192 | 693 | 1641 | 1653 | 3949 |
| Lagrange | $\frac{3}{2}n^3+11n^2+\frac{31}{2}n-2$ | $3n^3+\frac{31}{2}n^2+\frac{39}{2}n-11$ | 184 | 268 | 4360 | 7639 | 16708 | 30579 |
| Newton-Euler | $\frac{4}{3}n^3+25n^2+\frac{617}{3}n-43$ | $\frac{4}{3}n^3+\frac{33}{2}n^2+\frac{1057}{6}n-50$ | 663 | 835 | 6744 | 8329 | 20740 | 24737 |

Solving the system of equation for vector $\ddot{\mathbf{q}}$ using Triangular Decomposition:

$$M=\tfrac{4}{3}n^3+10n^2+\tfrac{44}{3}n+6$$
$$A=\tfrac{4}{3}n^3+8n^2+\tfrac{29}{3}n+3.$$

The total numbers of additions and multiplications in this method are:

$$M=\tfrac{4}{3}n^3+\tfrac{33}{2}n^2+\tfrac{1057}{6}n-50$$
$$A=\tfrac{4}{3}n^3+25n^2+\tfrac{617}{3}n-43.$$

## VI. COMPARATIVE DISCUSSION

In this section we intend to have a comparison between different approaches presented in this paper. This comparison will be accomplished in various aspects:

- Number of operations
- CPU time
- Error in computations

### *VI.1. Number of operations*

The numbers of algebraic operations in each method are listed in Table I. We have also listed the numbers of operations for different values of $n$, in this table. As it can be seen, the order of required operations in the Kane's method is $O(n^2)$, while the operation order in the two other methods is $O(n^3)$.

### *VI.2. CPU time*

To compare the CPU time between the Kane and Lagrange's methods a simulation is done. In this simulation, the execution time for different values of $n$ is determined in each method and results are plotted in Figure 2. Parameters and initial conditions used in this simulation are listed in Table II. As is expected, the execution time in Kane's method is much smaller than that in Lagrange's method.

Table II. System parameters and initial conditions.

| | |
|---|---|
| $T_k$ | $\sin(t)/(n-k)$ N.m |
| $L_k$ | 0.2 m |
| $m_k$ | 1 kg |
| $I_k$ | 0.005 kg m$^2$ |
| $q_k$ | $\pi\times(-1)^k/6$ |
| $\{\mathbf{u_0}\},\ \{\dot{\mathbf{q}}_\mathbf{0}\}$ | 0 |

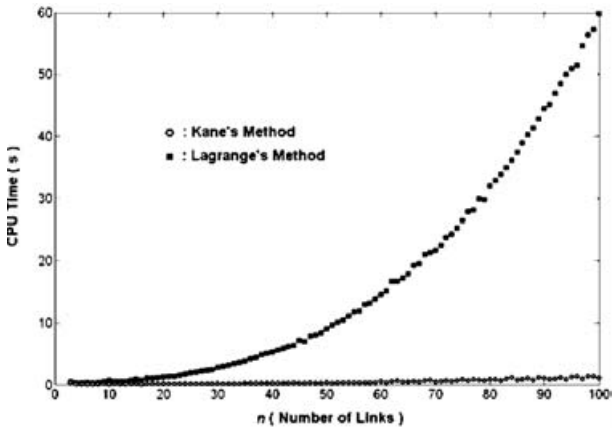


Fig. 2. CPU time in Kane and Lagrange's method.

To have a better view of difference of CPU time in these methods, we have plotted the ratio of the CPU time for Lagrange's method to the CPU time for Kane's method in Figure 3. As it can be seen, this ratio is approximately a linear function of $n$.

### *VI.3. Error in computations*

We have investigated the error in computation in the two different aspects. The first aspect of comparison is the difference between responses obtained from the different precisions in our solving procedure. To achieve this comparison, a simulation with listed conditions in Table III is accomplished. The results are plotted in Figures 4 and 5. As it can be observed, the single precision result is closer to the double precision result in Kane's method, while

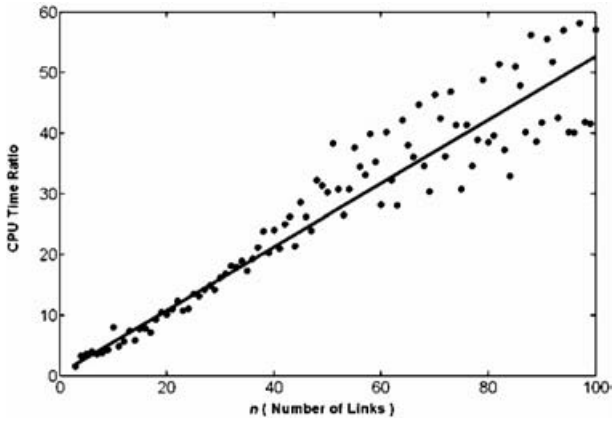


Fig. 3. CPU time ratio.

Table III. System parameters and initial conditions.

| | |
|---|---|
| $N$ | 30 |
| Simulation Time | 100 s |
| $T_k$ | 1 N.m |
| $L_k$ | 0.2 m |
| $m_k$ | 1 kg |
| $I_k$ | 0.005 kg m$^2$ |
| $q_k$ | 0 |
| $\{\mathbf{u_0}\}$, $\{\dot{\mathbf{q}}_\mathbf{0}\}$ | 0 |

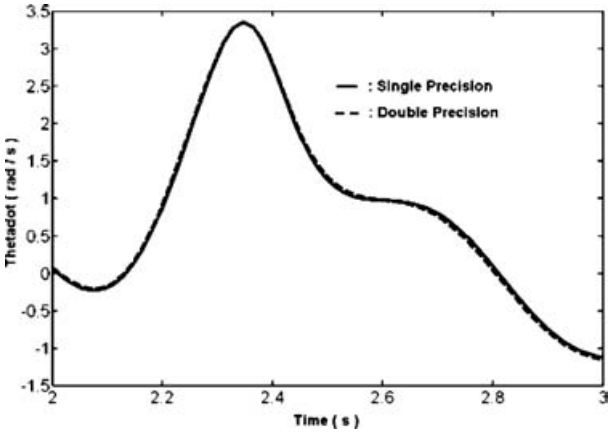


Fig. 4. Kane's method response for different precision in the solving procedure.

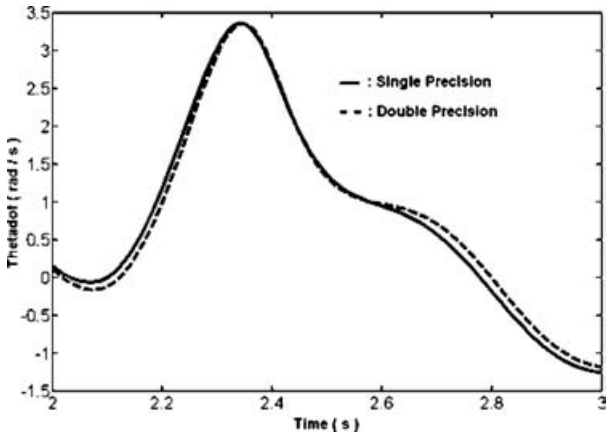


Fig. 5. Lagrange's method response for different precision in the solving procedure.

in Lagrange's method there is a considerable difference between single and double precision results.

The second aspect of comparison is based on the error in the energy function that is derived from the computation error. The energy function is defined as below:

$$S = K(t) - \int_0^t \sum_{i=1}^{n-1} T_i(\omega_{i+1} - \omega_i)\, dt$$

The model parameters and initial conditions used in this simulation are listed in Table III. The results of the simulation are shown in Figure 6. As it can be seen, the error in Kane's method is very smaller than that in Lagrange's method.

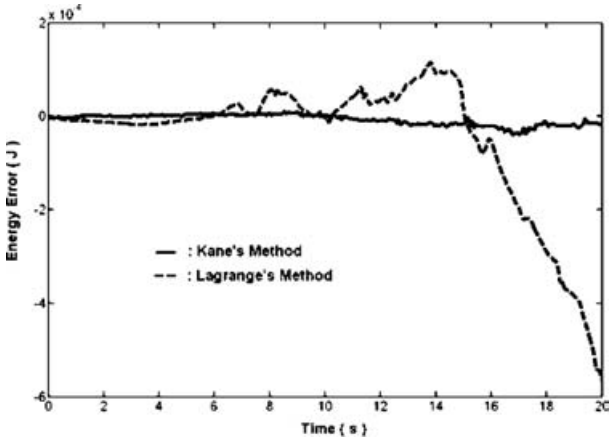


Fig. 6. Energy error in Kane's and Lagrange's method.

The main cause of the error in Lagrange's equations arises from differentiating nonholonomic constraints. Of course, this problem is solvable, but it involves increasing the required operations and the CPU time.

## VII. CONCLUDING REMARKS

In this paper we have presented an effective and novel form of Kane's equations of motion, which can be utilized for deriving a recursive formulation of a snake-like robot. The formulation has been obtained as a function of $n$, the number of the links. Then the Lagrange and Newton's equations of motion have been derived. Finally, we have compared these methods in three different aspects, *viz*. the number of operations, the CPU time and the computation error. We have proved that Kane's method is the most suitable approach for the kinematic and dynamic analysis of the snake robots, and also it is an appropriate method for deriving closed-form equations.

## References

1. S. Hirose and A. Morishima, "Design and control a mobile robot with an articulated body", *Int. J. Robotic Research* **9**, No. 2, 99–114 (1990).
2. G. Migadis and J. Kyriakopoulos, "Design and forward kinematic analysis of a robotic snake", *IEEE Int. Conf. on Robotics and Automation* (1997) pp. 3493–3498.
3. P. Prautsch and T. Mita, "Control and Analysis of the gait of snake robots", *IEEE Int. Conf. on Control Applications* (1999), pp. 502–507.
4. K. Sarrigeorgidis and K. J. Kyriakopoulos, "Stabilization and trajectory tracking of a robotic snake", *IEEE Int. Conf. on Robotics and Automation* (1998), (1998) pp. 2977–2981.
5. M. W. Walker and D. E. Orin, "Efficient Dynamic Computer simulation of robotic mechanisms", *Journal of Dynamic System, Measurement and Control* **104**, 205–211 (1982).
6. J. Angeles and O. Ma, "An algorithm for the inverse dynamic of n-axis general manipulators using Kane'2 equations", *Computer Math. Applic.* **17**, No. 12, 1545–1561 (1989).
7. T. R. Kane and D. A. Levinson, *Dynamics: Theory and Applications*, 1st edition, Series in Mechanical Engineering (McGraw-Hill, New York, 1985).
8. H. Baruh, *Analytical Dynamics*, International Editions (McGraw-Hill Press, New York, 1999).
9. J. H. Ginsberg, *Advanced Engineering Dynamics*, 2nd Edition (Cambridge Univ. Press, Cambridge, 1998).
10. B. Nobel, *Applied Linear Algebra* (Prentice-Hall, New Jersey, 1969).