



# Journal of Applied and Computational Mechanics



Research Paper

## An Automatic Program of Generation of Equation of Motion and Dynamic Analysis for Multi-body Mechanical System using GNU Octave

Yonghui Park<sup>ID</sup>

Department of Mechanical Engineering, Yuhan University, 590, Gyeongin-ro, Bucheon-si, Gyeonggi-do, Republic of Korea

Received June 02 2020; Revised July 30 2020; Accepted for publication July 30 2020.

Corresponding author: Y. Park (yhpark@yuhan.ac.kr)

© 2021 Published by Shahid Chamran University of Ahvaz

**Abstract.** Multi-body dynamics is used to calculate the physical quantities required for component design, such as calculating the dynamic response of mechanical components and the time history of dynamic loads. Advances in analysis software, including DADS, ADAMS, RecurDyn, and DAFUL, have made it possible to easily calculate dynamic responses by defining relationships between components and operating environments from 3D modeling on user-created components. However, when the understating of dynamic analysis is lacking, it is difficult to apply multi-body dynamics analysis in the design process, and it is difficult to analyze the acquired response data. In this study, we developed an automatic code to derive equations of motion in the matrix format and calculate dynamic responses of multi-body systems using GNU Octave, a free high level language. In particular, the process of defining matrices and vectors such as inertia matrix, stiffness matrix, and external force vector concerning the degrees of freedom of components by using Euler-Lagrange equations is shown to understand the structure and process of dynamic analysis. The code application by explaining how to use the code in a different mechanical system is also shown to help understand the usage method for who wants to study Multi-body dynamics.

**Keywords:** Multi-body dynamics, Equation of motion, Euler-Lagrange equation, Numerical integration, GNU Octave.

### 1. Introduction

Multi-body dynamics analysis is a study that analyzes the dynamic characteristics of a multi-body system in which multiple objects are connected and have been developed mainly in the fields of machinery, aerospace, and robots. In the machinery field dealing with closed chain systems, the system design process brought a lot of attention to deriving the optimal design through sensitivity analysis between design variables and dynamic characteristics, and in the aerospace field dealing with open-chain systems, research on the safety of dynamic response of the system was mainly considered [1]. About the history of multi-body dynamics, improving the method of deriving equations of motion, expanding the scope of analysis, deriving efficient calculation algorithms with improved hardware computational performance, and improving the software for user convenience were developed in priority. As a result of the development, several multi-body dynamics software such as ADAMS, MESA VERDE, and NEWEUL have been commercialized in overseas, and RecurDyn and DAFUL programs were commercialized in South Korea [2]. Multi-body dynamics has become an essential discipline for designing a dynamic mechanical system because it can calculate position, velocity, and acceleration, and calculate essential physical quantities such as time history of dynamic loads acting on each component. Even though most large manufacturers dealing with the finished system, have prepared the process to study the relationship among each mechanical element with Multi-body dynamics, small manufacturers dealing with the mechanical element only think about the strength and stiffness of the component and do not have human resources that can analyze the product in the variety of ways of engineering. In other words, they have insufficient opportunity to learn multi-body dynamics analysis and could not consider how to apply the technology to their work due to the limitation of the business model. In particular, the business structure of small manufacturer depending on the large manufacturer can no longer be competitive, because it is difficult to secure sales channels, such as the narrow domestic market and emerging countries' participation in the world manufacturing industry. From this point of view, they need to expand their engineering activities such as multi-body dynamics analysis to expand their products. Multi-body dynamics can play a very good role in enhancing the ability to look at the design from the view of the system. Additionally, they will have some opportunities to work with other manufacturers by Mass customization and by providing a solution service that assists customer's engineering activities. Accordingly, it is necessary to develop a free program for small and medium-sized manufacturers of workers, prospective workers, or students to understand the basic theory of multi-body dynamics and partially utilize it in production activities.

However, commercial software capable of inducing motion of equations automatically and calculating dynamic characteristics has already been developed [3], and theoretical study on the method of deriving dynamic models for efficient control in the robot



industry [4-7] have been studied actively, so the need for program development is rather ambiguous. In particular, a MATLAB package program has been developed to derive a linear motion equation of a robot using the TMT method and calculate dynamic characteristics automatically [8-9]. However, it is hard to understand the theory for those who lack experience in the robot field, because some mathematical and scientific background are needed to understand the structure of the program before using it. Besides, the additional cost investment, such as purchasing commercial software and requiring human resources to operate and maintain it, also causes difficulties in access.

In this study, a basic program of multi-body dynamics analysis was developed to enable small manufactures to produce traditional mechanical elements such as fastening, power transmission, and shaft to cultivate design ability from a system perspective. Using the GNU Octave program and the Symbolic package, we developed the code of derivation of motion of equations and matrix transformation for the multi degree of freedom system and combined the code to derive numerical solutions using the Newmark integration method. By introducing the mathematical expression and symbolic processing method represented by the code, a manual that is easy for beginners such as a worker and student was provided.

## 2. Theoretical background

### 2.1 Numerical integration

Before deriving equations of motion regarding the generalized coordinate, the code has a structure that derives a numerical solution starting with input including design variables and environmental conditions and calculates the dynamic responses with the Newmark method (Fig. 1). The Newmark method updates the state variables by calculating the initial acceleration by substituting the initial displacement and initial velocity of each coordinate into the equations of motion. Then, based on the assumptions of the time integration method, such as the average acceleration method and the constant acceleration method, calculating the displacement, velocity, and acceleration of each time proceeds as a certain unit time passes sequentially [10]. Since various calculation methods and codes have been established to calculate the response, it is not a big problem to apply the numerical integration method necessary for the problem that the user wants to deal with the specific area [11]. The important thing is to derive the equations of motion of the system, to derive mass, damping, and stiffness matrices from the equations of motion and input the matrices into the numerical integration code. In other words, if you can define the number of degrees of freedom and coordinates required to express the system, can derive the equation of motion, and can express it in the matrix form, it is very helpful in understanding the process of dynamic analysis.

### 2.2 Equation of motion

An equation of motion that describes the motion, which is the position, velocity, and acceleration of an object over time, is expressed as a second order differential equation [12]. Various types of equations can be derived according to the method of deriving the equation of motion, but Newton's second law is the basis (Eqs. (1-2)).  $F$  and  $T$  are external force and torque respectively,  $M$  and  $I$  are mass and moment of inertia respectively,  $x$ ,  $v$ , and  $a$  are displacement, velocity, and acceleration respectively, and  $\theta$ ,  $\omega$ , and  $\alpha$  are angular displacement, angular velocity, and angular acceleration, respectively.

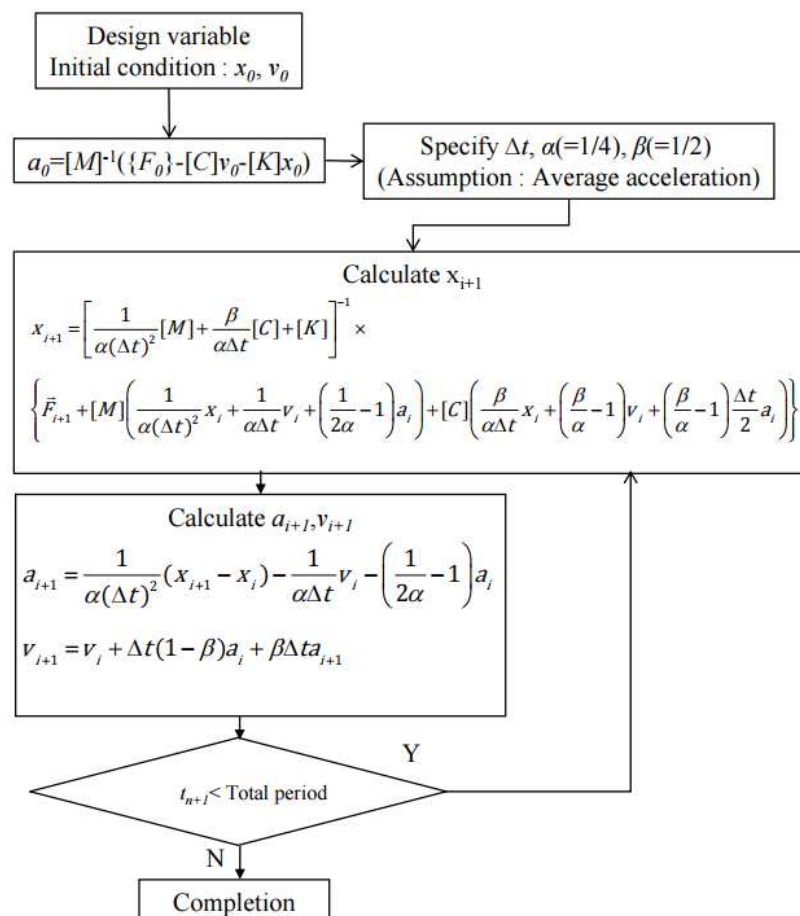


Fig. 1. Flow of the numerical integration



The Newton's second law may vary depending on the translational and rotational motions, but represents the relationship between external force or moment, mass or moment of inertia, and acceleration or angular acceleration. When the object is converted from a simple object to a system with a mechanical element, the second law is used as Eqs. (3-4), because the mechanical element must maintain a shape to enable repetitive motion, and it must be able to perform a role of transmitting motion by combining with other mechanical elements. In other words, it must have the resistance to withstand the mechanical forces such as the reaction force from other elements. Damping  $C$  and stiffness  $K$  are included in the equation of motion to express the mechanical forces.

The equation of motion of the mechanical system is standardized only depending on the number of the degree of freedom. The degree of freedom is the product of the number of objects such as mass or moment inertia and the number of motion types each object can move. In Fig. 2, there are 2 masses in the system, and each mass has one type of motion that moves in the horizontal direction, so the degree of freedom of the system is  $2 \times 1 = 2$ . When the number of the degree of freedom is determined, the equations of motion and the matrices such as mass matrix  $[M]$ , damping matrix  $[C]$ , and stiffness matrix  $[K]$  from the equations of motion, are determined. For instance, the size of the matrices is 1 by 1 in case of a single degree of freedom, and the size of the matrices is  $N$  by  $N$  in case of an  $N$ -th degree of freedom. In Fig. 2, two equations of motion are derived and the size of the matrix is 2 by 2.

$$F = Ma = M\ddot{u} = M\ddot{x} \quad (1)$$

$$T = I\alpha = I\dot{\omega} = I\ddot{\theta} \quad (2)$$

$$F = M\ddot{x} + C\dot{x} + Kx \quad (3)$$

$$T = I\ddot{\theta} + C_t\dot{\theta} + K_t\theta \quad (4)$$

It is easy to derive the equations of motion by defining the relationship between mass, damping, and stiffness around each mass in Fig. 2. However, when deriving the equations of motion of a system in which the number of the degree of freedom increases and both translational and rotational motions exists, the Newtonian approach to vector perspective that synthesizes or decomposes a force component according to each direction of the coordinate needs a unique sense and lots of training. To solve this difficulty for an complex system, several methods including derivation of the equations of motion with singular mass matrices [13], derivation of the equations of motion with highly nonlinear, non-autonomous, and possibly yield motion [14], and estimation of the actual uncertainties [15], have been proposed, but Lagrange mechanics is most effective to derive the equations of motion regarding the generalized coordinates for an non-complex system. Lagrange's mechanics is an approach to describing motion in terms of scalars using invariants with the generalized coordinate. It has the advantage of deriving the equations of motion in a formalized form. After defining the Lagrangian that includes the kinetic energy and potential energy of the system (Eq. (5)), the equation of motion regarding the generalized coordinate is derived from the mathematical operation as Eq. (6). This process is based on the principle of least action that chooses a minimum value with respect to all paths between the points that correspond to the same energy when motion between any two points is in a conservative dynamical system.

$$L = T - V \quad (5)$$

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \frac{d}{dt} \frac{\partial (T - V)}{\partial \dot{q}} - \frac{\partial (T - V)}{\partial q} = Q \quad (6)$$

Even though it is hard to understand the theory that is based on Lagrange mechanics, including the Hamiltonian principle, there is no great difficulty in understanding the process of deriving the equation of motion mechanically using Eqs. (5-6). In the next chapter, the automatic process to derive the equations of motion and the matrices using Eqs. (5-6) is represented. The user needs to derive a Lagrange equation by defining the number of the degree of freedom and a generalized coordinate system and to define the Lagrangian extracting kinetic and potential energy from a system. To show how to use the code for users that are inexperienced in dynamic analysis, the derivation of the equations of motion and the calculating procedure of dynamic response by applying some examples are contained.

### 3. Code

#### 3.1 System definition

Fig. 3 is a target system selected to explain the use of the developed code. It has two rigid bodies that have  $2m$  [kg] and  $5m$  [kg] respectively, and three translational springs are interconnected to two rigid bodies or between the  $2m$  rigid body and the ground [16]. To obtain the dynamic response of the two bodies, the equations of motion should be derived firstly. The most important thing is to extract the kinetic and potential energy of the system to define the Lagrangian. The kinetic energy of the  $2m$  body is about the translational motion  $q_1$  and the rotational motion  $\theta_1$  about the center of mass of the rigid rod. The kinetic energy of the  $5m$  body is about the translational motion  $q_2$  about the center of mass of the rigid body (Eq. (7)). About the potential energy, if the spring is connected to the bodies, the potential energy terms by the motion between both ends of the rigid rod and the ground, between  $q_1$  and  $q_2$  should be defined. Additionally, if gravity is applied to the system, the potential energy terms of the  $2m$  rigid rod and the  $5m$  rigid body by gravity should be defined (Eq. (8)).

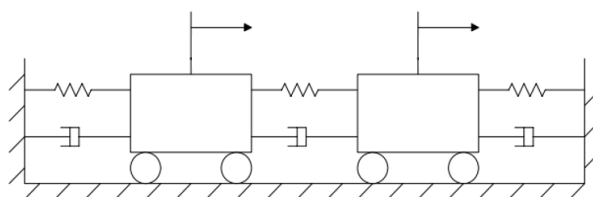


Fig. 2. Example of degree of freedom



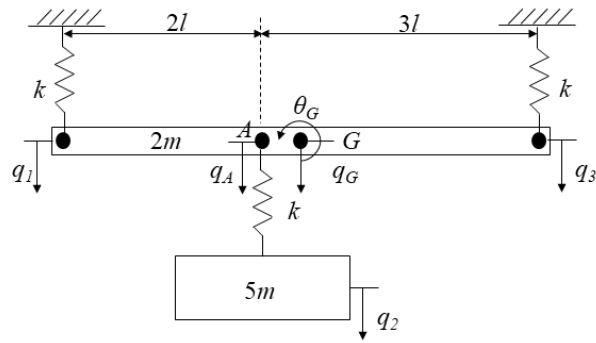


Fig. 3. Example of the system

After defining the kinetic and potential energy, some coordinates that have a geometric relationship to each other can be omitted. That is, unnecessary coordinates are present in Eqs. (7-8) can be removed. Eqs. (9-11) are about the translational and rotational coordinates of the  $2m$  rigid rod and the translational coordinate of the eccentric position A by representing them with the other coordinates.

$$T = \frac{1}{2}(2m)\dot{q}_G^2 + \frac{1}{2}(J_G)\dot{\theta}_G^2 + \frac{1}{2}(5m)\dot{q}_2^2 \quad (7)$$

$$V = \frac{1}{2}kq_1^2 + \frac{1}{2}kq_3^2 + \frac{1}{2}k(q_2 - q_A)^2 + (2m)gq_G + (5m)gq_2 \quad (8)$$

$$\theta_G = \frac{q_1 - q_3}{5l} \quad (9)$$

$$q_G = \frac{q_1 + q_3}{2} \quad (10)$$

$$q_A = q_1 - 2l\theta_G = \frac{3}{5}q_1 + \frac{2}{5}q_3 \quad (11)$$

### 3.2 Derivation of the equations of motion

This section deals with the automatic induction process of the equations of motion from Eqs. (7-11) by applying the process Eq. (6). The Python and GNU Octave & Symbolic packages should be installed on your computer before using the code. The main script consists of the derivation of the equations of motion, the derivation of the mass and stiffness matrix and external force vector, and the calculation of the dynamic response (Fig. 4). The derivation of the equations of motion consists of two sub codes. In the first part of the *unconservation\_MakingEquation* code, the displacements  $q_1$ ,  $q_2$ ,  $q_3$ , the velocity coordinates  $\dot{q}_1$ ,  $\dot{q}_2$ ,  $\dot{q}_3$ , and system variables such as  $m$ ,  $l$ ,  $k$ ,  $t$ , and  $g$  are defined as symbolic variables (Fig. 5). Unlike the other variables, the displacements are represented as a function depending on the time variable, and the velocity coordinates are re-defined as the derivation of the displacement respect to the time variable since the displacement and velocity depend on the time variable.

```

clc
clear all
close all

pkg load symbolic

%% Equation of motion
unconservation_MakingEquation
save temp_dtuc.mat dtuc
clear all
load('temp_dtuc.mat')
MakingEquation

%% Mass and stiffness matrix
[M]=mmass(Eq);
[K,Generalizedl]=mstiff(Eq);
%% Gravity force vector
Gravityterms
[Gravity]=mgravity(dgv);
save Matrix.mat M K Generalizedl Gravity
%% Numerical integration
Numerical_analysis

```

Fig. 4. Main script



After defining the basic variables, the second part defines the geometrical relationship to the coordinates and physical quantities defined in Eqs. (9-11). And in the third part, the total kinetic energy  $T$  defined in Eq. (7) is represented by using the defined coordinates. The fourth part is the derivation of the kinetic energy respect to the displacement coordinates as shown in the underline display part in Eq. (12), corresponding to the partial process of the derivation of the Lagrangian (Eq. (6)) In the case of the closed system, the kinetic energy is differentially derived concerning the generalized displacement coordinate, resulting in zero. However, if a system has coupled translational and rotational motions among different objects or is an open system, additional consideration should be given because the terms have non-zero values. For example, in the trailer composite pendulum, a nonlinear term should be applied as an external force due to the coupled motion between the translational motion of the trailer and the rotational motion of the composite pendulum [16]. Through this procedure,  $dtuc$  term that is sub-lined in Eq. (12) is obtained and is stored as a temporary file to load the results easily for calculating other terms of the main script to derive the equation of motion.

$$\frac{d}{dt} \left( \frac{\partial T}{\partial \dot{q}} - \frac{\partial V}{\partial \dot{q}} \right) - \left( \frac{\partial T}{\partial q} - \frac{\partial V}{\partial q} \right) = Q \quad (12)$$

The second code, *MakingEquation*, is deriving the other terms of the equations of motion. The code from the definition of the symbolic variable to the definition of the total kinetic energy  $T$  is omitted as it is the same as *unconservation\_MakingEquation* (Fig. 5). Part 1 calculates the term that is sub-lined in Eq. (13). To be possible of differentiating the Lagrange's equation regarding the velocity coordinates  $dq1$ ,  $dq2$ , and  $dq3$ , the velocity variables of  $\text{diff}(q1(t),t)$ ,  $\text{diff}(q2(t),t)$ , and  $\text{diff}(q3(t),t)$  were replaced into  $dq1$ ,  $dq2$ , and  $dq3$  temporarily, and the velocity variables were changed into  $dq1(t)$ ,  $dq2(t)$ , and  $dq3(t)$  by applying the time variable to be possible of the time derivative.

$$\frac{d}{dt} \frac{\partial T}{\partial \dot{q}} - \frac{d}{dt} \frac{\partial V}{\partial \dot{q}} - \left( \frac{\partial T}{\partial q} - \frac{\partial V}{\partial q} \right) = Q \quad (13)$$

To perform the sub-lined calculation in Eq. (14), the sum of potential energies  $V$  by the spring was calculated, and the partial derivative regarding the generalized coordinate was performed in Part 2.

$$\frac{d}{dt} \frac{\partial T}{\partial \dot{q}} - \frac{d}{dt} \frac{\partial V}{\partial \dot{q}} - \left( \frac{\partial T}{\partial q} - \frac{\partial V}{\partial q} \right) = Q \quad (14)$$

Part 3 is the derivation process of the equations of motion (Eq. (6)) by using the terms derived from Eqs. (12-14). The term with the partial derivative of the potential energy regarding the velocity is not defined because the term is zero in a conservative system without energy loss. Through the above process, the equations of motion for  $q1$ ,  $q2$ , and  $q3$  were derived (Fig. 7). Gravity is applied to the external force of the equation of motion and is similar to derive the terms as the *MakingEquation* code.

<pre>pkg load symbolic % coordinate and variable syms q1 q2 q3 ... % displacement dq1 dq2 dq3 ... % velocity m l k ... % mass, length, spring stiffness t g ... % time, gravity  % Define temporary displacement variable % including time variable tmp1=[char(q1) '(t)']; % tmp2=[char(q2) '(t)']; % tmp3=[char(q3) '(t)']; %  % Define temporary velocity variable % including time variable tmp37=[char(dq1) '(t)']; % tmp38=[char(dq2) '(t)']; % tmp39=[char(dq3) '(t)']; %  % Substitute the temporary displacement variable % into the coordinate and differentiate it q1=subs(q1,q1,tmp1); dq1=diff(q1,'t'); q2=subs(q2,q2,tmp2); dq2=diff(q2,'t'); q3=subs(q3,q3,tmp3); dq3=diff(q3,'t');</pre>	Part 1	<pre>% Lagrange's equation % Kinetic Energy term T=(1/2)*(2*m)*(d_x_G^2) ... +(1/2)*J_G*(d_theta_G^2)+(1/2)*(5*m)*(dq2^2);</pre>	Part 3
<pre>% Define variables from the geometric relation x_G=(q1+q3)/2; d_x_G=(dq1+dq3)/2; d_theta_G=(dq1-dq3)/5/1; x_A=(3/5)*q1+(2/5)*q3; J_G=25/6*m*l^2</pre>	Part 2	<pre>% Change diff(qi(t),t) to dq1 in T % for partial derivative T=subs(T,'diff(q1(t),t)','dq1'); T=subs(T,'diff(q2(t),t)','dq2'); T=subs(T,'diff(q3(t),t)','dq3');  % Change qi(t) to q1 in T for partial derivative T=subs(T,'q1(t)','q1'); T=subs(T,'q2(t)','q2'); T=subs(T,'q3(t)','q3');  % Partial derivate of T regarding each coorinate dtuc(1)=diff(T,'q1'); dtuc(2)=diff(T,'q2'); dtuc(3)=diff(T,'q3');  for i=1:1:3; dtuc(i)=subs(dtuc(i),'diff(dq1(t),t)','ddq1(t)'); dtuc(i)=subs(dtuc(i),'dq1','dq1(t)'); dtuc(i)=subs(dtuc(i),'q1','q1(t)');  dtuc(i)=subs(dtuc(i),'diff(dq2(t),t)','ddq2(t)'); dtuc(i)=subs(dtuc(i),'dq2','dq2(t)'); dtuc(i)=subs(dtuc(i),'q2','q2(t)');  dtuc(i)=subs(dtuc(i),'diff(dq3(t),t)','ddq3(t)'); dtuc(i)=subs(dtuc(i),'dq3','dq3(t)'); dtuc(i)=subs(dtuc(i),'q3','q3(t)'); end</pre>	Part 4

Fig. 5. The script of *unconservation\_MakingEquation*





<pre> % Change diff(q1(t),t) to dq1 in T % for partial derivative T=subs(T,'diff(q1(t),t)','dq1'); T=subs(T,'diff(q2(t),t)','dq2'); T=subs(T,'diff(q3(t),t)','dq3');  % Partial derivate of T % regarding each velocity coordinate tt(1)=diff(T,'dq1'); tt(2)=diff(T,'dq2'); tt(3)=diff(T,'dq3');  % Define temporary velocity variable % including time variable again tmp37=[char(dq1) ' (t) ']; % tmp38=[char(dq2) ' (t) ']; % tmp39=[char(dq3) ' (t) ']; %  % Change qi and dqi to qi(t) % and dqi(t) respectively in tt(i) for i=1:1:3; tt(i)=subs(tt(i),'dq1','aa1'); tt(i)=subs(tt(i),'dq2','aa2'); tt(i)=subs(tt(i),'dq3','aa3');  tt(i)=subs(tt(i),'q1','q1(t)'); tt(i)=subs(tt(i),'q2','q2(t)'); tt(i)=subs(tt(i),'q3','q3(t)');  tt(i)=subs(tt(i),'dq1','dq1(t)'); tt(i)=subs(tt(i),'dq2','dq2(t)'); tt(i)=subs(tt(i),'dq3','dq3(t)');  tt(i)=subs(tt(i),'aa1','dq1(t)'); tt(i)=subs(tt(i),'aa2','dq2(t)'); tt(i)=subs(tt(i),'aa3','dq3(t)'); end  % Define time derivate of % the partial derivative of kinetic energy, tt for i=1:1:3; dtt(i)=diff(tt(i),'t'); end </pre>	<div data-bbox="667 163 753 197" style="border: 1px solid black; padding: 2px; text-align: center;">Part 1</div> <pre> V=(1/2)*k*(q1^2)+(1/2)*k*((q2-x_A)^2) ... +(1/2)*k*(q3^2);  % Change qi(t) to qi in Potential energy term V V=subs(V,'q1(t)','q1'); V=subs(V,'q2(t)','q2'); V=subs(V,'q3(t)','q3');  % Partial derivate of V regarding each coordinate dv(1)=diff(V,'q1'); dv(2)=diff(V,'q2'); dv(3)=diff(V,'q3');  % Change qi to qi(t) in dv(i) for time derivative for i=1:1:3; dv(i)=subs(dv(i),'q1','q1(t)'); dv(i)=subs(dv(i),'q2','q2(t)'); dv(i)=subs(dv(i),'q3','q3(t)'); end  % Final Equation : Lagrange's equation % regarding each coordinate for i=1:1:3; Eq(i)=dtt(i)-dtuc(i)+dv(i); end  % Change 'diff' and diff(diff) % to d and dd respectively for i=1:1:3; Eq(i)=subs(Eq(i),'diff(dq1(t),t)','ddq1(t)'); Eq(i)=subs(Eq(i),'diff(q1(t),t)','dq1(t)'); Eq(i)=subs(Eq(i),'q1(t)','q1(t)');  Eq(i)=subs(Eq(i),'diff(dq2(t),t)','ddq2(t)'); Eq(i)=subs(Eq(i),'diff(q2(t),t)','dq2(t)'); Eq(i)=subs(Eq(i),'q2(t)','q2(t)');  Eq(i)=subs(Eq(i),'diff(dq3(t),t)','ddq3(t)'); Eq(i)=subs(Eq(i),'diff(q3(t),t)','dq3(t)'); Eq(i)=subs(Eq(i),'q3(t)','q3(t)'); end  % expand the Lagrange's equations for i=1:1:3; Eq(i)=expand(Eq(i)); end </pre> <div data-bbox="1257 566 1343 600" style="border: 1px solid black; padding: 2px; text-align: center;">Part 2</div> <div data-bbox="1257 728 1343 761" style="border: 1px solid black; padding: 2px; text-align: center;">Part 3</div>
--	---

Fig. 6. The partial script of MakingEquation

### 3.3 Derivation of matrix

The equations of motion derived from Section 3.2 are needed to be applied to the numerical integration method to calculate the response. However, it is necessary to convert the obtained equations of motion into a form acceptable to the numerical integration method. In other words, the equations of motion obtained in Fig. 7 should be converted into the matrix form regarding the generalized acceleration, velocity, and displacement coordinates as Eq. (3).

To substitute the equations of motion into the numerical integration method, the equations of motion should be organized as the inertia matrix and acceleration vector, and stiffness matrix and displacement vector (Eqs. (3), (6)). The *mmass* code removes all non-acceleration terms and extracts the mass matrix by expressing the remaining terms as the multiple of the mass matrix and the acceleration vector (Fig. 8). The *mstiff* code removes all non-displacement terms and extracts the stiffness matrix by expressing the remaining terms as the multiple of the stiffness matrix and the displacement vector. Especially, nonlinear terms that can't be arranged as a matrix form, is extracted as an external force (Fig. 9). By using two codes, the equations of motion defined in Eq. (15) are replaced in the multiplication of the inertia matrix and acceleration vector and the stiffness matrix and displacement vector as Eq. (16).

$$\begin{aligned}
 Eq(1) &= \frac{2m}{3}\ddot{q}_1 + \frac{m}{3}\ddot{q}_3 + \frac{34k}{25}q_1 - \frac{3k}{5}q_2 + \frac{6k}{25}q_3 \\
 Eq(2) &= 5m\ddot{q}_2 - \frac{3k}{5}q_1 + kq_2 - \frac{2k}{5}q_3 \\
 Eq(3) &= \frac{m}{3}\ddot{q}_1 + \frac{2m}{3}\ddot{q}_3 + \frac{6k}{25}q_1 - \frac{2k}{5}q_2 + \frac{29k}{25}q_3
 \end{aligned} \tag{15}$$

$$Eq = \begin{bmatrix} \frac{2m}{3} & 0 & \frac{m}{3} \\ 0 & 5m & 0 \\ \frac{m}{3} & 0 & \frac{2m}{3} \end{bmatrix} \begin{Bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \end{Bmatrix} + \begin{bmatrix} \frac{34k}{25} & -\frac{3k}{5} & \frac{6k}{25} \\ -\frac{3k}{5} & k & -\frac{2k}{5} \\ \frac{6k}{25} & -\frac{2k}{5} & \frac{29k}{25} \end{bmatrix} \begin{Bmatrix} q_1 \\ q_2 \\ q_3 \end{Bmatrix} \tag{16}$$



```
>> Eq(1)
ans = (sym)

      34*k*q1(t)      3*k*q2(t)      6*k*q3(t)      2*m*ddq1(t)      m*ddq3(t)
      -----      -----      + -----      + -----      + -----
      25              5              25              3              3

>> Eq(2)
ans = (sym)

      3*k*q1(t)      2*k*q3(t)
      -----      + k*q2(t) - -----      + 5*m*ddq2(t)
      5              5

>> Eq(3)
ans = (sym)

      6*k*q1(t)      2*k*q2(t)      29*k*q3(t)      m*ddq1(t)      2*m*ddq3(t)
      -----      -----      + -----      + -----      + -----
      25              5              25              3              3
```

Fig. 7. Equations of motion in the command window

<pre>function [MM]=mmass(Eq) S=Eq; for i=1:1:3; S(i)=subs(S(i),'ddq1(t)','ddq1'); S(i)=subs(S(i),'ddq2(t)','ddq2'); S(i)=subs(S(i),'ddq3(t)','ddq3');  S(i)=subs(S(i),'dq1(t)','dq1'); S(i)=subs(S(i),'dq2(t)','dq2'); S(i)=subs(S(i),'dq3(t)','dq3');  S(i)=subs(S(i),'cos(q1(t))','cos(displ)'); S(i)=subs(S(i),'cos(q2(t))','cos(displ2)'); S(i)=subs(S(i),'cos(q3(t))','cos(displ3)');  S(i)=subs(S(i),'sin(q1(t))','sin(displ)'); S(i)=subs(S(i),'sin(q2(t))','sin(displ2)'); S(i)=subs(S(i),'sin(q3(t))','sin(displ3)');  S(i)=subs(S(i),'cos(q1(t))','cos(displ)'); S(i)=subs(S(i),'cos(q2(t))','cos(displ2)'); S(i)=subs(S(i),'cos(q3(t))','cos(displ3)');  S(i)=subs(S(i),'cos(q1(t))^2','cos(displ)^2'); S(i)=subs(S(i),'cos(q2(t))^2','cos(displ2)^2'); S(i)=subs(S(i),'cos(q3(t))^2','cos(displ3)^2');  S(i)=subs(S(i),'sin(q1(t))^2','sin(displ)^2'); S(i)=subs(S(i),'sin(q2(t))^2','sin(displ2)^2'); S(i)=subs(S(i),'sin(q3(t))^2','sin(displ3)^2');  S(i)=subs(S(i),'sin(q1(t))','sin(displ)'); S(i)=subs(S(i),'sin(q2(t))','sin(displ2)'); S(i)=subs(S(i),'sin(q3(t))','sin(displ3)');  S(i)=subs(S(i),'q1(t)^2','displ^2'); S(i)=subs(S(i),'q2(t)^2','displ2^2'); S(i)=subs(S(i),'q3(t)^2','displ3^2');  S(i)=subs(S(i),'q1(t)','q1'); S(i)=subs(S(i),'q2(t)','q2'); S(i)=subs(S(i),'q3(t)','q3'); end</pre>	<pre>Constant=S; for i=1:1:3; Constant(i)=subs(Constant(i),'ddq1','0'); Constant(i)=subs(Constant(i),'ddq2','0'); Constant(i)=subs(Constant(i),'ddq3','0'); end S=S-Constant;  for i=1:1:3; S(i)=expand(S(i)); end  for i=1:1:3; mS(i)=S(i); end  for i=1:1:3; CConstant(i)=Constant(i); end  for j=1:1:3; mM=mS; clear UUU UUU=zeros(3,1); UUU(j,1)=1; aaa1=UUU(1,1); aaa2=UUU(2,1); aaa3=UUU(3,1);  for i=1:1:3; mM(i)=subs(mM(i),'ddq1',aaa1); mM(i)=subs(mM(i),'ddq2',aaa2); mM(i)=subs(mM(i),'ddq3',aaa3); MM(i,j)=mM(i); end end  function_handle(MM,'file','Mass')</pre>
---	---

Fig. 8. The script of mmass

### 3.4 Dynamic response

After executing the codes introduced in Sections 3.1 to 3.3, the function of the mass matrix, the function of the stiffness matrix, the function of the nonlinear force vector, and the gravity force vector are derived. These functions take design variables as input values and derive results in matrix and vector form as output values. Table 1 shows the design variables and parameters for numerical integration. By substituting the values into the numerical integration code shown in Fig. 1, the dynamic responses of each generalized coordinate are represented graphically, and a Fourier Transform of the responses is executed to derive the main peak frequency appearing. Through the process, the operating conditions and responses can be checked by changing the values of the operating conditions and design variables to avoid critical vibrations that can make the system unstable (Fig. 10).



```

function [KK,CConstant]=mstiff(Eq)
S=Eq;
for i=1:1:3;
S(i)=subs(S(i),'ddq1(t)','0');
S(i)=subs(S(i),'ddq2(t)','0');
S(i)=subs(S(i),'ddq3(t)','0');

S(i)=subs(S(i),'dq1(t)','dq1');
S(i)=subs(S(i),'dq2(t)','dq2');
S(i)=subs(S(i),'dq3(t)','dq3');

S(i)=subs(S(i),'cos(q1(t))','cos(displ)');
S(i)=subs(S(i),'cos(q2(t))','cos(displ2)');
S(i)=subs(S(i),'cos(q3(t))','cos(displ3)');

S(i)=subs(S(i),'sin(q1(t))','sin(displ)');
S(i)=subs(S(i),'sin(q2(t))','sin(displ2)');
S(i)=subs(S(i),'sin(q3(t))','sin(displ3)');

S(i)=subs(S(i),'cos(q1(t))','cos(displ)');
S(i)=subs(S(i),'cos(q2(t))','cos(displ2)');
S(i)=subs(S(i),'cos(q3(t))','cos(displ3)');

S(i)=subs(S(i),'cos(q1(t))^2','cos(displ)^2');
S(i)=subs(S(i),'cos(q2(t))^2','cos(displ2)^2');
S(i)=subs(S(i),'cos(q3(t))^2','cos(displ3)^2');

S(i)=subs(S(i),'sin(q1(t))^2','sin(displ)^2');
S(i)=subs(S(i),'sin(q2(t))^2','sin(displ2)^2');
S(i)=subs(S(i),'sin(q3(t))^2','sin(displ3)^2');

S(i)=subs(S(i),'sin(q1(t))','sin(displ)');
S(i)=subs(S(i),'sin(q2(t))','sin(displ2)');
S(i)=subs(S(i),'sin(q3(t))','sin(displ3)');

S(i)=subs(S(i),'q1(t)^2','displ^2');
S(i)=subs(S(i),'q2(t)^2','displ2^2');
S(i)=subs(S(i),'q3(t)^2','displ3^2');

S(i)=subs(S(i),'q1(t)','q1');
S(i)=subs(S(i),'q2(t)','q2');
S(i)=subs(S(i),'q3(t)','q3');
end

for i=1:1:3;
S(i)=expand(S(i));
end

Constant=S;
for i=1:1:3;
S(i)=subs(S(i),'pi','0');
REST_GEN(i)=S(i);
REST_GEN(i)=subs(REST_GEN(i),'q1','0');
REST_GEN(i)=subs(REST_GEN(i),'q2','0');
REST_GEN(i)=subs(REST_GEN(i),'q3','0');
end

S=S-REST_GEN;
Constant=Constant-S;

for i=1:1:3;
kS(i)=S(i);
end

for i=1:1:3;
CConstant(i)=Constant(i);
end

for j=1:1:3;
mK=kS;
clear UUUU
UUU=zeros(3,1);
UUU(j,1)=1;
aaa1=UUU(1,1);
aaa2=UUU(2,1);
aaa3=UUU(3,1);
for i=1:1:3;
mK(i)=subs(mK(i),'q1',aaa1);
mK(i)=subs(mK(i),'q2',aaa2);
mK(i)=subs(mK(i),'q3',aaa3);
KK(i,j)=mK(i);
end
end

function_handle(KK,'file','Stiffness')
function_handle(CConstant,'file','Generalized1')

```

Fig. 9. The script of mstiff

## 4. Code application

### 4.1 System definition

In this section, we reviewed whether the GNU code can be used to derive equations of motion and to calculate dynamic responses easily when an arbitrary system is given. Fig. 11 is a drive train system that has a mechanism in which each  $G_i$  rotor rotates due to torque applied to the  $G_1$  rotor, and the  $G_6$  rotor rotates through the power transmission. Each  $G_i$  rotor has the number of  $n_i$  teeth as a gear, and the adjacent gears, except the gear engagement, are connected by the shaft with torsional stiffness.

To apply the definition of the system to the code for calculating the response from the system, it is necessary to set the generalized coordinates by deriving the number of degrees of freedom and to define the kinetic energy and potential energy. Even though the system has six rotors, the number of the generalized coordinates is only four, because  $G_3$  and  $G_5$  rotors have the geometrical relationship of gear engagement with  $G_2$  and  $G_4$  respectively. In other words, the generalized coordinates of  $G_3$  and  $G_5$  can be expressed by the generalized coordinates of the  $G_2$  and  $G_4$ . Therefore, the two coordinates are reduced. Additionally, the rotational direction of  $G_3$  and  $G_4$  rotors rotate in the opposite direction to the direction indicated in Fig. 11. The kinetic energy and potential energy considering the above facts are shown in Eqs. (17)-(18). To proceed with the further process to derive the equations of motion, some corrections are required in all the codes mentioned in Part 3 below.

Table 1. Design variable and simulation condition

Design variable	Value	Simulation condition	Value
$m$	10 kg	$t$	0 ~ 10 sec
$l$	10 m	$\Delta t$	0.001 sec
$k$	10,000 N/m	$b$	1/4
$g$	9.81 m/s <sup>2</sup>	$\gamma$	0.5





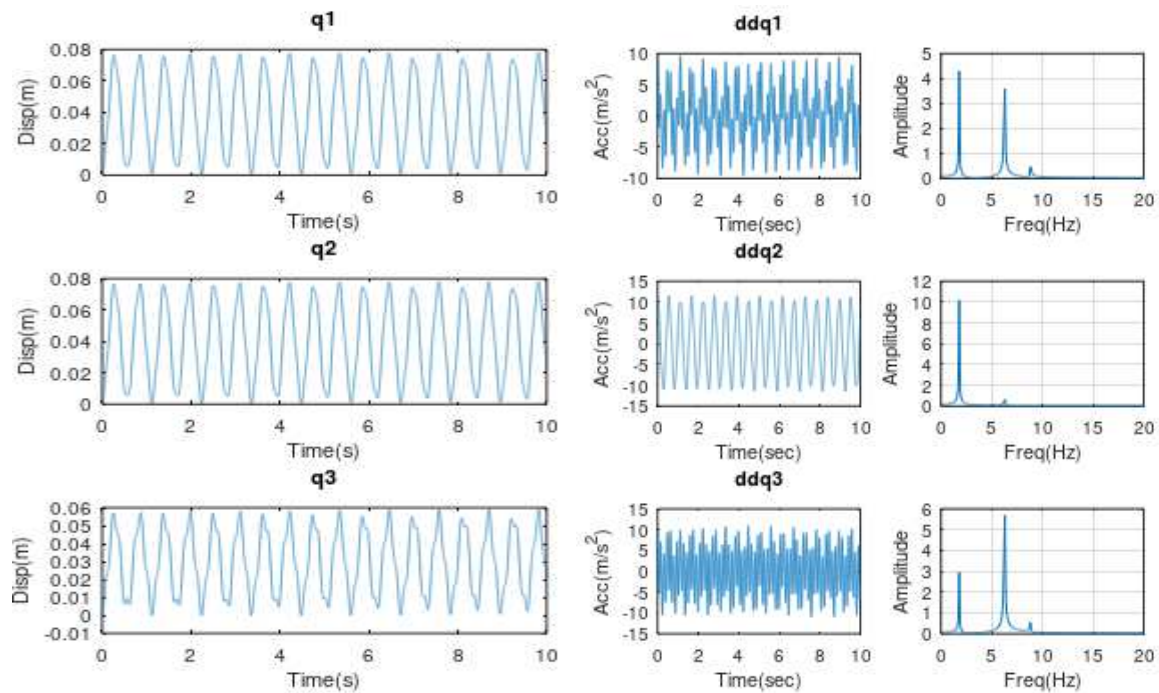


Fig. 10. Displacement in the time domain and acceleration in the time and frequency domain

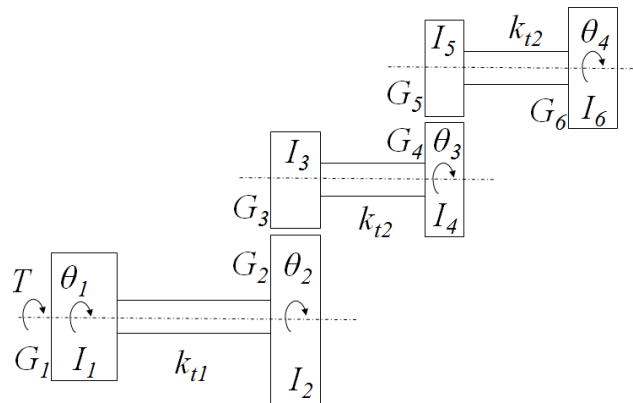


Fig. 11. Drive-train system

- Update the symbolic expression of the generalized coordinates and design variables
- As the number of the generalized coordinates changes from 3 to 4, the number of 'subs' process, the number of iteration of 'for' loop, and the number of 'diff' derivative processing should be increased.
- Update the kinetic energy and potential energy.
- Apply the number of process 4 instead of 3 in other processing.

$$T = \frac{1}{2} I_1 \dot{\theta}_1^2 + \frac{1}{2} \left( I_2 + I_3 \frac{n_2^2}{n_3^2} \right) \dot{\theta}_2^2 + \frac{1}{2} \left( I_4 + I_5 \frac{n_4^2}{n_5^2} \right) \dot{\theta}_3^2 + \frac{1}{2} I_6 \dot{\theta}_4^2 \quad (17)$$

$$V = \frac{1}{2} k_{t1} (\theta_2 - \theta_1)^2 + \frac{1}{2} k_{t2} \left( \theta_3 - \left( -\frac{n_2}{n_3} \theta_2 \right) \right)^2 + \frac{1}{2} k_{t3} \left( \theta_4 - \left( -\frac{n_3}{n_5} \theta_3 \right) \right)^2 \quad (18)$$

#### 4.2 Response

The results obtained from the derivation of the equations of motion and matrix transformation and the dynamic responses of substituting arbitrary design variables and operating conditions shown in Table 2 into the numerical integration are shown in Fig. 12. The angular displacement of each rotor may be increased or decreased constantly in one rotational direction by a constant torque applied to  $I_1$ , but if you check the angular velocity and acceleration responses, each rotor has the vibrational displacement due to the torsional stiffness of the shaft that makes the reaction torque when the angular displacement of the relative rotor is different. The peak frequencies shown in the Fast Fourier Transform result are natural characteristics of the system caused by the coupled motion among rotors. When the angular velocity of the rotor has the same value corresponding to the natural frequency, the vibrations are amplified, so the operating condition or the design variable should be changed to avoid this phenomenon in the design stage.



Table 2. Design variable and simulation condition

Design variable	Value	Simulation condition	Value
$I_1$	10 kg·m <sup>2</sup>	t	0 ~ 10 sec
$I_2$	20 kg·m <sup>2</sup>		
$I_3$	30 kg·m <sup>2</sup>		
$I_4$	40 kg·m <sup>2</sup>	$\Delta t$	0.001 sec
$I_5$	50 kg·m <sup>2</sup>	T (Torque on $I_i$ )	10 N·m
$I_6$	60 kg·m <sup>2</sup>		
$n_2 \sim n_5$	10/30/15/20	b	1/4
$k_{t1}$	10 <sup>9</sup> N·m/rad		
$k_{t2}$	10 <sup>9</sup> N·m/rad	$\gamma$	0.5
$k_{t3}$	10 <sup>9</sup> N·m/rad		

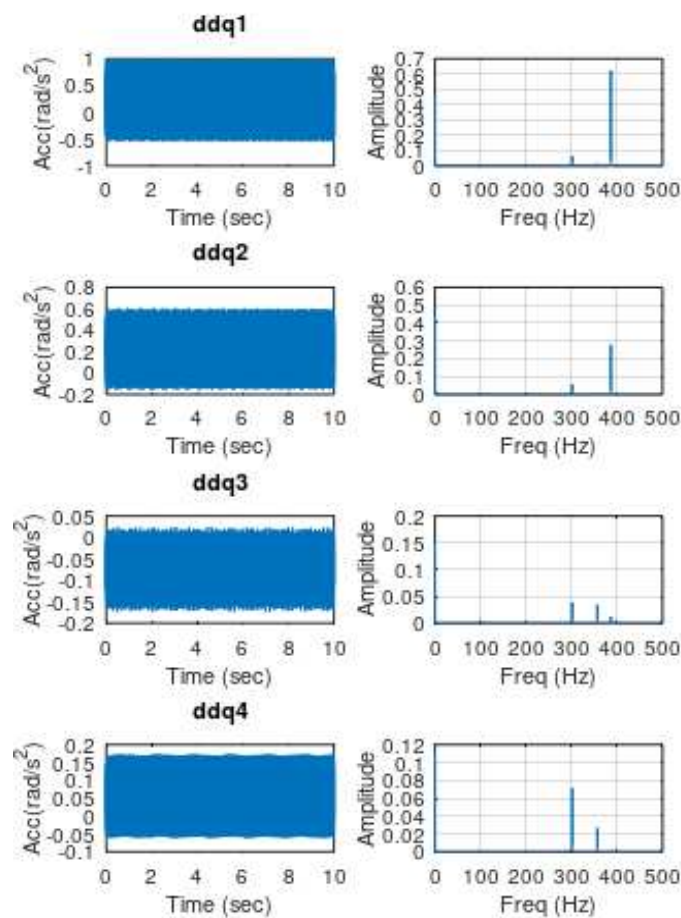


Fig. 12. Drive-train system

5. Conclusion

In this paper, we developed the automatic GNU Octave codes that can derive the equations of motion based on Lagrange dynamics and can apply the equations of motion to the Newmark numerical integration method to calculate the responses. By showing the structure of the code and the application, users that are not familiar with multi-body dynamics can easily calculate the dynamic characteristics of the system. To validate the code whether the dynamic characteristics of the system is correct or not, some arbitrary mechanical systems were analyzed by deriving the equations of motion, by calculating the dynamic characteristics with the numerical integration, and by comparing them to the analytical solution. Through the processes, the fact that the user can easily deal with the matrix transformation problem by defining the system definition such as the kinetic-potential energies and system variables and by classifying them based on the rule was confirmed. In the future, we plan to expand the usability of the code in the industry and to be possible to use it as educational materials in the course and lecture by upgrading the application of the code such as an open-chain system.



## Acknowledgments

This work was supported by Yuhan University.

## Conflict of Interest

The author declared no potential conflicts of interest with respect to the research, authorship and publication of this article.


## Funding

The author received no financial support for the research, authorship and publication of this article.

## References

- [1] Yoo, W. S., Park, S. J., Dmitrochenko, O., Pogorelov, D., Verification of Absolute Nodal Coordinate Formulation in Flexible Multibody Dynamics via Physical Experiments of Large Deformation Problems, *Journal of Computational and Nonlinear Dynamics*, 1(1), 2006, 81-93.
- [2] Yoo, W. S., Kim, T. Y., Jung, S., History and Future of Multi-body Dynamics, *Transactions of the Korean Society of Mechanical Engineers - A*, 43(7), 2019, 483-491.
- [3] Munro, N., *Symbolic methods in control system analysis and design*, Institution of Engineering and Technology, Stevenage, 1999.
- [4] Van Khang, N., Kronecker product and a new matrix form of Lagrangian equations with multipliers for constrained multi-body systems, *Mechanics Research Communications*, 38(4), 2011, 294-299.
- [5] Negrut, D., Dyer, A., *ADAMS/Solver Primer*, MSC Software Documentation, Ann Arbor, 2004.
- [6] Bonev, I. A., *Geometric analysis of parallel mechanisms*, Université Laval, Quebec City, 2002.
- [7] vander Linde, R. Q., Schwab, A. L., *Lecture Notes Multi-body Dynamics B*, Delft University, Delft, 1998.
- [8] Sadati, S. M. H., Naghibi, S. E., Naraghi, M., An automatic algorithm to derive linear vector form of Lagrangian equation of motion with collision and constraint, *Procedia Computer Science*, 76, 2015, 217-222.
- [9] Sadati, S. M. H., Naghibi, S. E., Shiva, A., Zschaler, S., Hauser, H., Walker, I., Althoefer, K., Nanayakkara, T., AutoTMTDyn: A Matlab software package to drive TMT Lagrange dynamics of series rigid-and continuum-link mechanisms, *IROS 2018 Workshop on Soft Robotic Modeling and Control: Bringing Together Articulated Soft Robots and Soft-Bodied Robots*, Madrid, Spain, 2018.
- [10] Newmark, N. M., A method of computation for structural dynamics, *Journal of the Engineering Mechanics Division*, 85, 1959, 67-94.
- [11] Wong, C., *Online Referencing*, <https://www.mathworks.com/matlabcentral/fileexchange/71007-newmark-beta-method-for-non-linear-single-dof-structures> (2019, accessed 9 March 2020).
- [12] Kane, R. T., Levinson, D. A., *Dynamics: Theory and Application*, McGraw Hill College, New York, 1985.
- [13] Udwadia, F. E., Wanichanon, T., *Nonlinear Approaches in Engineering Applications 2: A New Approach to the Tracking Control of Uncertain Nonlinear Multi-Body Mechanical Systems*, Springer, New York, 2006.
- [14] Schutte, A., Udwadia, New approach to the modeling of complex multibody dynamical systems. *Journal of Applied Mechanics*, 78(2), 2011, 1-11.
- [15] Udwadia, F. E., Wanichanon, T., Control of uncertain nonlinear multibody mechanical systems, *Journal of Applied Mechanics*, 81(4), 2014, 1-11.
- [16] Singiresu, S. R., *Mechanical vibrations*, Addison Wesley, Boston, 1995.

## ORCID iD

Yonghui Park  <https://orcid.org/0000-0001-6716-6935>



© 2021 Shahid Chamran University of Ahvaz, Ahvaz, Iran. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0 license) (<http://creativecommons.org/licenses/by-nc/4.0/>).

**How to cite this article:** Park Y. An Automatic Program of Generation of Equation of Motion and Dynamic Analysis for Multi-body Mechanical System using GNU Octave, *J. Appl. Comput. Mech.*, 7(3), 2021, 1687-1697. <https://doi.org/10.22055/JACM.2020.33826.2293>

**Publisher's Note** Shahid Chamran University of Ahvaz remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

