LagranTexPac: A Software Tool to Obtain the Dynamic Equations of Mechanical Systems

Fernando Henrique Gomes Zucatelli, Magno Enrique Mendoza Meza, and Andr é Fenili

***Abstract*—Modeling of a dynamical system is an important research area to obtain mathematical models of real systems. There are several approaches to get mathematical models, one of them is utilizing the Lagrangian of holonomic systems. This paper introduces LagranTexPac, a package Matlab-based framework aimed at obtaining, and simulate the equations of motion of a mechanical system. LagranTexPac uses the Lagrangian function of a system to automatically generate the system equations of motion for numerically simulate and saves all phase portraits. Motion constraints can be included in Lagrangian, and a summary report can be generated. Utility of this software tool includes any engineer area, in which is used a Lagrangian function to obtain the equations of motion of a dynamic system, e.g., aerospace, biomedical, robotics and mechanical engineering. Three application examples are included to illustrate the usefulness of this software tool.**

***Index Terms*—Dynamical systems, contact dynamics, matlab®, code generation, LATEX report generation, simulation tool.**

I. Introduction

Dynamical systems are studied in several areas of engineering, in which system modeling is necessary to study and analyze the system behaviors. Newton's laws can be used to model dynamical systems and the development procedure may be complex. An alternative approach used to model systems is the Lagrangian function, wherewith the Lagrangian equations are obtained. Last approach can be less complex than the first one

A few tool packages are available to obtain the Lagrangian equations in Matlab framework. Some years ago, there was no software package to help in the procedure to obtain the Lagrangian equations of dynamical systems. To derive equations of the dynamical system is a fatigued procedure.

In [1], [2] were shown Matlab® toolboxes that resolve Euler- Lagrange equation and provide ODE equations for numerically simulation of a system. Those toolboxes do not interact with LaTeX. In [3] was reported a package to generate some LaTeX features to use with Matlab®, in [4] was viewed a similar technique used to create a simple LaTeX files directly from Matlab® and in [5] was used a package to format the Matlab® code syntax inside a LaTeX document.

Lagrangian is a mathematical function, which is a function of the generalized coordinates. These coordinates and their time derivatives contain information about the dynamics of the system. The dynamical model of a generic system can be described as follows:

$$\dot{\mathbf{y}} = f(\mathbf{y}) \tag{1}$$

where $y \in R^n$ is a vector state and $f: U \subseteq R^n \rightarrow R^n$ is a vector field and describes an autonomous system [6]-[8].

In [9]-[11] were introduced different approaches to determine differential equations of the system dynamics. Examples about robot manipulators can be found in [10], [12], in [7], [8] were shown examples about nonlinear dynamical systems, in [13], [14] were illustrated examples about triple-pendulum arm and constrain to the movement for a mechanical system, respectively.

In the triple-pendulum context, [13], [15] show the fatigued work to obtain differential equations of dynamic system, and can be seen some development mistakes like the forgotten term $m_2 z_2^2$, in [13] at the mass matrix. This was one of motivations to develop the this tool to avoid writing mistakes or forgetting mistakes, and another motivation was to provide a file with the differential equations to simulate in Matlab® framework.

An important feature of the tool is to implement equations of the system dynamics taking into account constraints to the motion, which is part ot the contacts dynamics and this topic was not considered in [1], [2]. Studied system in [14] was considered as an application of this feature of tool. Another feature of tool is to produce a summarized report of all development steps to obtain the differential equations of the system dynamics, and the report is generated to be used in a LaTeX framework. The command used to write a line in a LaTeX file is reported in [4], in this tool it is created a function to write the whole report selecting the language between English, German, Spanish or Potuguese.

Therefore, the aim of Matlab® tool is to make possible a fast analysis and simulation of dynamic equations with or without constraints to the motion, avoiding written mistake and forgotten mistakes and to make possible checking intermediate steps.

This article is organized as follows: in Section II is shown a summary of the Euler-Lagrange's equation of motion; in Section III is briefly designed the development of the tool; in Section IV is shown three application examples and finally in Section V is shown conclusions.

II. Euler-Lagrange's Equations of Motion.

Dynamical systems can be described by a set of simultaneous differential equations known as Lagrange's equations according to [6], [9] and the system dynamics can be defined

Manuscript received August 7, 2016; revised December 7, 2016. This work was supported in part by the agency CAPES.

F. H. G. Zucatelli, M. E. M. Meza and A. Fenili are with the Graduate Program in Mechanical Engineering, Federal University of ABC, Santo André, SP09210-580, BRAZIL (e-mail: fernando.zucatelli@aluno.ufabc.edu.br, magno.meza@ufabc.edu.br, andre.fenili@ufabc.edu.br)

DOI: 10.7763/IJCTE.2017.V9.1145

as:

$$\frac{d}{dt}\left(\frac{\partial L(q_i,\dot{q}_i)}{\partial \dot{q}_i}\right)-\frac{\partial L(q_i,\dot{q}_i)}{\partial q_i}+\frac{\partial R(q_i,\dot{q}_i)}{\partial \dot{q}_i}=\tau_i\,;\forall i \quad (2)$$

where i is the index for all generalized coordinates q_i and τ_i are generalized forces; R is the Rayleigh dissipation function, and the Lagrangian function is defined as follows:

$$L=T-V \quad (3)$$

where T is the kinetic energy, and V is the potential for the free system, as well as for the constrained system [14]:

$$L=T-V+\Lambda \quad (4)$$

with

$$\Lambda=\sum_{i=1}^{P}\lambda_i\,\phi_i \quad (5)$$

for the system with constraints, λ_i is the Lagrange multiplicators and φ_i are the P constraint equations of the motion. In this context, when there are constraints motion the tool package analyze the free, the constraints and contact dynamics separately. The energy equations for a generic dynamical system are defined as follows:

$$\begin{aligned}&2T=T_1+T_2\\&T_1=2\text{ times rotation kinetic energy}\\&T_2=2\text{ times translation kinetic energy}\\&V=\text{All kinds of potential energy}\\&R=\text{All kinds of dissipation}\end{aligned} \quad (6)$$

To avoid the division by two at kinetics energies was chosen this form of these energies; and did not have division on potential energies, and the gravitational term have to be included.

Applying Euler-Lagrange equation from Eq. (2) at Lagrangian function L from Eq. (3) results in a set of n second order differential equations [9], which can be rewritten in the matrix notation form [9], [10] and are given by:

$$M\cdot q+\underbrace{B\left(\underline{q}\;\underline{q}\right)q}_{\underline{C}}+G\left(q\right)=\underline{\tau} \quad (7)$$

where $M \in R^{n\times n}$; $\underline{C}, \underline{G}, \underline{\tau} \in R^{n\times 1}$.

III. Structure and Flow Diagram of Toolbox

Desired terms of kinetic energy must be entered as $T(l),\forall l=\{1,2\}$, *e.g.*, $2T=m_1\dot{q}_1^2$ is needed to enter "T(1) = m_1 * dq(1)^2", with the variable, *num_coord_gen_qq*, which previously define for the correct dimension of *q(1)* and *len_T* for the correct dimension for *T (l)*. Two main functions are performed by the package, which are:

1) Develops the dynamical system, the left side of Eq. (2).

2) Performs the numerical integrator, ode45 from Matlab®.

Fig. 1 shows the flow diagram of the main program called Base, which run all other functions. The Base is composed by severals sections, in the 'Preparations" section are defined parameters. In *def_coord* were created the co- ordinate variables as were defined at *Base*, and *def_coord* may not be edited. To create coordinates variables was used the variables *num_coord_gen_qq* to define dimesions of $\underline{q}\in R^{num\_coord\_gen\_qq}$ - and *num_coord_gen_pp* - $\underline{p}\in R^{num\_coord\_gen\_pp}$. In the script Energy are written the Lagrangian equations for energies 2T, V and R, separately for a better presentation of results. The script *Barriers* is used for the constraint equations. In the variable "*strings_latex*" are stored all the strings for exhibited text. The parameter *exec* is used to split the flow to generate equations for numerical integration or to initiate the simulation. First of all, the equations must be generated and so *exec* = 0. After that, the Lagrangian equations are applied and an expression for the generalized forces is generated, and finally the result is separated in *M* matrix, $\underline{C}$ and $\underline{G}$ vectors which are described in Eq. (7).

In the case of a system with constraint, parameter *restrito* = 1, the same procedure is done considering the constraints. The variable "*eq_gen*" is the core for the generation of the equations of the system dynamics. In this script, the equations are written as described in Eq. (1). Equations for the integrator are shown in the command window and stored on a *.mat* file and the "`function dy = z_Dinamica_ode_dy_sel_000 (t,y)`" is created, where "000" has the variable *sel* value. This is exactly the name of the function called by the integrator at *eq_int* script.

Fig. 2 shows the definition of *str_latex*, which is useful and necessary to change some notations from Matlab® to LaTeX e.g., "d2pp" to "\ddot p". Thus, the correct LaTeX notation is applied and the report is easier to read. To do this procedure is used the *mega_loc_sub* script. The program inserts a pre-defined header and footer as comments, then it searches specific terms in *mega_loc_sub* based on an alphabet defined at *alfabeto*.

The *size_alfabeto* must be correctly set and every term to be searched must be named as *alfabeto_in_000*, changing "000" sequentially and the new value must be *alfabeto_out_000*, always with 3 digits. Finally the *save_file_tex* function receives the. *tex* file, which name was constructed with *nome_arquivo* as a base and adding some of the control variable to it, this file will be created at the path specified at *caminho* (path).

On the other side, the dynamical system is copied into *dy*, which is accessed by *odeXY* inside the *eq_int* script and this script is accessed from the Base when *exec* = 1 is defined. If the constrains have been set, the user may choose to integrate step by step with *int_passos* = 1, so the *eq_int* keeps verifying the collision.

The change at the free system behavior is updated as the user described at *restri*, this script must contain the first update of λ, part of this update is also given by *eq_gen*, and should be complemented by the user with the equations of conservation of momentum (linear and/or angular) at the collision instant using the correct notation outside the *odeXY* script.

Finished integration, phase portrait graphs are printed and data is stored, which allows the user redo or create specific

graphs, all figures are also stored and at the end the LaTeX code for the stored figures are exhibited at the command window.

This code is also saved in a *.tex* which may be directly inserted at the report *.tex* file at the correct folder.

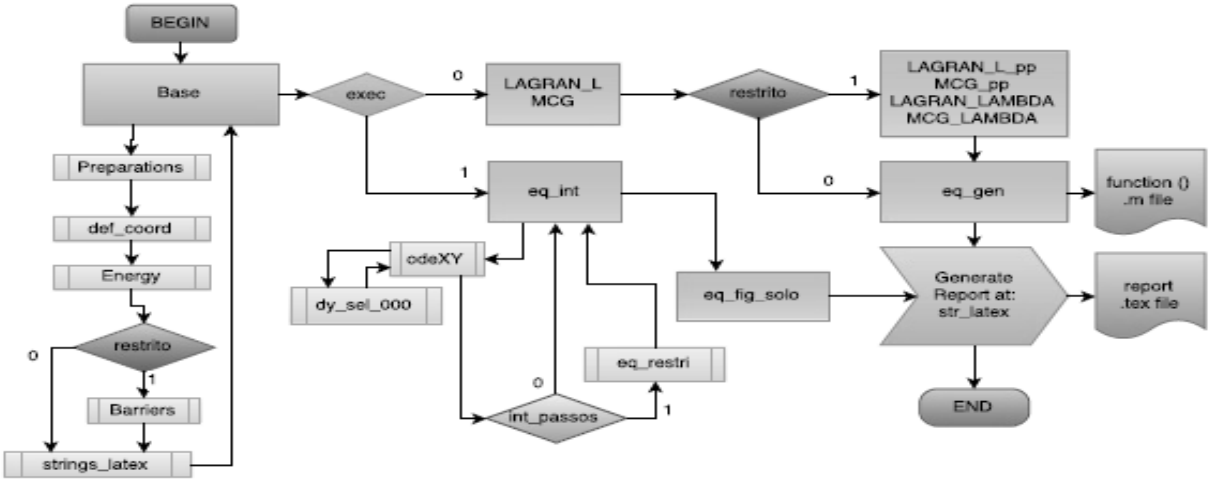

Fig. 1. Program flow diagram of the tool.

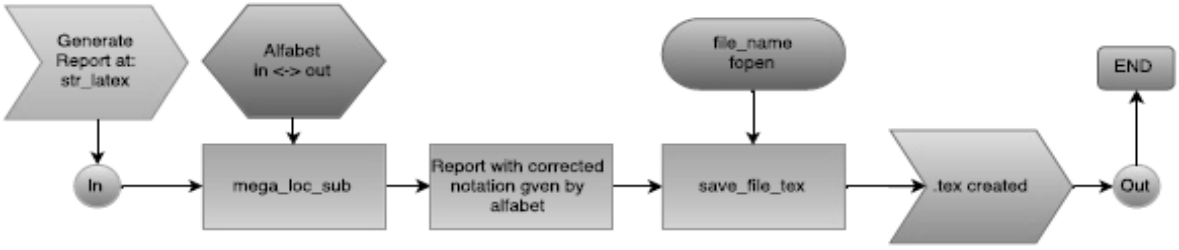

Fig. 2. Program data flow - .tex file creation with correct notation.

Although all automatic development and creation given by the program, the correct description of the constraints and how they interact with the free system, relationships and impact dynamics are not a concern of this package, and it should be correctly described by the project designer. A tutorial file describing the program and how to handle the generated code and report has also been developed.

IV. Application Examples

In this section are shown three application examples, which are based on references [12]-[14]. Only the line before the figure and figure itself have been written by us. All development and equations have been directly inserted by our package as it is shown (therefore some equations are a little bit misaligned) writing only the "*\input{file}*" command.

A. Example in Aerospace: Constrained Mass-Spring System

The constrained mass-spring system is shown at Fig. 3, it is detailed at [14] while investigation contacts dynamics of bodies. There is the object of mass m_1 and the wall m_w, they are separated for the distance *d* while at the equilibrium points. We reproduce the same result from [14] with the interaction force λ from Lagrangian multipliers.

Fig. 4, Fig. 5 and Fig. 6 were created and inserted at the LaTeX file automatically, LagranTexPac generates all combinations of coordinates of each system, coordinates q for free system and p for wall system. To avoid numerical errors, it is defined ε_1. If the distance between the mass and the wall is smaller than ε_1, then the system is defined "constrained" and changes to the constrained equations. It returns if the distance becomes greater than $\varepsilon_1$or if the constrained force λ is smaller than $\varepsilon_{\lambda 1}$.

Fig. 4 shows the time dependent variables x_1 to x_4 , system oscillates until position x_1 hits the wall and it starts to move as can be seen by the x_3 and x_4 appearing out of zero, x_3 starts from zero of its own coordinates system and x_4 starts with non-zero velocity after the collision momentum transfer. "*Free System*" line has value 1 when system is "free", no contact, and 0 when it does. The impact instant is also shown as a thin line just after "Free System" line reaches 0. While system is constrained, there is the contact force λ_1 , this force begins at its highest value and decrease to 0 when the system is free once again. Fig. 5 and Fig. 6 are the phase portraits $x_1 \times x_2$ and $x_3 \times x_4$ respectively, the impact instant is identified by the abrupt chance in both phase portraits.

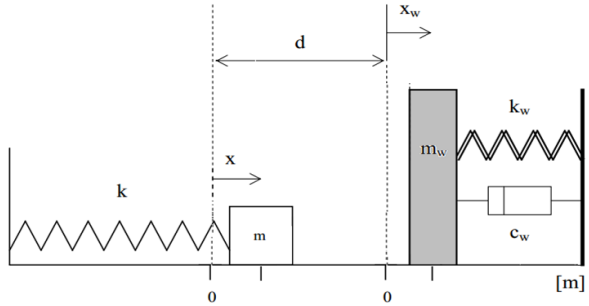

Fig. 3. Constrained mass-spring system, with q1 = x1 and q1 = xw . source: [14].

System energies

Kinetics:

$$2?T = \dot{q}_1^{\,2} m_1 + \quad (8)$$

Potential:

$$V = \frac{k_1 q_1^{\,2}}{2} \quad (9)$$

Rayleigh dissipation:

$$\mathrm{R} = 0 \quad (10)$$

Barriers system energies

Kinetics:

$$2?T_{pp} = \dot{p}_1^{\,2} m_w + \quad (11)$$

Potential:

$$V = \frac{k_w p_1^{\,2}}{2} \quad (12)$$

Rayleigh dissipation:

$$R = \frac{c_w \dot{p}_1^{\,2}}{2} \quad (13)$$

Free system Euler-Lagrange

Euler-Lagrange term to T:

$$2tauT(1) = \left(2m_1\right)\ddot{q}_1 \quad (14)$$

Euler-Lagrange term to V:

$$tauV(1) = \left(-k_1\right) q_1 \quad (15)$$

Euler-Lagrange term to R:

$$\mathrm{tauR}(1) = 0 \quad (16)$$

Results for with kinetics, potential and dissipation energies:

$$tau(1) = m_1 \ddot{q}_1 + k_1 q_1 \quad (17)$$

Free system matrices

Matrix *M* for the system:

$$M(1,1) \;=\; m_1 \quad (18)$$

Vector *C* for the system:

$$\mathrm{C}(1) = \mathrm{k}_1 \cdot q_1 \quad (19)$$

Vector *G(q)* for the system:

$$G(1) = (-g) \quad (20)$$

Barriers system Euler-Lagrange

Euler-Lagrange term to T:

$$2tauT_{pp}(1) = \left(2m_w\right)\ddot{p}_1 \quad (21)$$

Euler-Lagrange term to V:

$$tauV_{pp}(1) = -k_w p_1 \quad (22)$$

Euler-Lagrange term to R:

$$tauR_{pp}(1) = c_w \dot{p}_1 \quad (23)$$

Results for τ with barriers kinetics, potential and dissipation energies:

$$tau_{pp}(1) = m_w \ddot{p}_1 + c_w \dot{p}_1 + k_w p_1 \quad (24)$$

Barriers system matrices

Matrix *M* for the barrier system:

$$M(1,1) = m_w \quad (25)$$

Vector *C* for the barrier system:

$$C(1) = c_w \dot{p}_1 + k_w p_1 \quad (26)$$

Vector *G(q)* for the barrier system:

$$G(1) = (-g) \quad (27)$$

Additional Euler-Lagrange with constraints Λ

Development to τ_λ:

$$tau\lambda(1) = q_1 - p_1 - dist1 \quad (28)$$

Development to τ_Λ :

$$tau\Lambda(1) = \lambda_1 \quad (29)$$

Development to τ_{pp_Λ}:

$$tau_{pp}\Lambda(1) = -\lambda_1 \quad (30)$$

Matrices with constraints λ

To generalized coordinates q:

Matrix *M* for the system:

$$M(1,1) = m_1 \quad (31)$$

Vector *C* for the system:

$$C(1) = k_1 q_1 + \lambda_1 \quad (32)$$

Vector *G(q)* for the system:

$$G(1) = (-g) \quad (33)$$

To barriers coordinates p:

Matrix *M* for the barrier system:

$$M(1,1) = m_w \quad (34)$$

Vector *C* for the barrier system:

$$C(1) = c_w \dot{p}_1 + k_w p_1 - \lambda_1 \quad (35)$$

Vector *G(q)* for the barrier system:

$$G(1) = (-g) \quad (36)$$

Equations to λ:

Definition of p and derivatives as functions of q and derivatives:

$$\begin{aligned} p_1 &= q_1 - dist1 \\ \dot{p}_1 &= \dot{q}_1 \\ \ddot{p}_1 &= \ddot{q}_1 \end{aligned} \tag{37}$$

Isolating λ_i:

$$\begin{aligned} \lambda_1 &= m_w\ddot{p}_1 + c_w\dot{p}_1 + k_w p_1 \\ \lambda_1 &= m_w\ddot{p}_1 + c_w\dot{p}_1 - k_w\left(dist1 - q_1\right) \\ \lambda_1 &= m_w\ddot{p}_1 + \left(c_w\dot{q}_1 - k_w\left(dist1 - q_1\right)\right) \\ \lambda_1 &= m_w\ddot{q}_1 + c_w\dot{q}_1 + k_w q_1 - dist1k_w \end{aligned} \tag{38}$$

Numerical simulation

Program with fixed integration step ode45.

Selected options: coef_restitui = 0, restrito = 1, int_passos = 1, tspan(end) = 3.500, dt = 0.010.

Parameters of this batch: $m_1 = 10.00$; $k_1 = 7.00$; $m_w = 5.00$; $c_w = 20.29$; $k_w = 7.00$; dist1 = 0.005 ; $\varepsilon_1 = 0.0001$; $\varepsilon_{\lambda 1} = 0.0001$.

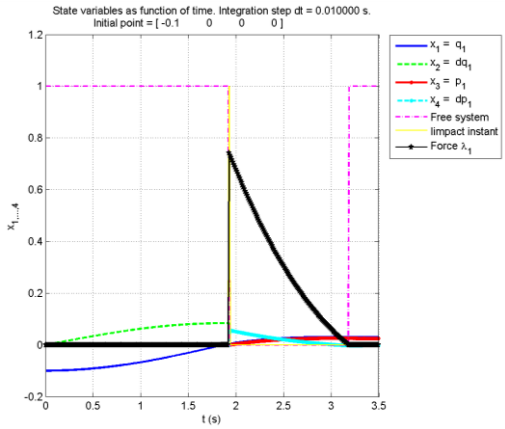

Fig. 4. Coordinates q and derivatives q.

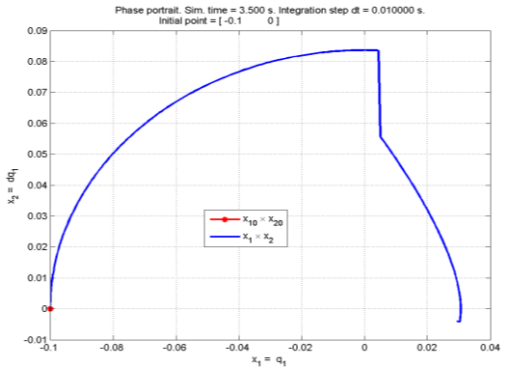

Fig. 5. Phase portrait x1 ×x2.

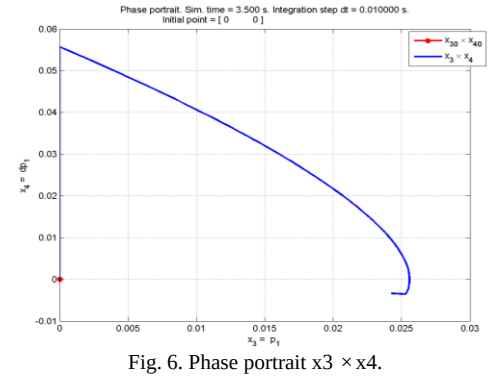

Fig. 6. Phase portrait x3 ×x4.

B. Example in Biomedical: Triple Pendulum Arm

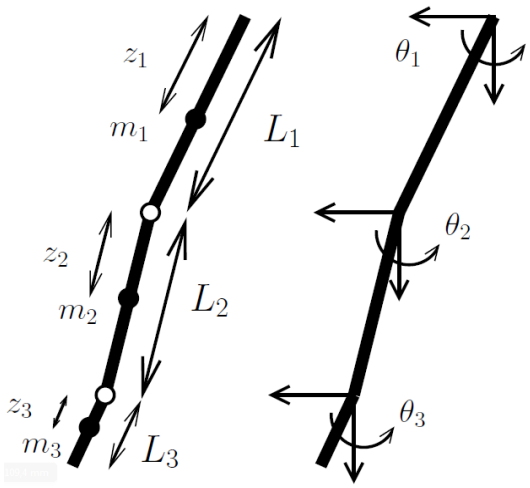

Fig. 7. Triple pendulum arm, with qi = θi , ∀ i = 1, 2, 3. source: [13].

Fig. 7 shows the triple pendulum arm system. In [13] treated about concepts in physiology and mechanics applied to rehabilitation studies. The system studied in [13] is an excellent application example how the extended equations may become complex and how the proposed package handles this problem writing the LaTeX summarized report as clear as possible directly from Matlab® and this package tries to avoid writing mistakes and forgetting mistakes.

System energies

Kinetics:

$$\begin{aligned} 2T &= L_1^2\dot{q}_1^2 m_2 + L_1^2\dot{q}_1^2 m_3 + L_2^2\dot{q}_2^2 m_3 \\ &+\dot{q}_1^2 m_1 z_1^2 + \dot{q}_2^2 m_2 z_2^2 + \dot{q}_3^2 m_3 z_3^2 \\ &+2L_1L_2\dot{q}_1\dot{q}_2 m_3\left(\cos(q_1)\cos(q_2)+\sin(q_1)\sin(q_2)\right) \\ &+2L_1\dot{q}_1\dot{q}_2 m_2 z_2\left(\cos(q_1)\cos(q_2)+\sin(q_1)\sin(q_2)\right) \\ &+2L_1\dot{q}_1\dot{q}_3 m_3 z_3\left(\cos(q_1)\cos(q_3)+\sin(q_1)\sin(q_3)\right) \\ &+2L_2\dot{q}_2\dot{q}_3 m_3 z_3\left(\cos(q_2)\cos(q_3)+\sin(q_2)\sin(q_3)\right) \end{aligned} \tag{39}$$

Potential:

$$V = g\begin{pmatrix} m_1 z_1\cos(q_1) + m_2 z_2\cos(q_2) + m_3 z_3\cos(q_3) + \\ L_1 m_2\cos(q_1) + L_1 m_3\cos(q_1) + L_2 m_3\cos(q_2) \end{pmatrix} \tag{40}$$

Rayleigh dissipation:

$$R = 0 \tag{41}$$

Free system Euler-Lagrange

Euler-Lagrange term to T:

$$\begin{aligned} 2tauT(1) &= \left(2L_1^2 m_2 + 2L_1^2 m_3 + 2m_1 z_1^2\right)\ddot{q}_1 \\ &+\begin{pmatrix} 2L_1 m_2 z_2\left(\cos(q_1)\cos(q_2)+\sin(q_1)\sin(q_2)\right) \\ +2L_1L_2 m_3\left(\cos(q_1)\cos(q_2)+\sin(q_1)\sin(q_2)\right) \end{pmatrix}\ddot{q}_2 + \\ &\begin{pmatrix} -2L_1 m_2 z_2\left(\cos(q_1)\sin(q_2)-\cos(q_2)\sin(q_1)\right) \\ -2L_1L_2 m_3\left(\cos(q_1)\sin(q_2)-\cos(q_2)\sin(q_1)\right) \end{pmatrix}\dot{q}_2^2 \\ &+\left(2L_1 m_3 z_3\left(\cos(q_1)\cos(q_3)+\sin(q_1)\sin(q_3)\right)\right)\ddot{q}_3 + \\ &\left(-2L_1 m_3 z_3\left(\cos(q_1)\sin(q_3)-\cos(q_3)\sin(q_1)\right)\right)\dot{q}_3^2 \end{aligned} \tag{42}$$

$$
\begin{aligned}
2tauT(2) &= \begin{pmatrix} 2L_1m_2z_2\left(\cos(q_1)\cos(q_2)+\sin(q_1)\sin(q_2)\right) \\ +2L_1L_2m_3\left(\cos(q_1)\cos(q_2)+\sin(q_1)\sin(q_2)\right)\end{pmatrix}\ddot{q}_1 \\
&+\begin{pmatrix} 2L_1m_2z_2\left(\cos(q_1)\sin(q_2)-\cos(q_2)\sin(q_1)\right) \\ +2L_1L_2m_3\left(\cos(q_1)\sin(q_2)-\cos(q_2)\sin(q_1)\right)\end{pmatrix}\dot{q}_1^{\,2} \\
&+\left(2m_3L_2^{\,2}+2m_2z_2^{\,2}\right)\ddot{q}_2 \\
&+\left(2L_2m_3z_3\left(\cos(q_2)\cos(q_3)+\sin(q_2)\sin(q_3)\right)\right)\ddot{q}_3 \\
&+\left(-2L_2m_3z_3\left(\cos(q_2)\sin(q_3)-\cos(q_3)\sin(q_2)\right)\right)\dot{q}_3^{\,2}
\end{aligned} \tag{43}
$$

$$
\begin{aligned}
2tauT(3) &= \left(2L_1m_3z_3\left(\cos(q_1)\cos(q_3)+\sin(q_1)\sin(q_3)\right)\right)\ddot{q}_1 \\
&+\left(2L_1m_3z_3\left(\cos(q_1)\sin(q_3)-\cos(q_3)\sin(q_1)\right)\right)\dot{q}_1^{\,2} \\
&+\left(2L_2m_3z_3\left(\cos(q_2)\cos(q_3)+\sin(q_2)\sin(q_3)\right)\right)\ddot{q}_2 \\
&+\left(2L_2m_3z_3\left(\cos(q_2)\sin(q_3)-\cos(q_3)\sin(q_2)\right)\right)\dot{q}_2^{\,2} \\
&+\left(2m_3z_3^{\,2}\right)\ddot{q}_3
\end{aligned} \tag{44}
$$

Euler-Lagrange term to V:

$$tauV(1) = g\sin(q_1)\left(L_1m_2 + L_1m_3 + m_1z_1\right) \tag{45}$$

$$tauV(2) = g\sin(q_2)\left(L_2m_3 + m_2z_2\right) \tag{46}$$

$$tauV(3) = gm_3z_3\sin(q_3) \tag{47}$$

Euler-Lagrange term to R:

$$\text{tauR}(1) = 0 \tag{48}$$

$$\text{tauR}(2) = 0 \tag{49}$$

$$\text{tauR}(3) = 0 \tag{50}$$

Results for with kinetics, potential and dissipation energies:

$$
\begin{aligned}
tau(1) &= \left(L_1^{\,2}m_2 + L_1^{\,2}m_3 + m_1z_1^{\,2}\right)\ddot{q}_1 \\
&+\left(L_1L_2m_3\cos(q_1-q_2)+L_1m_2z_2\cos(q_1-q_2)\right)\ddot{q}_2 \\
&+\left(L_1L_2m_3\sin(q_1-q_2)+L_1m_2z_2\sin(q_1-q_2)\right)\dot{q}_2^{\,2} \\
&+\left(L_1m_3z_3\cos(q_1-q_3)\right)\ddot{q}_3 \\
&+\left(L_1m_3z_3\sin(q_1-q_3)\right)\dot{q}_3^{\,2} \\
&-\left(L_1gm_2\sin(q_1)+L_1gm_3\sin(q_1)+gm_1z_1\sin(q_1)\right)
\end{aligned} \tag{51}
$$

$$
\begin{aligned}
tau(2) &= \left(L_1L_2m_3\cos(q_1-q_2)+L_1m_2z_2\cos(q_1-q_2)\right)\ddot{q}_1 \\
&+\left(-L_1L_2m_3\sin(q_1-q_2)-L_1m_2z_2\sin(q_1-q_2)\right)\dot{q}_1^{\,2} \\
&+\left(m_3L_2^{\,2}+m_2z_2^{\,2}\right)\ddot{q}_2+\left(L_2m_3z_3\cos(q_2-q_3)\right)\ddot{q}_3 \\
&+\left(L_2m_3z_3\sin(q_2-q_3)\right)\dot{q}_3^{\,2} \\
&-\left(L_2gm_3\sin(q_2)+gm_2z_2\sin(q_2)\right)
\end{aligned} \tag{52}
$$

$$
\begin{aligned}
tau(3) &= \left(L_1m_3z_3\cos(q_1-q_3)\right)\ddot{q}_1 \\
&+\left(-L_1m_3z_3\sin(q_1-q_3)\right)\dot{q}_1^{\,2} \\
&+\left(L_2m_3z_3\cos(q_2-q_3)\right)\ddot{q}_2 \\
&+\left(-L_2m_3z_3\sin(q_2-q_3)\right)\dot{q}_2^{\,2} \\
&+\left(m_3z_3^{\,2}\right)\ddot{q}_3 - gm_3z_3\sin(q_3)
\end{aligned} \tag{53}
$$

Free system matrices

Matrix *M* for the system:

$$M(1,1) = L_1^{\,2}m_2 + L_1^{\,2}m_3 + m_1z_1^{\,2} \tag{54}$$

$$M(1,2) = L_1\cos(q_1-q_2)\left(L_2m_3 + m_2z_2\right) \tag{55}$$

$$M(1,3) = L_1m_3z_3\cos(q_1-q_3) \tag{56}$$

$$M(2,1) = L_1\cos(q_1-q_2)\left(L_2m_3 + m_2z_2\right) \tag{57}$$

$$M(2,2) = m_3L_2^{\,2} + m_2z_2^{\,2} \tag{58}$$

$$M(2,3) = L_2m_3z_3\cos(q_2-q_3) \tag{59}$$

$$M(3,1) = L_1m_3z_3\cos(q_1-q_3) \tag{60}$$

$$M(3,2) = L_2m_3z_3\cos(q_2-q_3) \tag{61}$$

$$M(3,3) = m_3z_3^{\,2} \tag{62}$$

Vector $\underline{C}$ for the system:

$$
\begin{aligned}
C(1) &= \left(L_1L_2m_3\sin(q_1-q_2)+L_1m_2z_2\sin(q_1-q_2)\right)\dot{q}_2^{\,2} \\
&+\left(L_1m_3z_3\sin(q_1-q_3)\right)\dot{q}_3^{\,2}
\end{aligned} \tag{63}
$$

$$
\begin{aligned}
C(2) &= \left(-L_1L_2m_3\sin(q_1-q_2)-L_1m_2z_2\sin(q_1-q_2)\right)\dot{q}_1^{\,2} \\
&+\left(L_2m_3z_3\sin(q_2-q_3)\right)\dot{q}_3^{\,2}
\end{aligned} \tag{64}
$$

$$
\begin{aligned}
C(3) &= \left(-L_1m_3z_3\sin(q_1-q_3)\right)\dot{q}_1^{\,2} \\
&+\left(-L_2m_3z_3\sin(q_2-q_3)\right)\dot{q}_2^{\,2}
\end{aligned} \tag{65}
$$

Vector $\underline{G(q)}$ for the system:

$$G(1) = (-g)\sin\left(q_1\right)\left(L_1 m_2 + L_1 m_3 + m_1 z_1\right) \quad (66)$$

$$G(2) = (-g)\sin\left(q_2\right)\left(L_2 m_3 + m_2 z_2\right) \quad (67)$$

$$G(3) = (-g) m_3 z_3 \sin\left(q_3\right) \quad (68)$$

C. Example in Robotics: Polar Telescopic Robot Manipulator

Fig. 8 shows the polar telescopic robot manipulator system and is an example studied in [12], which presents the Lagrangian approach as a method to obtain the equations of motion of a robot.

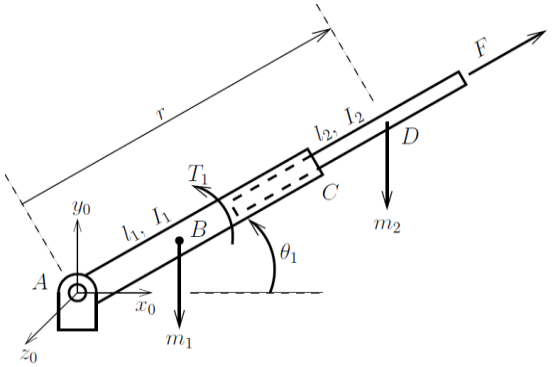

Fig. 8. Polar Telescopic Robotic, with q1 = r, q2 = θ. Source: [12].

1) System energies

Kinetics:

$$2T = \frac{\dot{q}_2^{\,2} l_1^{\,2} m_1}{3} + m_2\left(\dot{q}_1^{\,2} + \dot{q}_2^{\,2} q_1^{\,2}\right) + \frac{\dot{q}_2^{\,2} l_2^{\,2} m_2}{12} \quad (69)$$

Potential:

$$V = g\sin\left(q_2\right)\left(\frac{l_1 m_1}{2} + m_2 q_1\right) \quad (70)$$

Rayleigh dissipation:

$$\mathrm{R} = 0 \quad (71)$$

2) Free system Euler-Lagrange

Euler-Lagrange term to T:

$$2tauT(1) = \left(2m_2\right)\ddot{q}_1 + \left(-2m_2\right) q_1 \dot{q}_2^{\,2} \quad (72)$$

$$2tauT(2) = \left(4m_2\right)\dot{q}_1 q_1 \dot{q}_2 + \left(2m_2\right) q_1^{\,2}\ddot{q}_2 + \left(\frac{2m_1 l_1^{\,2}}{3} + \frac{m_2 l_2^{\,2}}{6}\right)\ddot{q}_2 \quad (73)$$

Euler-Lagrange term to V:

$$tauV(1) = -g m_2 \sin\left(q_2\right) \quad (74)$$

$$tauV(2) = \left(-g m_2 \cos\left(q_2\right)\right) q_1 - \frac{g l_1 m_1 \cos\left(q_2\right)}{2} \quad (75)$$

Euler-Lagrange term to R:

$$\mathrm{tauR}(1) = 0 \quad (76)$$

$$\mathrm{tauR}(2) = 0 \quad (77)$$

Results for with kinetics, potential and dissipation energies:

$$tau(1) = m_2\ddot{q}_1 + \left(-m_2 q_1\right)\dot{q}_2^{\,2} + g m_2 \sin\left(q_2\right) \quad (78)$$

$$tau(2) = \left(2m_2 q_1\right)\dot{q}_1\dot{q}_2 + \left(\frac{m_1 l_1^{\,2}}{3} + \frac{m_2 l_2^{\,2}}{12} + m_2 q_1^{\,2}\right)\ddot{q}_2 + \frac{g\cos\left(q_2\right)\left(l_1 m_1 + 2m_2 q_1\right)}{2} \quad (79)$$

3) Free system matrices

Matrix *M* for the system:

$$M(1,1) = m_2 \quad (80)$$

$$M(1,2) = 0 \quad (81)$$

$$M(2,1) = 0 \quad (82)$$

$$M(2,2) = \frac{m_1 l_1^{\,2}}{3} + \frac{m_2 l_2^{\,2}}{12} + m_2 q_1^{\,2} \quad (83)$$

Vector $\underline{C}$ for the system:

$$C(1) = \left(-m_2\right) q_1 \dot{q}_2^{\,2} \quad (84)$$

$$C(2) = \left(2m_2\right)\dot{q}_1 q_1 \dot{q}_2 \quad (85)$$

Vector $\underline{G(q)}$ for the system:

$$G(1) = (-g)\left(-m_2 \sin\left(q_2\right)\right) \quad (86)$$

$$G(2) = (-g)\left(-\frac{\cos\left(q_2\right)\left(l_1 m_1 + 2m_2 q_1\right)}{2}\right) \quad (87)$$

V. Conclusions

This paper has showed a software tool to obtain and simulate the equations of motion of holonomic mechanical systems. The tool provides a fast and precise development of the differential equations of holonomic dynamical systems and simulation of the system models of any dimension that can be modeled with Euler-Lagrange's equations [12]-[15]. This tool is particular useful for great order systems, which give many terms to differentiate and even have great dimension matrices

An important feature of this tool is to give an option to analyze all steps by looking at the summary report, otherwise, the user should manually write it in LaTeX, which demands some effort, time and is susceptible to errors. Others features

are: (i) Write the differential equation in a *.m* file for a numerical simulation and run it; (ii) Save all portrait phase graph and all time evolution of variables; (iii) For system with constraints, save and plot the contact force on time dependent graph; and finally (iv) Write a summarized report in LaTeX.

This tool seems to apply in different engineering areas as shown in Section IV, as a start this tool proved useful in aerospace, biomedical and robotics and mechanical engineering. Reader is invited to access and test the tool. Files will be sent by authors' e-mails.

REFERENCES

[1] Ivanovich. (2009). Lagrange's equations [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/23037-lagrange-s-equations

[2] M. Kim. (2015). Euler-lagrange tool package. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/49796-euler-lagrange-tool-package

[3] F. Soler, A. Camarero, C. Palomino, J. L. A. Gárate *et al.*, "Labbtex: Toolbox para generación de informes en latex para matlab," *Pensamiento Matemático*, vol. 2, no. 2, pp. 151–156, 2012.

[4] S. Talole and S. Phadke, "Generating LATEX documents through Matlab," *TUGboat*, vol. 24, no. 2, pp. 245–248, 2003. [Online]. Available: http://tug-2000.tug.org/TUGboat/Articles/tb24-2/tb77seta.pdf

[5] F. Knorn. (2005). M-code latex package. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/8015-m-code-latex-package

[6] L. Monteiro, "Sistemas dinˆamicos," *Editora Livraria da F´ısica*, 2006.

[7] S. Sastry, *Nonlinear Systems: Analysis, Stability, and Control*, *ser. Interdisciplinary Applied Mathematics.* New York: Springer, 1999.

[8] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Prentice Hall, 2002.

[9] L. Meirovitch, *Methods of Analytical Dynamics*, ser. Advanced engineering series. MacGraw-Hill, 1970.

[10] J. J. Craig, *Introduction to Robotics*, 3rd ed. Pearson Education, Inc., 2005.

[11] R. Miller and A. N. Michel, *Ordinary Differential Equations*. USA: Academic Press, 1982.

[12] S. B. Niku, *Introduction to Robotics*. John Wiley and Sons, Inc., 2011.

[13] L. A. Al-Zu'be, A. A. Al-Tamimi, T. D. Al-Momani, A. J. Alkarala, and M. A. Alzawahreh, "Modeling and simulating human arm movement using a 2 dimensional 3 segments coupled pendulum system," *World Academy of Science, Engineering and Technology*, vol. 6, 2012.

[14] A. Fenili, "Contact dynamics investigations," *Technical Report Postdoctoral, Deutsches Zentrum f ¨ur Luft- und Raumfart e V. Institute of Robotics and Mechatronics Oberpfaffenhofen, Germany*, 2007.

[15] K. F. Chen, C. Zhang, and F. Huang, "Triple-pendulum model for studying the vibration of multi-degree-of-freedom systems," *Latin America Journal of Physics Education*, 2011.

Fernando H. G. Zucatelli (B.Eng.) received the M.Eng. degree from the Federal University of ABC at 2015. He is a master student of the graduate program in mechanical engineering at Federal University of ABC. He has got experience in instrumentation, automation and robotics engineering with emphasis in control of dynamical systems, development of user interfaces for engineer in Matlab. Mr. Zucatelli received the CREA award for being the best student graduated of his University and course in 2015.

M. E. M. Meza (D.Sc.) received his B.Eng. degree from the Pontifical Catholic University of Peru, and his MSc and DSc from the Federal University of Rio de Janeiro at 1999 and 2004, respectively. He is an assistant professor at the Federal University of ABC, since 2009 until the present and he has got experience in electrical engineering with emphasis in nonlinear control of dynamical systems acting on the following topics: backstepping control, adaptive control, variable structure control, control Lyapunov function and theoretical mathematical ecology.

A. Fenili (D.Sc.) received his B.Eng. degree and his MSC from the State University of Campinas at 1995 and 1997, respectively, and his DSc from the State University of Campinas and Virginia Tech (EUA) at 2000. He has two post-doctoral at INPE (Brazil) and DLR (Germany). Currently, he is an assistant professor level 4 at the Federal University of ABC and he has got experience in aerospace engineering with emphasis in modelling and nonlinear control systems acting on the following topics: flexible structures in rotation; nonlinear system; nonlinear control; contact dynamics; modelling and control of artificial satellites; and fluid-structure interaction.