Design of generic algorithm to calculate the lagrange dynamics of serial robot manipulators and implementing it using Matlab

Submitted by/ Mohamed Magdy Abdelsatar

Under supervision of/ Dr. Roshdy Abu Shanab

Kafrelsheikh University - Faculty of Engineering

Department of Mechatronics

Subject/ Industrial Robots Dynamics, Control and Simulation

Table of Content

1.Scientific Background

1.1. Serial Manipulators

Tsai (1999) defined a serial manipulator as a mechanism consisting of “several links connected in series by various types of joint, typically revolute and prismatic joints”. Another common name for a serial manipulator is ‘robotic arm’. Serial manipulators are the most common industrial robots. Often they have an anthropomorphic arm structure described as having a "shoulder", an "elbow", and a "wrist".

1.2. Lagrange Dynamics

Derivation of the dynamic model of a manipulator plays an important role for simulation of motion, analysis of manipulator structures, and design of control algorithms. The dynamic model of a manipulator provides a description of the relationship between the joint actuator torques and the motion of the structure. With Lagrange formulation, the equations of motion can be derived in a systematic way independently of the reference coordinate frame. Once a set of variables qi,i=1,...,n termed generalised coordinates, are chosen which effectively describe the link positions of ann-DOF manipulator, the Lagrangian of the mechanical system can be defined as a function of the generalised coordinates.

1.3. Mathematical equations

To derive the equation of motion of the manipulator, a set of equations demonstrated below are used. Firstly, forward kinematics of the manipulator should be calculated using the DH-Parameters:

$$ {}^{i-1}A_i = \begin{bmatrix} c\theta_i & -c\alpha_i s\theta_i & s\alpha_i s\theta_i & a_i c\theta_i \\ s\theta_i & c\alpha_i c\theta_i & -s\alpha_i c\theta_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^{A}T_B = \begin{bmatrix} {}^{A}R_B & \vdots & {}^{A}\mathbf{q} \\ \cdots\cdots & \vdots & \cdots\cdots \\ 0\,0\,0 & \vdots & 1 \end{bmatrix} $$

$$ {}^{0}A_1\,{}^{1}A_2\,{}^{2}A_3\cdots{}^{n-1}A_n = {}^{0}A_n $$

Inertia tensor can be thought of as a generalisation of the scalar moment of inertia of an object. The inertia tensor describes the object's mass distribution.

$$ {}^{A}I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix} $$

$$ \begin{aligned} I_{xx} &= \iiint_V (y^2 + z^2)\rho dv, \\ I_{yy} &= \iiint_V (x^2 + z^2)\rho dv, \\ I_{zz} &= \iiint_V (x^2 + y^2)\rho dv, \\ I_{xy} &= \iiint_V xy\rho dv, \\ I_{xz} &= \iiint_V xz\rho dv, \\ I_{yz} &= \iiint_V yz\rho dv, \end{aligned} $$

Ri is the rotation matrix from Link i frame to the base frame.When Referred to the link frame, the inertia tensor is constant.Let $\boldsymbol{I}^{i}_{\ell_i}$ denote such tensor; then it is easy to verify the following relation for transformation of inertia matrix to calculate angular momentum of the link body about origin of the manipulator:

$$ \boldsymbol{I}_{\ell_i} = \boldsymbol{R}_i \boldsymbol{I}^{i}_{\ell_i} \boldsymbol{R}_i^T $$

Link jacobian is calculated using following equations:

$$J_{vi}^{j} = \begin{cases} \mathbf{z}_{j-1} \times {}^{j-1}\mathbf{p}_{ci}^{*} & \text{for a revolute joint,} \\ \mathbf{z}_{j-1} & \text{for a prismatic joint.} \end{cases}$$

$$J_{\omega i}^{j} = \begin{cases} \mathbf{z}_{j-1} & \text{for a revolute joint,} \\ \mathbf{0} & \text{for a prismatic joint.} \end{cases}$$

Mass matrix of manipulator is nxn matrix calculated by:

$$M = \sum_{i=1}^{n} (J_{vi}^{\mathrm{T}} m_i J_{vi} + J_{\omega i}^{\mathrm{T}} I_i J_{\omega i})$$

The elements of matrix B(q,q_dot) are:

$$B_{ij} = \sum_{k=1}^{n} \frac{\partial M_{ik}}{\partial q_j} \dot{q}_k$$

$$\boldsymbol{B}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = [\boldsymbol{B}_1 \, \boldsymbol{B}_2 \dots \boldsymbol{B}_n]$$

$$\boldsymbol{B}_j = \frac{\partial \boldsymbol{M}}{\partial q_j} \dot{\boldsymbol{q}}$$

$$\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{B}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \frac{1}{2} \boldsymbol{B}^{T}(\boldsymbol{q}, \dot{\boldsymbol{q}})$$

Velocity vector is calculated using:

V= C * q_dot

Gravitational vector is calculated using:

$$G_i = -\sum_{j=1}^{n} m_j \mathbf{g}^{\mathrm{T}} J_{vj}^{i}$$

Dynamic motion Equation: **tau= M*q_double_dot + V + G**

2. Software Implementation

2.1. Methodology & Algorithm flowchart

The presented mathematical formulations along with the developed algorithm and methodology for the matlab program can be represented in the form of a flow chart shown in visio file in the lagrange dynamics algorithm flowchart pdf file

2.2. Key techniques used in programming

Matlab software provides a lot of features for working with Matrices and symbols, thus enabling writing algebraic robotic softwares easily. A cell array is a data type with indexed data containers called cells, where each cell can contain any type of data. We used that data type to store matrices resulting from different calculations through the code, then advanced array indexing techniques were used to extract desired cells from the cell arrays. For loops were used to represent any summation or looping over certain portions of code. Also, if conditions were implemented as logical thresholdings. The sequence of operations was taken into consideration and the different data types used were chosen based on its position in the program. Some data were treated as symbols using the symbolic toolbox and others were used for mathematical comparisons and operations and were treated as numbers.

3. References

1- Matlab documentation: https://www.mathworks.com/help

2- B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics: Modelling, Planning and Control

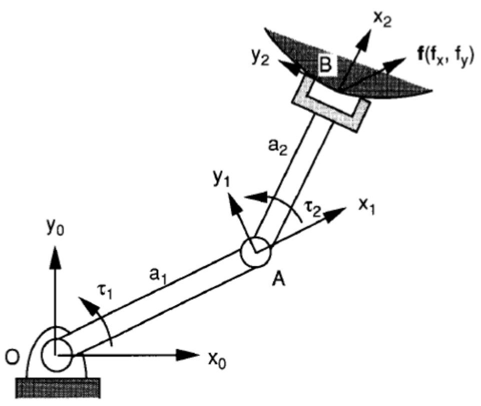
3- LUNG-WEN TSAI, Robot Analysis: The Mechanics of Serial and Parallel Manipulators

4- John J. Craig, Introduction to Robotics: Mechanics and Control

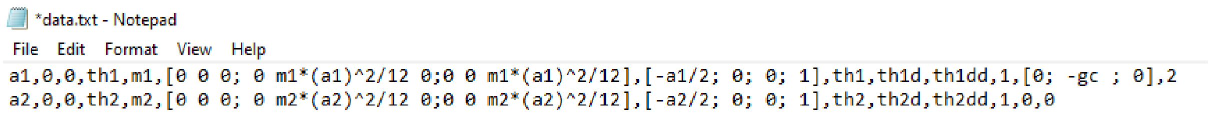
4. Appendix

Example of using the program

Calculating motion equation of two link manipulator RR planer :



Input data file:



```
a1,0,0,th1,m1,[0 0 0; 0 m1*(a1)^2/12 0;0 0 m1*(a1)^2/12],[-a1/2; 0; 0; 1],th1,th1d,th1dd,1,[0; -gc ; 0],2
a2,0,0,th2,m2,[0 0 0; 0 m2*(a2)^2/12 0;0 0 m2*(a2)^2/12],[-a2/2; 0; 0; 1],th2,th2d,th2dd,1,0,0
```

Program execution:

tau =

th2dd*((a2^2*m2)/12 + (a2*m2*(a2 + 2*a1*cos(th2)))/4) + th1dd*((m2*(4*a1^2 + 4*cos(th2)*a1*a2 + a2^2))/4 + (a1^2*m1)/3 + (a2^2*m2)/12) + gc*m2*((a2*cos(th1 + th2))/2 + a1*cos(th1)) + (a1*gc*m1*cos(th1))/2 - (a1*a2*m2*th2d*sin(th2)*(2*th1d + th2d))/2

(a1*a2*m2*sin(th2)*th1d^2)/2 + (a2*m2*(2*a2*th1dd + 2*a2*th2dd + 3*a1*th1dd*cos(th2)))/6 + (a2*gc*m2*cos(th1 + th2))/2

tau(1,1)=tau1

tau(2,1)=tau2