EasyChair Preprint

№ 1619

Kinematics/Dynamics Analysis with ADAMS/MATLAB Co-simulation of a SolidWorks Designed Spatial Robot Arm with Control and Validation of Results

Vikash Kumar

October 10, 2019

Kinematics/Dynamics Analysis with ADAMS/MATLAB Co-simulation of a SolidWorks Designed Spatial Robot Arm with Control and Validation of Results

Vikash Kumar

Department of Mechanical Engineering
Thapar Institute of Engineering and Technology Patiala Punjab- 147004

email: inbox2vikash@yahoo.com

ABSTRACT. This paper present kinematic and dynamic analysis of a 3-R robot arm with Adams/Matlab Co-Simulation, and its control with PID and PID based fuzzy logic controller, finally simulated results is compared with numerical method in Matlab, for kinematics/dynamic analysis a three degree of freedom robot arm with all revolute joint is designed in Solidworks, and designed model is exported in Adams, Exported plant motion is controlled with the Matlab Simulink software. ADAMS software is now widely applied to most of the areas such as Robotics, automobile, mechanism and so on for an automatic dynamic analysis of the mechanical system. The virtual model plant created by ADAMS software is approaching the real model dynamics analysis in Matlab with numerical methods and it can provide a credible result for the real model simulation with ode45 numerical method test and analysis.

Keywords: Spatial Arm, Co-Simulation, Fuzzy Control, PID Control, Dynamic Analysis

1 Introduction

Robotics a field that's is grooving day by day and many researchers still trying to develop an effective algorithm (position and force control algorithms) to develop a robot that is cost-effective and utilized in most of the field for the maximum task. Nowadays, robots used in industries have a wide range of applications, mostly for welding and painting robots in car plants [1], electronic board assembly in electronics and computer fields, repairing nuclear installations in nuclear plant [2] etc. Even though all these tasks can a robot do accurately and precisely and also having more advantages, development of a robot required a lot of time and amount, that involve for testing and validation of robot before it installation is done for the final task which it is made for. With the help of Adams /Matlab Simulation robot can be analyzed kinematically and dynamically [3]. The advantage of co-simulation, it provides a virtual environment for analysis purpose and it does do not require any type of dynamics equation or modelling of system [4]. That's makes it easy and effective for dynamics analysis and its results are comparable

with any numerical and analytical method for validation. Co-simulation of a SolidWorks designed robotic arm and its control with PID [5] and PID based Fuzzy controller is the aim of the paper. To test the simulation results and method for analysis of robot manipulator, a robot manipulator is designed in SolidWorks with 3-DOF as shown in figure 1. and all links dimensions, taken for analysis are shown in table 1.

Table 1. Manipulator parameter used for robot links design in SolidWorks

Links	**Length in mm**	**Width in mm**	**height in mm**
Link1	500	Diameter of shaft 50 mm	
Link2	370.2460679378	44.0492135876mm	27.0246067938
Link3	237.7796579879	27.7779657988	18.8889828994

With taking table 1. Parameters, a three-link manipulator is designed in SolidWorks and it is exported in ADAMS for dynamics analysis, the designed and exported model is shown in fig.1.

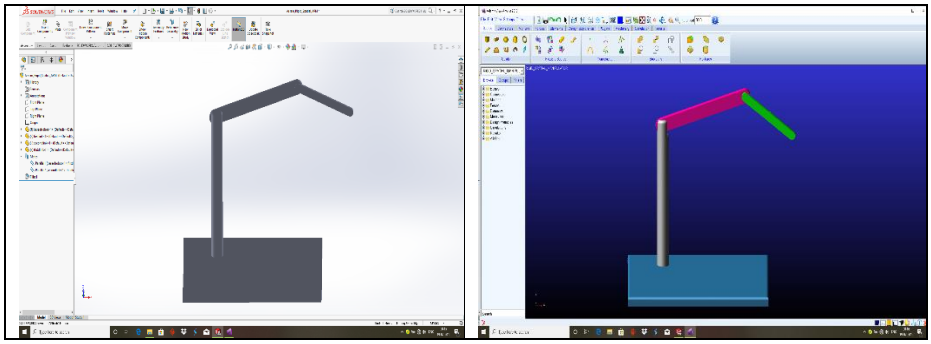

Fig. 1. SolidWorks Designed Three-Links Robot manipulator and same designed model is Exported ADAMS

To export the designed model into ADAMS, the initial step is, first assembled each of the three parts of the robot arm in SolidWorks from "Assembly command" in SolidWorks software. Then assembled arm is saved as "Parasolid" file format, because ADAMS can only recognize some specific file format of SolidWorks. At next this saved model is imported into ADAMS, the imported model of manipulator does not contain mass and inertia as per material, so in ADAMS we defined the type of material for each link and according to type of material all link parameter is automatically assigned to model. All assigned parameter as shown in table 2 are taken to analyses the dynamics.

Table 2. Manipulator links mass and inertia parameters data taken from ADAMS

Links	**Mass**	**I_{xx} (Kg*MM^2)**	**I_{yy} (Kg*MM^2)**	**I_{zz} (Kg*MM^2)**
Link1	7.173866 kg	1.3414708741E+05	1.34149E+05	2231.75143
Link2	3.754276 Kg	5.1823273E+04	5.145724E+04	822.1331641
Link3	1.063915 Kg	6020.2525946	5985.052737	98.2997074

2. Kinematic analysis, Dynamic modelling and Control of the manipulator

2.1 Kinematic analysis

Before exporting ADAMS model of manipulator into MATLAB software, here discussion on forward kinematic equations and analysis of them is done. For this coordinates frames of each joint and Denavit-Hartenberg (D-H) parameter [3] is taken for analysis of kinematics as shown in Fig. 2 and Table 3.

Table 3. Denavit -Hartenberg Parameters

Frame	$\mathbf{a_{i-1}}$	$\boldsymbol{\alpha_{i-1}}$	$\mathbf{d_i}$	$\boldsymbol{\theta_i}$
0-1	0	0	L_1	θ_1
1-2	0	90·	0	θ_2
2-3	L_2	0	0	θ_3
3-4	L_3	0	0	0

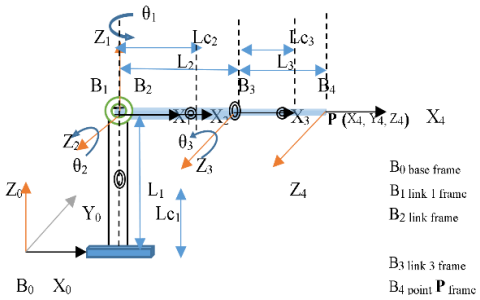

Fig. 2. Attached coordinates of robot joints

The transformation matrices for each frame are computed with the help of table 3. D-H parameter and transformation matrix [3] as given below is used.

$$ {}^{i-1}_{1}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i \cdot c\alpha_{i-1} & c\theta_i \cdot c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} \cdot d_i \\ s\theta_i \cdot s\alpha_{i-1} & c\theta_i \cdot s\alpha_{i-1} & c\alpha_{i-1} & ca_{i-1} \cdot d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1} $$

After using the D-H Parameter Table we get a transformation matrix for each frame, like putting first frame D-H parameters in the transformation table, we get the transformation of frame 1 to frame 0. It is represented as ${}^{0}T_1$ and given in the following way

$$ {}^{0}T_1 = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & 0 \\ s\theta_1 & c\theta_1 & 0 & 0 \\ 0 & 0 & 1 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2} $$

Similarly, we get the transformation matrix for each frame, from frame 4 to frame 0 and multiply all transformation matrices to get the relation between third link endpoint P (X4, Y4, Z4) and base frame B_0 and get ${}^0T_4$ transformation matrix as given below

$$ {}^0T_4 = {}^0T_1 {}^1T_2 {}^2T_3 {}^3T_4 \tag{3} $$

The last column of the ${}^0T_4$ matrix gives the relation between the X, Y, and Z coordinate of the endpoint P with respect to base frame B_0, which are

$$ P_{(X4, Y4, Z4)} = {}^0P_4 = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} \cos\theta_1[L_2\cos\theta_1 + L_3\cos(\theta_2+\theta_3)] \\ \sin\theta_1[L_2\cos\theta_1 + L_3\cos(\theta_2+\theta_3)] \\ L_1 + L_2\sin\theta_1 + L_3\sin(\theta_2+\theta_3) \end{bmatrix} \tag{4} $$

2.2 MATLAB and ADAMS Co-Simulation:

Robot kinematics, dynamics, and control analysis, with the help of Adams and Matlab, makes easy and accurate, so it's the best tool for applying control command to examine system behaviour. With co-simulation of MATLAB/ADAMS, that makes a combined programme and it is a more efficient package to simulate and analysis a system like robot manipulator. In order to make co-simulation workable a common format of input and output is needed, that helps to make changes in one programme automatically which is made by user in other programs. To recognise Matlab exported data by ADAMS, Adams requires, the activation of its control plug-in manager that exports Adams control plant model to Matlab Simulink software.

ADAMS and MATLAB SIMULINK Co-Simulation mean that to make a multi-body system in ADAMS software, define input-output parameters related to system equations and then export system control plant created by Adams to MATLAB/SIMULINK and set control scheme [6]. During the simulation, the data exchange process between ADAMS's virtual prototype and Matlab control program is happening. Where ADAMS works on the mechanical system equations solution and MATLAB works to resolve the equations of control system. They together complete the whole system simulation and control process. The flow chart of co-simulation process is displayed in Fig. 4. ADAMS has complex seven matrices for the dynamic model of the robot and its motion, that is sent to MATLAB during simulation Process. The first four are state-space representation matrices A, B, C, and D. The fifth and sixth matrices are related to inputs and outputs that are predefined. And the last one has information about plant state variables [7,8].

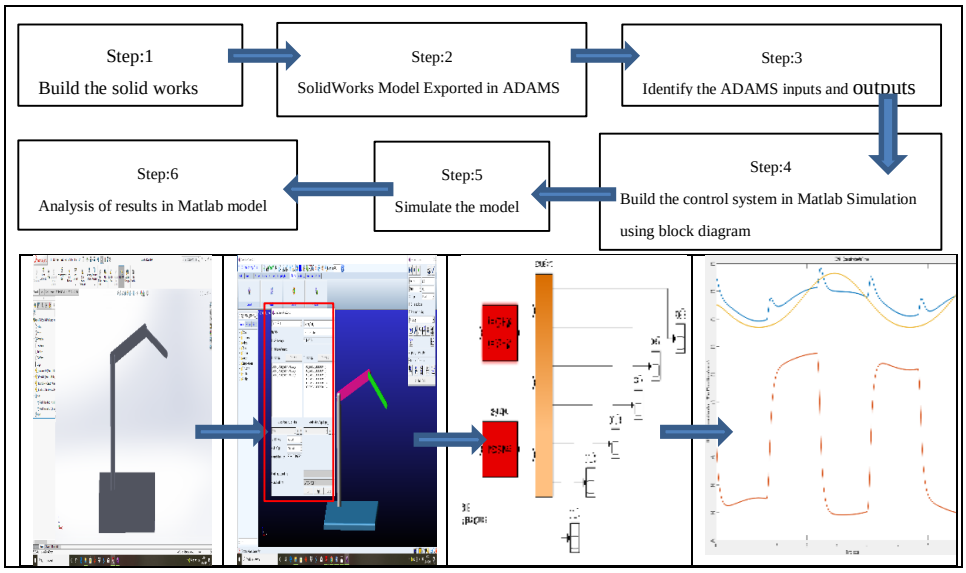

Fig. 3. ADAMS/MATLAB Co-Simulation process Steps

2.2 Dynamics and control system of the manipulator

Dynamics of manipulator

For analyzing robot arm dynamics, here we used transformation matrix ${}^{0}T_{4,}$ which fourth column as given in equation (4), this X, Y and Z coordinate gives a relationship between endpoint and base of the manipulator and set dynamic parameters (Fig. 2). Where the base coordinate system coincides with one end of the link 1. Basic dynamic parameters of links are defined as. m1, m2 and m3 are respectively masses of the links 1 2 and 3; I1, I2 and I3 are respectively rotational inertia around Z-axis; θ1, θ2 and θ3 are respectively angle variable; L1, L2 and L3 are respectively lengths of the three links and Lc1, Lc2 and Lc3 are respectively lengths between centre of mass of the links and joint axes.

Euler-Lagrange method: Total kinetic energy of robot manipulator is given as:

$$T_K = \frac{1}{2}[I_1 + I_2 + I_3 + m_2 L^2{}_2 \cos\theta_2 + m_3\{L^2{}_2 \cos\theta_2 + L^2{}_3 \cos(\theta_2+\theta_3) + 2L_2L_3\cos\theta_2\cos(\theta_2+\theta_3)\}]\dot{\theta}^2{}_1 + \frac{1}{2}[I_2 + I_3 + m_2L^2{}_2]\dot{\theta}^2{}_2 + \frac{1}{2}[I_3 + m_3L^2{}_2]\dot{\theta}^2{}_3 + \frac{1}{2}m_3[L^2{}_3(\dot{\theta}_2+\dot{\theta}_3)^2 + 2L_2L_3\,\dot{\theta}_2((\dot{\theta}_2+\dot{\theta}_3)\cos\theta_3] \quad (5)$$

Total potential energy is given as:

$$U_P = g[m_1 L_{c1} + m_2(L_1 + L_{c2}\sin\theta_2) + m_3(L_1 + L_2\sin\theta_2 + L_{c3}\sin(\theta_2 + \theta_3)] \quad (6)$$

Lagrange function is given by

$$L = T_K - U_P \quad (7)$$

And with above Lagrange, we get the system dynamic equation as:

$$\frac{\mathbf{d}}{\mathbf{dt}}\left(\frac{\partial \mathbf{L}}{\partial \dot{\boldsymbol{\theta}}_\mathbf{i}}\right) - \left(\frac{\partial \mathbf{L}}{\partial \boldsymbol{\theta}_\mathbf{i}}\right) = \boldsymbol{\tau}_\mathbf{i} \quad (8)$$

With the use of equation (8), we can calculate each joint torque as

$$\tau = \mathrm{M}(\mathrm{q})\ddot{q} + C(q)\dot{q} + g(q) + f(\dot{q}) \quad (9)$$

where $\boldsymbol{\tau}$ is torque on each joint actuator, M is inertia matrix, q is joint coordinate vector, C is Coriolis and Centrifugal acceleration matrix, g is gravity vector, and *f* is residual dynamics matrix.

For this example, the manipulator has the following mass matrix in the dynamic model [3]:

$$\mathrm{M} = \begin{vmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{vmatrix} \quad (10)$$

$$\text{Where } m11 = I_1 + I_2 + I_3 + m_2 l_{c2}^2 \cos^2\theta_2 + m_3(l_2\cos\theta_2 + l_{c3}\cos(\theta_2 + \theta_2)]^2 \quad (11)$$

$$m_{33} = I_3 + m_3 l_{c3}^2 \quad (12)$$

Similarly, Centrifugal force, Coriolis force and gravity force and with obtained torque equations we can analysis robot manipulator system numerically with different numerical method.

Control system

In robotics, control of a robot manipulator is considered as a spine of the system, with the help of controller a robot can do the assigned task automatically and precisely. So, the controller is very important part in today's robotics, a controller is nothing it's a form of equation and written is in form of algorithms that can be understood by the computer programs.

The objective of controller implementation in robotics is that the endpoint of a robot called end-effector follows the desired path and orientation accurately and complete the assigned task automatic with higher precision [9]. The motion control of a manipulator either serial or parallel manipulator is done with wide control scheme like feed-forward [2], optimal feedback linearization [10], PID, adaptive backstepping tracking, PID with

sliding mode control [25], inverse dynamics control [9,10, 14], kinematics and dynamics based PID controllers [15] sliding mode control [13], output integral sliding mode controllers [14], and smooth integral sliding mode controllers [16].
Most of these control schemes require mathematical models of the system to implement a control law, so these controllers need the kinematic and dynamic analysis of the model before implementing controller, however in ADAMS /Matlab co-simulation ADAMS take cares of kinematic and dynamic part and Matlab sees control part of the system. In below-given two-section PID and PID based fuzzy controller implementation on manipulator is discussed

PID Controller

Today most of the automatic system used a combination of PID (proportional, integral and derivative) control. This combination is widely applied in the automatic system because of its robustness and ease of implementation, in PID controller we specified some gain to each error signal (proportional error signal, derivative of proportional error signal and integration of proportional error signal from 0 to current time) the control low for PID controller is given as below

$$u = K_P\, e(t) + K_D \frac{de(t)}{dt} + K_I \int e(t)\, d(t) \tag{13}$$

If we ignore dynamic properties of joint driver, then according to PID control law, the driven torque become

$$\boldsymbol{\tau} = K_P\,(\theta_{desired} - \theta) + K_V\,(\dot{\theta}_{desired} - \dot{\theta}) + K_I \int (\theta_{desired} - \theta) \tag{14}$$

$\theta_{desired}$ and $\dot{\theta}_{desired}$ = Respectively desired and feedback signal of joint angle and angular joint velocity.

K_P, K_V and K_I = Respectively related coefficient

In the above equation K_P, K_D and K_I is gain of the error signal in the control loop and their value remain fix after assigning according to desired response pattern of the output results of the system. These gain modified the main system equation and according to that damping coefficient and frequency of the system is changed. For any system to see its performance its rise time, settling time, percentage overshoot and steady-state error for the input signal is used. each gain (K_P, K_D and K_I) value has different effect on these performance parameters. For getting desired performance of the system a very fine-tuning of the gain parameter is required and it's an iterative process and done in a way like if we want to make system response fast we increase proportional element, and if we want to make system slow we increase damp that reduces system overshoot, with increment of differential element we can analyse each element effect on system behaviour. With this process we get PID position control. Matlab also has its inbuilt PID controller block in Matlab Control Toolbox (MCT), however for this paper we have used our own calculated gain value that is computed by analyzing transfer function output on different gain values with step input signal. In this co-simulation work a sim-

ple PID control system was implemented for each individual angle outputs. Implementation of PID controller is shown in fig.4 and its tuned gain parameter and corresponding gains output results are discussed in results section

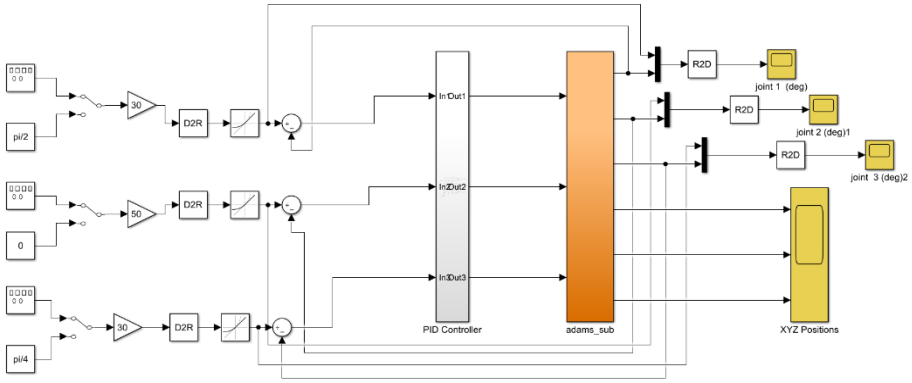

Fig. 4. PID Controller for Co-Simulation

PID Based Fuzzy Logic Controller

As explained above section, the PID controller is used in the industry because of their reliability, robustness and ease of implementation. Although these conventional controllers are very easy to use and effective only with the linear system. However, they are not highly applicable for complex, time-variant, nonlinear due to variations in parameter because of variations in load, friction and external disturbances [17, 18, 19] and also systems with delay. For nonlinear system fuzzy controller is efficient tools. So a combination of conventional [20, 21] PID controller with fuzzy logic controller is better option that maintains easiness and robustness of PID controller and its non-linear effect of the arm is handle by Fuzzy Logic Controller. It improves the system performance in steady and transient state, in this paper a PID controller is tuned with fuzzy logic controller, for this work Matlab control toolbox fuzzy controller is used.
The fuzzy system basic configuration is shown in Figure 5. it has four components [18,20,21, 22, 23,24,25]:

1) Fuzzifier: maps crisp points into fuzzy sets.
2) Fuzzy inference engine: mapping between the fuzzy sets in the input space $U \subset R_n$ to the fuzzy sets in the output space $V \subset R_m$.
3) Fuzzy rule-based system: it is the heart of the whole system in the sense which comprises fuzzy rules describing how the fuzzy system performs.
4) Defuzzifier: maps fuzzy sets of output space into crisp points.

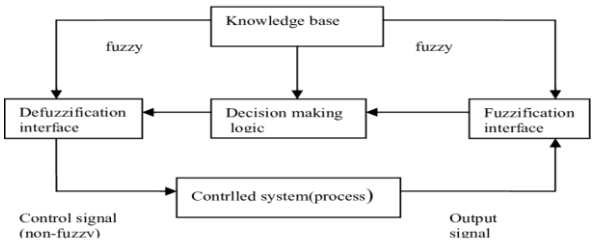

Fig. 5. Fuzzy system basic configuration

There are various forms for fuzzy controller P, PD, PI, controller and PD-type fuzzy controller. In the present work, a PID-type fuzzy controller is applied. In the fuzzy PID controller, the control variables are the error (e) and the change of the error (Δe).
The rule base is given as:
If error (e) is Negative large(NL) and derivative of error(Δe) is Positive large(PL) then u is Zero.
If error (e) is Zero and derivative of error(Δe) is Negative large (NL), then u is Negative large
If error is Zero(ZE) and derivative of error(Δe) is Negative small(NS), then u is Negative small
If error is Zero and derivative of error(Δe) is Zero, then u is Zero
If error is Zero and derivative of error(Δe) is Positive small(PS), then u is Positive small
If error is Zero and derivative of error(Δe) is Positive large then u is Positive large
If error is Positive large and derivative of error(Δe) is Negative large then u is Zero
If error is Positive large and derivative of error(Δe) is Zero, then u is Positive large
In the present work, the product inference rule and Based on the above rule the membership function used is given in below fig.6 and table 4.

Table 4. Rule base for Fuzzy control

Δe\e	NL	NS	ZE	PS	PL
NL	NL	NS	NS	PS	PL
PL	NL	NS	PS	PS	PL

membership functions for error input and output gains

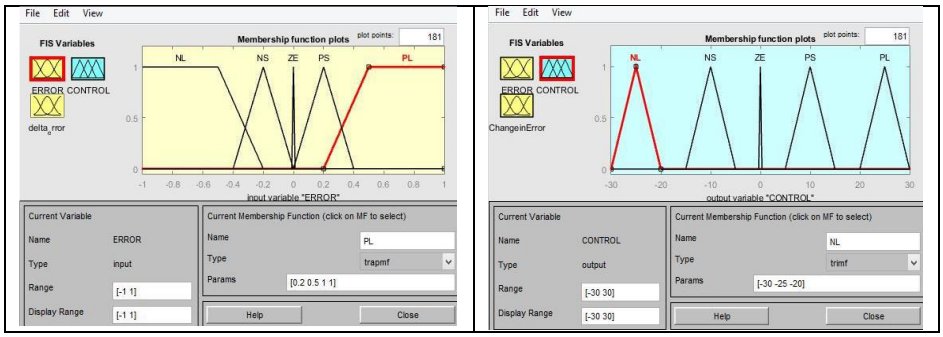

Fig. 6. Membership Functions for inputs (e) and (Δ***e***). and *Control Signal* Memberships function for outputs (K_p, K_i, and K_d).

Implementation of PID based Fuzzy Controller is Shown in fig.7 in this gain of the controller signal is decided by Fuzzy controller, it's not remain fixed as like PID controller in this gain is varying according to some defined limit and controller output signal vary according to gains, controller gains and its output results is discussed in results section.

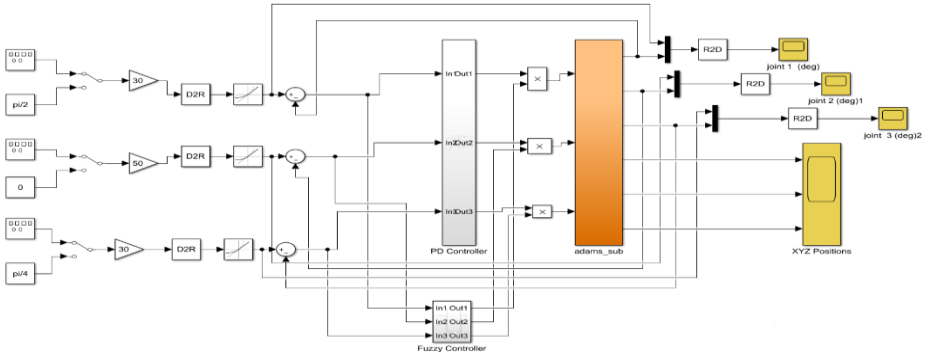

Fig. 7. Fuzzy based PID controller for co-simulation

3. Simulation and Results

To run the simulation, ADAMS exported plant name assigned during plant export time. In MATLAB command window is called by entering 'Adams sys' command and a control plant is created that contains dynamic model information. For simulation, MATLAB Simulink tool is used in which a complete simulation control model is created and output results are analysed with different gains value. Gains selected as shown in table 5. after some trail end error method for PID controller and for Fuzzy based controller limits of the gains is extended that vary according to the input error signal, output results are shown. Fig.8. It shows joint angle in degree versus time graph for each joint with PID and PID based Fuzzy Controller it output compared with the desired signal.

Table 5. Controller gains for PID and PID Fuzzy controller

Constants	PID Controller Gains	PD based Fuzzy Controller
K_P	5	2 to 5
K_I	1	1 to 2
K_D	.3	0.1 to 0 .9

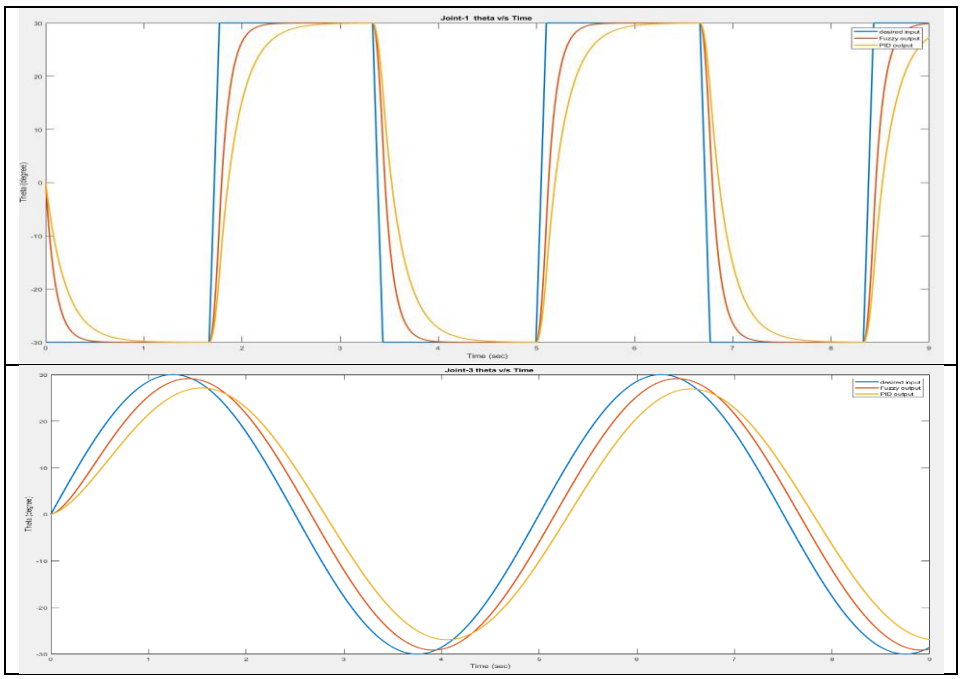

Fig. 8. Joint 1 and joint 2 angle variation with time for square and sine were taken as an input

The endpoint of the link three taken as an end effector and its coordinate X, Y and Z variation with input sigal is shown below in fig 9.

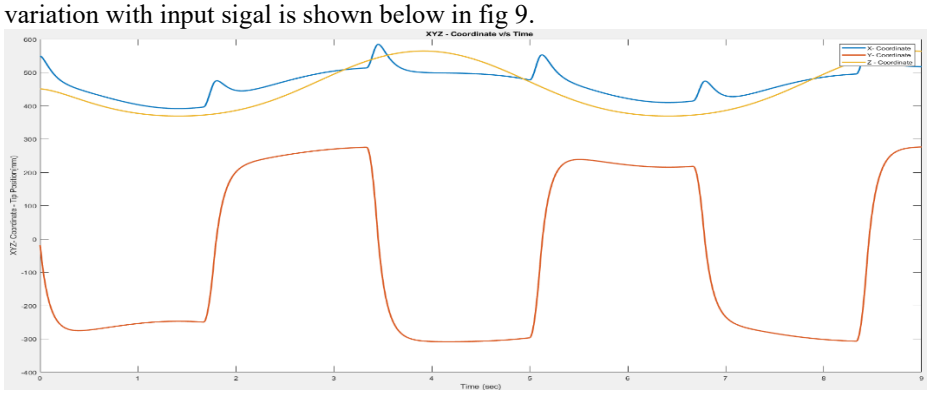

Fig. 9. Variation in X, Y, and Z coordinate of the endpoint with joints inputs

4. Conclusions

In this work, an alternative approach with the use of ADAMS/MATLAB software for the kinematics, dynamics, and control of the virtual developed three DOF robot has been presented and proposed. With the proposed approach we are able to analysis inverse kinematics, forward kinematic, the forward dynamic of the manipulator without the generation of complex dynamics equations, and without hard & complex analytical methods. In this paper kinematic, dynamic, control and variation of each joint angle with time are shown and output of each joint is compared with desired input. With ADAMS/MATLAB software variation of controller gains effect on joints is also shown in joints angle graphs. Further by Adams post-processer we can cross-check all the data, like inertia, centrifugal acceleration, gravity, Coriolis acceleration and instantaneous power of the drive motors with the results that were obtained from MATLAB calculation. As future work, with the help of this approach, several robotic systems behaviour will be tested. Also, trajectory generation behaviour of the manipulator will be analyzed in 3D space to study its velocity, acceleration and force on each manipulator's joints. Some other control algorithms like optimal control can be also tested.

5. References

1. K.T. Park, Y.J. Shin, C.H. Park, Y.S. Mo, and D.C. Jeong, ICCAS, "Robot application for the assembly process of an engine part,"(2008)
2. D. D. Ray and M. Singh, CARPI, “Development of a force reflecting Tele-robot for remote handling in nuclear installations," (2010)
3. J. J. Craig, Introduction to Robotics: Mechanics and Control, 3rd ed. United Kingdom: Prentice Hall, (2003).
4. Deira Sosa-M´endez, Esther Lugo-Gonza´lez, Manuel Arias-Montiel and Rafael A Garcı´a-Garcı´a “ADAMS-MATLAB co-simulation for kinematics, dynamics, and control of the Stewart–Gough platform “ International Journal of Advanced Robotic Systems July-August 2017: 1–10
5. Kisir S and Bingul Z. “ Position control and trajectory tracking of the Stewart platform “. In: Serdar Kucuk (ed) Serial and parallel robot manipulators – kinematics, dynamics, control and optimization. Croatia: InTech, Rijeka, 2012, pp.179–202.
6. Zhu, D.L., J.Y. Qin and Y. Zhang. Research on co-simulation using ADAMS and MATLAB for active vibration isolation system. IEEE ICICTA, 2: 2010, 1126-1129.
7. Z. Affi, L. Romdhane, "ADAMS/SlMULINK interface for Dynamic Modeling and Control of Closed-Loop Mechanisms", Proceedings of the 7th WSEAS International Conference on automatic control, modelling and simulation, Prague, Czech Republic, 2006, pp. 353-356,
8. Z.Z. Naing, M.P. AI-Mamun, "Integrated ADAMS+MATLAB environment for the design of an autonomous single wheel robot, " Industrial Electronics,.IECON 35th Annual Conference of IEEE, 2009, p.p 2253-2258.
9. Taghirad HD. Parallel robots: mechanics and control. 1st ed. Boca Raton, FL, USA: CRC Press, 2013.
10. Tourajizadeh H, Yousefzadeh M, and Tajik A. Closed-loop optimal control of a Stewart platform using an optimal feedback linearization method. Int J Adv Robot Syst 2016; 13(144): 1–9.

11. Lee SH, Song JB, Choi WC, et al. Position control of a Stewart platform using inverse dynamics control with approximate dynamics. Mechatronics 2003; 13: 605–619.
12. Huang CI and Fu LC. Adaptive backstepping tracking control of the Stewart platform. In: Proceedings of 43rd IEEE conference on decision and control, Atlantis, Paradise Island, Bahamas, 14–17 December 2004, pp. 5228–5233.
13. Kisir S and Bingul Z. Position control and trajectory tracking of the Stewart platform. In Serdar Kucuk (ed) Serial and parallel robot manipulators kinematics, dynamics, control and optimization. Croatia: InTech, Rijeka, 2012, pp. 179–202.
14. Fraguela SH, Fridman L, and Alexandrov VV. Output integral sliding mode control to stabilize the position of a Stewart platform. J Franklin Inst 2012; 349: 1526–1542.
15. Ding B, Cazzolato BS, Stanley RM, et al. Stiffness analysis and control of a Stewart platform-based manipulator with decoupled sensor-actuator locations for ultrahigh accuracy positioning under large external loads. J Dyn Sys Meas Control 2014; 136(6): 1–12.
16. Kumar P, Chalamga A, and Bandyopadhyay B. Smooth integral sliding mode controller for the position control of Stewart platform. ISA Trans 2015; 58: 543–551.
17. Huang, S. J., and Lee, J. S.," A stable self-organizing fuzzy controller for robotic motion control" IEEE Trans. Ind. Electro., vol. 47, no. 2, 2000, pp. 421-428.
18. Yoo, B. K, and Ham, W. C.," Adaptive control of robot manipulator using fuzzy compensator" IEEE Trans. Fuzzy Systems, vol. 8, no. 2, 2000, pp. 186-199.
19. Lewis, F. L., Yesildirek, A., and Liu, K., "Multilayer neural-net robot controller guaranteed tracking performance," IEEE Trans. Neural networks, vol. 7, no. 2, 1996, pp. 388-398.
20. Wei Li, "Design of a Hybrid Fuzzy Logic Proportional Plus Conventional Integral Derivative Controller", IEEE trans. on Fuzzy Systems, Vol. 6, No. 4. 1998, pp. 449-63.
21. Li, W. et al, "Tracking Control of a Manipulator under Uncertainty by FUZZY P + ID Controller", Fuzzy Sets and Systems, vol. 122, 2001, pp. 125-137.
22. Guanrong Chen, Young Hoon Joo, 2001, Introduction to Fuzzy Control Systems, Department of Electrical and Computer Engineering, University of Houston, Houston, Texas 77204-4793, USA.
23. Fuzzy Logic Controllers and Methodology, Advantages and Drawbacks; Antonio Sala, Pedro Albertos and Manuel Olivares.
24. MATHWORKS http://in.mathworks.com/help/fuzzy/types-of-fuzzy-inference-systems.html
25. MATHWORKS http://in.mathworks.com/help/fuzzy/what-is-sugeno-type-fuzzy-inference.html