Int j simul model 15 (2016) 2, 302-312
ISSN 1726-4529 Original scientific paper

PATH PLANNING AND CO-SIMULATION CONTROL OF 8 DOF ANTHROPOMORPHIC ROBOTIC ARM

Sudharsan, J.* & Karunamoorthy, L.
Department of Mechanical Engineering, CEG Campus, Anna University, Chennai, India
E-Mail: sudharsan.jayabalan@gmail.com (* Corresponding author)

Abstract
In this research article a kinematic equation for 8 Degrees of Freedom (DOF) anthropomorphic robotic arm was developed and it is modelled using Pro-E software and invoked in ADAMS software tool for further analysis. A cubic path planning algorithm is mathematically derived for actuating the joints and simulated using the MATLAB environment for proper joint motions. With the help of MATLAB/ADAMS Co-Simulation environment the robotic arm invoked in ADAMS model is actuated using the path planning algorithm written in MATLAB environment. The robotic arm traversed the desired trajectory effectively, which confirms the effectiveness of the path planning and control algorithm. These simulated results were used to analyse the dynamic behaviour of the robot arm and gave us a clear insight about the parameters like torque, joint position, velocity and acceleration of the robotic arm and the results have been discussed in detail. This research article is a part of a real time humanoid robot research project titled – RALA (Robot based on Autonomous Learning Algorithm).
(Received in July 2015, accepted in November 2015. This paper was with the authors 1 month for 2 revisions.)

Key Words: Robotics, Humanoid Arm, Dynamic Analysis of Robot Arm, MATLAB / SIMULINK, ADAMS, 8 Degrees of Freedom

1. INTRODUCTION

There is increase in need for humanoids in social and industrial applications. The humanoid arm is used to manipulate the objects in its work environment by properly positioning and orienting the objects as per the requirements. The manipulation capability of the arm will be greatly increased as the dexterity of the arm increases. The dexterity of the arm can be increased by two methods [1]. In the first method, increase the number of DOF in the robotic arm. In the second method, increase the number of DOF in the tools attached with the end effector.

By adding additional degrees of freedom to the end effector tool as suggested by the second method the tool will become bulky as it requires a separate actuation and control unit. Some additional DOF are added to the end effector tools to perform different varieties of operations like drilling, painting, pick and place. This will indirectly increase the complexity and control algorithm in the tool design. This is a major disadvantage of the second method which will practically increase the manufacturing cost of the tools. In this regard the first method stands as best compared to the second method. But the major problem in the first method is that as the number of DOF of the arm increases the kinematic analysis becomes more complex and redundancy of the arm increases [2, 3]. However this can be overcome by training the robot with the help of autonomous learning algorithm. With the help of this algorithm the arm will decide which joint should be moved to achieve the desired manipulation. So the complexity and cost of the robot arm is reduced.

In general minimum 6 degrees of freedom are required for the robot arm to position and orient the object in its three dimensional work space [4-6]. Anthropomorphic arm developed so far has 6 degrees of freedom [5-9] and 7 degrees of freedom [9-12]. Even though the work envelopes of those arms are similar to human work envelope, some of the human movements

cannot be replicated by these arms [13]. Human arm has 3 DOF in shoulder joint, 2 DOF freedom in the elbow joint and 3 DOF in the wrist joint, totalling to 8 DOF excluding the actuation of the fingers [14]. Since the human arm is highly dexterous an 8 DOF is required for the robot arm to imitate the exact motions of the human arm [13]. As suggested by method one [1] an additional degree of freedom is added to the robot arm to increase the dexterity of the object. The need for 8^{th} degree of freedom and kinematic analysis has been explained in the previous research work of RALA [13].

This research article focuses on the efficiency of the path planning/ trajectory tracking of control algorithm with the help of ADAMS/ MATLAB/Simulink Co-Simulation environment. The reason behind the selection of this environment is to execute the control algorithm in real time and analyse the dynamic behaviour of the system through a virtual environment for the real world data's. This will provide accurate information as how the system will run when it is developed and deployed in a real working environment. Dynamic parameters like position, velocity, torque, force acting on the system, and friction generated at each points can be easily evaluated. This greatly reduces the need for mathematical derivation for the above mentioned parameters. It saves the time required to design a complex system and also helps to easily modify the system and simulate the same to achieve the required results. Moreover the dynamic analysis of the Co-Simulation environment provides a real time visual/ video output which greatly helps us to verify the path in which the robotic arm is traversing. It provides accurate information about how the applied trajectory/ path planning algorithm programmed in MATLAB software will behave with the robot in the real world. This reduces the need for prototyping the hardware of the system and then testing the path planning algorithm. This saves money, time and gives ability to modify the system for the user requirements before it has been fabricated. This greatly reduces the need for designing some complex inverse kinematic equation for the robot arm during simulation analysis.

A joint cubic path control algorithm is written in MATLAB/ Simulink programming environment to control the various joint motions of the robot arm. A Co-Simulation environment had been established between ADAMS and MATLAB software. These values are passed among software during real time execution [15, 16]. These values are used for further processing within their (MATLAB, ADAMS) working environment. This was used to verify the control algorithm and also analyse the dynamics response of the system for the given input.

The paper layout is as follows. Section 2 of the paper describes the mathematical modelling of the robot arm. The mathematical equation was obtained for the forward kinematics of the robot arm with the help of frame diagram and Denavit Hartenberg table. The robot arm is modelled using Pro-E environment and it is invoked in ADAMS software tool. Material specification, joint motion and appropriate markers of the robot arm are assigned in ADAMS environment. Section 3 describes the derivations for cubic joint path planning algorithm, which are coded in MATLAB environment. Section 4 describes establishing an ADAMS/ MATLAB/ Simulink Co-Simulation environment and assigning necessary control parameters for data transfers between them during real time execution. Section 5 discuses about the Co-Simulation results obtained for the input trajectories derived using path planning algorithm. Section 6 gives the conclusion to the paper followed by acknowledgements and references.

2. MODELLING OF ANTHROPOMORPHIC ARM

Bio-inspired anthropomorphic robot arm has 8 DOF to position and orient the end effector. 3 DOF is assigned for the shoulders joint, which is connected between the base of the system and Link 1, three revolute joints are assigned to achieve human like spherical joint motion at

shoulder. The revolute joints, Joint-1, Joint-2 and Joint-3 collectively together reproduce the spherical joint motion similar to human arm at the shoulder joint as shown in Fig.1. The revolute joint, Joint-4 and Joint-5, which has 2 DOF, contributes to the elbow joint which is connected between the Link 1 and Link 2 as shown in Fig. 1. Revolute joints Joint-6, Joint-7 and Joint-8 perform motion similar to human wrist; all are assigned for the wrist motion connecting the Link 2 and Link 3, which is the end effector of the robot arm as shown in Fig. 1.

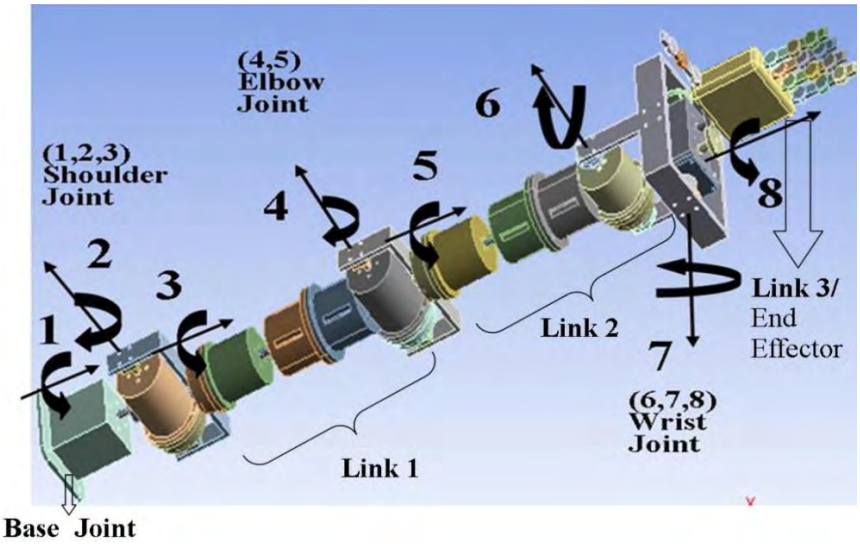

Figure 1: Humanoid arm modelled and simulated using ADAMS simulation software tool.

The tool or end effector in this robotic arm is similar to human palm and fingers. It is used to grasp the objects as shown in Fig. 1. The humanoid arm is modelled using Pro-E software tool and it is shown in Fig. 1. The frames are assigned at home position of the humanoid arm (see Fig. 2). With the help of these frames Denavit Hartenberg (DH) parameters were found and they are tabulated as a DH table for the robot arm as shown in Table I.

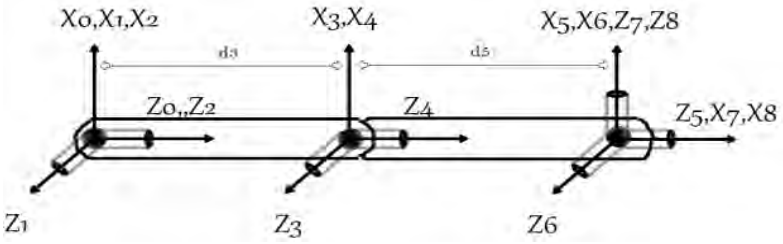

Figure 2: Frame assignment for the humanoid arm.

Table I: Denavit Hartenberg parameters of the robotic arm.

Joint 'i'	$\boldsymbol{\theta_i}$	$\boldsymbol{\alpha_i}$ (degree)	$\boldsymbol{d_i}$ (cm)	$\boldsymbol{a_i}$ (degree)	**Range** (degree)
1	θ_1	-90	0	0	-150^0 to 90^0
2	θ_2	90	0	0	-90^0 to 90^0
3	θ_3	-90	d3	0	0^0 to -90^0
4	θ_4	90	0	0	-15^0 to 140^0
5	θ_5	0	d5	0	0^0 to 150^0
6	θ_6	-90	0	0	0^0 to 360^0
7	θ_7	-90	0	0	75^0 to -80^0
8	θ_8	0	0	0	30^0 to -20^0

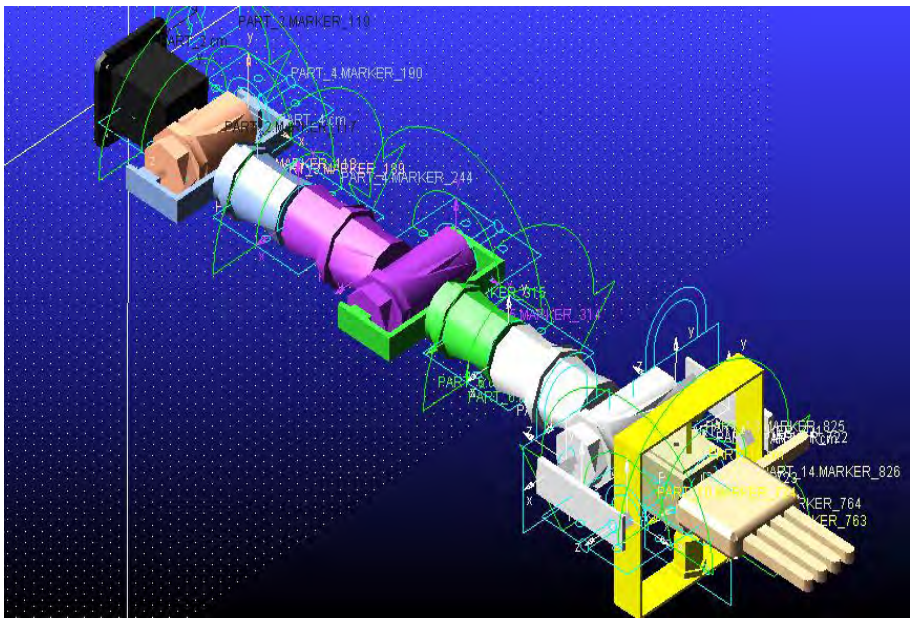

Figure 3: Revolute joint motions are assigned to robot arm, in ADAMS software tool.

The homogeneous transformation matrix is a simple (4 × 4) transformation is used in the kinematics model in robot controllers to examine the rigid-body position and orientation of a sequence of robotic links and joint frames it is shown in Eqs. (1) to (4). The transformation matrix provides information about the position and orientation of the end effector from the base and other joints or vice versa. The homogeneous transformation matrix consists of a (3 × 3) rotational matrix, (3 × 1) position vector, (1 × 3) perspective transformation vector, (1 × 1) scaling vector as shown in Eqs. (1) to (4). The (3 × 1) position vector provides Cartesian co-ordinates value of the link or end effector in a 3-Dimensional space. The (3 × 3) rotational matrix provides information about the orientation of the link or end effector tool at the Cartesian co-ordinates mentioned by the position vector.

The Homogenous Transformation matrix:

$$ {}^{i-1}_{i}T = T_z(d_i) * T_x(a_i) * T_z(\theta_i) * T_x(\alpha_i) \qquad [17] \qquad (1) $$

$$ {}^{i-1}_{i}T = \begin{bmatrix} c_i & -s_i & 0 & a_{i-1} \\ s_i c\alpha_{i-1} & c_i c\alpha_{i-1} & -s a_{i-1} & -d_i s\alpha_{i-1} \\ s_i s\alpha_{i-1} & c_i s\alpha_{i-1} & c a_{i-1} & d_i c\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad [15] \qquad (2) $$

$$ {}^{i-1}_{i}T = \begin{bmatrix} n_x & s_x & a_x & d_x \\ n_y & s_y & a_y & d_y \\ n_z & s_z & a_z & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3) $$

$$ {}^{i-1}_{i}T = \begin{bmatrix} \boldsymbol{Rotational\ matrix}\ [3\times3] & \boldsymbol{position\ vector}\ [3\times1] \\ \boldsymbol{perspective\ transformation}\ [1\times3] & \boldsymbol{scalinig}\ [1\times1] \end{bmatrix} \qquad (4) $$

The following steps help in identifying the DH-Parameters of the Robot arm with which transformation matrix is obtained.

Step 1: Initially home position for the robot arm is achieved by making all the revolute joint angles to zero that is (θ_0 to $\theta_n = 0$). Since the joints are revolute the θ will be the varying parameter (see Fig. 3).

Step 2: Assigning frames to each joint based on Denavit Hartenberg (DH) algorithm as shown in Fig. 2.

Step 3: Joint link parameters are evaluated and they are tabulated in Table I.

Step 4: Denavit Hartenberg parameter values that are obtained from the robotic arm are substituted in the Homogeneous transformation equation (2).

Note: $T_i^{(i-1)}$ represents the homogeneous transformation matrix values from (i-1)th frame to frame (i); i.e. t_{01} represents the homogeneous matrix values to represent 1st frame from 0th frame. Similarly all the values of t_{01} to t_{78} are calculated and they are mentioned below as shown in Eq. (5), where:

$s_n = \sin\theta_n$, $c_n = \cos\theta_n$, $s_{ij} = \sin\theta_i \cdot \sin\theta_j$, $c_{ij} = \cos\theta_i \cdot \cos\theta_j$

d_3 – Link 1 distance, d_5 – Link 2 distance

$$t_{01} = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_1 & c_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad t_{12} = \begin{bmatrix} c_2 & -s_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ -s_2 & c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad t_{23} = \begin{bmatrix} c_{32} & -s_{32} & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ -s_3 & c_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$t_{34} = \begin{bmatrix} c_4 & -s_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ s_4 & c_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad t_{45} = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ s_5 & c_5 & -1 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad t_{56} = \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_6 & c_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$t_{67} = \begin{bmatrix} c_7 & -s_7 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_7 & c_7 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad t_{78} = \begin{bmatrix} c_8 & -s_8 & 0 & 0 \\ s_8 & c_8 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

The end effectors location and orientation from the base frame is obtained by t_{08} in Eq. (6). This is derived by multiplying the values of all the homogeneous transformation equation (5) in a sequential order.

$$t_{08} = t_{01} * t_{12} * t_{23} * t_{34} * t_{45} * t_{56} * t_{67} * t_{78} \tag{6}$$

The material properties are assigned for the robotic arm in ADAMS software tool and are listed in Table II.

Table II: Material properties of the robotic arm.

Joint / Link	**Material Type**	**Mass** (kg)	I_{xx} (kg/m^2) × e^{-04}	I_{yy} (kg/m^2) × e^{-04}	I_{zz} (kg/m^2) × e^{-04}
Joint-1	Aluminium	0.31	3.40	3.40	2.92
Joint-2	Aluminium	0.41	3.20	3.08	1.22
Joint-3	Aluminium	0.91	1.33	8.94	6.86
Link-1	Aluminium	1.09	2.63	2.23	7.37
Joint-4	Aluminium	0.41	3.20	3.08	1.22
Joint-5	Aluminium	0.91	1.33	8.94	6.86
Link-2	Aluminium	1.09	2.63	2.23	7.37
Joint-6	Aluminium	0.16	9.39	5.54	4.08
Joint-7	Aluminium	0.34	1.82	1.41	1.39
Joint-8	Aluminium	0.05	.057	.053	.060
Link-3	Aluminium	0.44	8.09	5.47	2.94

3. DERIVATION OF JOINT CUBIC PATH

Path or trajectory planning is used to control the motions of the robot arm to attain the desired position and orientation in a free space or in presence of any obstacles. With the help of joint path planning the time evolution of joint variables are clearly obtained. Since it does not require inverse kinematics joint path planning helps us to achieve the motion quickly by greatly reducing the computing time. The cubic path planning is derived by providing the initial and final joint conditions such as initial and final joint values and joint acceleration.
Let,

$q_i(t_0) = q_0$, value of joint-*i* at time t_0 (initial joint value) (7)

$q_i(t_f) = q_f$, value of joint-*i* at time t_f (final joint value) (8)

$q_i(t_0) = q_0'$, velocity of joint-*i* at time t_0 (initial velocity) (9)

$q_i(t_f) = q_f'$, velocity of joint-*i* at time t_f (final velocity) (10)

A cubic path in joint space for the joint variable is $\boldsymbol{q_i(t)}$ between two points and $\boldsymbol{q_i(t_0)}$ at initial time $\boldsymbol{t_0}$, and $\boldsymbol{q_i(t_f)}$ at final time $\boldsymbol{t_f}$ is:

$$q_i(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad (11)$$

$$q_i'(t) = a_1 + 2a_2 t + 3a_3 t^2 \quad (12)$$

where,

$$a_0 = -\frac{q_1(t_0^3 - 3t_0^2 t_f) + q_0(3t_0 t_f^2 - t_f^3)}{(t_f - t_0)^3} - \frac{q_0'(t_0 t_f^3 - t_0^2 t_f^2) + q_1'(t_0^2 t_f^2 - t_0^3 t_f)}{(t_f - t_0)^3} \quad (13)$$

$$a_1 = \frac{6q_0 t_0 t_f - 6q_1 t_0 t_f}{(t_f - t_0)^3} - \frac{q_0'(t_f^3 + t_0 t_f^2 - 2t_0^2 t_f) + q_1'(2t_0^2 t_f^2 - t_0^3 - t_0^2 t_f)}{(t_f - t_0)^3} \quad (14)$$

$$a_2 = -\frac{q_0(3t_0 + 3t_f) + q_1(-3t_0 - 3t_f)}{(t_f - t_0)^3} - \frac{q_1'(t_0 t_f - 2t_0^2 + t_f^2) + q_0'(2t_f^2 - t_0^2 - t_0 t_f)}{(t_f - t_0)^3} \quad (15)$$

$$a_3 = -\frac{2q_0 - 2q_1 + q_0'(t_f - t_0) + q_1'(t_f - t_0)}{(t_f - t_0)^3} \quad (16)$$

Similarly cubic path equations are generated for all the 8 joints of the robotic arm by providing the corresponding initial and final conditions for the respective joint. MATLAB code is written for the obtained equations and they are stored in the M-file. This M-file is used in Co-Simulation environment for communicating with ADAMS software tool with the help of MATLAB Simulink environment, which is discussed in detail in section 4. Equations derived for various joint motions are shown as sample calculation and the results are discussed in section 5 for some predefined trajectories.

4. ADAMS/MATLAB CO- SIMULATION ENVIRONMENT

Initially the robot arm is designed using a Pro –E software and it is stored as IGES files. This file is then invoked in ADAMS and necessary markers are assigned as per the requirement of ADAMS simulation environment. Necessary input and output variables in ADAMS software environment are created and assigned to each of these parameters (see Fig. 3). Input parameters are assigned for each revolute joint motion (j1, j2, j3, j4, j5, j6, j7, j8 joint values in degrees) are assigned at appropriate joints. The output parameters like position (angle1 to angle 8 in degrees), torque (torque 1 to torque 8 in Nm) etc. are obtained with the help of appropriate markers placed at each joints (see Fig. 4).

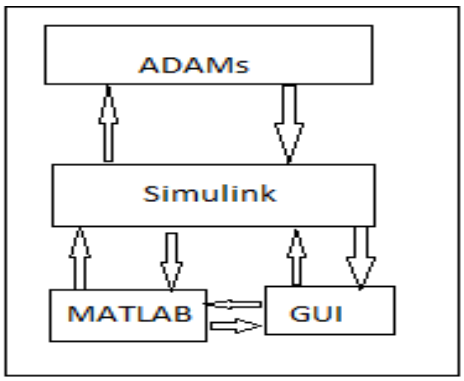

Figure 4: Block diagram of ADAMS/ MATLAB Co-Simulation execution.

A Co-Simulation environment is established using Adams/ Control Plant export. The Adams/ Control plant creates a Matlab type file controls_plant.m, which is invoked in MATLAB programming environment. Adams_sub Simulink block in that file contains the necessary functions to establish communication between Adams and MATLAB/ Simulink environment. Input to this Adams_sub block is provided from "from workspace" block of Simulink library. So that input values can be sent from MATLAB workspace, GUI and using pre written MATLAB files (See Fig. 5). The output from Adams_sub block is connected to "scope" block of Simulink library. From the scope the values were stored in .mat file which are accessed by the control algorithm for further operations.

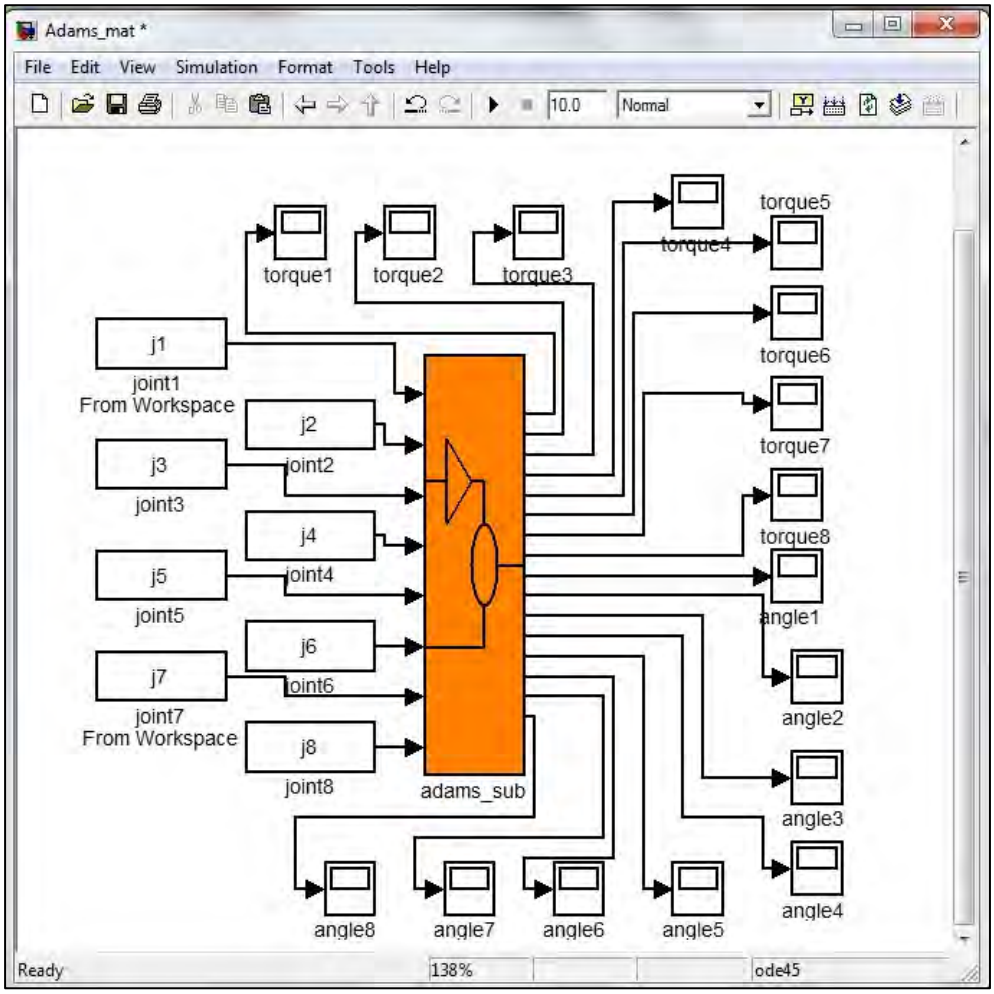

Figure 5: Simulink file containing Adams_sub block for Co-Simulation execution.

The robotic arm is actuated for the desired path by providing the necessary inputs. Input path for the joints are passed from the Matlab file with the help of Matlab workspace are j1, j2, j3, j4, j5, j6, j7, j8 which represent the joint values in degrees to control the arm modelled in the ADAMS environment. The output from the ADAMS can be either viewed in ADAMS software and can also be passed to the MATLAB using Adams_sub block. Since the control algorithm requires torque and angle values from the ADAMS environment these values alone are passed to the Simulink with the help of Adams_sub Simulink block.

5. RESULTS AND DISCUSSION

The humanoid arm is made to traverse through a predefined path/trajectory using the cubic joint path planning technique for the necessary initial conditions. The path planning is obtained for all the joints and it is tabulated in Table III. Joint motions are applied to appropriate joints to achieve this desired path using MATLAB / ADAMS Co-Simulation environment. The simulated output values are analysed and they are compared with theoretical results. Sample numerical calculations are done to verify the simulation results at appropriate values and they are tabulated in Table III. Initial and final joint motions are shown in Fig. 6.

Table III: Trajectory traversed by the robot arm.

Joint / theta	**Initial Conditions**				**Sample calculation at $t = 3$**		**Simulated results at $t = 3$**		
	q_0 (deg)	t_0 (s)	q_f (deg)	t_f (s)	$q_i(3)$ (deg)	$q_i(3)$ (deg/s)	$q_i(3)$ (deg)	$q_i(3)$ (deg/s)	**Torque** (Nm)
1	0	0	0	0	0	0	0	0	2.02
2	0	0	30	4	25.31	8.43	25.31	8.43	11.14
3	0	0	45	5	29.16	12.56	29.16	12.96	1.30
4	0	0	40	4	33.75	11.25	33.75	11.25	2.00
5	0	0	0	0	0	0	0	0	0.10
6	0	0	70	5	45.36	20.16	18.1	31.1	0.12
7	0	0	0	0	0	0	0	0	0.10
8	0	0	0	0	0	0	0	0	0.06

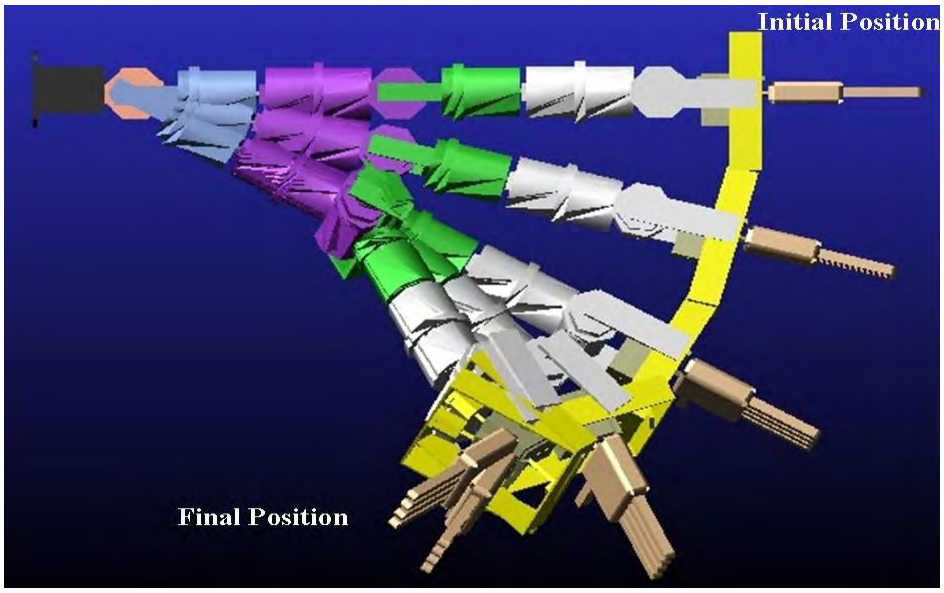

Figure 6: Projectile path generated by the robotic arm for the applied trajectory.

The trajectory traversed by each joint of the robot arm is shown in Fig. 7. Angular velocity generated at each joint of the arm generated during the simulation is shown in Fig. 8. The torques generated at each joint is shown in Fig. 9.

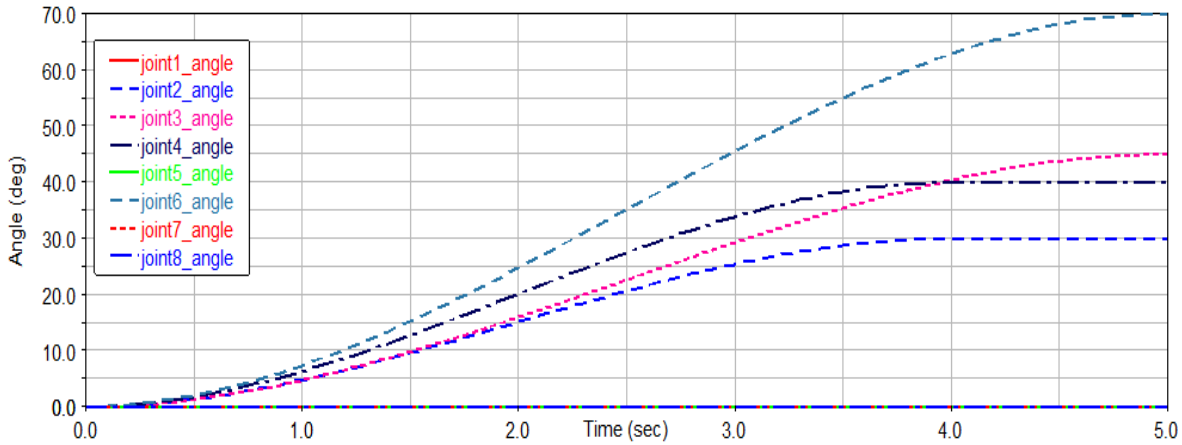

Figure 7: Trajectory traversed by each joint for the applied path.

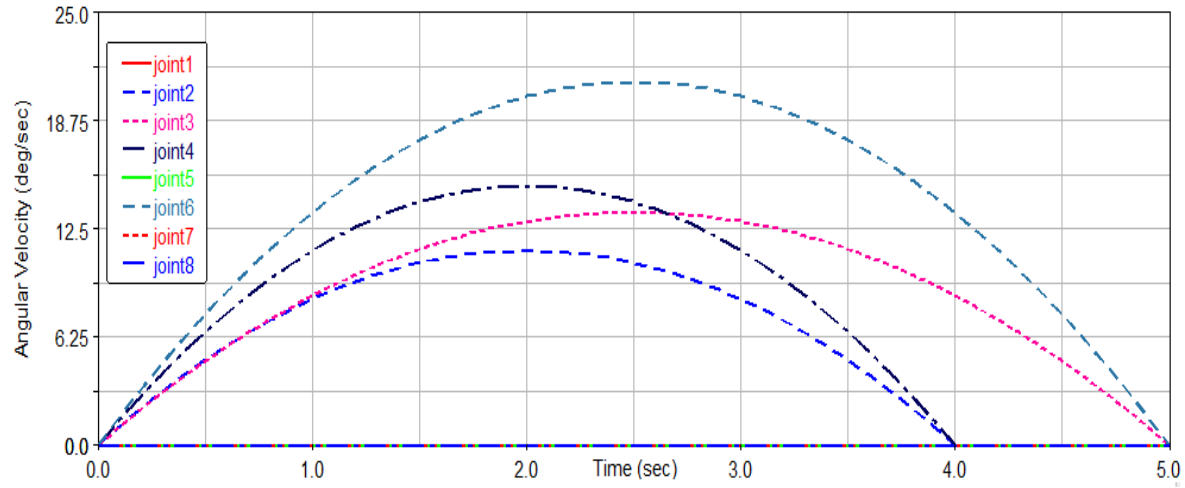

Figure 8: Angular velocity generated at each joint for the applied path.

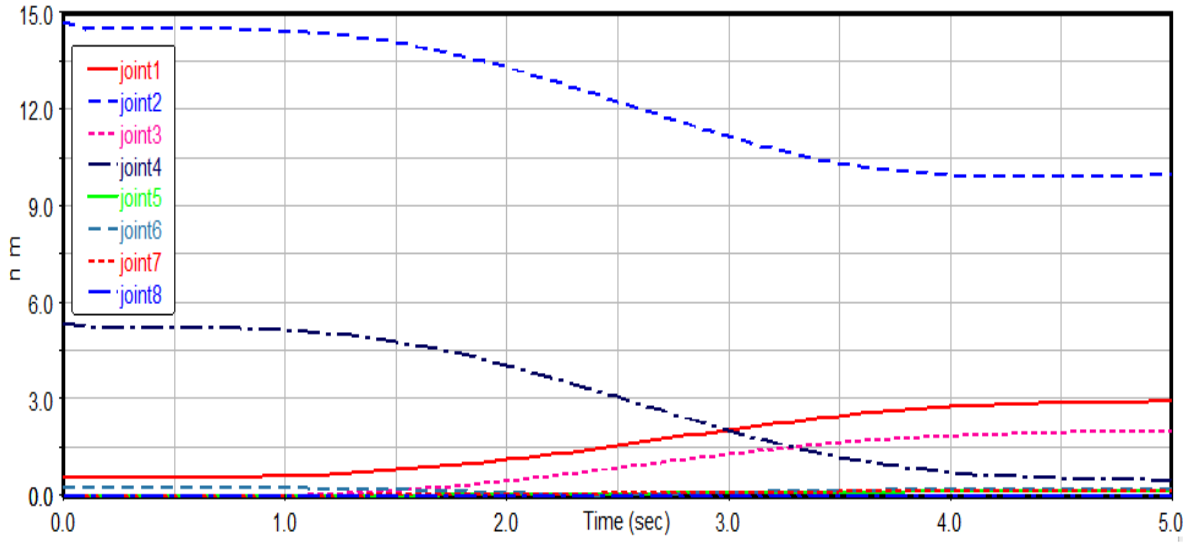

Figure 9**:** Torque generated at each joint for the applied path.

A sample hand calculation for Joint-3, to derive $q_i(t)$ – Eq. (11) and $q_i(t)$ – Eq. (12) for the initial conditions $t_0 = 0,\ t_f = 5,\ q_i(t_0) = 0,\ q_i(t_f) = 45,\ q_i(t_0) = 0,\ q_i(t_f) = 0$ is obtained by substituting these values in Eqs. (11) to (16) at $t = 3$:

$$\boldsymbol{a_0 = 0;\ a_1 = 0;\ a_2 = 5.4;\ a_3 = -0.72}$$

$$q_3(t) = (5.4)t^2 + (-0.72)t^3\ ; \quad \boldsymbol{q_3(3) = 29.16} \tag{17}$$

$$q_3'(t) = 2(5.4)t + 3(-0.72)t^2\ ; \quad \boldsymbol{q_3'(3) = 12.56} \tag{18}$$

Similar hand calculations are made for the applied path, and their values are tabulated in Table III. Theoretical results and simulated results are almost equal, which concludes that the simulated joint cubic path algorithm works well and yields accurate results.

The simulated values provides information about the torque generated at each joints (see Fig. 8) of the system this is helpful to find out the torques required by the motors to drive the system. Due to the dynamic motion of the robotic arm some amount of torque is generated at joints where motions are not applied. The angular velocity at these joints is zero, which infers that this is the holding torque of the joint during this motion which keeps the arm stable. So during the selection of the actuators it is very important to keep this torque into consideration as it can significantly impact the efficiency of the motion generated by the robotic arm

5. CONCLUSION AND FUTURE WORKS

An 8 degree of freedom anthropomorphic robotic arm, which is first of its kind, was designed, simulated, controlled and analysed using ADAMS / MATLAB Co-Simulation environment. The Co-Simulation had worked perfectly as the arm was traversed in the path as desired by the control algorithm. Also the trajectory of the control algorithm is tested and proved to be highly efficient by validating using the theoretical and simulated results. The simulation results are satisfying and give us the clear idea about torque, position, velocity, acceleration of the system. It gives us a clear insight about the behaviour of the control algorithm in the real world environment.

In this research work we have focused on the joint cubic path/ trajectory control of the robot arm. In future an autonomous control algorithm will be designed using neural network and fuzzy logic tool box in MATLAB software to control the Humanoid arm. The output obtained at the end of each phase will be discussed in future communications.

ACKNOWLEDGEMENT

Research Scholar expresses his sincere thanks to Anna University, Chennai, India, for providing Anna Centenary Research Fellowship.

REFERENCES

[1] Ma, R. R.; Dollar, A. M. (2011). On dexterity and dexterous manipulation, *15th International Conference on Advanced Robotics (ICAR)*, Tallinn, 7 pages, doi:10.1109/ICAR.2011.6088576

[2] Craig, J. J. (2005). *Introduction to Robotics: Mechanics and Control*, 3rd edition, Pearson / Prentice Hall, New Jersey

[3] Siciliano, B.; Sciavicco, L.; Villani, L.; Oriolo, G. (2009). *Robotics: Modelling, Planning and Control*, Springer-Verlag, London

[4] Jazar, R. N. (2010). *Theory of Applied Robotics: Kinematics, Dynamics, and Control*, 2nd edition, Springer Science+Business Media, New York.

[5] Shibuya, Y.; Maru, N. (2009). Control of 6 DOF arm of the humanoid robot by linear visual servoing, *2009 IEEE International Symposium on Industrial Electronics*, Seoul, 1791-1796, doi:10.1109/ISIE.2009.5218128

[6] Goldenberg, A.; Benhabib, B.; Fenton, R. (1985). A complete generalized solution to the inverse kinematics of robots, *IEEE Journal of Robotics and Automation*, Vol. 1, No. 1, 14-20, doi:10.1109/JRA.1985.1086995

[7] Iqbal, J.; Ullah, M. I.; Khan, A. A.; Irfan, M. (2015). Towards sophisticated control of robotic manipulators: An experimental study on a pseudo-industrial arm, *Strojniski vestnik – Journal of Mechanical Engineering*, Vol. 61, No. 7-8, 465-470, doi:10.5545/sv-jme.2015.2511

[8] Maronek, M.; Barta, J.; Ertel, J. (2015). Inaccuracies of industrial robot positioning and methods of their correction, *Technical Gazette*, Vol. 22, No. 5, 1207-1212, doi:10.17559/TV-20140416123902

[9] Jerbic, B.; Nikolic, G.; Chudy, D.; Svaco, M.; Sekoranja, B. (2015). Robotic application in neurosurgery using intelligent visual and haptic interaction, *International Journal of Simulation Modelling*, Vol. 14, No. 1, 71-84, doi:10.2507/IJSIMM14(1)7.290

[10] Klanke, S.; Lebedev, D.; Haschke, R.; Steil, J.; Ritter, H. (2006). Dynamic path planning for a 7-DOF robot arm, *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3879-3884, doi:10.1109/IROS.2006.281798

[11] Seraji, H.; Long, M. K.; Lee, T. S. (1993). Motion control of 7-DOF arms: The configuration control approach, *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 2, 125-139, doi:10.1109/70.238277

[12] Kurbanhusen Mustafa, S.; Yang, G.; Yeo, S. H.; Lin, W.; Chen, I-M. (2008). Self-calibration of a biologically inspired 7 DOF cable-driven robotic arm, *IEEE/ASME Transactions on Mechatronics*, Vol. 13, No. 1, 66-75, doi:10.1109/TMECH.2007.915024

[13] Sudharsan, J.; Karunamoorthy, L. (2014). Derivation of forward and inverse kinematics of 8 – degrees of freedom based bio –inspired humanoid robotic arm, *Advanced Materials Research*, Vols. 984-985, 1245-1252, doi:10.4028/www.scientific.net/AMR.984-985.1245

[14] Thompson, J. C.; Netter, F. H. (2010). *Netter's concise orthopaedic anatomy*, 2nd edition, Saunders Elsevier, Philadelphia

[15] Luo, H.; Liu, Y.; Chen, Z.; Leng, Y. (2013). Co-simulation control of robot arm dynamics in ADAMS and MATLAB, *Research Journal of Applied Sciences, Engineering and Technology*, Vol. 6, No. 20, 3778-3783

[16] Yang, C.; He, J.; Jiang, H.; Han, J. (2008). Modeling and simulation of 6-DOF parallel manipulator based on PID control with gravity compensation in Simulink/ADAMS, *International Workshop on Modelling, Simulation and Optimization (WMSO '08)*, 391-395, doi:10.1109/WMSO.2008.57

[17] Goehler, C. M. (2007). *Design of a humanoid shoulder-elbow complex*, Doctoral Dissertation, Aerospace and Mechanical Engineering, University of Notre Dame, Notre Dame