I.J. Intelligent Systems and Applications, **2012, 9, 81-91**
Published Online August 2012 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2012.09.11

A Novel Application of Artificial Neural Network for the Solution of Inverse Kinematics Controls of Robotic Manipulators

Santosh Kumar Nanda
Department of Computer Science and Engineering, Eastern Academy of Science and Technology, Bhubaneswar, Odisha, India – 754001
Email: sknanda@eastodissa.ac.in, santoshnanda@live.in

Swetalina Panda
Department of Computer Science and Engineering, Eastern Academy of Science and Technology, Bhubaneswar, Odisha, India – 754001
Email: sweta15panda@gmail.com

P Raj Sekhar Subudhi
Department of Computer Science and Engineering, Eastern Academy of Science and Technology, Bhubaneswar, Odisha, India – 754001
Email: yushraj@sify.com

Ranjan Kumar Das
Department of Computer Science and Engineering, Eastern Academy of Science and Technology, Bhubaneswar, Odisha, India – 754001
Email: ranjandas2k4@gmail.com

Abstract— In robotic applications and research, inverse kinematics is one of the most important problems in terms of robot kinematics and control. Consequently, finding the solution of Inverse Kinematics in now days is considered as one of the most important problems in robot kinematics and control. As the intricacy of robot manipulator increases, obtaining the mathematical, statistical solutions of inverse kinematics are difficult and computationally expensive. For that reason, now soft-computing based highly intelligent based model applications should be adopted to getting appropriate solution for inverse kinematics. In this paper, a novel application of artificial neural network is used for controlling a robotic manipulator. The proposed methods are based on the establishments of the non-linear mapping between Cartesian and joint coordinates using multi layer perceptron and functional link artificial neural network.

Index Terms— Inverse Kinematics, Artificial Intelligence, Multilayer Perceptron, Functional Link Artificial Neural Network

I. Introduction

A manipulating arm of a manipulator are consists of links that are connected by joints. These joints are either rotational or prismatic, i.e. change their length and thus providing additional degrees of freedom. The position of the manipulators end-effectors is fully described by its joint angles and joint offset by giving the knowledge of length of all rigid links. Therefore, to developing control mechanism for the robot arms, we have to understand the relationship between the actuators. If the position or angle of each joint known to us, then we can calculate the position of its end effectors using mathematical relationship (trigonometric) and this process is known as Forward Kinematics. Alternative relationship to forward kinematics is known as Inverse Kinematics [1-3].

In comparison to forward kinematics, understanding inverse kinematics is difficult. In forward kinematics, real systems are under-constrained, for a given goal position, there could be infinite solutions. In briefly, for inverse kinematics, many different joint configurations could lead to the same end point. Therefore, the inverse kinematics solution is a major problem in robotic research area. For solving the inverse kinematics, many traditional methods such as algebraic, geometric and iterative solutions are adopted. These methods are time consuming, high computational cost and suffer from numerical problems. These demerits of the mathematical models enthusiastic researchers for applying intelligent systems such as Fuzzy Logic, Neural Network etc. for solving inverse kinematics [4-6].

From the literature review, it was found that many methods have been proposed to solve the inverse kinematics instead of traditional computational methods. Utilization of Artificial Neural Network (ANN), Fuzzy

Logic System and Evolutionary computation algorithms for solving the inverse kinematics was reported in literatures. For ANN, presence of hidden layers, for fuzzy logic, number of rule bases and for evolutionary methods, the complex algorithms are associated with high complexity and high computational load. These methods may give good results, however these methods are difficult to implemented hardware. To overcome this problem, Functional Link Artificial Neural Network (FLANN) was applied to solve the inverse kinematics after successfully implemented in other complex engineering problem [5-7].

In this paper, Multi Layer Perceptorn (MLP) and FLANN are used to solve the inverse kinematics problem. The paper is organized as follows, in section 2, introduction to robot manipulator, inverse kinematics are discussed. Section 3 represents system architecture of MLP and FLANN. Simulation result and conclusion are followed in section 4 & 5 respectively.

II. Introduction to Manipulator

According to robotic research, an industrial robot is comprised of a robot manipulator, power supply, and controllers. In general, robot manipulators are created from a sequence of link and joint combinations. The links are the rigid members connecting the joints, or axes. The axes are the movable components of the robotic manipulator that cause relative motion between adjoining links. Therefore, manipulators deal with inverse kinematics and direct kinematics. Manipulator control problem can be identified into two special areas and that are kinematics control (the coordination of the links of kinematics chain to produce desire motion of the robot), and dynamic control (driving the actuator of the mechanism to follow the commanded position velocities). Most of the problems are dealing with kinematics control and this paper highlighted one application on inverses kinematics hence dynamic control not discussed here. The details of inverse kinematics were briefly discussed as follows [2-4].

A. Introduction to Inverse-Kinematics

Inverse kinematics refers to the use of the kinematics equations of a robot to determine the joint parameters that provide a desired position of the end-effectors. Specification of the movement of a robot so that its end-effectors achieves a desired task is known as motion planning. Inverse kinematics transforms the motion plan into joint actuator trajectories for the robot. The Inverse Kinematics is the opposite problem as compared to the forward kinematics. The set of joint variables when added that give rise to a particular end effectors or tool piece pose. Kinematics is the study of motion [3-4]. In this proposed work, solution of inverse kinematics of 3R and 2R planar manipulator is discussed. Fig.1 represent the schematic diagram of both 3R and 2R planar manipulator

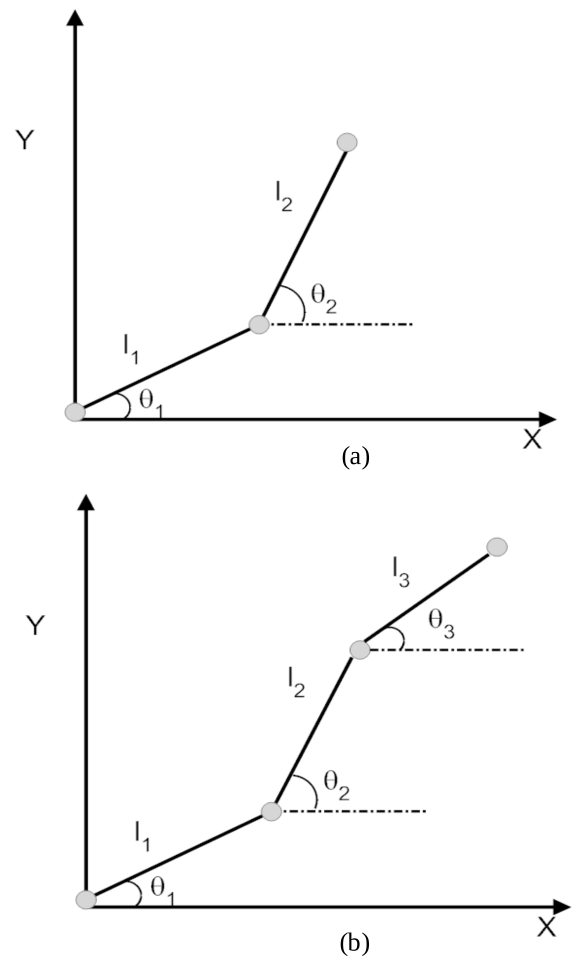


Fig.1: (a) Two DOF Manipulator (b) Three DOF Manipulator

B. Mathematical Solution to 3R Planar:

From the basic trigonometric, the position and orientation of the end effectors can be written in terms of the joint coordinate, the direct kinematics equations are:

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \quad (1)$$

$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\theta_1 + \theta_2 + \theta_3) \quad (2)$$

$$\varphi = \theta_1 + \theta_2 + \theta_3 \quad (3)$$

Equations 1 , 2 & 3 are the set of non-linear equation that describes the relationship between end effectors coordinates and joint coordinates, with a notice that all angles should be measured counter clockwise. For using the forward equations, the inverse kinematics solutions are as follows. Substitute equation 3 in to equation 1 & 2, θ_3 can be eliminated and two equations are generated

$$x - l_3 \cos(\varphi) = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \quad (4)$$

$$y - l_3 \sin(\varphi) = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \quad (5)$$

Replacing $x^{/} = x - l_3 \cos(\varphi)$, $y^{/} = y - l_3 \sin(\varphi)$ regrouping and add each term of equation, we get a single non-linear equation in θ_1.

$$(-2l_1x' \cos\theta_1) + (-2l_1 y' \sin\theta_1) + (x'^2 + y'^2 + l^2{}_1 - l^2{}_2) = 0 \quad (6)$$

Solving to equation (6) as consider it a single non-linear equation (P Cosα + Q Sinα + R =0) the θ_1 becomes

$$\theta_1 = \gamma + \sigma \cos^{-1}\left|\frac{x'^2 + y'^2 + l^2{}_1 - l^2{}_2}{2l_1\sqrt{x'^2 + y'^2}}\right| \quad (7)$$

where

$$\gamma = a\tan 2\left|\frac{-y'}{\sqrt{x'^2 + y'^2}}, \frac{-x'}{\sqrt{x'^2 + y'^2}}\right| \quad (8)$$

There are two solution for θ_1, as $\sigma = \pm 1$, hence the tow solutions are formed

$$Cos(\theta_1 + \theta_2) = \frac{x' - l_1\cos\theta_1}{l_2} \quad (9)$$

$$Sin(\theta_1 + \theta_2) = \frac{y' - l_1\sin\theta_1}{l_2} \quad (10)$$

These two equations (9 and 10) are helping to solve the θ_2 using atan2 and we get the value of θ_2

$$\theta_2 = a\tan 2\left|\frac{y' - l_1\sin\theta_1}{l_2}, \frac{x' - l_1\cos\theta_1}{l_2}\right| - \theta_1 \quad (11)$$

Squaring the equations (4) and (5) and using Cos (a + b) formulae, we get the solution for θ_1

$$\theta_1 = a\tan 2\,\{(k_1y' - k_2x'), (k_1x' - k_2y')\} \quad (12)$$

where $k_1 = l_1 + l_2\cos\theta_2$, $k_2 = l_2\sin\theta_2$, $x' = x - l_3\cos(\varphi)$, $y' = y - l_3\sin\theta_2$. Then the solution of θ_2

$$\theta_2 = a\tan 2\,(\sin\theta_2, \cos\theta_2) \quad (13)$$

$$Cos\theta_2 = \frac{x^2 + y^2 - l^2{}_1 - l^2{}_2}{2l_1l_2} \quad (14)$$

$$Sin\theta_2 = \sqrt{\pm(1 - \cos\theta_2)} \quad (15)$$

Solution for θ_3

$$\theta_3 = \varphi - (\theta_1 + \theta_2) \quad (16)$$

Using the above expressions (12, 14, 16), a solution for 3-R planner is developed using MATLAB Graphical user interface (GUI). Fig. 2 represents the solution for 3-R Planner using MATLAB GUI. Fig. 3 represents the Simulation result for 3R planner using MATLAB GUI.

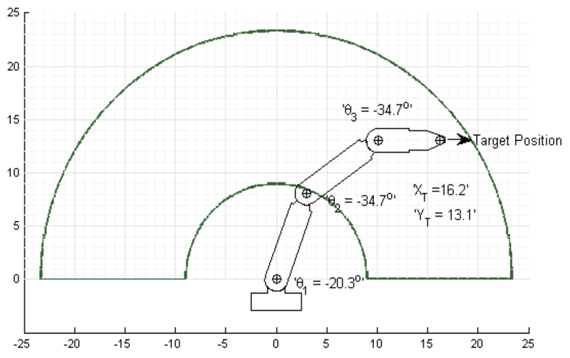
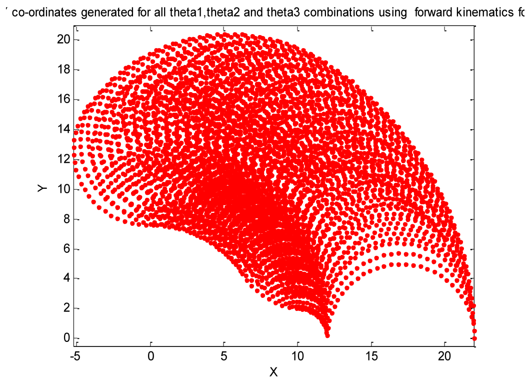
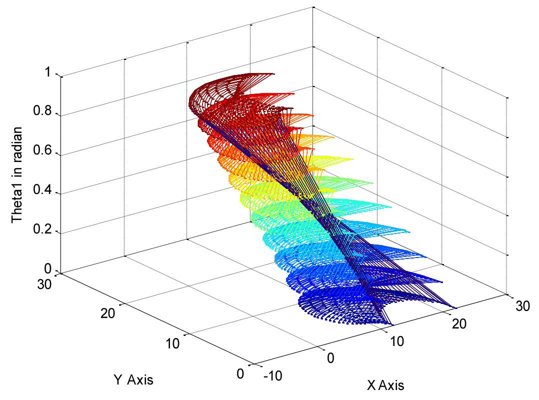


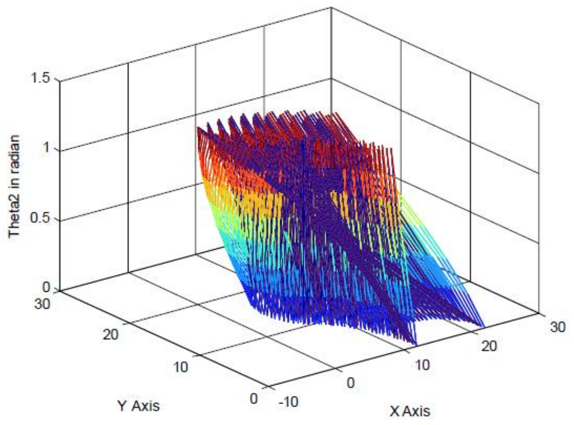
Fig.2: Solution to 3-R Planner using MATLB-GUI



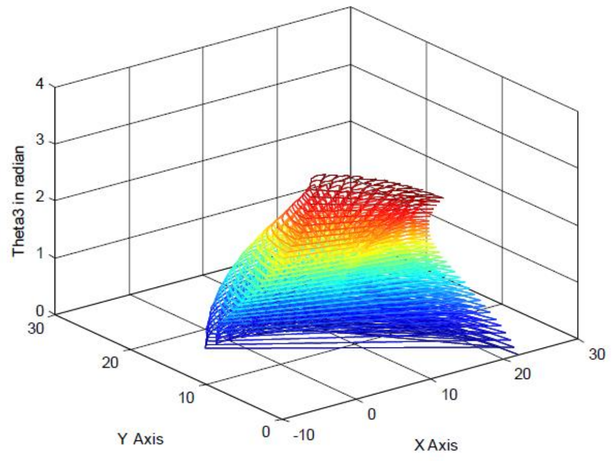
(a)



(b)



(c)



(d)

Fig.3: Simulation result of 3-R Planner using MATLB-GUI
(a) Workspace of the three-link planner
(b) Surface plot of the three-link planner (θ_1 with X and Y coordinates)
(c) Surface plot of the three-link planner (θ_2 with X and Y coordinates)
(d) Surface plot of the three-link planner (θ_3 with X and Y coordinates)

Mathematical Solution to 2R Planar:

Similar to 3-R Planner, forward kinematic equations for 2-R planner are

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \quad (17)$$

$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \quad (18)$$

Squaring the equations (17) and (18) and add these equations we will get

$$x + y = l^2{}_1 + l^2{}_2 + 2l_1 l_2 \cos(\theta_1 + \theta_2) \quad (19)$$

Solving the equation (19), we will get

$$Cos\theta_2 = \frac{x^2 + y^2 - l^2{}_1 - l^2{}_2}{2l_1 l_2} \quad (20)$$

and $Sin\theta_2 = \sqrt{\pm(1 - \cos\theta_2)}$ (21)

Hence we will find the solution for θ_1, θ_2 as follows:

$$\theta_1 = a\tan 2\,\{(y, x), (k_1, k_2)\} \quad (22)$$

$$\theta_2 = a\tan 2\,(\sin\theta_2, \cos\theta_2) \quad (23)$$

III. Application of Artificial Neural Network for Solution of Inverse Kinematics

In this proposed work, the inverse kinematics problem was simulated using Artificial Neural Network (ANN) methodologies [8-12]. Two type of ANN methods, Multi-Layered Perceptron (MLP) and Functional Link Artificial Neural Network (FLANN) [13-16] were used to solve Inverse Kinematics problem. The algorithms of both the techniques were discussed as follows:

Application of Multi Layer Perceptron based solution to Inverse Kinematics

Stepwise algorithm of MLP system is represented as following manner:

Step 1: Select the total number of layers m, the number n_i (i=1,2, . . . ,m−1)of the neurons in each hidden layer and an error tolerance parameter 0.

Step 2: Randomly select the initial values of the weight vectors $w^m i,j$, for i=1,2, . . . ni and m=2 (number of layers).

Step 3 (Initialization): All the weights $w^m\ i,j$ were initialized to random number and given as $w^m\ i,j\,(0)$

$$w^m\ i,j \leftarrow w^m\ i,j(0) \quad (24)$$

Step 4 (Calculation of the neural outputs):

$$\begin{cases} a^m{}_{i,j} = (w^1{}_{i,j}) \times X_k \\ y_j = (w^2{}_{i,j}) \times a^m{}_j \end{cases} \quad (25)$$

Step 5 (Calculation of the output error):
The error was calculated as *ej=dj–yj*. It may be seen that the network produces a scalar output.

Step 6 (Calculation of the derivative of network output of each layer):

For hidden layer (*m*=1)

The derivative of activation function of hidden layer can be represented as

$$\overset{\bullet}{f}{}^1(n^1) = \frac{d}{dn}\left(\frac{1}{1+\exp^{-n}}\right) = \left(1 - \frac{1}{1+\exp^{-n}}\right) \times \left(\frac{1}{1+\exp^{-n}}\right) = (1 - a^m{}_{i,j})(a^m{}_{i,j}) \quad (26)$$

For output layer (*m*=2)

The derivative of activation function of output layer can be represented as

$$\overset{\bullet}{f}{}^2(n^2) = \frac{d}{dn}(n) = 1 \quad (27)$$

Step 7 (Back-propagation of error by sensitivities at each layer):

Back-propagation of error by sensitivities at each layer was calculated as follows:

For output layer (*m*=2)

$$s^2{}_j = -2\overset{\bullet}{F}{}^2(n^2)(d_j - y_j) = -2\overset{\bullet}{f}{}^2(n^2) \times (e_j) \quad (28)$$

For hidden layer (m=1)

$$s^1{}_j = \overset{\bullet}{F}{}^1(n^2)(w^2{}_{i,j})^T s^2{}_j = \begin{bmatrix} (1-a^1{}_{i,j})(a^1{}_{i,j}) & 0 & \cdots & 0 \\ 0 & (1-a^2{}_{i,j})(a^2{}_{i,j}) & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & (1-a^1{}_{n_i,j})(a^1{}_{n_i,j}) \end{bmatrix} \times (w^2{}_{i,j})^T s^2{}_j \qquad (29)$$

Step 8 (Updating the weight vectors):

The weight matrices are updated next using the following relationship

$$\begin{aligned} w^2{}_{i,j}(new) &= w^2{}_{i,j}(old) + \eta s^2{}_j (a^1{}_{i,j})^T \\ w^1{}_{i,j}(new) &= w^1{}_{i,j}(old) + \eta s^1{}_j \end{aligned} \qquad (30)$$

Step 9:

If error ≤ ε (0.01) then go to Step 10, otherwise go to Step 3.

Step 10:

After the learning is completed, the weights were fixed and the network can be used for testing.

Application of Functional Link Artificial Neural based solution to Inverse Kinematics

Step 1: Initialize the inputs as X and Y coordinates and the desired output of the manipulator.

Step 2: Randomly select the initial values of the weight vectors w_i, for i = 1, 2, . . . l, where "i" is the number of functional elements.

Step 3 (Initialization): All the weights w_i were initialized to random number and given as $w_i(0)$

$$w_i \leftarrow w_i(0) \qquad (31)$$

Step 4 (Produce functional block): For FLANN the functional block is made as follows:

$$X_i = [1, x_1, sin(\pi x_1), cos(\pi x_1), x_2, sin(\pi x_2), cos(\pi x_2) \ldots] \qquad (32)$$

Step 5 (Calculation of the system outputs): For functional based neural network models, the output was calculated as follows:

$$O_j = \tanh\left(\sum_{i=1}^{N} w_i \times X_i\right) \qquad (33)$$

Step 6 (Calculation of the output error): The error was calculated as $e_i = d_i - O_i$. It may be seen that the network produces a scalar output.

Step 7 (Updating the weight vectors): The weight matrixes are updated next using the following relationship:

$$w_i(k+1) = w_i(k) + \alpha e_i(k) X_i(k) \qquad (34)$$

where k is the time index and α is the momentum parameter.

Step 8: If error ≤ ε (0.01) then go to Step 8 otherwise, go to Step 3.

Fig.4 represents the flowchart of the proposed FLANN based architecture.

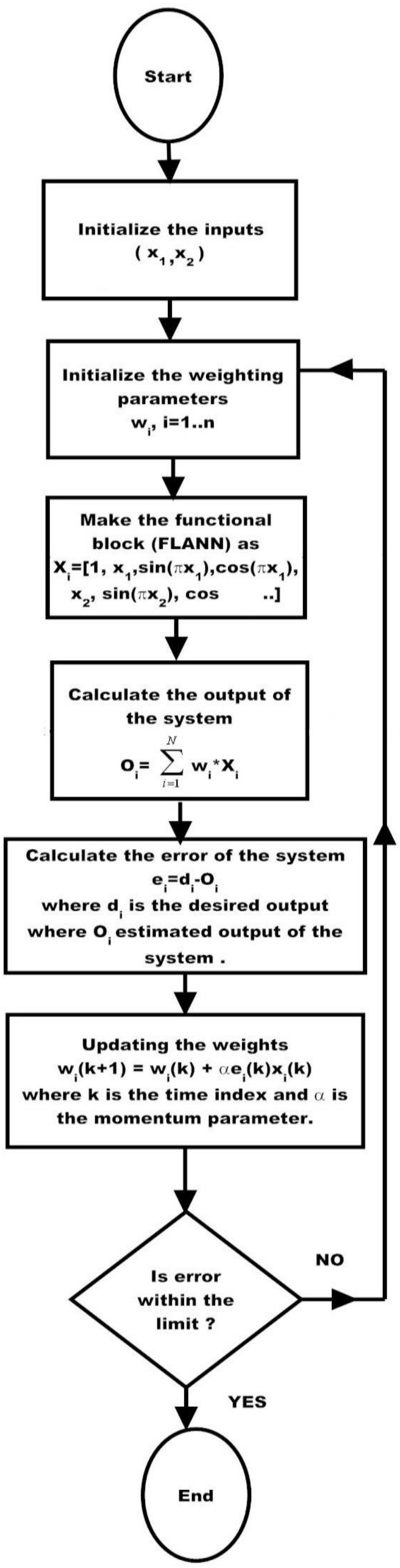

Start
Initialize the inputs (x1,x2)
Initialize the weighting parameters wi, i=1..n
Make the functional block (FLANN) as Xi=[1, x1,sin(πx1),cos(πx1), x2, sin(πx2), cos ..]
Calculate the output of the system Oi= Σ wi*Xi
Calculate the error of the system ei=di-Oi where di is the desired output where Oi estimated output of the system .
Updating the weights wi(k+1) = wi(k) + αei(k)xi(k) where k is the time index and α is the momentum parameter.
Is error within the limit ?
NO
YES
End

Fig. 4: Flowchart of FLANN based Inverse Kinematic elucidation

IV. Result and Disscussion

To solve the Inverse Kinematics, here two types of ANN methodologies (MLP and FLANN) are adopted. This entire simulation is done using the MATLAB simulation environment. This proposed system was simulated with the help of Pentium Core due Processor 3.2 GHz CPU, 2 GB based RAM and 300 GB storage capacity based Personal Computer. Using the mathematical relationship, both the model (MLP and FLANN) were applied to predict the joint angles of the manipulator using inverse kinematics equation. Let the analysis of MLP is discuss first. For 2D planner, 2-20-2 based MLP structure was chosen, where two number of inputs (*CARTESIAN COORDINATE (X AND Y)*), twenty numbers of hidden nodes in the hidden-layer and two number of output in the systems. Fig. 5 represents the Mean square error plot for θ_1, where Fig. 6 represents the testing result of the model. The mean square errors of the proposed system were calculated as follows:

$$MSE = \frac{1}{n}\left(\sum_{i=1}^{N} Desired_i - Eastimated_i\right)^2$$

Where n is the total number of samples of the proposed system.

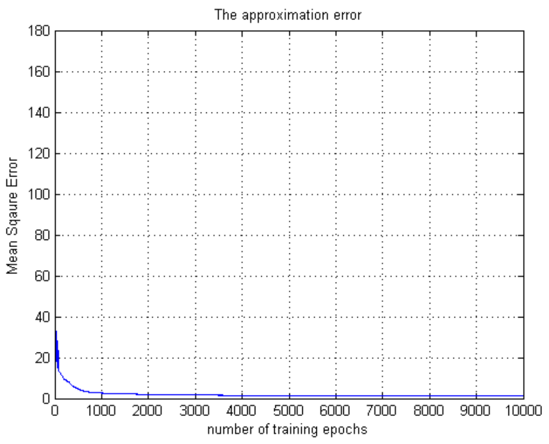


Fig. 5: MSE of Multi Layer Perceptron (2-20-2) for θ_1

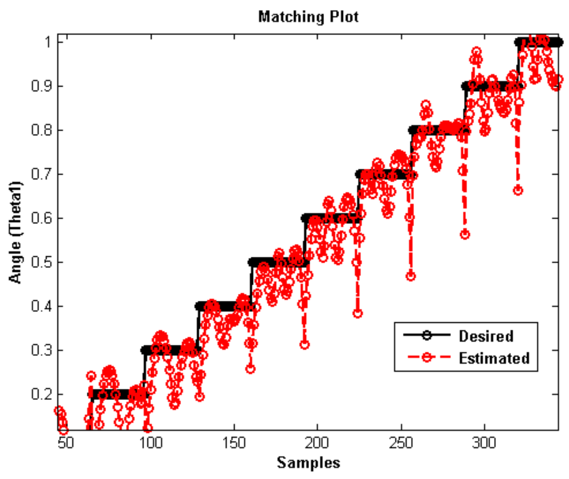


Fig. 6: Testing Result of Multi Layer Perceptron (2-20-2) for θ_1

Fig. 7 represents the Mean square error plot of MLP system for θ_2, where Fig. 8 represents the testing result of the model.

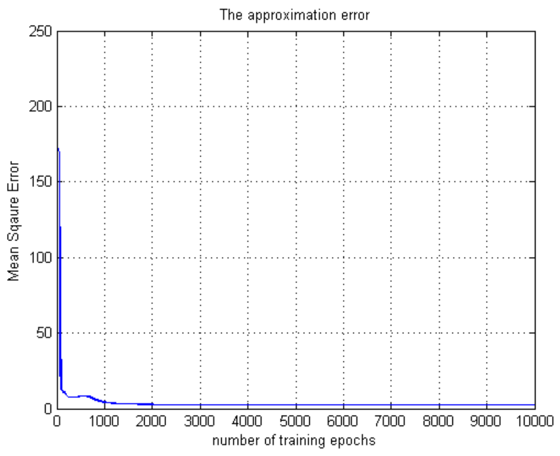


Fig. 7: MSE of Multi Layer Perceptron (2-20-2) for θ_2

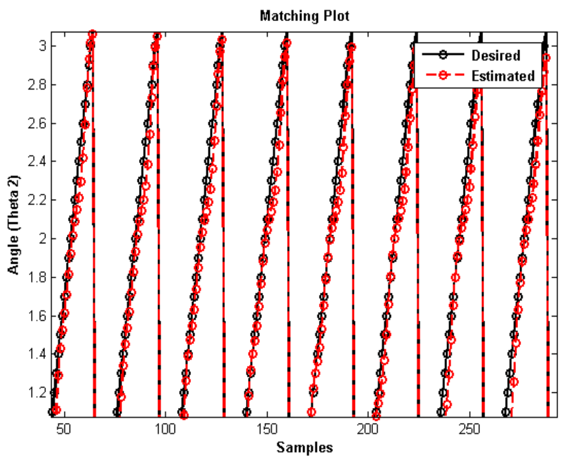


Fig. 8: Testing Result of Multi Layer Perceptron (2-20-2) for θ_2

Like MLP, for 2D planner, 2-7-2 based FLANN structure was chosen, where two number of inputs (CARTESIAN COORDINATES), seven numbers of hidden nodes in the hidden-layer (seven number of expansion) and two number of output in the systems. Fig. 9 represents the Mean square error plot for θ_1, where Fig. 10 represents the testing result of the model.

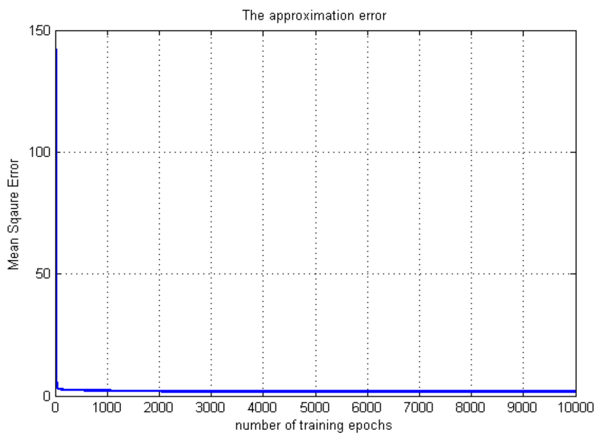


Fig. 9: MSE of FLANN (2-7-2) for θ_1

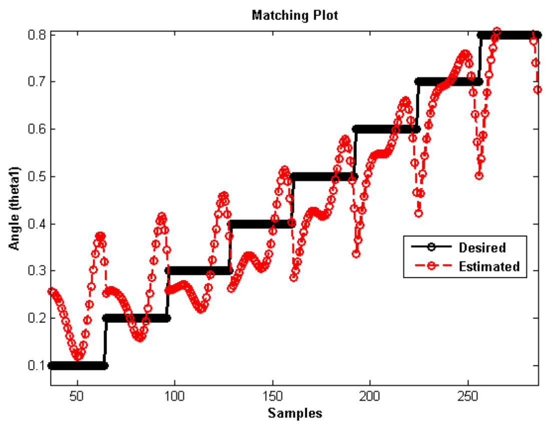


Fig. 10: Testing Result of FLANN (2-7-2) for θ_1

Fig. 11 represents the Mean square error plot of FLANN system or θ_2, where Fig. 12 represents the testing result of the model.

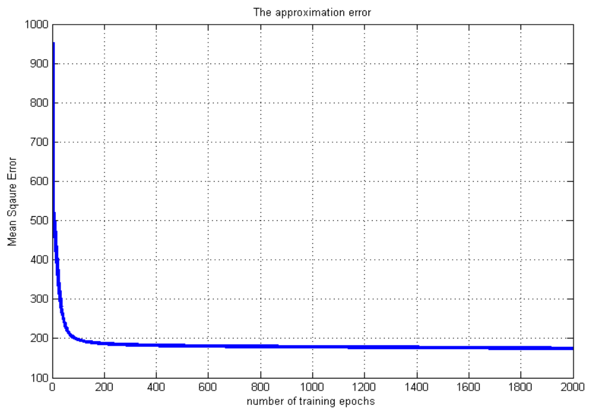


Fig. 11: MSE of FLANN (2-7-2) for θ_2

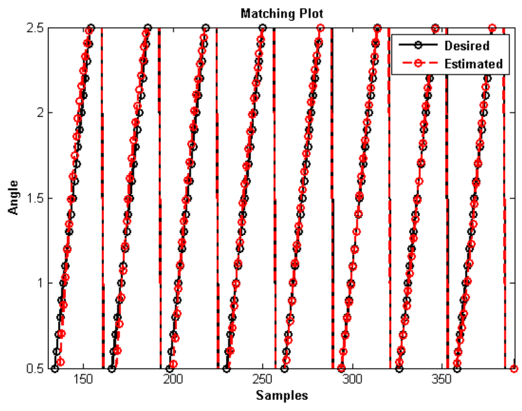


Fig. 12: Testing Result of FLANN (2-7-2) for θ_2

Result analysis of 3-D Planner

Similar to 2-D Planner, Using the mathematical relationship, both the model (MLP and FLANN) were applied to predict the joint angles of the manipulator using inverse kinematics equation. Let the analysis of MLP is discuss first. For 3D planner, 2-20-3 based MLP structure was chosen, where two number of inputs (CARTESIAN COORDINATES), twenty numbers of hidden nodes in the hidden-layer and three number of output in the systems. Fig. 13 represents the Mean square error plot for θ_1, where Fig. 14 represents the testing result of the model.

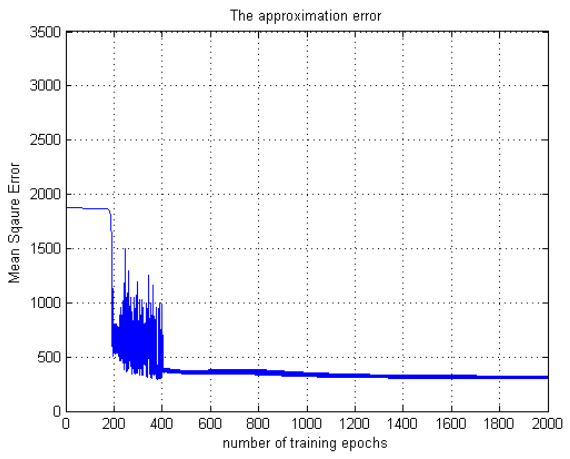


Fig. 13: MSE of Multi Layer Perceptron (2-20-3) for θ_1

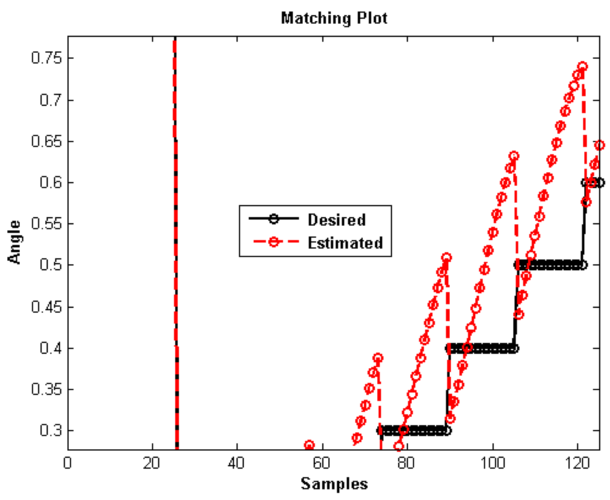


Fig. 14: Testing Result of Multi Layer Perceptron (2-20-3) for θ_1

Fig. 15 represents the Mean square error plot of MLP system for θ_2, where Fig. 16 represents the testing result of the model.

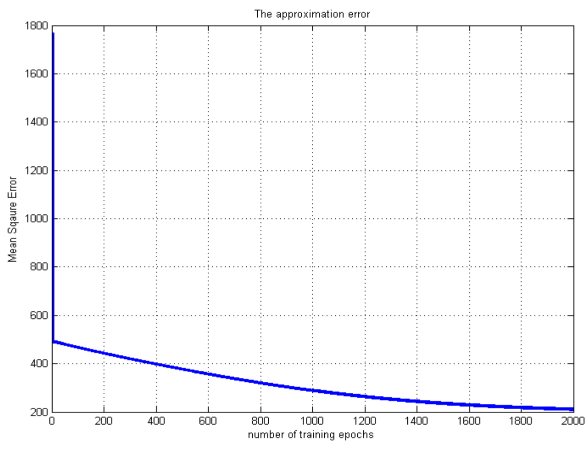


Fig. 15: MSE of Multi Layer Perceptron (2-20-3) for θ_2

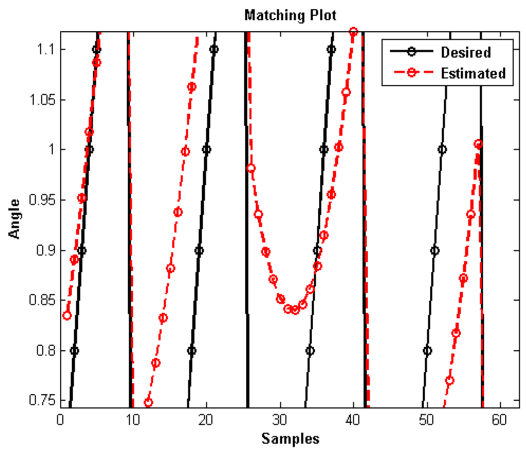


Fig. 16: Testing Result of Multi Layer Perceptron (2-20-3) for θ_2

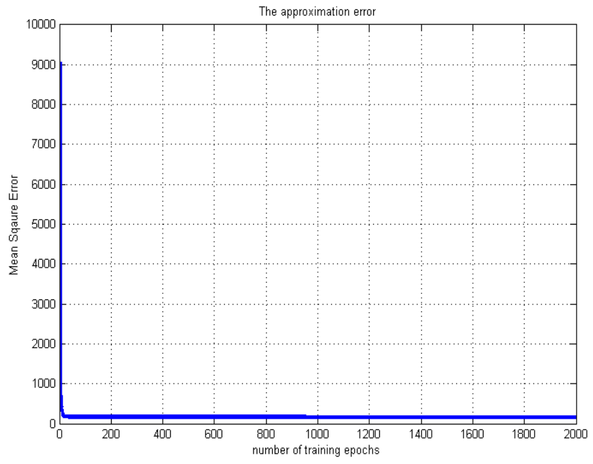


Fig. 17: represents the Mean square error plot of MLP system for θ_3, where Fig. 18 represents the testing result of the model.

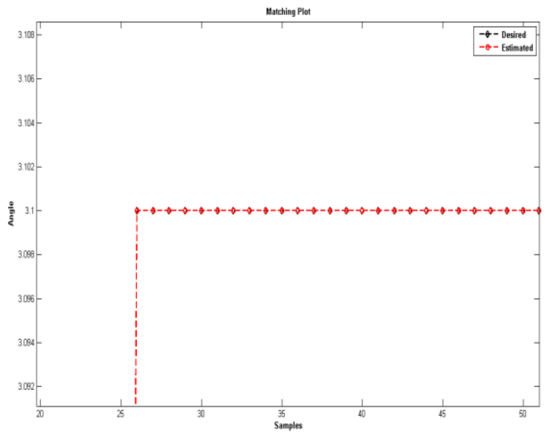


Fig.18: Testing Result of Multi Layer Perceptron (2-20-3) for θ_3

Like MLP, for 3D planner, 2-7-3 based FLANN structure was chosen, where two number of inputs (CARTESIAN COORDINATES), seven numbers of hidden nodes in the hidden-layer (seven number of expansion) and three number of output in the systems. Fig. 19 represents the Mean square error plot for θ_1, where Fig. 20 represents the testing result of the model.

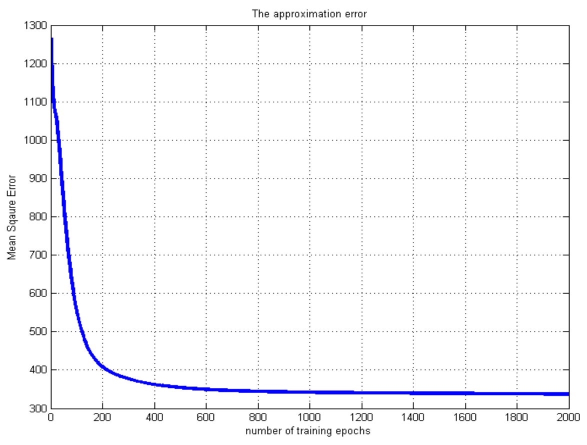


Fig. 19: MSE of FLANN (2-7-3) for θ_1

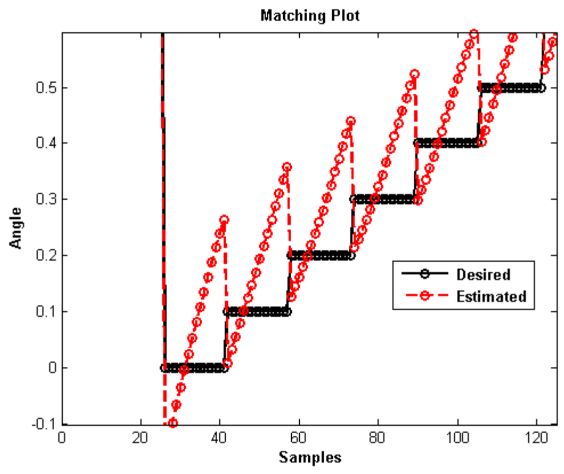


Fig. 20: Testing Result of FLANN (2-7-3) for θ_1

Fig. 21 represents the Mean square error plot of FLANN system for θ_2, where Fig. 22 represents the testing result of the model.

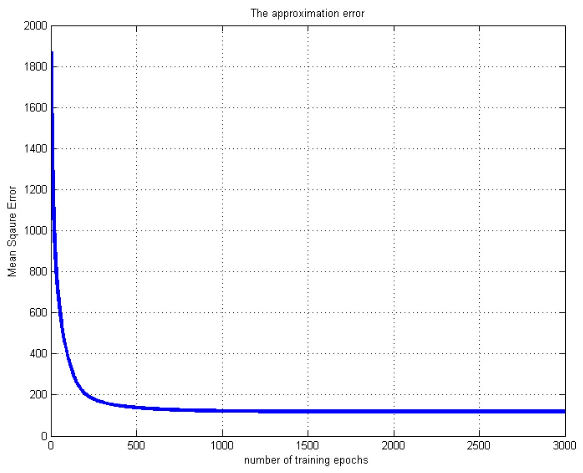


Fig. 21: MSE of FLANN (2-7-3) for θ_2

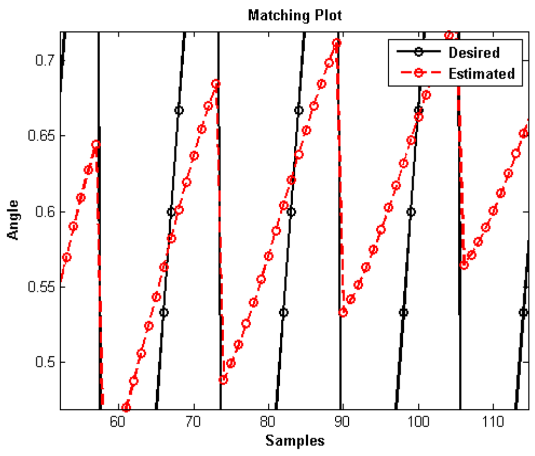


Fig. 22: Testing Result of FLANN (2-7-3) for θ_2

Fig. 23 represents the Mean square error plot of FLANN system for θ_3, where Fig. 24 represents the testing result of the model.

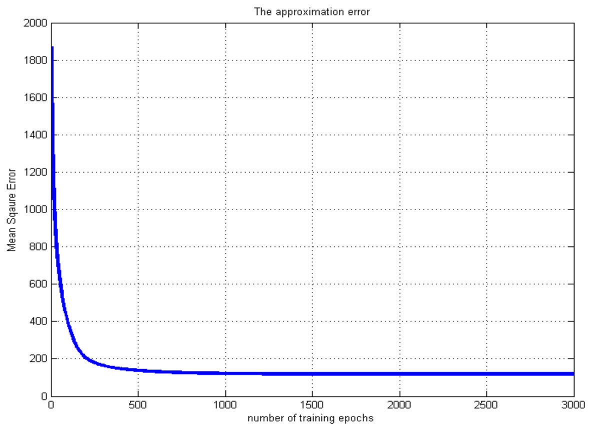


Fig. 23: MSE of FLANN (2-7-3) for θ_3

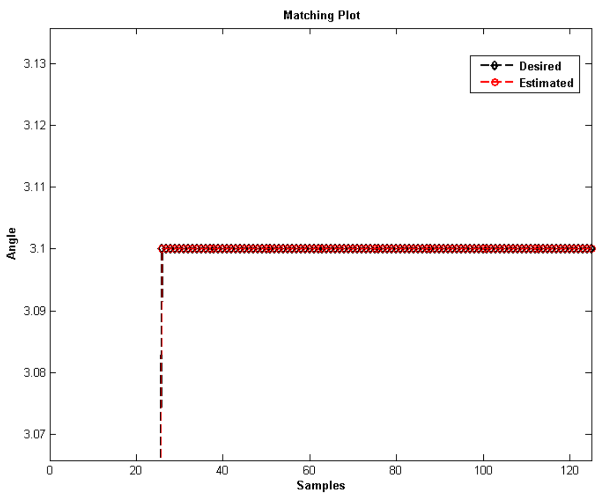


Fig. 24: Testing Result of FLANN (2-7-3) for θ_3

To find the performance analysis of the both the proposed model, CPU time comparison was done. Table 1 represents the performance analysis of the proposed system. The performances of the both models were compared by considering the CPU time.

Table 1: Performance analysis of the proposed model (MLP and FLANN)

Type of the structure	Architecture	Applications	CPU time in sec.
2-20-2	MLP	2-D Planner	15.2389
2-7-2	FLANN	2-D Planner	2.8411
2-20-3	MLP	3-D Planner	46.2173
2-7-3	FLANN	3-D Planner	8.2745

V. Conclusion

In this proposed research work two types of ANN architecture were applied to solve a complex problem of Industrial Manipulator. In this problem MLP and FLANN models were successfully implemented for both 2-D and 3-D planar. Categorically the proposed research problem can be alienated into two steps;

In the first step, using MATLAB Simulation Environment, the mathematical solution for both 2-D and 3-D Planar were executed and summarized. It was observed that the mathematical models are very complex with large number of computational load (Multiplication, Addition, Subtraction, Division) and therefore to reduce this complexity, here MLP and FLANN systems were proposed.

After the simulation it was found that the FLANN structure has very less CPU-time, with comparison to MLP system. It was observed that the proposed FLANN system was 5 times faster than MLP system and it can easily be implemented in hardware.

Acknowledgement

The authors like to thank to **Prof. (Dr.) Radhakanta Mishra**, Chairman of Eastern Academy of Science and Technology for his guidance, valuable comments, giving research facilities and words of encouragement. His motivation and guidance helped to improve the clarity of this paper.

References

[1] Robert J. Schilling, Fundamentals of Robotics: Analysis and Control, Prentice Hall, New Jersey, 1990.

[2] John J. Craig, Introduction to Robotics: Mechanics and Control, Prentice Hall, New Jersey, 2004.

[3] Santosh Kumar Nanda, Ashok Kumar Dash, Sandigdha Acharya, Abikshyana Moharana. Application of Robotics in Mining Industry: A

Critical Review, Mining Technology-Extraction, Beneficiation for Safe & Sustainable Development, Indian Mining & Engg. Journal, Mine TECH'10, pp.108-112, 2010.

[4] Christian Smith, Henrik I. Christensen. Robot Manipulator, IEEE Robot and Automation Magazine, vol. 64, 2009, pp 75-83.

[5] A. S. Morris, A. Mansor. Finding the Inverse Kinematics of Manipulator Arm using Artificial Neural Network with Lookup Table, Robotica, vol. 15,1997, pp. 617–625.

[6] S Alavandar, M.J. Nigam. Neuro-Fuzzy based Approach for Inverse Kinematics Solution of Industrial Robot Manipulators, Int. J. of Computers, Communications & Control, vol. 3, 2008, pp 224-234.

[7] Jolly Shah, S.S. Rattan, B.C. Nakra. Kinematics Analysis of 2-DOF Planar Robot Using Artificial Neural Network", World Academy of Science, Engineering and Technology, vol. 81, 2011, pp. 282-285.

[8] Simon Haykin. Neural Networks: A Comprehensive Foundation, Prentice Hall, 1998

[9] Madan Gupta, Liang. M, Jin, Noriyasu Homma. Static and Dynamic Neural Networks: From Fundamentals to Advanced Theory, Wiley-IEEE Press, USA, 2003.

[10] Allon Guez, James Ellbert, M. L. Kam. Neural networks architecture for control, IEEE Control System magazine, 1998, pp.22-25.

[11] Y. –H Pao. Adaptive pattern recognition and neural networks, Addison-Wesley, Reading, Massachusetts, 1989

[12] S.K. Nanda, D.P. Tripathy, S.K. Patra. Application of Soft Computing for noise prediction model for opencast mines, Noise Control Engineering Journal, USA, vol.59, issue 5, 2011, pp. 432-446.

[13] S.K. Nanda, D.P. Tripathy. Application of Functional Link Artificial Neural Network for Noise Prediction in Mining Industry, International Journal of Advances of Fuzzy Logic System, USA. http:// dx.doi.org/10.1155/2011/831261

[14] S.K. Nanda, D.P. Tripathy. Application of Legendre Neural Network for Air Quality Prediction, ICET-2011, The 5th PSU-UNS International Conference on Engineering and Technology (ICET2011), Phuket, May 2-3, Malaysia, 2011, pp. 267-272.

[15] J. C. Patra, R. N. Pal. Functional link artificial neural network-based adaptive channel equalization of nonlinear channels with QAM signal," in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, vol. 3, 1995, pp.2081–2086.

[16] J. C. Patra, R. N. Pal, R. Baliarsingh, G. Panda. Nonlinear channel equalization for QAM signal constellation using artificial neural networks, IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, vol.29 issue 2, 1999, pp. 262–271.

Santosh Kumar Nanda served as Professor in Computer Science and Engineering Department of Eastern Academy of Science & Technology (EAST). His research interests are Soft Computing, Artificial Intelligence, Image Processing, Prediction of machinery noise and vibration, Noise and vibration control, Mathematical modelling, Pattern Recognition. He has more than 50 research articles in reputed International Journals and International conferences etc. He is now **Editor in Chief** of International Journal of Logic and Computation (IJLP, CSC Journal Press, and Malaysia). He is now Editor of International Journal of Computer Application (IJCA), International Journal of Advancements in Computing Technology (IJACT), International Journal of Computer Applications (IJCA, Acta Press). International Journal of Open Problems in Computer Science and Mathematics (IJOPCM). Recently his name was selected for **Editor in Chief** for Asian Journal of Information Management, USA. In addition to this, his name was also selected for **Regional Editor** of International Journal of Applied Science, **Regional Editor** of International Journal of Artificial Intelligence, **Regional Editor** of International Research Journal of Information Technology and Regional Editor of Trends in Applied Science Research. He is also International Program Committee of 16th Online World Conference on Soft Computing in Industrial Applications (WSC16) 2011" and "15th Online World Conference on Soft Computing in Industrial Applications (WSC15) 2010". He is also acting as reviewers in many reputed International Journals. Currently he is an Individual Member in International Rough Set Society and Member of International Association of Engineers (IAENG). Currently his name was selected for Marquis Who's and Who 2011 and 2012.

Swetalina Panda currently continuing her Bachelor of Technology in Computer Science and Engineering in Department of Computer Science and Engineering of Eastern Academy of Science & Technology (EAST). Her research interests are Soft Computing, Artificial Intelligence, Robotics and Machine Vision. She has also active in departmental activities, quiz, seminar competitions etc.

P Raj Sekhar Subudhi currently continuing his Bachelor of Engineering in Computer Science and Technology in Department of Computer Science and Engineering of Eastern Academy of Science & Technology (EAST). His research interests are Soft Computing, Artificial Intelligence, Robotics , Machine Vision. He has also active in departmental activities, seminar competitions etc.

Ranjan Kumar Das served as Senior Lecturer in Computer Science and Engineering Department of Eastern Academy of Science & Technology (EAST). His research interests are Soft Computing, Artificial Intelligence, Robotics, Image Processing, Distributed Computing. He has published 3 research articles in International Conferences and Journals.

How to cite this paper: Santosh Kumar Nanda,Swetalina Panda,P Raj Sekhar Subudhi,Ranjan Kumar Das,"A Novel Application of Artificial Neural Network for the Solution of Inverse Kinematics Controls of Robotic Manipulators", International Journal of Intelligent Systems and Applications(IJISA), vol.4, no.9, pp.81-91, 2012. DOI: 10.5815/ijisa.2012.09.11