ISSN: 0974-2115
www.jchps.com Journal of Chemical and Pharmaceutical Sciences

Inverse Kinematics Solution of a Five Joint Robot Using NARX Algorithm

I. Jacob Raglend[1*], M. Dev Anand[2], G. Glan Devadhas[3], D. M. Mary Synthia Regis Prabha[1]
[1]Department of Electrical and Electronics Engineering, Noorul Islam Centre for Higher Education, Kumaracoil - 629 180, Thuckalay, Kanyakumari District, Tamilnadu State, India
[2]Department of Mechanical Engineering, Noorul Islam Centre for Higher Education, Kumaracoil - 629 180, Thuckalay, Kanyakumari District, Tamilnadu State, India
[3]Department of Electronics and Instrumentation Engineering, Vimal Jyothi Engineering College, Chemperi, Kannur, Kerala
***Corresponding author: E-Mail: jacobraglend@rediffmail.com**

ABSTRACT
One of the significant problem in robot kinematics is to optimising the solution of inverse kinematics which deals with obtaining the joint variables in terms of the end-effector position and orientation and is difficult than the forward kinematics problem. As the degree of freedom of a robot increases the inverse kinematics calculation become more difficult and expensive. In industrial and manufacturing field the use of robot is an inevitable factor and thus the motion of its manipulator is necessary to express more efficiently and simply. Because of this reason kinematical approach is considered here. The Kinematics is the analytical study of geometry of motion of a robot arm and is of two types, Direct and Inverse Kinematics. Here the Neural Network approach is adopted and the algorithm employed is Nonlinear Autoregressive Network (NARX) Neural Network. This paper proposes neural network architecture to optimise the inverse kinematics solution. In this paper the Neural Network idea NARX algorithm is proposed to solve the inverse kinematics problem of a five degree of freedom robot end effector. This technique causes a decrease in the difficulty and calculations faced when using the traditional methods in robotics. Thus the optimised output is evaluated to ensure the efficiency of this approach.

KEY WORDS: Degrees of Freedom, Inverse Kinematics, NARX algorithm.

1. INTRODUCTION

Nowadays robots are considered as an indispensable part of modern manufacturing field with their inherent capability of executing complex and risky jobs more efficiently and reliably. A robotic manipulator is composed of several links connected together through joints. Kinematics deals with the geometric motion of a robotic manipulator and the inverse kinematics is considered as the most popular and efficient method of controlling robot arm. Figure 1 gives a view of the general structure of a series manipulator with revolute joints (5 DOF). Figure 2 shows a five Degree of Freedom joints related to waist, shoulder, elbow, pitch and roll.

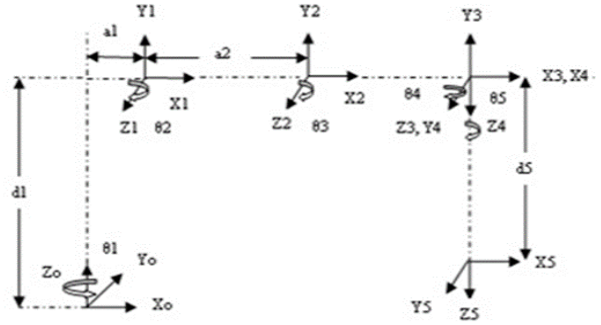

Fig.1. Five Joint Manipulator Arm Links and Joints

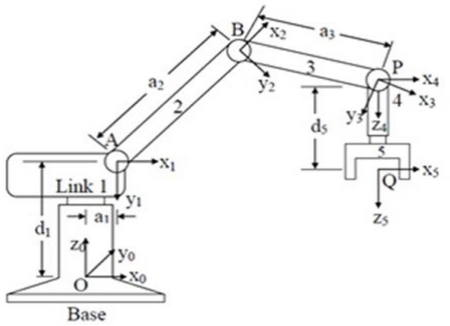

Fig.2. A Five Joint Robot Arm

The proposed approach is a strategy that can be implemented to solve the inverse kinematics problems faced in robotics with highest DOF more efficiently.

Researchers proposed a Neuro-Genetic Approach to determine the inverse kinematics solution of robotic manipulators. The proposed solution method is based on using Neural Networks (NN) and Genetic Algorithms (GA) in a hybrid system. Here an Elman NN as well as GA ideas are implemented. The error introduced by the NN can be minimised by the application of GA. The main problems are the test errors and learning time is larger and it requires large number of hidden neurons.

Adrian-Vasile Duka (2011), proposed a Novel Approach on NN Based inverse kinematics solution for trajectory tracking of a robotic arm. It employs the conventional feed forward NN. By using this idea the desired trajectories can be generated by solving inverse kinematics problem. Since it employs the conventional method it performs simple operation.

Nonlinear autoregressive models with exogenous inputs (NARX) recurrent neural as opposed to other recurrent neural models, have limited feedback architectures that come only from the output neuron instead of from hidden neurons. The NARX models are commonly used in the system of identification area. All the specific dynamic networks discussed so far have either been focused networks, with the dynamics only at the input layer, or feed forward networks. The nonlinear autoregressive network with exogenous inputs (NARX) is a recurrent dynamic

network, with feedback connections enclosing several layers of the network. The NARX model is based on the linear ARX model, which is commonly used in time series modeling.

Daniela Tarnitaa (2013), presented an analysis of a hand arm system. The algorithm is based on the calculation of largest Lyapunov function. By calculating theLLE thestability of a dynamical system can be determined. But the calculation of Lyapunov exponent is very complex. Researchers proposed an approach using NN architecture for inverse kinematics problem in robotics. The NN utilized a Multi-Layered Perceptron (MLP) training algorithm using back-propagation. Complexity of the algorithm will be reduced. The applied traditional algorithm is not much effective for complex structures.

ANNs' based inverse kinematics solution for serial robot manipulators passing through singularities was given in. This paper proposes the back propagation algorithm with sigmoid function as an activation function. Since it is a simpler algorithm, it is very easy in calculation and implementation. But it does not have the ability to learn huge number of patterns thus it is limited to small number of data patterns and the error percentage is higher.

Researchers introduced a developmental approach to robotic pointing via human–robot interaction. MRAN technique is employed which has an incremental feature that fits the developmental robot very well. It works by first applying a reinforcement learning algorithm to guide the robot to create attempt movements towards a salient object that is out of the robot's initial reachable space. But it has only less accuracy.

To optimize the joint angles of robotics manipulator using GA was given. The system would adopt the advantage of GA to optimize its performance in terms of path control and accuracy. Once a path is being generated and given as input to the robot, the manipulator's end tip moves along that specified path. Thus parameters can be found with high accuracy even with low resolution encoders. But it is a time consuming procedure which is a disadvantage. Similarly the Identification of time-varying nonlinear systems using minimal Radial Basis Function NNs (RBFNN) was given. An identification algorithm for time varying nonlinear systems using a sequential learning scheme with a minimal RBFNN is presented. The learning algorithm combines the growth criterion of the resource allocating network with a pruning strategy.

Ramirez, and Rubiano (2011), proposed an optimization of inverse kinematics of a 3R robotic manipulator using GAs. Under certain conditions, GA is not appropriate to solve inverse kinematics problems in fast and accurate way since it has a higher response time.

Researchers proposed antrajectory planning for the planar redundant manipulator. The goal is to minimize the sum of the end-effector position error at each intermediate point along the trajectory which makes the end-effector to track the prescribed trajectory accurately for a 3DOF planar manipulator with different end-effector trajectories have been carried out.

This paper proposes the neural network idea Minimum Resource Allocation Network algorithm, and here the algorithms undergo the growing as well as the pruning method to train the network.

Forward kinematic model of Scorbot-ER Vu Plus industrial robot: The answer for the forward kinematics issue comprises of discovering the estimation of the extreme location of TCP. This result is a capacity of 5 joint qualities, and D-H parameters. There are a few techniques to intensify this issue. This research is carried out utilizing the homogeneous conversion matrices technique, and the D-H's deliberate representation of the reference frameworks. Despite the fact that the last position might be discovered geometrically, the technique proposed in this work offers a reaction which could compare the location of the extreme of each connection in the kinematics network, contrasted with the past or the worldwide standard framework, so as to characterize the position of every explanation in the robot.

Frame Assignment and Structure: The joints of the mechanical arm of the Scorbot-ER Vu Plus Robot are identified. The D-H parameters as indicated by this model are shown in Table 1. The kinematics model is demonstrated in, with the edge assignments as per the D-H documentations.

Table.1. D-H Parameters for the Scorbot- ER Vu Plus Robot

Joint i	α_i	a_i	d_i	θ_i	Operating Range
1	$-\pi/2$	$a_1=10$	$d_1=5$	$\theta_1=30°$	310°
2	0	$a_2=15$	$d_2=0$	$\theta_2=45°$	+130° / – 35°
3	0	$a_3=20$	$d_3=0$	$\theta_3=60°$	±130°
4	$-\pi/2$	$a_4=0$	$d_4=0$	$\theta_4=50°$	±130°
5	0	$a_5=0$	$d_5=5$	$\theta_5=70°$	±570°

Denavit-Hartenberg Representation: The D-H matrix is a special form of a homogeneous transformation matrix, a 4x4 matrix, having the property of transforming a vector from one coordinate frame to another, by means of a translation or rotation. For a kinematic chain with *n*-joints and *n* - 1-links, every joint is assigned a frame of reference. Thus, each joint can be represented by a homogeneous transformation matrix, describing the particular rotation or translation needed to align the i^{n-1th} joint with the i^{th} joint. The product of these matrices gives the final position of the n^{th} joint. Additionally, it proposed a methodical documentation for allotting the right united ortho-normal correlate

skeleton, every one connection in a chain of open kinematic connections. When these connections appended direction frames are doled out, the conversions between neighbouring coordinate frames could be spoken to by a solitary standard 4x4 homogeneous coordinate transformation matrix. The coordinates are allotted to the connections utilizing the accompanying methodology.

a) Joints number from 1 to n, beginning with the base and finishing with the device yaw, pitch, and roll, in a specific order.

b) Allot coordinate frame L_0 of right-handed ortho-normal to the robot, verifying that z° adjusts to the axis of joint 1. Set k = 1.

c) Adjust the axis of joint k + 1 with z^k.

d) Locate the source of L_k at the crossing point of the z^k and z^{k-1} axis. On the off chance that they don't converge, use convergence of z^k with a typical ordinary in the middle of z^k and z^{k-1}.

e) Choose x^k to be orthogonal to z^k and z^{k-1} both. In the event that z^k and z^{k-1} are parallel, point x^k far from z^{k-1}.

f) Choose y^k to structure an ortho-normal coordinate outline L_k.

g) Make k = k + 1. In the event that k < n, go to step 2; else, proceed.

h) Make the starting point of L_n at the device tip. Adjust z^n to the methodology vector, y^n the sliding vector, and x^n with the typical vector of the tool. Set k = 1.

i) Allocate point b^k at the convergence of the x^k and Z^{k-1} pivot. In the event that they don't cross, utilize the crossing point of x^k with a typical ordinary in the middle of x^k and z^{k-1}.

j) Find θ^k as the angle of turn from x^{k-1} to x^k measured about z^{k-1}.

k) Find d_k as the distance from the beginning of frame L_{k-1} to point b^k, measured along z^{k-1}.

l) Find a_k as the distance from point b^k to the beginning of frame L_k, measured along x^k.

m) Find α_k as the angle of turn from z^{k-1} to z^k measured about x^k.

n) Set k = k + 1. In the event that k <= n, go to step 8; else, stop.

Transformation Matrix: In the wake of creating the D-H coordinate framework for every connection, a similar matrix of transformation can without much of a stretch be produced, bearing in mind body{i-1} and body{i} change comprising of four essential conversions. The general complex homogeneous matrix of transformation can be shaped by sequential applications of basic changes. This transformation comprises of four essential conversions.

T1: Angle θ_i for z_{i-1} axis of rotation, T2: Distance d_i for z_{i-1} axis of translation, T3: Distance a_i along x_i axis of translation and, T4: Angle α_i about x_i axis of rotation.

Taking into account the D-H gathering, the transformation matrix from joint *i* to joint *i+1* is prearranged by:

$$^{i-1}T_i = \begin{bmatrix} C\theta i & -S\theta i C\alpha i & S\theta i S\alpha i & aiC\theta i \\ S\theta i & C\theta i C\alpha i & -C\theta i S\alpha i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

Where $S\theta_i = \sin\theta_i$, $C\theta_i = \cos\theta_i$, $S\alpha_i = \sin\alpha_i$, $C\alpha_i = \cos\alpha_i$. The Overall Transformation Matrix,

$$^0T_5 = {}^0T_1 * {}^1T_2 * {}^2T_3 * {}^3T_4 * {}^4T_5 \quad (2)$$

$$^0T_1 = \begin{bmatrix} C_1 & 0 & -S_1 & a_1C_1 \\ S_1 & 0 & C_1 & a_1S_1 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Substitute $\theta_1=30°$, $\alpha_1=-\frac{\pi}{2}$, $a_1=10$, $d_1=5$ (3)

$$^0T_1 = \begin{bmatrix} 0.866 & 0 & -0.5 & 8.660 \\ 0.5 & 0 & 0.866 & 5 \\ 0 & -1 & 0 & 5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$$^1T_2 = \begin{bmatrix} C_2 & -S_2 & 0 & a_2C_2 \\ S_2 & C_2 & 0 & a_2S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Substitute $\theta_2=45°$, $\alpha_2=0$, $a_2=15$, $d_2=0$ (5)

$$^1T_2 = \begin{bmatrix} 0.7071 & -0.7071 & 0 & 10.6066 \\ 0.7071 & 0.7071 & 0 & 10.6066 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6)$$

$$^2T_3 = \begin{bmatrix} C_3 & -S_3 & 0 & a_3C_3 \\ S_3 & C_3 & 0 & a_3S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Substitute θ_3=60°, α_3=0, a_3=20, d_3=0 (7)

$$^{2}T_{3} = \begin{bmatrix} 0.5 & -0.8660 & 0 & 10 \\ 0.8660 & 0.5 & 0 & 17.3205 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

$$^{3}T_{4}=\begin{bmatrix} C_4 & 0 & S_4 & 0 \\ S_4 & 0 & -C_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Substitute θ_4=50°, $\alpha_4=-\frac{\pi}{2}$, a_4=0, d_4=0 (9)

$$^{3}T_{4} = \begin{bmatrix} 0.6428 & 0 & -0.7660 & 0 \\ 0.7660 & 0 & 0.6428 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (10)$$

$$^{4}T_{5} = \begin{bmatrix} C_5 & -S_5 & 0 & 0 \\ S_5 & C_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Substitute θ_5=70°, α_5=0, a_5=0, d_5=5 (11)

$$^{4}T_{5} = \begin{bmatrix} 0.3420 & -0.9397 & 0 & 0 \\ 0.9397 & 0.3420 & 0 & 0 \\ 0 & 0 & 1 & 5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (12)$$

We know that,

$$^{0}T_{2}=^{0}T_{1}*^{1}T_{2}=\begin{bmatrix} 0.6123 & -0.6123 & -0.5 & 17.8453 \\ 0.3536 & -0.3536 & 0.866 & 10.3033 \\ -0.7071 & -0.7071 & 0 & -5.6066 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (13)$$

$$^{0}T_{3}=^{0}T_{2}*^{2}T_{3}=\begin{bmatrix} -0.2241 & -0.8364 & -0.5 & 13.3630 \\ -0.1294 & -0.4830 & 0.866 & 7.7148 \\ -0.9659 & 0.2587 & 0 & -24.9249 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (14)$$

$$^{0}T_{4}=^{0}T_{3}*^{3}T_{4}=\begin{bmatrix} -0.7847 & 0.5 & -0.3660 & 13.3630 \\ -0.4532 & -0.8660 & -0.2114 & 7.7148 \\ -0.4227 & 0 & 0.9062 & -24.9249 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (15)$$

$$^{0}T_{5}=T_{e}=^{0}T_{4}*^{4}T_{5}=\begin{bmatrix} 0.2015 & 0.9084 & -0.3660 & 11.533 \\ -0.9688 & 0.1297 & -0.2114 & 6.6578 \\ -0.1446 & 0.3972 & 0.9062 & -20.3939 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (16)$$

Verification of the Model by MATLAB: MATLAB is a capable environment for direct arithmetical and graphical depiction that is accessible on an extensive variety of machine stages. The center usefulness could be stretched out by application particular toolboxes. The Simulink tool kit gives numerous capacities that are needed in robotics, and locations ranges, for example, kinematics, dynamics, and path creation. This Tool is valuable for dissection and also investigating results from tries different things with true robots, and could be an influential device for instruction. The Toolbox is focused around an extremely common system for speaking to the kinematics and dynamics of serial-connection controllers, by depiction matrices. These involve, in the least difficult case, the D-H parameters of the robot, and could be made by the client for any serial-link controller. The controller portrayal could be explained, by increasing the matrix, to incorporate connection inertial, and motor inertial, and frictional parameters. Such frameworks give a succinct method for depicting a robot model, and may encourage the imparting of robot models over the exploration group. This would permit the simulation results to be looked at in a considerably more compelling path than is right now done in the writing. The Toolbox additionally gives capacities to controlling information sorts, for example, vectors, homogeneous changes and unit-quaternion, which are important to speak to a 3D position and introduction. The schedules are by and large composed in a direct, or course book, way for pedagogical reasons, instead of for most extreme computational proficiency. With the forward and reverse kinematics for a controller tackled, these equations could be utilized to discover the inputs (joint angles) important to plot the robot. A trajectory could be characterized regarding a tool space move, say, a consecutive-line move of the end-effector, and afterward invigorate the robot over this move. A case is indicated underneath, with a simulated

screen of the MATLAB system yield. Figure 3 speaks to the MATLAB result yield document for the forward kinematic.

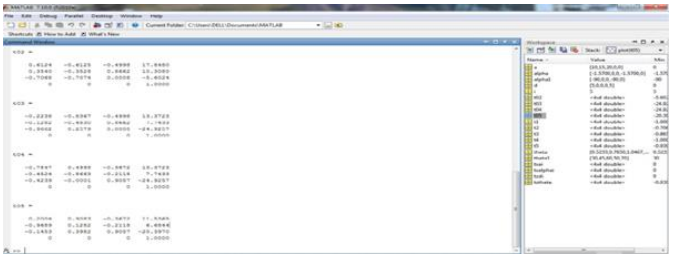

Fig. 3. MATLAB Result for the Forward Kinematic

Knowing the following values:

a_1 = 10 mm; a_2 = 15 mm; a_3 = 20 mm; d_1 = 5 mm; d_5 = 5 mm; $\theta_1 = 30^0$; $\theta_2 = 45^0$; $\theta_3 = 60^0$; $\theta_4 = 50^0$; $\theta_5 = 70^0$;

Case Study: (Mathematical Solution)

$$^0T_5=\begin{bmatrix} 0.2015 & 0.9084 & -0.3660 & 11.5330 \\ -0.9688 & 0.1297 & -0.2114 & 6.6578 \\ -0.1446 & 0.3972 & 0.9062 & -20.3939 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 \end{bmatrix}$$

Case-study: (MATLAB Program Output)

The final matrix $_0T^5$result:

$$^0T_5=\begin{bmatrix} 0.2004 & 0.9083 & -0.3672 & 11.5365 \\ -0.9689 & 0.1282 & -0.2118 & 6.6844 \\ -0.1453 & 0.3982 & 0.9057 & -20.3970 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 \end{bmatrix}$$

The Final values of matrix ($_0T^5$) are compared with the physical locations of the robot arm in Table.2.

Table.2. Differences between the Analytical and Physical Values of the Robot (Forward Kinematic)

Position Values	Analytical $_0T^5$Values(mm)	Matlab $_0T^5$Values(mm)	Difference (mm)	Analytical $_0T^5$Values(mm)	Robo cell $_0T^5$Values(mm)	Difference (mm)
P_x	11.5330	11.5365	0.0035	11.5330	11.4665	0.0665
P_y	6.6578	6.6844	0.0266	6.6578	6.6711	0.0133
P_z	-20.3939	-20.3970	0.0031	-20.3939	-20.395	0.0015

Inverse Kinematic Model: The inverse kinematics is also done on the robot. In this stage, the inverse kinematics formulae are generated using the inverse matrices, and a model is validated and verified, using ROBOCELL and MATLAB. This chapter explains how the inverse kinematics Scorbot-ER Vu Plus Robot mathematical model is created effectively.

Inverse Kinematics of Scorbot-ER Vu Plus Industrial Robot Manipulator: By placing the ${}^1_4T$ to the other side of the equation, we can separate θ_1 to make easier for its computation as follows:

$$[{}^0_1T]^{-1}[P\,] = [{}^1_4T] \quad (17)$$

$$[P\,] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (18)$$

The inverse transformation matrix can be computed by using the upcoming matrix.

$$[T]^{-1} = \left(\begin{array}{c|c} R^T & -R^T\ P \\ \hline 0 & 1 \end{array}\right) \quad (19)$$

Applying this to ${}^0_1T$ yields the following result,

$$[{}^0_1T]^{-1} = \begin{bmatrix} C_1 & S_1 & 0 & 0 \\ -S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & -d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (20)$$

Therefore,

$$\begin{bmatrix} C_1 & S_1 & 0 & 0 \\ -S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & -d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_{23} & -S_{23} & 0 & a_1 + a_2C_2 + a_3C_{23} \\ 0 & 0 & 1 & d_2 \\ -S_{23} & -C_{23} & 0 & -a_2S_2 + a_3S_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (21)$$

For compute θ_1 the element (2, 4) in the matrix can be utilized.

$$d_2 = -\sin(\theta_1)\,p_x + \cos(\theta_1)\,p_y \quad (22)$$

Letting,

$p_x=\rho \cos(\phi)$ (23)

$p_y=\rho \sin(\phi)$ (24)

Where, $\rho=\sqrt{p_x^2 + p_y^2}$ (25)

ϕ = Arctan2 (p_y,p_x) (26)

We know that,

$\frac{d_2}{\rho} = -\sin(\theta_1)\cos(\phi) + \cos(\theta_1)\sin\phi$ (27)

Therefore,

$\sin(\phi - \theta_1) = \frac{d_2}{\rho}$ (28)

$\cos(\phi - \theta_1) = \pm\sqrt{1 - \frac{d_2^2}{\rho^2}}$ (29)

$\phi - \theta_1 = \arctan 2\left(\frac{d_2}{\rho}, \pm\sqrt{1 - \frac{d_2^2}{\rho^2}}\right)$ (30)

This leaves the solution for θ_1 as shown in,

$\theta_1 = \arctan 2\ (p_y,p_x) - \arctan 2\left(d_2, \pm\sqrt{p_x^2 + p_y^2 - d_2^2}\right)$ (31)

For compute the joint angle three, we must verify the elements from (1, 4) and (3, 4). First,

$C_1p_x + S_1p_y = a_3C_{23} + a_2C_2 + a_1$ (32)

$p_z - d_1 = -a_3S_{23} - a_2S_2$ (33)

Next by squaring Equation (3.32) and Equation (3.33) followed by addition, θ_3 can be determined.

$(C_1p_x + S_1p_y - a_1)^2 + (d_1 - p_z)^2 = (a_3C_{23} + a_2C_2)^2 + (a_3S_{23} + a_2S_2)^2$ (34)

$(C_1p_x + S_1p_y - a_1)^2 + (d_1 - p_z)^2 = a_2^2 + a_3^2 + 2a_2a_3(C_2C_{23} + S_2S_{23})$ (35)

$(C_1p_x + S_1p_y - a_1)^2 + (d_1 - p_z)^2 = a_2^2 + a_3^2 + 2a_2a_3C_3$ (36)

This leaves Equation (3.37) for θ_3,

$\theta_3 = \pm\arccos\left(\frac{(C_1p_x+S_1p_y-a_1)^2 + (d_1-p_z)^2 - a_2^2-a_3^2}{2a_2a_3}\right)$ (37)

Next by transferring the matrix ${}_2^1T$ onto the other side, next equation can be found that will permit us to find θ_2.

$[{}_2^1T]^{-1}[{}_1^0T]^{-1}[P] = [{}_4^2T]$ (38)

$[{}_2^1T]^{-1} = \begin{bmatrix} C_2 & 0 & -S_2 & -a_1C_2 \\ -S_2 & 0 & -C_2 & a_1S_2 \\ 0 & 1 & 0 & -d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ (39)

$[{}_2^1T]^{-1}[{}_1^0T]^{-1} = \begin{bmatrix} C_2 & 0 & -S_2 & -a_1C_2 \\ -S_2 & 0 & -C_2 & a_1S_2 \\ 0 & 1 & 0 & -d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} C_1 & S_1 & 0 & 0 \\ -S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & -d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ (40)

$[{}_2^1T]^{-1}[{}_1^0T]^{-1} = \begin{bmatrix} C_1C_2 & S_1C_2 & -S_2 & S_2d_1 - a_1C_2 \\ -C_1S_2 & -S_1S_2 & -C_2 & C_2d_1 + a_1S_2 \\ -S_1 & C_1 & 0 & -d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ (41)

$$\begin{bmatrix} C_1C_2 & S_1C_2 & -S_2 & S_2d_1 - a_1C_2 \\ -C_1S_2 & -S_1S_2 & -C_2 & C_2d_1 + a_1S_2 \\ -S_1 & C_1 & 0 & -d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_3 & -S_3 & 0 & a_3C_3 + a_2 \\ S_3 & C_3 & 0 & a_3S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ (42)

By finding elements (1, 4) of the above Equation (42) we stick with the following formula.

$C_1C_2p_x + S_1C_2p_y - S_2p_z + d_1S_2 - C_2a_1 = a_3C_3 + a_2$ (43)

$C_2(C_1p_x + S_1p_y - a_1) + S_2(d_1 - p_z) = a_3C_3 + a_2$ (44)

Substituting $A = (C_1p_x + S_1p_y - a_1)$, $B = (d_1 - p_z)$ and $C = a_3C_3 + a_2$ it is easy to solve θ_2 via reduction to a polynomial.

$C_2A + S_2B = C$ (45)

Where, $C_2 = \frac{1-U^2}{1+U^2}$ (46)

$$S_2 = \frac{2U}{1+U^2} \quad (47)$$

Substituting the above two Equation (46) and Equation (47) into Equation (45) and rearranging, Equation (48) is obtained.

$$(C + A)U^2 - 2UB + (C - A) \quad (48)$$

The quadratic formula rendering can be used to solve this:

$$U = \frac{B \pm \sqrt{B^2 + A^2 - C^2}}{A + C} \quad (49)$$

Where, $\theta_2 = 2\arctan(U)$ (50)

The angle of the 4th joint,θ_4, can also be simply determined, based on θ_2 and θ_3.

$$p_z = p'_z - a_4 \quad (51)$$

$$\theta_4 = 90 - \theta_2 - \theta_3 \quad (52)$$

Replace the z offset in Equation (49), we find the upcoming set of formulae for the inverse kinematics of the Scorbot-ER Vu Plus Robot.

$$\theta_1 = \arctan2\,(p_y, p_x)\ \arctan 2\left(d_2, \pm\sqrt{p_x^2 + p_y^2 - d_2^2}\right) \quad (53)$$

$$\theta_3 = \pm \arccos\left(\frac{(C_1 p_x + S_1 p_y - a_1)^2 + (d_1 - p'_z + a_4)^2 - a_2^2 - a_3^2}{2a_2 a_3}\right) \quad (54)$$

$$\theta_2 = 2\arctan\left(\frac{(d_1 - p'_z + a_4) \pm \sqrt{(d_1 - p'_z + a_4)^2 (C_1 p_x + S_1 p_y - a_1)^2 - (a_3 C_3 + a_2)^2}}{(C_1 p_x + S_1 p_y - a_1) + (a_3 c_3 + a_2)}\right) \quad (55)$$

$$\theta_4 = 90 - \theta_2 - \theta_3 \quad (56)$$

Although in this case $\theta_5$was never changed, by substituting all the values of Table.3, in the above formulae, the θ values can be found.

Verification of the Model by MATLAB and ROBOCELL: ROBOCELL Software is used to validate the Inverse Kinematic Model of the Scorbot-ER Vu Plus Industrial Robot.

Components of ROBOCELL: ROBOCELL software coordinates the four segments: a) SCORBASE, full-emphasized robotics control programming software, which gives an easy to use device for robot programming and process, b) A module with graphic exhibit that provides robot's 3D dissection of the robot and different gadgets in a fundamental robotic environment, where one can characterize (instruct) the robot movements and accomplish robot programs, c) Cell setup, which permits a user to make another virtual automated work cell, or change a current work cell, d) 3D simulation software display to exhibit ROBOCELL's capacities.

ROBOCELL's illustration of a robot and gadgets is focused around the genuine measurements and capacities of the Scorbot-ER Vu Plus Robot equipment. In this way, working and programming the robot in ROBOCELL could be utilized with a genuine robotic establishment. Realistic presentation peculiarities and programmed operations, for example, cell reset and send robot commands, empower fast and exact programming. SCORBASE's imitate the client interface and menus of ROBOCELL. SCORBASE operations, menus and commands are depicted in the SCORBASE hand book.

ROBOCELL gives the strategies depicted beneath, for characterizing the positions of the robot. Allotted number indicates its position.

Recording Position (First Method): The virtual robot is controlled by the SCORBASE physical dialog box like a real robot. In the teach position (Simple) dialog box a number is written in the location number field when the position is reached. Click the record button. The new position will replace the existence position data in the event that the position number has been utilized long ago.

Recording Position (Second Method): Position and above position tools are used for specific location of robot, while robot moves to object. Calibration is done by utilizing the manual movement dialog box. Until the position is reached, location number field's in the teach position (Simple) dialog box is written. Click the record button. In the event that the position number has been utilized at one time, the new position will overwrite the past position information.

Teaching Position: Object X and Y coordinates notice in the Graphic Display window, the selection has been made on "View/Show Positions". To record the coordinates of the object or point, it should be zoomed in. To open the Teach Position dialog box (Figure 4) the "Teach Position (Simple) Expanded button" is clicked. Position coordinates X, Y, Z, P, and R are entered in their respective fields. The position number field is entered with a number. The teach button is clicked.

Fig.4. Teach Position (Expand)

In the event that the "Record Position Catch" is clicked, the location of previous robot is taken (and not the locations characterized by the directions imported in the X, Y, Z, P, and R fields).

The new position will replace the previous position data's when the position number has been used formerly.

Fine-Tuning a Position: To modify the existing positions: The "Teach Position (Simple) Expanded button" is selected to unwrap the Teach Position dialog box. To modify a position, it is selected in the "Position Number Field". The "Get Position" button is clicked. The position data presents in the fields of X, Y, Z, P, and R. The required coordinate is modified. To overwrite the previous position, the "Teach button" is clicked.

Program Execution: ROBOCELL project execution is the similar as performing project while utilizing a genuine robot framework. Diverse cell arrangements might be stacked and changed in ROBOCELL. Be that as it may the positions and projects are not stacked together with their work cell. For another task, to consider the work cell and its positions, the Save as choice in the File menu might be utilized. The undertaking with work cell and positions is spared under an alternate name. At that point the project is erased and another one is composed. (The positions and the cell stay unaltered).

For a given set of parameters (Table 1), a program in MATLAB 7.10 and ROBOCELL is created, and the comparison between its outputs with the experimental output as mentioned below. See Figure5 and 6. Table 3 shows the differences between the Analytical and physical values of the robot.

Table.3. Differences between the Analytical and Physical Values of the Robot (Inverse Kinematic)

Position Values	Analytical Θ Values (°)	Matlab Θ Values (°)	Error	Analytical Θ Values (°)	Robocell Θ Values (°)	Error
θ_1	30 °	30 °	0°	30 °	30.16 °	0.16 °
θ_2	45°	45.31°	0.31°	45°	45.27 °	0.27 °
θ_3	60°	59.88°	0.12°	60°	60.25 °	0.25 °
θ_4	75°	75.43°	0.43°	75°	75.11 °	0.11 °
θ_5	75°	75.43°	0.43°	75°	75.11 °	0.11 °

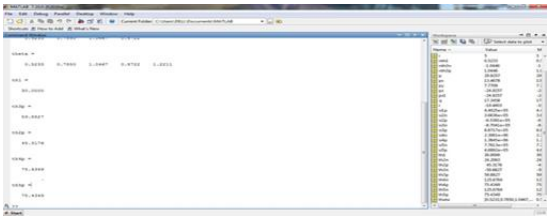

Fig. 5. MATLAB Result for the Inverse Kinematic

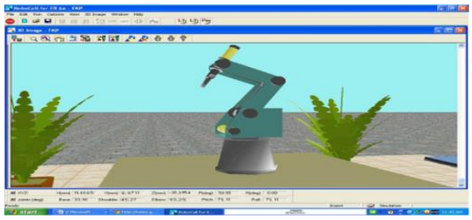

Fig.6. ROBOCELL Result for the Inverse Kinematic

2. METHODOLOGY

Artificial Neural Network: It is a computational system influenced by the structure, processing method and learning ability of a biological brain. The characteristics of an ANN include a large number of very simple processing neuron like processing elements, a large number of weighted connections between the elements and the distributed representation of knowledge over the connections. This knowledge is acquired by network through a learning process.

Nonlinear Autoregressive Network with Exogenous Inputs (NARX): Nonlinear autoregressive models with exogenous inputs (NARX) neural network architectures can be compared with the recurrent neural models. In NARX the side and later the delayed versions are provided as the input. But in the case recurrent layer Network the hidden unit output is get back to the input side and this helps to understand the performance of the hidden layer. The NARX models are commonly used in the system of identification area.

This network model is based on the linear ARX model, which is mainly adopted in time-series modelling. In Nonlinear Autoregressive with exogenous input (NARX), the output equations is obtained as:

$$y(n+1) = f\,[y(n), y(n - dy + 1)\, u(n - 1), u(n - du + 1)] \quad (57)$$

$$y(n + 1) = f[y(n);u(n)] \quad (58)$$

u (n) and y(n) denotes the input and the output for the discrete time n respectively. The terms du ≥ 1 and dy ≥ 1, du ≤ dy, are memory delays. The3 NARX Neural network architecture is given in Fig.7.

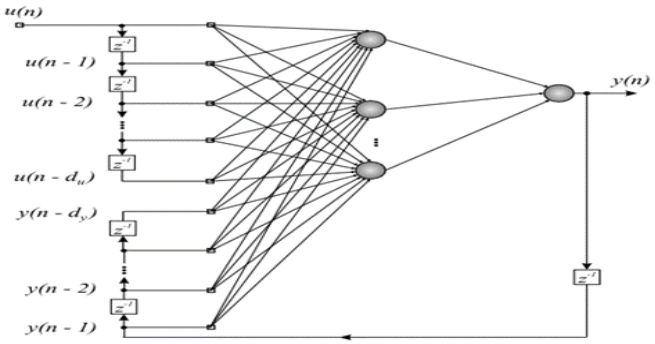

Fig.7. NARX Neural Network Architecture

Takens' Theorem is a delay embedding theorem provides the conditions for which a dynamical system can be reconstructed from a sequence of observations. This reconstruction process maintains properties and do not allow to change coordinates smoothness, but it doesn't consider the geometric shape of structures in phase space. Takens' Theorem is used to build the input regressor:

$$Y(n) = F[u(n).Y(n-1)] \quad (59)$$

$$U(n) = [x(n)\ x(n-\tau)..x(n-(dE-1)\tau)] \quad (60)$$

The equation implements the delay embedding theorem.

Where, x (n) - sample value of the time series at time n, dE - embedding dimension, τ is the embedding delay.

Based on this theorem, a set of time-lagged values in a dE-dimensional vector space gives necessary information for the reconstruction the states. For the nonlinear time series prediction by using the NARX network, aninnovative definitions for its input and output regressors are anticipated. ie.. du = dE. The output regressor y(n) can represented in two different methods based on the training modes of the NARX network:

$$Yp(n) = [bx(n),\ldots..,\ bx(n-dy+1)] \quad (61)$$

$$Ysp(n) = [x(n),\ldots\ldots,\ x(n-dy+1)] \quad (62)$$

The P-mode includes the dy past values of the obtained time series, at the same time the SP-mode involves the dy past values of the actual time series.

3. RESULT AND DISCUSSION

Since considering the inverse kinematics, in this work, the Cartesian coordinates X (150,320), Y (0,150) and Z (-24,505) are given as inputs and θ_1(0, 45), θ_2(-120, 43), θ_3(20,108), θ_4(-3, 89), θ_5(0, 70) are the angles between the inputs and joints which is called target angles. There are three inputs and five outputs and the goal is to obtain a minimum mean square error.

The positions and orientation based on the Cartesian coordinate system has been analyzed by employing the artificial neural network method NARX algorithms by using a set of joint angles as desired output.

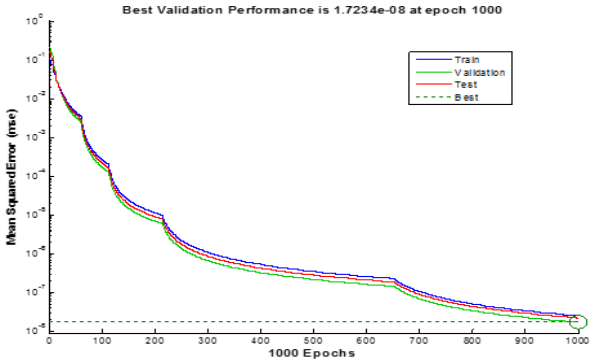

Fig.8. Performance Plot of NARX

The training state plot is a graphical representation of the progress of other training variables such as gradient magnitude; number of validation checks etc. is shown in Fig. 9.

Table.4. Desired Target

θ_1	θ_2	θ_3	θ_4	θ_5
0	-120.2	95.03	88.81	0
0	-8.93	107.87	-8.93	0
0	-8.88	89.59	9.29	0
0	-2.95	65.09	27.87	0
45	-9.35	105.05	5.70	-45.02
45	-6.17	76.21	19.95	45.02
45	14.78	20.48	54.73	45.02
67.19	-9.63	100.80	-1.17	0
21.56	-4.43	69.91	24.53	0
38.10	-15.86	98.92	-83.06	0
30	42.30	58.48	-10.79	70

The Table 4 shows the desired target which is given as input. The obtained output is compared with the desired target and based on the difference the adjustments made to the weight vectors and thus the desired target can be obtained. This process is known as Training.

By considering and analysing the obtained outputs from NRAX given in Table 5 with reference to Table 3 and 4, it can be concluded that the estimated output is very much closer to the desired target.

Table.5. Obtained Output Using NRAX Algorithm

θ_1	θ_2	θ_3	θ_4	θ_5
0	-120.27	95.03	88.81	0
0	-8.93	107.87	-8.93	0
0	-8.88	89.59	9.29	0
0	-2.95	65.09	27.87	0
45.00	-9.35	105.05	-5.70	-45.02
45.00	-6.17	76.21	19.95	45.02
45.00	14.78	20.48	54.73	45.02
67.19	-9.63	100.80	-1.17	0
21.56	-4.43	69.91	24.53	0
38.10	-15.86	98.92	-83.06	0
30.00	42.30	58.48	-10.78	70

Table 5 represents the optimized joint angles value which is the target. By comparing it with the desired target it can be noted that the output obtained is almost equal to the desired target. It denotes the performance reaches the goal to an extent.

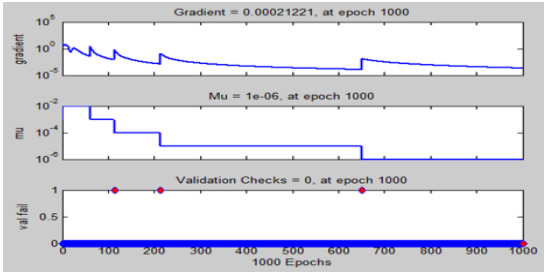

Fig.9. Training State of NARX NN Algorithm

The network got trained using different parameter values and can be simulated using MATLAB software. The program also gives the mean square error by which the performance can be analyzed. Figure 8 is the performance plot acquired by the NARX algorithm and is achieved as 1.7234e-08. Here in the input side rather than the applied input, an exogenous data fed back from the output side is also given as input.

4. CONCLUSION

This work formulates and simulates the solution for the inverse kinematic problem of a five DOF robot is obtained using the intelligent technique, Artificial Neural Network and can be analyzed by the performances of the adopted Nonlinear Autoregressive Network (NARX) algorithm. For improving the accuracy of the predicted joint and link angles, the number of input patterns of various coordinate values to be trained can also be increased.

REFERENCES

Adrian-Vasile Duka, Neural Network Based Inverse Kinematics Solution for Trajectory Tracking of a Robotic Arm, International Conference on Computer Engineering and Applications, 2, 2011.

Azme Khamis and Siti Nabilah Syuhada Binti Abdullah, Forecasting Wheat Price Using Back propagation and NARX Neural Network, The International Journal of Engineering and Science, 3 (11), 2014.

Bassam Daya, Shadi Khawandi, Mohamed Akoum, Applying Neural Network Architecture for Inverse Kinematics Problem in Robotics, Journal Software Engineering & Applications, 3, 2010, 230-239,

Daniel Tarnita, Dan B. Marghitu, Analysis of a Hand Arm System, International Journal of Robotics and Computer-Integrated Manufacturing, 29, 2013, 493–501.

Koker R, Oz C, Çakar T, and Ekiz H, A Study of Neural Network Based Inverse Kinematics Solution for a Three-Joint Robot, Robotics and Autonomous Systems, 49, 2004, 227–234.

Moshan Mirkhani, Rana Forsati, Alireza Mohammed Shari, A Novel Efficient Algorithm for Mobile Robot Localization, International Journal of Robotics and Autonomous, 61, 2013, 920–931.

Pham D, Castellani M, and Fahmy A, Accountability Learning the Inverse Kinematics of a Robot Manipulator Using the Bees Algorithm, 6th IEEE International Conference on Industrial Informatics, 2008, 493–498.

Ramirez A, and Rubiano F, Optimization of Inverse Kinematics of a 3R Robotic Manipulator Using Genetic Algorithms World Academy of Science, Engineering and Technology, 59, 2011.

Rasit Koker, A Neuro-Genetic Approach to the Inverse Kinematics Solution of Robotic Manipulators, Scientific Research and Essays, 6 (13), 2011, 2784-2794.

Wei L, Wang H, and Li Y, A New Solution for Inverse Kinematics of Manipulator Based on Neural Network, International Journal of Machine Learning and Cybernetics, 2, 2003, 1201–1203.

Zhen-Guo Che, Tzu-An Chiang, and Zhen-Hua Che, Feed Forward Neural Networks Training: A Comparison between Genetic Algorithm and Back-Propagation Learning Algorithm, 7 (10), 2011.