Article citation info:
Rashid ZH, Sarhan RA, Hassan MS. Arduino-based implementation of kinematics for a 4 DOF robot manipulator using artificial neural network. Diagnostyka. 2024;25(1):2024114. https://doi.org/10.29354/diag/184235.

DIAGNOSTYKA, 2024, Vol. 25, No. 1

e-ISSN 2449-5220
DOI: 10.29354/diag/184235

ARDUINO-BASED IMPLEMENTATION OF KINEMATICS FOR A 4 DOF ROBOT MANIPULATOR USING ARTIFICIAL NEURAL NETWORK

Zaid Hikmat RASHID *, **Riyadh Ahmed SARHAN**, **Mohammed Salih HASSAN**

Al-Furat Al-Awsat Technical University, Technical Institute of Al-Mussaib, Iraq
* Corresponding author: zhr.1986@atu.edu.iq

Abstract

Real-time motion control is basically dependent on robot kinematics analysis where there is no unique solution of the inverse kinematics of serial manipulators**.** The predictive artificial neural network is a powerful one for finding appropriate results based on training data. Therefore, an artificial neural network is proposed to implement on Arduino microcontroller for a 4-DOF robot manipulator where the inverse kinematics problem was partitioned to suit the low specification of the board CPU. With MATALB toolbox, multiple neural network configuration based were trained and experienced for the best fit of the dimensionless feedforward data that is obtained from the forward kinematics of reachable workspace with average MSE of 6.5e-7. The results showed the superior of the proposed networks reducing the joints error by 90 % at least with comparing to traditional one. An Arduino on-board program was developed to control the robot independly based on pre validated parameters of the network. The experimental results approved the proposed method to implement the robot in real time with maximum error of (± 0.15 mm) in end effector position. The presented approach can be applied to achieve a suitable solution of n-DOF self-depend robot implementation using micro stepping actuators.

Keywords: robot, kinematics, neural network, arduino, micro stepping

List of Symbols/Acronyms

c_i – $\cos(q_i)$;
c_{ij} – $\cos(q_i+q_j)$;
DH – Denavit-Hartenberg;
DOF – Degree of freedom;
MLP – multi-layer perceptron;
MSE – Mean sqaur erroe;
NN – Nueral network;
R_{ij} – Rotation matrix element;
s_i – $\sin(q_i)$;
s_{ij} – $\sin(q_i+q_j)$;
T – Transformation matrix;

1. INTRODUCTION

Finding an accurate and dependable kinematics solution is the main challenge in robotic arm motion control. Modern artificial intelligence has been involved with robotic control technology. It has significant performance benefits like precise control and faster computation.

In general, several approaches were presented to deal with the nonlinearity of robots' kinematics equations. In comparison to the forward kinematics of a serial manipulator, inverse kinematics solutions present numerous challenges. By Denavit-Hartenberg (DH) method the forward kinematic is directly derived. The inverse kinematics can be resolved analytically, geometrically, or iteratively[1]. For familiar types of manipulators, the kinematics is modeled in direct steps[2]. For a multiple Degree of Freedom (DOF) or redundant robot or singularity-free path planning, an artificial training may be required to resolve the kinematics. F.A. Raheem et al [3] proposed a multilayer perceptron structure by using a back propagation training algorithm to find the required joint angle for the end effector position. C. Kenshimov et al [4] trained a multi-layer perceptron neural network (MLPNN) for a 4-DOFs INMOOV manipulator to investigate the desired set of angle position from a given set of end effector pose, where the data base on the iterative logic of solving the inverse kinematics, the experimental results showed an acceptable mapping of robot workspace with 95.6% fit for all joints angles. For a 4-DOFs SCARA robot, P. Jah and B. Biswal [5] studied the application of MLP, they observed that the Neural Network (NN) minimized the joints variables errors and improves the performance index. The feed forward neural network with two layers is used to track specific trajectory by resolving inverse kinematics of three link planner robot based on input-output data[6]. A. Almusawi et al [7] suggested a new Artificial Neural

Received 2023-06-13; Accepted 2024-02-18; Available online 2024-02-19

Network (ANN) to resolve the 6-DOF manipulator's inverse kinematics, by adding the recent joint information into the conventional ANN, the new design enhances the system's overall performance. From forward kinematics equations the training data for ANN were obtained, where the model of the manipulator is visualized by MATLAB Simulink to evaluate the path errors [8]. S. Aravinddhakshan et al [9] experienced two neural networks for a 5DOFs industrial manipulator, where the networks differed in the input training data, one for only end effector position and the other for the complete pose information. The results showed the first network errors are very minimal compared to other networks. Furthermore, an advanced logics is proposed to solve the inverse of robot manipulator like adaptive fuzzy self-tuning control system [10]. An artificial neural fuzzy inference system (ANFIS) was based to analyze the kinematics of 4-DOFs SCARA robot in order to utilize for desired path generation [11]. Also, a multiple ANFIS networks system is built to resolve the SCARA inverse kinematics [12]. J. Demby's et al [13] investigate the solving of multiple robotic arms using artificial neural network and ANFIS where both methods converged approximately the same results.

The previous works showed the feasibility of the neural network approach in solving the inverse kinematics. But, the most of researchers implement the kinematics solutions in simulation programs using different structures of network. Based on software package's toolbox or a special program that developed by the manufacturer for a specific type of robot, the results were compared and discussed. Otherwise, in real time implementation there are many of troubles may to deal with in order to execute any of proposed algorithms. It depends on lab's facilities equipment in hardware like measurements sensors or the Professionalism in software programming like avoiding the floating of the solver, the overlapping of decision loop, signal data surging in receiving information, etc.

The main aim of this work is the experimental real time implementation of the onboard neural network. Where, an open-source microcontroller board (Arduino mega 2650) is utilized to implement the robot experimentally based on pre calculated and validated parameters of a specific neural network simulation. The neural network is trained for the best fit of the feedforward data that obtained from the forward kinematics of reachable workspace. For that purpose, a prototyping model of 4-DOF SCARA robot manipulator as shown in figure 1 was built using stepper motors as actuators of joints to achieve a high accuracy robot manipulator. In addition, a micro stepping technique were applied in order to smooth the controlled motion.

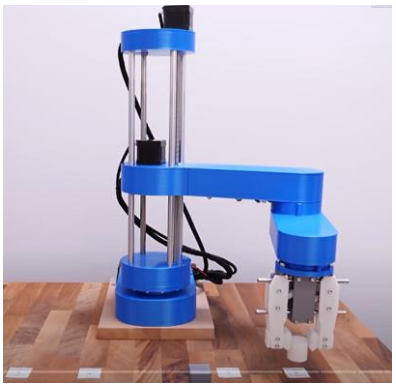

Fig. 1. SCARA robot model [14]

2. ROBOT KINEMATICS FORMULATION

The homogenous transformation convention of DH is based on an analysis of the robot's forward kinematics. by attaching a frame coordinate system at each joint and specifying the DH main parameters [15]:

- a_{i-1}: represents the distance from z_{i-1} axis to z_i axis measured on the x_i axis.
- α_{i-1}: represents the rotation angle from z_{i-1} axis to z_i axis measured on the x_i axis.
- d_i : represents the distance from x_{i-1} axis to x_i axis measured on the z_i axis.
- θ_i : represents the rotation angle from x_{i-1} axis to x_i axis measured on the z_i axis.

The transformation matrix from i frame to previous i-1 frame is:

$$ {}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_{i-1} & s\theta_i s\alpha_{i-1} & a_{i-1}c\theta_i \\ s\theta_i & c\theta_i c\alpha_{i-1} & -c\theta_i s\alpha_{i-1} & a_{i-1}s\theta_i \\ 0 & s\alpha_{i-1} & c\alpha_{i-1} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) $$

Table 1 lists the robot's DH parameters and joints limitations based on the associated coordinate frames, as shown in figure 2.

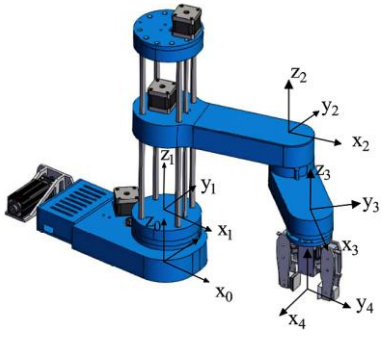

Fig. 2. The assigned robot frame coordinates

Table 1. DH Parameters of the Robot Manipulator

i	a_i	α_i	d_i	θ_i	Limits of joints variable
1	0	0	d_1	q_1	-150 to 150 (deg)
2	a_2	0	d_2	0	0 to 500 (mm)
3	a_3	0	d_3	q_3	-170 to 170 (deg)
4	0	0	d_4	q_4	-120 to 120 (deg)

Where a_2=252, a_3=172, d_1=77, d_3=-50 and d_4=-120 in millimeters and the negative sign refers for opposite direction.

The different transformation matrices can be multiplied sequentially to yield the overall transformation matrix of the tool frame relative to the origin frame as [16]:

$$ {}^{0}_{4}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T \tag{2}$$

$$ {}^{0}_{4}T = \begin{bmatrix} R_{11} & R_{12} & R_{13} & P_x \\ R_{21} & R_{22} & R_{23} & P_y \\ R_{31} & R_{32} & R_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

$$ {}^{0}_{4}T = \begin{bmatrix} c_{134} & -s_{134} & 0 & a_2c_1 + a_3c_{13} \\ s_{134} & c_{134} & 0 & a_2s_1 + a_3s_{13} \\ 0 & 0 & 1 & d_1 + d_2 + d_3 + d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

From equation 4, the invers kinematics can be obtained for the joint's variables. the prismatic joint has a direct solution as:

$$ d_2 = -\left(d_1 + d_3 + d_4\right) \tag{5}$$

Furthermore, it is noted that the end effector rotates simply along the z axis as follows:

$$ \psi = q_1 + q_3 + q_4 \tag{6}$$

Where it can be found as:

$$ \psi = a\tan 2(R_{21}, R_{11}) \tag{7}$$

The problem may be reduced to a two-link planner robot by admitting the q_2 and projecting the robot configuration on the base frame as shown in figure 3. The geometrically attainable solution can be found as follows [17]:

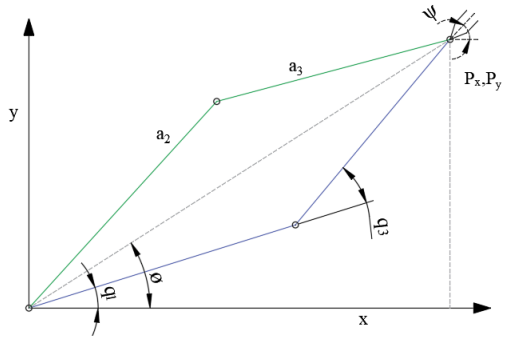

Fig. 3. Top view of the robot links

From equations 3 and 4, the elements of (1,4) and (2,4) can be solved by adding the squared quantities which yield to:

$$ P_x^2 + P_y^2 = a_2^2 + a_3^2 - 2a_2a_3c_3 \tag{8}$$

From above:

$$ c_3 = \frac{P_x^2 + P_y^2 - a_2^2 - a_3^2}{2a_2a_3} \tag{9}$$

$$ s_3 = \pm\sqrt{1 - c_3} \tag{10}$$

$$ q_3 = a\tan 2(\pm s_3, c_3) \tag{11}$$

The two solutions of q_3 reveal another two solutions for q_1:

$$ q_1 = a\tan 2(P_y, P_x) \pm a\tan 2(a_3s_3, a_2 + a_3c_3) \tag{12}$$

3. NEURAL NETWORK DESIGN

The closed form inverse of such a robot has a multi-jointed set of solution probabilities, as evidenced in equations 11, 12 and 6. Thus, a need arises to use the training neural network of a multilayer to fit the solutions of configurations [18].

For three input (x,y,ψ) of wrist pose and three target (q1,q3,q4) of joint angle the training may be extensive calculations with time consuming. It was found that the network has poor convergence with high errors for the wide range of training data of [-120 120] deg for q1 and [-150 150] deg for q3. As depicted in figures 4 and 5, the training stops at 17 epochs with unacceptable mean square error of 1.445. Even the hidden layers, number of neurons and training function were changed for best regression fitting, which did not go beyond the 63 percentage. At this point, performance substantially declines, leading to an increase in high output errors, as seen in figure 6.

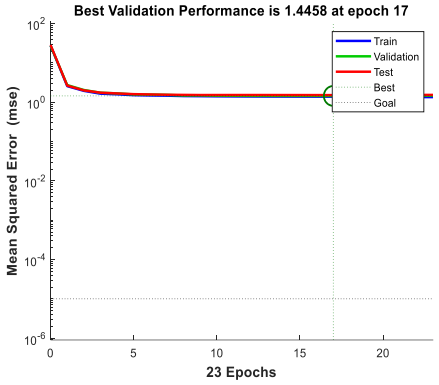

Fig. 4. The NN performance for the wide workspace training

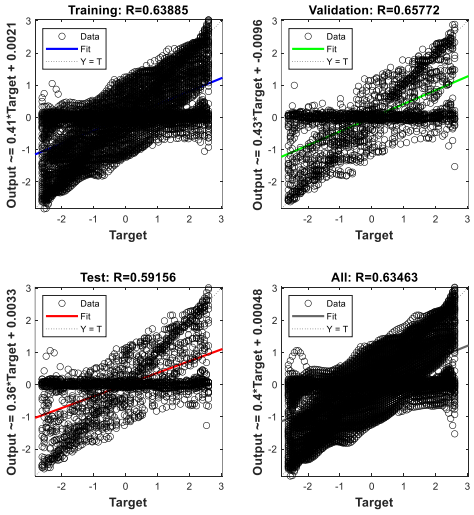

Fig. 5. Regression plot of NN wide workspace training

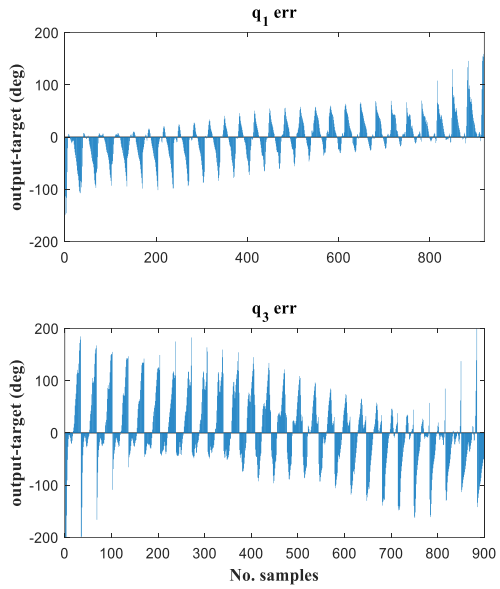

Fig. 6. NN errors for the wide workspace training

There are several ways to increase the neural network accuracy, such as increasing the epoch or the input data and selecting the appropriate activation function beside the increasing of neurons and hidden layers, but that may conflict with the aim of implementing the neural network on a microcontroller. The workspace can be resized for the effective area and depend on one configuration of the left elbow to train a network for the left half of the workspace (±x, +y), and another mirrored network for the right half (±x,-y). It can also decouple the trained data for separate networks for each joint variable and combine the solution together manually for the final target result. Table 2 summarized the performance of the individual networks.

Table 2. Performance of individual networks

Neural Network	NN1 of q_1	NN2 of q_3	NN3 of q_4
Best validation performance	9.521e-5	4.114e-5	5.114e-6
Instances min. errors	2.23 e-04	2.12 e-4	6.51 e-5
Instance max. errors	4.85 e-03	1.06 e-3	3.25 e-4

In order to increase the robot precision, the inverse kinematics problem may be partitioned by considering the vector of $\{P_x,P_y,\}^T$ as input data and the corresponding joints variables of $\{q_1,q_3\}^T$ as the target data. While the end effector orientation and height of Pz can be directly inversed to determine q_4 and q_2, as previously mentioned in equations 5 and 6. With couples of experimental trains in MATLAB, a satisfactory network that consistent with paper claim is built for a most active reachable workspace of range [-80,180] (deg) of q_1 and [-179,0] (deg) of q_3 for left elbow configuration. The reason of select these criteria of joints variable is the normalization process like to increase the network performance and reduce the noise of similar data.

In concepts, the training process of a robotic arm relates the configuration with end effector pose. From equation (4)

$$P_x = a_2c_1 + a_3c_{12} \quad (13)$$

$$P_y = a_2s_1 + a_3s_{12} \quad (14)$$

which can be written as follow in matrix form:

$$\begin{Bmatrix} P_x \\ P_y \end{Bmatrix} = \begin{bmatrix} c_1 & c_{12} \\ s_1 & s_{12} \end{bmatrix} \begin{bmatrix} a_2 \\ a_3 \end{bmatrix} \quad (15)$$

Since the trigonometric functions values from -1 to 1 in peaks, then the input values may be mapped to 1 and be dimensionless to suite the targets values which gives more precise regression. The max point in work space can be attained with value of extended arm which is equal to a_2+a_3, equation 15 can be written as

$$\begin{Bmatrix} \frac{P_x}{a_m} \\ \frac{P_y}{a_m} \end{Bmatrix} = \begin{bmatrix} c_1 & c_{12} \\ s_1 & s_{12} \end{bmatrix} \begin{bmatrix} \frac{a_2}{a_m} \\ \frac{a_3}{a_m} \end{bmatrix} \quad (16)$$

A four separated networks for the two dimension less inputs data network can be trained for each output of c_1,c_{12},s_1 and s_{12}. The proposed architecture for a single network can be seen in figure 7.

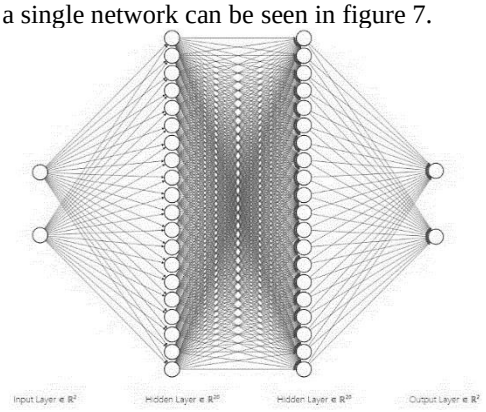

Fig. 7. The architecture of two layers with 20 neurons in each

Table 3 summarized the performance of the networks (see figure 8).

Table 3. The proposed configuration networks performance

Neural Network	NN1 of c_1	NN2 of c_{12}	NN3 of s_1	NN4 of s_{12}
Best validation performance	3.74 e-7	1.99 e-7	1.40 e-6	6.55 e-7
Instances min. errors	12.2 e-5	7.12 e-5	9.53 e-5	13.1 e-5
Instance max. errors	94.1 e-5	73.6 e-5	7.27 e-5	93.2 e-5
All regression training	~1	~1	~1	~1

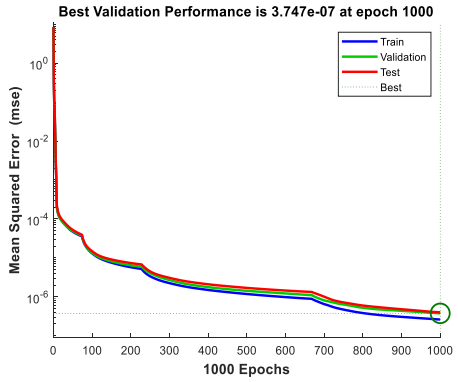

Fig. 8.a. NN of c_1

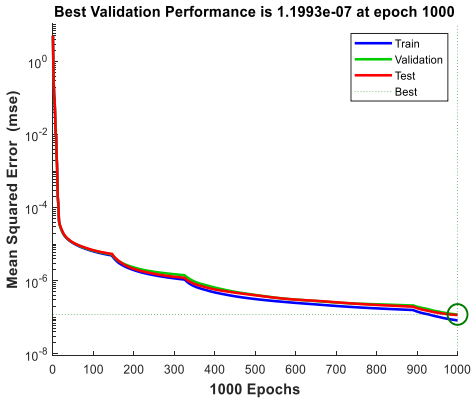

Fig. 8.b. NN of c_{12}

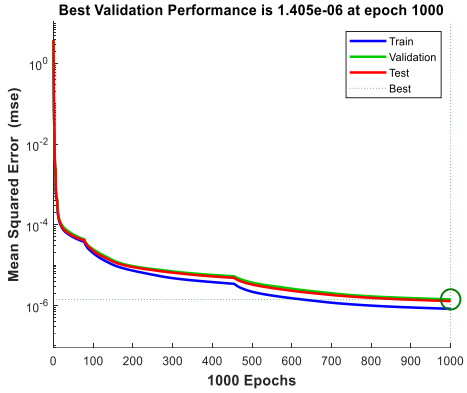

Fig. 8.c. NN of s_1

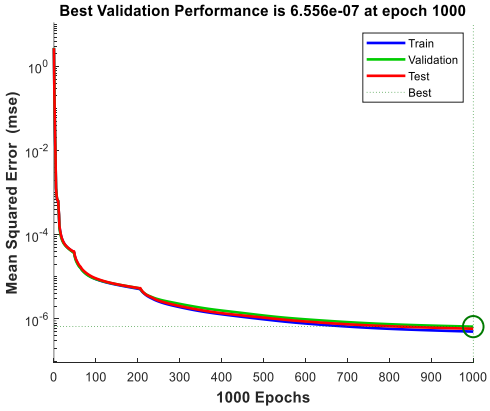

Fig. 8.d. NN of s_{12}

4. ARDUINO SETUP

The implementation program was developed in the Arduino C language using an Integrated Development Environment (IDE) for the open source microcontrollers. For a certain end effector position, the required pulses and direction for each joint may be determined. Since the pulses count is an integer, the round function may be used for that purpose. Also, the sign refers to the direction of rotation. There will be a small difference between the desired and actual pose in every transition. Its value depends on the step size; to reduce the offsets, the calculations may be based on the previous location anywhere. Then the actual actuator rotation angle can be calculated again, where it can be used to determine the actual end effector location using the forward kinematic equations.

$$j_i = Round\left(\frac{q_{i+1} - q_i}{360^o} * steps_i\right) \tag{17}$$

$$q_{i(m)} = q_{i-1(m)} + \frac{j_i}{steps_i} * 360^o \tag{18}$$

For joint 2, the required d_2 obtained from equation 5, then in the same way:

$$j_2 = Round((q_{i+1} - q_i) * (step / mm)) \tag{19}$$

$$q_{2(i)} = q_{2(i-1)} + \frac{j_2}{(step / mm)} \tag{20}$$

A typically bipolar NEMA 17 motor with 200 pulses per revolution was used for all joints with a DRV8825 driver. In combination with these drivers, the CNC shield of the Arduino was utilized to shorten the wire connections. Table 4 shows the total pulses per revolution for each actuator.

Table 4. Actuators total pulses

Joint No.	Joint 1	Joint 2	Joint 3	Joint 4
Microstepping set	1/8	1/2	1/8	1/4
Pulley ratio / Lead screw	6	8 mm (pitch)	5	4
Pulses per revelution	9600	50 per 1 mm	8000	3200

The neurona library [19] is used to process the inverse kinematics of the end effector's input data, which can run MLP algorithms on AVR boards and do inferencing in real time. The required weights of the network structure can be extracted either from the MATLAB neural network and rearranged to meet the command of the neurona library, also it can be generated directly by the alternative tool from the MLP topology workbench as shown in figure 9.

Hence, the actuators can be controlled based on forward and inverse kinematics analysis with the support of accelstepper library [20]. the actuators simultaneously were implemented with acceleration and deceleration to smooth the motion and reduce the vibration of sudden stop which may cause

drifting errors. Figure 10 shows the flowchart of the program.

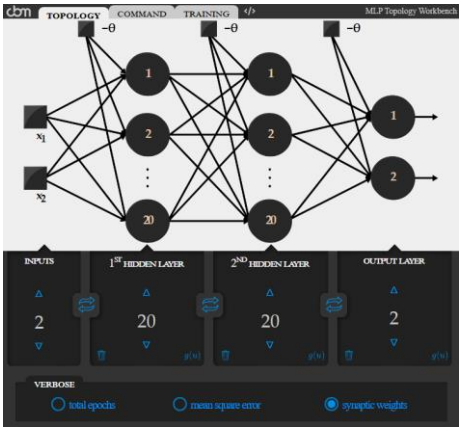

Fig. 9. MLP topology workbench training tool [19]

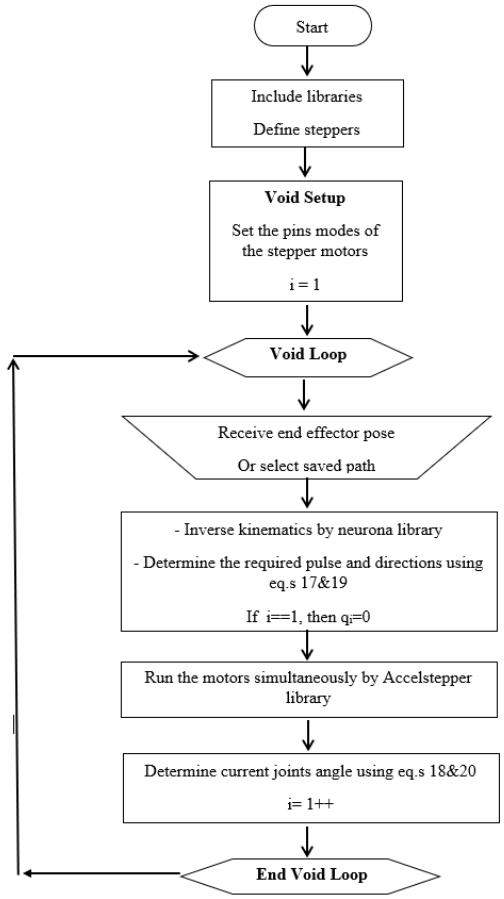

Fig. 10. Program flowchart

5. RESULTS AND DISCUSSION

In order to analyze the model efficiency an arbitrary circular path in plane within the workspace was selected and simulated as shown in 11. In details, the circular path error was presented for both traditional training and the proposed training in figure 12 and 13 respectively. it is observed that the maximum error 3 mm in end effector position in traditional network while it's not exceeded 0.15 mm in the proposed model.

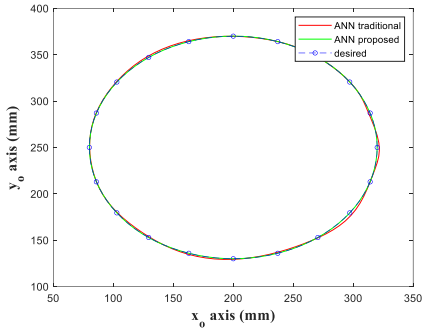

Fig. 11. Circular path of the end effector in x_o-y_o plane

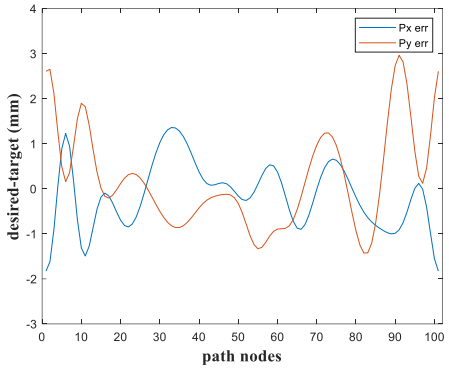

Fig. 12. Traditional NN errors of the end effector on circular path

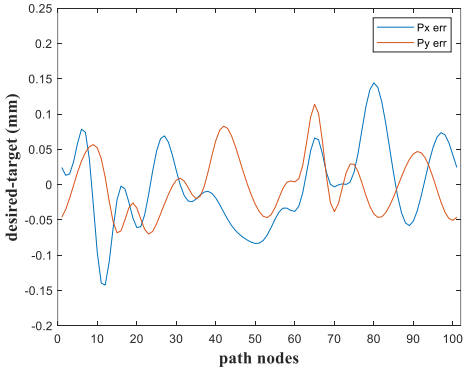

Fig. 13. Proposed NN errors of the end effector on circular path

Figures 14 and 15 showed the differences between the inverse kinematics closed form solution and the target data of joints q_1 and q_3 for both networks traditional and proposed training. It's noted that the joints error was reduced by 90 % at least from 0.4 to 0.04 deg in maximums.

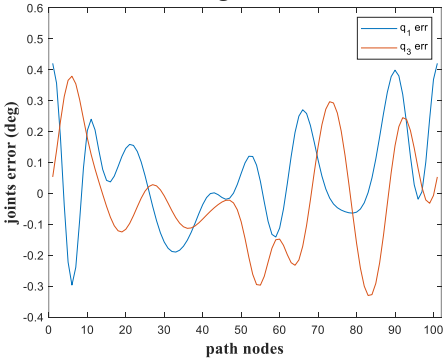

Fig. 14. Traditional NN joints error for circular path

A Simulink model is developed to calculate the expected experimental errors based on the adopted neural network of the dimensionless data, taking into account the generated pulses' differences of actuators. The close form solution is included as a user-defined function to compare the results and scope the overall performance for various points and paths. In addition, the forward kinematics block is modified to calculate the end effector tool orientation error of q_4 and the prismatic joint of q_2 to achieve the final experimental pose vector. Figure 16 shows the Simulink model.

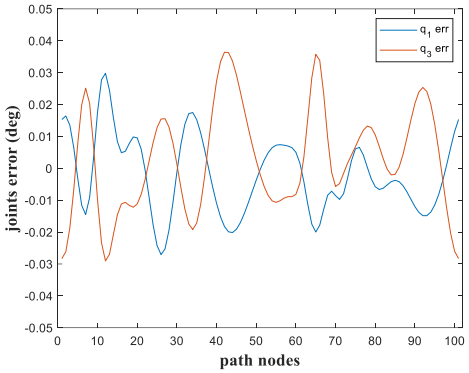

Fig. 15. Proposed NN joints error for circular path

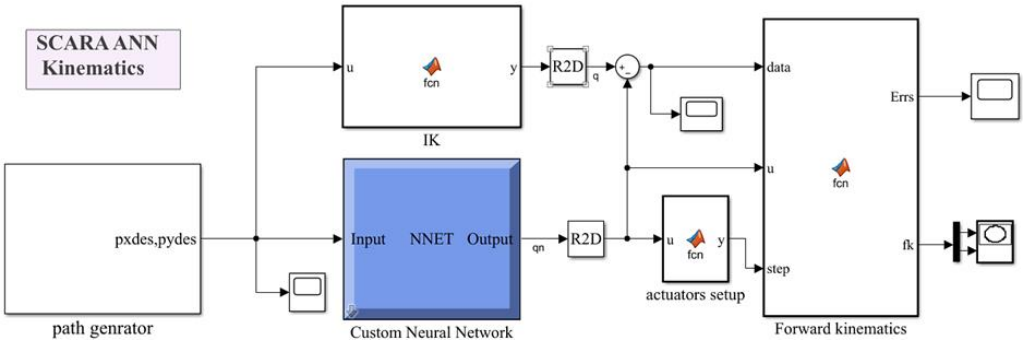

Fig. 16. Simulink model of the proposed neural network

Experimentally, six points were generated with different end effector orientation within the workspace. Figure 17.a shows the selected points as listed in table 5. By the cubic spline interpolation of Simulink, the path is drawn as shown in figure 17.b. the optimal path planning and trajectory tracking are beyond the scope of this paper. Thus, the motion between the points is assumed with full stop via the breakpoints to experiment the robot feasibility. In addition, the fourth point is selected carefully to meet a singular configuration of straight arm to discover the neural network capability in solving similar situation.

Table 5. Break points location

No.	Px (mm)	Py (mm)	Pz (mm)	ψ (deg)
1	273	-215	135	30
2	400	-50	130	10
3	301	178	135	5
4	254	453	110	-5
5	-60	420	80	-10
6	79.71	235.24	20	-30

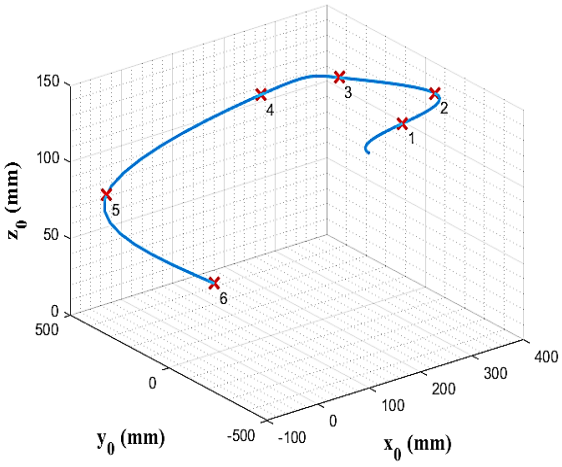

Fig. 17.a. The path of the end effector in cartesian space

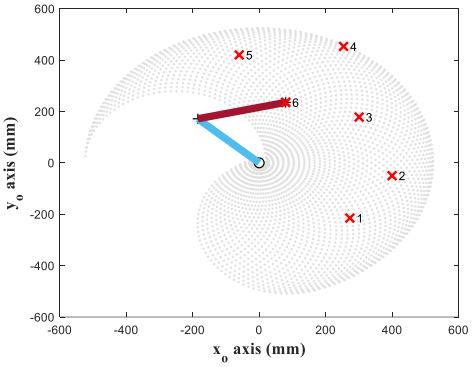

Fig. 17.b. The projection of the break points at x_o-y_o plane

Figure 18 depicts the calculated absolute differences between the desired joint angle and the actual angle of q_1, q_3 and q_4. By applying the forward kinematics equations, the position errors can be determined as shown in figure 19.

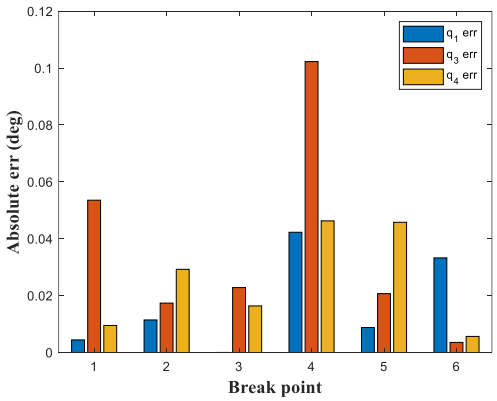

Fig. 18. The absolute errors in the revolute joints

It was observed that the maximum error was approximately limited to 0.14 mm. since the robot pose is a function of joints and links length the errors can vary according to point location. The experimental errors are expected due to the small differences in the stepping rotation of the actuators. Also, the Pz has no error to mention because it's a direct function of q_2 that has a small difference due to the direct implantation of the stepper pulses only.

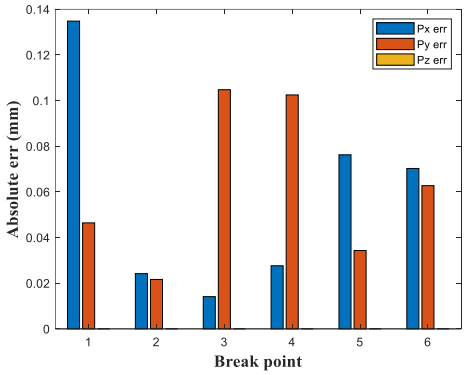

Fig. 19. The absolute errors of the end effector position.

In contrast to the traditional geometrical solution, the neural network achieves an adequate good solution for the singular configuration. Figure 20 depicts the overall end effector error, where it's based on the actual rotation of q_1 and q_3 as mentioned before.

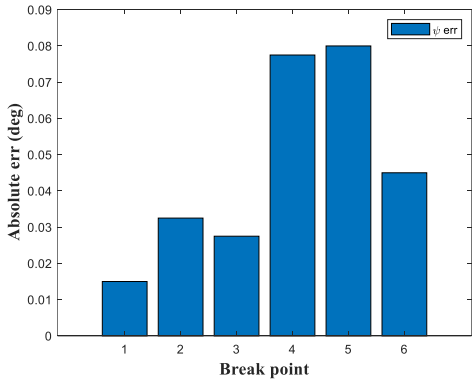

Fig. 20. The absolute errors of the end effector orientation

Finally, a comparison was made to similar kinds of robots to evaluate the robustness of the proposed method as listed in table 6.

6. CONCLUSION

On the basis of the results, it's concluded that the ANN method has a satisfactory and tracking error. Input-output information, hidden layers number and neurons counts beside the activation functions type and the rate value of training are enhanced the neural network accuracy. In contrast to simulations, the real-time implementation may be constrained by factors like network task and processing capability. When compared to geometric, analytic, and iterative approaches, the on-board ANN program is computationally inexpensive and responsive to control the real-time implementation. The proposed ANN structure is configrations based and the dimensioless mapped data has an effecient results by reducing the maximum tracking error to 0.15 mm in position at maximum. The results showed the network capability in solving the inverse kinematics across the robot's singular configuration in spite of the complex nonlinear behavior of the data set. The presented approach achieves a suitable solution for similar kinds of robot implementation and can be applied for the simple repititive tasks of robot or in educational laboratories.

Table 6. Methodology comparison

Reference	Robot type	MSE	approach	(End effector / joint) max error
Current paper	4 DOF SCARA	1.99 e-7	Multiple NN	0.15 mm / 0.04 deg
Narayan, J. and Singla, A. [11]	4 DOF SCARA	-	ANFIS	0.415 mm in y-direction
Duka et al [6]	3 DOF planner	5.4 e-3	ANN	-
F.A.Raheem et. al. [3]	3 DOF Reis Robot	-	MLP	0.15 deg in joint two

Source of funding: *This research received no external funding.*

Author contributions: *research concept and design, Z.H.R., R.A.S.; Collection and/or assembly of data, Z.H.R., M.S.H.; Data analysis and interpretation, Z.H.R., R.A.S.; Writing the article, Z.H.R., M.S.H.; Critical revision of the article, Z.H.R., M.S.H.; Final approval of the article, Z.H.R., R.A.S.*

Declaration of competing interest: *The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.*

REFERENCES

1. Xiao F, Li G, Jiang D, Xie Y, Yun J, Liu Y. An effective and unified method to derive the inverse kinematics formulas of general six-DOF manipulator with simple geometry. Mechanism and Machine Theory 2021; 159: 104265. https://doi.org/10.1016/j.mechmachtheory.2021.104265.
2. Mohammed AA, Sunar M. Kinematics modeling of a 4-DOF robotic arm. Proceedings - 2015 International Conference on Control, Automation and Robotics, ICCAR 2015. 2015; 87–91. https://doi.org/10.1109/ICCAR.2015.7166008.
3. Raheem FA, Kareem AR, Humaidi AJ. Inverse kinematics solution of robot manipulator end-effector position using multi-neural networks. Engineering and Technology Journal 2016. 28; 34(7): 1360–8. https://etj.uotechnology.edu.iq/article_115849.html
4. Kenshimov C, Sundetov T, Kunelbayev M, Sarzhan M, Kutubayeva M, Amandykuly A. Developing application techniques of kinematics and simulation model for InMoov robot. Eastern-European Journal of Enterprise Technologies. 2022; 30;4(7(118) SE-Applied mechanics): 79–88. https://doi.org/10.15587/1729-4061.2022.261039.
5. Jha P, Biswal BB. A neural network approach for inverse kinematic of a SCARA manipulator. IAES International Journal of Robotics and Automation (IJRA). 2014; 3(1): 52–61. https://doi.org/10.11591/ijra.v3i1.3201.
6. Duka AV. Neural network based inverse kinematics solution for trajectory tracking of a robotic arm. Procedia Technology. 2014; 12: 20–7. https://doi.org/10.1016/j.protcy.2013.12.451.
7. Almusawi ARJ, Dülger LC, Kapucu S. A new artificial neural network approach in solving inverse kinematics of robotic arm (Denso VP6242). Comput Intell Neurosci 2016. https://doi.org/10.1155/2016/5720163.
8. Çabuk N, Bakırcıoğlu V. Altı Serbestlik Dereceli Bir Aydınlatma Manipülatörünün Yapay Sinir Ağları Temelli Ters Kinematik Çözümü ve Benzetimi. Gazi Üniversitesi Fen Bilimleri Dergisi Part C: Tasarım ve Teknoloji. 2018; (January): 117–25. https://doi.org/10.29109/http-gujsc-gazi-edu-tr.328422
9. Aravinddhakshan S, Apte S, Akash SM. Neural network based inverse kinematic solution of a 5 DOF Manipulator for industrial application. J Phys Conf Ser 2021; 1969(1). https://doi.org/10.1088/1742-6596/1969/1/012010.
10. Elawady WM, Bouteraa Y, Elmogy A. An adaptive second order sliding mode inverse kinematics approach for serial kinematic Chain robot manipulators. Robotics. 2020; 9(1). https://doi.org/10.3390/robotics9010004.
11. Narayan J, Singla A. ANFIS based kinematic analysis of a 4-DOFs SCARA robot. 2017 4th International Conference on Signal Processing, Computing and Control (ISPCC). IEEE; 2017. 205–11. https://doi.org/10.1109/ISPCC.2017.8269676.
12. Refaai MRA. Using multiple adaptive neuro-fuzzy inference system to solve inverse kinematics of SCARA Robot. 18th IEEE International Multi-Conference on Systems, Signals and Devices, SSD 2021. 2021; 154–9. https://doi.org/10.1109/SSD52085.2021.9429498.
13. Demby's J, Gao Y, DeSouza GN. A study on solving the inverse kinematics of serial robots using artificial neural network and fuzzy neural network. 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). IEEE 2019; 1–6. https://doi.org/10.1109/FUZZ-IEEE.2019.8858872.
14. Dejan. SCARA robot 3D model. 2020. [available] https://thangs.com/designer/m/3d-model/38897.
15. Sciavicco L, Siciliano B. Modelling and control of robot manipulators. Springer Science & Business Media; 2012.
16. Xia C, Sun C, Han J. Design and structural parameter optimization of airborne horticultural multi-DOF Manipulator. Journal of Applied Science and Engineering 2020; 23(3): 531–8. https://doi.org/10.6180/jase.202009_23(3).0017.
17. Spong MW, Hutchinson S, Vidyasagar M. Robot Modeling and Control 1st ed. Vol. 9. JOHN WILEY & SONS, INC.; 2013.
18. Atify M, Bennani M, Abouabdellah A. Structure optimization of a hexapod robot. International Journal on Technical and Physical Problems of Engineering. 2022; 14(1): 42–9.
19. Moretti CB. Neurona library for arduino. 2016. [available] http://www.moretticb.com/Neurona/#reference-mlp-forward.
20. McCauley M. AccelStepper library for Arduino. [available] http://www.airspayce.com/mikem/arduino/AccelStepper/index.html.

Zaid H. RASHID
was born in Baghdad, Iraq in Januaray 1986. He received the B.Sc. of mechanical engineering from University of Technology, Iraq in 2007. He has the M.Sc. and the Ph.D. degree in applied mechanics from the same university, in 2010 and 2019, respectively. Currently, he is a lecturer in mechanical techniques department of Technical Institute of Mussaib at University of Al-Furat Al-Awsat Technical University. He is interested in dynamics, robotics and control. He had published many papers in Iraqi academic and international Journals.
e-mail: zhr.1986@atu.edu.iq

Riyadh A. SARHAN
was born in Baghdad in Augest 1985. He received the B.Sc., M.Sc. and the Ph.D. degree from mechanical engineering department of University of Technology, Iraq in 2007, 2011, and 2019, respectively. Currently, he is a lecturer in mechanical techniques department of Technical Institute of Mussaib at University of Al-Furat Al-Awsat Technical University. He is interested in materials, robotics and CAE design. He had

published many papers in Iraqi academic and international Journals.
e-mail: sarhan.riyadh@atu.edu.iq

Mohammed S. HASSAN was born in Babylon in April 1983. He received his B.Sc. degree in mechanical engineering from the University of Babylon in 2006, and the M.Sc. degree in applied mechanics from the same university in 2016. He is currently a lecturer in the University of Al-Furat Al-Awsat Technical University, Technical Institute of Mussaib. He is interested in software analysis packages like ANSYS and system design. He had published many papers in Iraqi academic and international Journals.
e-mail: hs.muhamad@atu.edu.iq