Development of an Artificial Intelligent Approach in Adapting the Characteristic of Polynomial Trajectory Planning for Robot Manipulator

Ze Han. Ang, Chun Kit. Ang, Wei Hong. Lim, Lih Jiun. Yu, and Mahmud Iwan Solihin
Mechanical Engineering Department, UCSI University, Kuala Lumpur, Malaysia
Email: Zehanx@hotmail.com, {angck, limwh, yulj, mahmudis}@ucsiuniversity.edu.my

***Abstract*—The trajectory planning of robot manipulator can be calculated by using the mathematical approach when the type of trajectories is known. However, the conventional mathematical method becomes prohibitive because of the complicated mathematical equation and derivation. This research introduces the use of artificial neural networks (ANN) to overcome these limitations by solving nonlinear functions and adapting the characteristics of trajectory planning. A virtual three-degree-of-freedom (DOF) robot manipulator is exploited in this research. The analysis and selection of hyper-parameter for ANN will go through in order to get the optimum performance for ANN. Finally, sample data will be used to evaluate the robustness of the developed ANN topology by comparing the actual results (mathematical approach) with ANN results.**

***Index Terms*—artificial neural networks, forward kinematics, trajectory planning, robotic manipulator**

I. Introduction

In recent years, robot manipulators are playing increasingly significant roles in the engineering research field due to its wide-range applications. However, the robot manipulator is hardly autonomous because they need preliminary configuration such as path planning and trajectory planning to accomplish particular tasks [1]. A key concept in the research of robotic manipulators is trajectory planning. The trajectory planning is defined as the motion in term of joint angle, velocity, acceleration and jerk with the time interval for each robot joint [2]. Its advantage is the possibility to reach a coordinate with the smoothest sequence of movements for accurate tracking and reduces the stresses to the arm structure and actuators [3].

Robot manipulator possess high complexity and nonlinearity characteristics. The nonlinearity characteristics increase the difficulty of the mathematical derivation. When the motion of the end-effector needs to be determined, the mathematical expression for trajectory planning and forward kinematics is required to be derived and computed [4]. However, the conventional mathematical method becomes prohibitive because of the complicated mathematical equation and derivation.

Previous researchers have investigated a lot of primitive trajectories, such as third-order polynomial trajectory, fifth-order polynomial trajectory and seventh-order polynomial trajectory which possess different characteristics. Polynomials are a common method for defining robot trajectory. Using polynomials in real time control applications can change the trajectory in real-time by redefining the polynomials. The Third-order polynomial can be applied when the position and the velocity at the initial and final point are known, but it will produce linear acceleration profile that leads to spikes jerk at the initial and final movement of the robot manipulator [5]. The fifth-order polynomial is preferable than third-order polynomial because it solves the linearity of the acceleration and generates a smooth acceleration profile [6]. Jerk is the general desired criteria for smooth motion trajectory planning of robotic manipulators in the industry [7]. Seventh-order polynomial interpolation can generate a minimum-time smooth motion trajectory with zero jerk at the beginning and end points of the trajectory [7]. In conclusion, a higher order of polynomial equation will provide higher accuracy in trajectory generation but required more complexity in designing the trajectory [6].

The main focus of this paper is to model an Artificial neural network (ANN) to predict the characteristics of trajectory planning and motion of a virtual three-DOF robot manipulator without deriving any trajectory and forward kinematics equations. Scarselli and Tsoi had proved that feedforward neural network is a type of ANN that able to approximate generic classes of functions, including polynomial that is dense in the continuous functions with a fixed number of hidden layer neurons with nonlinear transfer functions enable the neural network to adapt nonlinear relationships within input and output data [8].

One of the major challenges in the design of a neural network is the selection of hyper-parameter with minimal error and highest accuracy [9]. The training set and error are likely to be high before learning begins. During training, the network adapts to decrease the error on the training patterns. The accuracy of training is determined

Manuscript received May 7, 2019; revised December 17, 2019.

doi: 10.18178/ijmerr.9.3.408-414

by the hyper-parameters under consideration. The hyper-parameters include NN architecture, number of hidden neurons in the hidden layer and activation function [9]. In 2010, Doukim et al. proposed a technique to find the number of hidden neurons in MLP network using coarse-to-fine search technique which is applied in skin detection. This technique includes binary search and sequential search. This implementation is trained by 30 networks and searched for the lowest MSE [10]. Another approach to fix hidden neuron is the sequential orthogonal approach (SOA) proposed by Sun. J. This approach is about adding hidden neurons one by one increasingly sequentially until the error is sufficiently small [11].

II. Methodology

A. Forward Kinematics

To derive the forward kinematics three-degrees-of-freedom (DOF) robot manipulator, the frame must be assigned to each link by starting from the base frame to the end-effector frame. Fig. 1 shows the robot arm links, joints and assignment of frame according to the Denavit & Hartenberg (D-H) notations.

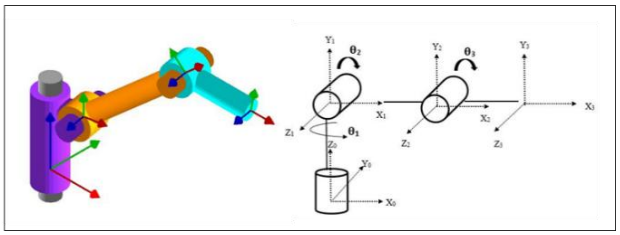

Figure 1. Frame assignment according to the (D-H) notations.

TABLE I. D-H Parameter Obtain Base on Frame Assignment

Frame	Link length (m) ,(a_i)	Link offset (m) ,(d_i)	Joint angle (°), (θ_i)	Link twist (°), (α_i)
0-1	0.05	0.1	Θ_1	90
1-2	0.2	0	Θ_2	0
2-3	0.1	0	Θ_3	0

Table I shows the D-H parameter obtain based on the frame assignment and configuration of the 3 DOF robot manipulator. The parameter values of a_i, d_i, θ_i and α_i obtained are substituted in the homogeneous transformation matrix A_i shown in (1). Forward kinematics equations can be obtained by multiplying A1, A2, A3 as shown in (2).

$$A_i = \begin{bmatrix} C\theta_i & -S\theta_i C\alpha_i & S\theta_i S\alpha_i & a_i C\theta_i \\ S\theta_i & C\theta_i C\alpha_i & -C\theta_i S\alpha_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

$$A1.A2.A3 = \begin{bmatrix} C_1C_2C_3 - S_2S_3C_1 & -S_3C_1C_2 - S_2C_1C_3 & S_1 & 0.1C_1C_2C_3 - 0.1C_1S_2S_3 + 0.2C_1C_2 + 0.05C_1 \\ S_1C_2C_3 - S_1S_2C_2 & -S_1S_2C_2 - S_1S_2C_3 & -C_1 & 0.1S_1C_2C_3 - 0.1S_1S_2S_3 + 0.2S_1C_2 + 0.05S_2 \\ S_2C_3 + C_2S_3 & -S_{23} + C_2C_3 & 0 & 0.1C_3S_2 - 0.1S_3C_2 + 0.2S_2 + 0.1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

B. Trajectory Planning

Trajectory planning for point to point motion indicate the joint angle as a function of time between specified points. Velocity, acceleration and jerk along the trajectory can be calculated by differentiating the polynomial equation with respect to time. In this paper, the trajectory planning is derived by considering acceleration, velocity and jerk as zero at the initial and final position at the time interval of 10 seconds. The initial joint angle (θ_0) represent value on the starting time t_0=0. The final joint angle (θ_f) represent value on the final time t_f as shown in (3).

$$\begin{cases} \theta(0) = \theta_0 \\ \theta(t_f) = \theta_f \\ \dot{\theta}(0) = 0 \\ \dot{\theta}(t_f) = 0 \end{cases} \quad (3)$$

The overall Third-order polynomial equation is shown in (4). The coefficient of third-order polynomial trajectory is shown as (5) by using the assumption of (3).

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad (4)$$

$$\begin{aligned} a_0 &= \theta_0 \\ a_1 &= 0 \\ a_2 &= \frac{3}{t_f{}^2}(\theta_f - \theta_0) \\ a_3 &= -\frac{2}{t_f{}^3}(\theta_f - \theta_0) \end{aligned} \quad (5)$$

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 \quad (6)$$

$$\begin{aligned} a_0 &= \theta_0 \\ a_1 &= \dot{\theta}_0 = 0 \\ a_2 &= \tfrac{1}{2}(\ddot{\theta}_0) = 0 \\ a_3 &= \frac{1}{2t_f{}^3}(20)(\theta_f - \theta_0) \\ a_4 &= \frac{1}{2t_f{}^4}(30)(\theta_0 - \theta_f) \\ a_5 &= \frac{1}{2t_f{}^5}(12)(\theta_f - \theta_0) \end{aligned} \quad (7)$$

The overall fifth-order polynomial equation is shown in (6). The coefficient of fifth-order polynomial trajectory is shown as (7) by using the assumption of (3). The acceleration of fifth-order polynomial trajectories will be considered as zero at the starting point and the final point.

The overall seventh-order polynomial equation is shown in (8). The coefficient of seventh-order polynomial trajectory is shown as (9) by using the assumption of (3). The acceleration and jerk of seventh-order polynomial trajectories will be considered as zero at the starting point and the final point.

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 + a_6 t^6 + a_7 t^7 \quad (8)$$

$$
\begin{aligned}
a_0 &= \theta_0 \\
a_1 &= \dot{\theta}_0 = 0 \\
a_2 &= \tfrac{1}{2}\left(\ddot{\theta}_0\right) = 0 \\
a_3 &= \tfrac{J_0}{6} = 0 \\
a_4 &= \tfrac{1}{6{t_f}^4}(210)(\theta_0 - \theta_f) \\
a_5 &= \tfrac{1}{2{t_f}^5}(-168)(\theta_f - \theta_0) \\
a_6 &= \tfrac{1}{6{t_f}^6}(420)(\theta_f - \theta_0) \\
a_7 &= \tfrac{1}{6{t_f}^7}(-120)(\theta_f - \theta_0)
\end{aligned} \tag{9}
$$

C. Data Collection

After deriving the mathematical trajectory equation. The data collection process takes place using mathematical approach for polynomial trajectory equation and forward kinematics. The data set contains the input variable and output variable. For third-order polynomial trajectories, fifth-order polynomial trajectories and seventh-order polynomial trajectory equation, the input variable of initial joint angle(θ_0), final joint angle(θ_f) and time interval(t) are substitutes into the coefficient equation to obtain the coefficient value. Next, coefficient value and time interval (t) are substituted into trajectory equation to obtain the output variable of joint angle $\theta(t)$, velocity $\dot{\theta}(t)$ and acceleration $\ddot{\theta}(t)$ of the joint at each second. For seventh-order polynomial trajectory, the additional value of jerk will be collected. The forward kinematics can be calculated to obtain the X, Y and Z Cartesian coordinate of the robot end effector at time interval by using each joint angle position. The data need to be normalized into the interval [-1, 1], to achieve a better training result. Equation 10 is adopted in this work to perform the normalization, where x is original data and X normalization is normalized data.

$$
X_{\text{normalization}} = \tfrac{X}{100} \tag{10}
$$

D. Topology of Neural Network

Although feedforward neural network is able to solve the nonlinear functions, it is still difficult for a single feedforward neural network to solve the different trajectory planning simultaneously. Therefore, the topology of multiple NNs proposed here can be used to predict joint angle, velocity, acceleration of each joint for different trajectory planning and the motion of the robot end effector at the time interval for a three-DOF robot manipulator shown in Fig. 1. The topology of multi-NNs proposed shown in Fig. 2. The architecture consists of four feedforward neural network NN_1, NN_2, NN_3, NN_4. This research will fix the type of trajectory planning for the first robot joint is third-order polynomials, second robot joint is fifth-order polynomials and third robot joint is seventh-order polynomial. Each NN can only predict one type of trajectory planning. NN_1 will predict third-order polynomials, NN_2 will predict fifth-order polynomials and NN_3 will predict seventh-order polynomial trajectory planning characteristics without deriving trajectory equations.

There are two main parts in the structure. Firstly, NN_1, NN_2 and NN_3 are used to predict the joint angle, velocity and acceleration of each joint. For NN_3, additional output variable of jerk ($J(t)$) will be predicted. The inputs for the NN_1, NN_2, NN_3 are the initial joint angle ($\theta(t_0)$), final joint angle ($\theta(t_f)$) and time interval (t) of each joint. Then, NN_4 is used to solve forward kinematics. The inputs for NN_4 are the angle value of joint 1 $\theta_1(t)$, angle value of joint 2 $\theta_2(t)$ and angle value of joint 3 $\theta_3(t)$ which predicted by NN_1, NN_2 and NN_3. The NN_4 is able to determine the end effector coordinate in Cartesian Space X, Y, Z at the time interval (t).

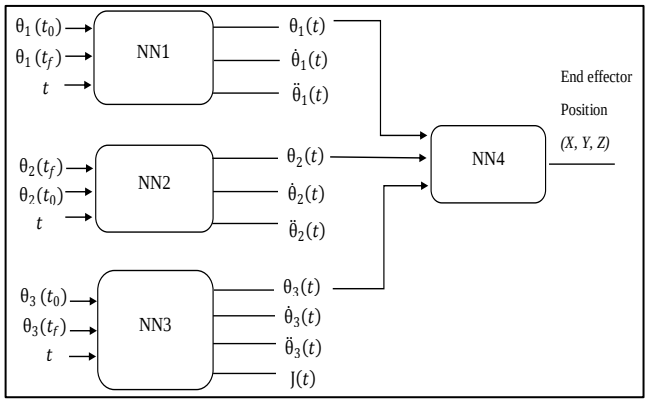

Figure 2. Multi-NNs topology.

E. Method for Selecting Transfer Function

For the hidden layers, a nonlinear transfer function should be used because the trajectory planning possess nonlinearity characteristics. The tan-sigmoid transfer function generates outputs between -1 and 1. The log-sigmoid transfer function generates outputs between 0 and 1. The transfer function will be limiting the amplitude of the output, by referring the range of the normalized dataset which is from -1 and 1. So, the suitable transfer function use in hidden layers is tan-sigmoid transfer function to constrain the outputs of a network between -1 and 1. For the output layers, the transfer function can be nonlinear or linear but need to generate outputs between -1 and 1. In this case, tan-sigmoid transfer function or purelin can be selected. Each neural network was trained and tested using the two different combinations of the transfer functions for hidden layer and output layer. First combination is tan-sigmoid for hidden layer and purelin for the output layer. Second combination is tan-sigmoid for hidden layer and tan-sigmoid for the output layer.

F. Method for Selecting Number of Neurons in Hidden Layers

Trial and error method is proposed by using the sequential approach. This approach begins by selecting a small number of hidden neurons that begin with five hidden neurons. After that train and test the neural network, the value of mean square error (MSE) and correlation coefficient(R) is recorded. Then increased the

number of hidden neurons and compared the mean square error(MSE) and correlation coefficient(R) with previous results. Repeat the above procedure until training and testing improved. When the mean square error (MSE) is considered very low, the hidden neurons numbers is adding one by one until the mean square error (MSE) is sufficiently small.

III. Results and Discussion

A. Analysis and Selection of Transfer Function

Two combinations of transfer function for hidden layer and output layers have been implemented and compared by using performance analysis for NN_1, NN_2, NN_3 and NN_4. The parameter for all NN and division of data for training and validation were same in order to make a comparison between each combination of transfer functions for hidden layer and output layer. Table II shows the results that tan-sigmoid for hidden layer and purelin for the output layer produce lower mean squared error (MSE) and higher correlation coefficient(R) value for NN_1, NN_2, NN_3 and NN_4. So, tan-sigmoid for hidden layer and purelin for the output layer would be chosen as the configuration of all neural network. Tansig-tansig combination for hidden and output layers also provided acceptable results for function approximation problem.

TABLE II. Comparison of Different Transfer Function

Neural network	(Hidden-Output)	MSE	Correlation coefficient(R)
NN_1	Tansig- Tansig	1.65×10^{-6}	0.99998
NN_1	Tansig-Purelin	4.15×10^{-7}	0.99999
NN_2	Tansig- Tansig	6.74×10^{-6}	0.99996
NN_2	Tansig-Purelin	2.29×10^{-6}	0.99988
NN_3	Tansig-Tansig	1.45×10^{-6}	0.99998
NN_3	Tansig-Purelin	4.84×10^{-7}	0.99999
NN_4	Tansig- Tansig	8.09×10^{-6}	0.99993
NN_4	Tansig-Purelin	2.48×10^{-6}	0.99997

B. Analysis and Selection of Neurons Number

The proposed method described in methodology was implemented and trained using NN_1, NN_2, NN_3 and NN_4 and searched for lowest mean squared error(MSE) and highest correlation coefficient (R) based on validation data to find the near to optimal number of neurons hidden layers after training the neural network with different number of neurons. The parameters for all neural network and division of data for training and validation were same in order to make comparison between different neurons number in hidden layers.

The best neurons number in hidden layer for NN_1, NN_2, NN_3, and NN_4 were 18, 20, 22 and 22 respectively. However, the same fixed neurons number should be selected for all neural networks. An average value of neuron number was calculated as shown in Table III. Finally, all the neural network was decided to consist of 20 neuron number in the hidden layers. The network with 20 neurons in the hidden layers would be considered satisfactory since it results low mean square error (MSE) and high Correlation coefficient(R).

TABLE III. Selection of Best Neurons Number for All NNs

Neural network	Best Neurons number in hidden layer
NN_1	18
NN_2	20
NN_3	22
NN_4	20
	Average:20

C. Performance Analysis of Neural Network Using Sample Data

After determining the best hyper-parameters for all neural network. NN_1, NN_2, NN_3 and NN_4 were trained using tan-sigmoid for hidden layer and purelin for the output layer. The neurons number in hidden layer set as 20 for all neural networks. The Levenberg-Marquardt (trainlm) based on quasi-Newton algorithms was used as the training algorithm in the neural network. The training and simulations were performed using MATLAB (R2016a). The mean square error (MSE) of the trained results was close to 0 (approximately $1.45\text{x}10^{-6}$). To test the reliability of the developed multiple neural network topology, five samples which consisted of total 15 set data were used for evaluating the performance of the neural network. Each sample had three set of data. First set of data was used for predicting the movement of first robot joint which possessed third-order polynomial trajectory planning. Second set of data was used for predicting the movement of second robot joint which possessed fifth-order polynomial trajectory planning and the third set of data was used for predicting the third robot joint which possessed seventh-order polynomial trajectory planning. The time interval was between 0 seconds until 10 seconds. The inputs for NN_4 were the inputs variable from joint 1 angle value, joint 2 angle value and joint 3 angle value which predicted by NN_1, NN_2 and NN_3 respectively. The output variable of NN4 was the robot end effector coordinate X, Y, Z in Cartesian space at the time interval (t).

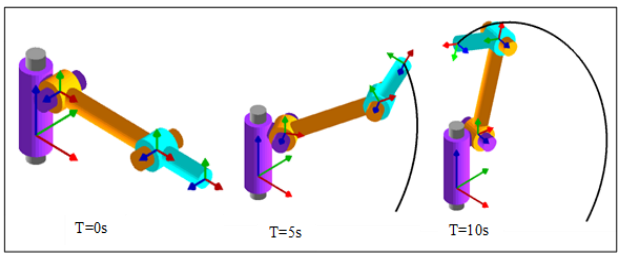

Figure 3. Motion of virtual robot manipulator for sample 1.

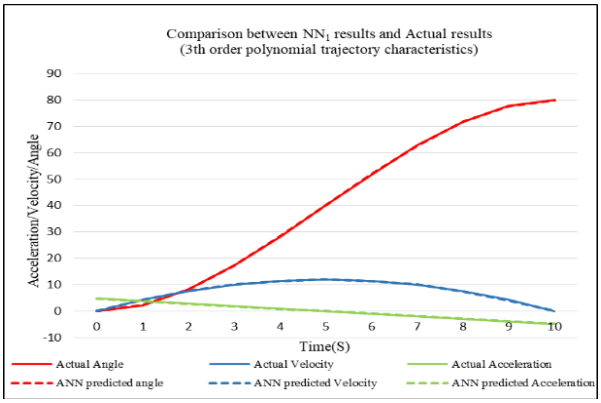

Figure 4. Comparison between NN_1 results and Actual results of sample 1.

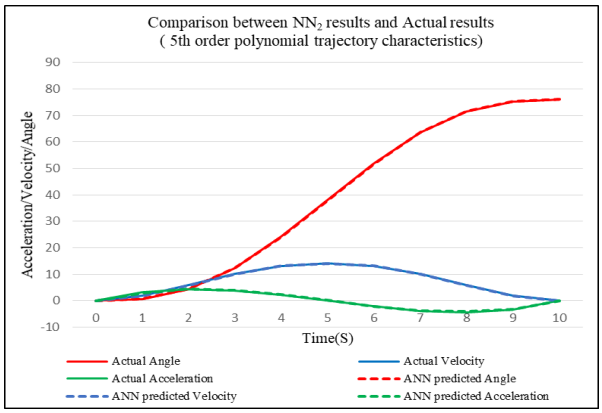

Figure 5. Comparison between NN_2 results and Actual results of sample 1.

For sample 1, the first joint moved from initial angle of 0 ° until 80 °, second joint moved from initial angle of 0 ° until 76 ° and third joint move from initial angle of 0 ° until 88 °. The virtual robot manipulator for sample 1 took 10 seconds to complete its motion as shown in Fig. 3. The characteristics of trajectory planning and the motion of the end effector robot manipulator was predicted. The generated trajectory profile and robot hand motion had been plotted for the comparison of NN results and actual results as shown in Fig. 4, Fig. 5, Fig. 6 and Fig. 7. The solid line refers to actual results and dotted line refer to neural network results. The four graphs show that dotted lines are fitted very close to the solid line. In other words, the neural network able to predict the results precisely.

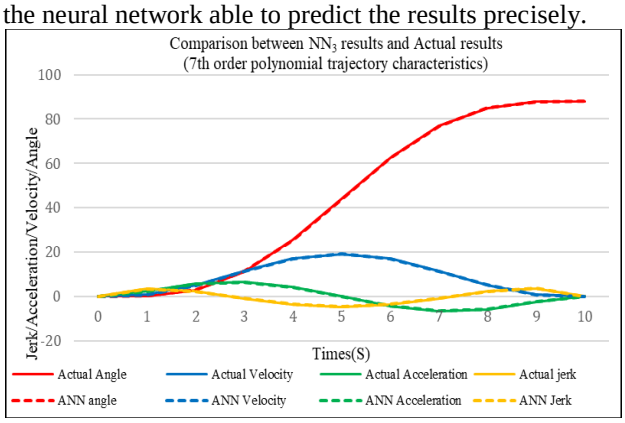

Figure 6. Comparison between NN_3 results and Actual results of sample 1.

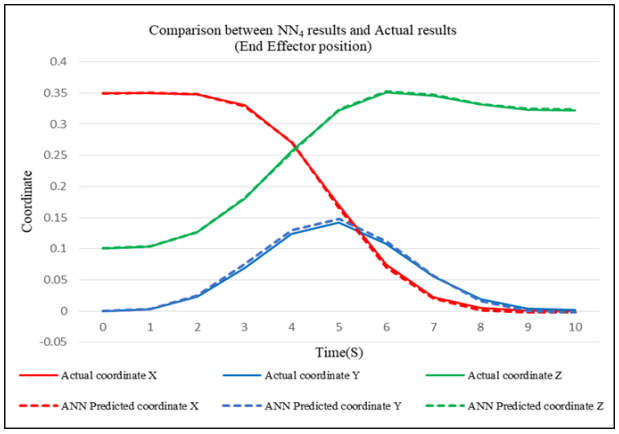

Figure 7. Comparison between NN_4 results and Actual results of sample 1.

Fig. 8 shows the graph of all sample performance analysis using RMSE and Fig. 9 shows the graph of all sample performance analysis using R^2. The results showed that the developed multi-NN topology produced a maximum of 0.04 RMSE and a minimum of 0.3 RMSE. The maximum coefficient of determination (R^2) obtained was 0.9998. The developed ANN topology was able to adapt the characteristics of trajectory planning and predict the motion of end effector for a virtual three-DOF robot manipulator.

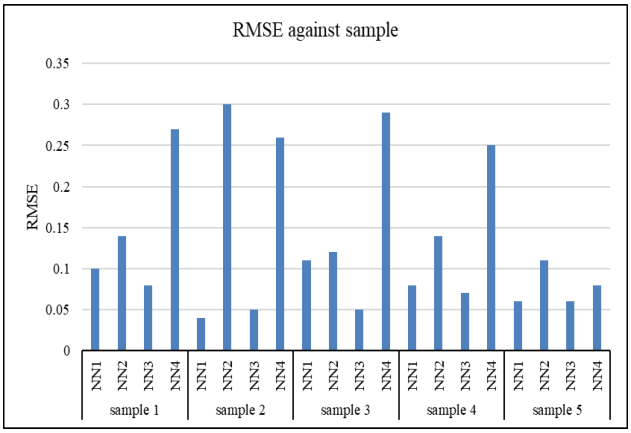

Figure 8. Performance analysis using RMSE.

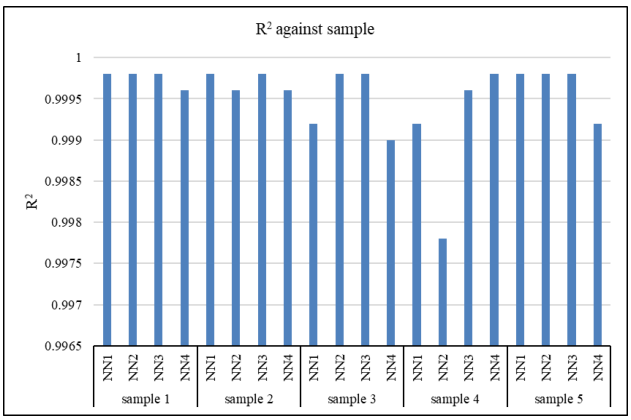

Figure 9. Performance analysis using R^2.

IV. Conclusion

A multi-NNs topology that made up of NN_1, NN_2, NN_3 and NN_4 had been proposed in this research. After validating the neural network with a different combination of the transfer function, tan-sigmoid for hidden layer and purelin for the output layer has been chosen as the transfer function of all neural network in this research. The network with 20 neurons in the hidden layers would be considered satisfactory since it results low mean square error (MSE) and high Correlation coefficient (R). Finally, the sample data were used to test the robustness of the multiple neural network topology which produced a maximum of 0.04 RMSE and minimum of 0.3 RMSE. The maximum coefficient of determination (R^2) obtained was 0.9998. The results show that the proposed multi-NNs topology able to adapt and predict the nonlinear characteristics of third-order polynomial, fifth-order polynomial, seventh-order polynomial as well as the motion of the end effector for a virtual three-degrees-of-freedom(DOF) robot manipulator with high accuracy without having to solve the complex mathematical equation.

Conflict of Interest

The authors declare no conflict of interest for this submitted work.

Author Contributions

For this paper, Ze Han Ang contributed in conducting the actual and virtual experiments. At the same time, he was also the key person in collecting the data. Chun Kit Ang, his contribution was to analyze the data and to develop the topology of the neural network. Mahmud Iwan and Wei Hong Lim contributed on the algorithm part. Lastly, Lih Jiun Yu contributed in paper writing and all authors had approved the final version.

References

[1] S. Chiddarwar, and N. Babu, "Optimal trajectory planning for industrial robot along a specified path with payload constraint using trigonometric splines," *International Journal of Automation and Control*, vol. 6, no.1, pp.39-65,2012.

[2] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Trajectory Planning in Robotics," *Mathematics in Computer Science*, vol. 6, no.3, pp.269-279, June 2012.

[3] B. R Munson, D. F Young, and T.H Okiishi, *Fundamentals of Fluid Mechanics*, 5th Ed. John Wiley & Sons, pp. 123-124.

[4] R. J. Chandran, I. J. Raglend, and M. D. Anand, "Forward kinematic solution for a five joint robot using artificial neural network," *International Conference on Circuits, Power and Computing Technologies*, ICCPCT, 2014.

[5] S. H. Tang, C.K. Ang, K. A. Mohd Ariffin, and S. B. Mashohor, "Predicting the Motion of a Robot Manipulator with Unknown Trajectories Based on an Artificial Neural Network," *International Journal of Advanced Robotic Systems*, vol. 11, no. 10, pp.1-9, Sep 2014.

[6] M. Q. Mohammed, F. M. Muhammad, B. B. Mohd Bazli, S. A. Ali, "Comparative study between quintic and cubic polynomial equations based walking trajectory of exoskeleton system," *International Journal of Mechanical and Mechatronics Engineering*, vol. 17, no.4, pp. 43–51, August 2017.

[7] S. Kucuk, "Optimal trajectory generation algorithm for serial and parallel manipulators," *Robotics and Computer Integrated Manufacturing*, vol. 48, pp. 219–232,2017.

[8] F. Scarselli, and A. C. Tsoi, "Universal Approximation Using Feedforward Neural Networks: A Survey of Some Existing Methods, and Some New Results. Neural Networks," vol. 11, no.1, pp.15-37,1998.

[9] K. G. Sheela, and S. N. Deepa, "Review on Methods to Fix Number of Hidden Neurons in Neural Networks," *Mathematical Problems in Engineering*, vol.2013, pp. 1–11, May 2013.

[10] C. A. Doukim, J. A. Dargham, and A. Chekima, "Finding the number of hidden neurons for an MLP neural network using coarse to fine search technique, " in Proc. *10th International Conference on Information Science, Signal Processing and their Applications*, June 2010.

[11] J. Sun, "Neurocomputing Learning algorithm and hidden node selection scheme for local coupled feedforward neural network classifier," *Neurocomputing. Elsevier*, vol. 79, pp. 158–163,2012.

Ze Han. Ang received the B.Eng. Degree in Mechatronic Engineering from UCSI University, Kuala Lumpur in 2019.

He is currently an R&D Engineer in Vitrox Corporation berhad, Penang, Malaysia. His research interests include robotics system and artificial intelligent.

Chun Kit. Ang received the B.Eng. Degree in Mechatronic Engineering from UCSI University with First Class Honor, Kuala Lumpur in 2009 and PhD in Mechanical Engineering from University Putra Malaysia in 2014.

He is currently an Assistant Professor with the Faculty of Engineering, Technology and Built Environment, UCSI University, Kuala Lumpur, Malaysia. His research interests include soft computing and robotics system.

Mahmud Iwan. Solihin is an Assistant Professor in with the Faculty of Engineering, Technology and Built Environment, UCSI University, Malaysia. He specialized in applications of artificial intelligence/machine learning, optimizations, control systems, robotics and automation. In the past, he was a research assistant for an eScienceFund project under MOSTI (Ministry of Science Technology & Innovation).

He is also a professional member of the Institution of Engineers of Indonesia and recognized by AFEO (ASEAN Federation of Engineering Organisations) and APEC Engineer. His current research interest is mainly on applications of machine learning and optimization algorithms in various filed including agriculture, medical field, sensor networks, etc., toward sustainable engineering development in the future.

Wei Hong. Lim received the B.Eng. Degree in Mechatronic Engineering and PhD degree in Computational Intelligence from the Universiti Sains Malaysia, Penang, Malaysia, in 2011 and 2014, respectively. From 2015 to 2017, he was a Post-Doctoral Researcher with the Department of Electrical Engineering, National Taipei University of Technology, Taipei.

He is currently an Assistant Professor with the Faculty of Engineering, Technology and Built Environment, UCSI University, Kuala Lumpur, Malaysia. His research interests include demand side management and application of computational intelligence in engineering applications.

Lih Jiun. Yu is a researcher in Malaysia. She obtain both PhD in Material Science and BSc in Material Science from National University of Malaysia (UKM) in 2012 and 2008. Currently, she is a professional member of Institute of Materials Malaysia (IMM).

She is currently an Assistant Professor with the Faculty of Engineering, Technology and Built Environment, UCSI University, Kuala Lumpur, Malaysia. Her research interest is processing and characterisation of polymer composite.