Journal of Engineering Science and Technology
Vol. 16, No. 6 (2021) 4910 - 4919

3DOF ROBOT ARM CONTROL USING FUZZY NEURAL PETRI NET (FNPN) CONTROLLER

WAEL H. ZAYER[1,*], ABDUL-JABBAR F. ALI[2], ZAHRAA A. MAEEDI[3]

[1]Engineering Technical college-Missan, Southern Technical University, Missan, Iraq
[2,3]College of Engineering, Wasit's University, Wasit, Iraq
*Corresponding Author: wael.zayer@stu.edu.iq

Abstract

With robots performing many difficult tasks that humans cannot do because it needs strength and accuracy in working in real-time, controlling these robots to increase their production and accuracy at work became the centre of attention for all workers and researchers in this domain. The problem of the kinematics (forward kinematic, inverse kinematic) of the robot arm of 3DOF is discussed in this research. Where it introduces a method to solve this problem using an intelligence algorithm known as Fuzzy Neural Petri Net (FNPN). Where the equations of motion of dynamic and kinematics (forward and inverse) are derived; then these equations are building using the Simulink library in the Matlab program. The results of the simulation are used as training data to FNPN algorithm to train the data to get the desired outputs in both cases of kinematics and compare it with FNPN results based on the mean square error are given. The results of the comparison show that FNPN results were acceptable and close to the simulation results. Which led to an increase in the use of FNPN in many control circuits used in various applications and the reason for this is due to the flexible ability of FNPN to deal with different and spread data in a scattered manner in addition to the data arranged within a certain range.

Keywords: 3DOF Robotic arm, Forward kinematic, FNPN controller, Inverse kinematic, Motion control.

1. Introduction

The robot is a term used in different fields refers to any machine that can be controlled to do various type of application that humans cannot do. From a general view, the robots are a continuous field to robotics engineers about making machines able to complete tasks as skillfully and accurate that humans cannot do. The development in the robotics field became possible due to using microcontrollers and microprocessors combined artificial intelligence with sensors and actuators and servo motors [1]. The robotic arm is a mechanical device that consists of joints and links that gives solidness and strength [1]. This research focuses on the kinematics problem of the arm. The kinematics problem divides into the problem of inverse kinematic and the problem of the forward kinematic. The problem of the forward kinematic represented in find the end-effector position of the robot arm in space based on the angles of each joint in this arm. The inverse kinematic problem represents in finding the angles of each joint based on the position given of the arm. Most researcher focus on designed a controller to solve this problem and reduced the errors of motion. In research conducted by Duka [2] a solution approach is introduced to solve the problem of inverse kinematic of 3DOF robot arm using an Adaptive Neuro-Fuzzy Inference System(ANFIS), where the inverse kinematic data is derived based on the forward kinematic, and then this data learn using ANFIS.

Gupta et al. [3] showed a geometric approach is introduced to solve the problem of inverse kinematic for 3DOF robot arm; in this method, the position is giving by the user as an input processed through software represents the geometric approach. The software calculates the output depends on the input nature. This method decreases the complexity in the calculations of inverse kinematics.

The main aim of this study is to analyse the movement of a robot arm consist of three joints and three links in the (*x*, *y*, *z*)plane. Where discussed the kinematics problem represented in forward and reverse kinematics of the arm and find a solution using one of the artificial intelligence algorithms (FNPN). This algorithm combined the simple representation of the Petri net and neural network learning procedure to calculate the final position of the arm in the case of the forward kinematic and calculate the angles of the joints in the inverse kinematic case.

2. Forward Kinematics of 3DOF Robot Arm

Kinematics is the science that concern with the movement of the body without the taken effect of different parameters or forces that affect the motion [4]. Forward kinematics means finding the end-effect position based on the angles joints of the arm as referenced previously that the arm consists of links and joints. The problem is to obtain the position of the end-effector of the arm's [4]. DenavitHartenberg (DH) transformation is used to calculate the forward kinematic equations [5]. The forward kinematics of the 3DOF robot arm shown in Fig. 1 is calculated using DH transformation that calculates the position of the arm and its direction as following [6]:

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

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

$$z = sin\theta_3 * r \quad (3)$$

$$r = \sqrt{{k_1}^2 + {k_2}^2} \quad (4)$$

where $k_1 = l_1 + l_2\cos(\theta_2)$, $k_2 = l_2\sin(\theta_2)$

In Fig. 1, l_1, l_2, l_3 are the lengths of each joint in the arm, and $\theta_1, \theta_2, \theta_3$ and are the angles of these joints respectively.

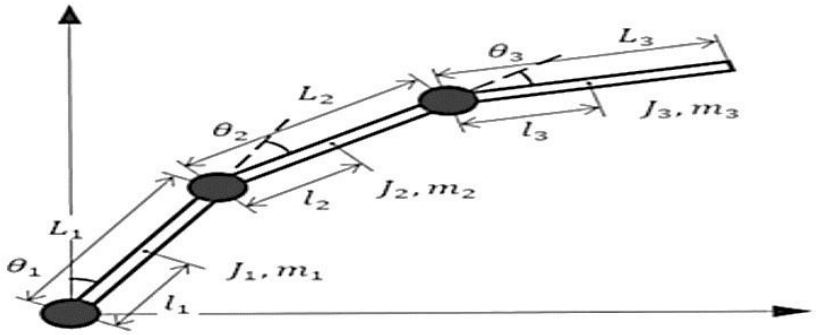

Fig. 1. Mass properties of 3DOF planar robot.

3. Inverse Kinematics of 3DOF Robot Arm

The opposite method to the forward kinematics is called inverse kinematics, where the position required of the arm is given (known) and the joints angles that give these locations are calculated, but the problem of inverse kinematic is that the positions can be reached by more than one solution where at a different time different angles could give the same position. The equations of the inverse kinematics [6] are calculated as follows:

$$\theta_1 = atan2\left(\frac{y}{r}, \frac{x}{r}\right) - atan2(k_2, k_1) \quad (5)$$

$$\theta_2 = atan2(\sin(\theta_2), \cos(\theta_2)) \quad (6)$$

$$cos\theta_2 = \frac{x^2 + y^2 - {l_1}^2 - {l_2}^2}{2l_1 l_2} \quad (7)$$

$$sin\theta_2 = \sqrt{1 - \cos^2(\theta_2)} \quad (8)$$

$$x = r * \cos(\theta_3) \quad (9)$$

$$\theta_3 = atan2(\sin(\theta_3), \cos(\theta_3)) \quad (10)$$

where $sin\theta_3 = \frac{z}{r}$ and $cos\theta_3 = \frac{x}{r}$

4. Dynamics of 3DOF

Dynamics is the study of motion. It describes how the motions happen and why when forces on huge bodies are applied. The dynamic equations of the robot arm are derived using the Lagrange-Euler formulation and parameters values of the dynamic model of the arm in Table 1 [7]. The deriving dynamic model equations of the 3DOF robot arm are:

$$\tau_1 = \left[m_1 {lc_1}^2 + m_2\left({l_1}^2 + {lc_2}^2 + 2l_1 lc_2 C_2\right) + m_3\left({l_1}^2 + {l_2}^2 + {lc_3}^2 + 2l_1 l_2 C_2 + 2l_1 lc_3 C_{23} + 2l_2 lc_3 C_3\right) + I_1 + I_2 + I_3\right]\ddot{\theta}_1 + \left[m_2\left({lc_2}^2 + l_1 lc_2 C_2\right) + m_3\left({l_2}^2 + {lc_3}^2 + l_1 l_2 C_2 + l_1 lc_3 C_{23} + 2l_2 lc_3 C_3 + I_2 + I_3\right]\ddot{\theta}_2 + \left[m_3\left({lc_3}^2 + l_1 lc_3 C_{23} + l_2 lc_3 C_3\right) + I_3\right]\ddot{\theta}_3 + \left[-m_2 l_1 lc_2\left(2\dot{\theta_1} + \dot{\theta_2}\right)S_2\dot{\theta_2}\right] - m_3\left[l_1 l_2\left(2\dot{\theta_1} + \dot{\theta_2}\right)S_2 + l_1 lc_3\left(2\dot{\theta_1} + \dot{\theta_2} + \dot{\theta_3}\right)S_{23} +\right.$$

$$l_2lc_3(2\dot{\theta}_1 + 2\dot{\theta}_2 + \dot{\theta}_3)S_3] + g[C_1(m_1lc_1 + m_2l_1 + m_3l_1) + C_{12}(m_2lc_2 + m_3l_2) + C_{123}(m_3lc_3)] \quad (11)$$

$$\tau_2 = [m_2({lc_2}^2 + l_1lc_2C_2) + m_3({l_2}^2 + {lc_3}^2 + l_1l_2C_2 + l_1lc_3C_{23} + 2l_2lc_3C_3 + I_2 + I_3]\ddot{\theta}_1 + [m_2{lc_2}^2 + m_3({l_2}^2 + {lc_3}^2 + 2l_2lc_3C_3) + I_2 + I_3]\ddot{\theta}_2 + [m_3({lc_3}^2 + l_2lc_3C_3) + I_3]\ddot{\theta}_3 - m_2\left[l_1lc_2\left({\dot{\theta}_1}^2 + \dot{\theta}_1\dot{\theta}_2\right)S_2 + l_1lc_1\dot{\theta}_1S_2\dot{\theta}_2\right] - m_3[l_1l_2\left({\dot{\theta}_1}^2 + \dot{\theta}_1\dot{\theta}_2\right) + l_1lc_3\left({\dot{\theta}_1}^2 + \dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_1\dot{\theta}_3\right)S_{23} + l_2lc_3\left({\dot{\theta}_1}^2 + 2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_1\dot{\theta}_3 + {\dot{\theta}_2}^2 + \dot{\theta}_2\dot{\theta}_3\right)S_3 + l_1l_2\dot{\theta}_1S_2\dot{\theta}_2 + l_1lc_3\dot{\theta}_1\dot{\theta}_2S_{23} + l_2lc_3(2\dot{\theta}_1 + 2\dot{\theta}_2 + \dot{\theta}_3)S_3\Big] + g[(m_2lc_2 + m_3l_2)C_{12} + m_3lc_3C_{123}] \quad (12)$$

$$\tau_3 = [m_3({lc_3}^2 + l_1lc_3C_{23} + l_2lc_3C_3) + I_3]\ddot{\theta}_1 + [m_3({lc_3}^2 + l_2lc_3C_3) + I_3]\ddot{\theta}_2 + [m_3lc_3 + I_3]\ddot{\theta}_3 - m_3[l_1lc_3\left({\dot{\theta}_1}^2 + \dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_1\dot{\theta}_3\right)S_{23} + l_2lc_3\left({\dot{\theta}_1}^2 + 2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_1\dot{\theta}_3 + {\dot{\theta}_2}^2 + \dot{\theta}_2\dot{\theta}_3\right)S_3 + l_1lc_3\dot{\theta}_1\dot{\theta}_3S_{23} + l_2lc_3(\dot{\theta}_1 + \dot{\theta}_2)S_3\dot{\theta}_3\Big] + g(m_3lc_3C_{123}) \quad (13)$$

In Eqs. (11) and (12); τ referred to the torque of the joint, $m,1,\theta$ referred to the mass, length, and angle of each joint and I is the moment of inertia, c means cosine and s mean sine, g is the gravity.

Table 1. Parameters values of the dynamic model of the arm [7].

Links	T(N.m)	M(kg)	1(m)	1c(m)	I(kgm^2)	g(ms^{-2})
1	0.2	1	0.2	0.1	0.5	9.81
2	0.2	1	0.2	0.1	0.5	9.81
3	0.2	1	0.2	0.1	0.5	9.81

Figure 2 shows the simulation of the dynamic model of the arm, where the Eqs. (11-13) is written using the function block from the Simulink library in Matlab. Figure 3 represent the forward model equations and the first part from the figure (out1,out2,out3) is the dynamic model in Fig. 2 since to calculate the position of the end-effector the angles from the dynamic model is needed. Figure 4 represents the inverse kinematic equations where the first part of the model is the outputs of the forward kinematic model that is it represents the Fig. 3.

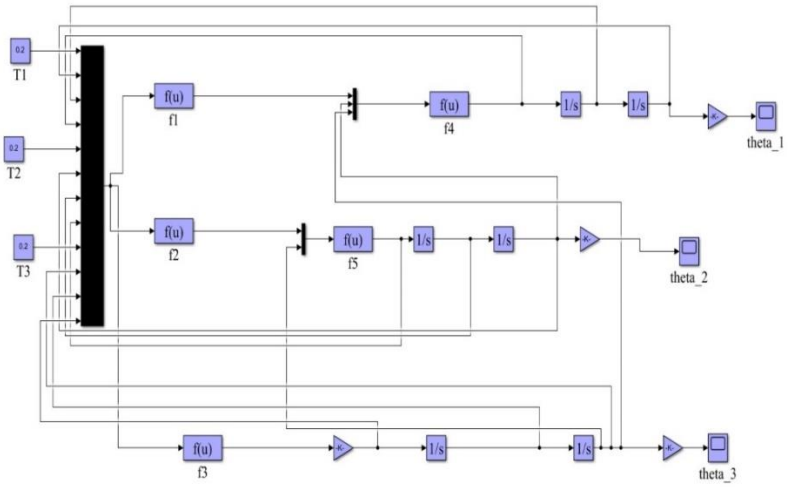

Fig. 2.Thesimulation of the dynamic model of 3DOF robot arm.

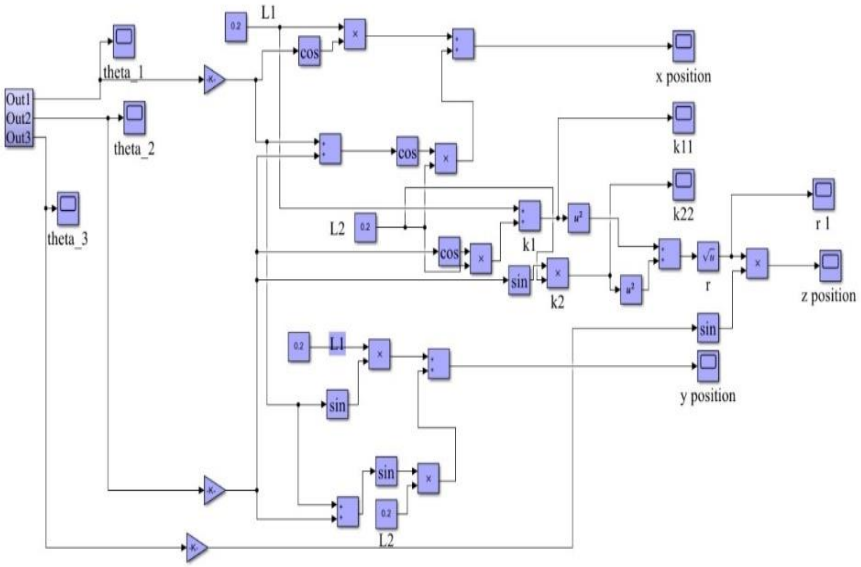

Fig. 3.The Simulation model of the forward kinematics of the arm.

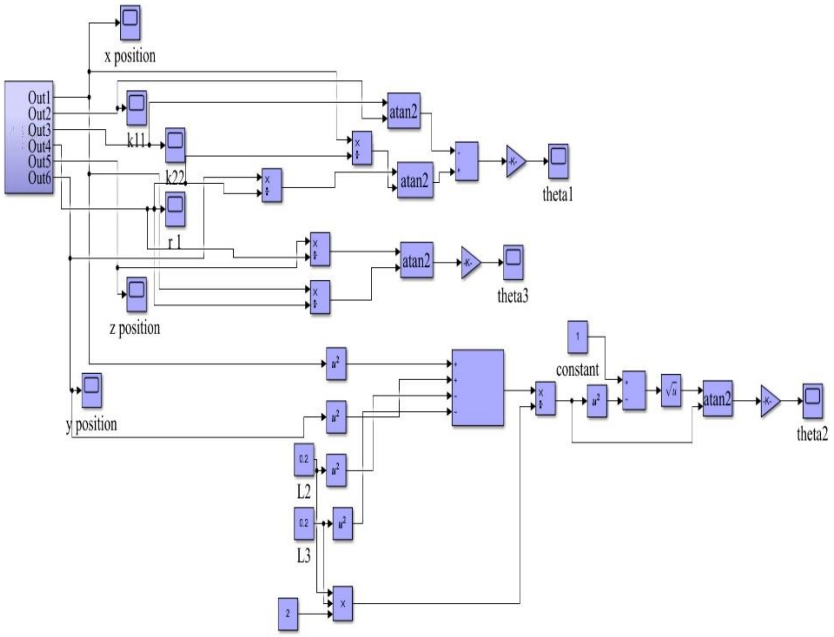

Fig. 4. The Simulation model of the inverse kinematics of the arm.

5. Fuzzy Neural Petri Net (FNPN)

Petri net is one of the tools used to represent graphical systems like block diagrams and visible communications and flow charts. It depends basically on the nature of

the synchronous and asynchronous systems in their work and to the fact that the different parts of the system can represent the relations between them as a net or graph [8]. FNPN It is an artificial intelligence algorithm that combined the structure of Petri Net in its simple representation and neural network learning procedure. Where the values of inputs place are fuzzified and the back-propagation algorithm is used to update the weights. Figure 5 shows the FNPN structure. Petri net consists of (input, output) places, transitions and the arc that connected from input places to transitions and from transitions to output places [9]. In FNPN, the input places refer to the input layer and the transitions refer to the hidden layer and finally, the output places refer to the output layer [10]. The equations of FNPN are as following:

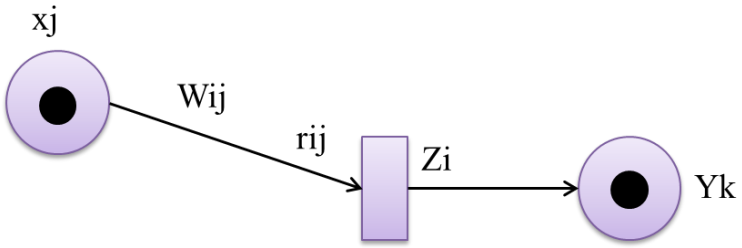

Fig. 5. FNPN structure

$$P_j = f(Input(j)) \tag{14}$$

f is a Gaussian membership functions (μ)

$$\mu_j = \exp\left(-(x_j - c_{ij})^2 / 2{s_{ij}}^2\right) \tag{15}$$

c_{ij}, s_{ij} are the mean of the Gaussian function and the standard deviation.

$$Z_i = \prod_{j=1}^{n}[W_{ij} S(r_{ij} \rightarrow P_j)] \quad j = 1,2,\dots\dots,n\,; i = 1,2,\dots\dots, hidden\ layer \tag{16}$$

$$r_{ij} \rightarrow P_j = \begin{cases} \frac{P_j}{r_{ij}}, if\ r_{ij} > P_j \\ 1, otherwise \end{cases} \tag{17}$$

$$Z_i = \prod_{j=1}^{n} W_{ij} \vee \begin{cases} \frac{P_j}{r_{ij}}, if\ r_{ij} > P_j \\ 1, otherwise \end{cases} \tag{18}$$

where W_{ij} the weight between the input places and transition, r_{ij} is the threshold level, Z_i is the activation level.

$$Y_k = f\left(\sum_{i=1}^{No\ of\ Transition} V_{ki} Z_i\right), \quad j = 1,2,\dots.,n \tag{19}$$

where Y_k is the marking level value of the output place, E called the performance index and is used to know the difference between two values one is the output places value, and the other is the target values needed from the network.

$$E = \frac{1}{2}\sum_{k=1}^{m}(t_k - Y_k)^2 \tag{20}$$

where t_k is the value of *k-th* target, Y_k is the value of *k-th* output .the standard sigmoid is the nonlinear function used to describe the output as following:

$$Y_k = \frac{1}{1+\exp(-\sum Z_i V_{ki})} \quad (21)$$

The parameters are update is derived as following :

$$param(iter+1) = param(iter) - \alpha \nabla_{param} E \quad (22)$$

where $\nabla_{param} E$ is a gradient of the performance index, α is the learning rate coefficient and *iter* is the iteration counter.

$$\Delta V_{ki} = -\zeta \, \partial E / \partial V_{ki} \quad (23)$$

$$\Delta V_{ki} = -\zeta [\exp(-Z_i V_{ki}) \, Z_i (t_k - Y_k)/(1 + \exp(-Z_i V_{ki}))] \quad (24)$$

$$V_{ki}(iter+1) = V_{ki}(iter) - \alpha \Delta V_{ki} \quad (25)$$

$$for \; k = 1,2,\ldots.,m; i = 1,2,\ldots.,no \; of \; hidden \; transitions$$

$$.\Delta r_{ij} = -\zeta \, \partial E / \partial r_{ij} \quad (26)$$

$$\Delta r_{ij} = -\zeta (t_k - Y_k)(\exp(-Z_i V_{ki}) \, V_{ki})/(1 + \exp(-Z_i V_{ki})) \prod_{\substack{l=1 \\ l \neq j}}^{n} [W_{il} S(r_{il} \rightarrow P_l)](1 - W_{ij}) \begin{cases} \frac{-P_j}{r_{ij}^2}, if \; r_{ij} > P_j \\ 0, otherwise \end{cases} \quad (27)$$

$$r_{ij}(iter+1) = r_{ij}(iter) - \alpha \Delta r_{ij} \quad (28)$$

$$\Delta W_{ij} = -\zeta \, \partial E / \partial W_{ij} \quad (29)$$

$$\Delta W_{ij} = -\zeta (t_k - Y_k)(\exp(-Z_i V_{ki}) \, V_{ki})/(1 + \exp(-Z_i V_{ki})) \prod_{\substack{l=1 \\ l \neq j}}^{n} [W_{il} S(r_{il} \rightarrow P_l)](1 - \begin{cases} \frac{P_j}{r_{ij}}, if \; r_{ij} > P_j \\ 1, otherwise \end{cases}) \quad (30)$$

$$W_{ij}(iter+1) = W_{ij}(iter) - \alpha \Delta W_{ij} \quad (31)$$

6. Result and Discussion

To control the robot arm and solve the movement problem represented by the forward and inverse kinematics problem a simulation program was used. The Matlab program used to simulate the dynamic system and the forward and inverse motions as follows:

Firstly, the data is generated using the Simulink library in Matlab, to represent the equations of the dynamic model and kinematic model. The simulated results save as data (inputs and outputs) to use for the FNPN algorithm and use for comparison with the algorithm results as following; from the dynamic model we get the joints angles as shown in Fig. 6. These joints angles used as an input to the forward kinematics part to calculate the desired position of the arm. While the opposite in the inverse kinematic model these positions represent the inputs to find the joints angles where in our case, we have three angles (*theta1*, *theta2* and *theta3*) in deg and three positions (*Px*, *Py* and *Pz*)in cm in *x y z* plane and after operating the simulation models the results are saved from the workspace in Matlab.

Secondly, using the data obtained from the simulation as inputs to obtain the desired outputs based on the given inputs; in the forward kinematic the input represented by the joints angles while the positions represent the outputs to be

accessed and the opposite is happening in the reverse kinematic where the inputs represent the positions while the outputs represent joint angles that give these locations. In each case, the hidden layers used was five and the rate of mean square error value used 0.00001 in the forward kinematic and 0.000001 in the case of reverse kinematic. In the end, the results obtained from the program were compared with the simulation results. Figure 7 shows the comparison between the desired positions from the simulation of the forward kinematic equations and predicted positions from FNPN algorithm. While Fig. 8 shows the comparison between the desired angles from the simulation of the inverse kinematic equations and predicted angles from FNPN algorithm.

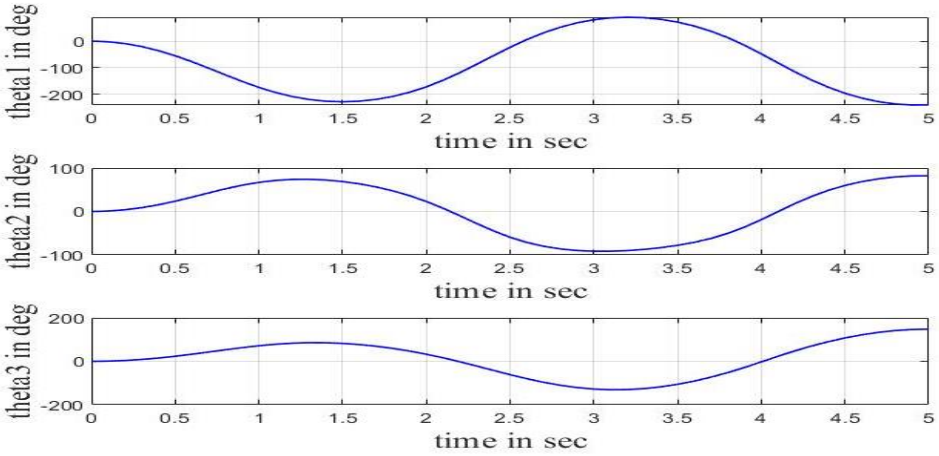

Fig. 6. Theta1, theta2, and theta3 of the dynamic model against time.

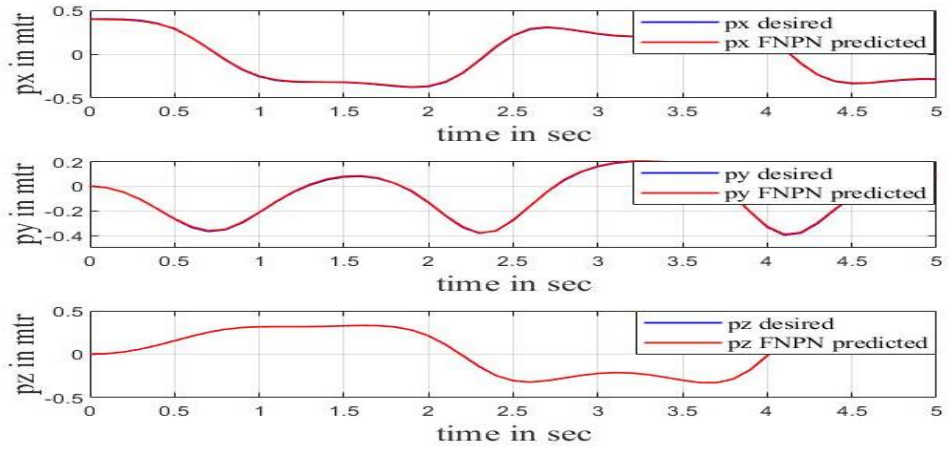

Fig. 7. A comparison of Desired and FNPN Predicted values (*x*, *y*, *z*) against time.

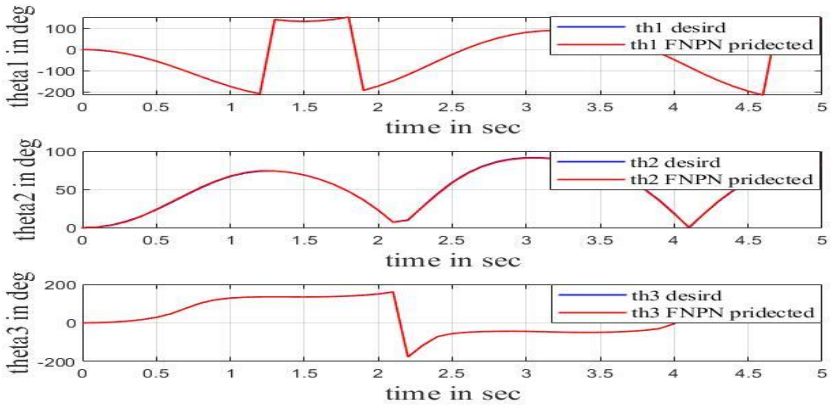

Fig. 8. A comparison of Desired and FNPN Predicted valuesTh1, Th2,Th3 against time.

7. Conclusions

This research discussed the kinematics (forward &inverse) problem of a 3DOF robotic arm and solved it's using the FNPN algorithm. The dynamic model and the forward and inverse model of this arm are first derived and simulated using the simulation environment in the Matlab Program. The results of this simulation used as data (input and output) to the algorithm. The FNPN algorithm can solve the problem with more accuracy without the need to use a complex model, know any information about the system, where only the input and the output data are used to describe and define the model structure, and can always update the network and get better results when new data are gotten and changing the Mean Square Error (MSE). Based on the MSE used 0.00001 in the forward kinematics case and 0.000001 in the inverse kinematics case, FNPN gave acceptable results compare to the simulation results. Where it was observed through the results of the ability of FNPN to track the system and control the movement of the arm and direct it accurately to reach the target in both cases of forward and reverse movement with high accuracy rates. The reason for this is because the FNPN system is a hybrid combination of FN and PN, and thus the benefits of both systems are combined to serve the system data to be controlled, and here is the robotic arm.

References

1. Sadegh, H.; and Zarabadipour, H. (2014). Modeling, simulation and position control of 3DOF articulated manipulator. *Indonesian Journal of Electrical Engineering and Informatics*, 2(3), 132-140.
2. Duka, A.V. (2015). ANFIS based Solution to the inverse kinematics of a 3DOF planar manipulator. *Procedia Technology*, 19, 526-533.
3. Gupta, A.; Bhargava, P.; Deshmukh, A.; Agrawal, S. and Chourikar, S. (2018). A geometric approach to inverse kinematics of a 3 DOF robotic arm. *International Journal for Research in Applied Scince and Engineering Technology*, 6(1), 3524-3530.

4. Almusawi, A.R.J.; Dülger, L.C.; and Kapucu, S. (2016). A new artificial neural network approach in solving inverse kinematics of robotic arm (Denso VP6242). *Computational Intelligence and Neuroscience*, 2016, 1-10.
5. Ramish.; Hussain, S.B.; and Kanwal, F. (2016). Design of a 3 DoF robotic arm. *Proceedings of Sixth International Conference on Innovative Computing Technology*. Dublin, Ireland, 145-149.
6. Sivasamy, D.; Anand, M.D.; and Sheela, K.A. (2019). Robot forward and inverse kinematics research using MATLAB. *International Journal of Recent Technology and Engineering*, 8(2S3), 29-35.
7. Kim, J.; Lee, A.S.; Chang, K.; Schwarz, B.; Gadsden, S.A.; and Al-Shabi, M. (2017). Dynamic modeling and motion control of a three-link robotic manipulator. *Proceedings of the First International Conference on Artificial Life and Robotics*. Miyazaki, Japan.
8. Maharjan, B. (2015). *Modeling humanoid swarm robots with petri nets.* Master's thesis, University of Stavanger, Norway.
9. Saleh, A.L.; Mohammed, M.J.; Kadhim, A.S.; Raadthy, H.A.M.; and Mohammed, H.J. (2018). Design fuzzy neural petri net controller for trajectory tracking control of mobile robot. International *Journal of Engineering and Technology*, 7(4), 2256-2262.
10. Ali, A.A.; and Mosa, G.S. (2017). Neural fuzzy petri net based arabic phoneme classifier with mfcc feature extraction. *Qalaai Zanist Scientific Journal*, 2(2), 375-389.