SINERGI Vol. 25, No. 3, October 2021: 237-244
http://publikasi.mercubuana.ac.id/index.php/sinergi
http://doi.org/10.22441/sinergi.2021.3.001

SELF-LEARNING OF DELTA ROBOT USING INVERSE KINEMATICS AND ARTIFICIAL NEURAL NETWORKS

Zendi Iklima[1], Muhammad Imam Muthahhar[1], Asif Khan[2], Arifiansyah Zody[3]
[1]Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana, Indonesia
[2]School of Computer Science and Technology, Beijing Institute of Technology, China
[3]Department of Information Technology, Faculty of Computer Science, Esa Unggul University, Indonesia

Abstract
As known as Parallel-Link Robot, Delta Robot is a kind of Manipulator Robot that consists of three arms mounted in parallel. Delta Robot has a central joint constructed as an end-effector represented as a gripper. An Analysis of Inverse Kinematic (IK) used to convert the end-effector trajectory (X, Y) into rotations of stepper motors (Z_A, Z_B and Z_C). The proposed method used Artificial Neural Networks (ANNs) to simplify the process of IK solver. The IK solver generated the datasets contain motion data of the Delta robot. There are 11 KB Datasets consist of 200 motion data used to be trained. The proposed method was trained in 58.78 seconds in 5000 iterations. Using a learning rate (α) 0.05 and produced the average accuracy was 97.48%, and the average loss was 0.43%. The proposed method was also tested to transfer motion data over Socket.IO with 115.58B in 6.68ms.

Keywords:
Artificial Neural Network;
Delta Robot;
Inverse Kinematics;
Motion Data;
Socket.IO;

Article History:
Received: May 20, 2020
Revised: August 22, 2020
Accepted: September 18, 2020
Published: July 2, 2021

Corresponding Author:
Zendi Iklima
Electrical Engineering Department, Universitas Mercu Buana, Indonesia
Email:
zendi.iklima@mercubuana.ac.id

INTRODUCTION

Robots can be classified into two topologies, namely Serial Manipulator and Parallel Manipulator [1]. The parallel manipulator has abilities such as acceleration, stiffness, low inertia, high precision, and faster per-cycle operations. Serial manipulator produced slower movement because of its connected link a chain, and each joint has a mass in which needs to be calculated to achieve the dynamic manipulator. Meanwhile, The Delta Robot joints are mechanically separated to achieves faster movement by using low inertia. Nevertheless, there are still shortcomings in the Delta robot that is a relatively small workspace robot.

Delta Robot consists of three arms that are connected to the base and effector using a universal joint. The key of Delta robot design is using a parallelogram mechanism to maintain the orientation of the end-effector to the base. Since the Delta robot's Original Patent expired in December 2006 (Europe) and December 2007 (America), the development of research and implementation of the Delta robot has become more widespread [2][3]. The implementation of the Delta robot is used as a type of 3D Printer, Sketcher Robot, Laser Cutting Robot, and Pick and Place robot. So far, the Delta robot has been developed with a focus on a kinematic analysis of the robot's movement.

Kinematics is a mathematical model that describes the movement of end-effectors through Cartesian coordinates of the active joint position variable or vice versa. However, mathematical calculations of kinematics do not fully reflect perfect results in actual application. There is a position error caused by component manufacturing tolerance, assembly programming, and the dynamic influence of robot mechanics. On the other hand, the translation of forwarding and inverse kinematics on Delta robots is a fairly difficult problem.

The Inverse Kinematics (IK) of a Delta Robot has been derived in several methods. Some methods are numerical algorithm, geometrical method, genetic algorithm (GA),

reduced method, polynomial method, fuzzy algorithm, fuzzy PID [4], Feed Forward Neural Network, Fuzzy Neural Network, Particle Swarm Optimization (PSO), etc. [5].

Previous work has been done to solve the IK problem using Neural Network (NN). NN can be used to control validations of the IK model. IK model is created based on the robot's geometry and dynamic to solve the forward kinematics problem [4][5].

Other solutions to solve the IK problem of Delta Robot can be by using a hybrid approach are artificial neural network and particle swarm optimization (ANN-PSO) model in the behavior prediction and optimization of channel connectors embedded in normal and high-strength concrete (HSC) [6, 7, 8].

An interesting topic also explained that IK solution could be solved using neural networks combined with a genetic algorithm (GA) [9, 10, 11]. NN-GA is an optimal design method for the parallel robot, which maximizes the volume of the workspace of parallel robots. The neural network learns the motion model of the robot. Then, the genetic algorithm uses this model to generate the optimal parameters of the robot [12].

To overcome this, the proposed method to solve the IK problem is using an Artificial Neural Network (ANN). Our proposed method's loss can be minimised by finding an optimum ANN architecture by constructing the layers, the activation function, and the optimizer. On the other hand, our proposed method is implemented using Socket.IO channels to distribute the pre-model of our IK-ANN data.

METHOD

This paper created the Delta robot using the Inverse Kinematics method as an initializing value for the Artificial Neural Network training data. The results of the end-effector position forwarded to the motor motion execution for a displacement of the effector position. Figure 1 shows the process of the proposed method while it trains and tests the ANN Model.

The IK dataset was generated in small rows (200 rows of IK data). The inputs contain any positions of the end-effector on the base frame $R'(O' - x'y'z')$. The outputs contain any directions on $Z - axis$ $(Z_A, Z_B, and\ Z_C)$. ANN Model (pre-trained) used to train (the dataset) and to test (random inputs) to predict the direction on $Z - axis$ $(Z_A, Z_B, and\ Z_C)$. Socket.IO implemented as a communication platform in which will broadcast any random input (x, y) from delta robot then tested on ANN trained model to generate a predictive value of $(Z_A, Z_B, and\ Z_C)$.

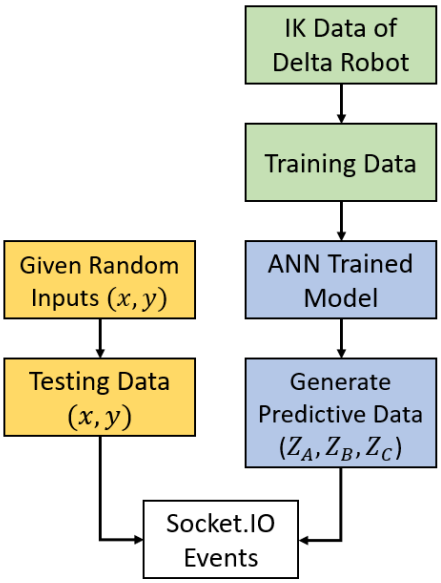

Figure 1. IK-ANN Block Diagram

Inverse Kinematics (IK) Analysis

Inverse Kinematics on this robot is very necessary for the design of robot motion. This function is to find the position of the entire linear actuator or slider towards the end effector. For example, if the end-effector position moves to position X with respect to position 0, then how much displacement changes (up/down) of the three linear actuators or sliders. In this section, the analysis is based on Stolfi [12].

Delta robot consists of the top platform, bottom platform, and end effector. The reference remains on the $R(O - xyz)$ frame system located in the middle on the upper platform (ABC triangle). The Z-axis is parallel to the end effector, and the Y-axis is parallel to CO, as shown in Figure 2.

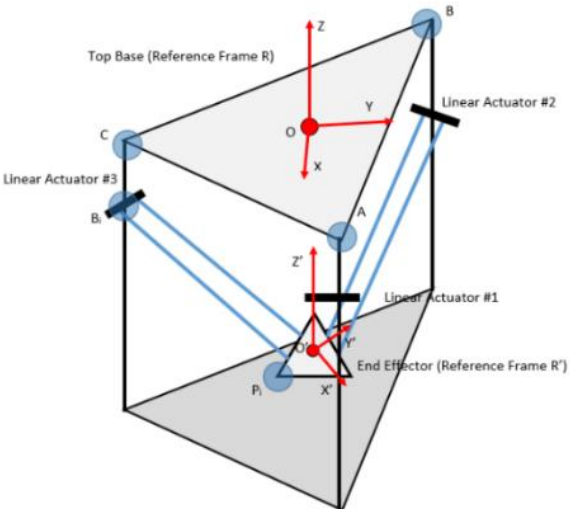

Figure 2. Delta Robot Kinematics Frame

Another reference is the system $R'(O' - x'y'z')$ located in the middle of the end effector (triangle P_1, P_2, P_3). The Z-axis 'is in the direction perpendicular to the end effector, and the Y-axis' is parallel along P_3O'.

As a geometry parameter, where [4]:

$$OA = OB = OC = R \quad (1)$$
$$O'P_1 = O'P_2 = O'P_3 = r \quad (2)$$
$$B_1P_1 = B_2P_2 = B_3P_3 = L \quad (3)$$

The object of inverse kinematics is to determine the position of three linear actuators due to changes in point O' (middle of the end effector) to the global reference of the R system. Point O' in frame R can be written as:

$$[O']R = (xyz)T \quad (4)$$

The coordinates of the point P_i in frame R' can be written as:

$$[P_i]R' = (rcos(n_i), rsin(n_i), 0)T \quad (5)$$

Where $i = 1, 2, 3$ and:

$$n(i) = \frac{4i-3}{6}\pi \quad (6)$$

Adding from two vectors at once will get the position of P_i with respect to the $O - xyz$ frame [3].

$$[P_i]R = (rcos(n_i) + x, rsin(n_i) + y, z)T \quad (7)$$

Meanwhile, to determine the Bi point on frame R:

$$[B_i]R = (Rcos(n_i), Rsin(n_i), x_i)T \quad (8)$$

Where Z_i Above is the Z-axis position to be searched as the position of each linear actuator concerning the frame R reference.
As it is known:

$$L = |[P_i - B_i]| \quad (9)$$

Then it can be specified as:

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = L^2 \quad (10)$$

Where:

$$x_i = (R - r)cos(n_i) \quad (11)$$
$$y_i = (R - r)sin(n_i) \quad (12)$$

By rearranging Equation 10:

$$z = \pm\sqrt{L^2 - (x - x_i)^2 - (y - y_i)^2} \quad (13)$$

To find the desired position z point on each actuator from the Delta robot, only the positive form of (13) is used.

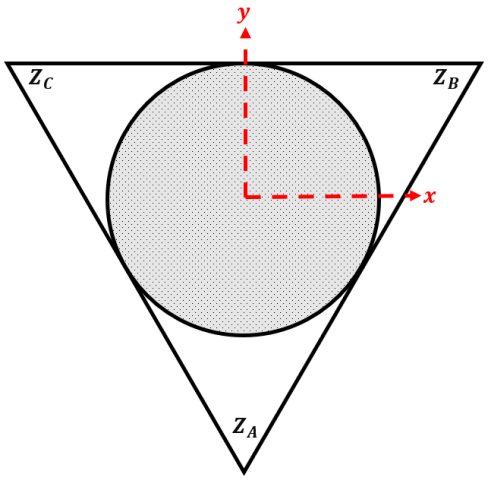

Figure 3. Sketch of dots for Dataset in Workspace

The coordinate frame of the Delta Robot shown in Figure 2 can be depicted as the desired coordinates of the end-effector. Figure 3 shows a sketch of dots for Dataset in Workspace. This method is easily used to see the data distribution of the delta robot workspace.

Artificial Neural Network (ANN)

An intelligence platform inspired by the biological neurons can conveniently learn patterns and predict high-dimensional data distributions [6][11]. ANN design is needed as an initial description of how ANN works as a substitute for inverse kinematics' repetitive mathematical calculation process to determine the Z-axis position of the slider against the $X - Y$ coordinate end-effector. ANN consists of the input layer, hidden layer, and output layer. Figure 4 shows the components among the layers [13, 14, 15].

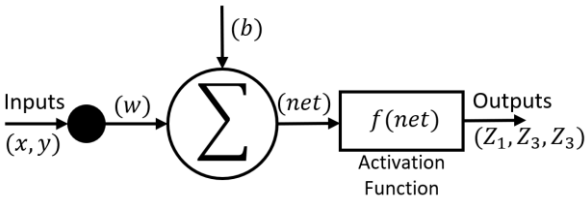

Figure 4. The architecture of ANN Delta Robot [6]

The Input layer contains activations of end-effector coordinates defined as x and y in which formulated as

$$F = f(net) = f(x, y) \quad (14)$$

$f(x, y)$ defines as the activation function in the desired direction should be [6, 16, 17],

$$f(x, y)_j = \frac{1}{1 + e^{\sum_{j=1}^{n} w_{jk} x_j + b_j}} \quad (15)$$

The output layer represents the desired position of z positions on each actuator described in (13) $(Z_A, Z_B, and\ Z_C)$. To calculate the backpropagation process defined as f' (or called δ_j Aka "delta"). The derivative of the activation function $f(x, y)$ and γ as a learning rate. So, the weights will be graded when the gradient descent is calculated by [18, 19, 20],

$$\delta_j = f'(x, y) \sum_{j=1}^{n} w_{jk}\, \delta_j \quad (16)$$

Socket.IO

Socket.IO is used to distribute the data among the environment by using events called '*onListening'* and events '*onEmit.'* Events '*onListening'* is an event to retrieve incoming data in which distributed through its event. Events '*onEmit*' is an event to transmit any data into its event [21][22].

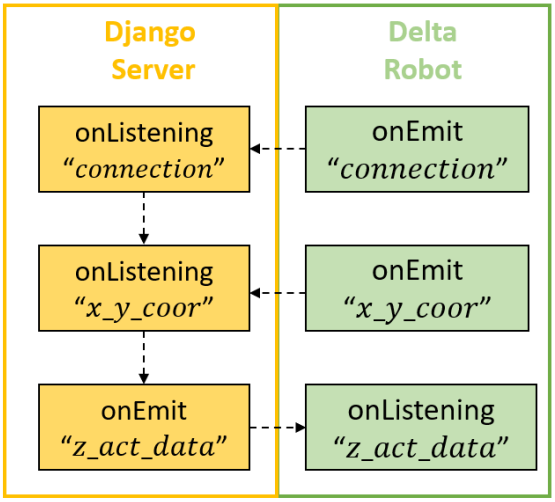

Figure 5. Socket.IO Events [20]

Figure 5 represents the Socket.IO events used to distribute data among platforms. The data was trained on the Django server and distribution through socket.IO event onListening 'x_y_coor' in which waiting to receive end-effector data (x, y) from the robot delta. then emitting an event 'z_act_data', which mean the predictive z actuator data $(Z_A, Z_B, and\ Z_C)$

Proposed Design

Table 1 shows the hyperparameter that might be used to find the optimum ANN architecture to the Sole IK problem as described above.

Table 1. Hyperparameter of ANN

Parameters	Choices	Selected Parameters
Number of Layers	[3,4,5,6,7,8]	3
Number of Neurons in Hidden Layers	$[2, 4, 6, ...]$	$2, 6, 12$
Activation Functions	['ReLU', 'Sigmoid', 'TanH', 'Leaky ReLu']	'Leaky ReLu'
Optimizer	['SGD', 'ADAM']	'ADAM'
Activation Functions Rate γ	[0.01. 0.02, …. 1.0]	0.05

The hyperparameters were selected as the parameter to build ANN architecture in which to find the optimum values of the number of layers, number of hidden layers, activation functions, optimizer, and the learning rate.

RESULT AND DISCUSSION

The following is the prototype that was completed as describes in Figure 2. As shown in Figure 3, the inverse kinematics inputs taken by the coordinate of the end-effector $(x$ and $y)$ to produce the desired Z position in direction A, B and C $(Z_A, Z_B, and\ Z_C)$

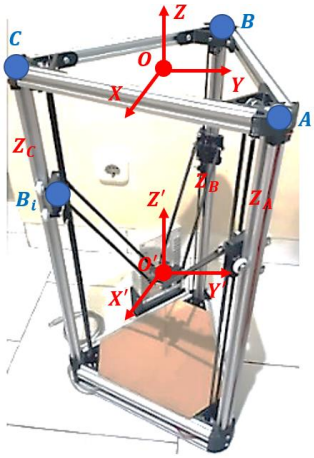

Figure 6. Prototype of Delta Robot

The dataset from robot motion in Figure 6. 200 rows of motion data were collected and stored as a CSV file.

ANN Training

The training process was performed using Intel(R) Core© i5-6300HQ CPU@2.30GHz, 16GB RAM, and NVIDIA GeForce GTX 960M 4GB VRAM. The training process followed the hyperparameter shown in Table 1. The training performance was captured in Figure 7.

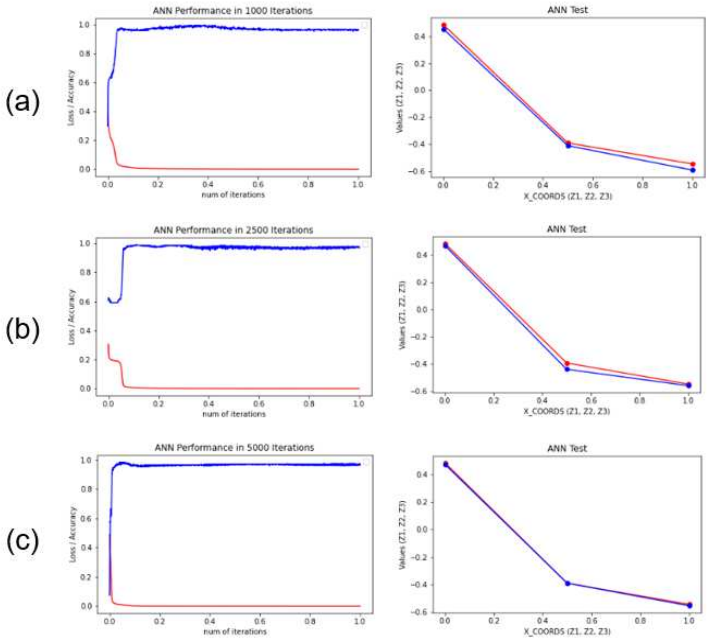

Figure 7. The ANN Loss and Accuracy in: (a) 1000 iterations, (b) 2500 iterations, and (c) 5000 iterations

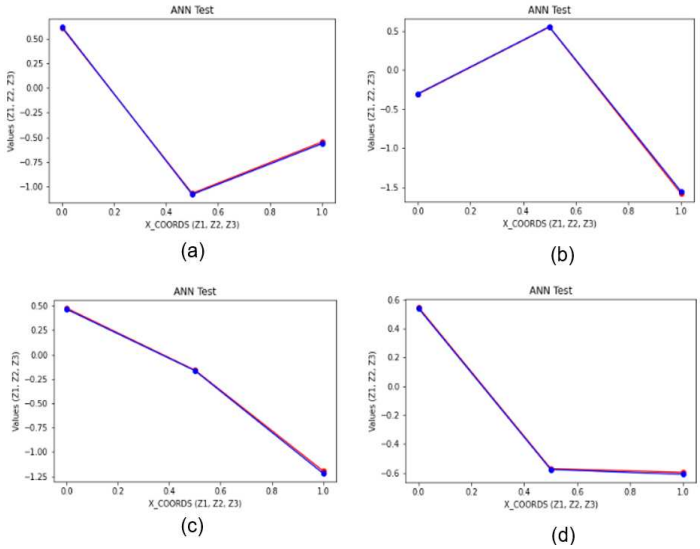

Figure 8. The Testing Performance of Single Motion: (a) Motion Set 1, (b) Motion Set 2, (c) Motion Set 3, and Motion Set 4

Figure 7 shows the performance of the ANN Loss and Accuracy. The loss is presented as the red line of the left graph, and the accuracy is presented as the blue line of the left graph. Figure 7(a) shows the loss decreased and the accuracy increased in 1000 iterations. Figure 7(b) shows the loss decreased and the accuracy increased in 2500 iterations. Figure 7(c) shows the loss decreased and the accuracy increased in 5000 iterations. The detailed values of ANN Loss and accuracy were captured in this Table 2.

Table 2 shows the selected hyperparameter was performed in 1000 iterations, 2500 iterations, and 5000 iterations. The loss was decreased by 0.0043 in 5000 iterations, and the accuracy was increased by 0.9748 in 5000 iterations. This iteration step was executed in 58.7826 seconds.

ANN Testing

Single point of end-effector coordinated tested as desired coordinated of x and y shown in Figure 7.

The proposed method was tested in single desired coordinated as a set of motion. Figure 8 represents four motions that were tested in Figure 8(a), Figure 8(b), Figure 8(c), and Figure 8(d). By

given any input x and y, will produce any desired actuators positions in Z-directions ($Z_A, Z_B, and\ Z_C$). The detailed values of ANN testing performance were captured in this Table 3.

The proposed method was implemented to perform n Motions and captured in actuator direction $Z_A, Z_B, and\ Z_C$. Figure 9 shows the algorithm performance to generate 50 motions in the direction Z_A (Figure 9(a)), Z_B(Figure 9(b)), Z_C(Figure 9(c)), and combined (Figure 9(d)).

Table 2. The ANN Loss and Accuracy in (a) 1000 iterations, (b) 2500 iterations, and (c) 5000 iterations

	1000 iterations	2500 iterations	5000 iterations
Executed time (seconds)	12.5525	29.7373	58.7826
Num of Epoch	1000	2500	5000
Average Loss	0.0787	0.0131	0.0043
Average Accuracy	0.9322	0.9528	0.9748
Activation Function	Leaky-ReLu	Leaky-ReLu	Leaky-ReLu
Optimizer	Adam	Adam	Adam
Learning Rate (α)	0.05	0.05	0.05
Layers	2 6 12 6 3	2 6 12 6 3	2 6 12 6 3

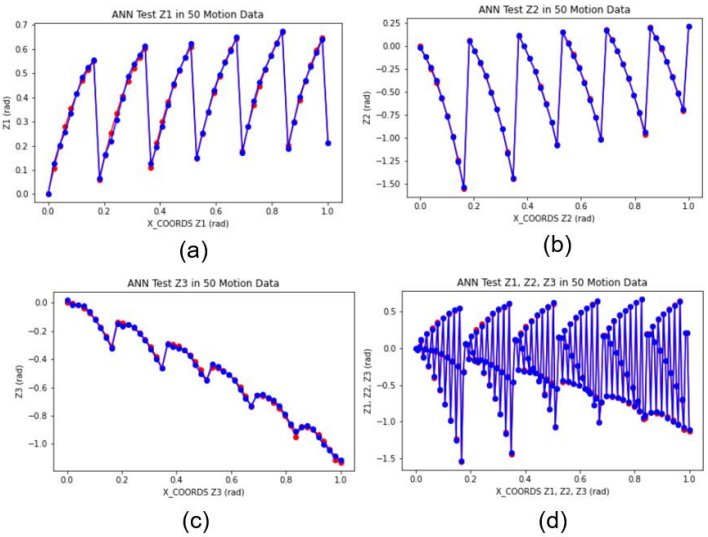

Figure 9. The Testing Performance of 50 Motions: (a) Motion Data of Z_A, (b) Motion Data of Z_B, (c) Motion Data of Z_CAnd (d) Motion Data of Z_A, Z_B, and Z_C

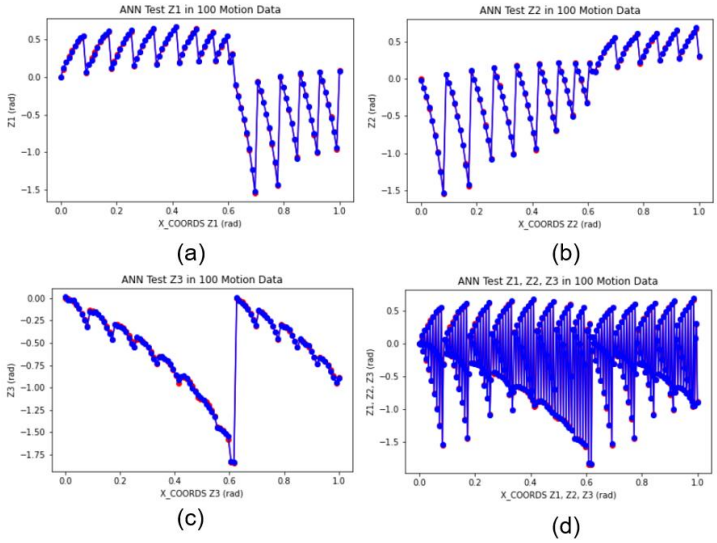

Figure 10. The Testing Performance of 100 Motions: (a) Motion Data of Z_A, (b) Motion Data of Z_B, (c) Motion Data of Z_CAnd (d) Motion Data of Z_A, Z_B, and Z_C

Table 3. The Testing Performance of Single Motion
(a) Motion Set 1, (b) Motion Set 2, (c) Motion Set 3, and Motion Set 4

Motion Set		1	2	3	4
Inputs (meter)	X	0.7	0.5	0.3	−0.4
	Y	0.2	0.3	0.6	0.7
Target (Radian)	Z_A	0.6086	0.5482	0.4784	−0.3098
	Z_B	−1.0696	−0.5705	−0.1597	0.5508
	Z_C	−0.5474	−0.5971	−1.1973	−1.5789
Predicted (Radian)	Z_A	0.6256	0.5405	0.4675	−0.3031
	Z_B	−1.0751	−0.5763	−0.1646	0.5553
	Z_C	−0.5255	−0.6099	−1.2226	−1.5499

Figure 10 shows the algorithm performance to generate 100 motions in the direction Z_A (Figure 10(a)), Z_B(Figure 10(b)), Z_C(Figure 10(c)), and combined (Figure 10(d)).

The desired 50 motions` Z values are shown in the blue line, which is compared with the actual values represented in the red line.

Table 4 shows the Socket.IO response within a set of motion with 115.6B processed in 6.68 ms, 50 sets of motion with 912.3B processed in 216.35 ms, and 100 sets of motion with 1712.1B processed in 335.56 ms.

Table 4. Socket.IO Response

Inputs (motions)	1	50	100
Data Size ($Byte$)	115.6	912.3	1712.1
Socket.IO Response (ms)	6.68	216.35	335.56

CONCLUSION

This study has developed the method to determine the end-effector position using inverse kinematics and substitute by Artificial Neural Network. All the equations are used to determine the position of the slider that will be dataset for ANN. The proposed method used Artificial Neural Networks (ANNs) to simplify the process of IK solver. The IK solver generated the datasets contain motion data of the Delta robot. 11KB Datasets consist of 200 motion data used to be trained. The proposed method was trained in 58.78 seconds in 5000 iterations. Using a learning rate (α) 0.05 produced an average accuracy of 97.48%, and the average loss was 0.43%. The proposed method was also tested to transfer motion data over Socket.IO with 115.58B in 6.68ms.

ACKNOWLEDGMENT

The author expresses gratitude to the Electrical Engineering Department and Research Center of Mercu Buana University, Jakarta, Indonesia. Under the guidance, the authors have been completed this research paper.

REFERENCES

[1] Z. Pandilov, V. Dukovski, “Comparison of The Characteristics Between Serial and Parallel Robots,” *Acta Tehnica Corviniensis: Bulletin of Engineering*, vol. 1, no. 7, pp. 143-160, 2014

[2] J. Brinker, N. Funk, P. Ingenlath, Y. Takeda, and B. Corves, “Comparative Study of Serial-Parallel Delta Robot with Full Orientation Capabilities,” *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 920-926, April 2017, doi: 10.1109/LRA.2017.2654551

[3] Z. Iklima, A. Adriansyah, S. Hitimana, “Self-Collision Avoidance of Arm Robot using Generative Adversarial Network and Partical Swarm Optimization (GAN-PSO),” *SINERGI*, vol. 25, no. 2, pp. 141-152, 2021, doi: 1022441/sinergi.2021.2.005

[4] H. Yuancan and H. Zongllin, “Neural Network Based Dynamic Trajectory Tracking of Delta Parallel Robot,” *IEEE International Conference on Mechatronics and Automation (ICMA)*, Beijing, China, Aug. 2015, pp. 1938-1943, doi: 10.1109/ICMA.2015. 7237782

[5] M. Heidari, S.M.R Faritus, S. Shateyi, “Using Artificial Neural Network for Feed Kinematic problem of Under-Constrained Cable Robots,” *Journal of Vibroengineering (JVE)*, vol. 20, no. 1, pp. 385-400, Feb. 2018, ISN: 1392-8716, doi: 10.21595/jve.2017. 18633

[6] M. Shariati, et al., “Application of a Hybrid Artificial Neural Network-Particle Swarm Optimization (ANN-PSO) Model in Behavior Prediction of Channel Shear Connectors Embedded in Normal and High-Strength

Concrete," *Applied Sciences*, vol. 9, no. 24, pp. 5534, 2019, doi:10.3390/app9245534
[7] P. Srisuk, A. Sento, Y. Kitjaidure, "Inverse Kinematic Solution using Neural Network from Forward Kinematics Equations," *The 9th International Conference on Knowledge and Smart Technology (KST)*, Thailand, 2017, pp. 61-65, doi: 10.1109/KST.2017.7886084
[8] EG López, W. Yu, X. Li, "Optimal Design of a Parallel Robot Using Neural Network and Genetic Algorithm," *The 10th International Conference on Intelligent Control and Information Processing (ICICIP)*, Morocco, February 2017, pp. 64-69, doi: 10.1109/ ICICIP47338.2019.9012182
[9] K. K. Pavan, M. J. Murali, D. Srikanth, "Generalized solution for inverse kinematics problem of a robot using hybrid genetic algorithms," *International Journal of Engineering & Technology (IJET)*, vol.7 no. 4.6, pp. 250-256, 2018, p.250-256, doi: 10.14419/ijet.v7i4.6.20486
[10] E. McCormick, W. Yanjun; L. Haoxiang, "Optimization of a 3-RRR Delta Robot for a Desired Workspace with Real-Time Simulation in MATLAB," *The 14th International Conference on Computer Science & Education (ICCSE 2019)*, Canada, August 2019, pp. 935-941, doi: 10.1109/ICCSE. 2019.8845388
[11] A.G.C Premachandra, et al., "Genetic Algorithm Pick and Place Sequence Optimization for a Color and Size Sorting Delta Robot," *The 6th International Conference on Control, Automation and Robotics (ICCAR)*, Singapore, April 2020, pp. 209-213, doi: 10.1109/ICCAR49639.2020. 9108045
[12] F. Tolfi, A. Avanzini, B. Welker, and A. Isinhue, "Design Review of Delta Robot," *MECE E3430 Engineering Design*, pp. 10-19, 2014
[13] A. A. Jaber and R. Bicker, "Industrial Robot Backlash Fault Diagnosis Based on Discrete Wavelet Transform and Artificial Neural Network," *American Journal of Mechanical Engineering*, vol. 4, no. 1, pp. 21-31, January 2016, doi:10.12691/ajme-4-1-4
[14] R.Y. Putra, et al., "Neural Network Implementation for Invers Kinematic Model of Arm Drawing Robot," *International Symposium on Electronics and Smart Devices (ISESD)*, Bandung, Indonesia, March 2017, pp. 153-157, doi: 10.1109/ ISESD.2016.7886710
[15] T. Collins, W. Shen, "PASO: An Integrated, Scalable PSO-based Optimization Framework for Hyper-Redundant Manipulator Path Planning and Inverse Kinematic," *ISI Tech Report*, January 2016
[16] H. Su, C. Yang, H. Mdeihly, A. Rizzo, G. Ferrigno and E. De Momi, "Neural Network Enhanced Robot Tool Identification and Calibration for Bilateral Teleoperation," in *IEEE Access*, vol. 7, pp. 122041-122051, 2019, doi: 10.1109/ACCESS.2019.2936334
[17] B. A. Garro and R. A. Vazquez. "Designing Artificial Neural Networks using Particle Swarm Optimization Algorithms," *Computational Intelligence and Neuroscience*, ID: 369298, June 2015, doi: 10.1155/2015/369298
[18] L. F. Carlos et al., "A soft computing approach for inverse kinematics of robot manipulators," *Engineering Applications of Artificial Intelligence*, vol. 74, pp. 104–120, September 2018, doi: 10.1016/j.engappai. 2018.06.001
[19] S. Mahdi, et al., "Application of a Hybrid Artificial Neural Network- Particle Swarm Optimization (ANN-PSO) Model in Behavior Prediction of Channel Shear Connectors Embedded in Normal and High-Strength Concrete," *Applied Sciences*, vol. 9, no. 24, December 2019, doi: 10.3390/app9245534
[20] L. Peiyuan, "Deep Neural Network Based Subspace Learning of Robotic Manipulator Workspace Mapping," *International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO)*, 2018, vol.1, pp. 109-120, doi: 10.1109/ICCAIRO. 2018.00027
[21] T. Dewi, C. Anggraini, P. Risma, Y, Oktarina, and M. Muslikhin, "Motion Control Analysis of Two Collaborative Arms Robots in Fruit Packaging System," *SINERGI*, vol. 25, no. 2, pp. 217-226, 2021, doi: 1022441/sinergi. 2021.2.013
[22] Z. Iklima, "Design of Distributed Control Systems in Lighting Installation in 3-Floor Buildings based on IoTaaS (Internet of Things as a Service) using Docker Container", *Jurnal Teknologi Elektro*, vol.10, no.1, pp. 26-33, 2019, doi: 10.22441/ jte.v10i1.004