Machine Learning-Based Estimation of End Effector Position in Three-Dimension Robotic Workspace

Hamna Baig[1], Ejaz Ahmed[2], Ihtesham Jadoon[1],
[1]Department of Electrical and Computer Engineering COMSATS University Islamabad Attock Campus Attock, Pakistan.
[2]Department of Electrical Engineering Aero Space and Aviation, Air University Islamabad Kamra Campus Attock Pakistan)
***Correspondence**: Ejaz Ahmed: 225458@aack.au.edu.pk

Citation | Baig. H, Ahmed. E, Jadoon. I, "Machine Learning-Based Estimation of End Effector Position in Three-Dimension Robotic Workspace", IJIST, Special Issue pp 83-92, May 2024
Received | May 04, 2024, **Revised |** May 12, 2024, **Accepted |** May 17, 2024, **Published |** May 21, 2024.

Introduction/Importance of Study: The Workspace is the area around the robot where a robot can freely move with possible input variations of different joint angles.
Novelty statement: Conventionally iterative simulation methods are used to find robotic workspace. Which are computationally slow and difficult to model. Our approach utilizes machine-learning algorithms to predict the workspace and position of an end effector.
Material and Method: Multiple Linear Regression (MLR), Decision-Tree Regression, and Artificial Neural Network (ANN) algorithms trained for prediction. The dataset, which is collected and used as train and test data, is further for the validation step.
Result and Discussion: By simulating the robot with the Denavit-Hartenberg (D-H) approach in MATLAB. The results findings show the accuracy of Machine learning algorithms specifically Artificial Neural Networks (ANN) perform better than conventional mathematical methods
Concluding Remarks: Artificial Neural Network (ANN) outperformed other machine learning methods.

Keywords: Robotics; Artificial Neural Network (ANN); Denavit-Hartenberg (D-H); Prediction and Machine Learning.

Acknowledgement:
Thank you to UET Peshawar for hosting the conference and considering this paper for conference ICTIS 2024.
Author's Contribution:
Ejaz Ahmed presented an idea and modeled the system also contributed to drafting the paper Hamna Baig made code for integrating robotics with Machine learning Ihtesham Jadoon works on paper drafting and results comparison.
Conflict of Interest:
No conflict of interest for publishing this manuscript in IJIST.
Project Details:
This work is done without any financial assistance from any source.

Introduction:

In many crucial cases, it is very difficult, if not impossible, to analytically predict the behavior of physical systems. The necessity to build a physical system prototype drives modeling, which in turn reveals strong motivations to investigate and analyze a system's operation. Modeling of robots is usually carried out through kinematics study. That deals with a model of the robot without any influence of force. The kinematics of a robot deals with the geometric and time-based properties under motion and in particular how various links of a robot move with respect to one another with variation of time. Which is the analytical way of explaining the relation between different joint variables. Kinematics modeling is divided into two types forward and inverse kinematics, The first one gives the position and orientation when joint angles and joint positions are known. While inverse kinematics deals with a set of complex equations computed by vector and analytical algebra to compute the joint angles once, the end effector position is given [1].

The robot is the heart of the automation industry used in manufacturing and assembly applications and many applications that require robots to move objects from one place to another using mechanically designed grippers these robots need to be precise in terms of placement of objects. To achieve the precise pick and place robot application it needs to be modeled with the least error. Intense research is carried out in the analytical modeling of robot systems, which are based on line and point transformation. Campos-Macías [2] proposed a geometric model to calculate and find the relation between input angles and output position unknown joint angles required for autonomous positioning of a robotic system. In [3] author Bayro-Corrochano uses a new algebraic method called quaternion for modeling different physical systems system. Popovic et al. [4] computed a method to model the upper extremity movement of the arm of a multi-leg moving robot inspired by an animal's movement. An analytical model-based approach to compute the kinematics of a humanoid robot was discussed in [5]. In [6] author presented an inverse kinematics model to calculate all the joint variables of a serial arm manipulator. Applications of machine learning in different robots are discussed in [7].

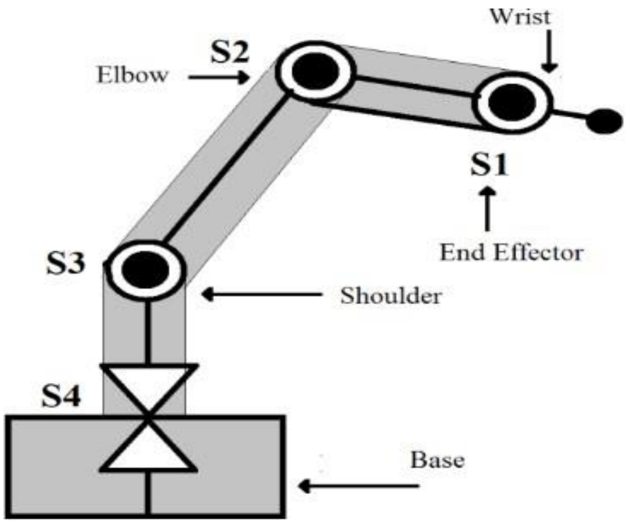

Figure 1: Kinematic Model Representation of 4-Degree-of-Freedom Robot.

Figure 1 shows the kinematic model's simplified view of the robotic arm in an inverted 'L' pose. The first joint S1 is used to move the arm claw to pick objects, and the joints S2 and S3 are the elbow and shoulder joint respectively to move the arm to the desired position. The S3 joint is the base joint to rotate the robot arm. The forward kinematic model of the 4 DOF Robot is presented in Section II Section III presents the discussion on forward kinematics using machine learning algorithms on MATLAB Section IV gives the result and Discussion of machine learning algorithms and Section V gives the final Conclusions.

The second last paragraph of the introduction section should explain the hierarchy/flow of research. The last paragraph should explain the objectives of the research and Novelty statement. This paper aims to focus on using machine-learning algorithms to estimate the end effector position in a three-dimensional robotic workspace. Specifically, the paper's objectives are as follows;

- **Dataset**: Generating dataset in MATLAB that contains values of joint angles (θ) and the corresponding end effector positions.
- **Model Training**: Using the dataset to train machine learning models MLR (Multiple Linear Regression), DTR (Decision-Tree Regression), and ANN (Artificial Neural Network) for estimating end effector position.
- **Prediction Analysis**: Using the values from the dataset to validate the trained models and access their RMSE (root Mean Squared Error) for prediction.

Model Comparison: Evaluate the trained MLR, DTR, and ANN models on the basis of computational time and RMSE

Material and Methods:

Forward Kinematics 4 Degree Robot:

The position and orientation of the end effector calculation of a robotic arm or mechanism based on the joint angles or displacement. The forward kinematics of a 4 Degree-of-Freedom robot usually has four rotational joints that can be calculated by imposing a sequence of transformations from the base frame to the end effector frame. A distinct coordinate system is introduced by every joint; the overall transformation of the end effector from the base can be calculated. This process can be implemented using a programming language such as Python and C++ in MATLAB in just a short code. The code will comprise parameters for each point for the Denavit-Hartenberg approach, resulting in transformation matrices and performing a multiplication process on matrices to calculate the overall transformation. Meanwhile, the implementation of the obtained transformation to the end effector's original location, the orientation, and the final position of the end effector in space can be calculated. This seemed to serve as the essential and indispensable tool for the robot's controlling and movement planning respectively.

Denavit-Hartenberg Parameters:

Denavit-Hartenberg parameters are employed commonly for the characterization of the robot's structure. These parameters typically serve as the foundation for the conduction of robot kinematics and analysis. There are four DH parameters listed below in Table 1 to describe the orientation and position of a link. Each parameter for attaching reference frames to robots' assembly link is linked to a specific convention. This standardization of coordinate frames across spatial linkages ensures consistency and facilitates analysis.

The Denavit-Hartenberg (D-H) decided to use the homogeneous transformation matrix instead. This matrix represents the end effector orientation and position of the robots with respect to the joint angles. Nonetheless, it does not specify the arm arrangement needed to reach this position. Figure 2 depicts the diagram of the link coordinate system which is created using DH parameters

Table 1: D.H Parameters Definition and symbols

Symbol	DH Parameters	Description	Symbol
$\boldsymbol{d_i}$	Joint Offset	Intersections Length of the joint axis to common normal	$\boldsymbol{d_i}$
$\boldsymbol{\theta_i}$	Joint Angles	Angle in between the normal plane to the joint axis and orthogonal projections	$\boldsymbol{\theta_i}$
L	Link Lengths	Axis to the common normal distance	L
$\boldsymbol{\alpha_{i-1}}$	Twist Angle	Orthogonal projections of joint axis on the normal plane to common plane Angle.	$\boldsymbol{\alpha_{i-1}}$

$\boldsymbol{a_{i-1}}$ = Translation x_1 *axis*

$\boldsymbol{a_{i-1}}$ = Rotation around x_i *axis*

$\boldsymbol{d_i}$ = Translation around z_{i-1} *axis*

$\boldsymbol{\theta_i}$ = Rotation around z_{i-1} *axis*

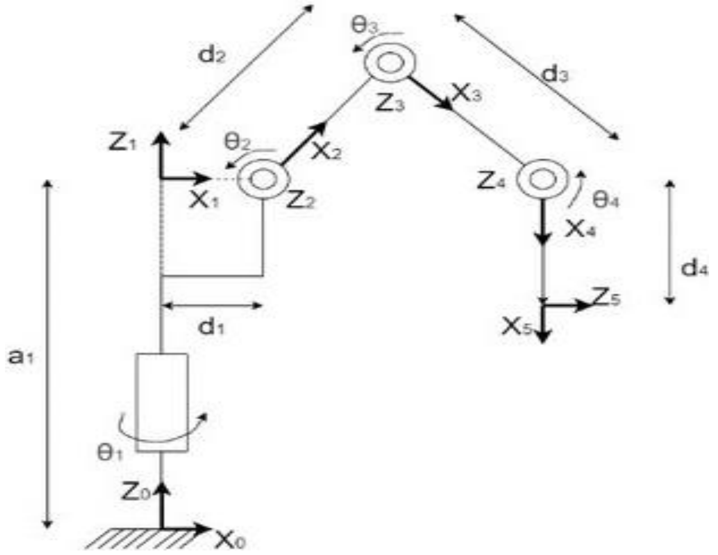

Figure 2: Denavit-Hartenberg Parameters Labeling of 4-Degree-of-Freedom Robot.

Denavit-Hartenberg (D-H) parameters are a collection of four parameters as depicted in Figure 2 for a 4 DOF robot that plays a crucial role in the robot kinematics. With the use of D-H representation, a systematic way to express relationships between consecutive links in a robotic serial manipulator is provided. Hence, this method provides a mathematical foundation, which is adaptable to a numerous robotic design. It also essentially defines the position and orientation of each link in relation to the other link that comes before this link [2].

Model of Forward Kinematics:

The 4-Degree-of-Freedom Forward kinematic model uses the Denavit-Hartenberg (DH) parameters in determining the end effector position and orientation based on joint angles for the robot. On an initial level, the DH parameters define each joint geometry including the link length, Joint angles, Joint offset, and Twist angles. With the help of these settings in place the transformation matrix for each joint explaining the transition between adjacent links as the robot moves is generated. A single transformation matrix indicating the total effect of each joint movement to the end effector frame from the base frame is generated via the multiplication of matrices. This transformation matrix provides a solution to the problems of forward kinematics by extracting the end effector orientation and location. The matrix for joint numbers 1 to ith can be calculated as shown in equation (1).

$$\boxed{A'_{i-1} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & \alpha \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}$$

$$A'_{i-1} = \begin{bmatrix} \cos\theta & -\sin\theta.\cos\alpha & \sin\theta.\sin\alpha & a.\cos\alpha \\ \sin\theta & \cos\theta\,.\cos\alpha & -\cos\theta.\sin\alpha & a.\sin\theta \\ 0 & \sin\alpha & \cos\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

By multiplication of matrices as depicted in equation (2) gives the transformation matric.

$$^0T_4 = {}^0A_1\,.\,{}^1A_2\,.\,{}^2A_3.\ {}^3A_4 \quad (2)$$

The model of the robot's forward kinematics was validated using MATLAB. A brief understanding of the kinematic behavior of the robot was obtained with the use of numerical analysis and also visualization in the MATLAB environment.

Table 2: D.H Parameters Description

Joint Angles θ	Symbol			
	a_{i-1}	d_i	a_{i-1}	θ_i
S1	0	0	0	0
S2	10	10	-90	0
S3	10	10	0	0
S4	10	10	0	0

For example, with a joint angle configuration of [S1 S2 S3 S4] for the values in Table 3, the transformation matrix is given below in Equation 3, and the visual representation is depicted in Figure 2

$$T = \begin{bmatrix} 1.0000 & 0.0000 & -0.0000 & -0.0000 \\ -0.0000 & -0.4481 & -0.8940 & -13.3992 \\ -0.0000 & 0.8940 & -0.4481 & -7.9014 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 \end{bmatrix} \quad (3)$$

Results:

Problems related to machine learning can be classified into three main types; regression, classification, and clustering. In the context of predicting claw positions in mapping with non-linear input poses, the problem at hand falls under regression. An intelligent approach is employed, utilizing machine learning algorithms for obtaining forward kinematics solutions. This Section focuses on predicting the claw position of a 4 Degree-of-Freedom Robot using Multiple Linear Regression (MLR), Artificial Neural Network (ANN), and Decision-Tree Regression Techniques. The algorithms of machine learning are trained and implemented in the MATLAB environment. The performance of these three models is being evaluated on the root mean squared error and R squared value basis. The predicted result is then compared with the actual value of the claw position. The result is basically the end effector position on the basis of joint angles.

Dataset For Training Machine Learning Model:

The dataset is generated through the code in MATLAB through initializing arrays to store workspace points and link points. The code iterates for all the combinations of joint angles within the specified limits of theta range (θ) for calculating forward kinematics using DH parameters. For every combination, the end effector position (x, y, z) is calculated and both workspace and the link point are stored. Finally, the dataset is populated with joint angles (θ) and the corresponding end effector positions. Figure 3 shows the robot workspace with links and Table 3 shows the dataset with values of joint angles and the end effector position.

Table 3: Data set Values

	$S1(\theta_1)$	$S2(\theta_2)$	$S3(\theta_3)$	$S4(\theta_4)$	X	Y	Z
1	-1.5708	-1.5708	-1.5708	-1.5708	-0.6031	-1.5708	-1.5708
2	-1.5708	-1.5708	-1.5708	-1.2217	-2.3396	-1.5708	-1.5708
3	-1.5708	-1.5708	-1.5708	0.8727	-5.0000	-1.5708	-1.5708
4	-1.5708	-1.5708	-1.5708	-0.5236	-8.2635	-1.5708	-1.5708
5	-1.5708	-1.5708	-1.5708	-0.1745	-1.7363	-1.5708	-1.5708
6	-1.5708	-1.5708	-1.5708	-0.1745	-5.0000	-1.5708	-1.5708
7	-1.5708	-1.5708	-1.5708	0.5236	-7.6604	-1.5708	-1.5708
8	-1.5708	-1.5708	-1.5708	0.8727	-193969	-1.5708	-1.5708
10000	-2.3701	-3.3498	-1.2217	-3.7892	-3.8495	-4.6590	4.8908

Columns 1 to column 4 show the values of joint angles (S1, S2, S3, S4), and columns 5, 6, and 7 show the corresponding end effector position variable values (x, y, z). The dataset as shown in Table 3 has 10,000 values. This dataset is used later for the training of machine learning algorithms.

Machine Learning Algorithms:

For determining the forward kinematics the machine learning algorithms used are Multiple Linear Regression (MLR), Artificial Neural Network (ANN), and Decision-Tree (DT) Regression techniques. The following machine learning algorithms are trained and implemented in Matlab

Multiple Linear Regression (MLR):

The Multiple Linear Regression is defined as a straightforward regression technique for employing multiple variables in predicting the response or output variable. A connection between each joint angle and the end effector position variables is used in this technique. The equation general form for Multiple Linear Regression of multiple independent variables and single dependent variables is expressed as follows in equation (4).

$$y_i = b_{0i} + b_{1i}.x_{1i} + \cdots + b_{4i}.x_{4i} \qquad (4)$$

Here for our work the "yi" represents the estimated end effector position of the 4 DOF robotic claw and x1 to x4 denotes the four joint angles (θ) of the robot S1, S2, S3, and S4. The first joint S1 is of arm claw, and the joints S2 and S3 are the elbow and shoulder joints respectively to move the arm to the desired position. The S3 joint is the base joint to rotate the robot arm.

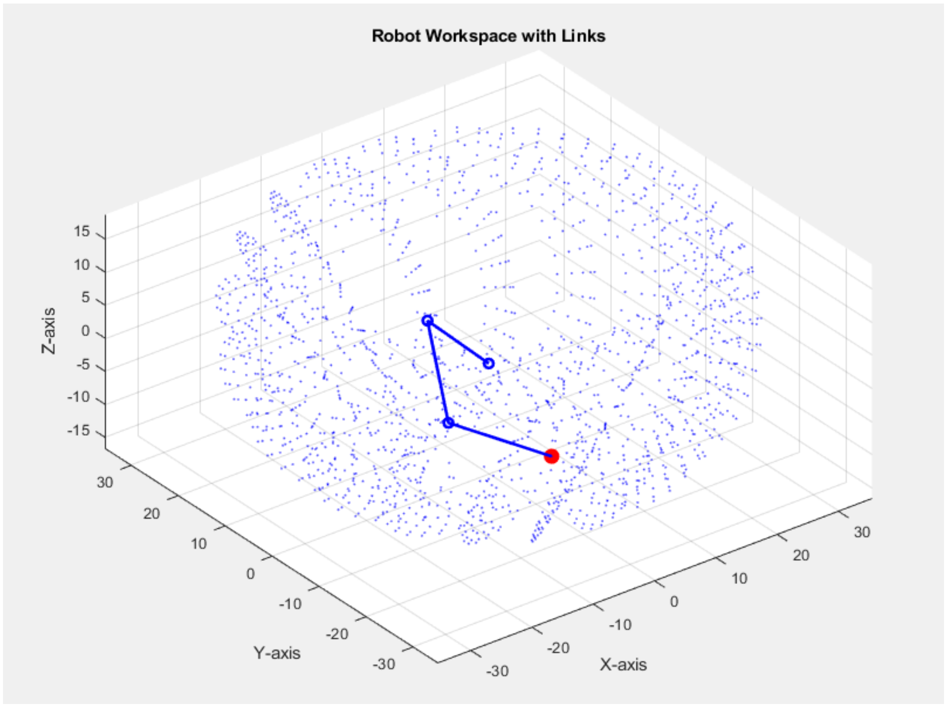

Figure 3: Workspace with Links demonstration on MATLAB.

Multiple Linear Regression (MLR):

The machine-learning algorithm is being used in predicting the target variable value by imposing the learning of simple decision rules inferred from the data features. This technique works by recursively partitioning the feature values (parent/root node) and then fitting a simple model specifically a constant value within each subset (Decision Node). Figure (4) illustrates the simple representation of Decision Tree Regression in general for this work with root node, Decision node, and leaf nodes. The decision tree regression predicts the continuous target variable by taking the average of the target values of all the training instances within each leaf node.

To illustrate how decision tree regression can be used to predict the end effector position (x, y, z) of a 4 DOF robot with 4 joint angles as input, we will train the Decision Tree Regression Model where the input features will be the joint angles ($(\theta_1, \theta_2, \theta_3, \theta_4$) and the output variable is the end effector position (x, y, z). The Decision Tree algorithm will learn to predict the position of the end effector based on each joint angle. After that the model performance will be evaluated on the root mean squared error (MSE) and R2 (R-squared error) basis. Hence in the work, the trained model can be used for the prediction. This prediction capability can be used in various robotics applications [8].

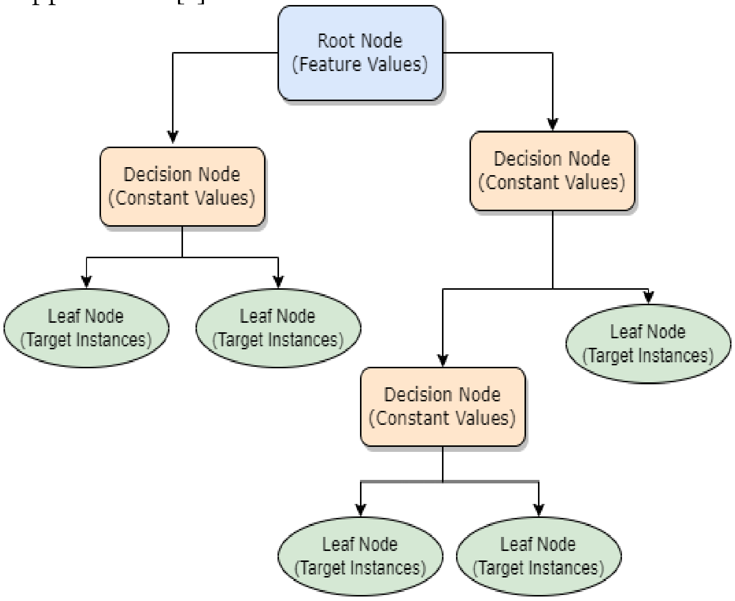

Figure 4: Decision Tree Algorithm 4-Degree-of-Freedom Robot.

Artificial Neural Network (ANN)

The ANN can be used to predict the end effector position of a 4-degree-of-freedom (DOF) robot based on 4 joint angles. We train the neural network using the training data. During the training of an ANN, the network adjusts its biases and weight iteratively to minimize the predicted end effector position and the actual positions in the training data. This is typically done by using the Gradient Descent algorithm on the backend of all the ML algorithms. Once the model is trained, we will evaluate its performance on the ground basis of RMSE (Root Mean Squared Error). The capability of ANN for capturing the complex and non-linear relationship between the inputs and the outputs makes it suitable for the prediction of the end effector position of a 4 DOF robot based on joint angles. Just by adjusting the architecture and training parameters of the neural network, the performance for specific predictions can be optimized. The ANN technique is widely used in robots. Figure 5 shows the artificial neural network topology used in this work.

During the ANN training, numerous parameters are adjusted, including hidden layer counts, neuron quality within each hidden layer, and the activation function choice applied at both the hidden and outer layers. The activation functions like sigmoid, tanh, Linear, and the (ReLU) Rectified Linear Unit are employed during the training duration. Moreover, optimization methods including the Stochastic Gradient Descent (SGD) and (Adam) adaptive moment estimation are utilized for refining the weights during the duration of training. The keen and careful selection of all these parameters leads to notable enhancement in prediction accuracy. The of epochs is set at 1,000 as a maximum number, as empirical evidence shows that the accuracy, as measured by metrics like (R2) R-squared and (RMSE) root mean square error, does not notably improve beyond this threshold. The result of training ANN in the MATLAB environment for our work is depicted in Table 4.

Table 4: ANN Model on MATLAB Platform

Data Division	**Random**
Performance	Mean Squared Error
Epoch	1000
Computational Time	0:00:13
Performance	0.000100
Gradient	0.266

Comparison and Discussion:

The dataset is compiled comprising of 10,000 entries for both inputs (joint angle values) and outputs (end predictor values) which were subsequently utilized for training of the Machine Learning Algorithms. The end effector position actual value is calculated through an analytical approach using the forward kinematics with DH parameters techniques. The predicted value is obtained from the machine learning algorithms implemented in the MATLAB environment. The performance of these algorithms is accessed by measuring the variance between the predicted and the actual value. However, the model's evaluation may vary depending on the random selection of samples within the training set, potentially resulting in either underestimation or overestimation.

The performance of every algorithm is evaluated based on RMSE. Table 4 shows the values of the end effector determined through (MLR) Multiple Linear Regression, (ANN) Artificial Neural Network (ANN) and Decision-Tree (DT) Regression. The actual value is also written. The values are found for the joint angle values. The visual representation of the actual and predicted end-effector value is shown in Figure (5). The values given for actual and predicted end effector position are calculated from joint angles given in equation 5.

$$[\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4] = \left[0, \quad -\frac{pi}{2}, \quad -\frac{pi}{2}, \quad 0\right] \qquad (5)$$

The dataset is compiled comprising of 10,000 entries for both inputs (joint angle values) and outputs (end predictor values) which were subsequently utilized for the training of the Machine Learning Algorithms. The actual value of the end effector position is calculated through an analytical approach using the forward kinematics with DH parameters techniques.

Table 5: Actual And predicted positions

End Effector Position	**X**	**Y**	**Z**
Actual Value	26.841406	-10.000000	18.918461
Multiple Linear Regression (MLR)	17.735263	-3.969662	17.148879
Decision Tree Regression	28.214849	-12.180011	18.782643
Artificial Neural Network	26.871022	-11.481421	18.872894

The predicted value is obtained from the machine learning algorithms implemented Here's a table summarizing the (R2) R-squared and (RMSE) Root Mean Squared Error values for each machine learning algorithm.

Table 6: Estimated Error

	Multivariable Linear Regression	**Decision Tree Regression**	**Artificial Neural Network**
RMSE(X)	9.105143	5.374741	0.030384
RMSE(Y)	6.030947	5.399014	1.481579
RMSE(Z)	0.000000	0.000000	0.045567
R2 (X)	0.628651	0.868257	0.999999
R2 (Y)	0.805418	0.840723	0.987424
R2 (Z)	1.000000	1.000000	0.999938

Table 5 shows the (RMSE) Root mean squared error and (R2) R-squared error for the (MLR) Multiple Linear Regression, (ANN) Artificial Neural Network, and Decision-Tree (DT) regression, which is found on MATLAB because of each algorithm. For Multiple Linear

Regression (MLR) RMSE values are relatively high indicating a noticeable deviation between the predicted and actual values. R2 values suggest a moderate to good fit for the X and Y coordinates, but an excellent fit for the Z coordinate. Thus, MLR did not satisfy the demand and lags in capturing the complex relationships between variables. Hence, it had a higher error as compared to the other two algorithms used in this study.

For the Artificial Neural Network, the root mean squared error is extremely low as mentioned earlier in Table 5. So, there is a very small deviation between the actual and predicted value which can be easily observed in Figure 5 in comparison with MLR and Decision-Tree Regression. R2 values are close to one suggesting an excellent fit for all coordinates. Hence, it proves that for this work ANN excels in capturing the complex patterns and non-linear relationships, resulting in better performance compared to other algorithms to predict end effector position of 4 DOF on the basis of joint angles.

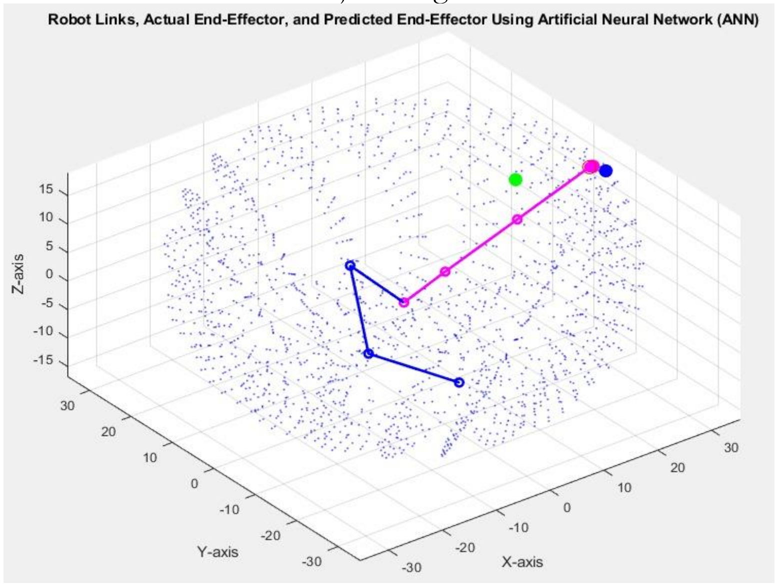

Figure 5: Graphical Comparison of different algorithms.

In summary, the decision tree regression and artificial neural network models outperform multivariable linear regression in accurately predicting the end effector position. The decision tree regression model performs exceptionally well, achieving zero error for the Z coordinate. The artificial neural network model demonstrates outstanding performance with negligible errors across all coordinates, displaying its effectiveness in handling complex data patterns for the work

Conclusion:

This study explores the machine learning algorithms to predict the operational area of the 4-Degree-of-Freedom robot end effector position based on joint angles. The Forward Kinematics While Multiple Linear Regression shows moderate performance, Decision Tree Regression excels with lower errors. However, Artificial Neural Network emerges as the top performer, showcasing remarkable accuracy in predicting end effector positions. These findings highlight the potential of machine learning in enhancing robotics autonomy and task planning.

References:

[1] "Forward and Inverse Kinematics Solution of A 3-DOF Articulated Robotic Manipulator Using Artificial Neural Network | Sharkawy | International Journal of Robotics and Control Systems." Accessed: May 05, 2024. [Online]. Available: https://pubs2.ascee.org/index.php/IJRCS/article/view/1017

[2] O. Carbajal-Espinosa, L. Campos-Macías, and M. Díaz-Rodriguez, "FIKA: A Conformal Geometric Algebra Approach to a Fast Inverse Kinematics Algorithm for an Anthropomorphic Robotic Arm," Machines, vol. 12, no. 1, p. 78, Jan. 2024, doi:

10.3390/MACHINES12010078/S1.

[3] E. Bayro-Corrochano, “A Survey on Quaternion Algebra and Geometric Algebra Applications in Engineering and Computer Science 1995-2020,” IEEE Access, vol. 9, pp. 104326–104355, 2021, doi: 10.1109/ACCESS.2021.3097756.

[4] J. Denavit and R. S. Hartenberg, “A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices,” J. Appl. Mech., vol. 22, no. 2, pp. 215–221, Jun. 1955, doi: 10.1115/1.4011045.

[5] I. Virgala, M. Kelemen, M. Varga, and P. Kuryło, “Analyzing, Modeling and Simulation of Humanoid Robot Hand Motion,” Procedia Eng., vol. 96, pp. 489–499, Jan. 2014, doi: 10.1016/J.PROENG.2014.12.121.

[6] R. Gao, “Inverse kinematics solution of Robotics based on neural network algorithms,” J. Ambient Intell. Humaniz. Comput., vol. 11, no. 12, pp. 6199–6209, Dec. 2020, doi: 10.1007/S12652-020-01815-4/METRICS.

[7] M. Soori, B. Arezoo, and R. Dastres, “Artificial intelligence, machine learning and deep learning in advanced robotics, a review,” Cogn. Robot., vol. 3, pp. 54–70, Jan. 2023, doi: 10.1016/J.COGR.2023.04.001.

[8] N. Kovincic, H. Gattringer, A. Müller, and M. Brandstötter, “A Boosted Decision Tree Approach for a Safe Human-Robot Collaboration in Quasi-static Impact Situations,” Mech. Mach. Sci., vol. 84, pp. 235–244, 2020, doi: 10.1007/978-3-030-48989-2_26.