Optimising Robot Manipulator Inverse Kinematics with Genetic Algorithm

K.B.A.D.C.T.D. Perera
Department of Mechanical Engineering
Faculty of Engineering,
University of Sri Jayewardenepura
Rathmalana, Sri Lanka
chamoddcs@gmail.com

C. D Makavita
Department of Mechanical Engineering
Faculty of Engineering,
University of Sri Jayewardenepura
Rathmalana, Sri Lanka
makavita@sjp.ac.lk

M. U. B Dias
Department of Mechanical Engineering
Faculty of Engineering,
University of Sri Jayewardenepura
Rathmalana, Sri Lanka
uvindubigumjith@gmail.com

***Abstract*— As traditional, robot manipulator inverse kinematics solutions are computationally intensive, numerous algorithms has been introduced to minimize the time required to compute inverse kinematics. In this paper an Artificial Neural Network (ANN) was utilized to solve the inverse kinematics of a robot manipulator, and a Genetic Algorithm (GA) was used to optimize the ANN weights and biases. This research primarily examined how crossover and mutation variation affect inverse kinematics problems solved with a GA optimised ANN model. In GA optimisation, single point and two-point crossover were performed to test the crossover effects while Gaussian and uniform mutations were used in the mutation. The ANN GA optimised inverse kinematics model was trained and tested using the Forward Kinematics data set. The single point crossover and Gaussian mutation produced the best results based on the results for the generated data. In conclusion GA can be used with variety of parameters to achieve optimal results which may differ depending on the parameters chosen.**

Keywords—Inverse Kinematics (IK), Evolutionary Algorithm (EA), Genetic Algorithm (GA), Forward Kinematics (FK), Neural Network (NN)

I. Introduction

Inverse kinematics (IK) is used to calculate the joint angles for a specific Cartesian location and end-effector orientation. To tackle IK problems, analytical and iterative methods might be applied. The major issue with this approach is that it takes more time to calculate IK solutions for robot manipulators with higher degrees of freedom (DOF) [1].

To reduce the calculation time required to solve the IK of a robot manipulator, numerous strategies are used, including different types of Evolutionary Algorithms (EA). EA has evolved as an essential optimisation and search strategy throughout the past decade. EA includes genetic algorithms (GA), a type of search algorithm that simulates the processes of natural selection and genetic inheritance. Specifically, evolutionary algorithms maintain a population of structures that evolve in accordance with rules of selection and other operators, including recombination and mutation. Each member in the population is assigned an environmental fitness score. Utilizing the available fitness information, selection concentrates on those with high fitness levels [2].The GA method differs significantly from conventional approaches to solving IK issues. GAs are blind and depend on the population to solve problems [3]. There are several forms of crossovers, mutations, fitness functions, and population selection that may be used solve the IK of a robot using genetic algorithm with different changes in parameters. The majority of the time, researchers mix numerous other learning algorithms with GA and test them using various robot manipulators. Neural Network GA [1], Fuzzy logic GA [4], Niching GA [5], and Artificial Immune GA [6] are some examples of such mixed algorithms GA.

In Artificial neural networks, a problem's solution is learned from a training dataset, which is a novel computing method. The original ideas for neural networks came from research into the function of biological nervous systems, particularly the human brain [1].

GA employ a population of potential solutions that change over time as a result of the usage of operators for selection, crossover, and mutation. A GA's objective is to identify the optimal solution to a given issue based on an evaluation of each potential solution's quality by a fitness function [7]. Crossover is the creation of a new solution by combining two existing ones. The reproductive process of natural selection adds superior members to the population. In an effort to produce superior offspring, the operator of the crossover is added to the reproduction pool. Mutation is the next step after crossover is done. The main advantages in mutation is preventing the local optima. Also mutation is very important in randomly spreading the algorithm's information [8].

Combining ANN with GA allows for the optimisation of the neural network's weights and biases to enhance performance on a specific task. This method can be very beneficial for system with many parameters where it is challenging to manually improve the network [3].

II. Methodology

A. *Choosing a Robot Manipulator for Further Research*

As the IK problem becomes increasingly difficult to solve as the number of degrees of freedom (DOFs) of the manipulator increases, this study uses the simple and widely used 3R planar robot for testing the proposed algorithm. The kinematic structure of a 3R planar robot is rather basic, using only three revolute joints in a two dimensional plane. The 3R planar robot's inverse kinematics has been thoroughly studied and is accessible in a closed-form solution, making it an ideal platform on which to evaluate and contrast the effectiveness of various optimisation techniques.

For the purposes of this study, the joint offset is ignored, and the end effector is regarded as the last link end point. For this study link lengths are considered to be **0.6 m**, **0.45 m**, and **0.2 m**.

B. GA optimised ANN for solve IK

In this section mainly focused about two frequently used methods for optimize ANNs,

- Backpropagation,

By using a loss function, backward propagation is used to calculate the difference between target and actual output values [8]. The gradient base methods can be used to update the weights and biases of the NN to get a minimal loss function. The weights and bias updates for forward propagation are done in a sequence of epochs to obtain the optimal output for the given input data.

- GA

When the GA compare with the backpropagation.GA is making possible solutions (chromosome) and improve the model over and over again. Selection, Crossover, Mutation are main components of a genetic algorithm. By simulating the way natural selection works, these genetic operators make it easier to search through the search area. The genetic algorithms operators allow to explore the search space more effectively by modelling the process of natural selection than the backpropagation [11].Therefore continue this study GA optimisation was selected to use with ANN.Several benefits to using GA in combination while training ANN rather than using backpropagation were mentioned in Table I.

TABLE I. GA ADVANTAGES

Backpropagation	**GA**
Can be stuck in local optima	Able to avoid being stuck in local optima and perhaps discover a more optimal global solution.
NN will fail in converge if the initial weights and biases select incorrectly.	Less sensitive to initialization and may converge to a satisfactory solution independent of the initial weights and biases
If the training data set is small model can lead to overfitting.	By using GA the NN overfitting to training data can be fixed
Training data can be over fit due to the NN leans to memorize the taring instances instead of generalizing [10].	Enhance the generalization to new data by exploring a larger variations.

III. Procedure

A. Developing Forward Kinematics model and Inverse Kinematic model for 3R planar robot

1) 3R robot manipulator geometrical equations

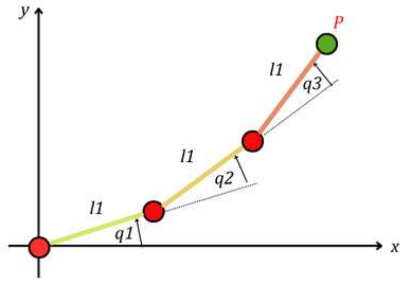

Fig. 1. 3R manipulator structure in xy plane.

- Link 1 length $l1$
- Link 2 length $l2$
- Link 3 length $l3$

- Link 1 angle respective to the x axis = q_1
- Link 2 angle respective to the x axis = q_1+q_2
- Link 3 angle respective to the x axis = $q_1+q_2+q_3$

Assumptions,

- Joint offset are negligible

Notations,

According to the Fig 1 'P' is the position of the end effector and X_{ef} and Y_{ef} are the end effector x,y coordinates.

$$P = \begin{bmatrix} X_{ef} \\ Y_{ef} \end{bmatrix} \tag{1}$$

Therefore Forward Kinematic of the 3R planar robot can be written as,

$$P = \begin{bmatrix} l1\cos(q1) + l2\cos(q1+q2) + l3\cos(q1+q2+q3) \\ l1\sin(q1) + l2\sin(q1+q2) + l3\sin(q1+q2+q3) \end{bmatrix} \tag{2}$$

Using this model (2) the MATLAB was used to generate a dataset which can be used to train and test the ANN GA model. For the dataset 1000 random dataset was generated according to the FK (2) which is built previously.

- Defined the link lengths in meters,
  - ✓ l1 = 0.6
  - ✓ l2 = 0.45
  - ✓ l3 = 0.2
- Random q1, q2, and q3 joint angles were generated using **For loop**. The angles were created in degrees. The joint angles were generated under specified limitations. Where all the joint has min 0 ° and max 180 ° angles.

B. Developing ANN GA model

In this case, the main goal is to develop an ANN to get the inverse kinematics results for a 3R robot manipulator based on GA optimisation. Where the GA is going to use for the weights and biases optimisation in ANN.

1. Inputs for the ANN are the calculated x, y, and orientation from the forward kinematics model. The initial biases and weights for the ANN are default values from the NN tool.
2. The FK results were separated into 2 section
   a. Training data set (80% from FK results)
   b. Testing data set (20% from FK results)
3. The ANN was separated in to 3 sections for reduce the complexity of the problem and make easy on analysis
   The ANN architecture include Fig. 2,
   a. Feed Forward NN
   b. No of hidden layers =1
   c. No of neurons per layer =30

d. Activation function = ReLu

e. ~~Input~~ features =3 (x , y , orientation)

f. Output features =3 (joint angles)

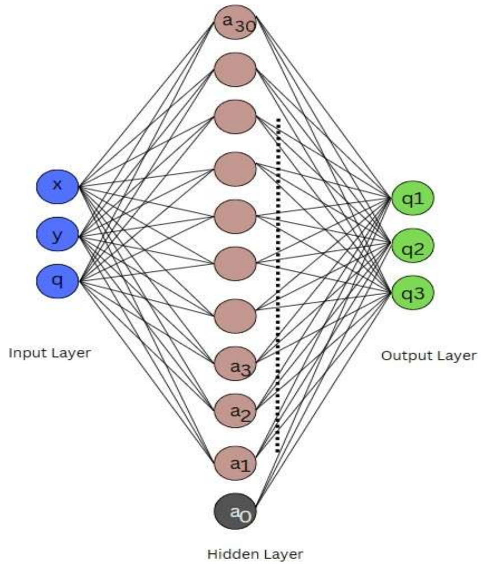

Fig. 2. NN architecture.

According to Fig. 2, the inputs x, y, and q (orientation) enter into the NN, and each link between neurons has associated weights. As mentioned before, the starting weights and bias are default settings in the programmed NN. According to the diagram, a_0 represents the bias value. The bias value can be used to shift the activation function, providing further flexibility in the learning process.

$$z = f(x) = \sum_{i=1}^{n} (x_i \times w_i) + b \tag{3}$$

- n- Number of neurons
- x_i – Input value at i instant
- w_i – Weight between Input and the neuron
- b – bias
- z – Weighted sum

The weighted sum is processed through an activation function, which generates nonlinearity into the model. The activation function modifies the neuron's output after ensuring that it activates or deactivates in accordance with the input values. In this model, the NN was combined with the ReLu activation function, which stands for Rectified Linear Unit. The ReLu function output values directly when the inputs are positive . The ReLu function can save processing time because of its simplicity.

4. Fitness Function is based on (4) RMSE (Root Mean Squad Error).The n is the count of total samples. The purpose of the fitness function is to measure how well or poorly network performed

$$RMSE = \sqrt{\frac{1}{n}\sum_{1}^{n} (target_i - estimate_i)^2} \tag{4}$$

- n- Number of data
- $target_i$ – Target data value at i instant
- $estimate_i$ – Estimate data value at i instant

5. The NN weights and biased were optimised by the looped GA. The optimisation is done for two type of crossover and mutation configuration.

a. Crossover types: 'Single point crossover' and 'Two point crossover'

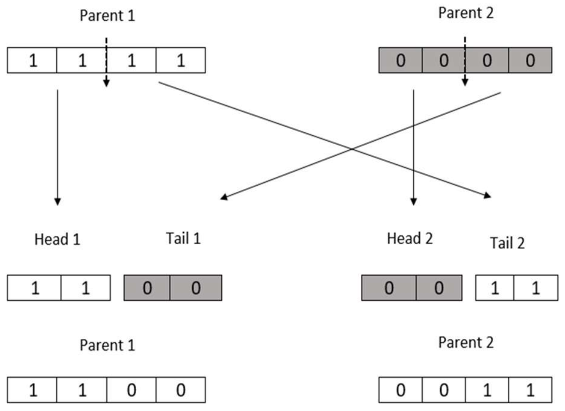

Fig. 3. Single point crossover example [8].

In single point cross over the individuals are cut once at a certain points and sections by exchanges parts as shown in Fig. 3.

In two points crossover a chosen two crossover points mated between two parents as shown in Fig. 4.

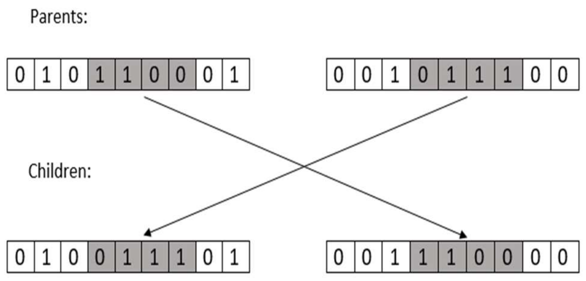

Fig. 4. Two point crossover example [8].

b. Selection method: Tournament Selection

c. Mutation types: 'Uniform' and 'Gaussian' [12]

d. Crossover rate 0.8 Mutation rate 0.2

e. Population size 50,Max iteration 2000

The flowchart in Fig. 5 illustrates the optimization process of the created model, where the GA is used to modify the weights and biases of an ANN in order to minimize prediction errors .The neural network processes input data using present weights and biases to generate expected results. The error (or loss) between the desired and estimated outputs is computed. This was done with the RMSE value based on (4). After calculating the RMSE, the method determines whether the current number of iterations has achieved the predefined maximum limit, which in this case is 2000 iterations. The objective of the selection process is to choose individuals from one generation to generate mating sets for the next generation.

Multiple individuals are chosen at random from a generation, and the best among them are chosen as parents in the tournament selection process. The fitness of each individual on the given list is tested using the previously mentioned fitness function. The individuals with the best fitness among them were chosen as the winners to form the next mating pool.

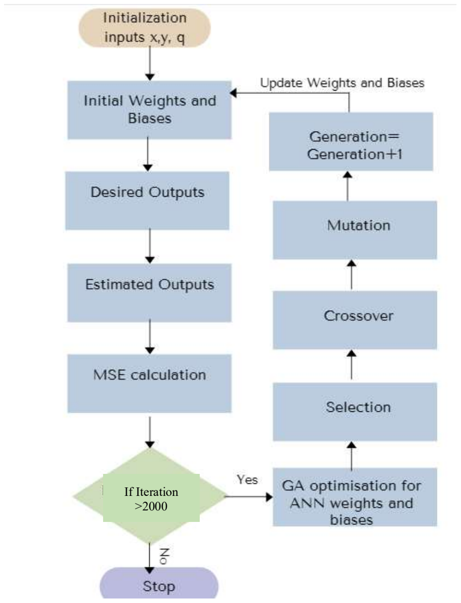

Fig. 5. ANN GA model flow chart.

The crossover rate is the frequency with which recombination occurs between pairs of individuals. When crossover is 0.8, 80% of the time the crossover occurs with new offspring are created by uniting two parents. The remaining 20% of children will be exact clones of their parents with no crossover. The mutation rate is a quantity in genetic algorithms that describes how frequently random changes are applied to the genetic individuals (chromosomes) of in a population. Mutation is responsible for maintaining genetic diversity and allowing the algorithm to seek new potential solutions rather than simply exploiting existing ones. If the mutation rate is 0.2, that means that 20% of an individual's genes will change randomly.

IV. Simulation And Results

MATLAB 2020a was used in the study to simulate the results of IK for the 3R robot manipulator. In this study, crossover and mutation variation were done to get the best results for IK. The MATLAB program was based on the flow chart in Fig. 5. Up to 2000 iterations, these stages were repeated depending on crossover and mutation independently. (Because the RMSE curve did not converge smoothly, 3000 iterations were used in the two point crossover) .

A. Comparison for Testing data set and ANN GA generated data

After the ANN GA was optimised using the training data set, the optimised biases and weights were saved and a new neural network was created for the optimised values.

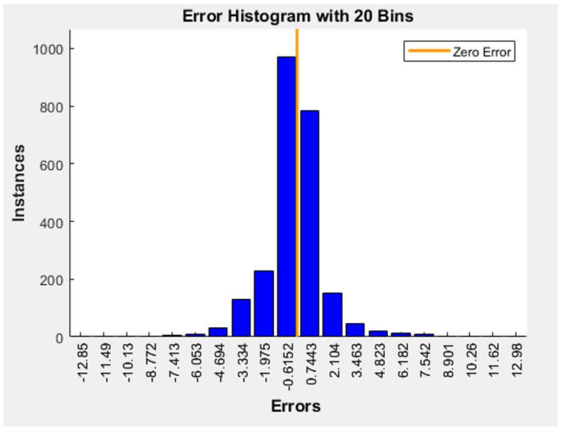

Fig. 6. Error histogram for single point crossover uniform mutation.

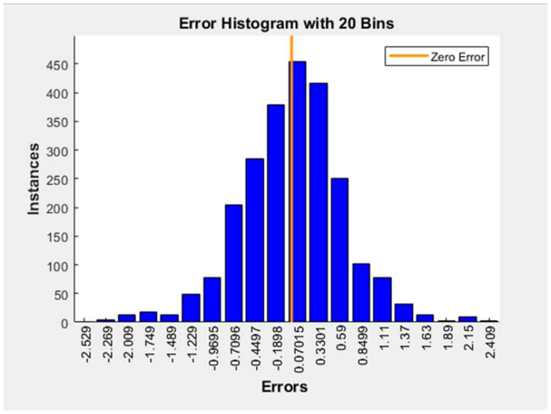

Fig. 7. Error histogram for single point crossover gaussian mutation iterations.

According to the Fig. 6 the single point crossover and the uniform mutation model errors are more spread out, indicating a less accurate model compared to single point Gaussian mutation as shown in Fig.7.But in two point crossover and uniform mutation errors are tightly clustered around zero, indicating high accuracy compared to single point crossover.

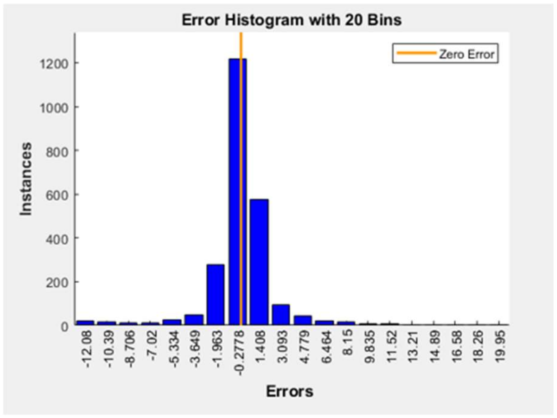

Fig. 8. Error histogram for two point crossover uniform mutation.

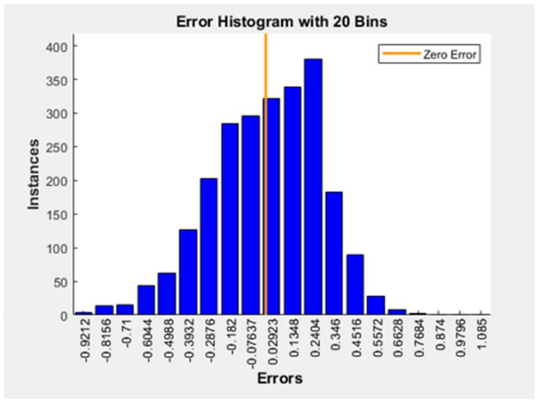

Fig. 9. Error histogram for two point crossover gaussian mutation iterations.

TABLE II. R VALUES FOR DATA SETS

Crossover type	Mutation type	R values(Correlation coefficient)	
Single point	Uniform	Training Data	0.31
		Vallidation Data	0.26
		Test Data	0.32
Single point	Gaussian	Training Data	0.83
		Vallidation Data	0.84
		Test Data	0.83
Two point	Uniform	Training Data	0.25
		Vallidation Data	0.40
		Test Data	0.34
Two point	Gaussian	Training Data	0.97
		Vallidation Data	0.97
		Test Data	0.97

The error histogram of the two point crossover and uniform mutation shows the better prediction which close to the actual values as shown in Fig. 8. But the R value is very lower for that model indicates predictions the model failed to capture underlying pattern than the other three process. The Fig. 9 shows better R values but the error histogram is not clustered around zero. That mean when compare to other models it shows the larger prediction with errors. Therefore, when comparing the R values and the histogram the Fig. 7 shows the best performance.

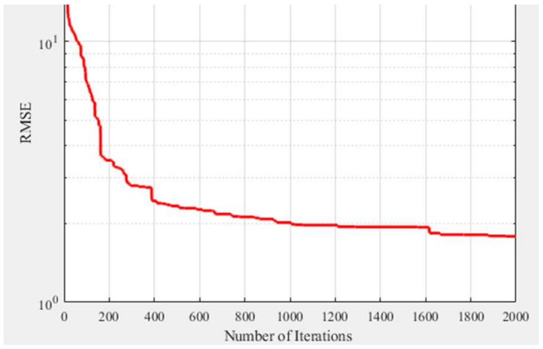

Fig. 10. RMSE vs Number of iterations plot for single point crossover uniform mutation.

The model in Fig. 11 model exhibits a fairly rapid reduction in RMSE initially, showing that it quickly adapts to find promising regions in the solution space when compare with other single point crossover model as shown in Fig. 10 and it reaches the Minimum RMSE value. 2000 iterations of the single point crossover were utilized for both models, and the RMSE curve converged as desired. However, compared to single crossover setups, two point crossover models Fig. 12 and Fig. 13 show larger fluctuation throughout iterations. In order to obtain more precise results, the procedure was repeated up to 3000 times for a two-point crossing.

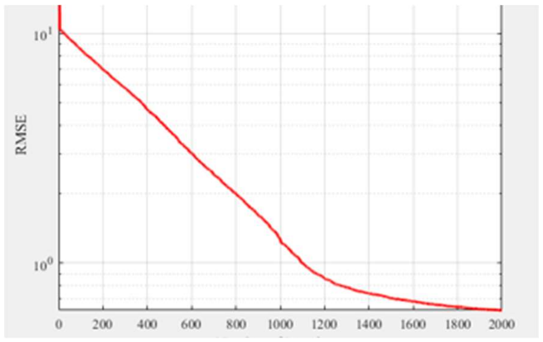

Fig. 11. RMSE vs Number of iterations plot for single point crossover gaussian mutation.

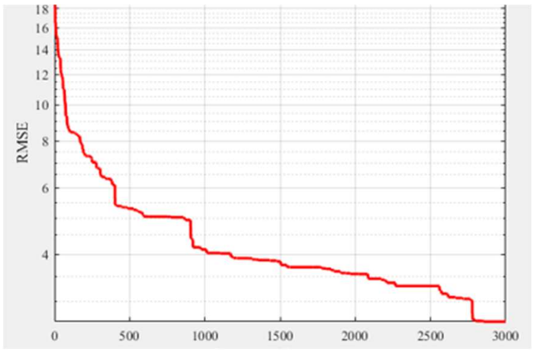

Fig. 12. RMSE vs Number of iterations plot for two point crossover uniform mutation.

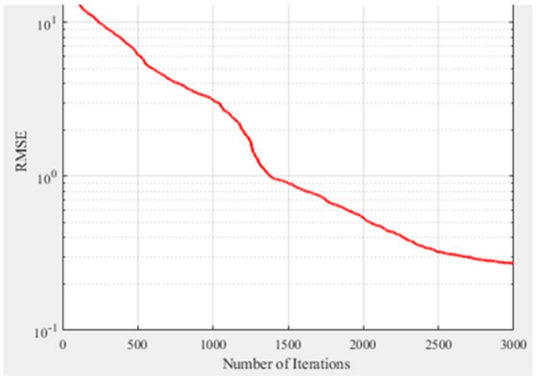

Fig. 13. RMSE vs Number of iterations plot for two point crossover gaussian mutation.

When compared to other results, Fig. 7 provides the best results with reduced iteration when the RMSE value becomes zero. Fig. 7 illustrate the results of the Gaussian mutation with the fewest errors. By comparing these data, the single point Gaussian mutation was found to be the best fit for this problem. However, as shown in Fig. 12, the two point crossover also achieved lower error levels for uniform mutation.

V. Discussion

In general, it is challenging to identify the optimal mutation and crossover operator capable of producing all desired effects. The effects of mutation operators vary across different genetic algorithms and problem-solving techniques. For this 3R problem, the optimal crossover operator and mutation operator were chosen based on the total errors in the optimised model compared to the testing data for all angles. The MATLAB program was used to compare the overall error values for each of the angles based on a simple grid search procedure. In this section of the results, there are significant differences between the values derived by the model from the target data set and the actual values. The optimal value for the crossover operator probability is dependent on the nature of the problem and the population characteristics. In overall, an increased crossover probability can expedite solution convergence. But it can also result in a loss of population variation. The efficiency of the genetic algorithm can be negatively affected by lots of variations. The crossover probability specified for simulation is 0.8. Typically, set the mutation operator probability to a low value, low value ensures that only a negligible proportion of the population mutates with each generation. On the other hand, a greater probability of crossover can lead to a greater exploitation of excellent solutions. However, it can also reduce population diversity and increase the possibility of converging on an unacceptable solution. Therefore there is a trade-off between selecting correct crossover rate and mutation rate for the model. Consequently, the crossover and mutation probabilities must also be optimised in

problems of this nature. Unbalanced crossover and mutation probabilities resulted in significant error margins in the results for certain angle values. If the optimised model is in perfect condition, the generated data set plot should lie on the target data set plot. By increasing the hidden layers in an ANN, the accuracy of the model can be increased. But more hidden layers than the required number of layers will over fit the training data set. By changing GA parameters, the accuracy of the model can also be improved. Increasing the populations and count of the maximum generation. The optimisation performance of the model can be increased.

The R-value, also known as the correlation coefficient In the context of an ANN or any regression model, the R-value indicates how well the model's predictions match the actual data. The higher R value indicates that the model's predictions are closely aligned with the actual data, suggesting that the model is capturing the underlying patterns in the data well. According to the Table II single point crossover Gaussian mutation and two point crossover Gaussian mutation gave the best R values. But when comparing the RMSE plots and the error distribution of these models. single point crossover performed the best with the Gaussian mutation. Single point crossover Gaussian mutation model maintains a very steady decrease in RMSE, it generate greater variety of solutions, enhancing the algorithm's ability to explore and exploit different regions more thoroughly. Thus, it results in superior convergence characteristics smooth reduction in RMSE and reaching lower error values more quickly. A well performing model will typically have an error histogram that is centred around zero, symmetric, and narrow. Single point ccrossover with Gaussian Mutation Fig.11 and Two point crossover Uniform mutation Fig. 12 achieve the lowest final RMSE among these configurations by indicating the most accurate models. As a result, crossover and mutation type can have an impact on the final outcomes of a GA optimised model.

VI. Conclusion

In this research paper, an ANN-GA optimised model was presented to solve IK for a 3R robot manipulator. The crossover and mutation variations were done for GA to check the model's accuracy. Based on the NN and the GA, the optimised model was checked for "single point and two point' crossover variations and also for the "uniform and Gaussian' mutation variation. Inputs for the ANN-GA model were generated using developed FK models. As the fitness function for the GA, RMSE was used. Instead of using backpropagation optimising, the GA is used and continues until the 2000 maximum iterations. By using the optimised network errors based on the testing data set, single point crossover and the Gaussian mutation gives the best results for the developed model. However, the optimum parameters for the ANN GA model may vary depending on the problem nature and complexity. The crossover rate, mutation rate, and fitness function may all have an effect on the results of the model. This work focused at how crossover and mutation types affect the ANN GA model.

References

[1] R. Köker, "A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization," *Information Sciences*, vol. 222, pp. 528–543, Feb. 2013, doi: 10.1016/j.ins.2012.07.051.

[2] W. M. Spears, K. A. Jong, T. Bäck, D. B. Fogel, and H. Garis, "An overview of evolutionary computation," in *Machine Learning: ECML-93*, vol. 667, P. B. Brazdil, Ed., in Lecture Notes in Computer Science, vol. 667. , Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 442–459. doi: 10.1007/3-540-56602-3_163.

[3] J. K. Parker, A. R. Khoogar, and D. E. Goldberg, "Inverse kinematics of redundant robots using genetic algorithms," presented at the 1989 IEEE International Conference on Robotics and Automation, IEEE Computer Society, Jan. 1989, p. 271,272,273,274,275,276-271,272,273,274,275,276. doi: 10.1109/ROBOT.1989.100000.

[4] C.-Y. Chen, M.-G. Her, Y.-C. Hung, and M. Karkoub, "Approximating a Robot Inverse Kinematics Solution Using Fuzzy Logic Tuned by Genetic Algorithms," *Int J Adv Manuf Technol*, vol. 20, no. 5, pp. 375–380, Sep. 2002, doi: 10.1007/s001700200166.

[5] S. Tabandeh, W. W. Melek, and C. M. Clark, "An adaptive niching genetic algorithm approach for generating multiple solutions of serial manipulator inverse kinematics with applications to modular robots," *Robotica*, vol. 28, no. 4, pp. 493–507, Jul. 2010, doi: 10.1017/S0263574709005803.

[6] "A hybrid swarm intelligence of artificial immune system tuned with Taguchi–genetic algorithm and its field-programmable gate array realization to optimal inverse kinematics for an articulated industrial robotic manipulator - Hsu-Chih Huang, Sendren Sheng-Dong Xu, Chang Han Wu, 2016." Accessed: Oct. 22, 2022. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/1687814015626380

[7] S. Agatonovic-Kustrin and R. Beresford, "Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research," *Journal of Pharmaceutical and Biomedical Analysis*, vol. 22, no. 5, pp. 717–727, Jun. 2000, doi: 10.1016/S0731-7085(99)00272-1.

[8] T. C. Institute, "GA (Genetic Algorithm) Operators," Our Education | Best Coaching Institutes Colleges Rank. Accessed: Jun. 18, 2023. [Online]. Available: https://blog.oureducation.in/ga-genetic-algorithm-operators/

[9] H. Z. Khaleel, "Inverse Kinematics Solution for Redundant Robot Manipulator using Combination of GA and NN," *alkej*, vol. 14, no. 1, pp. 136–144, Apr. 2018, doi: 10.22153/kej.2018.10.008.

[10] "Neural Networks in Computer Intelligence | Guide books." Accessed: Feb. 28, 2023. [Online]. Available: https://dl.acm.org/doi/abs/10.5555/541594

[11] P. Karlra and N. R. Prakash, "A neuro-genetic algorithm approach for solving the inverse kinematics of robotic manipulators," in *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483)*, Oct. 2003, pp. 1979–1984 vol.2. doi: 10.1109/ICSMC.2003.1244702.

[12] "Create custom shallow neural network - MATLAB network." Accessed: Jun. 18, 2023. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/network.html