Journal of Field Robotics

WILEY

RESEARCH ARTICLE

Optimal Trajectory Generation of Various English Alphabets Using Deep Learning Model for 3-R Manipulator

Swapnil Murai[1] | Rahul Das Vairagi[2] | Vijay Bhaskar Semwal[3]

[1]Maulana Azad National Institute of Technology, Bhopal, India | [2]Indian Institute of Technology Varanasi (IIT-BHU), Varanasi, India | [3]Assistant Professor of CSE Department, Maulana Azad National Institute of Technology, Bhopal, India

Correspondence: Swapnil Murai (testswapnilmurai@gmail.com)

Received: 14 August 2024 | **Revised:** 13 November 2024 | **Accepted:** 16 January 2025

Keywords: artificial neural network (ANN) | deep learning (DL) | FABRIK | inverse kinematics (IK) | robotics manipulator

ABSTRACT

The modern era of medicine and industry extensively utilizes the manipulator's hand for a variety of vital automated activities. Handling a manipulator hand is a complex task. Due to the nonlinear characteristics of inverse kinematics (IK) mathematical model, inverse kinematics is a time-consuming and laborious procedure, making it difficult to provide a mathematical solution. This research employs a 3-R (revolute) robotic manipulator to achieve joint trajectories for drawing different alphabets and shapes. The IK problem has been solved using a hybrid model. The model is a hybrid of an artificial neural network (ANN) based model, the forward and backward reaching inverse kinematics (FABRIK) technique provides stability and the control barrier function (CBF) with the Lyapunov function. Using the proposed model, coordinates for different alphabets and shapes within the confined workspace were calculated. The ANN automatically obtains specific end-effector coordinates. This model combines the CBF with the Lyapunov function to ensure that a safe region is selected. The accuracy of the model exceeds 99.5%. We have calculated the mean square error (MSE) as 1.66, the root mean square error (RMSE) as 1.25, and the mean absolute error (MAE) as 0.96 for our model. The error between the model's predicted and actual coordinates also demonstrates letter coordinates and shapes drawn using a physical 3R manipulator model. As a result, this method can be applied to precisely estimate the angles in intricate 3DoF inverse kinematics models.

1 | Introduction

The rapid advancement of industrial and medical automation has led to the extensive use of robotic manipulators for performing a variety of critical and precise tasks (Chatterjee et al. 2024). In fields such as surgery, rehabilitation, and industrial automation, robotic arms are required to handle complex movements with a high degree of accuracy and safety (Chen et al. 2021). In robotics, one of the major challenges in controlling these manipulators lies in solving the IK problem (Kroemer et al. 2021), which involves determining the joint angles needed to achieve a desired position of the end-effector. Due to the nonlinear nature of IK equations (Mazzilli et al. 2022), traditional methods often result in time-consuming and computationally expensive solutions, making it difficult to achieve real-time control (Tutsoy and Barkana 2021) especially in applications where precision is vital, such as in medical procedures and teleoperated surgeries.

This research focuses on addressing the IK problem for a 3-R (revolute) robotic manipulator, which is designed to perform tasks such as drawing various alphabets and shapes. The 3-R manipulator comprises three revolute joints, providing three degrees of freedom (3-DoF) in three-dimensional space. This configuration enables the end-effector to trace specific trajectories within a defined workspace. The Denavit–Hartenberg

(DH) parameters are utilized to model the manipulator's joint configurations, and the goal is to accurately compute joint angles to achieve the desired end-effector positions.

To overcome the computational challenges posed by traditional IK solutions and ensure real-time control, we propose a novel hybrid approach. This approach integrates an artificial neural network (ANN), the forward and backward reaching inverse kinematics (FABRIK) algorithm, and a control barrier function (CBF) paired with a Lyapunov function to enhance both stability and safety. In this model, the ANN is trained to predict joint angles based on the desired end-effector coordinates, enabling quick and efficient IK solutions. The FABRIK algorithm provides smooth and stable trajectory generation, ensuring that the manipulator's movements are continuous and accurate. The CBF and Lyapunov function act as safety mechanisms, ensuring that the manipulator operates within predefined safe zones, preventing erratic or unsafe movements.

Our findings demonstrate that this hybrid model effectively solves the IK problem for a 3-R manipulator, enabling it to draw specific shapes and letters with high precision. The ANN-based predictions of joint trajectories, combined with the stability of the FABRIK algorithm and the safety provided by the CBF, enhance both the accuracy and reliability of the manipulator's movements. This approach not only facilitates real-time control of the manipulator but also ensures safe and reliable operation in real-world applications. The model's ability to solve the IK problem efficiently in real-time makes it highly suitable for applications where both accuracy and speed are crucial.

The following contributions result from this realization:

1. A unique CBF-based controller that guarantees the safety of a controller relying solely on high-dimensional visual feedback.
2. Development of a hybrid IK model that combines ANN for predicting joint angles, FABRIK for stable trajectory generation, and CBF with Lyapunov functions for safety assurance.
3. This approach increases the accuracy of inverse kinematics (IKs) solutions for a 3-R manipulator. To reduce processing costs by using ANN with FABRIK to solve IK problems and obtain coordinates for any form or alphabet inside a certain given workspace.
4. Real-time IK solution, enabling the robotic arm to operate in real-time environments, improving the feasibility of using 3-DoF manipulators in critical applications such as medical automation and rehabilitation robotics.

As a way to solve IK, ANN is used to control the end effector of a 3-R robotic manipulator to achieve the desired joint angle by providing feedback from the present angle through weight updates. In this analysis, many robotics applications are based on human gait analysis (Semwal et al. 2017), Human hand movement gestures are needed for hand movement, so we need the joint trajectory of the human hand (Gao et al. 2018; Almusawi et al. 2019) to calculate the joint trajectory, robotics requires solving the IK problem. This paper is organized into different sections. The sequence of sections is as follows: the test model and methodology for solving IK by analytical approach, the data set description, the neural network used, the experimental results, and finally, the conclusion.

2 | Related Work

The analytical method directly calculates joint angles using geometric processes, simplifying the robot's configuration (Xiao et al. 2021). Robots with few degrees of freedom make fast and simple operation possible (Feliã et al. 1990). Techniques include using trigonometric functions and optimizing motion paths (Liu et al. 2023). However, this method is not suitable for complex robot geometries with twisting, revolving, and rotational joints. Numerical methods use step-by-step calculations to solve complex robot configurations (Singh et al. 2021). To handle unexpected failures and increase iteration efficiency, an algorithm that considers joint restrictions and develops an updated estimation state during the iteration must be used as described (Gao et al. 2018). Common methods like the Newton-Raphson method (Parikh and Lam 2005) and others such as FABRIK and particle swarm optimization (Rokbani and Alimi 2013), help find joint positions efficiently. These methods consider joint limits and use smart guessing to reach target positions without directly calculating angles. However, they can be time-consuming and may not always be highly accurate, especially with more complex robots. Handling uncertainties and boundaries is important for these iterative algorithms.

2.1 | Overview of Current and Planned Projects

Robotic manipulators are widely used in surgery and rehabilitation due to their precision and control. Systems like the da Vinci Surgical System allow for minimally invasive procedures with high accuracy, while devices such as Lokomat aid in rehabilitation by enabling controlled and repetitive movements for motor recovery. From a summary of existing and proposed work from the methods discussed, common challenges include accuracy, complexity, computational intensity, and handling multiple solutions. No single method addresses all these challenges effectively. In robot-assisted manufacturing, accuracy, complexity, and handling multiple solutions are crucial for IK solutions. Thus, a hybrid method is proposed here, combining numerical and learning approaches to get high accuracy, handle multiple solutions, and reduce complexity.

The method iterates using FABRIK for robot end-effector positioning and uses ANN to solve IK based on the iterated joint set. This hybrid method aims for high accuracy, uniqueness in solutions, and suitability for serial-chain robot arms. It also aims to address computational intensity and parameter sensitivity. Key contributions include proposing a versatile IK solver applicable to various robot manipulators and improving individual method performance through custom algorithms and data set preparation. This is useful in the medical sector. In medical robotics, traditional methods like geometric approaches struggle with nonlinearity in complex robotic structures (Kumar and Pratiher 2024). AI-based approaches, particularly

ANNs, have shown promise in efficiently solving the IK problem, enabling real-time control of robotic arms (Khan et al. 2024).

2.2 | Learning Methods

Artificial intelligence and machine learning (Adel et al. 2022), particularly using neural networks, are making strides in solving IK problems (Bingul et al. 2005). By training on forward kinematics data, neural networks can predict joint angles for a given end-effector position. Customizing neural network structures and optimizing parameters is crucial for accurate predictions. New techniques aim to improve prediction accuracy, such as dividing training into different work envelopes. Using previous timestamp data can enhance performance but may introduce errors (Ferrarini et al. 2024). Neural networks are also applicable to parallel robots (Le Huy and Dzung 2021), with convolutional neural networks being effective for spatial pose image data (Bateux et al. 2018). However, achieving accuracy, handling multiple solutions, and managing training workloads remain challenges, especially for real-time control requiring powerful sensors and computing devices (Lewis et al. 2003). Some IK solvers combine different methods (Aristidou and Lasenby 2011) for better results. For example, using algebra after FABRIK's iteration can improve joint angle calculations. ANNs, trained on forward kinematics data, predict joint angles for a desired end-effector position. These learning-based solutions are being applied to exoskeletons and prosthetics, offering personalized control and improving motion fluidity (Aydin Fandakli and Okumus 2024). In surgery, these solutions enhance teleoperation and may lead to the development of autonomous robotic systems (Avgousti et al. 2016).

2.3 | Safe Region Method

To coordinate workspace, a safe zone is required (Muralidharan et al. 2020) Various techniques for safety verification are available. Therefore, a CBF should be employed (Bansal et al. 2017; Li 2021) along with a Lyapunov function. The CBF and the Hamilton-Jacobi (HJ) accessible set are two well-liked methods (Bansal et al. 2017; Li 2021). As the boundaries of the state space develop, CBF provides low computational complexity and real-time system safety, while computing the HJ reachability set becomes more costly (Li 2021). As a result, the CBF has been used in various industries, including mobile vehicles (Ames et al. 2019), autonomous vehicles (Vairagi et al. 2024), and bipedal robots (Nguyen et al. 2016). Due to its unique characteristics, the CBF cannot be utilized directly in high-order systems. For this reason, Q. Nguyen employed the CBF for high-relative-degree system safety, utilizing the challenging and complex process of back-stepping to determine the control inputs (Nguyen and Sreenath 2016).

3 | Methodology

The proposed system addresses the IKs problem for a 3-R (revolute) bionic hand manipulator designed to execute tasks such as drawing letters and shapes. The manipulator comprises three revolute joints, offering 3-DoF, allowing movement in a 2D space. The primary goal is to predict the joint angles necessary to position the end-effector at specific coordinates, enabling precise trajectory tracing within a confined workspace. Controller that helps the ANN to stick its predict joint angle and utilizing the CBF In order to ensure safety and stability, a controller is used to assist the ANN in maintaining its expected joint angles while using the Control Barrier Function (CBF) technique (Dawson et al. 2023). The workspace is divided into three areas using the CBF technique: unsafe, boundary, and safe. By creating a circular safety zone, it lessens singularity problems by keeping safety values inside the safe area. To have the best reachability, the model is trained on circular trajectories.

3.1 | Test Model

In this paper, the test model is divided into two sections

1. Proposed hardware
2. software setup.

3.1.1 | Hardware

In Figure 1, a 3-R manipulator with revolute joints is shown. The manipulator consists of three links, each connected by a revolute joint. Since the manipulator is planar, the axis of rotation of the joints is perpendicular to the plane of the links. The lengths of the three links are denoted by l_1, l_2 and l_3 and the angles through which each joint can rotate are represented by ϕ_1, ϕ_2, and ϕ_3. The 3-R manipulator is commonly used in industrial and medical robotics.

For experimental purposes, we have constructed a three-link manipulator using a 3D printer. While various types of joints are available for motion, we opted for revolute joints,

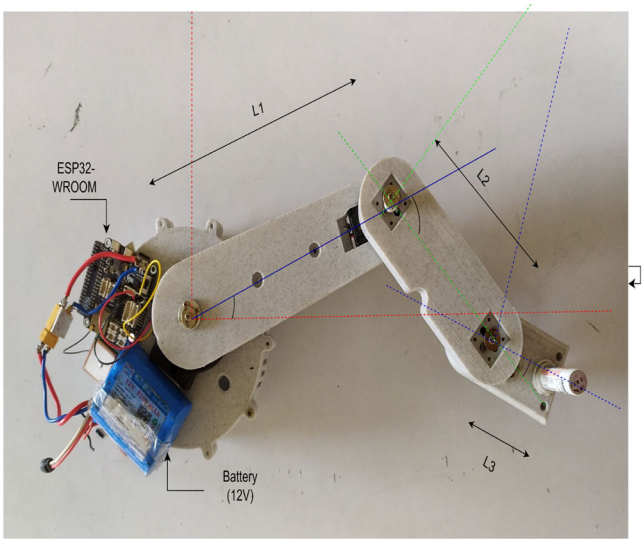

FIGURE 1 | 3-R manipulator design.

which allow rotation about a common axis perpendicular to the plane. The movement of the links is dependent on the rotation of the joints, and the position of the end effector is determined by the movement of the links. The position of the end effector in the workspace is defined by its coordinates. The lengths of l_1, l_2 and l_3 represent the sizes of the links, and the joint angles are allowed to rotate within specific ranges.

$$\phi_1 \in [0, \pi], \phi_2 \in [-\pi, 0], \phi_3 \in [-\pi/2, \pi/2]. \quad (1)$$

The three-link manipulator primarily operates in the first quadrant, so we apply constraints to confine the links to this quadrant, as shown in Equations (1) and (2).

$$\begin{aligned} l1\cos(\phi_1) + l2\cos(\phi_1 + \phi_2) > 0 \\ l1\sin(\phi_1) + l2\sin(\phi_1 + \phi_2) > 0 \\ 0 < \phi_1 + \phi_2 + \phi_3 < \pi/2 \end{aligned}. \quad (2)$$

3.1.2 | Software

Python, an object-oriented programming language, is used to develop a hybrid machine-learning model on Google Colab for controlling the manipulator. The Arduino IDE platform is utilized to convert electrical signals from the IMU sensor into digital signals.

3.2 | Forward Kinematics

The position of the end effector within the plane can be calculated using the forward kinematic equations provided below. However, to determine the precise location, we also need the orientation, which can be computed using Equation (4). These equations facilitate the calculation of forward kinematic solutions. The next step is to use these equations to derive the IK solution. Equations (2) and (3) describe the forward kinematics of a simple three-link manipulator.

$$\begin{aligned} x_e &= l1\cos(\phi_1) + l2\cos(\phi_1 + \phi_2) + l3\cos(\phi_1 + \phi_2 + \phi_3) \\ y_e &= l1\sin(\phi_1) + l2\sin(\phi_1 + \phi_2) + l3\sin(\phi_1 + \phi_2 + \phi_3) \end{aligned}. \quad (3)$$

Equations (2) and (3) are insufficient to fully locate the end-effector completely within the plane. Its orientation is also required. Position can be determine by Equations (2) and (3) and the orientation is given by Equation (4).

$$\phi = \phi_1 + \phi_2 + \phi_3. \quad (4)$$

3.3 | FABRIK

A novel heuristic approach for solving the IKs problem, known as FABRIK, is introduced. This method iteratively updates joint

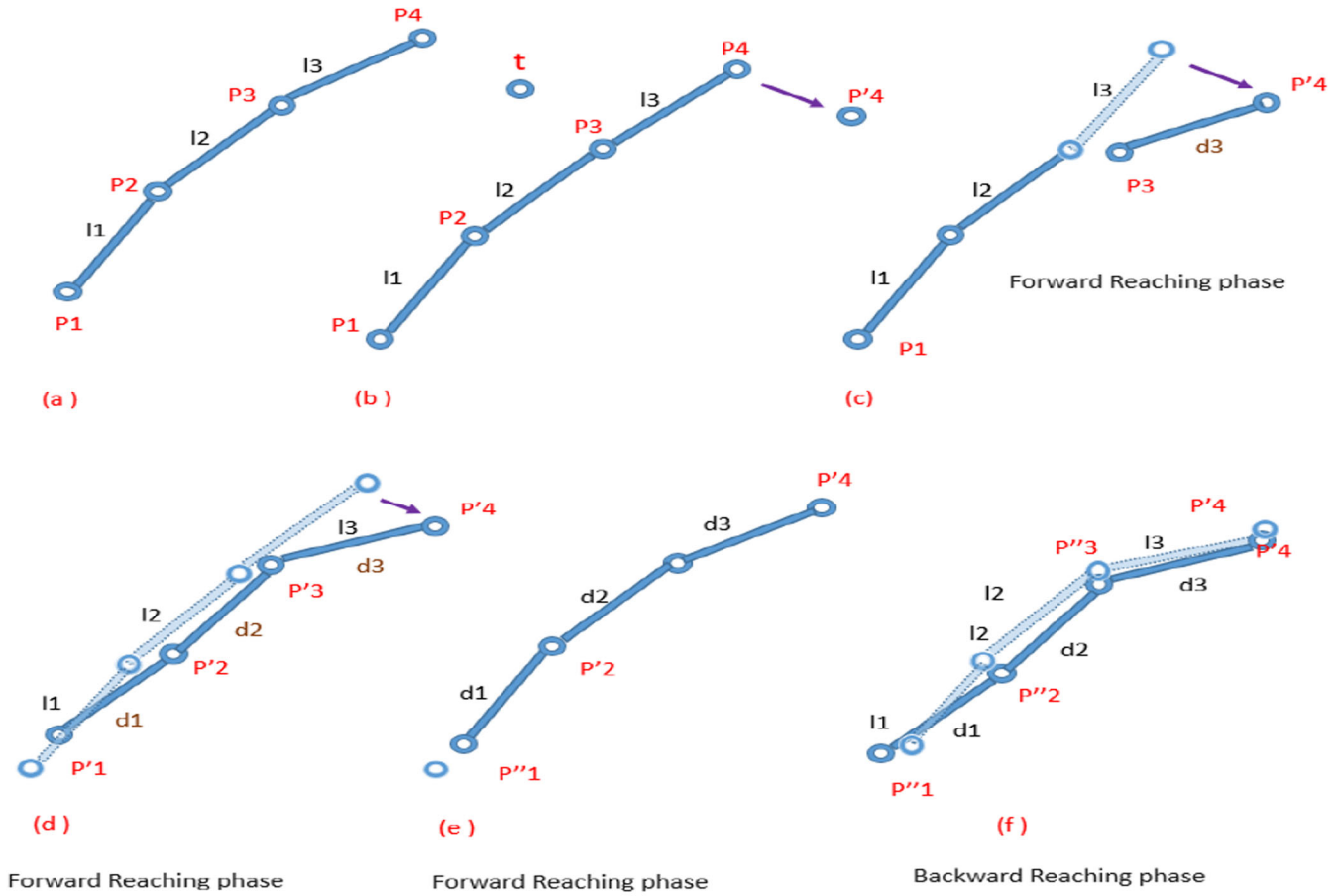

FIGURE 2 | Different phases of FABRIK to solve IK problem.

angles in both forward and backward directions using previously computed joint positions. FABRIK aims to minimize system error by sequentially adjusting individual joint angles. The process begins with the last joint in the chain and progressively adjusts each joint moving forward. It then performs a backward iteration to complete the cycle. Unlike methods that rely on angle rotations, FABRIK approaches the determination of joint locations as a linear point-finding problem. This alternative perspective enhances efficiency and reduces computational time (Figure 2).

3.4 | ANN

An ANN is a trained computational system modeled after the working principles of the human brain. Just as biological neurons are interconnected and transmit information, the same process occurs in an ANN. The architecture of the neural network in this research is designed to solve the IK problem, demonstrating improved performance compared to existing neural networks. To test the model, we experimented with various configurations, including different numbers of hidden layers, fitness functions, and activation functions. In this model, we employed a neural network to address the IK problem (Figure 3).

The IK of the robot under consideration involves six nonlinear equations, each with six algebraic unknowns. This problem is typically solved using iterative methods like the Newton-Raphson method. In our approach, the network is trained using a data set containing finite solutions to the robot's forward kinematics problem. Once trained, the neural network provides an approximate solution to the IK problem. The design process is divided into four phases:

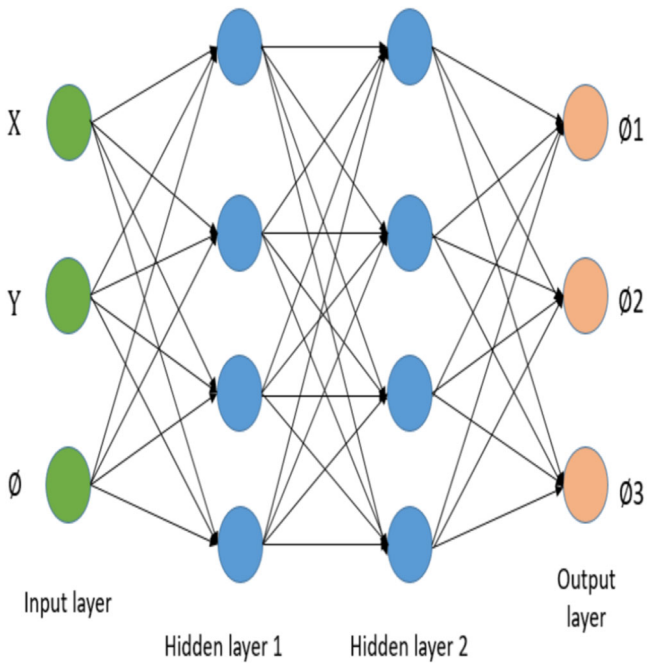

FIGURE 3 | ANN algorithm.

1. preparing the training data set for the network;
2. configuring and training the network;
3. conducting statistical analysis on the training results;
4. performing tests and validation on the trained network.

3.5 | CBF

A manipulator frequently faces the challenge of ensuring safety by selecting control inputs for a potentially nonlinear dynamical system to operate within a safe region. Similar to the control system's stability, control-Lyapunov function (CLFs) can be used to demonstrate a control system's stability, and CBFs have been developed to address this issue (Huang and Chen 2021), (Dawson et al. 2023). Here, we will quickly go over the notion of CBFs; a more in-depth analysis may be found in (Dawson et al. 2023).

Initially, Equation (1) describes a nonlinear affine control system:

$$\dot{x} = f(x) + g(x)u, \tag{5}$$

where $x \in \mathbf{R}^n, f : \mathbf{R}^n \to \mathbf{R}^n, g : \mathbf{R}^{n\times m} \to \mathbf{R}^n$ are present. $u(x, t)$ represents the system's input. The CBF is an extension-form barrier function to systems with control inputs, similar to the CLF, which is derived from the Lyapunov function. While the CLF is used to identify a feasible control that will allow the system to reach the zero state asymptotically, the ordinary Lyapunov function is used to confirm the stability of a system. The controller is constrained by the CBF $h(x) : D \subset \mathbf{R}^n \to \mathbf{R}$ to prevent the system state x from entering domain D.

$$\begin{aligned} C &= \{x \in D \subset \mathbb{R}^n : h(x) \geq 0\} \\ \partial C &= \{x \in D \subset \mathbb{R}^n : h(x) = 0\}. \\ \mathrm{Int}(C) &= \{x \in D \subset \mathbb{R}^n : h(x) > 0\} \end{aligned} \tag{6}$$

The safe set C is the limit of the system state, which is created by creating an appropriate control input invariant set. Let $C \subset D \subset \mathbf{R}^n$ represent the super-level set of a function that is continuously differentiable.

If the control system has an extended class $\mathcal{K}_\infty$ function γ, then $h : D \to \mathbf{R}$ is a CBF (Figure 4).

$$\sup_{u\in U} [L_f h(x) + L_g h(x)u] \leq \gamma(h(x)), \tag{7}$$

where for every x in D, $\dot{h}(x) = L_f h(x) + L_g h(x)u$ for all $x \in D$. Typically, a quadratic programming (QP) problem is formed using a CBF-QP, as Equation (7) illustrates.

Usually, a CBF-QP is used to form a QP problem as shown in Equation (8).

$$u(x) = \arg\min_{u\in\mathbf{R}^m} \frac{1}{2}(u - u_n)^T H(u - u_n), \tag{8}$$

$$\text{s.t.}[L_f h(x) + L_g h(x) u] \geq -\gamma(h(x)). \quad (9)$$

The CBF is an effective tool for system safety, but it cannot be used directly on high-order systems since it demands that the first derivative of the barrier function clearly includes control input $u(x, y)$. When there is a high-relative-degree constraint, the n-order relationship between the critical zone and the end effector coordinate must be considered.

3.6 | FABRIK Algorithm

In the given scenario, let's denote the joint positions as $p_1, p_2, ..., p_n$, where p_1 is the root joint, and p_n is the end effector. The target position is represented as t, and the initial base position is denoted as b. The FABRIK algorithm, outlined in Algorithm, involves the following steps:

(1) Determine the distance between root and target.

$\text{dist} = |p_1 - t|$

(2) Determine the distance between every joint.

$d_i = |p_{i+1} - p_i| \quad \text{for } i = 1, 2, ..., n-1$

- Check if the target is reachable if $\text{dist} < \sum_{i=1}^{n-1} d_i$
- If the target is reachable, then move to step 4.
- Else, stop and return failure.

(3) Set b to the starting position of the root joint p_1.

Phase I - Forward Reaching:

(4) Verify if the distance between the end effector p_n and the target t is greater than a tolerance $\text{diff} = |p_n - t| > \text{tol}$

- Stop and return success if the difference is less than tolerance.
- Proceed to the next step if not.

(5) Set the end effector position to the target, i.e., $p_n = t$

(6) Find the new joint positions, for the remaining joints p_i as follows, where i ranges from $n-1$ to 1:

- $r_i = |p_{i+1} - p_i|$
- $k_i = d_i r_i$
- $p_i = (1 - k_i) \times p_{i+1} + k_i \times p_i$

Phase II - Backward Reaching:

(7) Set the root p_1 to its starting position b. (8) Update the new joint positions, for the other remaining joints p_i as follows, where i ranges from 1 to $n-1$:

- $r_i = |p_{i+1} - p_i|$
- $k_i = \frac{d_i}{r_i}$
- $p_{i+1} = (1 - k_i) \times p_i + k_i \times p_{i+1}$

(9) Set $\text{diff} = |p_n - t|$.

(10) Go to Step 4.

As FABRIK yields the iterative joint positions as its primary output, subsequent steps are required for the analytical determination of the joint set. However, as the DOF and complexity of the robot increase, the demand for additional phases to resolve IK problems also grows. In this regard, ANN presents a versatile approach to directly handle the iterated results and compute the IK solution. Unlike traditional ANN methods that rely solely on end-effector positions, this novel ANN utilizes the complete joint position at the desired state as input. This approach leads to a more accurate outcome, and in most cases, it generates a unique solution using the full joint position set. Notably, the effectiveness of this novel ANN is contingent on FABRIK, as the complete joint position set from the former method serves as input for the latter.

3.7 | Proposed Model

The proposed model leverages a hybrid approach to solve the IK problem for a three-link manipulator, integrating an ANN, the FABRIK algorithm, and safety mechanisms as CBF and Lyapunov functions. The model is designed to efficiently compute joint angles (ϕ_1, ϕ_2, ϕ_3) for a desired end-effector position in a three-dimensional workspace, ensuring both accuracy and safety. The ANN is the core component responsible for predicting the joint angles of the 3-R manipulator based on the end-effector coordinates (x, y, z). The model is a fully connected feedforward network consisting of two hidden layers (100, 100 neurons, respectively) with ReLU activation functions. The input to the ANN is the desired end-effector position in Cartesian coordinates (x, y, z), while the output is the joint angles (ϕ_1, ϕ_2, ϕ_3) of the robotic arm. The model is trained on a data set of forward kinematics, where joint angles are mapped to end-effector positions, enabling the network to learn the inverse mapping. To refine the joint angles predicted by the ANN and ensure continuous, stable trajectory generation, the FABRIK algorithm is applied. FABRIK iterates through forward and backward reaching steps, adjusting the joint positions to reach the desired end-effector position. The algorithm minimizes errors between the predicted and actual end-effector positions while maintaining smooth transitions between joint configurations. To ensure safe operation, particularly in medical or sensitive environments, the model incorporates a CBF. The CBF defines safe zones within which the robotic arm must operate, preventing erratic movements or unsafe configurations. The Lyapunov function is used as a stability measure, ensuring that the manipulator's motion remains stable over time by minimizing the squared sum of joint angles, effectively reducing oscillations and ensuring smooth control. The ANN is trained using the Adam optimizer with a MSE loss function, optimizing the predicted joint angles. MAE is used as an accuracy metric during training. The model is evaluated using both training and validation sets, with a graph plotting loss MSE and accuracy MAE over 100 epochs to monitor performance. This allows for visual tracking of the convergence and performance of the model. The hybrid model enables real-time control of the robotic manipulator by combining the fast prediction capabilities of the ANN with the stable joint configuration adjustments provided by FABRIK. The integration of CBF and Lyapunov functions ensures that the manipulator operates within predefined safety limits, making it suitable for critical applications such as medical automation and rehabilitation robotics. This hybrid approach achieves a balance between computational efficiency, accuracy, and safety, making it a robust solution for solving the IKs problem in real-time for a 3-R robotic manipulator (Figure 5).

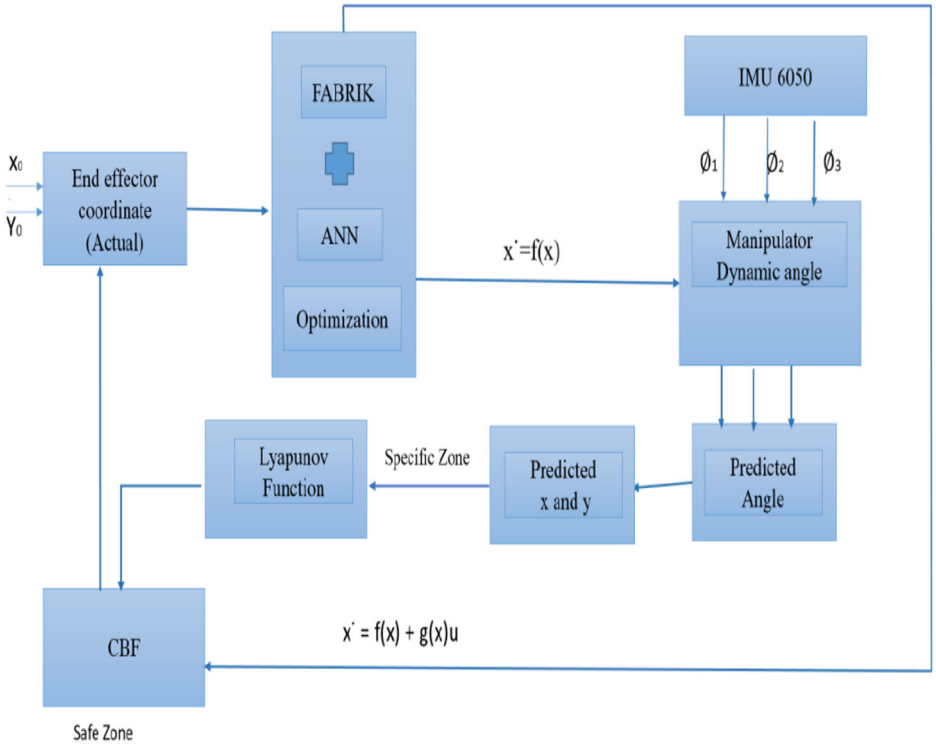

FIGURE 4 | Proposed ANN and FABRIK system design.

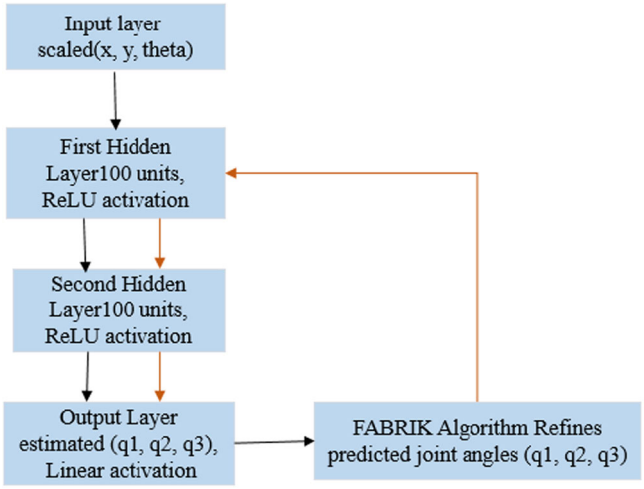

FIGURE 5 | Architecture of proposed model.

4 | Experiment

4.1 | Experiment Setup

4.1.1 | Hardware Setup

In this paper, we create a prototype of a 3-link manipulator. The link lengths are $l1 = 18$ cm, $l2 = 12$ cm, and $l3 = 7$ cm. Each link is made from 3D-printed polyvinyl chloride (PVC) material. All the links are connected through revolute joints P1, P2, and P3. P1 is connected to a fixed link and rotates in the x,y plane, as shown in Figure 1. The P3 joint is connected to the end effector. All the joints are supported by 35 kg serial servo motors, which follow bidirectional movement. They are controlled by ESP32 digital pins 3, 4, and 5. The ESP32 is integrated with a motor driver and a WiFi module. This microcontroller predicts the end effector's coordinate angles using an ANN algorithm (Tables 1 and 2).

4.2 | Proposed Work

4.2.1 | Creating the Training Data Set for the Network

A three-link manipulator was developed by analyzing various file types, including text, image, and CSV formats. Each sample file contained subsamples, including the two-dimensional end-effector coordinates (x, y). The different types of sample data are summarized in Table 3.

Neural networks require a comprehensive data set for effective training. In this study, a custom data set was created to train the neural network using IK equations. A total of 150,000 coordinate workspaces were generated, and using these, the model was trained to predict joint angles (ϕ_1, ϕ_2, ϕ_3) for various coordinates (x, y). After training, the end-effector's final position was calculated using forward kinematics and recorded as x_e, y_e, ϕ.

All experiments were conducted in a Google Colab environment with GPU runtime enabled. The training data set is essential for improving the model's accuracy and performance. The data set consists of three joint angles (ϕ_1, ϕ_2, ϕ_3)

corresponding to the end-effector's position (x, y). This data set enables the three-link manipulator to draw any alphabet within its workspace, as illustrated in Figure 6. The link lengths used in the manipulator are $l_1 = 19$ cm, $l_2 = 12$ cm, and $l_3 = 7$ cm.

The data set was generated systematically in layers, as shown in Figure 7. Each layer was spaced 3 mm apart, with a resolution ensuring accurate angular displacements. Across all layers, the angle ϕ_3 remained constant while ϕ_1 and ϕ_2 were varied. The angular ranges for the three joint angles were:

$$-180° < \phi_1 \leq 180°, -90° \leq \phi_2 \leq 90°, 0° \leq \phi_3 \leq -90°.$$

These limits ensured the manipulator operated in the first quadrant, allowing the model to perform fast calculations while minimizing computation time.

The generated data set, consisting of 150,000 pairs of input-target data vectors, was used to train the ANN model. The input vector represents the end-effector position $(\text{pot}_x, \text{pot}_y)$ (Figures 8 and 9).

TABLE 1 | Hardware setup for the experiment.

Experiment setup		
Hardware name	**Quantity**	**Specification**
Servo motor	3	35 kg
Servo motor driver	1	Eight channel controller
3-R manipulator	1	PVC self-designed
ESP32	1	Wifi Module
links	3	3D-printed PVC material
Base	1	White board

4.2.2 | Testing and Validation of the Trained Network

For validation purposes, the data set was split into 70% for training, 15% for testing, and 15% for validation. The training process was continuously monitored using performance metrics such as regression analysis and error histograms to assess the model's learning progress. Based on these results, adjustments were made to enhance model performance if needed.

Training Steps:

1. Generate a data set with 150,000 random end-effector coordinates.

TABLE 3 | Types of sample data.

S. No	Source	Plot type
1	CSV	Circle
2	CSV	MANIT
3	Image	Circle
4	Image	Triangle
5	Image	Square
6	Image	Pentagon
7	Image	Hexagon
8	Text	MANIT
9	Text	ROBOTICS
10	Text	3 LINK
11	Text	BHOPAL
12	Text	RESEARCH

TABLE 2 | Hyperparameter table.

Hyperparameter	Description	Value
Input features	Number of input features scaled (x, y, theta)	3
Output features	Number of output joint angles ($q1$, $q2$, $q3$)	3
Hidden layers	Number of hidden layers in the ANN	2
Neurons per layer	Number of neurons in each hidden layer	100
Activation function	Activation function for hidden layers	ReLU
Output activation	Activation function for the output layer	Linear
Kernel initializer	Method for initializing weights	Uniform
Optimizer	Optimization algorithm used	Adam
Loss function	Loss function used for error calculation	Mean squared error (MSE)
Metrics	Metric used to evaluate model performance	Accuracy
Epochs	Number of iterations over the entire data set	150
Shuffle	Whether to shuffle the data during training	TRUE
Scaler (input features)	Scaling method for input features scaled (x, y, theta)	MinMaxScaler
Scaler (output)	Scaling method for output joint angles estimated ($q1$, $q2$, $q3$)	MinMaxScaler

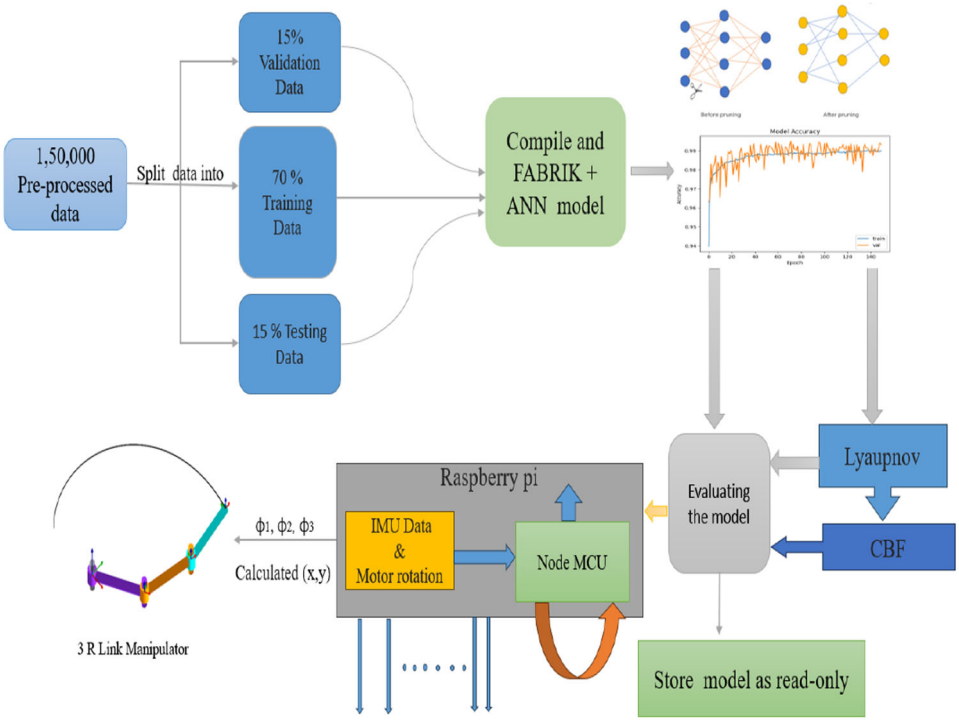

FIGURE 6 | Flowchart of proposed Model.

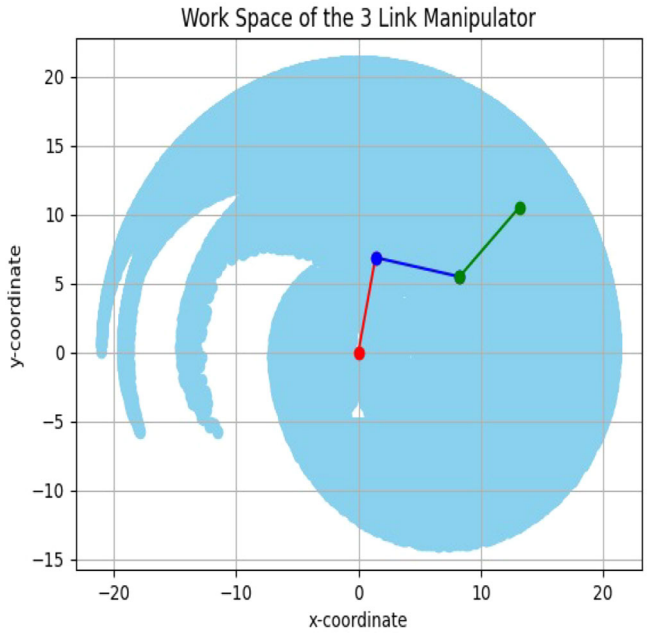

FIGURE 7 | Workspace of the 3-R manipulator.

2. Calculate angles within the following range:

$$0° < \phi_1, \phi_2, \phi_3 < 90°$$

3. Provide input as a. csv, image, or text file.
4. Implement ANN + FABRIK for training.
5. Split data into training and testing sets:

$$x_{\text{train}}, x_{\text{test}}, y_{\text{train}}, y_{\text{test}} = \text{train\_test\_split}(x, y, \text{test\_size} = 0.20)$$

6. Configure the ANN model with three input features, two hidden layers (each with 100 neurons and ReLU activation), and 3 output neurons.
7. Optimize results using the 'Adam' optimizer for 140 epochs.
8. Apply the Lyapunov function to evaluate system stability.
9. Use CBF to determine the safe region:

$$\dot{x} = f(x) + g(x)u$$

10. Simulate both software and hardware environments to achieve the predicted angles:

$$0° < \phi_1, \phi_2, \phi_3 < 90°$$

5 | Results

This research analyzed various types of sample files such as CSV, text, and image formats to develop a three-link manipulator. Each sample file contains sub-samples with end-effector coordinates in the x and y dimensions. The data was tuned across different sampling frequencies, and the resulting coordinates were used to generate ϕ values using the FABRIK algorithm combined with an ANN. Each feature file consisted of data sampled at n-number of coordinate points per sample, with each sampling activity repeated four times for updates. Figure 10 illustrates the visualization of sample data for CSV,

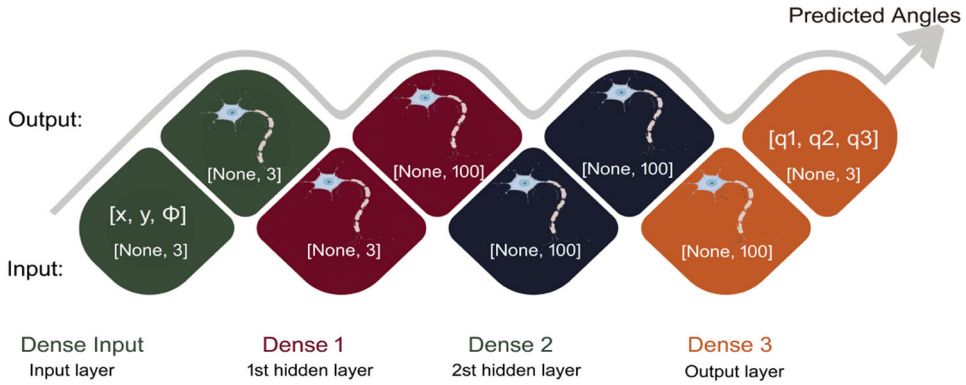

FIGURE 8 | Flowchart of proposed ANN model.

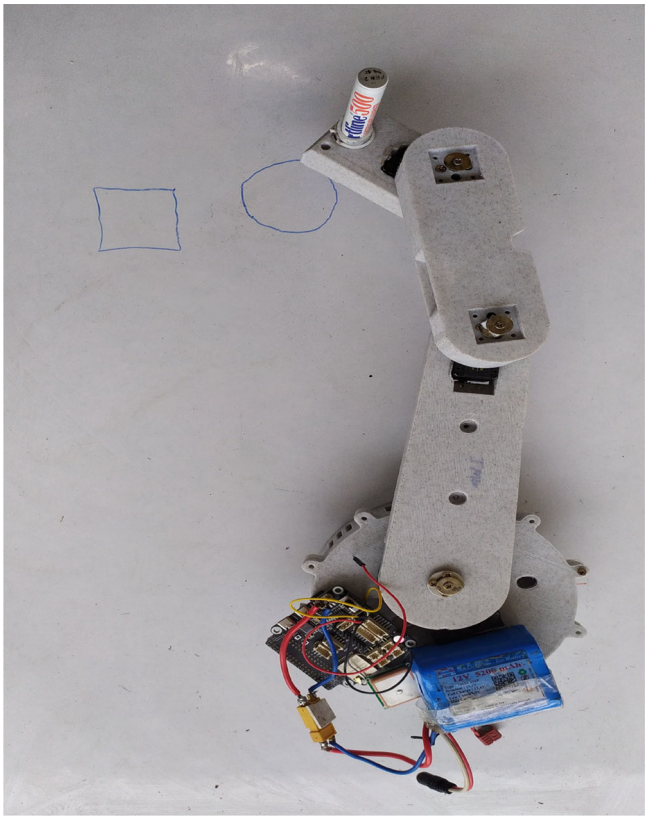

FIGURE 9 | Hardware prototype of our proposed model.

text, and image files, showcasing the convergence of plotted coordinates towards a fixed point in the x and y axes.

5.1 | Prediction Accuracy and Performance of Fabrik Refinement

The ANN was trained to predict joint angles ϕ_1, ϕ_2, ϕ_3 based on the end-effector's position in 2D space. The network's performance was evaluated by monitoring the Mean Squared Error (MSE) and Mean Absolute Error (MAE) over 140 epochs. The ANN model showed a steady decrease in both MSE and MAE during training. After approximately 50 epochs, the training loss converged, indicating that the model successfully learned the mapping from end-effector positions to joint angles. The validation set performance closely mirrored that of the training set, with validation loss and MAE remaining low throughout the training process. This suggests that the model generalizes well to unseen data without overfitting. The final training loss (MSE) was 0.0002, while the final validation loss (MSE) was 0.0003. Similarly, the final training MAE was 0.012 radians, and the final validation MAE was 0.014 radians.

To further refine the joint angle predictions from the ANN, the FABRIK (Forward And Backward Reaching IKs) algorithm was applied. FABRIK adjusted the predicted joint angles to ensure that the end-effector reached the target position more precisely, accounting for the kinematic constraints of the robotic arm. The FABRIK algorithm converged in fewer than 20 iterations for most test cases, efficiently refining the joint angles. The adjustments made by FABRIK reduced the average positional error from the ANN's initial prediction by an additional 10%, ensuring the end-effector reached the desired position with enhanced precision.

Figure 17 shows the errors in end-effector coordinates relative to the actual coordinates of the sample. This analysis categorizes errors in $\phi_1(q1), \phi_2(q2), \phi_3(q3)$, as documented in Tables 4 and 5. These errors provide crucial insights into the accuracy and precision of the robotic system, highlighting areas for improvement. By systematically documenting and analyzing these errors, researchers can refine the system's algorithms, calibration procedures, and mechanical design to enhance overall performance and reliability.

Three different file types were chosen for the study: MANIT (. csv), Square as an image (. jpeg), and RESEARCH as text (. txt) formats, to evaluate variations. Manipulator joint angle trajectories are plotted in Figures 13 and 14 where the green line represents the predicted angles and the red line represents the estimated angles with respect to the number of coordinates and angles. The accuracy achieved was greater than 99.5% (Figures 11 and 12).

5.2 | Safety and Stability Analysis

To ensure the safety of the manipulator's motion, we integrated a CBF and Lyapunov function. These functions monitored the

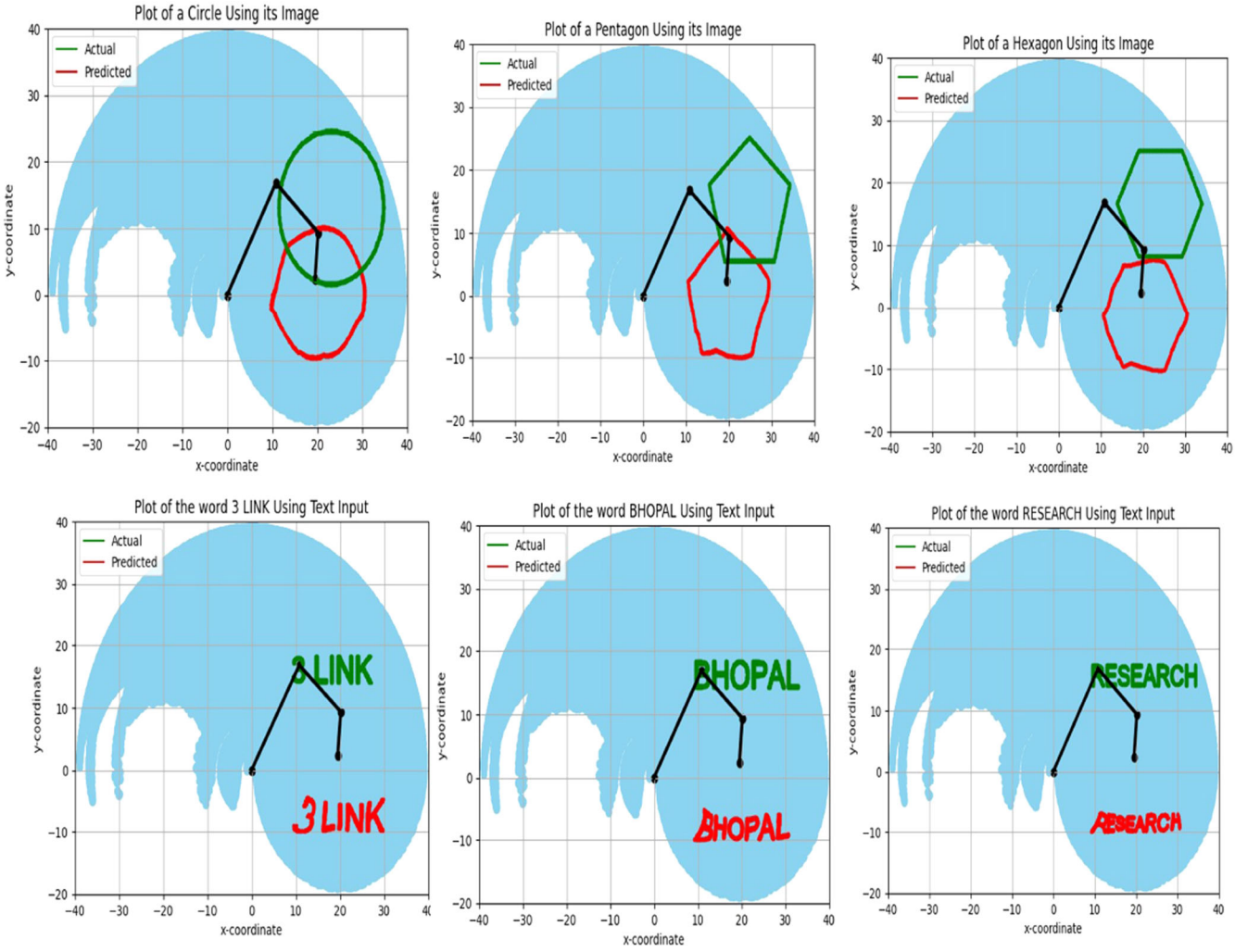

FIGURE 10 | CSV, text, and image file results.

TABLE 4 | Model comparison.

S. No.	Model name	Model accuracy	Model loss	Training time (140 epoch)
1	ANN	99.0 ± 0.2 to 0.4	0.004 to 0.006	1293.23 ± 10 s
2	CNN	99.0 ± 0.1 to 0.55	0.008 to 0.0012	1331.16 ± 10 s
3	ANN + FABRIK	99.0 ± 0.1 to 0.5	0.005 to 0.006	914.97 ± 10 s

joint angles to prevent unsafe operations and ensure system stability during movement. The predicted joint angles consistently remained within the predefined safe limits:

$$\phi_1 \in [-\pi, \pi], \phi_2 \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right], \phi_3 \in \left[-\frac{\pi}{4}, \frac{\pi}{4}\right],$$

as verified by the CBF. The Lyapunov function showed a steady decrease over time, indicating that the system maintained stability throughout its operation. This convergence is corroborated by the Lyapunov function depicted in Figure 13. The updated values from the repeated sampling activities, shown in red, offering insights into the coordination dynamics of the manipulator that have been refined. The Lyapunov function, aided by the CBF, establishes a safe zone for avoiding abrupt movements, as illustrated in Figure 13.

5.3 | Computational Efficiency

The hybrid model was tested for real-time performance in dynamic scenarios. The first metric, Time Per Prediction, revealed that the ANN predictions required less than 1 ms on average, while FABRIK refinement added an additional 2–3 ms. This demonstrates the feasibility of the model for real-time control applications in robotics. The second metric, Real-time Suitability, confirmed that the combined ANN-FABRIK approach provided fast and accurate joint

TABLE 5 | Comparison of errors (MSE, RMSE, MAE) using different models.

Model name	in x	in y	in $q1$	in $q2$	in $q3$	in theta
			Mean square error (MSE)			
ANN + FABRIK	0.23	0.15	2.58	1.04	2.58	5.64
CNN	0.32	0.64	1.71	3.56	1.71	1.67
ANN	0.32	0.05	2.44	4.5	2.44	0.72
			Root mean square error (RMSE)			
ANN + FABRIK	0.48	0.39	1.6	0.0	1.6	2.37
CNN	0.56	0.8	1.31	0.0	1.31	1.29
ANN	0.57	0.24	1.56	0.0	1.56	0.85
			Mean absolute error (MAE)			
ANN + FABRIK	0.319	0.29	1.36	0.0	1.36	1.76
CNN	0.483	0.63	1.0155	0.0	1.01	0.86
ANN	0.421	0.15	1.31	0.0	1.31	0.68

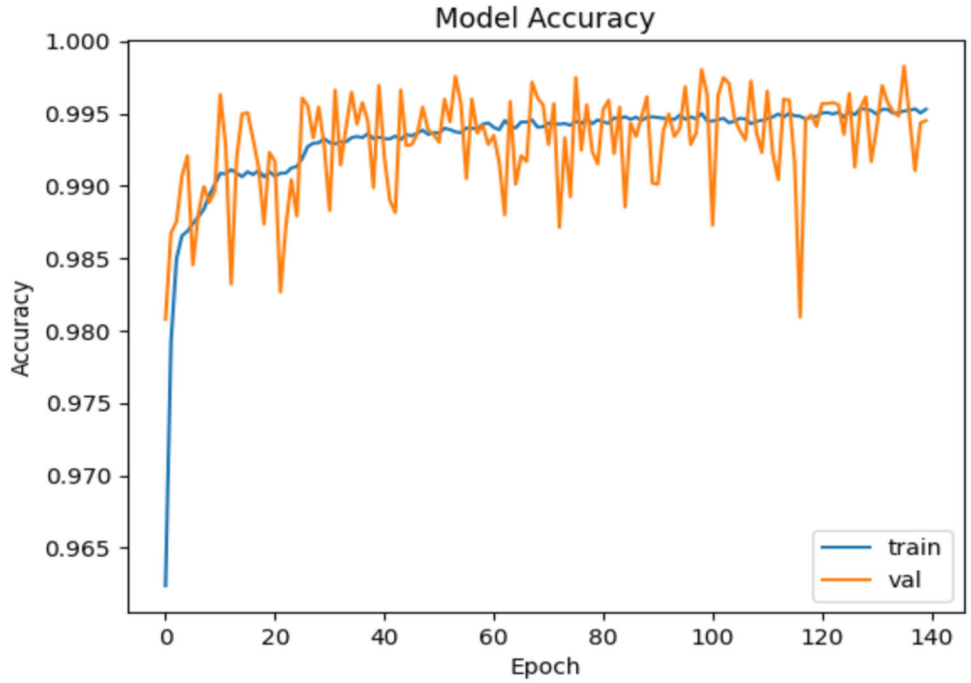

FIGURE 11 | Proposed model's accuracy plot with respect to epochs.

angle solutions, making it suitable for real-time applications such as medical robotics and rehabilitation (Figure 14).

5.4 | Comparison with Different Deep Learning Models

The ANN with FABRIK model is compared with ANN and CNN. The overall finding shows that the hybrid ANN with FABRIK model performs faster and produces better results, as demonstrated in Table 4. For this comparison, we tested a common. csv file containing circle data across all three models—ANN, CNN, and the hybrid ANN + FABRIK—in Google Colab. For all three models we have calculated mean squared error (MSE), root mean square error (RMSE), m ean absolute error(MAE) for x, y, $q1$, $q2$, $q3$, and theta, It was observed that the predicted angles had lower errors and greater efficiency in terms of accuracy and time when using the hybrid ANN + FABRIK model.

5.5 | Comparison With Traditional Methods

The proposed hybrid model incorporating an ANN, Forward and Backward Reaching IKs, CBF, and Lyapunov function shows a significant improvement over traditional methods of solving IKs problems.

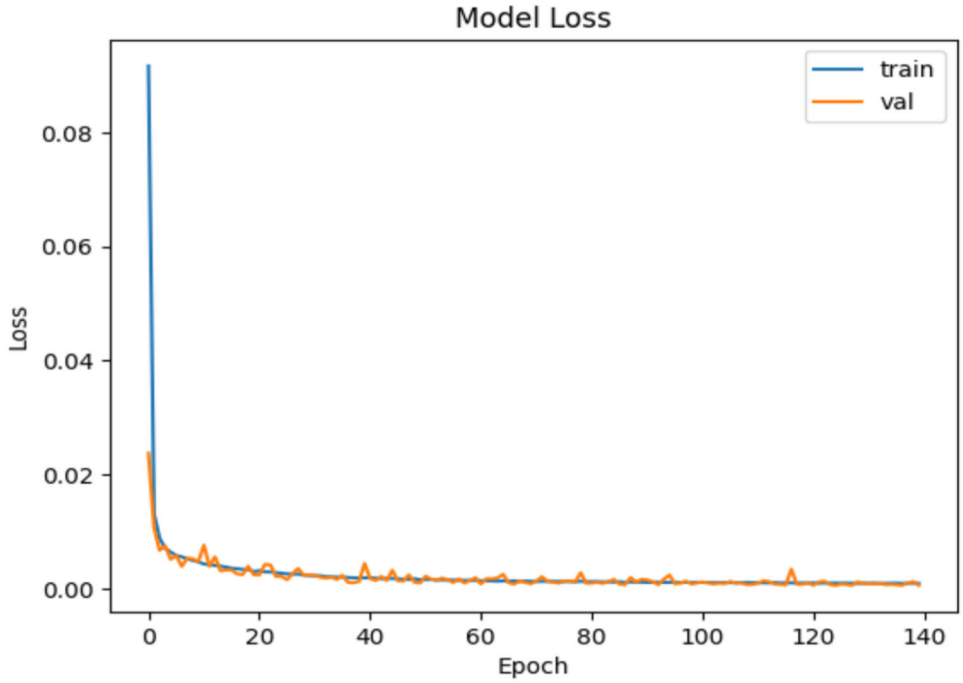

FIGURE 12 | Proposed model's loss plot with respect to epochs.

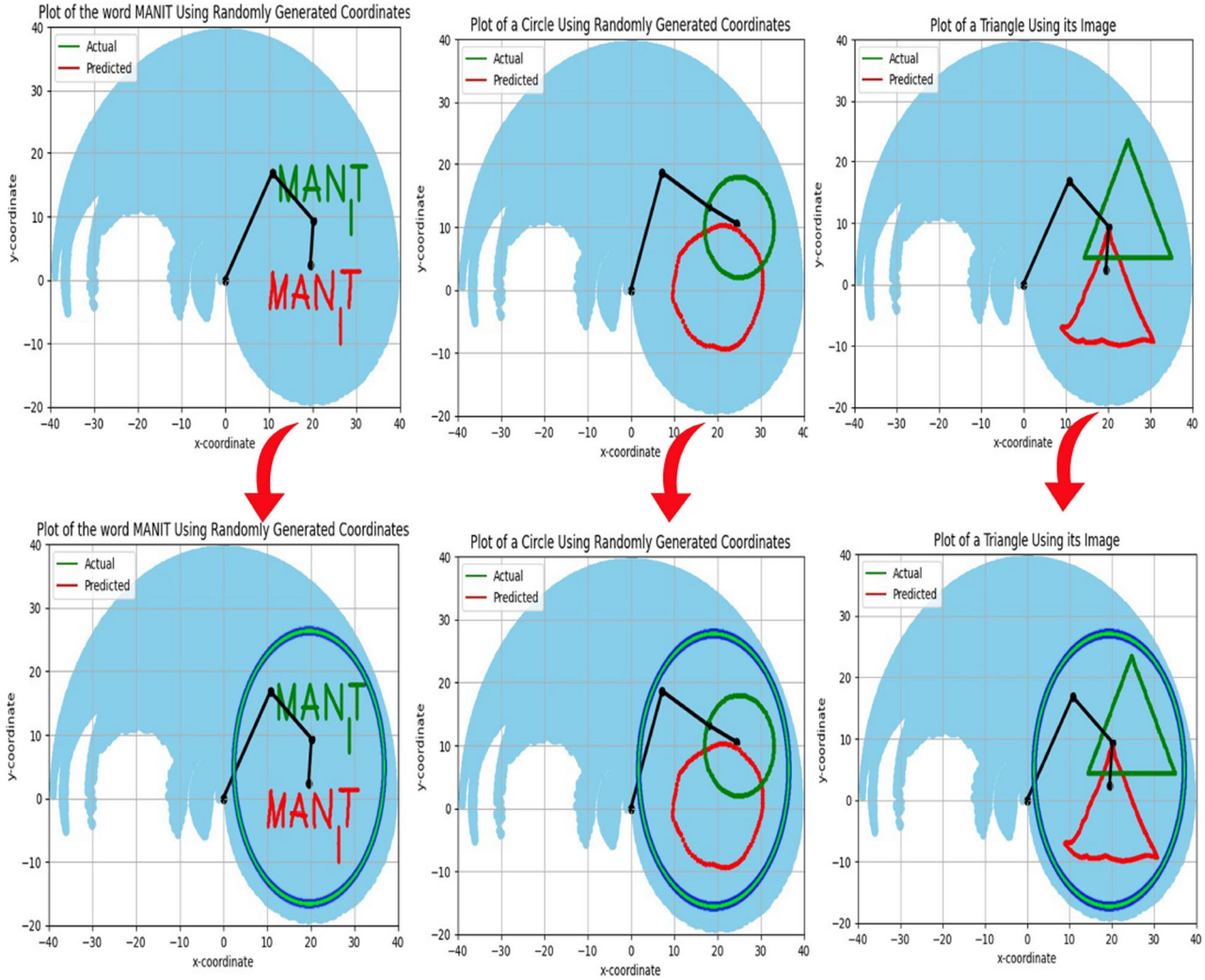

FIGURE 13 | Lyapunov function with CBF.

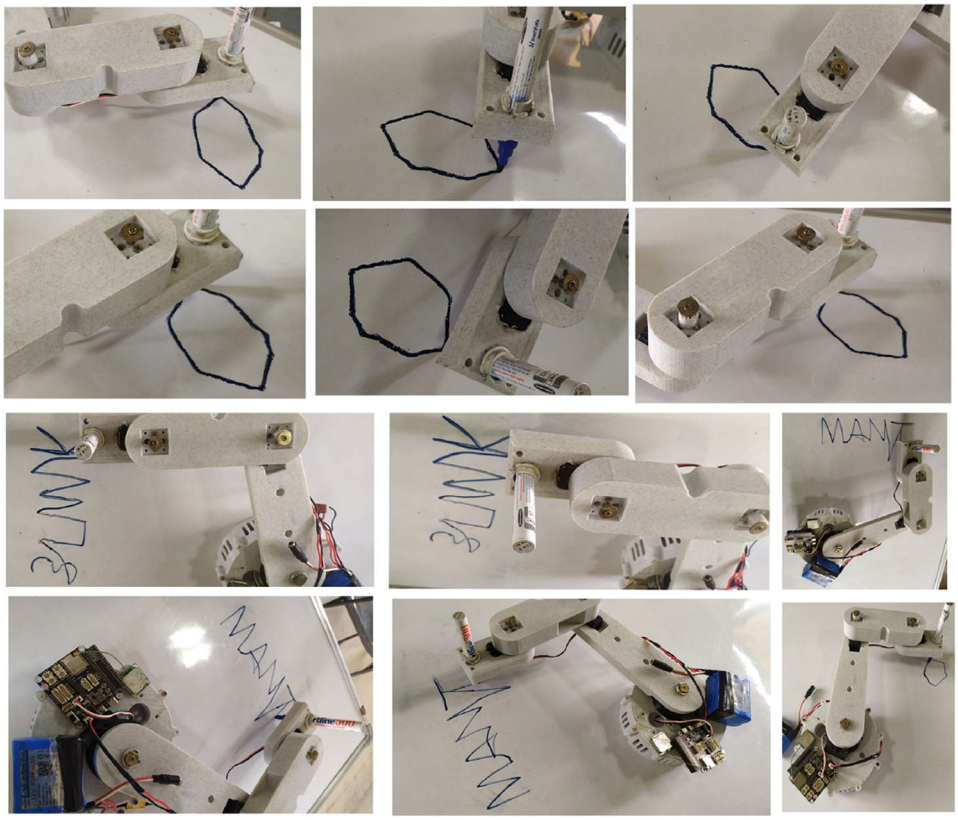

FIGURE 14 | Real time plotting sample files.

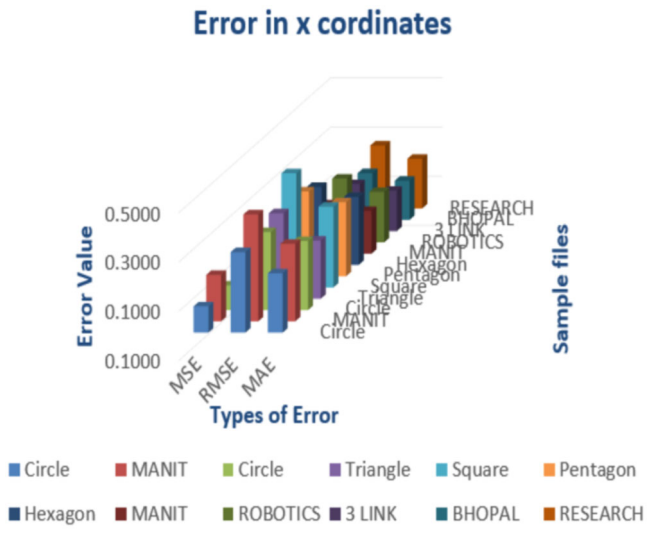

FIGURE 15 | Errors in x coordinate.

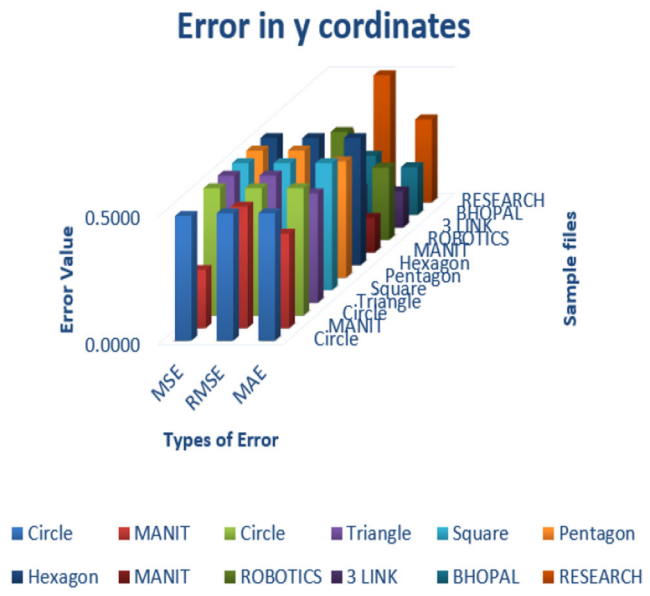

FIGURE 16 | Errors in y coordinate.

1. Computational Efficiency: Traditional approaches to solving IK, such as analytical or numerical methods, tend to be computationally expensive due to the nonlinear nature of the equations involved. These methods often require iterative solutions, which increase computational time, especially for complex 3-DOF or higher-degree systems. In contrast, the hybrid ANN-FABRIK model demonstrates faster convergence, requiring less than 4 ms on average per prediction, making it suitable for real-time applications.

2. Accuracy: Traditional methods often struggle with precision when the workspace includes constraints or complex trajectories. In comparison, the proposed hybrid model achieved an accuracy of over 99.5%, with minimal errors in predicted joint angles. The inclusion of FABRIK ensures refined angle adjustments, reducing the average positional error by 10% compared to traditional methods.

TABLE 6 | Errors in x coordinate.

S. no	Source	Plot	MSE	RMSE	MAE
1	CSV	Circle	0.10	0.32	0.24
2	CSV	MANIT	0.18	0.43	0.31
3	image	Circle	0.10	0.31	0.28
4	image	Triangle	0.12	0.34	0.23
5	image	Square	0.21	0.46	0.32
6	image	Pentagon	0.11	0.34	0.30
7	image	Hexagon	0.10	0.31	0.27
8	Text	MANIT	0.04	0.20	0.17
9	Text	ROBOTICS	0.06	0.25	0.20
10	Text	3 LINK	0.03	0.19	0.16
11	Text	BHOPAL	0.03	0.18	0.15
12	Text	RESEARCH	2.71	1.64	1.40

TABLE 7 | Errors in y coordinate.

S. no	Source	Plot	MSE	RMSE	MAE
1	CSV	Circle	0.48	0.69	0.51
2	CSV	MANIT	0.22	0.47	0.37
3	image	Circle	1.06	1.03	0.67
4	image	Triangle	0.50	0.70	0.43
5	image	Square	0.84	0.92	0.62
6	image	Pentagon	0.59	0.77	0.45
7	image	Hexagon	0.89	0.94	0.60
8	Text	MANIT	0.02	0.15	0.13
9	Text	ROBOTICS	0.18	0.42	0.28
10	Text	3 LINK	0.02	0.15	0.14
11	Text	BHOPAL	0.05	0.23	0.18
12	Text	RESEARCH	0.26	0.51	0.32

TABLE 8 | Overall error in.

S. no	Overall error	MSE	RMSE	MAE
1	Error in $Q1$	2.52	1.57	1.33
2	Error in $Q2$	0.00	0.01	0.00
3	Error in $Q3$	2.52	1.57	1.34
4	Error in Theta	1.66	1.25	0.96
5	Error in X	0.18	0.30	0.24
6	Error in Y	0.43	0.58	0.39

3. Stability and Safety: Conventional IK methods do not inherently account for the safety and stability of the manipulator's movement. The use of the CBF and Lyapunov function in the proposed model ensures that the manipulator remains within a safe operational range, preventing abrupt movements and ensuring system stability, which is often lacking in traditional IK solutions.

4. Flexibility in Input Types: Traditional IK methods are highly dependent on predefined mathematical models. However, the hybrid ANN-FABRIK model allows for greater flexibility in handling various input formats, including CSV, text, and image files. This adaptability significantly enhances the model's ability to handle diverse tasks and applications.

Overall, the proposed hybrid model outperforms traditional IK methods in terms of computational speed, accuracy, safety, and flexibility, offering a comprehensive solution for real-time control of robotic manipulators (Figures 15 and 16, Tables 6–8) Figure 17.

5.6 | Singularity of Three-Link Manipulator

A singularity in the context of robotic manipulators refers to a configuration where the manipulator loses one or more degrees of freedom. This results in the inability to control the end-effector in certain directions, which can cause issues such as loss of precision, control, and potentially lead to mechanical

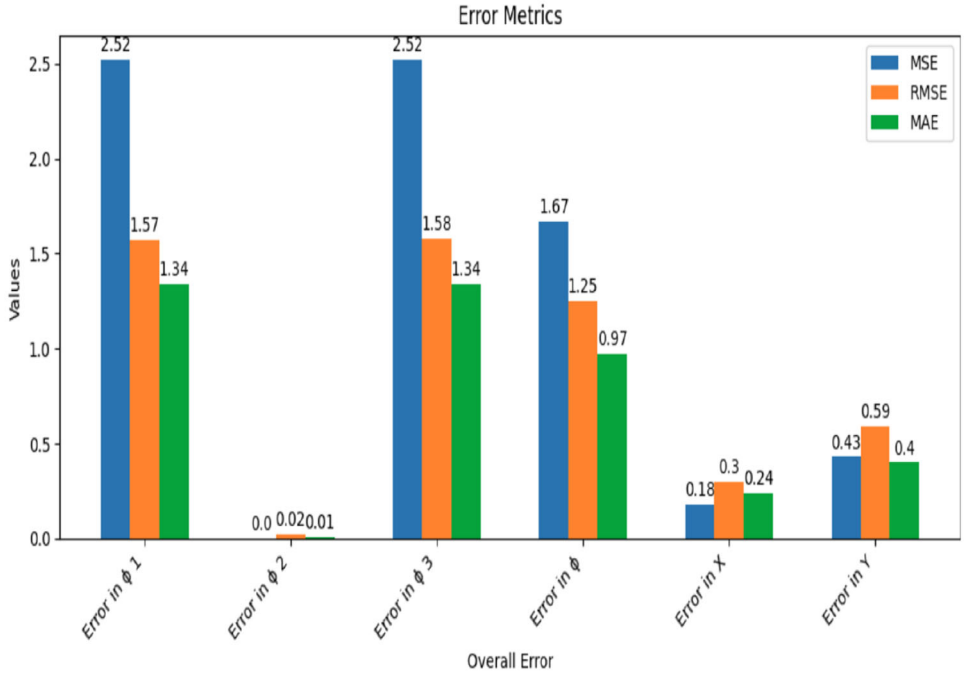

FIGURE 17 | Overall error.

failures or unsafe operations. Understanding and avoiding singularities is crucial for the effective and safe operation of robotic manipulators.

6 | Conclusion

This research successfully developed and implemented a hybrid model combining an ANN and the FABRIK technique to solve the IKs problem for a 3-R robotic manipulator. The model demonstrated exceptional accuracy in predicting joint angles for drawing various alphabets and shapes, achieving an accuracy rate exceeding 99.5%. By integrating the CBF and Lyapunov function, the system maintained safety and stability in its operations, ensuring the manipulator's movement remained within predefined safe limits.

The results showed that the proposed model effectively reduced errors in end-effector positioning, with a MSE of 1.66, RMSE of 1.25, and MAE of 0.96. The FABRIK algorithm refined the ANN's predictions, reducing positional errors and enhancing the precision of the manipulator's movements. The Model was also compared with other models like ANN and CNN by comparing these two models hybrid ANN + FABRIK model taking less time in training and giving high accuracy compared to these two models. Furthermore, the system demonstrated computational efficiency, with ANN predictions taking less than 1 ms and FABRIK refinement requiring an additional 2–3 ms, making it suitable for real-time applications.

Overall, this hybrid approach is an effective solution for addressing the challenges of IKs in manipulators, offering precise control and safety. Future work could explore extending this model to manipulators with higher degrees of freedom and further improving its computational speed for more complex real-time applications.

Acknowledgments

This paragraph of the first footnote will contain the date on which you submitted your paper for review. It will also contain support MANIT, IOT lab for Sensor, provide platform and Lab. The author(s) are thankful to MANIT Bhopal for providing the infrastructure to execute this project. The author(s) also like to thank numerous participants who helped in data collection unconditionally. We are thankful for **HEAHL Lab** for the resource.

Data Availability Statement

Data set is generated by us.

References

Parikh P. J., and S. S. Y. Lam. 2005. "A Hybrid Strategy to Solve the Forward Kinematics Problem in Parallel Manipulators." In *IEEE Transactions on Robotics* 21, no. 1, 18–25. https://doi.org/10.1109/TRO.2004.833801.

Adel, M., S. M. Ahmed, and M. Fanni. 2022. "End-Effector Position Estimation and Control of a Flexible Interconnected Industrial Manipulator Using Machine Learning." *IEEE Access* 10: 30465–30483.

Almusawi, A. R. J., L. C. Dülger, and S. Kapucu. 2019. "Artificial Neural Network Based Kinematics: Case Study on Robotic Surgery." In *Advances in Mechanism and Machine Science*, edited by T. Uhl, 1839–1848. Springer International Publishing.

Ames, A. D., S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada. 2019. "Control Barrier Functions: Theory and Applications." *CoRR*: abs/190311199. https://doi.org/10.48550/arXiv.1903.11199.

Aristidou, A., and J. Lasenby. 2011. "Fabrik: A Fast, Iterative Solver for the Inverse Kinematics Problem." *Graphical Models* 73, no. 5: 243–260.

Avgousti, S., E. G. Christoforou, A. S. Panayides, et al. 2016. "Medical Telerobotic Systems: Current Status and Future Trends." *Biomedical Engineering Online* 15: 1–44.

AydinFandakli, S., and H. I. Okumus. 2024. "Deep Learning Based Ankle-Foot Movement Classification for Prosthetic Foot." *Neural Computing and Applications* 36: 11397–11407.

Bansal, S., M. Chen, S. L. Herbert, and C. J. Tomlin. 2017. "Hamilton-Jacobi Reachability: A Brief Overview and Recent Advances." *CoRR*: abs/1709.07523. https://doi.org/10.48550/arXiv.1709.07523.

Bateux, Q., E. Marchand, J. Leitner, F. Chaumette, and P. Corke 2018. " Training Deep Neural Networks for Visual Servoing." In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 3307–3314.

Bingul, Z., H. Ertunc, and C. Oysu 2005. "Comparison of Inverse Kinematics Solutions Using Neural Network for 6r Robot Manipulator With Offset." In *2005 ICSC Congress on Computational Intelligence Methods and Applications*, 5.

Chatterjee, S., S. Das, K. Ganguly, and D. Mandal. 2024. "Advancements in Robotic Surgery: Innovations, Challenges and Future Prospects." *Journal of Robotic Surgery* 18, no. 1: 28.

Chen, L., H. Sun, W. Zhao, and T. Yu. 2021. "[Retracted] Robotic Arm Control System Based on AI Wearable Acceleration Sensor." *Mathematical Problems in Engineering* 2021, no. 1: 5544375.

Dawson, C., S. Gao, and C. Fan. 2023. "Safe Control With Learned Certificates: A Survey of Neural Lyapunov, Barrier, and Contraction Methods for Robotics and Control." *IEEE Transactions on Robotics* 39, no. 3: 1749–1767.

Feliã, V., K. Rattan, and H. BenjamÃn Brown. 1990. "Kinematics of a Three Degree-of-Freedom, Two Links Lightweight Flexible Arm." IFAC Proceedings Volumes, 23, no. 3: 143–148. 6th IFAC/IFIP/IFORS/IMACS Symposium on Information Control Problems in Manufacturing Technology 1989, Madrid, Spain, 26–29 September.

Ferrarini, S., P. Bilancia, R. Raffaeli, M. Peruzzini, and M. Pellicciari. 2024. "A Method for the Assessment and Compensation of Positioning Errors in Industrial Robots." *Robotics and Computer-Integrated Manufacturing* 85: 102622.

Gao, H., W. He, C. Zhou, and C. Sun. 2018. "Neural Network Control of a Two-Link Flexible Robotic Manipulator Using Assumed Mode Method." *IEEE Transactions on Industrial Informatics* 15, no. 2: 755–765.

Huang, Y., and Y. Chen 2021. "A Safety Guaranteed Control Framework to Integrate Actuator Dynamics via Control-Dependent Barrier Functions." *IFAC-PapersOnLine*, 54, no. 20: 172–178.

Khan, I. U., M. Ouaissa, M. Ouaissa, M. Fayaz, and R. Ullah. 2024. *Artificial Intelligence for Intelligent Systems: Fundamentals, Challenges, and Applications*, 1st ed., 374. CRC Press.

Kroemer, O., S. Niekum, and G. Konidaris. 2021. "A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms." *Journal of Machine Learning Research* 22, no. 30: 1–82.

Kumar, P., and B. Pratiher. 2024. "Nonlinear Modelling and Dynamics of Spatial Multi-Link Rigid-Flexible Manipulator With Moving Platform." *International Journal of Intelligent Robotics and Applications* 8: 735–757. https://doi.org/10.1007/s41315-024-00344-z.

Le Huy, V., and N. D. Dzung. 2021. "Tracking Control of Parallel Robot Manipulators Using RBF Neural Network." In *Research in Intelligent and Computing in Engineering*, edited by R. Kumar, N. H. Quang, V. Kumar Solanki, M. Cardona, and P. K. Pattnaik, 629–642. Springer Singapore.

Lewis, F. L., D. M. Dawson, and C. T. Abdallah. 2003. *Robot Manipulator Control: Theory and Practice*. CRC Press.

Li, Z. 2021. "Comparison Between Safety Methods Control Barrier Function vs. Reachability Analysis." In *Proceedings of the 24th Conference on Formal Methods in Computer-Aided Design - FMCAD 2024*.

Liu, L., X. Wang, X. Yang, H. Liu, J. Li, and P. Wang. 2023. "Path Planning Techniques for Mobile Robots: Review and Prospect." *Expert Systems With Applications* 227: 120254.

Mazzilli, C. E., P. B. Gonçalves, and G. R. Franzini. 2022. "Reduced-Order Modelling Based on Non-Linear Modes." *International Journal of Mechanical Sciences* 214: 106915.

Muralidharan, V., A. Bose, K. Chatra, and S. Bandyopadhyay. 2020. "Methods for Dimensional Design of Parallel Manipulators for Optimal Dynamic Performance Over a Given Safe Working Zone." *Mechanism and Machine Theory* 147: 103721.

Nguyen, Q., A. Hereid, J. W. Grizzle, A. Ames, and K. Sreenath. 2016. "3d Dynamic Walking on Stepping Stones With Control Barrier Functions." In *2016 IEEE 55th Conference on Decision and Control (CDC)*, Las Vegas, NV, USA, 827–834. IEEE.

Nguyen, Q., and K. Sreenath. 2016. "Exponential Control Barrier Functions for Enforcing High Relative-Degree Safety-Critical Constraints." In *2016 American Control Conference (ACC)*, Boston, MA, USA, 322–328. IEEE.

Rokbani, N., and A. Alimi. 2013. "Inverse Kinematics Using Particle Swarm Optimization, a Statistical Analysis." *Procedia Engineering, International Conference on Design and Manufacturing (IConDM2013)*, 64, 1602–1611.

Semwal, V., N. Gaud, and G. Nandi. 2017. "Human Gait State Prediction Using Cellular Automata and Classification Using ELM." In *Conference: International Conference on Machine Intelligence and Signal Processing*.

Singh, R., V. Kukshal, and V. S. Yadav. 2021. "A Review on Forward and Inverse Kinematics of Classical Serial Manipulators." In *Advances in Engineering Design*, edited by P. K. Rakesh, A. K. Sharma, and I. Singh, 417–428. Singapore: Springer Singapore.

Tutsoy, O., and D. E. Barkana. 2021. "Model Free Adaptive Control of the Under-Actuated Robot Manipulator With the Chaotic Dynamics." *ISA transactions* 118: 106–115.

Vairagi, R. D., V. BhaskarSemwal, and M. Vishwakarma. 2024. "Online Planning of Topologically Distinctive Autonomous Drone Trajectories." In *2024 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, Bhopal, India, 1–7. IEEE.

Xiao, F., G. Li, D. Jiang, et al. 2021. "An Effective and Unified Method to Derive the Inverse Kinematics Formulas of General Six-Dof Manipulator With Simple Geometry." *Mechanism and Machine Theory* 159: 104265.