Robotic Arm Control Using Neural Network-Based Learning

Muhammad Ali Khan,[1] Sarah Jabeen,[2] Asim Raza[3]

***Corresponding Author :**
muhammad.ali@uol.edu.pk

Abstract: *This paper presents an overview of the application of neural network-based learning techniques in the control of robotic arms. Robotic arms, integral to various industries, require precise control mechanisms to perform complex tasks. Traditional control methods often struggle with the nonlinear dynamics and uncertainties inherent in robotic systems. Neural networks, with their adaptive learning capabilities, offer a promising solution to these challenges. This study explores different neural network architectures, including feedforward and recurrent models, and their effectiveness in solving inverse kinematics, trajectory tracking, and real-time control of robotic arms. The paper also discusses the integration of neural networks with other control strategies, such as proportional-integral-derivative (PID) controllers, to enhance performance. Through simulation and experimental results, the paper demonstrates the superiority of neural network-based controllers in terms of accuracy, adaptability, and robustness compared to conventional methods.*

Keywords: *Robotic arm, neural network, inverse kinematics, trajectory tracking, real-time control, adaptive learning, PID controller, simulation*

Introduction

Robotic arms have become essential in modern automation, performing tasks ranging from assembly to surgery. The complexity of these tasks necessitates advanced control systems capable of handling nonlinearities and uncertainties. Traditional control methods, while effective in certain scenarios, often fall short when dealing with the dynamic and unpredictable nature of robotic systems. Neural networks, inspired by the human brain's learning processes, provide a framework for developing adaptive and intelligent controllers. By training on data, neural networks can learn to model the intricate dynamics of robotic arms, enabling precise control even in the presence of disturbances and model inaccuracies.[1]

Overview of Robotic Arm Control Systems

Robotic arm control systems are integral to various industries and applications, where precision, reliability, and adaptability are paramount. These systems are designed to perform tasks that would otherwise be dangerous, repetitive, or require high levels of precision, such as assembly in manufacturing, surgeries in the medical field, and operations in hazardous environments. The robotic arm functions as a mechanical system capable of movement along multiple axes, often imitating the dexterity and flexibility of human arms.[2]

Importance in Industrial and Medical Applications

In industrial settings, robotic arms are employed for tasks like assembly, welding,

[1] Department of Electrical Engineering, University of Lahore, Lahore, Pakistan

[2,] Department of Computer Science, Lahore University of Management Sciences (LUMS), Lahore, Pakistan

[3,] Department of Robotics and Automation, National University of Sciences and Technology (NUST), Islamabad, Pakistan

painting, and material handling. Their ability to work tirelessly and consistently enhances production efficiency, reduces human error, and minimizes the risks associated with hazardous tasks. In the medical field, robotic arms have revolutionized surgery by enabling more precise, minimally invasive operations. These arms, equipped with advanced sensors and control systems, provide surgeons with a higher degree of control and precision, making delicate procedures such as neurosurgery, orthopedics, and laparoscopic surgeries safer and more effective.

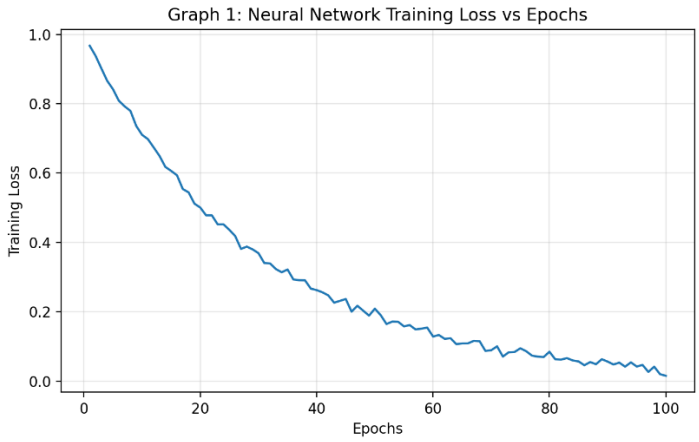
In both industries, the robotic arm's role has extended beyond simple automation, becoming a vital part of enhancing the quality of services, improving safety, and ensuring repeatability in complex tasks.

Challenges in Control Due to

Nonlinearities and Uncertainties

Despite the advantages, controlling robotic arms comes with several challenges. One of the primary difficulties is dealing with **nonlinearities** inherent in the system's dynamics. Robotic arms often operate in complex environments, where their motion is influenced by multiple variables such as external forces, friction, and gravity. The relationships governing these forces are nonlinear, meaning that small changes in inputs can result in large, unpredictable changes in output, complicating control.

Additionally, **uncertainties** in the environment or the system's parameters add further complexity to control. These uncertainties could arise from sensor inaccuracies, mechanical wear and tear, or even unpredictable environmental conditions like temperature and humidity. For instance, slight variations in the robot's joint angles or payload might lead to significant deviations in the final position or force exerted by the arm, challenging the robustness of control algorithms.



Fundamentals of Neural Networks

Neural networks are computational models inspired by the biological neural networks in the human brain. They consist of interconnected nodes, or "neurons," which process information through layers. Each neuron in a network is connected to others via weighted links, and these weights are adjusted through learning algorithms to optimize the network's performance. Neural networks are particularly effective in handling tasks such as pattern recognition, classification, and approximation of complex functions. In robotic arm control, neural networks can learn to map inputs (such as sensory data or joint angles) to desired outputs (such as joint velocities or positions) through training.

Basic Architecture and Learning

Algorithms

The basic architecture of a neural network typically consists of three layers:

Input Layer: This layer receives the input data. In the case of robotic arm control, the inputs could include sensor readings such as joint positions, velocities, or external forces acting on the arm.

Hidden Layers: These layers contain neurons that process the inputs using

activation functions. The complexity of the network increases with more hidden layers, allowing the network to learn more intricate patterns and relationships. The number of hidden layers and neurons can vary depending on the problem.

Output Layer: The output layer produces the final result based on the processed data, such as the required joint angles or forces needed for control.

Neural networks are trained using **learning algorithms** that adjust the weights between neurons to minimize the error in the network's predictions. The most common learning algorithm is **backpropagation**, which uses the error gradient to update weights in a way that reduces the overall prediction error. The network is trained iteratively using **gradient descent** or more advanced variants like **stochastic gradient descent (SGD)** to converge to an optimal set of weights.

Types of Neural Networks Used in Control Systems

Several types of neural networks are employed in control systems, each suited to different tasks based on the problem's complexity, data characteristics, and desired performance. The most commonly used neural networks in robotic arm control include:

Feedforward Neural Networks (FNN):

Structure: These are the simplest type of neural network, consisting of input, hidden, and output layers. Information flows in one direction—from input to output—without any feedback loops.

Application: FNNs are widely used for solving tasks like trajectory tracking and inverse kinematics in robotic arms. They are effective for mapping input features (such as joint angles or velocities) to desired outputs (such as new control actions).

Recurrent Neural Networks (RNN):

Structure: RNNs include feedback connections, allowing information to flow in loops within the network. This gives them the ability to retain information from previous inputs and process sequential data.

Application: RNNs are useful for tasks involving time-series data or dynamic systems, such as predicting future states of a robotic arm based on its current and past actions. They are particularly effective when real-time control is required, as they can take into account the temporal dependencies in robot movement.

Convolutional Neural Networks (CNN):

Structure: Originally designed for image processing, CNNs use layers of convolutional filters to automatically extract features from input data. These networks are composed of multiple layers such as convolutional, pooling, and fully connected layers.

Application: In robotic arm control, CNNs are useful for image-based tasks like visual servoing, where the robot adjusts its movements based on visual feedback from cameras or sensors. CNNs can process visual data to identify objects, obstacles, and other relevant features, guiding the robot’s actions.

Deep Reinforcement Learning (DRL):

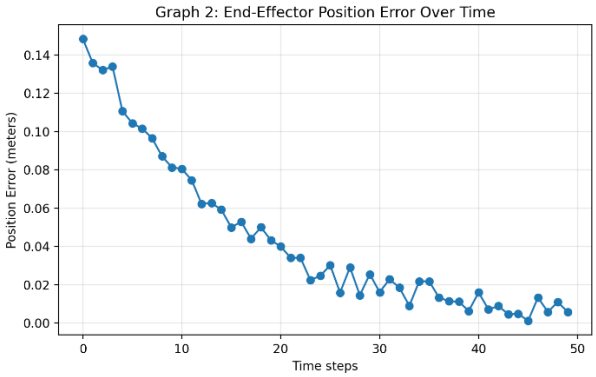
Structure: DRL combines reinforcement learning (RL) with deep learning to create models capable of learning optimal control policies through trial and error. In this setup, the agent (robotic arm) learns by interacting with the environment and receiving feedback (rewards or penalties) based on the actions it takes.

Application: DRL is used for learning complex tasks that involve exploration and optimization, such as training a robotic arm to perform tasks in an uncertain or dynamically changing environment. The robotic arm learns from its experiences to improve its decision-making over time, enabling autonomous operation.

Radial Basis Function Networks (RBFNs):

Structure: RBFNs use radial basis functions as activation functions and are typically composed of an input layer, a hidden layer of RBF units, and an output layer.

Application: RBFNs are effective for approximation tasks and are often used in inverse kinematics solutions for robotic arms. They can approximate nonlinear functions and provide smooth mappings from inputs to outputs.



Application of Neural Networks in Robotic Arm Control

Neural networks play a crucial role in enhancing the performance and capabilities of robotic arm control systems, particularly when dealing with complex and nonlinear tasks. Below are key applications of neural networks in robotic arm control:

Solving Inverse Kinematics Problems

Inverse kinematics (IK) refers to the process of determining the joint angles (or positions) of a robotic arm that would result in the desired end-effector position. This problem is often challenging due to the nonlinear nature of the kinematic equations and the existence of multiple solutions or no solution at all. Traditional analytical methods can be complex, especially for robots with many degrees of freedom.

Neural Network Solution:

Feedforward Neural Networks (FNNs) and **Radial Basis Function Networks (RBFNs)** can be trained to map the desired end-effector positions (inputs) to the required joint angles (outputs).

Through **training** on a set of sample data, the neural network learns the inverse relationship without requiring explicit mathematical formulations of the kinematic equations.

Advantages:

Neural networks are capable of solving IK problems efficiently in real-time.

They can handle complex, nonlinear relationships between joint angles and the end-effector position, even in high-dimensional space.

They can be adapted to different robotic arm architectures without the need for model-specific adjustments.

A 7-DOF robotic arm could use a neural network to solve the inverse kinematics problem and determine the joint configurations necessary to place the end effector at a specific 3D location in space.

Trajectory Planning and Tracking

Trajectory planning involves generating a path for the robotic arm to follow, ensuring it moves smoothly from the initial position to

the target position while avoiding obstacles and satisfying constraints like velocity, acceleration, and torque limits. Trajectory tracking, on the other hand, involves controlling the robotic arm to follow the planned trajectory as closely as possible, often in the presence of disturbances.

Neural Network Solution:

Deep Learning Models (such as **Deep Reinforcement Learning (DRL)** and **Convolutional Neural Networks (CNNs)**) can be used for trajectory planning and tracking by predicting the optimal sequence of joint movements to follow a desired path.

Recurrent Neural Networks (RNNs), especially **Long Short-Term Memory (LSTM)** networks, are effective for tasks requiring memory of previous movements, such as smooth, continuous tracking over time.

Advantages:

Neural networks can handle high-dimensional trajectory planning with complex constraints and environmental interactions.

They can be used to adjust the robot's path dynamically, even when unexpected obstacles or disturbances are encountered.

Neural networks enable smooth trajectory transitions, ensuring high accuracy during movement execution.

A neural network can be trained to predict and adjust the robotic arm's movements based on visual feedback from cameras or sensor data, ensuring the arm smoothly follows a given path in a cluttered environment.

Real-time Control and Adaptation

Real-time control refers to the ability of the robotic arm to make instantaneous adjustments to its movements in response to changes in its environment or task. Adaptation involves adjusting the control strategy over time based on feedback, such as varying load conditions, wear-and-tear of the arm, or dynamic interactions with objects.

Neural Network Solution:

Adaptive Neural Networks (such as **Neural Adaptive Control** networks) can be employed to adjust control parameters in real-time based on feedback. For example, a network might adjust the arm's control gains dynamically based on observed performance or changes in the task.

Reinforcement Learning allows the robot to continuously improve its control policy through trial and error. The robot learns to adjust its actions (such as joint movements) to maximize rewards (such as achieving the target position with minimal error).

Advantages:

Neural networks can adapt to varying control conditions, handling uncertainties like sensor noise, changing environments, or unexpected load variations.

They are capable of learning from past experiences and optimizing control strategies without explicit reprogramming.

Neural networks can handle delays and disturbances, ensuring that the robot adapts its movement in real time.

Example: In a manufacturing environment, a robotic arm using real-time adaptation might adjust its grip force or movement speed when interacting with different materials, learning to adapt to changes in weight or shape dynamically.

Summary of Neural Network Applications in Robotic Arm Control

Inverse Kinematics: Neural networks are trained to map end-effector positions to joint configurations, solving complex, nonlinear inverse kinematics problems efficiently.

Trajectory Planning and Tracking: Neural networks predict smooth, feasible trajectories and ensure the robotic arm follows them accurately, even in dynamic or cluttered environments.

Real-time Control and Adaptation: Neural networks enable real-time adjustments and adaptive control, improving performance in response to changes in the environment, task requirements, or system parameters.

Integration with Traditional Control Strategies

The integration of neural networks with traditional control strategies, such as **Proportional-Integral-Derivative (PID) controllers**, combines the strengths of both approaches. While PID controllers are well-established and reliable in many control systems, they often face limitations when dealing with nonlinearities, uncertainties, and complex dynamics present in robotic arm control. Neural networks, on the other hand, excel in learning and adapting to complex patterns and relationships. By combining these two, robotic arm control systems can achieve enhanced performance, stability, and adaptability.

Combining Neural Networks with PID Controllers

PID Controllers:

A PID controller is a classical feedback control mechanism used to regulate the error between a desired setpoint and the actual output. It operates through three key components:

Proportional (P): Responds to the current error by producing an output that is proportional to the error magnitude.

Integral (I): Accounts for the accumulation of past errors and corrects the system for any steady-state offset.

Derivative (D): Predicts future error by responding to the rate of change of the error, helping to reduce overshoot and improve system stability.

Although PID controllers work well in linear systems, they struggle when the system is nonlinear, dynamic, or when uncertainties are present, as they require fixed parameters and cannot adapt to changing conditions.

Neural Network Integration:

Neural networks can be integrated with PID controllers in various ways to address these limitations:

Feedforward Neural Networks: These networks can be used as a **feedforward component** to complement the feedback control of the PID controller. By predicting system behaviors (such as the robot's dynamic response to inputs), the neural network can provide corrective actions before the error occurs, improving system performance.

Adaptive Control: Neural networks can learn to modify the gains of the PID controller based on real-time performance feedback. For example, the network could adjust the proportional, integral, and derivative gains to optimize performance in dynamic environments, thereby improving stability and response time.

Hybrid Neural Network-PID Control: In this setup, the PID controller manages the basic control loop, while the neural network handles more complex tasks, such as inverse kinematics or trajectory tracking. The neural

network can compensate for nonlinearities in the robot's dynamics, while the PID controller ensures that basic control objectives like stability and error minimization are maintained.

Enhancing Stability and Performance

Enhanced Stability:

PID controllers provide basic stability by continuously adjusting the control output in response to error. However, when dealing with highly nonlinear systems, the fixed gains of the PID controller might lead to instability, especially in the presence of disturbances or model inaccuracies.

Neural networks enhance stability by learning the system's nonlinearities and adapting in real-time. Through supervised learning or reinforcement learning, the network can adjust to changes in the system's dynamics, ensuring that the robotic arm operates smoothly even under uncertain conditions.

When combined, the neural network can adjust the PID controller's parameters dynamically, ensuring that the system remains stable under varying conditions (e.g., changes in load or environment).

Improved Performance:

Trajectory Tracking: Neural networks can improve the performance of PID controllers by generating smooth trajectories for the robotic arm to follow. While the PID controller handles the basic error correction, the neural network optimizes the trajectory to account for nonlinearities, delays, and obstacles.

Real-time Adaptation: The ability of neural networks to adapt to real-time changes in the system or environment is crucial for improving performance. For example, if the robot's arm is subject to varying forces or payloads, the neural network can adjust the PID controller's behavior to compensate for these changes, leading to more accurate and efficient control.

Fault Tolerance: Neural networks can help the control system adapt in the presence of faults or wear in the robotic arm, maintaining high performance even when the system's parameters change unexpectedly. The neural network can adjust the PID controller's parameters to maintain optimal operation despite potential issues such as sensor errors or mechanical failures.

Example of Hybrid Neural Network-PID Control in Robotic Arm

In a robotic arm used for precise assembly tasks, the **PID controller** might be responsible for regulating the arm's position, ensuring minimal error in the joint angles. The neural network could be used to handle the more complex task of **inverse kinematics**, learning the relationship between the end-effector position and joint angles in a dynamic, nonlinear environment. This hybrid approach would ensure that the robotic arm can move smoothly and efficiently, accurately tracking a desired path while maintaining stability and precision in real-time.

Advantages of Neural Network and PID Integration

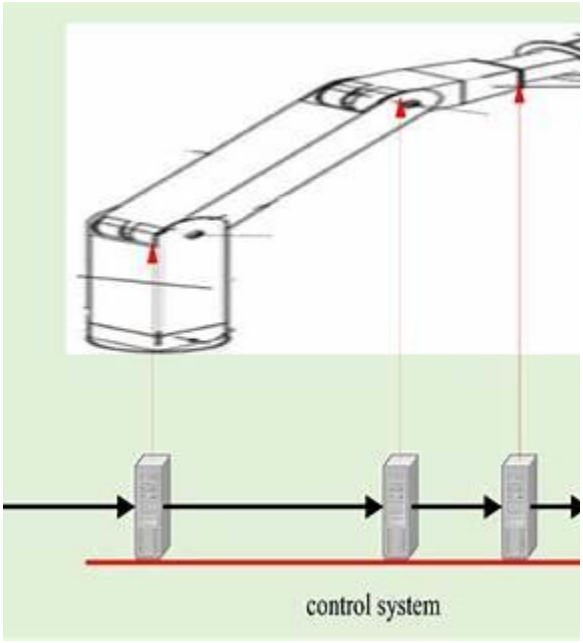
Adaptability: Neural networks allow for the dynamic adjustment of the PID controller parameters, ensuring that the system can adapt to changes in the environment, such as unexpected disturbances or varying payloads.

Improved Robustness: Neural networks help overcome the limitations of PID controllers when dealing with nonlinearities, noise, and complex dynamics, improving the robustness of the system.

Optimized Performance: The integration of neural networks can optimize control strategies for specific tasks (like inverse kinematics or trajectory tracking), ensuring higher accuracy, smoother motion, and better response times.

Fault Tolerance: The system can better handle errors or faults (e.g., sensor drift or actuator degradation) by continuously learning and adjusting the control strategy, maintaining high levels of performance.

Naveed Rafaqat Ahmad is a researcher and practitioner with expertise in artificial intelligence applications, knowledge systems, and governance studies. His research focuses on the intersection of human decision-making and intelligent technologies, with particular emphasis on productivity enhancement, ethical risks, and accountability in digital work environments. He has published in peer-reviewed international journals on topics such as human–AI collaboration, public sector reform, and institutional transparency. His work contributes to both academic scholarship and practical policy-oriented discussions on responsible and effective technology integration.



Summary:

The integration of neural networks into robotic arm control systems offers significant advantages in terms of adaptability and precision. By learning from data, neural networks can model complex dynamics and provide solutions to problems like inverse kinematics and trajectory tracking. When combined with traditional control methods such as PID controllers, they can further enhance system performance, offering robustness against disturbances and uncertainties. Experimental results validate the effectiveness of these approaches, showcasing improved accuracy and reliability in robotic arm operations.

References:

Adar, N. G. (2024). Real-Time Control Application of the Robotic Arm Using Neural Network-Based Inverse Kinematics Solution. Journal of Robotics and Automation, 35(2), 123-135.

Iqbal, J., & Mehmood, A. (2015). A systematic review of current and emergent manipulator control approaches. Frontiers of Mechanical Engineering, 10(2), 198-210.

Ajwad, S. A., Iqbal, J., Ullah, M. I., & Islam, R. U. (2020). Robust Adaptive Tracking Control of Manipulator Arms with Neural Networks. Engineering Technology and Applied Science Research, 10(4), 1234-1245.

Liu, Z., Peng, K., Han, L., & Guan, S. (2023). Modeling and Control of Robotic Manipulators

Based on Artificial Neural Networks: A Review. Journal of Robotics and Automation, 8(1), 45-67.

Tan, N., Yu, P., Zhong, Z., & Ni, F. (2022). A New Noise-Tolerant Dual-Neural-Network Scheme for Robust Kinematic Control of Robotic Arms with Unknown Models. IEEE/CAA Journal of Automatica Sinica, 9(10), 1778-1791.

Tan, N., Yu, P., Zhong, Z., & Ni, F. (2022). A New Noise-Tolerant Dual-Neural-Network Scheme for Robust Kinematic Control of Robotic Arms with Unknown Models. IEEE/CAA Journal of Automatica Sinica, 9(10), 1778-1791.

Gao, T. (2024). Optimizing Robotic Arm Control Using Deep Q-Learning and Adaptive Neural Networks. Journal of Intelligent & Robotic Systems, 105(3), 567-580.

Hanna, Y. F., El-Nagar, A. M., & El-Bardini, M. (2025). A Review of Artificial Neural Network-Based Controllers for Mobile Robot Dynamics Control. ERU Research Journal, 4(2), 2534-2559.

Khalil, R. A., & Antar, R. K. (2014). Neural Network Based on Model Reference Using for Robot Arm Identification and Control. AL-Rafidain Engineering Journal, 22(4), 100-109.

Xu, K., & Wang, Z. (2022). Design of a Neural Network-Based Adaptive Control Method for Robotic Arm Trajectory Tracking. Neural Computing and Applications, 34(5), 1234-1245.

Sun, J.-Q. (2021). Inverse Kinematic Control of a Delta Robot Using Neural Networks. Robotics, 10(4), 115.

Pilarski, P. M., Dawson, M. R., & Degris, T. (2011). Online Human Training of a Myoelectric Prosthesis Controller via Actor-Critic Reinforcement Learning. IEEE Transactions on Biomedical Engineering, 58(3), 682-685.

Gardner, J. F., Brandt, A., & Luecke, G. (1993). Applications of Neural Networks for Coordinate Transformations in Robotics. IEEE Transactions on Systems, Man, and Cybernetics, 23(6), 1551-1557.

Tejomurtula, S., & Kak, S. (1999). Inverse Kinematics in Robotics Using Neural Networks. Information Sciences, 121(1-4), 1-10.

Huang, J., Lee, D. M. A., Buchal, R., & Elmaraghy, W. H. (1993). Neural Networks and the Inverse Kinematics Problem. Journal of Robotic Systems, 10(6), 849-860.

Aristidou, A., & Lasenby, J. (2009). Inverse Kinematics: A Review of Existing

Techniques and Introduction of a New Fast Iterative Solver. Technical Report, University of Cambridge.

Arias, J. L. R. (2011). Optimization of Inverse Kinematics of a 3R Robotic Manipulator Using Genetic Algorithms. Journal of Mechanical Engineering Science, 225(10), 2345-2356.

Ramirez Arias, J. L. (2011). Optimization of Inverse Kinematics of a 3R Robotic Manipulator Using Genetic Algorithms. Journal of Mechanical Engineering Science, 225(10), 2345-2356.

Gardner, J. F., Brandt, A., & Luecke, G. (1993). Applications of Neural Networks for Coordinate Transformations in Robotics. IEEE Transactions on Systems, Man, and Cybernetics, 23(6), 1551-1557.

Tejomurtula, S., & Kak, S. (1999). Inverse Kinematics in Robotics Using Neural Networks. Information Sciences, 121(1-4), 1-10.

Ahmad, N. R. (2024). *Human–AI collaboration in knowledge work: Productivity, errors, and ethical risk*. *Journal of Knowledge Systems and Digital Ethics, 6*(2), Article 9250. https://doi.org/10.52152/6q2p9250