Enhancing Robotic Autonomy: A Review and Case Study of Traditional and Deep Learning Approaches to Inverse Kinematics

Omar A. Al-Sharif[1*], Nourhan A. Abbass[1], Ahmed M. Hanafi[1], Abdelrady O. Elnady[1]

1 Department of Mechatronics Engineering, Faculty of Engineering, October 6 University, 6th of October City, 12585, Giza, Egypt

* Email of Corresponding Author: omarahmed.eng@o6u.edu.eg

Abstract – Inverse kinematics (IK) is a fundamental concept in robotics, essential for calculating the necessary joint angles to achieve a desired position and orientation of an end-effector in robotic arms and other articulated systems. Traditional methods of solving IK, such as analytical and numerical approaches, face significant challenges when addressing modern robotic applications' complexities and computational demands. This paper explores the integration of deep neural networks (DNNs) as a transformative approach to these IK challenges. DNNs offer a significant advantage in handling the nonlinearities inherent in robotic systems and provide a flexible framework for optimizing joint configurations with energy efficiency and obstacle avoidance considerations. The study presents a detailed comparison of traditional and neural network-based methods, highlighting the enhanced adaptability, efficiency, and robustness of neural networks. The paper discusses neural network architectures and their implementation for a 2D robotic arm. This advancement represents a pivotal shift from rigidity to flexibility in robotic motion planning and control, promising substantial improvements in robotic autonomy and functionality.

Keywords: Inverse Kinematics, Neural Networks, Robotics, Deep Learning, Articulated Robots

1 Introduction

Inverse kinematics (IK) is a pivotal concept in robotics essential for calculating the necessary joint angles of a robotic arm or other articulated systems to achieve a desired position and orientation of its end-effector, such as a gripper. This process contrasts with forward kinematics, which involves determining the end-effector's location based on known joint angles. IK is fundamental in various applications, from controlling robots and planning their motions to generating realistic animations in computer graphics [1].

Suppose having a robotic arm with two joints (a 2-link planar arm), as shown in Figure 1, and determining the joint angles that allow the end-effector (hand) of the robot to reach a specific point in the 2D plane. The inverse kinematics problem here is finding the joint angles θ1 and θ2 given the end-effector's desired position (x, y).

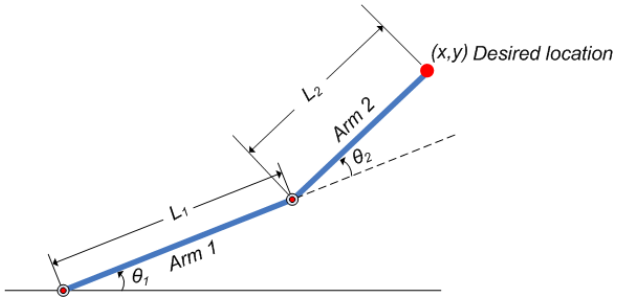

Figure 1: the two-joint robotic arm with the two angles, θ1 and θ2.

Inverse kinematics (IK) is a fundamental concept in robotics used to calculate the required joint angles of a robotic arm or similar articulated system to achieve its end-effector's desired position and orientation. This contrasts with forward kinematics, which determines the end-effector's location based on joint angles, as shown in Figure 2. IK plays a vital role in robot control, motion planning, and the creation of realistic animations.

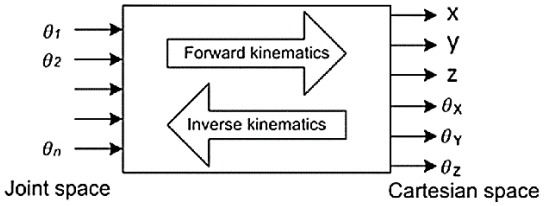

Figure 2: Relationship between forward and inverse kinematics.

The accuracy and efficiency of solving IK problems directly impact a robot's ability to interact with its environment effectively and perform highly precise tasks. However, traditional IK methods often face challenges related to system complexity and computational demands, especially in real-time applications [2].

Neural networks present a promising alternative to traditional IK solutions due to their ability to model complex, non-linear relationships inherent in robotic systems. They are particularly advantageous for systems with redundant degrees of freedom, offering the potential to select optimal joint configurations based on additional criteria such as energy efficiency or obstacle avoidance [3].

This paper discusses the traditional approaches to solving IK problems, outlining their strengths and limitations, and introduces how modern advancements in neural networks offer potent solutions to overcome these challenges. By enhancing the ability to compute joint configurations accurately and efficiently, neural networks are revolutionizing the field, providing more adaptive and robust systems for complex robotic applications.

2 Traditional IK Methods

Traditional methods for solving inverse kinematics (IK) in robotics and computer graphics are typically categorized into two main approaches: analytical and numerical methods, each with its unique strengths and suited for different problem types.

Numerical methods approach inverse kinematics through iterative refinement, using techniques like manipulating the Jacobian matrix, gradient descent, and heuristic methods like Cyclic Coordinate Descent. These methods are flexible and capable of handling complex systems but tend to be slower and risk failing to find the optimal solution.

The choice between analytical and numerical methods for solving inverse kinematics depends on factors such as the complexity of the robotic arm, the need for real-time processing, and the precision required in positioning the end effector. Each method requires a deep understanding of the system's kinematics and involves complex calculations.

2.1 Traditional IK Challenges

Traditional methods for solving inverse kinematics (IK) encounter several challenges, particularly as the complexity of robotic systems and application requirements increase. A primary issue is the complexity of systems with a high number of degrees of freedom, which makes the IK equations more complicated to solve. Physical constraints such as joint limits and mechanical interferences further complicate the IK problem.

Non-linear relationships between joint angles and the end effector's position contribute to the complexity, resulting in non-linear equations that are difficult to solve. Singularities in the Jacobian matrix used in numerical methods can also occur, leading to undefined solutions and unpredictable robotic arm behaviors.

Computational efficiency is another significant challenge. While analytical solutions can be very fast, they are limited to simpler systems. Numerical methods offer more flexibility but can be computationally intensive, which might not be practical for real-time applications. These methods often rely on iterative algorithms that may converge slowly and require parameter tuning to achieve acceptable performance, potentially leading to inaccuracies and instability, especially in high-speed movements or near-singular configurations.

Optimization-based methods for IK can get trapped in local minima, making it difficult to find the global optimum solution. Additionally, IK solutions need to integrate seamlessly with other parts of the robotic system, such as path planning and collision detection. Misalignments in integration can lead to inefficient or unsafe operations. Addressing these challenges often involves a blend of advanced mathematical techniques and sophisticated algorithm design, sometimes integrating learning-based methods with traditional approaches to developing robust solutions for real-world applications.

Addressing these challenges often requires combining advanced mathematical techniques, sophisticated algorithm design, and sometimes integrating learning-based methods with traditional approaches to create robust and flexible solutions for real-world applications.

3 Inverse Kinematics (IK) Using Neural Networks.

Neural networks can be a powerful solution for the inverse kinematics challenge due to several key advantages:

The non-linear nature of many robotic systems, along with the potential for redundant degrees of freedom (wherein multiple joint configurations may achieve the same end-effector position), presents challenges for conventional IK methods. Neural networks are well-suited to model these non-linear relationships and can learn efficient solutions for redundant systems, choosing among many possible joint configurations to optimize additional criteria like smooth motion or obstacle avoidance.

A key strength of neural networks lies in their ability to generalize. When trained on a diverse dataset of joint configurations and end-effector positions, a well-trained neural network can often produce reasonable solutions even for positions it has not encountered during training. This makes them more adaptable than some traditional IK methods, which might be prone to failure when presented with new scenarios outside their specifically programmed parameters.

Solving inverse kinematics problems using analytical or iterative methods can be computationally expensive, especially for robots with many joints. In contrast, once trained, a neural network acts as a highly optimized function. Given a desired end-effector position, it can rapidly compute the corresponding joint angles. This speed advantage opens up the possibility of using neural networks for real-time robot control, where fast reactions are essential.

Neural networks can incorporate data from various sensors (e.g., cameras, force, tactile sensors). Integrating this sensor data allows the network to refine its inverse kinematics solutions. This might involve adjusting joint angles to avoid obstacles detected by a camera or responding to the feel of an object through tactile sensors. Integrating sensor data makes robots more adaptable and responsive to their environments.

When designing neural networks for inverse kinematics (IK), the choice of architecture can significantly influence the performance and efficiency of the system. Below are some common neural network architectures frequently used to solve IK problems:

3.1 Feed-forward Neural Networks (FNNs)

FNNs have proven to be effective in solving the Inverse Kinematics (IK) problem, particularly in robots with limited degrees of freedom (DOF). Studies have shown that multilayer FNNs can be trained to map desired end-effector poses to corresponding joint angles. This approach offers a straightforward and efficient solution, overcoming the limitations associated with traditional algorithmic methods[1].

3.2 Reinforcement Learning (RF)

Reinforcement learning has become a valuable tool in robotics, particularly in addressing complex issues like inverse kinematics. Inverse kinematics, which involves determining the joint parameters of a robot to achieve a desired end-effector position, can be challenging to solve analytically for intricate robotic systems. Reinforcement learning techniques can provide a promising alternative to manually specifying reward functions [4].

Recent studies have investigated the use of deep reinforcement learning algorithms in solving inverse kinematics problems. For example, Lin et al. [5] applied automatic goal generation to a deep reinforcement learning algorithm to effectively learn an inverse kinematics solution for a hybrid banana-harvesting robot. Similarly, Sekkat et al.[6] proposed a methodology that employs deep reinforcement learning to generate joint-space trajectories for stable configurations, aiding in solving inverse kinematics for robotic arm control.

Furthermore, integrating inverse reinforcement learning with real trajectories has been demonstrated to enhance the reliability of simulations. Utilized Inverse Reinforcement Learning with algorithms like Maximum Entropy and Feature Matching to estimate reward functions for interactions between cyclists and pedestrians, demonstrating the practical applications of this approach [7].

Moreover, the literature emphasizes the importance of inverse reinforcement learning in learning reward functions directly from expert demonstrations. This approach provides a valuable paradigm for understanding underlying reward mechanisms without manual intervention, highlighting the potential of inverse reinforcement [8], [9] learning in inferring expert reward functions from demonstrations, offering a viable solution to the challenge of reward engineering.

3.3 Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) have been extensively studied and applied in various fields due to their ability to model sequential data effectively. RNNs, such as Long Short-Term Memory (LSTM) networks, are known for capturing long-range temporal dependencies and addressing issues like the vanishing gradient problem in conventional RNNs [10]. These networks have been successfully used in speech recognition [11], heart failure onset detection [12], activity recognition [13], speech synthesis [14], and even for computing the Drazin inverse of matrices [15]. The flexibility and power of RNNs lie in their capability to learn and represent complex patterns in sequential data [16].

In the context of robotics and kinematics, Recurrent Neural Networks (RNNs) have shown promise in solving inverse kinematics problems. For instance, a novel integration-enhanced recurrent neural network (IE-RNN) was proposed specifically for resolving kinematic resolutions of redundant robot manipulators [17]. Additionally, RNNs have been utilized for transient detection in robotic assembly tasks, where joint torques of robots are used as input [18]. These applications demonstrate the versatility of RNNs in handling complex tasks in robotics and automation.

Moreover, RNNs have been employed in dynamic models for soft continuum manipulators, showcasing their ability to approximate high-dimensional equations in real-time scenarios [19]. The use of RNNs in real-time applications, such as edge inference accelerators, highlights their significance in enabling low-latency processing for tasks like IoT, robotics, and human-machine interaction [20], [21].

3.4 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) have gained significant traction in various fields, particularly in computer vision and robotics. CNNs have been extensively utilized in tasks such as image classification, object detection, and segmentation due to their ability to extract hierarchical features from visual data [22], [23]. These networks have shown remarkable performance in diverse applications, including depth estimation, object recognition, and grasp detection in robotics [24], [25], [26].

In the context of inverse kinematics, CNNs offer a promising approach to solving complex problems. By leveraging the capabilities of CNNs in extracting features and learning spatial relationships, these networks can be applied to infer joint configurations from end-effector positions in robotic systems. The use of CNNs for robot grasp detection and object gripping algorithms demonstrates their potential to handle kinematic tasks efficiently [24], [25], [26].

Moreover, CNNs have been employed in robot path planning, where the environment is classified based on visual inputs to facilitate navigation [27]. Additionally, CNNs have been integrated with reinforcement learning techniques for robot motion planning under uncertain conditions, showcasing their adaptability and robustness in dynamic environments [28].

Furthermore, the application of CNNs in real-time grasp detection and assembly manipulation tasks highlights their effectiveness in enhancing human-robot interaction and improving the efficiency of industrial processes [29], [30]. The ability of CNNs to process visual data and make informed decisions based on learned patterns makes them a valuable tool for addressing kinematic challenges in robotics.

3.5 Autoencoders

Autoencoders, particularly Variational Autoencoders (VAEs), have gained prominence in unsupervised learning tasks due to their ability to capture complex distributions effectively [31]. In the context of robotics and, specifically, inverse kinematics, autoencoders offer a promising approach to solving intricate problems. By leveraging the encoding-decoding mechanism of autoencoders, these models can learn meaningful representations of input data, which can be particularly useful in inferring joint configurations from end-effector positions in robotic systems.

The application of autoencoders in solving inverse kinematics problems has been demonstrated in various studies. For instance, a study utilized a convolutional autoencoder's decoder function for reduced-order modeling of non-linear time-dependent parametrized partial differential equations [32]. Additionally, the latent space features of a convolutional autoencoder were employed in a hybrid machine learning inversion method for estimating subsurface velocity models [33]. These applications showcase the versatility of autoencoders in handling complex tasks in different domains.

Moreover, autoencoder-based solutions have been proposed for fault diagnosis in wind turbine generators, emphasizing the robustness and generalizability of autoencoder models in diverse engineering applications [34]. Furthermore, using autoencoders in generative graphical inverse kinematics models highlights their capacity to represent diverse solution efficiently sets [35].

3.6 Graph Neural Networks (GNNs)

Graph Neural Networks (GNNs) have emerged as a powerful tool in various domains, including robotics, due to their ability to effectively model graph-structured data. Recent advances in GNNs have shown considerable success

in addressing complex challenges, such as inverse kinematics in robotic systems [36]. By leveraging the flexibility and adaptability of GNNs, researchers have explored innovative approaches to tackle inverse kinematics problems efficiently.

One study investigated a novel distance-geometric robot representation coupled with a graph structure to leverage the capabilities of GNNs for generative graphical inverse kinematics [35]. This approach aimed to address key shortcomings in traditional methods by utilizing the flexibility of GNNs to model complex relationships in robot kinematics effectively. Additionally, a learning framework was proposed to discover the representation of a robot's kinematic structure and motion embedding spaces using GNNs[10]. This framework highlights the potential of GNNs in capturing intricate relationships within robotic systems for tasks like motion planning and control.

Moreover, a study presented a gradient neural network (GNN) to solve the time-varying inverse kinematics problem of a four-wheel mobile robotic arm, showcasing the ability of GNNs to approximate time-varying inverse kinematics solutions efficiently [37]. This application demonstrates the utility of GNNs in addressing dynamic kinematic challenges in robotic systems.

3.7 Attention Mechanisms and Transformers

Attention Mechanisms and Transformers, initially developed for natural language processing, have been adapted to solve inverse kinematics (IK) problems in robotics, specifically for soft robotic limbs. By reframing the IK challenge as a sequential prediction problem, Kinematics Transformer offers a sophisticated approach to modeling the intricate movements of soft robots. This methodology leverages the transformer architecture to perform numerical simulations that control soft limbs with higher accuracy and precision compared to traditional feed-forward neural networks. Such advancements demonstrate the potential of advanced computational models to enhance the functionality and adaptability of robotic systems in dynamic environments [38].

3.8 Integration with Classical Techniques

Hybrid models that combine neural networks with classical kinematic solvers can be effective in solving inverse kinematics (IK) problems, particularly for complex mappings or when physical constraints need to be met. These hybrid models use neural networks to approximate complex mappings and classical methods to refine solutions or ensure physical constraints are met.

One example of a hybrid model is the use of an Extreme Learning Machine (ELM) to solve IK problems with a hybrid forward model [39]. The ELM learner is used to exploit the hybrid forward model for IK, which is a combination of classical kinematics and data-driven modeling. This approach is effective in improving IK for complex systems.

Another example is the use of a hybrid algorithm for IK using deep learning and coordinate transformation [40]. This algorithm derives and verifies rigid and flexible body kinematic and dynamic models of complex parallel robots, including crank and slider mechanisms.

A third example is the use of a hybrid analytical-neural inverse kinematics solution (HybrIK) for 3D human pose and shape estimation [41], [42]. HybrIK directly transforms accurate 3D joints to relative body-part rotations for 3D body mesh reconstruction via the twist-and-swing decomposition. The swing rotation is analytically solved with 3D joints, and the twist rotation is derived from visual cues through a neural network. This approach preserves both the accuracy of 3D pose and the realistic body structure of the parametric human model, leading to a pixel-aligned 3D body mesh and a more accurate 3D pose than pure 3D key point estimation methods.

4 Selecting The Appropriate Neural Network Architecture

The selection of a suitable neural network architecture for inverse kinematics (IK) is critical and varies; several key factors need to be considered beyond just the complexity of the robot and the specific tasks it needs to perform. Utilizing neural networks for IK offers an innovative approach to estimating joint configurations, enhancing error reduction, and improving adaptability in robotic control systems. To ensure optimal performance and efficiency, it is essential to consider various key factors when choosing the appropriate neural network architecture for IK solutions. These factors include:

The complexity of the robot, particularly the number of degrees of freedom, directly impacts the choice of neural network architecture. Robots with more joints and moving parts have more complex kinematic chains, making the modeling of joint angles to end-effector positions more challenging. Deep neural networks are preferred for these applications because they can learn complex, hierarchical feature representations. The deep layers of these networks can understand intricate, non-linear relationships which are typical in highly articulated robots.

The specific operational tasks assigned to the robot also dictate the architectural choice. For tasks requiring high precision, such as assembly operations, the stability and accuracy of the network are paramount. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), including LSTMs and GRUs, are highlighted as beneficial for these tasks. CNNs excel in spatial recognition which can be crucial in assembly where precise placement is necessary. RNNs, on the other hand, are effective at handling sequences and temporal data, making them ideal for tasks that involve complex movements and require maintaining temporal dependencies. These capabilities ensure the production of consistent and reliable outputs needed for precision tasks.

The size and quality of available training data are paramount. When diverse, abundant data is available, more complex models can be employed without the risk of overfitting, leveraging deep networks to extract intricate patterns. Conversely, limited data necessitates simpler architectures, less prone to overfitting, and better generalizing with less data. Transfer learning may also be appropriate with limited training examples.

In scenarios where real-time processing is essential, the choice of neural network architecture leans heavily towards those that are computationally efficient. Lightweight models that support parallelization, such as Convolutional Neural Networks (CNNs), are preferred because they require less computational power and can provide faster inference times. This is particularly critical in environments with limited computational resources, such as embedded systems or edge devices, where the capability to handle complex models is restricted. In these cases, choosing inherently efficient architectures or applying optimizations like Depth-wise Separable Convolutions can be essential to meet performance criteria without overburdening the hardware.

One crucial factor is the generalization needs of the system. The ability of the neural network to generalize well to new, unseen scenarios is particularly important for robots operating in dynamic environments. Robots often need to handle a wide range of situations and adapt to changing conditions, so the neural network architecture must be able to generalize beyond the training data. Techniques like regularization and architectures that can handle variance well, such as Variational Autoencoders and Generative Adversarial Networks, can help improve the generalization capabilities of the neural network.

Another important consideration is the robustness and noise tolerance of the architecture. In real-world environments, the input data the robot receives may be noisy or incomplete due to sensor imperfections or other environmental factors. Choosing a neural network architecture that can handle such uncertainties is essential for robust performance. Architectures incorporating attention mechanisms or integrating uncertainty estimation within their framework, like Bayesian Neural Networks, can be beneficial in these scenarios.

A critical aspect of deploying neural networks in robotic systems is their ability to integrate with other components of the robotic system, such as sensors, control systems, and additional processing units. Effective integration ensures that the neural network can receive inputs from and send outputs to other systems smoothly, maintaining low latency and proper synchronization. This is essential for the overall performance and functionality of the robotic system, as poor integration can lead to delays, errors, and inefficiencies.

5 Methods

5.1 Solving IK Problem for a 2D Robotic Arm

Forward Kinematics Simulation: Use forward kinematics to generate training data as shown in Figure 3. For each pair of joint angles θ_1 and θ_2, calculate the corresponding end-effector position using the arm's forward kinematics equations:

$$X = L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2) \quad (1)$$

$$Y = L_1 \sin(\theta_1) + L_2 \sin(\theta_1 + \theta_2) \quad (2)$$

where L_1 and L_2 are the lengths of the first and second arm segments, respectively.

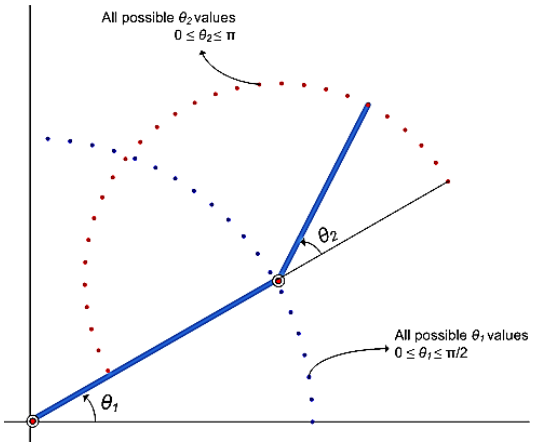

Figure 3: All possible θ1 and θ2 values.

5.2 Generate Training Data

Randomly sample values of θ_1 and θ_2 within their allowable ranges (0 to $\pi/2$ and 0 to π respectively) generate various (x, y) positions.

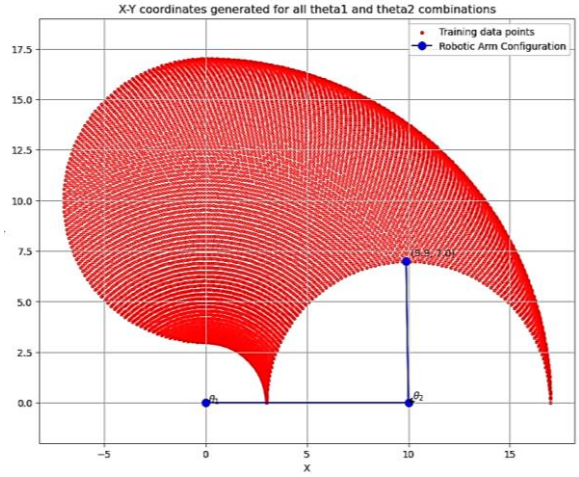

Figure 4: X-Y coordinates generated for all θ_1 and θ_2 combinations using the forward kinematics formula.

5.3 Neural Network Architecture and Training

The neural network designed for inverse kinematics features a two-neuron input layer that captures the X and Y coordinates of a robotic arm's end-effector. This feeds into a structure with two hidden layers—the first with 64 neurons and the second with 32, both utilizing ReLU activation functions for non-linear processing. The network concludes with an output layer of two neurons predicting the joint angles, θ_1 and θ_2. During training, it employs a Mean Squared Error loss function and an Adam optimizer over 1000 epochs at a learning rate of 0.001, facilitating efficient weight adjustments and convergence to accurate angle predictions.

6 Results

Table 1summarizes the performance of a neural network model trained to predict the joint angles, θ_1 and θ_2, for the inverse kinematics of a two-joint robotic arm. Across ten test points (P1 to P10), the model's predictions are compared with the actual angles, with the discrepancies quantified as end-effector errors.

Table 1. θ1and θ2 Actual vs Predicted Values and End Effector Error

Point	Actual		Predicted		End Effector Error
	θ_1	θ_2	θ_1	θ_2	
P1	1.49	2.95	1.51	2.91	0.31
P2	1.55	0.25	1.52	0.41	0.55
P3	1.55	0.73	1.55	0.75	0.11
P4	0.10	1.87	0.13	1.87	0.30
P5	0.68	0.73	0.73	0.60	0.36
P6	0.40	2.60	0.39	2.64	0.27
P7	0.10	1.75	0.12	1.75	0.25
P8	0.81	2.03	0.83	2.01	0.23
P9	1.11	0.83	1.14	0.72	0.40
P10	1.54	1.08	1.55	1.06	0.08

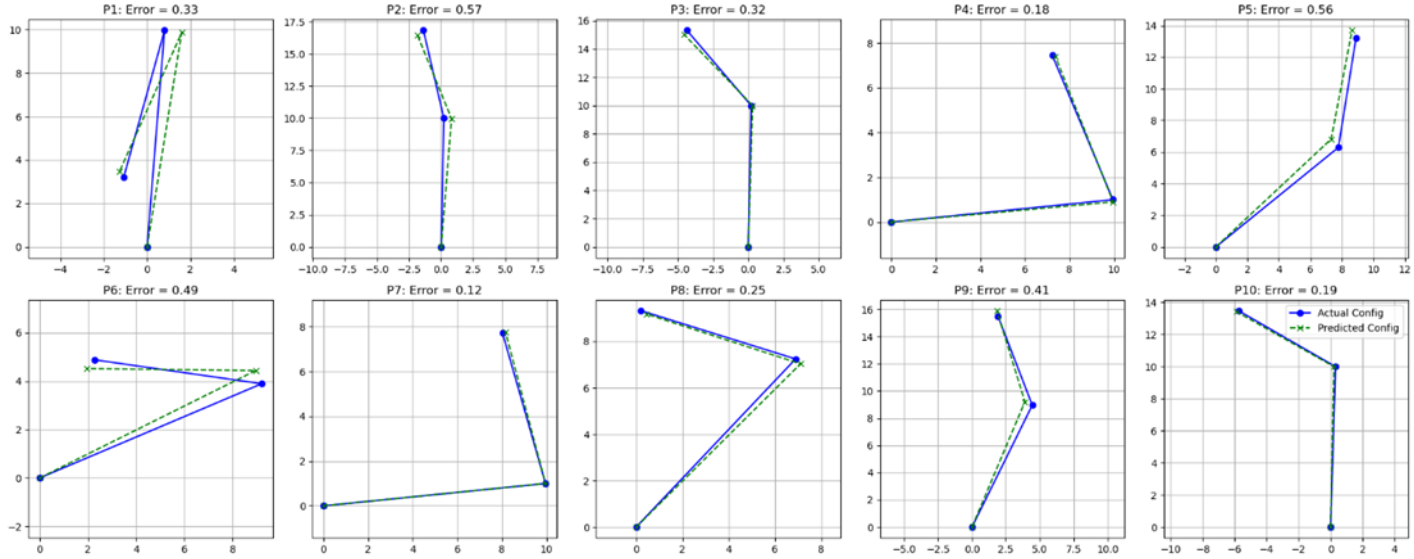

Figure 5: the Actual θ_1 and θ_1 vs. Predicted Values and End Effector Error

The results show that for most points, the predicted angles are quite close to the actual angles, with θ_1 predictions being particularly accurate. The end-effector errors, which measure the Euclidean distance between the actual and predicted positions of the end effector, are generally low, indicating a good model performance. For example, at point P3, the error is just 0.11 radians, showcasing high accuracy in the model's prediction. The largest error observed is at point P2, with an error of 0.55 radians, suggesting that certain areas of the input space are more challenging for the model to learn.

Table 2: $\theta_1$and θ_2 Actual vs Predicted Values and End Effector Error

Point	Actual		Predicted		End Effector Error
	θ1	θ2	θ1	θ2	
P1	1.49	2.95	1.51	2.91	0.31
P2	1.55	0.25	1.52	0.41	0.55
P3	1.55	0.73	1.55	0.75	0.11
P4	0.10	1.87	0.13	1.87	0.30
P5	0.68	0.73	0.73	0.60	0.36
P6	0.40	2.60	0.39	2.64	0.27
P7	0.10	1.75	0.12	1.75	0.25
P8	0.81	2.03	0.83	2.01	0.23
P9	1.11	0.83	1.14	0.72	0.40
P10	1.54	1.08	1.55	1.06	0.08

Figure 5, which visualizes the actual vs. predicted configurations, would provide a visual confirmation of this performance, highlighting where the end effector lands for both actual and predicted states. The model seems to perform well, with low end-effector errors on average, indicating that it has learned to effectively approximate the inverse kinematics function of the robotic arm.

6.1 Important Considerations

The success of a neural network in handling the IK problem largely depends on the quality and diversity of the training data. It's essential that the dataset comprehensively covers the robot's operational workspace with a wide range of end-effector positions and orientations. This ensures that the network learns to generalize effectively across the scenarios it might encounter. Additionally, incorporating a diverse set of examples helps avoid overfitting to specific patterns and biases, which is crucial for the robot to perform reliably in varying conditions.

The IK problem is unique because it can have multiple valid solutions; different joint configurations can achieve the same end-effector position and orientation. To manage this, one strategy is to guide the neural network toward a preferred solution by providing additional context, such as previous joint values or specific motion preferences (e.g., avoiding obstacles or minimizing energy usage). Another approach is

to constrain the training data to specific regions of the workspace where a unique solution is more likely, simplifying the learning process but reducing the model's flexibility.

6.2 Limitations

Neural networks are primarily used in inverse kinematics to provide approximate solutions. These solutions are often highly accurate and can be sufficient for many practical applications. However, they are rarely perfect, particularly in scenarios where the slightest deviation can lead to significant errors or inefficiencies. Traditional analytical methods, which are based on mathematical formulations specific to the mechanics of the system, often provide more exact solutions. These methods may still be preferred in critical applications requiring ultimate precision.

Neural networks learn from the data they are trained on. While they can generalize to some extent, their performance might degrade when presented with end-effector positions drastically different from those in their training set. If the robot is likely to encounter vastly novel scenarios, it might need to carefully curate the training data or develop additional strategies to handle unexpected input.

One of the more significant criticisms of neural networks is their "black box" nature. Unlike traditional analytical methods, where the steps of a computation can be clearly defined and followed, neural networks process inputs through hidden layers and non-linear transformations that are not easily interpretable. This makes it challenging to understand exactly how they arrive at a particular solution.

7 Conclusion

This research has explored the evolution and enhancement of inverse kinematics (IK) solutions from traditional methods to advanced neural network approaches. Traditional IK methods, while foundational in robotics for determining joint angles and end-effector positions, have been shown to face significant limitations, particularly when dealing with systems that exhibit high degrees of freedom and require real-time operational capabilities. These traditional methods struggle with the complexity and computational demands imposed by modern robotic applications, which often prevent them from achieving the desired efficiency and accuracy.

Deep neural networks offer significant advantages in solving IK problems. They leverage their ability to learn from extensive training data and handle inherent nonlinearities in robotic systems, enhancing both the adaptability and computational efficiency compared to traditional methods. Furthermore, the diverse methodologies of deep learning, including recurrent neural networks, convolutional neural networks, and deep reinforcement learning, offer a powerful toolkit for robotics. Each approach possesses unique strengths, from modeling complex data relationships to enhancing real-time control.

This transition from traditional IK methods to those powered by deep neural networks marks a significant milestone in robotics. It not only enhances the performance and capabilities of robotic systems but also opens up new avenues for future research and application, potentially leading to more sophisticated and autonomously functioning robots with applications in many areas. As this technology continues to evolve, it promises to refine the interaction between robots and their environments, making them more responsive and adept at performing complex tasks.

Declarations

Availability of data and materials

The authors have not used any data in our study.

Competing interests

The authors declare that they have no competing interests.

Funding

his research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Authors' contributions

All authors contributed extensively to the work presented in this paper. OA led the entire process of this study. All authors read and approved the final manuscript.

Acknowledgments

Not applicable

References

[1] University of Illinois Urbana-Champaign, https://motion.cs.illinois.edu/RoboticSystems/InverseKinematics.html.

[2] M Jin, Q Liu, B I N Wang, and H Liu2020, An Efficient and Accurate Inverse Kinematics for 7-DOF Redundant Manipulators Based on a Hybrid of Analytical and Numerical Method, vol. **8**.

[3] C Ho and C King2022, Automating the Learning of Inverse Kinematics for Robotic Arms with Redundant DoFs, vol. **X**, no. X, pp. 1–14.

[4] J Kober, J A Bagnell, and J Peters2013, The International Journal of,

[5] G Lin, P Huang, M Wang, Y Xu, R Zhang, and L Zhu2022, An Inverse Kinematics Solution for a Series-Parallel Hybrid Banana-Harvesting Robot Based on Deep Reinforcement Learning, pp. 1–15.

[6] H Sekkat, S Tigani, and R Saadane2021, applied sciences Vision-Based Robotic Arm Control Algorithm Using Deep Reinforcement Learning for Autonomous Objects Grasping.

[7] F Martinez-gil, M Lozano, I García-fernández, and P Romero2020, Using Inverse Reinforcement Learning with Real Trajectories to Get More Using Inverse Reinforcement Learning with Real Trajectories to Get More Trustworthy Pedestrian Simulations, no. September,

[8] P Henderson, W Chang, P Bacon, D Meger, J Pineau, and D Precup2017, Inverse Reinforcement Learning.

[9] J Fu, K Luo, and S Levine2018, L EARNING R OBUST R EWARDS WITH A DVERSARIAL, pp. 1–15.

[10] M Kim, B Cao, K An, and J Wang2018, Dysarthric Speech Recognition Using Convolutional LSTM Neural Network, no. September,

[11] A M and G H Alex Graves, SPEECH RECOGNITION WITH DEEP RECURRENT NEURAL NETWORKS Alex Graves , Abdel-rahman Mohamed and Geoffrey Hinton Department of Computer Science , University of Toronto, no. 3.

[12] E Choi, A Schuetz, W F Stewart, and J Sun2016, Using recurrent neural network models for early detection of heart failure onset Using recurrent neural network models for early detection of heart failure onset, no. August,

[13] F J Ordóñez and D Roggen2016, Deep Convolutional and LSTM Recurrent Activity Recognition,

[14] B Prihasto, T Tai, P Chang, and J Wang2022, Fast Gated Recurrent Network for Speech Synthesis, no. 9, pp. 1634–1638.

[15] Y Stanimirović, P, Živković, I, & Wei2015, Recurrent Neural Network for Computing the Drazin Inverse, *IEEE Trans. Neural Networks Learn. Syst.*, vol. **26(11):283**,

[16] C Gulcehre, F Bougares, H Schwenk, and Y Bengio2014, Learning Phrase Representations using RNN Encoder – Decoder, no. June,

[17] L Kong, Kinematic Resolutions of Redundant Robot Manipulators using Integration-Enhanced RNNs.

[18] M Karlsson, On Motion Control and Machine Learning for Robotic Assembly.

[19] A Tariverdi, V K Venkiteswaran, and M Richter2021, A Recurrent Neural-Network-Based Real-Time Dynamic Model for Soft Continuum Manipulators, vol. **8**, no. March, pp. 1–19,

[20] C Gao, S Member, A Rios-navarro, X Chen, and S Member2020, EdgeDRNN : Recurrent Neural Network Accelerator for Edge Inference, vol. **10**, no. 4, pp. 419–432.

[21] C Gao, A Rios-navarro, X Chen, T Delbruck, and S Liu2020, EdgeDRNN : Enabling Low-latency Recurrent Neural Network Edge Inference, no. September.

[22] O Russakovsky, J Deng, H Su, J Krause, S Satheesh, S Ma, Z Huang, A Karpathy, C V Jan, J Krause, and S Ma, ImageNet Large Scale Visual Recognition Challenge.

[23] B A Krizhevsky, I Sutskever, and G E Hinton2012, ImageNet Classification with Deep Convolutional Neural Networks,

[24] R Jiménez-moreno, A R Fonseca, and J L Ramírez2020, Object gripping algorithm for robotic assistance by means of deep leaning, vol. **10**, no. 6, pp. 6292–6299,

[25] H Ponce, E Moya-albor, and J Brieva2018, A Novel Artificial Organic Control System for Mobile Robot Navigation in Assisted Living Using Vision- and Neural-Based Strategies, vol. **2018**.

[26] Z Wang, Z Li, B Wang, and H Liu2016, Robot grasp detection using multimodal deep convolutional neural networks, vol. **8**, no. 9, pp. 1–12,

[27] P Wu, Y Cao, Y He, and D Li2017, Vision-Based Robot Path Planning with Deep Learning, vol. **2**, pp. 101–111,

[28] Z Chen, L Zhou, and M Guo2018, Robot Motion Planning Under Uncertain Condition Using Deep Reinforcement Learning, vol. **149**, no. Mecae, pp. 94–100.

[29] H & L Liu, Yi & Cong, Ming & Dong2019, Human skill integrated motion planning of assembly manipulation for 6R industrial robot., *Ind. Robot Int. J. Robot. Res. Appl.*, vol. **46(1)**, **171**,

[30] R O Feb, Real-Time Grasp Detection Using Convolutional Neural Networks.

[31] C Doersch2017, Tutorial on Variational Autoencoders, no. August.

[32] S Fresca, L Dede, and A Manzoni2021, A Comprehensive Deep Learning-Based Approach to Reduced Order Modeling of Non-linear Time-Dependent Parametrized PDEs, *J. Sci. Comput.*, vol. **87**, no. 2, pp. 1–36,

[33] Y Chen, E Saygin, D Earth, I Future, and S Platform2020, Seismic Inversion by Hybrid Machine Learning, pp. 1–36.

[34] S Wang, W Zhang, G Zheng, X Li, and Y Zhao2022, Fault Diagnosis of Wind Turbine Generator with Stacked Noise Reduction Autoencoder Based on Group Normalization,

[35] O Limoyo, F Mari, M Giamou, P Alexson, and I Petrovi, Generative Graphical Inverse Kinematics, pp. 1–17.

[36] B Zhao2009, Modeling pressure drop coefficient for cyclone separators: A support vector machine approach, *Chem. Eng. Sci.*, vol. **64**, no. 19, pp. 4131–4136,

[37] Z Zhou, Yanpeng & Liu, Keping & Li, Chunxu & Wang, Gang & Liu, Yongbai & Sun2021, A Gradient Neural Network for online Solving the Time-varying Inverse Kinematics Problem of Four-wheel Mobile Robotic Arm., *IEEE 10th Data Driven Control Learn. Syst. Conf.*, vol. **00**,

[38] A Alkhodary and B Gur, Kinematics Transformer : Solving The Inverse Modeling Problem of Soft Robots using Transformers.

[39] R F Reinhart and J J Steil2016, Hybrid Mechanical and Data-driven Modeling Improves Inverse Kinematic Control of a Soft Robot, *Procedia Technol.*, vol. **26**, pp. 12–19,

[40] Z Al-qurashi and B D Ziebart2019, Hybrid Algorithm for Inverse Kinematics using Deep Learning and Coordinate Transformation, *2019 Third IEEE Int. Conf. Robot. Comput.*, pp. 377–380,

[41] J Li, C Xu, Z Chen, S Bian, L Yang, and C Lu, HybrIK : A Hybrid Analytical-Neural Inverse Kinematics Solution for 3D Human Pose and Shape Estimation.

[42] W Sohn, A Shafieloo, and D Kumar, Deblurring the early Universe : reconstruction of primordial power spectrum from Planck CMB using image analysis techniques.