





# Inverse Kinematic Solution Using Neural Networks for Multimodal Inputs and Optimization in Constrained Workspace

Saatvik Tammishetty<sup>1</sup> , Vijay Bhaskar Semwal<sup>1</sup> , Yadunath Pathak<sup>1</sup>  and Deepak Joshi<sup>2</sup> 

<sup>1</sup>Department of Computer Science and Engineering, Maulana Azad National Institute of Technology, Bhopal, 462003, India

<sup>2</sup>Centre for Biomedical Engineering, Indian Institute of Technology, Delhi, 110016, India

**Abstract**—Kinematics is a fundamental aspect of robotics that deals with the position and orientation of a robotic system. It comprises forward kinematics (FK) and inverse kinematics (IK). While FK is simple, IK is complex due to the nonlinear nature of the equations. Most existing IK solutions are available for unconstrained environments, but few address constrained environments. Constraining the workspace ensures the operation of the robotic system within a predefined safe zone, avoiding potential collisions, which is crucial for applications where safety is a top priority. This paper proposes a method that enables multimodality in a Neural Network model trained to predict the joint angles for a given configuration in a constrained workspace, allowing it to take inputs not just in the form of CSV files of Cartesian coordinates but also images and text inputs. The multimodal nature expands its potential applications across various disciplines where the ability to interpret information in different formats is valuable. This model generates optimal results for all input types and achieves an accuracy of 99% with minimal error.

**Index Terms**—Robotics, Inverse Kinematics, Sensor & Encoder, Internet of Things (IoT), Neural Networks, Wireless Sensor Network (WSN)

## I. INTRODUCTION

Kinematics encompasses two main components: Forward Kinematics (FK) and Inverse Kinematics (IK). They play a critical role in various applications, from industrial robotic arms[1] in medical[2] and assembly[3] operations, to humanoid robots[4]. This enables precise trajectory planning[5] and execution.

FK deals with the determination of the final position of the end-effector based on the orientation of joints. It provides a mapping from joint space to cartesian space. On the contrary, IK involves determining the joint angles for a given end-effector position. The resolution of IK is crucial for calculating the precise position and orientation of the end-effector of a robot manipulator to accomplish its task. Figure 1 illustrates the relation between FK and IK.

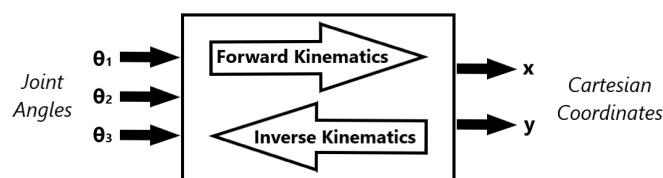


Fig. 1: Relationship between Forward and Inverse Kinematics

The dependence of FK and IK on the manipulator under consideration lies in its unique mechanical and geometric constraints. Our test model is a 3R planar manipulator, consisting of three links and three revolute joints, having with the following constraints.

$$\theta_1 \in [0, \pi] \quad (1)$$

$$\theta_2 \in [-\pi, 0] \quad (2)$$

$$\theta_3 \in [-\pi/2, \pi/2] \quad (3)$$

The workspace of a manipulator refers to the region in space that its end-effector can reach by varying the joint angles within their

permissible range. For a 3R planar manipulator, the workspace is obtained in the cartesian coordinate plane as shown in Figure 2a.

The Confined Work Space is restricted to a rectangular area, as shown in Figure 2b. The final plot must be confined to this region by necessary translation and scaling operations.

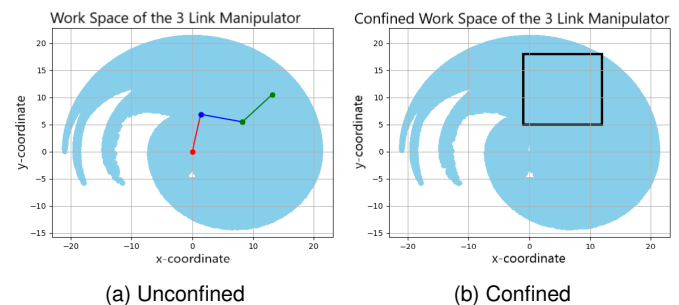


Fig. 2: Work Space of 3-R Planar Manipulator

The solutions to IK are categorized into precise solutions and approximate solutions. A precise solution such as the Analytical Method[6] aims for the exact joint configuration that satisfies the desired position of the end effector. Approximate solutions such as FABRIK[7], Pseudo-Inverse Jacobian Method[8], Newton-Raphson Method[9], Neural Network Approach[10], and Particle Swarm Optimization[11] aim for feasible joint configurations that may not match the desired end-effector position precisely but are close enough for real-life purposes. Incorporation of approximate solutions for IK based machine learning models is beneficial in various ways. The main limitation of precise solutions is that they require extensive calculation and explicit programming for every type of manipulator. Another potential problem associated with these approaches is the non-existence of an inverse solution. This condition is known as singularity[12]. Approximate solutions, on the other hand, enable faster computation, making them more suitable for real-time applications. As real-world robotic systems are prone to uncertainties, these solutions provide the model with the flexibility to handle such cases. Precise solutions may be sensitive to minor variations, leading

Corresponding author: Saatvik Tammishetty  
(e-mail: saatvik.tammishetty@gmail.com)

to instability in the model and difficulty in converging during training. Approximate solutions provide stability to the model, which makes it easier to converge during training. Their capability of identifying the underlying patterns and relationships in the training data contributes to the model's versatility to handle configurations not encountered during training. A Neural Network is well-suited for approximate solutions in its ability to handle complex non-linear relationships between input and output data, resulting in a flexible and adaptable model. This method, being an approximate solution, successfully overcomes the limitations of multiple solutions and singular configurations.

Existing solutions based on the neural network[6] typically address manipulator control in unconstrained environments and rely on coordinate inputs. In contrast, this work presents a significant advancement by developing a neural network model designed to operate in a constrained workspace. Additionally, the multimodal capability of the model enhances its versatility by enabling it to receive input in various formats, such as CSV files of coordinates, images, and text. This advancement not only widens the scope of applicability but also ensures adaptability to a broader range of real-world scenarios, including medical robotics. For instance, this model can guide surgical robots[13] to navigate confined spaces with greater precision, thus improving the results by minimizing the margin for error. By enhancing robotic operations in constrained environments, this model marks significant progress in making IK more flexible, secure, and effective for real-world applications. Integrating multimodal inputs expands the model scope by enhancing its adaptability, distinguishing it from earlier solutions that predominantly depend on coordinate-based data in unconstrained environments.

The workflow of our proposed solution is illustrated in Figure 3. The preprocessing of image input, as described in Algorithm 2 involves detecting black pixels and saving their coordinates, transforming the image into spatial data that the neural network can process. Similarly, text input preprocessing converts user-provided text into an image, allowing the model to interpret it. This standardized approach to handling different input types ensures the reusability of the preprocessing code, enabling the same framework across both image and text inputs. This is described in Algorithm 3.

## II. METHODOLOGY

### A. Forward Kinematics

FK deals with the generation of final position of the end-effector based on the orientation of the joints. It provides a mapping from Joint Space to Cartesian Space. The equations of FK, obtained from the geometric orientation of the manipulator, are as follows.

$$x_e = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \quad (4)$$

$$y_e = l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\theta_1 + \theta_2 + \theta_3) \quad (5)$$

$$\theta = \theta_1 + \theta_2 + \theta_3 \quad (6)$$

Denavit Hartenberg[14] is an alternate method of deriving the equations of FK. It is a more commonly adopted approach when more number of links are contained in the robotic manipulator.

### B. Inverse Kinematics

IK involves determining the joint variables for a given position of the end effector. The resolution of IK is crucial for calculating the precise position and orientation of the end-effector.

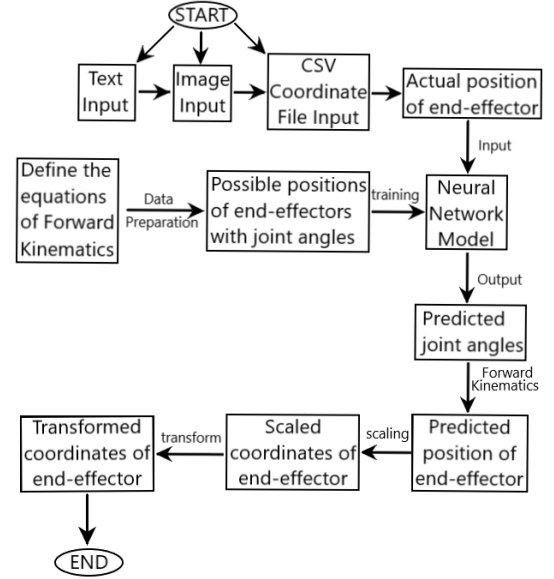


Fig. 3: Proposed Work Flow

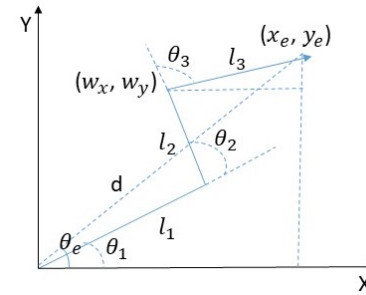


Fig. 4: Orientation of a 3-Link Manipulator

1) *Analytical Approach:* IK solution can be obtained by solving the equations of FK for the joint angles.

From Figure 4, we get the position  $(w_x, w_y)$  as follows:

$$w_x = x_e - l_3 \cos(\theta_e) \quad (7)$$

$$w_y = y_e - l_3 \sin(\theta_e) \quad (8)$$

Now, using the cosine rule,

$$\theta_2 = \cos^{-1} \left( \frac{w_x^2 + w_y^2 - l_1^2 - l_2^2}{2l_1 l_2} \right) \quad (9)$$

From Figure 4, we can also infer the following :

$$w_x = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \quad (10)$$

$$w_y = l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) \quad (11)$$

On solving for  $\cos(\theta_1)$ ,  $\sin(\theta_1)$  we get

$$\cos(\theta_1) = \frac{(l_1 + l_2 \cos(\theta_2))x_e + l_2 \sin(\theta_2)y_e}{x_w^2 + y_w^2} \quad (12)$$

$$\sin(\theta_1) = \frac{(l_1 + l_2 \cos(\theta_2))y_e - l_2 \sin(\theta_2)x_e}{x_w^2 + y_w^2} \quad (13)$$

Now,  $\theta_1$  can be calculated as shown below.

$$\theta_1 = \tan^{-1} \left( \frac{\sin(\theta_1)}{\cos(\theta_1)} \right) \quad (14)$$

Also,

$$\theta_e = \tan^{-1}(y_e/x_e) \quad (15)$$

Substituting the values of  $\theta_e$ ,  $\theta_1$  and  $\theta_2$  in equation 6,

$$\theta_3 = \tan^{-1}(y_e/x_e) - \theta_1 - \theta_2 \quad (16)$$

2) *Neural Network Approach*: Neural networks can be used to solve various types of equations, including the complex nonlinear equations of IK. For this purpose, we have used a feed-forward neural network as represented in Figure 5 and Table 1. It consisting of 3 neurons in the input layer, two hidden layers of 100 neurons each and 3 neurons in the output layer.

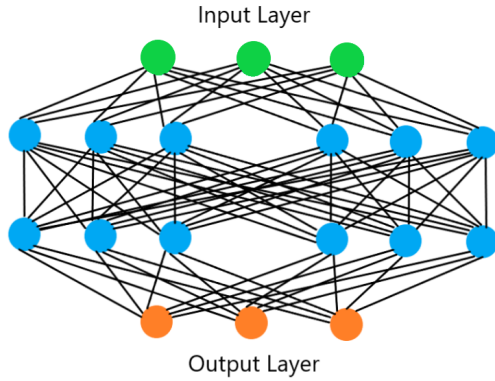


Fig. 5: Proposed Neural Network Architecture

Layer Type	# Units	Activation Function	# Parameters
Dense	3	-	12
Dense	100	ReLU	400
Dense	100	ReLU	10100
Dense	3	Linear	303
Epochs	150		
Learning rate	Default provided by Keras		
Optimizer	Adam		
Loss function	Mean Squared Error		
Regularizer	Early Stopping		

Table 1: Network Architecture and list of Hyperparameters

The dataset consists of 150000 samples of x coordinate, y coordinate, total angle ( $\theta$ ), and joint angles ( $\theta_1, \theta_2, \theta_3$ ). This data is generated through kinematic equations of the three-link manipulator using a random uniform distribution of the joint angles within specified ranges. This data does not require normalization, as the values are within manageable ranges for the model. A small subset of this dataset is shown in Table 2.

x	y	$\theta$
12.46	13.76	7.45
-18.55	5.17	190.79
2.63	11.02	13.75
7.79	3.85	21.2
-13.59	14.63	120.32

Table 2: Sample Training Data

### C. Performance Metrics

Let  $y$  and  $\hat{y}$  represent the actual and predicted values. Based on the results obtained from various experiments, we can evaluate the model performance based on various metrics as follows.

- Mean Squared Error (MSE)

$$MSE(y, \hat{y}) = \frac{\sum_{i=0}^{N-1} (y_i - \hat{y}_i)^2}{N} \quad (17)$$

- Root Mean Squared Error (RMSE)

$$RMSE(y, \hat{y}) = \sqrt{\frac{\sum_{i=0}^{N-1} (y_i - \hat{y}_i)^2}{N}} \quad (18)$$

- Mean Absolute Error (MAE)

$$MAE(y, \hat{y}) = \frac{\sum_{i=0}^{N-1} |y_i - \hat{y}_i|}{N} \quad (19)$$

### D. Proposed Algorithms

The translation and scaling operations needed to restrict and optimize operations in the constrained workspace can be achieved by Algorithm 1.

#### Algorithm 1 Operating and Optimizing in a Constrained Workspace

**Input:** Actual coordinates of a point ( $x, y$ )

**Output:** Target coordinates ( $x_t, y_t$ )

##### Procedure:

**Step 1** Define the boundary of the confined workspace.

- 1)  $x_{bMin}$  represents the lower boundary for x-coordinate.
- 2)  $y_{bMin}$  represents the lower boundary for y-coordinate.
- 3)  $x_{bMax}$  represents the upper boundary for x-coordinate.
- 4)  $y_{bMax}$  represents the upper boundary for y-coordinate.

**Step 2** Find the range of x and y coordinates in the actual coordinates.

- 1)  $x_{min}$  represents the smallest value of x-coordinate.
- 2)  $y_{min}$  represents the smallest value of y-coordinate.
- 3)  $x_{max}$  represents the largest value of x-coordinate.
- 4)  $y_{max}$  represents the largest value of y-coordinate.

**Step 3** Define the scaling factor for x and y coordinates

$$scaling_x = \frac{(x_{bMax} - x_{bMin})}{(x_{max} - x_{min})}$$

$$scaling_y = \frac{(y_{bMax} - y_{bMin})}{(y_{max} - y_{min})}$$

**Step 4** Obtaining the predicted coordinates ( $x_p, y_p$ ) for the actual coordinates.

**Step 5** Scaling the predicted coordinates.

$$x_s = (x_p - x_{min}) * \min(scaling_x, scaling_y)$$

$$y_s = (y_p - y_{min}) * \min(scaling_x, scaling_y)$$

**Step 6** Translation of the scaled coordinates

$$x_t = x_s + x_{bMin}$$

$$y_t = y_s + y_{bMin}$$

Algorithm 2 is used to work with image inputs. It generates a CSV file of coordinates corresponding to the given image, which is useful for systems that require image-based inputs for movement control, such as vision-guided robots.

Algorithm 3 is used to work with text inputs. It generates an image for the given text input, which could be beneficial in systems requiring user inputs in the form of text, such as robots in education.

Angle	$\theta_1$			$\theta_2$			$\theta_3$			$\theta$		
Source	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE
CSV	2.3820	1.5433	1.3751	0.0003	0.0005	0.0004	2.3819	1.5433	1.3751	0.3445	0.5869	0.4524
Image	3.0880	1.7573	1.5712	0.0002	0.0004	0.0003	3.0883	1.7573	1.5712	0.5764	0.7592	0.6522
Text	3.1025	1.7613	1.5565	0.0002	0.0004	0.0003	3.1021	1.7614	1.5535	0.7540	0.8686	0.7483

Table 3: Errors for "ROBOT" Plot

**Algorithm 2** Algorithm for Generating a Coordinate File from an Image

**Input:** An image in JPG or PNG format

**Output:** A CSV file containing the coordinates of black pixels in the image.

**Procedure:**

**Step 1** Read the specified image.

**Step 2** Define the target color in BGR format (0, 0, 0 represents black).

**Step 3** Initialize empty lists x and y to store the coordinates of black pixels.

**Step 4** Get the height and width of the image.

**Step 5** Loop through each pixel in the image:

- Get the color of the current pixel.
- If the color matches the target color (black), append the x and y coordinates to their respective lists.

**Step 6** Create a CSV file for writing the coordinates.

**Step 7** Write each coordinate (x, y) as a new row in the CSV file.

**Step 8** Close the CSV file.

**Algorithm 3** Algorithm for Generating an Image from the Text

**Input:**

- text: The text to be written on the image
- fontSize: The size of the font to be used
- imageWidth: The width of the image
- imageHeight: The height of the image

**Output:** An image corresponding to the text input.

**Procedure:**

**Step 1** Create a blank image with the specified dimensions using the RGB color format, initializing it to white.

**Step 2** Create a draw object to draw on the image.

**Step 3** Load the specified font.

**Step 4** Calculate the width and height of the text using the specified font size.

**Step 5** Calculate the position (x, y) to place the text at the bottom-left corner of the image with a margin.

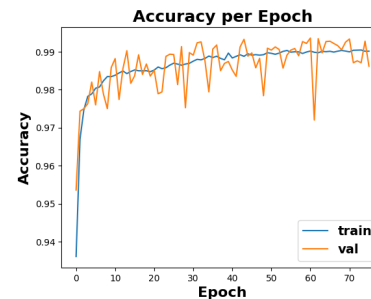
**Step 6** Draw the text on the image at the calculated position using the specified font and fill color.

**Step 7** Save the generated image to the specified output path.

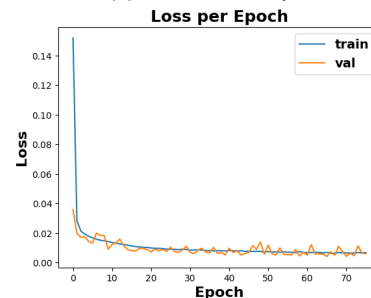
### III. RESULTS AND DISCUSSION

The accuracy and loss curves obtained during the training of the proposed model are presented in Figure 6a and Figure 6b respectively. It converges at an accuracy of 99% and a loss of 0.004.

The model can also be evaluated for various modes of inputs by comparing the evaluation metrics corresponding to the same input in various formats. The plots corresponding to the word "ROBOT" is as shown in Figure 7. The error (in radian) obtained through various modes of inputs are as described in Table 3 and Figure 8.



(a) Model Accuracy



(b) Model Loss

Fig. 6: Plots across Epoch for the Proposed Neural Network Model

### IV. CONCLUSION

Developing a Neural Network model capable of predicting the joint angles in the robotic arm manipulator is a significant achievement. A constrained workspace enables fine-tuning of robotic movements, enhancing accuracy and precision. It is critical for medical and assembly operations, which require high levels of accuracy. By constraining the workspace, we can ensure the operation of the robotic system in the predefined safe zone, avoiding potential collisions, which is crucial for those applications where safety is of top priority. Certain operations require operating in a specific subset of the overall workspace. Here, constrained optimization ensures the optimization of robotic movements, thus reducing energy consumption. Hence, it is also critical for the development of energy-efficient systems. Also, the multi-modal nature of our model expands its potential applications across multiple disciplines.

However, there are a few limitations that must be considered. First, the model's reliance on multimodal inputs may pose challenges in ensuring robust performance if any input modalities are noisy. This could impact real-world deployment as the model may occasionally encounter noisy data. Second, the model's adaptability to variable and unseen constrained workspaces has not been thoroughly tested, which could limit its generalizability to diverse applications. Lastly, the computational overhead of processing multimodal inputs and running neural network models in real time may be a limiting factor for applications with low latency requirements. Addressing these limitations will enhance the model's practicality and scalability.



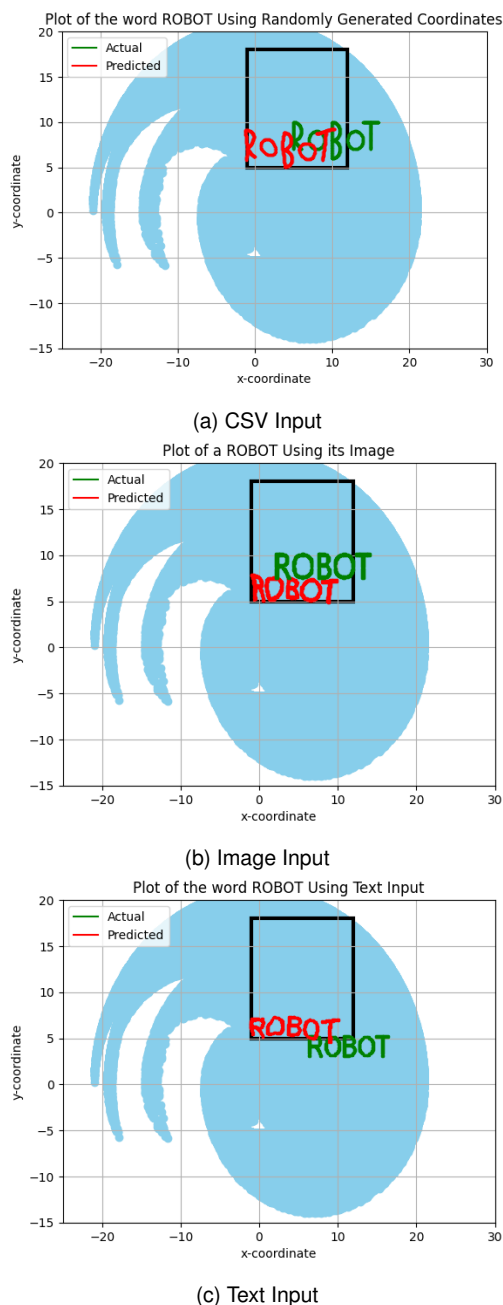


Fig. 7: Plot of the word "ROBOT"

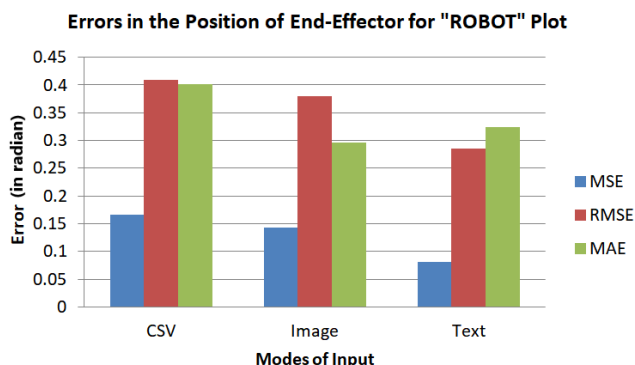


Fig. 8: Visualisation of Errors for "ROBOT" Plot

## V. FUTURE SCOPE

The development of a working prototype of a 3-link manipulator involves many steps. The manipulator links need to be 3D-printed with polyvinyl chloride (PVC). These links are joined by serial servo motors. An ESP32-WROOM-32 microcontroller, embedded with an MPU-6050 IMU (Inertial Measurement Unit), would be integrated into a motor driver to control the rotation along the joints. We also plan to incorporate an optical absolute encoder to precisely monitor the manipulator's rotation, while collision-free trajectories will be achieved using a LIDAR sensor. The encoder will be connected to the joints, providing real-time feedback on angular position, thus enhancing the movement predictions. Meanwhile, the LIDAR sensor will scan the environment to detect obstacles. By combining data from both, the system can dynamically adjust the manipulator's trajectory in real-time, improving accuracy and environmental awareness.

## REFERENCES

- [1] P. D. H. J.-B. Y. T. C. S.-g. K. S. K. H. and L. Y., "Development of an autonomous cleaning robot with a hydraulic manipulator arm for the cleaning of niche areas of a ship hull," 2023.
- [2] C. F. Pană, V. M. Rădulescu, D. D. Voicilă, D. M. Pătrașcu-Pană, and I. C. Reșceanu, "Position control of active arms of da vinci robotic surgical system," in *2023 24th International Carpathian Control Conference (ICCC)*, pp. 321–325, 2023.
- [3] B. Kristyanto, B. B. Nugraha, A. K. Pamosoaji, and K. A. Nugroho, "Analysis of human arm motions at assembly work as a basic of designing dual robot arm system," in *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pp. 1316–1320, 2017.
- [4] V. B. Semwal and G. C. Nandi, "Generation of joint trajectories using hybrid automate-based model: a rocking block-based approach," *IEEE Sensors Journal*, vol. 16, no. 14, pp. 5805–5816, 2016.
- [5] A.-V. Duka, "Neural network based inverse kinematics solution for trajectory tracking of a robotic arm," *Procedia Technology*, vol. 12, p. 20–27, 12 2014.
- [6] V. B. Semwal and Y. Gupta, "Performance analysis of data-driven techniques for solving inverse kinematics problems," in *Intelligent Systems and Applications: Proceedings of the 2021 Intelligent Systems Conference (IntelliSys) Volume 1*, pp. 85–99, Springer, 2022.
- [7] A. Aristidou and J. Lasenby, "Fabrik: A fast, iterative solver for the inverse kinematics problem," *Graphical Models*, vol. 73, no. 5, pp. 243–260, 2011.
- [8] Y. Cao, Y. Gan, X. Dai, and J. Cao, "Inverse kinematics of 7-dof redundant manipulators with arbitrary offsets based on augmented jacobian," in *2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 33–42, 2018.
- [9] S. Lee, J. Lee, J. Bang, and J. Lee, "7 dof manipulator construction and inverse kinematics calculation and analysis using newton-raphson method," in *2021 18th International Conference on Ubiquitous Robots (UR)*, pp. 235–238, 2021.
- [10] N. Lathifah, A. N. Handayani, H. W. Herwanto, and S. Sendari, "Solving inverse kinematics trajectory tracking of planar manipulator using neural network," in *2018 International Conference on Information and Communications Technology (ICOIAC)*, pp. 483–488, 2018.
- [11] L. A. Nguyen, H. Danaci, and T. L. Harman, "Inverse kinematics for serial robot manipulator end effector position and orientation by particle swarm optimization," in *2022 26th International Conference on Methods and Models in Automation and Robotics (MMAR)*, pp. 288–293, 2022.
- [12] A. Hasan, H. Al-Assadi, and A. Azlan, *Neural Networks Based Inverse Kinematics Solution for Serial Robot Manipulators Passing Through Singularities*. 04 2011.
- [13] A. Schwarz, A. Mehrfard, G. Amirkhani, H. Phalen, J. H. Ma, R. B. Grupp, A. M. Gomez, and M. Armand, "Uncertainty-aware shape estimation of a surgical continuum manipulator in constrained environments using fiber bragg grating sensors," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5913–5919, 2024.
- [14] H. P. Nurbai, D. Hadian, N. Lestari, K. A. Munastha, H. Mistialustina, and E. Rachmawati, "Performance evaluation of 3 dof arm robot with forward kinematics denavit-hartenberg method for coffee maker machine," in *2022 16th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*, pp. 1–6, 2022.