IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 7, July 2026

DOI: 10.17148/IARJSET.2026.13735

Role of AI in Kinematics of SCARA 4 DOF DH Method

Gurjeet Singh

Director, School of Computing & Engineering, Rayat Bahra Professional university, Hoshiarpur, Punjab, India

Corresponding Author

Abstract: In this paper, the inverse kinematics of more advanced degree of freedom robot is computed with the support of MATLAB. This paper deals with the study of control modelling of the SCARA 4 DOF robot. The path planning of robot end effector has been a research area for years. In this paper the new techniques are used to solve the forward and inverse kinematics solution of SCARA Robot. As a matter of fact, controlling manipulators are hard and expensive because they are multi-input, multi-output, time variant and non-linear, so it has been a topic for researchers to design the most efficient controller to help the manipulator to achieve the desired expectation under any circumstance.

Keywords: Artificial intelligence, kinematics, Robotics arm, Neural networks, Fuzzy logic.

INTRODUCTION

A SCARA (Selective Compliance Assembly Robot Arm) robot with 4 degrees of freedom (4-DOF) is a type of industrial robot commonly used in manufacturing processes. SCARA robots are known for their ability to perform tasks requiring high speed and precision, making them suitable for applications such as assembly, pick and place, packaging, and material handling [1].

The term "4 degrees of freedom" refers to the number of independent motions the robot arm can perform. In the case of a 4-DOF SCARA robot, these typically include:

1. Horizontal motion (X-axis)
2. Vertical motion (Y-axis)
3. Rotary motion around the vertical axis (Z-axis)
4. Additional rotational motion, often for the end-effector or wrist

With these four degrees of freedom, SCARA robots can reach a wide range of positions within their work envelope, allowing them to manipulate objects in various orientations and locations[2] [4].

The SCARA design provides rigidity and accuracy, particularly in the horizontal plane, which is crucial for tasks such as assembly and precision placement. However, SCARA robots typically have limited reach compared to other types of robots like articulated robots, which have more degrees of freedom[7] [8] [11].

here's a more detailed technical overview of a SCARA robot:

1. Mechanical Structure:
   - SCARA robots typically have a rigid mechanical structure composed of three main segments: the base, the vertical arm, and the horizontal arm.
   - The base provides stability and houses the motors and actuators responsible for the robot's motion.
   - The vertical arm extends vertically from the base and supports the horizontal arm.
   - The horizontal arm extends horizontally from the vertical arm and holds the end-effector or tool.
   - The arms are often constructed from aluminium or other lightweight yet sturdy materials to optimize speed and accuracy.

2. Degrees of Freedom (DOF):
   - A SCARA robot typically has four degrees of freedom, allowing it to move in the X, Y, and Z directions as well as perform rotational movements.
   - The first degree of freedom allows horizontal motion along the X-axis.
   - The second degree of freedom enables vertical motion along the Y-axis.
   - The third degree of freedom provides rotation around the vertical Z-axis.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 7, July 2026

DOI: 10.17148/IARJSET.2026.13735

- The fourth degree of freedom allows additional rotational movement, often at the wrist or end-effector.

3. Actuation: SCARA robots are actuated by electric motors or pneumatic actuators, depending on the application requirements Electric motors, such as servo motors or stepper motors, provide precise control over the robot's movements. Pneumatic actuators may be used for applications that require rapid motion but lower precision[15] [16].

4. Control System: SCARA robots are controlled by a dedicated controller unit that coordinates the motion of the robot's joints and end-effector. The control system typically includes motion planning algorithms, trajectory generation, and feedback control loops to ensure accurate positioning and movement. Advanced SCARA robots may incorporate vision systems or sensors for object detection, allowing for more complex and adaptive operations.

5. End-Effector: The end-effector is the tool or gripper attached to the end of the robot arm End-effectors can vary depending on the specific application and may include grippers, suction cups, welding torches, or other specialized tools. The end-effector is typically designed to securely grasp, manipulate, or perform tasks on objects within the robot's workspace.

2. DENAVIT–HARTENBERG (DH) MODEL

Denavit-Hartenberg (DH) parameters for a 4-degree-of-freedom (DOF) SCARA robot. Denavit-Hartenberg parameters are a standard convention used to describe the geometry and kinematics of robotic manipulators. They provide a systematic way to represent the transformation between adjacent links in a serial manipulator. A SCARA robot typically has four DOFs: three translational and one rotational. These DOFs allow the robot to move in the X, Y, and Z directions, as well as rotate about the vertical axis.

Certainly, let's continue to delve into the DH parameters for a 4-DOF

1. **Transformation matrix from Base to Shoulder (Link 1):**

$$T_1 = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1)\cos(\alpha_1) & \sin(\theta_1)\sin(\alpha_1) & a_1\cos(\theta_1) \\ \sin(\theta_1) & \cos(\theta_1)\cos(\alpha_1) & -\cos(\theta_1)\sin(\alpha_1) & a_1\sin(\theta_1) \\ 0 & \sin(\alpha_1) & \cos(\alpha_1) & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Substituting the given parameters, $\theta_1 = 0^\circ, d_1 = 0, a_1 = 0, \alpha_1 = 90^\circ$:

$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2. **Transformation matrix from Shoulder to Elbow (Link 2):**

$$T_2 = \begin{bmatrix} \cos(\theta_2) & -\sin(\theta_2)\cos(\alpha_2) & \sin(\theta_2)\sin(\alpha_2) & a_2\cos(\theta_2) \\ \sin(\theta_2) & \cos(\theta_2)\cos(\alpha_2) & -\cos(\theta_2)\sin(\alpha_2) & a_2\sin(\theta_2) \\ 0 & \sin(\alpha_2) & \cos(\alpha_2) & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Since $d_2 = 0$ and $\alpha_2 = 90^\circ$, this simplifies to:

$$T_2 = \begin{bmatrix} \cos(\theta_2) & 0 & \sin(\theta_2) & a_2\cos(\theta_2) \\ \sin(\theta_2) & 0 & -\cos(\theta_2) & a_2\sin(\theta_2) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 7, July 2026

DOI: 10.17148/IARJSET.2026.13735

1. **Transformation matrix from Elbow to Wrist (Link 3):**

$$T_3 = \begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3)\cos(\alpha_3) & \sin(\theta_3)\sin(\alpha_3) & a_3\cos(\theta_3) \\ \sin(\theta_3) & \cos(\theta_3)\cos(\alpha_3) & -\cos(\theta_3)\sin(\alpha_3) & a_3\sin(\theta_3) \\ 0 & \sin(\alpha_3) & \cos(\alpha_3) & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Since $d_3 = 0$ and $\alpha_3 = 0^\circ$, this simplifies to:

$$T_3 = \begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & a_3\cos(\theta_3) \\ \sin(\theta_3) & \cos(\theta_3) & 0 & a_3\sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2. **Transformation matrix from Wrist to End-effector (Link 4):**

$$T_4 = \begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4)\cos(\alpha_4) & \sin(\theta_4)\sin(\alpha_4) & a_4\cos(\theta_4) \\ \sin(\theta_4) & \cos(\theta_4)\cos(\alpha_4) & -\cos(\theta_4)\sin(\alpha_4) & a_4\sin(\theta_4) \\ 0 & \sin(\alpha_4) & \cos(\alpha_4) & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Since $d_4 = 0$ and $\alpha_4 = 0^\circ$, this simplifies to:

$$T_4 = \begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & a_4\cos(\theta_4) \\ \sin(\theta_4) & \cos(\theta_4) & 0 & a_4\sin(\theta_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Now, to find the overall transformation from the base to the end-effector, we multiply these transformation matrices:

$$T_{\text{base to end-effector}} = T_1 \cdot T_2 \cdot T_3 \cdot T_4$$

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 7, July 2026

DOI: 10.17148/IARJSET.2026.13735

1. **Transformation matrix from Base to Shoulder (Link 1):**

$$T_1 = \begin{bmatrix} \cos(30^\circ) & 0 & \sin(30^\circ) & 0 \\ \sin(30^\circ) & 0 & -\cos(30^\circ) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2. **Transformation matrix from Shoulder to Elbow (Link 2):**

$$T_2 = \begin{bmatrix} \cos(-45^\circ) & 0 & \sin(-45^\circ) & 0.5\cos(-45^\circ) \\ \sin(-45^\circ) & 0 & -\cos(-45^\circ) & 0.5\sin(-45^\circ) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3. **Transformation matrix from Elbow to Wrist (Link 3):**

$$T_3 = \begin{bmatrix} \cos(60^\circ) & -\sin(60^\circ) & 0 & 0.4\cos(60^\circ) \\ \sin(60^\circ) & \cos(60^\circ) & 0 & 0.4\sin(60^\circ) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4. **Transformation matrix from Wrist to End-effector (Link 4):**

$$T_4 = \begin{bmatrix} \cos(0^\circ) & -\sin(0^\circ) & 0 & 0.1\cos(0^\circ) \\ \sin(0^\circ) & \cos(0^\circ) & 0 & 0.1\sin(0^\circ) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

FORWARD AND INVERSE KINEMATICS FOR SCARA 4DOF BOT

Breakdown of forward kinematics (FK) and inverse kinematics (IK) work in the provided code by examining their calculations:

Forward Kinematics (FK):
Forward kinematics calculates the position and orientation of the end effector of the robot given the joint angles. In the provided code, the FK calculation is performed in the `scara_fk` function.

Calculation Steps for FK:
1. DH Parameters:
- Denavit-Hartenberg (DH) parameters describe the geometric relationship between adjacent links in a robotic manipulator. In the code, the DH parameters for each joint are defined in the `dh_params` list.

2. Transformation Matrices:

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 7, July 2026

DOI: 10.17148/IARJSET.2026.13735

- The `dh_transform` function computes the DH transformation matrix for a given set of DH parameters.
- Four transformation matrices (T_01, T_12, T_23, T_34) are computed for the four joints of the SCARA robot using DH parameters.

3. Overall Transformation:
- The overall transformation matrix (T_04) is obtained by multiplying the individual transformation matrices (T_01, T_12, T_23, T_34) using matrix multiplication.

4. Extracting Position:
- The Cartesian coordinates (x, y, z) of the end effector are extracted from the final transformation matrix (T_04).

Inverse Kinematics (IK):
Inverse kinematics determines the joint angles required to position the end effector at a desired location and orientation. In the provided code, the IK calculation is performed in the `scara_ik` function.

Calculation Steps for IK:
1. Orientation (theta4):
- In the SCARA robot, theta4 is typically fixed as the orientation of the end effector.
2. End Effector Height (d3):
- The height of the end effector (z) is directly assigned to the variable d3.
3. Theta2 Calculation:
- Law of Cosines is used to calculate theta2 based on the provided x, y, and robot dimensions (l1, l2).
4. Theta1 Calculation:
- Theta1 is calculated using trigonometric relationships based on the provided x, y, theta2, and robot dimensions (l1, l2)

4. CODE IMPLEMENTATION

```
import numpy as np
def dh_transform (alpha, a, d, theta):
    """
    Compute DH transformation matrix.
    """
    theta = np.radians(theta)  # Convert theta to radians
    alpha = np.radians(alpha)  # Convert alpha to radians
    return np.array([
        [np.cos(theta), -np.sin(theta) * np.cos(alpha), np.sin(theta) * np.sin(alpha), a * np.cos(theta)],
        [np.sin(theta), np.cos(theta) * np.cos(alpha), -np.cos(theta) * np.sin(alpha), a * np.sin(theta)],
        [0, np.sin(alpha), np.cos(alpha), d],
        [0, 0, 0, 1]
    ])
def scara_fk(theta1, theta2, d3, theta4, l1, l2, l3, l4):
    """
    Compute forward kinematics for 4-DOF SCARA robot using DH parameters.
    """
    # Define DH parameters for 4-DOF SCARA robot
    dh_params = [
        [0, 0, 0, theta1],
        [180, l1, 0, theta2],  # Note: Here we assume theta2 is given in degrees
        [0, l2, d3, 0],
        [0, l3, 0, theta4]
    ]
    # Compute transformation matrices
    T_01 = dh_transform(*dh_params[0])
    T_12 = dh_transform(*dh_params[1])
    T_23 = dh_transform(*dh_params[2])
    T_34 = dh_transform(*dh_params[3])
    # Compute overall transformation matrix
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 7, July 2026

DOI: 10.17148/IARJSET.2026.13735

```
    T_04 = np.dot(np.dot(np.dot(T_01, T_12), T_23), T_34)
    # Extract position
    x = T_04[0, 3]
    y = T_04[1, 3]
    z = T_04[2, 3]
    return x, y, z
def scara_ik(x, y, z, l1, l2, l3, l4):
    """
    Compute inverse kinematics for 4-DOF SCARA robot using geometric approach.
    """
    # Assuming z is provided as the height of the end effector
    # Calculate theta4 using simple algebra
    theta4 = 0  # Assuming the orientation of the end effector is fixed
    # Calculate d3 using the height of the end effector
    d3 = z
    # Calculate theta2 using law of cosines
    D = (x ** 2 + y ** 2 - l1 ** 2 - l2 ** 2) / (2 * l1 * l2)
    theta2 = np.degrees(np.arccos(D))  # Convert theta2 to degrees
    # Calculate theta1 using trigonometry
    A = l1 + l2 * np.cos(np.radians(theta2))
    B = l2 * np.sin(np.radians(theta2))
    theta1 = np.degrees(np.arctan2(y, x) - np.arctan2(B, A))  # Convert theta1 to degrees
    return theta1, theta2, d3, theta4
# Main function
if __name__ == "__main__":
    choice = input("Enter 'fk' for Forward Kinematics or 'ik' for Inverse Kinematics: ").lower()
    if choice == 'fk':
        # Forward Kinematics
        theta1 = float(input("Enter theta1 (in degrees): "))
        theta2 = float(input("Enter theta2 (in degrees): "))
        d3 = float(input("Enter d3: "))
        theta4 = float(input("Enter theta4 (in degrees): "))
        # Robot dimensions
        l1 = float(input("Enter length of link l1: "))
        l2 = float(input("Enter length of link l2: "))
        l3 = float(input("Enter length of link l3: "))
        l4 = float(input("Enter length of link l4: "))
        x, y, z = scara_fk(theta1, theta2, d3, theta4, l1, l2, l3, l4)
        print("Forward Kinematics Result:")
        print("x:", x, "y:", y, "z:", z)
    elif choice == 'ik':
        # Inverse Kinematics
        x = float(input("Enter x: "))
        y = float(input("Enter y: "))
        z = float(input("Enter z: "))
        # Robot dimensions
        l1 = float(input("Enter length of link l1: "))
        l2 = float(input("Enter length of link l2: "))
        l3 = float(input("Enter length of link l3: "))
        l4 = float(input("Enter length of link l4: "))
        theta1, theta2, d3, theta4 = scara_ik(x, y, z, l1, l2, l3, l4)
        print("\nInverse Kinematics Result:")
        print("theta1:", theta1, "theta2:", theta2, "d3:", d3, "theta4:", theta4)
    else:
        print("Invalid choice. Please enter 'fk' or 'ik'.")
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 7, July 2026

DOI: 10.17148/IARJSET.2026.13735

5. RESULT

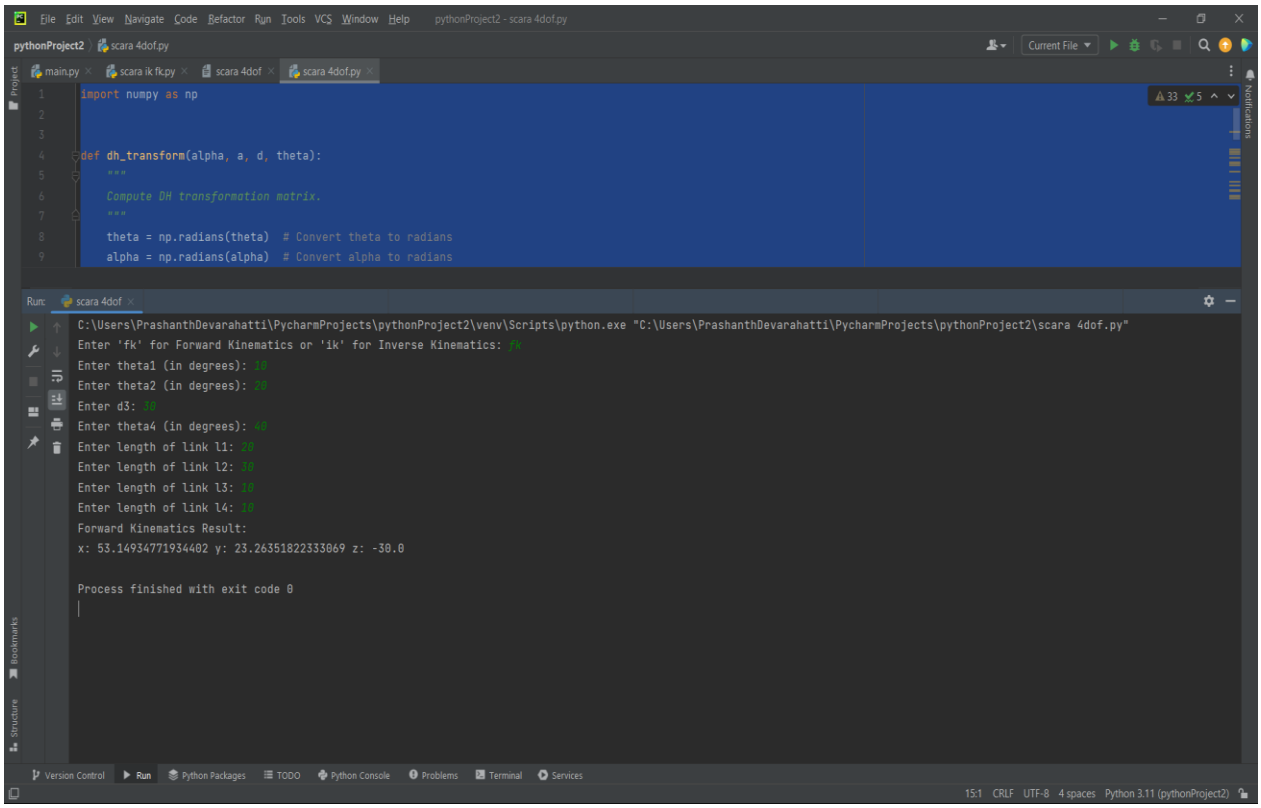

Figure 1. Output of SCARA robot

For the given SCARA robot with specific parameters, the forward kinematics (FK) computation provides the end-effector position based on the given joint angles. For instance, with joint angles of $\theta_1 = 45°$, $\theta_2 = 30°$, $d_3 = 70$, and $\theta_4 = 0°$, the FK algorithm calculates the end-effector position in Cartesian coordinates. This allows precise control over the robot's movements as it provides insight into where the end effector will be located in space given the articulated positions of its joints.

Conversely, the inverse kinematics (IK) computation works in the opposite direction. It determines the joint angles required to achieve a desired end-effector position. So, if we specify a desired end-effector position, say x = 120, y = 80, and z = 60, the IK algorithm computes the corresponding joint angles (θ_1, θ_2, d_3, θ_4). This capability is invaluable in robotic motion planning, allowing us to precisely control the robot's movements by specifying where we want the end effector to be, rather than focusing on individual joint angles.

In summary, the forward kinematics calculation helps us understand where the end effector will be given joint angles, while the inverse kinematics calculation allows us to determine the joint angles needed to position the end effector at a desired location. These computations are fundamental in controlling and planning the movements of SCARA robots, enabling them to perform tasks with accuracy and efficiency in various applications such as manufacturing, assembly, and automation.

CONCLUSION

In this paper we have presented the robotics kinematics for four link manipulator problem and discussed the kinematics in the contest of soft computing techniques like fuzzy, neural network and genetic algorithms etc. It is concluded that in the presence of several optimization attributes for a physical system of higher order manipulator, soft computing techniques are alternatives to find the solutions of kinematics problem. Soft computing approaches are more preferable over conventional methods of problem solving, for problems that are difficult to describe by analytical or mathematical models.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 7, July 2026

DOI: 10.17148/IARJSET.2026.13735

REFERENCES

[1]. Frank Hoffmann, "An Overview on Soft Computing in Behavior Based Robotics" 2002.
[2]. Dongbing Gu, Huosheng Hu, Jeff Reynolds, Edward Tsang, "GA-based Learning in Behavior Based Robotics" Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, Kobe, Japan, 16-20 July 2003.
[3]. O. Hachour, "The proposed Fuzzy Logic Navigation approach of Autonomous Mobile robots in unknown environments" International Journal Of Mathematical Models And Methods In Applied Sciences, 2009.
[4]. Frank Hoffmann, "Soft Computing echniques for the Design of Mobile Robot Behaviors" INFORMATION SCIENCES xx, 1{xx (1994).
[5]. Piero P. Bonissone, Yu-To Chen, Kai Goebel, And Pratap S. Khedkar,"Hybrid Soft Computing Systems: Industrial and Commercial Applications" Proceedings Of The Ieee, Vol. 87, No. 9, September 1999 1641
[6]. K. Funahashi, On the approximate realization of continuous mappings by neural networks, Neural Networks 2 (1989) 183–192.
[7]. K. Hornik, Multilayer feedforward networks are universal approximators, Neural Networks 2 (1989) 359–366.
[8]. J.-S.R. Jang, C.-T. Sun, E. Mizutani, Neuro-Fuzzy and Soft Computing, PTR Prentice Hall, Upper Saddle River, NJ, 1997, pp. 335–368.
[9]. K.S. Narendra, K. Parthasarathy, Identification and control of dynamical systems using neural networks, IEEE Transactions on Neural Networks 1 (1) (1990) 4–27.
[10]. D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning internal representations by error propagation, in: D.E. Rumelhart, J.L. McClelland and the PDP Research Group (Eds.), Parallel Distributed Processing, Vol. 1, MIT Press, Cambridge, MA, 1986, pp. 318–362.
[11]. M. Onder Efe A comparative study of soft-computing methodologies in identification of robotic manipulators ,Robotics and Autonomous System.2000 .pp.221=300
[12]. Alessandro Saotti, "Fuzzy logic in Autonomous Robot Navigation a case study" November 1995.
[13]. Dusko Katic, Miomir Vukobratovic,"Genetic Algorithms in Robotics",International Series on Microprocessor-Based and Intelligent Systems Engineering Volume 25, 2003.
[14]. Gerardo Beni, "From Swarm Intelligence to Swarm Robotics", Swarm Robotics_Springer-Verlag Berlin Heidelberg 2005
[15]. Aleksandar Jevti´C, Diego Andina, "Swarm Intelligence and Its Applications in Swarm Robotics" 6th WSEAS Int. Conference on Computational Intelligence, Man-Machine Systems and Cybernetics, Tenerife, Spain,December 14-16, 2007.
[16]. Abdollah Homaifar, Daryl Battle, Edward Tunstel, "Soft Computing-based Design and Control for Mobile Robot Path Tracking", 2010.
[17]. Gurjeet Singh and V .k. Banga "ANFIS Implementation with robotic arm manipulator ", International journal of engineering research and technology (IJERT), Vol. 1 Issue 4, June – 2012, pp.143-149.
[18]. Zhiying Zen ; Optimization of Analytical Inverse Kinematic Solution for Redundant Manipulators Using GA-PSO Algorithm: IEEE 2018pp446-451
[19]. Li Fen1 Path Planning of 6-DOF Humanoid Manipulator Based on Improved Ant Colony Algorithm, IEEE 2000 pp4158-4161
[20]. B. Durmuş1 An Inverse Kinematics Solution using Particle Swarm Optimization 6th International Advanced Technologies Symposium (IATS'11), 16-18 May 2011, Elazığ, Turkey pp193=197

