Optimal Control of 6-DOF Robotic Arm using PSO Tested on Hardware Chassis

2025 International Conference on Computing, Intelligence, and Application (CIACON) | 979-8-3315-4411-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/CIACON65473.2025.11189352

Soumyendu Banerjee*, Arnab Ghosh, Sanjay Bhadra, Sougata Das, Srijit Gangopadhyay, Md Jishan Mondal
Departmen. of Electrical Engineering, Institute of Engineering and Management Newtown, University of Engineering and Management, Kolkata, India: 700160
*banerjeesoumyendu@gmail.com

***Abstract*—The effective control of robotic manipulators represents a significant and complex area of research within the robotics industry. This study introduces a six-degree-of-freedom (DOF) robotic manipulator characterized by revolute joints, accompanied by a comprehensive hardware implementation. Initially, a schematic model featuring five revolute joints was developed, and the final position of the end-effector was determined using Denavit–Hartenberg (DH) parameters through forward kinematics. Following the specification of the desired end-effector position in Cartesian coordinates, a feedback control system was designed to accurately position the end-effector at the target coordinates. In this context, a particle swarm optimization algorithm was employed to optimize the joint angles by minimizing the cost function, which is defined as the squared sum of the error between the current and desired end-effector positions. The simulation was conducted using MATLAB, and upon determining the optimal joint angles, these values were implemented in the hardware setup of the manipulator, which utilized five MG996R servo motors for overall control. Additionally, a sixth servo motor was incorporated to manage the end-effector's gripping mechanism, which was operated manually via a joystick. This optimal control strategy for the robotic manipulator, integrated with the hardware setup, enhances the applicability of the proposed research across various sectors within the robotics industry.**

Keywords— Denavit–Hartenberg-parameters, forward kinematics, 6-dof robotic arm, optimal positioning using PSO.

I. INTRODUCTION

A robotic manipulator, commonly referred to as a robotic arm, is a mechanical apparatus characterized by various types of joints, which can be operated either through programming or manual intervention [1]. The primary types of joints that constitute a manipulator arm are the revolute joint, denoted as 'R', and the prismatic joint, denoted as 'P'. The revolute joint facilitates rotary motion, allowing for the adjustment of joint angles to enable the arm's rotation, while the prismatic joint permits changes in longitudinal length, thereby altering the joint's extension. The integration of these joints culminates in the formation of an 'end-effector', which is the terminal component of the robotic arm designed for interaction with the surrounding environment. This interaction enables essential functions such as gripping, picking and placing objects, and pointing. The control of a robotic arm involves positioning the end-effector at a specified coordinate, necessitating the optimization of joint values to achieve the desired final position. Recent advancements in internet technology have led to the development of a web-based interface for robotic arms, as demonstrated by Kadir et al., [2] which allows for user-friendly control from any location globally, utilizing active internet connectivity. A well-designed system should ideally be adaptable to any unspecified robotic model. This challenge was addressed by Tan et al. [3], who introduced a cerebellum-inspired methodology for position-based visual servo control of dual robotic arms with unknown kinematic configurations. Their approach enhances the control of robotic arms by leveraging visual feedback and adaptive strategies, thereby improving system performance even when complete kinematic information is lacking. Additionally, a significant development in robotic arm control for the medical field involves enabling individuals with disabilities to manipulate devices using brain signals. Meng et al. [4] proposed a Brain-Computer Interface (BCI) that utilizes Steady-State Visual Evoked Potentials (SSVEP) to facilitate control of robotic arms for reaching and grasping tasks by individuals with disabilities. Furthermore, Zhou et al. [5] advanced BCI research by enabling three-dimensional arm control, allowing multiple users to operate the arm concurrently, which significantly enhances its applicability in rehabilitation and assistive technologies. Iterative control of the end-effector plays a crucial role in calculating feedback errors, thereby enhancing overall performance under diverse input constraints. Meng et al. [6] introduced an innovative algorithm that significantly advances the learning process and operational efficiency of robotic arms, even in the presence of input signal limitations. Another critical area of research in robotic arm control is trajectory management, which involves planning the end-effector's path while circumventing specific obstacles. Carron et al. [7] developed a machine learning-based approach that enhances trajectory tracking and adaptability in dynamic settings, validated through a data-driven model predictive control framework. Particle Swarm Optimization [8-11] is an iterative approach utilized swarm intelligence to optimize a given cost function in a certain search space using feedback control loop. Rokbani et al. [12] focused on the optimal control of a 5-degree-of-freedom (5DOF) robotic arm utilizing Particle Swarm Optimization (PSO) to address the inverse kinematics challenge. This method targets multiple objectives, thereby improving both the efficiency and precision of inverse kinematics solutions for intricate robotic systems. Mehdi et al. [13] integrated PSO with Lyapunov-based motion and force control techniques to manage model uncertainties in robotic arms, ensuring robust control despite variations in the dynamic model. Tutunji et al. [14] proposed a three-stage PSO-based strategy aimed at optimizing and refining Proportional-Derivative (PD) controllers for robotic arm manipulators.

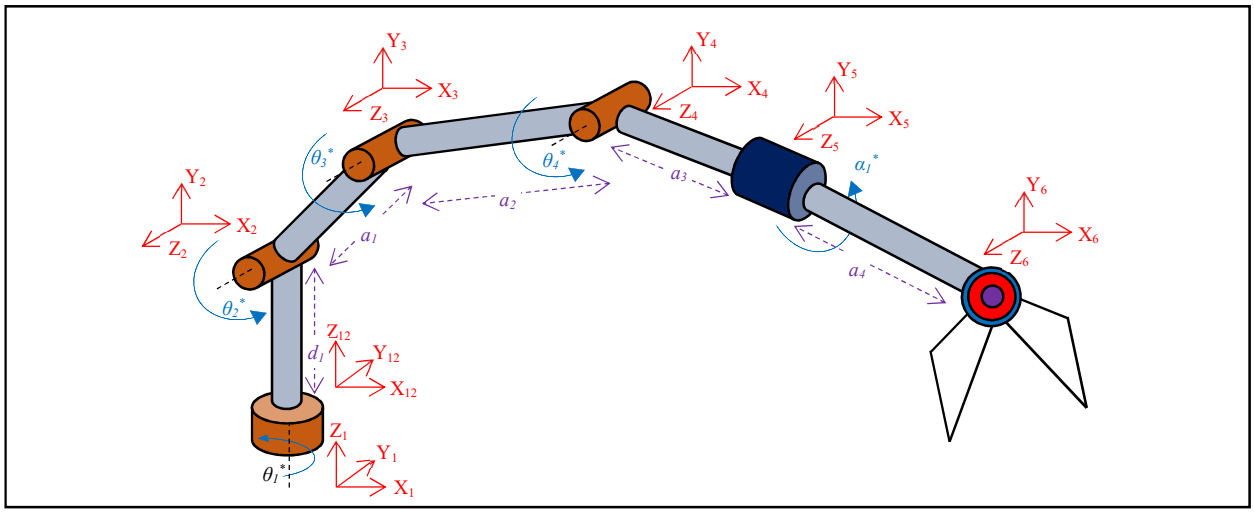

Fig. 1 Schematic diagram 6-DOF Robotic arm with revolute joints

In this study, a six-degree-of-freedom (6 DOF) robotic manipulator was initially developed using MATLAB, drawing inspiration from a comparable hardware configuration constructed with metallic links and MG996R servo motors. The lengths of the links were kept consistent in both the software design and the hardware assembly. The end effector was designed for gripping applications and was manually controlled via a joystick, leading to the optimization of only five joint angles through PSO [15-18]. Utilizing the Denavit–Hartenberg (DH) parameters of the robotic manipulator, the Cartesian coordinates of the end effector in a three-dimensional space were calculated through forward kinematics. Subsequently, the discrepancy between the calculated end-effector position and the target position was minimized using PSO, which identified the global optimum, thereby ensuring the optimal values for the link parameters (joint angles and link lengths for revolute and prismatic joints, respectively) to achieve the desired end-effector positioning. These optimized angles were then transmitted to the servo motors of the hardware setup, validating the effectiveness of the software algorithm in real-time applications. The results section presents a comprehensive analysis of various test outcomes based on arbitrary end-effector positions, including the deviations between the desired and actual results. The novelty of this study is based on the PSO-driven optimization and the placement of the end-effector at the specified coordinates, utilizing a number N of revolute and prismatic joints, thereby facilitating various applications of this algorithm. The remainder of this paper is structured as follows: Section II outlines the methodology and mathematical tools employed in this research, Section III presents experimental results obtained from various robotic manipulators utilizing the proposed algorithm, and Section IV offers concluding remarks.

II. Methodology

This article presents a method for position control of a 6-DOF robotic manipulator with revolute joints, utilizing PSO. The process begins by assigning arbitrary values to the joint angles of the simulated manipulator in MATLAB, alongside a target coordinate in 3D Cartesian space for the end-effector's placement. Following the calculation of the squared sum error between the desired and actual end-effector coordinates, a PSO algorithm was developed to adjust the joint angles. Notably, only five joint angles were modified, as the sixth joint was manually controlled to facilitate the gripping mechanism. The objective was to ensure that, with the optimized joint angle values and fixed link lengths, the end-effector could accurately reach the specified coordinate. Subsequently, a corresponding hardware setup was constructed using servo motors and identical link lengths, where the optimized joint angle values were implemented to align the simulated outcomes with the physical hardware module.

A. Denavit–Hartenberg (DH) Parameter

The spatial interactions between successive links in a robotic arm or manipulator are described by the Denavit-Hartenberg (DH) [1] parameters, which have four different values. These characteristics are essential to robotics because they make it easier to model and analyze the kinematic behavior of robotic systems. They are particularly useful when calculating the forward kinematics of the manipulator, which is used to position the end-effector. These parameters are stated for the *i*-th link as,

1) d_i: Offset or Link length translation from link '*i*-1' to '*i*', along Z_i axis. ;

2) θ_i : Rotational angle between X_{i-1} and X_i , measured across Z_{i-1} axis, representing revolute joint angle.

3) a_i : Offset or Link length translation along X_{i-1} axis link '*i*-1' to '*i*', along X_i axis.

4) α_i :Link twist is rotational angle measured between Z_{i-1} and Z_i, measured across X_{i-1} axis.

The schematic diagram of the robotic manipulator utilized in this study is illustrated in Fig. 1, based on the specified DH-parameters. This manipulator, characterized by a total of five revolute joints, can be classified as an RRRRR type. The diagram clearly indicates that at the initial Frame-1, designated as (X_1, Y_1, Z_1), the first revolute joint, denoted as 'R', is associated with the joint angle ''θ_1*' relative to the intermediate Frame-12, represented as (X_{12}, Y_{12}, Z_{12}), which has an offset length of d_1. Subsequently, a rotation is executed from Frame-12 to Frame-2 with an angle of $\alpha_{12}=90^0$. Following this, Frames-2, 3, 4, and 5 incorporate four variable rotational angles, labeled as θ_2*, θ_3*, α_1*, and θ_4*

respectively. In addition, all the link lengths, associated with each respective frame, are also provided which are fixed and similar to that of actual hardware setup. A concise overview of DH parameters employed in this manipulator is provided in Table I.

B. Forward Kinematics using DH-parameters

The end-effector position was determined through forward kinematics by employing the DH-parameters. In this process, a homogeneous transformation matrix M_i can be constructed for each link utilizing the four distinct matrices outlined below.

$$
{}^{i-1}X_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{i-1}Y_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{i-1}Z_i = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}
$$

$$
{}^{i-1}W_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

These four parameters are used to find out the position (coordinate in 3D Cartesian Space) of i-th frame using coordinate of i-1 th frame. Now, for each frame, independent four parameters were created using DH-parameters, and successive multiplication of these matrices, finally provided the value of end-effector coordinate using following questions.

Using, Frame-ith coordinate, $P_i = (X_i, Y_i, Z_i)$, Frame-i+1th coordinate, $P_{i+1} = (X_{i+1}, Y_{i+1}, Z_{i+1})$ was calculated as,

$$
{}^{i-1}M_i = {}^{i-1}X_i \times {}^{i-1}Y_i \times {}^{i-1}Z_i \times {}^{i-1}W_i
$$

$$
{}^{i-1}M_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ -\sin\theta_i & \cos\theta_i\sin\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}_{(4\times4)}
$$

$$
{}^{i-1}N_i = {}^{i-1}M_i \times \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix}_{(1\times3)} \tag{2}
$$

$$
P_{i+1} = \begin{bmatrix} X_{i+1} \\ Y_{i+1} \\ Z_{i+1} \end{bmatrix} = \begin{bmatrix} {}^{i-1}N_i(1,4) \\ {}^{i-1}N_i(2,4) \\ {}^{i-1}N_i(3,4) \end{bmatrix}_{(1\times3)}
$$

Now, using successive multiplications, end-effector coordinate $P_{n+1} = (X_{n+1}, Y_{n+1}, Z_{n+1})$, can be computed as,

TABLE I. ALL VALUES OF DH-PARAMETRS USED IN THIS WORK, SHOWN IN FIG. 1

Frame	θ	d	a	α
1	θ_1*	d_1	0	0
1-2	0	0	0	90^0
2	θ_2*	0	a_1	0
3	θ_3*	0	a_2	0
4	θ_4*	0	a_3	0
5	0	0	a_4	α_1*
6	90^0	0	0	0

$$
{}^{n-1}N_n = \prod_{i=2}^{n-1} {}^{i-1}M_i \times \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix}_{(1\times3)}
$$

$$
P_n = \begin{bmatrix} X_n \\ Y_n \\ Z_n \end{bmatrix} = \begin{bmatrix} {}^{n-1}N_n(1,4) \\ {}^{n-1}N_n(2,4) \\ {}^{n-1}N_n(3,4) \end{bmatrix}_{(1\times3)} \tag{3}
$$

C. PSO Algorithm based Optimization of End-effector

The primary aim of the proposed study is to position the end-effector at a specified coordinate. By employing forward kinematics and utilizing the initial Frame-1 coordinates established at the origin (0, 0, 0), it is possible to achieve the placement of the end-effector at any desired location through the application of arbitrary joint angles. To enhance the optimization of these joint angles, this research implements a PSO algorithm [18-21]. In each iteration of the PSO process, the squared sum error between the end-effector's coordinates (referred to as the 'candidate solution') and the target coordinates (designated as the 'reference solution') is fed back into the system. This feedback mechanism allows for the simultaneous iterative adjustment of all joint angles, ultimately enabling the end-effector to be accurately positioned within the desired three-dimensional Cartesian space once the target error is achieved.

Initially, random values were assigned to each link variable within predetermined ranges, and these values were input into the forward kinematics equation to calculate the position of the end-effector. Subsequently, the discrepancy between the calculated coordinates and the target coordinates of the end-effector was minimized through an iterative method, ensuring that the squared sum of the error fell below (1×10^{-12}). This process involved the cost function of PSO. '$f_n(\bullet)$' was computed as,

$$
\min_{j\in D}, f_n(q), \text{subject to } \left| pos_j \sim pos_d \right| \le 1\times10^{-14} \tag{4}
$$

The variable 'j' represents the joint angle that was optimized within the designated search space 'D', with distinct search spaces allocated for each variable. Consequently, PSO was employed to refine these values by reducing the discrepancy between the candidate solution, denoted as 'pos_j' (the end-effector position at each iteration), and the target solution, referred to as 'pos_d' (the desired end-effector position). In each iteration, the positions and velocities of the particles were adjusted according to the following equation.

Fig. 2 Hardware setup of 6 DOF Robotic manipulator

$$y_{k+1}^{i} = y_{k}^{i} + v_{k+1}^{i}$$
$$v_{k+1}^{i} = v_{k}^{i} + c_1 r_1 \left(p_{k}^{i} - y_{k}^{i} \right) + c_2 r_2 \left(p_{k}^{g} - y_{k}^{g} \right) \quad (5)$$

where, y_k^i : particle position; v_k^i : particle velocity; p_k^i : best position retained by each particle (p_b); p_k^g : optimal location recalled by the swarm (g_b); r_1, r_2 : random variables between 0 and 1; c_1, c_2: cognitive and social parameters.

This algorithm underwent testing on the proposed manipulator system, which consists of a configuration featuring five revolute (R) joints. The parameters for the PSO were established as follows:

- Maximum number of iteration: 1000
- Number of particle: 50
- Learning rate: 0.99
- Cognitive and social parameter: 2

The search space varied across different frames. For Frame-1, the search space ranged from 0 to 360^0, while the search space for the remaining four joints was limited to a range of 0 to 180^0. With these initial conditions established, the PSO algorithm commenced its execution to determine the optimized joint angles for the first five joints, enabling the end-effector to reach the specified coordinates. To enhance the efficiency of the optimization process, the algorithm was modified to terminate if the cost function fell below 10^{-12}, at which point the current joint angle values would be considered the final optimized values. The cost function was computed using the following equation:

$$f_n(\bullet) = \left| \left(P_{xd} - P_{xj} \right)^2 + \left(P_{yd} - P_{yj} \right)^2 + \left(P_{zd} - P_{zj} \right)^2 \right| \quad (6)$$

where (P_{xj}, P_{yj}, P_{zj}) represents the candidate solution following the j^{th} iteration, while (P_{xd}, P_{yd}, P_{zd}) denotes the target coordinate position of the end-effector along the X, Y, and Z axes, respectively.

(a) (b)

Fig. 3 MG995R and MG996R Servo motor

D. Hardware setup

To realize the proposed project in a hardware module, a six-degree-of-freedom (6-DOF) robotic manipulator was constructed utilizing servo motors. The metallic framework comprises six links and an end-effector equipped with a gripping mechanism, which is operated manually via a joystick and controlled by a MG996R servo motor capable of 180 degrees of rotation. The initial frame incorporates a MG995R servo motor with a 360-degree rotation capability, while the remaining four frames are fitted with MG996R servos. Fig. 2 illustrates the assembled configuration of the robotic manipulator, and Fig. 3 depicts the two types of servo motors employed in this project. The specifications for these motors are detailed in Table II. Additionally, a joystick, as shown in Fig. 4 (explained later), was utilized to manage the gripping mechanism of the end-effector, following the provision of optimized joint angle values to the preceding five servos.

E. Circuit Diagram

The circuit diagram illustrated in Fig. 4 features an Arduino UNO connected to six servos, labeled M1 through M6, where M1 serves as the base motor and M6 functions as the end-effector motor. The motors are linked to the Arduino's digital pins D4 through D9, while the joystick is connected to the analog pins A4 through A9. The entire system is powered by a 5V, 20 Amp DC supply. Initially, the PSO algorithm was implemented using MATLAB software, which generated the angle values for the six revolute joints. These values were subsequently supplied to the servos to facilitate optimal control and assess the effectiveness of the PSO algorithm.

TABLE II. SERVO MOTOR SPECFATIONS

Specifications	**MG995R**	**MG996R**
Rotation type	360° Continuous Rotation	$0\text{-}180^0$ rotation
Dimension	40.7mm ×19.7mm × 42.9mm	
Stall Torque	9.4 kg-cm (4.8V)	
Operating Voltage	4.8V ~ 6.6V	
Gear Type	Metal gear	
Current Draw at idle	10mA	
Stall Current:	1.4A	
Wire Specifications		
Red	Positive	
Brown	Negative	
Yellow	Signal	

TABLE III. EZPERIMENTAL RESULT BASED ON PSO AND HARWARE CHASSIS

Fixed values	Sl. No.	End effector position	Experiment No.	*Variable Joint angles with search space limits*					Error in PSO (10^{-14})	Actual measurements in hardware			
				θ_1 (0-360^0)	θ_2 (0-180^0)	θ_3 (0-180^0)	θ_4 (0-180^0)	α_1 (0-180^0)		Px	Py	Pz	Error
$d_1 = 2$ $a_1 = 10.5$ $a_2 = 15$ $a_3 = 7.5$ $a_4 = 2.5$	1	P_x=14.7 P_y=8.5 P_z=32.2	Exp. 1	209.0378^0	108.8359^0	0^0	31.2109^0	172.8456^0	9.4	14.6	8.4	32.02	0.1
			Exp. 2	209.0378^0	98.2834^0	21.2901^0	14.9083^0	99.5370^0	9.7	14.5	8.3	32.3	0.4
			Exp. 3	29.0378^0	37.3329^0	30.4058^0	0^0	101.2427^0	9.0	15.0	8.6	32.03	0.6
	2	P_x=-4 P_y=-4.7 P_z=4.7	Exp. 1	228.6^0	0^0	132.734^0	172.701	21.2028^0	6.26	-4.1	-4.7	4.67	0.07
			Exp. 2	48.6^0	68.9658^0	102.499^0	126.59	137.5439^0	5.24	-3.96	-4.7	4.8	0.178
			Exp. 3	48.6^0	135.5298^0	150.225^0	180^0	71.3661^0	3.7	-3.99	-4.7	4.64	0.12

F. Result Analysis

The results obtained are presented in Table III. The first column lists the fixed link lengths. For the analysis, two distinct experiments (Sl. No.) were conducted, each featuring different end-effector positions. Under the section titled 'Variable Joint Angles with Search Space Limits,' three separate experiments utilizing PSO-optimized results are detailed for each case. The error associated with the PSO results falls within the range of 10^{-14}, which is considered negligible, as obtained from MATLAB. An experimental result is shown in Fig. 5. This indicates that PSO effectively generates optimized outputs for the same end-effector position, adhering to the specified range of rotational angles, as illustrated in the table. When these values were applied to the hardware setup, the angles were rounded before being assigned to the corresponding servo motors. The actual end-effector positions achieved by the hardware setup are also documented as 'Actual Measurements in Hardware,' with the squared-sum error presented in the final column for each case. Thus, it is evident that the hardware setup can accurately reach the desired coordinates following the application of PSO-optimized outputs. Nonetheless, trajectory planning [22] and path control may also be further developed as a prospective extension of this research.

III. CONCLUSIONS

This study presents a method for optimal control and positioning of the end-effector of a 6-DOF robotic arm utilizing PSO in MATLAB. The angles of the five revolute joints were optimized within defined search parameters, allowing the PSO algorithm to iteratively determine the optimal joint angles necessary to position the end-effector at the desired 3D Cartesian coordinates, achieving a squared-sum error as low as 10^{-14}. Subsequently, these optimized

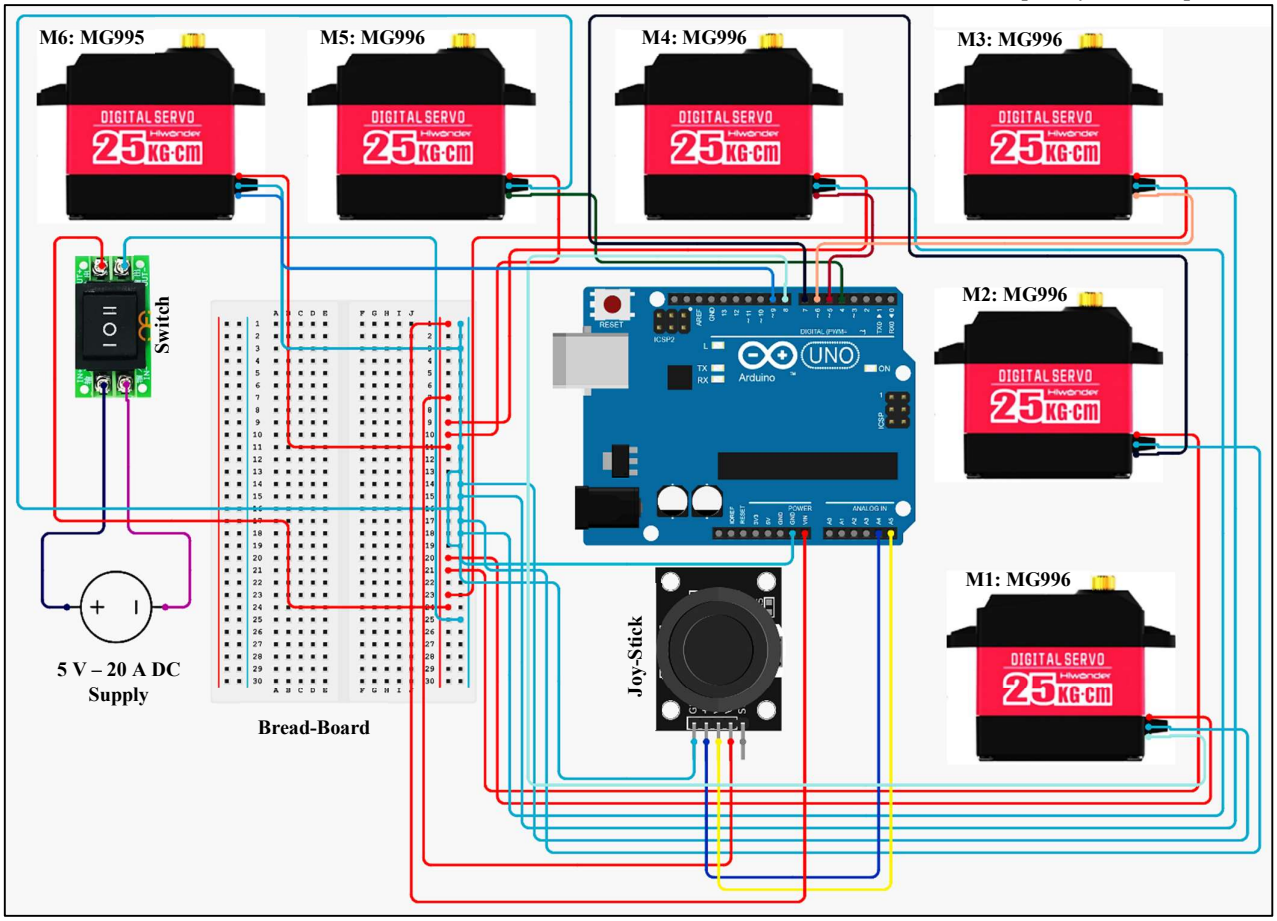

Fig. 4 Circuit diagram of the proposed work using Arduino UNO and six servo motors

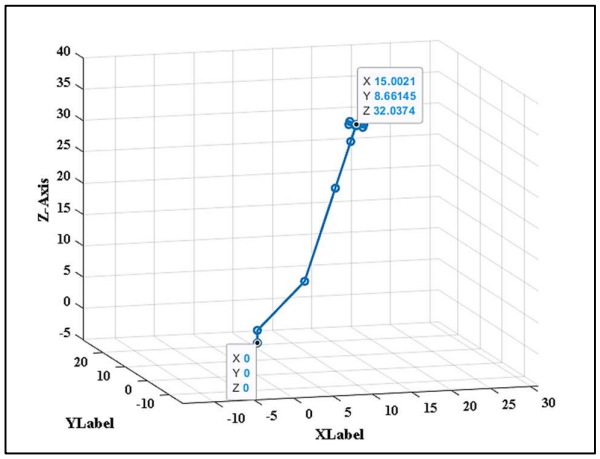

Fig. 5 PSO based optimized result tested on MATLAB, as shown in Sl. No. 1, Exp. 3 in Table III.

angles were transmitted to servo motors connected to a corresponding hardware setup, facilitating synchronization between the software output and the hardware module, which resulted in a squared-sum error ranging from 0.1 to 0.2. This demonstrates effective control of the robotic arm through MATLAB and PSO. The sixth motor was manually controlled via a joystick for the gripping mechanism. It is important to note that PSO operates as an offline approach, which may lead to longer execution times. Future work could involve comparing the performance of PSO with other algorithms such as Ant Colony Optimization (ACO) and Genetic Algorithms (GA).

REFERENCES

[1] Lee, "Robot Arm Kinematics, Dynamics, and Control," Computer, vol. 15, no. 12, pp. 62–80, Dec. 1982.

[2] W. M. H. W. Kadir, R. E. Samin, and B. S. K. Ibrahim, "Internet Controlled Robotic Arm," Procedia Engineering, vol. 41, pp. 1065–1071, 2012.

[3] P. Yu, N. Tan, and M. Mao, "Position-Based Visual Servo Control of Dual Robotic Arms With Unknown Kinematic Models: A Cerebellum-Inspired Approach," IEEE/ASME Transactions on Mechatronics, vol. 28, no. 4, pp. 2328–2339, Jan. 2023.

[4] J. Ai, J. Meng, X. Mai, and X. Zhu, "BCI Control of a Robotic Arm Based on SSVEP With Moving Stimuli for Reach and Grasp Tasks," IEEE journal of biomedical and health informatics, vol. 27, no. 8, pp. 3818–3829, Aug. 2023.

[5] Y. Zhou et al., "Shared Three-Dimensional Robotic Arm Control Based on Asynchronous BCI and Computer Vision," IEEE transactions on neural systems and rehabilitation engineering, vol. 31, pp. 3163–3175, Jan. 2023.

[6] T. Meng and W. He, "Iterative Learning Control of a Robotic Arm Experiment Platform with Input Constraint," IEEE Transactions on Industrial Electronics, vol. 65, no. 1, pp. 664–672, Jan. 2018.

[7] A. Carron, E. Arcari, M. Wermelinger, L. Hewing, M. Hutter, and M. N. Zeilinger, "Data-Driven Model Predictive Control for Trajectory Tracking With a Robotic Arm," IEEE Robotics and Automation Letters, vol. 4, no. 4, pp. 3758–3765, Oct. 2019.

[8] S. Banerjee and G. K. Singh, "Quality Guaranteed ECG Signal Compression Using Tunable-Q Wavelet Transform and Möbius Transform-Based AFD," IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1–11, 2021.

[9] S. Banerjee and G. K. Singh, "Deep neural network based missing data prediction of electrocardiogram signal using multiagent reinforcement learning," Biomedical Signal Processing and Control, vol. 67, p. 102508, May 2021.

[10] S. Banerjee and G. K. Singh, "Quality Aware Compression of Multilead Electrocardiogram Signal using 2-mode Tucker Decomposition and Steganography," Biomedical Signal Processing and Control, vol. 64, p. 102230, Feb. 2021.

[11] S. Banerjee, R. Gupta, and J. Saha, "Compression of Multilead Electrocardiogram Using Principal Component Analysis and Machine Learning Approach," vol. 40, pp. 24–28, Dec. 2018.

[12] N. Rokbani, B. Neji, M. Slim, S. Mirjalili, and R. Ghandour, "A Multi-Objective Modified PSO for Inverse Kinematics of a 5-DOF Robotic Arm," Applied Sciences, vol. 12, no. 14, p. 7091, Jan. 2022.

[13] H. Mehdi and O. Boubaker, "PSO-Lyapunov motion/force control of robot arms with model uncertainties," Robotica, vol. 34, no. 3, pp. 634–651, Jul. 2014.

[14] T. A. Tutunji, Mustafa Al-Khawaldeh, and Malek Alkayyali, "A three-stage PSO-based methodology for tuning an optimal PD-controller for robotic arm manipulators," Evolutionary Intelligence, vol. 15, no. 1, pp. 381–396, Nov. 2020.

[15] S. Banerjee and G. K. Singh, "A new approach of ECG steganography and prediction using deep learning," Biomedical Signal Processing and Control, vol. 64, p. 102151, Feb. 2021.

[16] S. Banerjee and G. K. Singh, "Monte Carlo Filter-Based Motion Artifact Removal From Electrocardiogram Signal for Real-Time Telecardiology System," IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1–10, 2021.

[17] S. Banerjee and G. K. Singh, "A Robust Bio-signal Steganography with Lost-data Recovery Architecture using Deep Learning," IEEE Transactions on Instrumentation and Measurement, pp. 1–1, 2022.

[18] S. Banerjee and G. K. Singh, "A new real-time lossless data compression algorithm for ECG and PPG signals," Biomedical Signal Processing and Control, vol. 79, p. 104127, Jan. 2023.

[19] S. Banerjee and Girish Kumar Singh, "Agent-based beat-by-beat compression of 12-lead electrocardiogram signal using adaptive Fourier decomposition," Biomedical Signal Processing and Control, vol. 75, pp. 103628–103628, Mar. 2022.

[20] S. Banerjee and G. K. Singh, "A New Moving Horizon Estimation Based Real-Time Motion Artifact Removal from Wavelet Subbands of ECG Signal Using Particle Filter," Journal of Signal Processing Systems, vol. 95, no. 8, pp. 1021–1035.

[21] S. Banerjee and G. K. Singh, "AUDSER: Auto-detect and self-recovery reversible steganography algorithm for biological signals," Biomedical Signal Processing and Control, vol. 100, pp. 106974–106974, Oct. 2024.

[22] N. Wu, D. Jia, Z. Li, and Z. He, "Trajectory Planning of Robotic Arm Based on Particle Swarm Optimization Algorithm," Applied Sciences, vol. 14, no. 18, pp. 8234–8234, Sep. 2024