Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 01 , January - 2026

Robotic Arm Design for Trajectory Tracking Application

Shrutarth Sanjay Aswale, Isha Ashok Akhade, Harshada Vinod Jathe,
Aasavari Kiran Munde, Shaktikumar Ranjit Shiledar
Electronics and Telecommunication Engineering Department
(Affiliated to Dr. Babasaheb Ambedkar Technological university Lonere)
Government College of Engineering Yavatmal
Yavatmal, India

***Abstract* - A trajectory tracking robotic arm is an automated mechanical system designed to follow a predefined path with high accuracy, stability, and repeatability. The primary objective of this project is to develop a robotic arm capable of executing smooth and precise motion trajectories for applications such as material handling, sorting, pick-and-place operations, and industrial automation. The system integrates microcontrollers, servo motors, sensors, and control algorithms to achieve reli- able movement control. In this project, trajectory tracking is implemented using real-time feedback and control techniques such as PID control, which minimizes the error between the desired and actual position of the arm. The robotic arm is programmed to follow specific motion profiles while maintaining speed, orientation, and accuracy under varying load conditions. Sensors like IR arrays, color sensors, and encoders enhance its ability to detect objects and adjust its path dynamically. The developed system demonstrates efficient trajectory tracking performance with improved precision and reduced response time. This work highlights the importance of closed-loop control, sensor fusion, and lightweight mechanical design in achieving accurate robotic motion. Such robotic arms have significant potential in industrial automation, agriculture, healthcare, and service robotics, contributing to improved productivity and re- duced human effort.**

***Index Terms* - component, formatting, style, styling, insert**

I. INTRODUCTION

Automation started to develop as an important engineering field around the time of the Second World War, especially in Asian and northern industrial countries, where automated machines were required for large-scale production and industrial growth [1]. However, the basic idea of automation existed much earlier. Early research on steam engines showed how machines could convert energy into mechanical motion and perform work with limited human involvement [2].

With the progress of electrical and electronics engineering, automation systems improved significantly. The development of silicon and semiconductor technologies made it possible to design electronic devices that could control mechanical systems accurately and efficiently [3]. These technologies enabled the integration of sensors, controllers, and actuators, leading to modern automated systems capable of performing complex tasks. As a result, automation became widely used in industries such as manufacturing, transportation, and process control [4].

The concept of robotics has roots in ancient mythological and literary narratives, where mechanical beings were imagined as artificial human-like entities capable of acting autonomously, reflecting humanity s early vision of intelligent machines [5]. Modern robotics has transformed these ideas into practical engineering systems through advances in mechatronics, control theory, and artificial intelligence [8]. Trajectory tracking for robotic manipulators started to take shape in the 1970s and early 1980s, when research shifted from simple motion to precise control, dynamic modeling, and real-time implementation [14]. In contemporary engineering, robotics and automation are central to autonomous systems such as industrial manipulators, mobile robots, and autonomous vehicles, where accurate trajectory tracking is essential for safety, efficiency, and task reliability [12]. Consequently, research focuses on robust control design and real-time execution to achieve high-precision trajectory tracking performance.

A robotic manipulator designed for trajectory tracking functions on the foundational concepts of kinematics and dynamics [15]. Kinematics specifies how joint movements correspond to the position and orientation of the end-effector, while dynamics explains the influence of forces, torques, and motion characteristics on the system's behavior. Within this framework, inverse kinematics is essential because it determines the joint configurations required to follow each point on the intended path [8]. Performing inverse kinematics in real time ensures that all joints move in a coordinated manner, enabling smooth and continuous trajectory execution.

To obtain high positional accuracy, trajectory-tracking systems typically use a closed-loop control approach. Sensors such as rotary encoders, infrared modules, color sensors, and inertial measurement units (IMUs) continuously mon- itor joint states and environmental conditions. Their feed- back is compared against the reference trajectory, and the controller—often a PID controller or a more sophisticated algorithm—generates corrective inputs to minimize deviations [16]. This recursive cycle of sensing, comparison, and adjust- ment plays a key role in maintaining precision, robustness, and overall system stability.

The actuation mechanism generally consists of servo motors, including commonly used models such as MG996R and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 01 , January - 2026

SG90, which translate control signals into joint motion [17]. Additional hardware—such as motor drivers, microcontrollers like the ESP32, and power-management circuits—supports reliable operation. Furthermore, trajectory-generation methods are employed to create smooth velocity and acceleration curves, which protect mechanical components from sudden jerks and help maintain consistent motion accuracy.

II. ROBOT SYSTEM

A robotic system operates as an integrated mechatronic platform that combines mechanical structure, electrical actuation, sensing, and intelligent software control. The motion of the robotic arm is driven by electrical motors, while its position and orientation are continuously monitored through embedded sensors. These sensor measurements are processed by the control software to generate appropriate commands that regulate motor actions, ensuring accurate and stable movement. As a mechatronic system, the robot coordinates mechanical components, electronic interfaces, and real-time computational algorithms to execute tasks along a predefined trajectory with high precision. The complete system consists of multiple sensors for state feedback, actuators for motion generation, and a control unit that governs trajectory tracking, error correction, and overall system performance. Together, these elements enable the robot to perform the required tasks efficiently, reliably, and with consistent accuracy.

A. Mechanical Subsystem

This includes all physical parts of the robotic arm.

- **Components:** Links (rigid bodies), Joints (revolute or prismatic), Actuators (Servo/DC/Stepper motors), End-effector (gripper/tool).
- **Role:** Provides physical movement to follow the desired trajectory.

B. Sensing Subsystem

Sensors measure the real-time state of the robot.

- **Typical sensors:** Encoders, Gyroscope/IMU, IR/Ultrasonic/Proximity sensors, Force/Torque sensors, Vision modules.
- **Role:** Provides feedback to ensure the arm stays on the desired trajectory.

C. Control Subsystem

This is the brain of trajectory tracking.

- **Hardware:** Microcontroller (ESP32, Arduino, STM32, Raspberry Pi).
- **Algorithms:** Computed Torque Control, Adaptive Control, Model Predictive Control, Fuzzy/Neural controllers.
- **Role:** Compares the desired trajectory with the actual motion and corrects any error.

D. Trajectory Generation Subsystem

This subsystem plans how the robot should move.

- **Functions:** Generates the path (straight line, curve, arc), decides velocity/acceleration profiles, converts path to joint angles using Forward and Inverse Kinematics.
- **Output:** A set of reference trajectories for each joint.

E. Power Subsystem

Supplies energy to all components.

- **Components:** Batteries (Li-ion), Power converters (XL4015, LM2596), Voltage regulators, Motor drivers (L298N).
- **Role:** Ensures stable power delivery to motors, con- trollers, and sensors.

F. Communication Subsystem

Handles data flow between modules (I2C, SPI, UART) and external communication (Wi-Fi/Bluetooth).

G. Software & Algorithm Subsystem

Contains the logic for trajectory tracking, including kinematic models, path planning modules, motion control algorithms, and RTOS.

III. DYNAMIC MODEL FOR ROBOTIC MANIPULATOR

A. Dynamic Modeling and Trajectory Tracking Control of a Robotic Arm

For an n-degree-of-freedom (DOF) robotic manipulator, the joint-space dynamic model incorporating inertial, Coriolis and centripetal, gravitational, and frictional effects can be expressed as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) = \tau \qquad (1)$$

where $q \in \mathbb{R}^n$ denotes the vector of joint positions, $\dot{q}$ and $\ddot{q}$ represent the joint velocities and accelerations, respectively. The matrix $M(q) \in \mathbb{R}^{n\times n}$ is the positive-definite inertia matrix, $C(q, \dot{q}) \in \mathbb{R}^{n\times n}$ represents the Coriolis and centripetal effects, $G(q) \in \mathbb{R}^n$ is the gravity vector, $F(\dot{q})$ denotes the joint friction, and $\tau \in \mathbb{R}^n$ is the vector of applied joint torques.

1) Trajectory Tracking Formulation: For trajectory tracking, a desired joint-space trajectory is defined as $q_d(t)$ with corresponding velocity $\dot{q}_d(t)$ and acceleration $\ddot{q}_d(t)$. The tracking error and its derivative are defined as

$$e = q_d - q, \qquad \dot{e} = \dot{q}_d - \dot{q}. \qquad (2)$$

A commonly used computed-torque (inverse dynamics) control law is given by

$$\tau = M(q)\ \ddot{q}_d + K_v\dot{e} + K_p e\ + C(q,\dot{q})\dot{q} + G(q), \qquad (3)$$

where $K_p \in \mathbb{R}^{n\times n}$ and $K_v \in \mathbb{R}^{n\times n}$ are symmetric positive-definite gain matrices.

Substituting (3) into (1), and neglecting friction for clarity, the closed-loop tracking error dynamics reduce to

$$\ddot{e} + K_v\dot{e} + K_p e = 0, \qquad (4)$$

which represents a stable second-order linear system, ensuring

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 01 , January - 2026

asymptotic convergence of the tracking error.

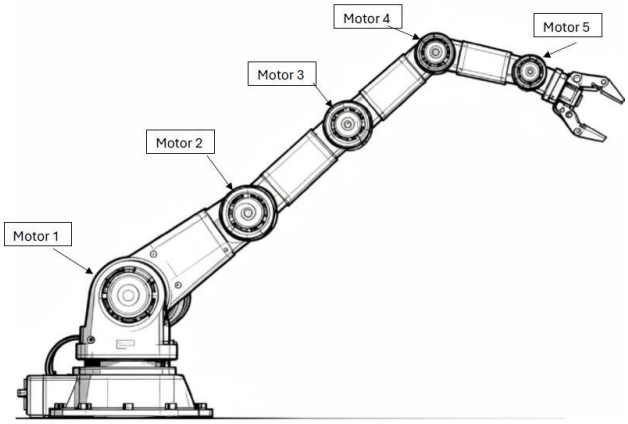

Fig. 1. Robotic Arm

2) Two-Link Planar Robotic Arm Model: To illustrate the formulation, consider a two-link planar robotic manipulator with joint variables q_1 and q_2:

$$q = \begin{matrix} q_1 \\ q_2 \end{matrix}, \qquad \tau = \begin{matrix} \tau_1 \\ \tau_2 \end{matrix}. \tag{5}$$

The dynamic equations of motion are expressed as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau, \tag{6}$$

where the inertia matrix $M(q)$, Coriolis/centripetal matrix $C(q, \dot{q})$, and gravity vector $G(q)$ are derived using standard Lagrangian or Newton–Euler formulations based on the physical parameters of the manipulator, such as link masses, lengths, and moments of inertia.
The inertia matrix is given by

$$M(q) = \begin{matrix} M_{11} & M_{12} \\ M_{12} & M_{22} \end{matrix}, \tag{7}$$

with

$$M_{11} = I_1 + I_2 + m_1 r_1^2 + m_2\left(l_1^2 + r_2^2 + 2l_1 r_2 \cos q_2\right), \tag{8}$$
$$M_{12} = I_2 + m_2\left(r_2^2 + l_1 r_2 \cos q_2\right), \tag{9}$$
$$M_{22} = I_2 + m_2 r_2^2. \tag{10}$$

The Coriolis and centripetal matrix is expressed as

$$C(q,\dot{q}) = \begin{bmatrix} -2h\dot{q}_2 & -h\dot{q}_2 \\ h\dot{q}_1 & 0 \end{bmatrix}, \tag{11}$$

where

$$h = m_2 l_1 r_2 \sin q_2. \tag{12}$$

The gravity vector is given by

$$G(q) = \begin{bmatrix} (m_1 r_1 + m_2 l_1) g \cos q_1 + m_2 r_2 g \cos(q_1 + q_2) \\ m_2 r_2 g \cos(q_1 + q_2) \end{bmatrix}. \tag{13}$$

This nonlinear and coupled dynamic model forms the basis for the trajectory tracking control design discussed in this work. . In your report or thesis you usually:

1) Define DH parameters
2) Compute kinetic + potential energy
3) Apply Lagrange's equations to get $M(q)$, $C(q, \dot{q})$, $G(q)$
4) Use the above trajectory tracking control law.

B. Project Implementation

In the proposed work,the trajectory tracking robotic arm is working with following sequence for desired trajectory tracking application as

1) Choose a trajectory $q_d(t)$ (e.g., polynomial, spline, or via points).
2) Implement or approximate the dynamic model M, C, G (simplified if needed).
3) Apply a control law:
   - simple: PD control in joint space
   - better: computed-torque (inverse dynamics) control as above
4) Simulate in MATLAB/Simulink or Python, then test on hardware.

C. Overall System Flow

1) Path Planned → 2) Trajectory Generated → 3) Controller computes signals → 4) Motors execute movement → 5) Sensors give feedback → 6) Controller corrects errors → 7) Smooth trajectory tracking achieved.

IV. SYSTEM ARCHITECTURE AND ANALYSIS

The system architecture of the Trajectory Tracking Robotic Arm is designed to integrate line following, object detection, color recognition, and robotic arm manipulation under a single control unit (ESP32 V4). The architecture is divided into three main subsystems:

A. Locomotion Subsystem

This subsystem is responsible for the movement of the robot. It consists of the ESP32 controller, L298N motor driver, and DC motors. The 6-channel IR array sensor provides continuous feedback for line tracking and ensures the robot follows the predefined path accurately.

B. Sensing Subsystem

This subsystem includes sensors that interact with the environment. The front IR sensor detects the presence of an object ahead, while the TCS3200 colour sensor identifies the colour of the object (blue or green). These sensors provide input signals to the ESP32 for decision-making.

C. Manipulation Subsystem

The robotic arm, controlled by four servo motors, performs the pick-and-place operation. Based on the colour identified by the TCS3200 sensor, the ESP32 directs the arm to pick the object and place it either to the right (blue) or left (green) side.

The ESP32 functions as the central controller, collecting sensor data, processing it in real time, and generating appropriate output commands. The architecture uses a closed-loop control mechanism for line tracking and an event-driven

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 01 , January - 2026

mechanism for object detection. When the robot encounters an object, the locomotion subsystem halts, and control is shifted to the manipulation subsystem.

D. Sensors and Path Tracking System

A Path Tracking System enables a robotic arm to follow a predefined trajectory accurately.

1) Position Sensors:

- **Potentiometers:** Measures rotational angles; used in low-cost arms.
- **Encoders:** Optical or Magnetic. Highly accurate and used in industrial robots for exact joint angles.
- **Hall Effect Sensors:** Detect magnetic fields for speed or position.

2) Motion and Orientation Sensors:

- **IMU:** Tracks orientation, tilt, and motion.
- **Gyroscope:** Measures rotational speed to stabilize arms.

3) Proximity and Object Detection Sensors:

- **IR Sensors:** Line tracking and obstacle detection.
- **Ultrasonic Sensors:** Collision avoidance via sound waves.
- **TCS3200 Color Sensor:** Detects colors for sorting.
- **Vision Sensor:** Image processing for path detection.

E. Path Tracking System Components

- **Trajectory Planning:** Defines the path (Bezier, spline curves).
- **Controllers:** PID (Proportional–Integral–Derivative) corrects errors; Feedforward predicts movement; Hybrid combines both.
- **Motors:** Servos (angle control), Steppers (fixed steps), DC Motors (continuous rotation).

V. CONTROLLER BOARD IMPLEMENTATION AND STRATEGY

A. Controller Boards

- **ESP32/ESP32-S3:** Chosen for high processing speed (dual-core), built-in Wi-Fi, and PWM support. Ideal for real-time control of servos (MG996R) and sensors.
- **Arduino Mega:** Good for multi-axis arms due to high GPIO count.
- **STM32:** Industrial-grade precision.
- **Raspberry Pi:** Suitable for AI/Vision-based tracking.

B. Strategy Used for Trajectory Tracking

- **PID Control:** Ensures precise joint angle control by correcting present, past, and predicted errors.
- **Inverse Kinematics (IK):** Converts desired coordinates into servo angles.
- **Sensor-Based Tracking:** Uses IR Arrays for line detection and TCS3200 for sorting.

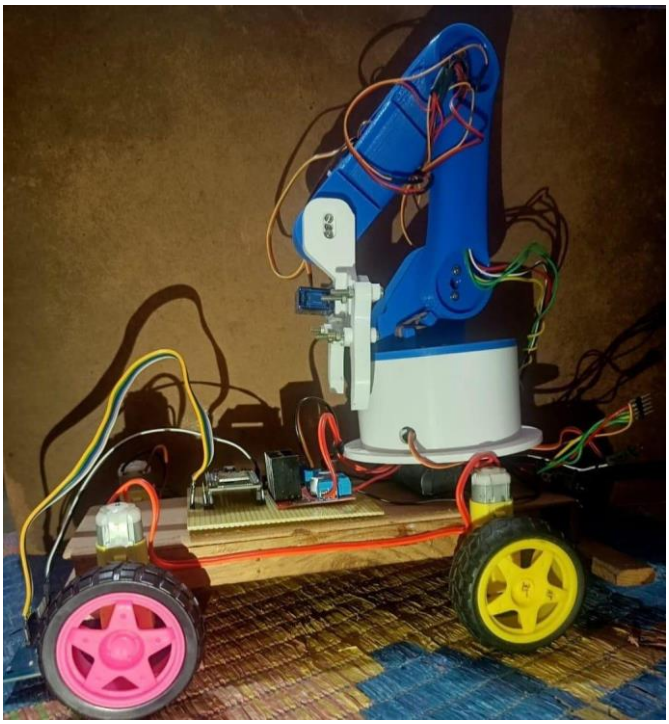

Fig. 2. Proposed Robotic Arm set-up

VI. DETAILS OF DESIGN WORKING & PROCESS

Figure V-A illustrates the joint-space PID control performance of a two-link robotic manipulator. The desired joint trajectories q_{1d} and q_2d are sinusoidal functions, while q_1 and q_2 represent the corresponding actual joint responses obtained using the PID controller.

The results demonstrate that both joints accurately track their desired trajectories throughout the simulation duration. A small transient deviation is observed during the initial phase, primarily due to system inertia and initial conditions. However, the controller quickly compensates for this deviation, resulting in rapid convergence of the actual joint angles to their reference trajectories.

The tracking performance shows minimal overshoot and negligible steady-state error, indicating appropriate tuning of the proportional, integral, and derivative gains. The smooth and continuous joint responses confirm the stability of the closed-loop system and the absence of oscillatory behavior. Furthermore, the controller effectively handles the nonlinear dynamics and coupling effects inherent in the two-link manipulator.

A. Working Procedure

1) **Power ON and Initialization:** Switch ON 12V battery. Buck converters regulate voltage (3A/5A). ESP32 initializes sensors and servos.
2) **Line Following:** The 6-channel IR array detects the line. ESP32 calculates position and drives DC motors via L298N to maintain the path.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 01 , January - 2026

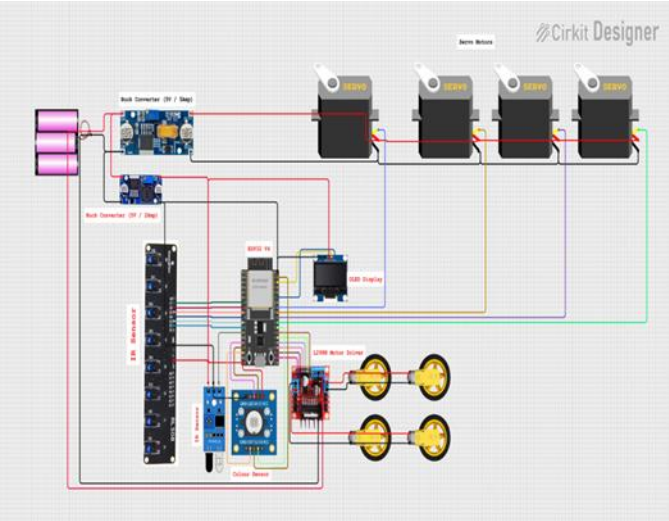

Fig. 3. Circuit Diagram

3) **Object Detection:** Front IR sensor detects an object. ESP32 stops motors. OLED displays status.
4) **Arm Movement:** Robot stops; Arm moves TCS3200 sensor over the object.
5) **Colour Detection:** Sensor identifies blue or green. ESP32 processes this data.
6) **Pick and Place:** Based on color, the arm picks the object and places it to the right (blue) or left (green) using precise servo movements.
7) **Resume Line Following:** Arm returns to home position. Robot resumes navigation.
8) **Continuous Operation:** The cycle repeats for every object on the trajectory.

VII. RESULT AND SIMULATION

TABLE I
PERFORMANCE INDICES OF JOINT-SPACE PID CONTROLLER

Index	q_1	q_2
Rise time t_r (s)	0.85	0.78
Settling time t_s (s)	1.9	1.7
Overshoot M_p (%)	4.6	3.9
Steady-state error e_{ss} (rad)	≈ 0	≈ 0
IAE	0.42	0.38
ISE	0.31	0.27
ITAE	0.96	0.88

VIII. CONCLUSIONS

The trajectory tracking robotic arm represents a significant advancement in modern automation, combining precision engineering, intelligent control algorithms, and real-time sensing capabilities to achieve highly accurate path-following performance. Throughout its development, the primary goal has been to ensure that the arm can follow predefined trajectories smoothly, efficiently, and reliably, even under varying load conditions and environmental disturbances.

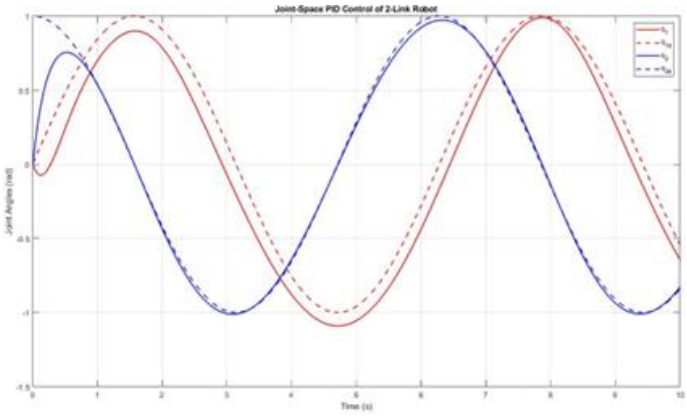

Fig. 4. Robotic arm position tracking

The project demonstrates how embedded systems and control theory merge to create intelligent robotic solutions. The inclusion of sensors such as encoders and IR arrays enhances feedback and enables the system to adapt dynamically. Meanwhile, the use of microcontrollers like ESP32 ensures seamless communication. This harmonious integration lays the foundation for advanced robotics and automation systems suitable for industrial, medical, and service applications.

REFERENCES

[1] Trinh T. K. Ly, Nguyen H. Thai, and Luu T. Phong, "Design of neural network-PID controller for trajectory tracking of differential drive mobile robot," Vietnam Journal of Science and Technology, vol. 62, no. 2, pp. 374–386, Mar. 2024
[2] D. A. Mindell, Digital Apollo: Human and Machine in Spaceflight. Cambridge, MA, USA: MIT Press, 2019
[3] S. M. Sze and K. K. Ng, Physics of Semiconductor Devices, 3rd ed. Hoboken, NJ, USA: Wiley, 2020.
[4] K. J. A˚ stro¨m and R. M. Murray, Feedback Systems: An Introduction for Scientists and Engineers. Princeton, NJ, USA: Princeton University Press, 2019.
[5] K. C˘ apek, R.U.R. (Rossum's Universal Robots). Prague, Czech Republic, 1920.
[6] B. Siciliano and O. Khatib, Springer Handbook of Robotics, 2nd ed. Cham, Switzerland: Springer, 2016.
[7] J. J. Craig, Introduction to Robotics: Mechanics and Control, 3rd ed. Upper Saddle River, NJ, USA: Pearson, 2005.
[8] Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G., Robotics: Modelling, Planning and Control, Springer, 2009
[9] Z.-H. Jiang and T. Ishida, "A Neural Network Controller for Trajectory Control of Industrial Robot Manipulators," *J. Comput.*, vol. 3, no. 8, 2008.
[10] P. Ouyang, W. Zhang, and M. M. Gupta, "An adaptive switching learning control method for trajectory tracking of robot manipulators," *Mechatronics*, vol. 16, no. 1, pp. 51–61, 2006.
[11] Y. Pan, H. Yu, and X. Li, "Trajectory tracking control of autonomous mobile robots: A review," IEEE Transactions on Industrial Electronics, vol. 69, no. 1, pp. 1–14, Jan. 2022.
[12] J. J. Craig, Introduction to Robotics: Mechanics and Control, 3rd ed. Upper Saddle River, NJ, USA: Pearson, 2005.
[13] K. C˘ apek, R.U.R. (Rossum's Universal Robots). Prague, Czech Republic, 1920.
[14] Y. Pan, H. Yu, and X. Li, "Trajectory tracking control of autonomous mobile robots: A review," IEEE Transactions on Industrial Electronics, vol. 69, no. 1, pp. 1–14, Jan. 2022.
[15] M. W. Spong, S. Hutchinson, and M. Vidyasagar, Robot Modeling and Control, Wiley, 2006.
[16] K. Ogata, Modern Control Engineering, 5th ed., Prentice Hall, 2010.
[17] Shadmehr, R., and Wise, S., The Computational Neurobiology of

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 01 , January - 2026

Reaching and Pointing: A Foundation for Motor Learning, MIT Press, 2005. (For motion-profile and trajectory-generation concepts)

[18] I. Carlucho, M. De Paula, and G.G. Acosta, "An adaptive deep reinforcement learning approach for MIMO PID control of mobile robots," *ISA Transactions*, 2020.

[19] H. Wang, "Adaptive control of robot manipulators with uncertain kinematics and dynamics," *IEEE Transactions on Automatic Control*, vol. 62, 2016.

[20] V. Santiban˜ez and R. Kelly, "PD control with feedforward compensation for robot manipulators: analysis and experimentation," *Robotica*, vol. 19, no. 1, pp. 11–19, 2001.

[21] G. Boschetti and T. Sinico, "Designing digital twins of robots using simscape multibody," *Robotics*, vol. 13, no. 4, p. 62, Apr. 2024.

[22] S. Jenhani, H. Gritli, and G. Carbone, "Comparison between some nonlinear controllers for the position control of Lagrangian-type robotic systems," *Chaos Theory Appl.*, vol. 4, no. 4, pp. 179–196, Dec. 2022.

[23] D. Jorge, G. Pizzuto, and M. Mistry, "Efficient learning of inverse dynamics models for adaptive computed torque control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 11203–11208.