एक शतरंज खेलने वाले रोबोटिक आर्म का डिज़ाइन और विकास
Design and Development of a Chess Playing Robotic Arm

प्रोजेक्ट रिपोर्ट

PROJECT REPORT

डिग्री प्रदान किए जाने की आवश्यकता की पूर्ति में प्रस्तुत किया गया

SUBMITTED IN FULFILLMENT OF THE REQUIREMENT
FOR THE AWARD OF THE DEGREE OF

प्रौद्योगिकी स्नातक

यांत्रिक अभियांत्रिकी

BACHELOR OF TECHNOLOGY

MECHANICAL ENGINEERING

प्रस्तुतकर्ता

सुयश तिवारी (20216041)

श्यामल कृष्णन (20213077)

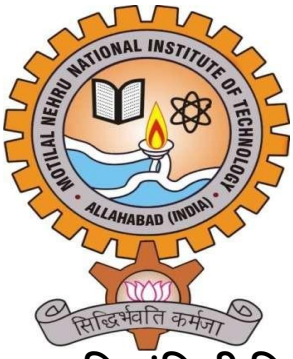
मो. कैफ़ (20213010)

SUBMITTED BY

Suyash Tiwari (20216041)

Shyamal Krishnan (20213077)

Md Kaif (20213010)



यांत्रिक अभियांत्रिकी विभाग

मोतीलाल नेहरू राष्ट्रीय प्रौद्योगिकी संस्थान, इलाहाबाद

प्रयागराज - 211004, (भारत)

Mechanical Engineering Department

MOTILAL NEHRU NATIONAL INSTITUTE OF TECHNOLOGY
ALLAHABAD PRAYAGRAJ - 211004, (INDIA)

मई 2025 May 2025

Candidate's Declaration

We hereby certify that the work which is being presented in the project report entitled "Design and Development of a Chess Playing Robotic Arm" in partial fulfillment of requirements for the award of degree of Bachelor of Technology in Mechanical Engineering Department at MOTILAL NEHRU NATIONAL INSTITUTE OF TECHNOLOGY ALLAHABAD is an authentic record of our work carried out during a period from August 2024 to May 2025 under the supervision of Dr. Samir Saraswati, Professor, MED MNNIT Allahabad. The matter embodied in the thesis has not been submitted to any other University / Institute for the award of any degree.

Signature of the Students
Suyash Tiwari(20216041)

Shyamal Krishnan(20213077)

Md Kaif(20213010)

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Date: Dr. Samir Saraswati
Place: (Professor MNNIT)

ACKNOWLEDGMENT

We would take this moment to express our heartfelt thanks for all the mentorship, motivation, and support that we have received over the period of our project, “Design and Development of a Chess Playing Robotic Arm”. This project has provided us with a very rich experience of learning, and we would not have completed it without the contributions of many individuals and organizations.

I begin by placing on record my deep sense of gratitude to Dr. Samir Saraswati, Professor in the Department of Mechanical Engineering at MNNIT Allahabad, for the constant guidance, helpful criticisms, and insightful suggestions that made the whole exercise possible. His expertize and encouragement inspired me to work with perfection in each phase of the research work.

We sincerely thank Prof. Mukul Shukla, Dr. J. C. Mohanta, Dr. Anurag Deepak, and Dr. Sumit Tiwari for their valuable guidance and constant support throughout our work.

We are also grateful to the staff of the Mechanical Workshop at MNNIT Allahabad for providing essential facilities and technical assistance. Additionally, we appreciate our colleagues for their inspiring discussions that enriched the quality of our work.

Suyash Tiwari (20216041)
Shyamal Krishnan (20213077)
Md Kaif (20213010)

सारांश / Abstract

इस परियोजना, "**चयन और स्थानांतरण करने वाले रोबोटिक आर्म का डिज़ाइन और विकास**" का उद्देश्य एक ऐसा रोबोटिक आर्म बनाना है जो चयन (पिक) और स्थानांतरण (प्लेस) की क्रियाओं को सटीकता और विश्वसनीयता के साथ पूरा कर सके। इस रोबोटिक आर्म का डिज़ाइन एक मॉड्यूलर प्रणाली पर आधारित है, जिससे भविष्य में उन्नयन की संभावनाएं बनी रहें।

इस परियोजना का प्रमुख लक्ष्य एक मजबूत और कुशल प्रणाली तैयार करना था, जो वस्तुओं को संभालते समय मानव क्रियाओं की प्रभावी नकल कर सके। इस लक्ष्य की प्राप्ति के माध्यम से यह रोबोटिक आर्म आगे के विकास चरणों में शतरंज खेलने जैसी उच्च क्षमताओं के प्रदर्शन की नींव रखता है।

यांत्रिक और इलेक्ट्रॉनिक घटकों के समन्वय को सुनिश्चित करने के लिए हमने गहन अध्ययन, सावधानीपूर्वक योजना और कठोर परीक्षण किया। इस प्रक्रिया में कई चुनौतियों का सामना करना पड़ा, जैसे उपयुक्त एक्ट्यूएटर्स का चयन, संरचनात्मक मजबूती सुनिश्चित करना, और सटीक गति के लिए नियंत्रण तंत्र का परिष्करण। इन चुनौतियों को पार कर हमने रोबोटिक्स और प्रणाली डिज़ाइन के क्षेत्र में अमूल्य ज्ञान प्राप्त किया।

This project, “Design and Development of a Pick-and-Place Robotic Arm” aims to build a robotic arm that can execute the operations of pick-and place with precision and reliability. The design of this robotic arm is based on a modular system to provide a basis for future upgrades.

The major goal was to design a robust and efficient system capable of effectively replicating human actions while handling items. By attaining this goal, the robotic arm provides the framework for exhibiting higher capabilities, such as chess play, in later stages of development.

To ensure that mechanical and electronic components were seamlessly integrated, we conducted extensive study, meticulous planning, and rigorous testing. The method faced various obstacles, including choosing the best actuators, assuring structural integrity, and fine-tuning control mechanisms for accurate movement. Overcoming these challenges gave us invaluable insights into robotics and system design.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Robotics integrates mechanical engineering, electronics, and computer science to automate tasks, improve precision, and reduce human effort. Beyond manufacturing, robots now serve in healthcare, homes, and education.

This project, *Design and Development of a Chess-Playing Robotic Arm*, showcases robotics combining mechanical control with intelligent decision-making, reflecting advances in adaptable and autonomous systems.

1.2 Types of Robotic Arms

Robotic arms vary by structure, degrees of freedom, and application:

- **Articulated Arms:** Multi-jointed, flexible; used in welding, painting, assembly.
- **SCARA Arms:** Vertical rigidity, horizontal compliance; suited for fast planar tasks like electronics assembly.
- **Cartesian Arms:** Move along X, Y, Z axes; used in CNC, 3D printing.
- **Cylindrical Arms:** Linear and rotary motion in a cylindrical workspace; applied in machine loading, spot welding.

- **Delta Arms:** Lightweight, high-speed parallel robots; ideal for pick-and-place and sorting.
- **Polar Arms:** Rotating base with telescopic arm; used for heavy handling and die casting.
- **Anthropomorphic Arms:** Human-like dexterity; applied in surgery and prosthetics.
- **Collaborative Arms (Cobots):** Safe for human interaction; used in quality control and collaboration.
- **Hybrid Arms:** Combine features for custom automation and multifunctional tasks.

1.3 Motivation

Chess requires both strategic thinking and precise movement. This project demonstrates a robotic arm combining AI-driven decision-making with accurate manipulation, moving robotics beyond repetitive tasks toward interactive intelligence.

1.4 Objectives

- Build a precise pick-and-place robotic arm.
- Integrate autonomous chess logic.
- Ensure smooth hardware-software integration.
- Design a modular, scalable system.
- Showcase interactive robotics outside industrial use.

Figure 1.1: Articulated Arm – Barrett Technology and MIT, 2004

Figure 1.2: SCARA Arm – shibaura-machine.co.jp

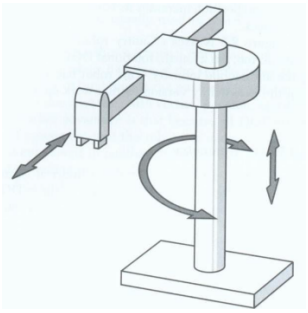


Figure 1.3: Cylindrical Arm – bcon-instruments.nl

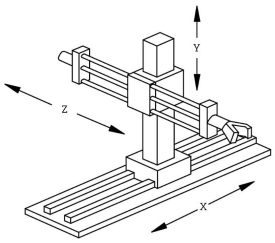


Figure 1.4: Cartesian Arm – iqsdirectory.com

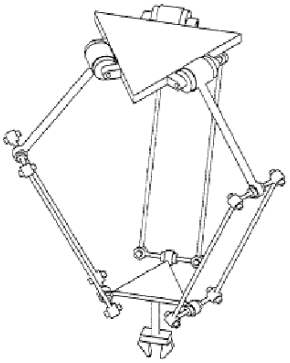


Figure 1.5: Delta Arm – youuvs.com

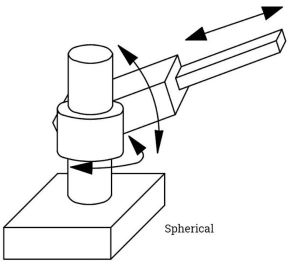


Figure 1.6: Polar Arm – iqsdirectory.com

Figure 1.7: Anthropomorphic Arm – nature.com

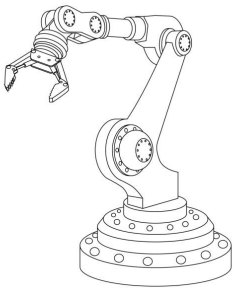


Figure 1.8: Cobot – mingqirobot.en.made-in-china.com

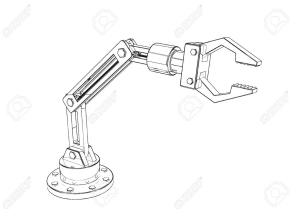


Figure 1.9: Hybrid Arm – istockphoto.com

1.5 Introduction to 3D Printing

The arm was 3D printed using PLA for cost-effective, accurate prototyping and assembled with sensors and actuators.

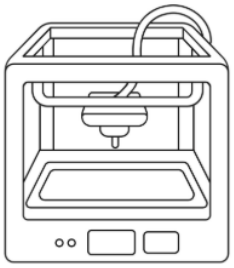


Figure 1.10: 3D Printer at MNNIT Workshop

1.6 Problem Statement

The objective of this project is to develop an affordable robotic arm that integrates precise mechanical design with accurate sensing and control systems to autonomously play chess. The design should focus on creating a robust yet cost-effective manipulator capable of reliably picking and placing chess pieces on a standard chessboard.

Key design challenges include:

- **Mechanical Precision:** Developing a mechanical structure with sufficient degrees of freedom and repeatability to accurately reach and manipulate chess pieces without damaging them or the board.
- **Sensing and Detection:** Implementing reliable sensors (e.g., reed switches, limit switches, or vision systems) to detect piece positions, board state, and confirm successful piece movements.
- **Control Algorithms:** Designing smooth and responsive control algorithms for coordinated motion, including forward and inverse kinematics, to ensure precise and timely execution of chess moves.
- **Affordability:** Selecting cost-effective components (servos, motors, sensors, materials) and manufacturing methods (such as 3D printing)

to maintain a reasonable overall project cost without compromising performance.

- **Integration:** Seamlessly integrating mechanical, electrical, and software components into a unified system that can autonomously play a full game of chess, responding to opponent moves and making legal decisions.

The final design should balance these aspects to create a functional robotic arm capable of autonomous chess gameplay suitable for educational, research, or hobbyist applications.

1.7 Methodology

- CAD design
- Hardware integration
- Motion and chess control software
- Testing and documentation

1.8 Scope & Applications and Future Prospects

Mechanical design, control, and AI for chess-playing robotic arm, with automation and educational uses. Uses in automation, education, AI, healthcare, and home robotics.

Chapter 2

Related Work & Literature Review

Overview

This project explores the design of a chess-playing robotic arm, integrating robotics, AI, and mechanical engineering. It serves as a platform for advancing autonomous decision-making, precise actuation, and human-robot interaction.

Historically inspired by "The Turk," the concept has evolved through AI and robotic developments to modern systems capable of intelligent, physical chess play. Advances in machine learning, computer vision, and actuator technology now enable robots to interact accurately with dynamic environments.

A review of relevant literature provides insight into existing systems, highlighting methodologies for integrating vision, decision-making, and mechanical execution. This background lays the foundation for building an efficient, adaptable robotic arm for chess.

Literature Review

Nguyen Duy Anh, Luong Thanh Nhat, and Tran Van Phan Nhan developed a 3-DoF SCARA robot integrated with a computer vision system capable of detecting chess moves. The system implemented the Minimax algorithm with Alpha-Beta pruning to select optimal moves and executed them using

a PID-controlled servo arm. Their work achieved a 95% accuracy rate, with minor errors attributed to positioning and camera calibration. The robot's modular design and ability to interact with physical chess pieces make it a notable contribution to real-world robotic applications in gameplay.

(See Reference [1] in the References section)

Firas Abdullah Thweny Al-Saedi and Ali H. Mohammed surveyed multiple chess-playing robotic systems, comparing methods such as vision-based move detection and tactile sensor integration. Their research emphasized the use of reed switches for move detection as a low-cost and more reliable alternative to computer vision. This aligns well with the move detection approach used in our project, where reliability and simplicity are prioritized for accurate gameplay and fault tolerance.

(See Reference [2] in the References section)

Prabin Kumar Rath et al. introduced a system combining Stockfish with an overhead camera and a CNC XY-slider mechanism to move magnetic chess pieces. The robot offered wireless control and automatic move logging, with emphasis on replicating professional interaction. Their paper proposed future incorporation of machine learning for adaptive gameplay, which reflects ongoing trends in intelligent autonomous systems and sets the groundwork for advancing robot interactivity in strategic tasks.

(See Reference [3] in the References section)

Conclusion

The literature identifies fundamental considerations for developing a robotic arm to play chess. Adaptable vision-based systems are unreliable due to positioning problems, whereas reed switches are more reliable and cost-effective. Move calculation tools such as Stockfish are established game engines. Drawing on these findings, our project employs reed switches to ensure reliable move detection, Stockfish for decision-making, and a modular robotic arm for adjustability and accuracy—integrating practical merits of each method to construct a reliable, effective system.

Chapter 3

Kinematic Analysis & Trajectory Planning

3.1 Overview

Kinematic analysis enables robotic arm control through Forward Kinematics (FK), Inverse Kinematics (IK), and trajectory planning. FK computes the end-effector pose from joint parameters, IK does the reverse, and trajectory planning ensures smooth motion. This chapter presents FK, IK, and trajectory planning for the chess-playing arm.

3.2 Forward Kinematics

FK determines end-effector position and orientation from known joint values using transformation matrices.

3.2.1 DH Parameters

Each joint is described by:

- θ_i: joint angle
- d_i: link offset
- a_i: link length
- α_i: link twist

Transformation matrix:

$$T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.2.2 Cumulative Transformation

Overall pose:

$$T_{0n} = T_1 \times T_2 \times \cdots \times T_n \tag{3.1}$$

Where each

$$T_i = \begin{bmatrix} R_i & P_i \\ \mathbf{0} & 1 \end{bmatrix} \tag{3.2}$$

- Position: last column $P = [x\ y\ z]^T$
- Orientation: R is the top-left 3×3 matrix

3.2.3 Simulink Implementation

FK was modeled in Simulink to compute end-effector position from joint angles.

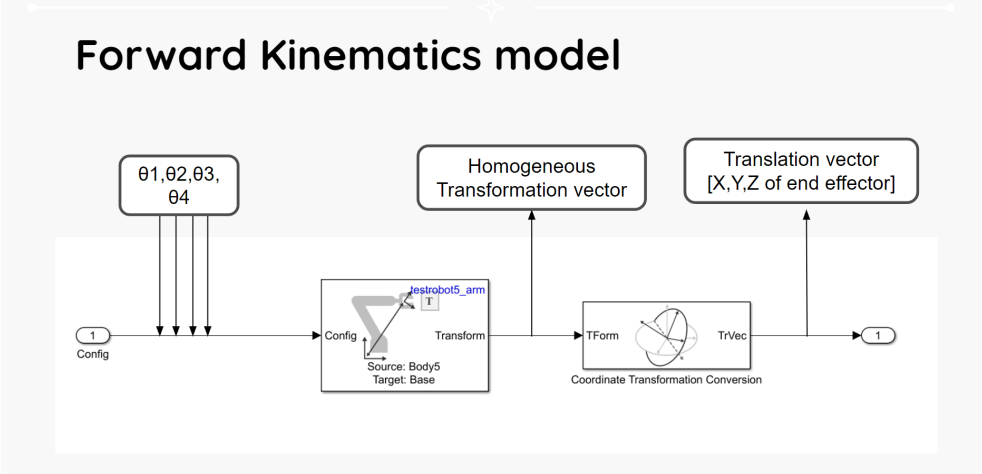


Figure 3.1: Simulink Forward Kinematics Model

Result: The model outputs the Cartesian coordinates and transformation matrix from joint inputs.

3.3 Inverse Kinematics

Inverse Kinematics (IK) calculates joint values needed to reach a desired end-effector pose, essentially reversing Forward Kinematics.

3.3.1 Mathematical Framework

IK solves:

$$T_{0n} = T_1 \cdot T_2 \cdot \cdots \cdot T_n$$

where T_{0n} is the target pose and T_i are individual joint transforms. Solutions can be unique, multiple, or nonexistent.

3.3.2 Solution Methods

- **Analytical:** Closed-form, using geometry (for simpler arms).
- **Numerical:** Iterative (e.g., Levenberg–Marquardt), used here for flexibility and robustness.

3.3.3 Simulink Implementation

IK was modeled in Simulink using numerical solvers to compute joint angles for given end-effector positions.

3.3.4 Challenges

- **Singularities** (unstable Jacobian)
- **Multiple solutions** (pose ambiguity)
- **Convergence issues** (local minima)

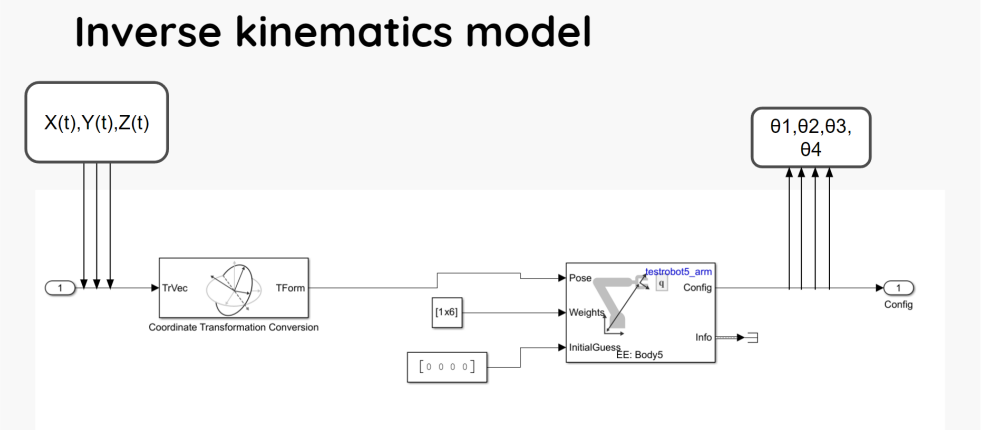


Figure 3.2: Simulink Inverse Kinematics Model

Result: The model computes joint angles accurately within the workspace.

3.4 Trajectory Planning

Trajectory planning defines a time-based path for the robot's end effector to move smoothly and within constraints from start to goal. Unlike path planning, it includes velocity and acceleration profiles.

Key goals are collision avoidance, constraint compliance, and smooth motion, essential in precise tasks like manipulation and CNC machining.

3.4.1 Mathematical Framework

Trajectory is a function mapping time to joint positions:

$$q_i(t) : [0, T] \to \mathbb{R}$$

where $q_i(t)$ is joint i's position at time t, and T is total motion time. Trajectories must:

- Pass through waypoints
- Meet boundary conditions (position, velocity, acceleration)
- Respect robot limits (max velocity, acceleration)

Commonly used smooth functions include cubic splines and Bézier curves.

3.4.2 Trajectory Types

- **Point-to-Point (P2P):** Direct moves, e.g., pick-and-place.
- **Continuous Path (CP):** Follow exact paths, e.g., welding.
- **Time-Optimal:** Fastest movement within limits.
- **Minimum-Jerk:** Smooth motions minimizing jerk.
- **Cyclic:** Repeated trajectories for repetitive tasks.

3.4.3 Generation Techniques

- **Polynomial Interpolation:** Smooth curves through points.
- **Spline Interpolation:** Piecewise polynomials for smoothness.
- **Trapezoidal Velocity Profile:** Acceleration-constant-deceleration phases.
- **Optimal Control:** Minimizes cost under constraints.
- **Sampling-Based Methods:** E.g., RRT for complex path planning.

3.4.4 Simulink Implementation

The project used Simulink with cubic spline interpolation to generate smooth joint trajectories, respecting velocity and acceleration limits. This ensured continuous and smooth transitions in position, velocity, and acceleration.

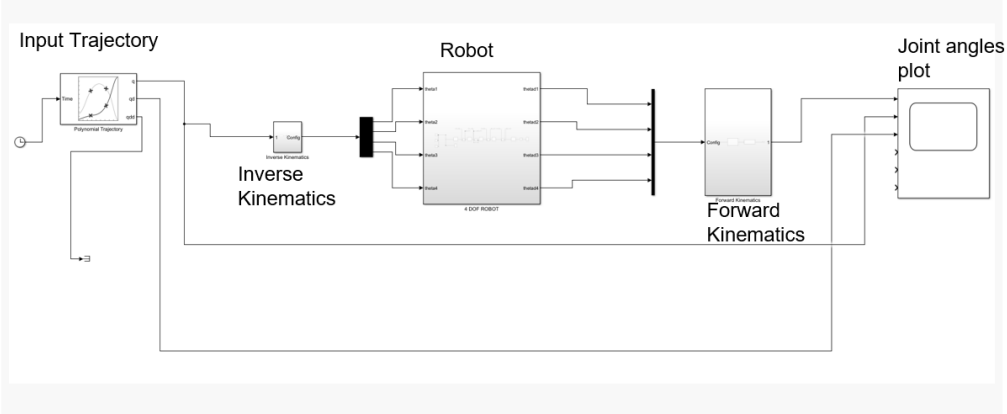


Figure 3.3: Simulink Total Robot Model

Result: The Simulink-based system successfully generated smooth and feasible joint-space trajectories. The robot was able to follow the planned motion without violating any of the specified constraints.

The plots below show the time variation of joint angles, end-effector position, and the resulting 3D trajectory of the robotic arm. The full robot model and simulation environment are discussed in Chapter 5.

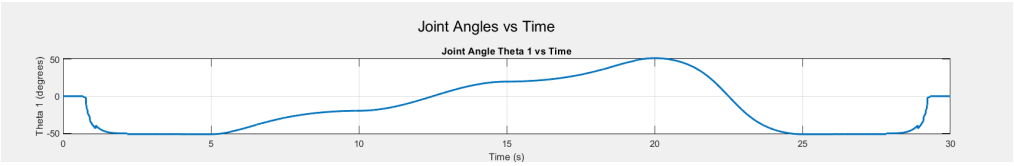


Figure 3.4: Joint Angle θ_1 vs Time - https://www.mathworks.com

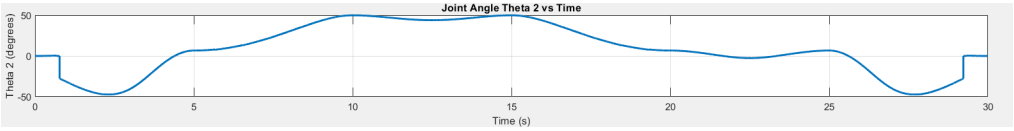


Figure 3.5: Joint Angle θ_2 vs Time - https://www.mathworks.com

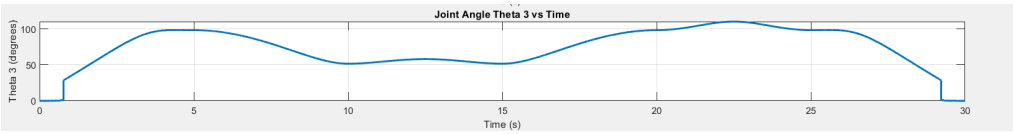


Figure 3.6: Joint Angle θ_3 vs Time - https://www.mathworks.com

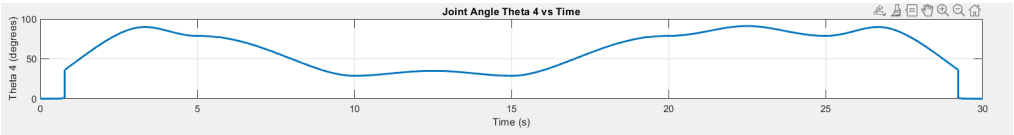


Figure 3.7: Joint Angle θ_4 vs Time - https://www.mathworks.com

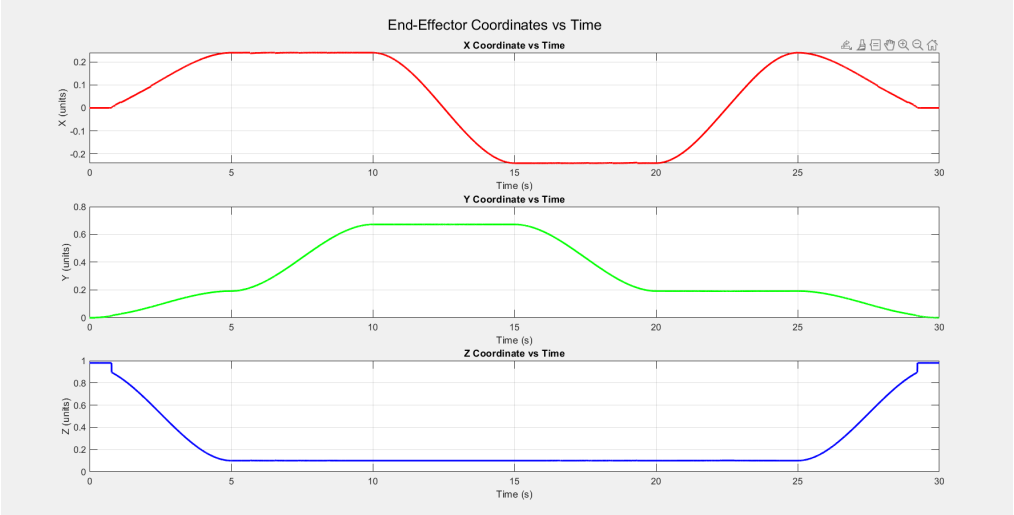


Figure 3.8: End-Effector Position vs Time - https://www.mathworks.com

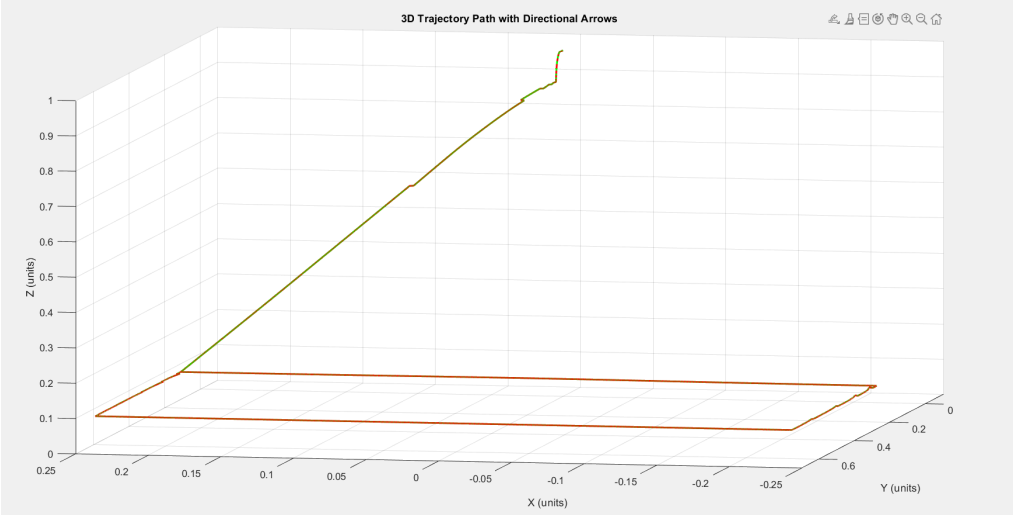


Figure 3.9: 3D Trajectory of End Effector - https://www.mathworks.com

Chapter 4

Hardware

4.1 Overview

This chapter describes the key hardware of the chess-playing robotic arm. The 3D-printed PLA arm has four DOF and uses servo and stepper motors for precise pick-and-place tasks.

An Arduino Uno controls the system, while reed switches improve piece detection using magnetic sensitivity. A stable power supply supports all components, including actuators, sensors, controllers, drivers, mechanical parts, circuits, connectors, and 3D-printed elements.

4.2 Components Classification

This section classifies the main hardware of the chess-playing robotic arm by function: power sources, actuators, sensors, and structural parts.

4.2.1 Power Sources

Figure 4.1: Pro-Range NMC 18650 11.1V 5000mAh 3C 3S2P Li-Ion Battery Pack, source: robu.in

The power supply is a Pro-Range NMC 18650 11.1V 5000mAh Li-Ion Battery Pack, providing sufficient voltage and capacity for long run times and motor/microcontroller demands. Motors are powered directly by 11.1V, while a 6V regulator steps down voltage for the Arduino Uno. Capacitors stabilize the supply, preventing voltage fluctuations during motor state changes. The system is designed to handle varying current draws and peak loads to maintain stable operation.

4.2.2 Actuators

The robotic arm uses a hybrid set of actuators to provide precise, strong, and reliable multi-axis movement for chess gameplay. Motor choices were based on torque, range, response time, and load requirements for each axis. The actuators used include:

- Three Pro-Range OT5330M 7.4V 35.5kg.cm 180° Metal Gear Digital Servo Motors
- Two TowerPro MG995 180° Metal Gear Servo Motors
- One JK42HS48-1684-01 NEMA17 4.4 kg-cm Stepper Motor
- One 60kg DS Servo Motor (8.4W High Torque)

Pro-Range OT5330M 35.5kg.cm High Torque Servo Motors

Pro-Range OT5330M servo motors power the shoulder and elbow joints, providing high torque (35.5 kg.cm at 7.4V) and precision. Their metal gears ensure durability, and the 180° range allows flexible, accurate positioning of the end-effector under varying loads.

These servo motors are pulse-width modulated and were chosen specifically for:

- Their high stall torque enabling stable mid-air chess piece hold.
- Reliable response for inverse kinematics-based trajectory paths.
- Low latency and low jitter essential for pick-and-place operations.

TowerPro MG995 Metal Gear Servo Motors

TowerPro MG995 servo motors, with 10–12 kg.cm torque and 180° rotation, are used in secondary joints like the wrist and gripper tilt. They prioritize faster rotation and precise adjustments over heavy lifting, aiding in fine piece alignment.

Key benefits of using MG995 include:

- Compact size suitable for end-effector regions.
- Economical and widely supported in Arduino environments.
- Moderate torque handling, sufficient for controlling wrist rotation.

JK42HS48-1684-01 NEMA17 Stepper Motor

The NEMA17 JK42HS48-1684-01 stepper motor rotates the robotic arm's base for full chessboard access. It provides 4.4 kg-cm holding torque, 1.8° step angle (200 steps/rev), and uses bipolar stepping for improved torque control.

This motor is controlled via a dedicated A4988 driver module, providing microstepping capability and precise rotation control. The use of a stepper motor in the base rotation provides:

- Stable and accurate angular positioning.
- Reduced cumulative drift compared to servo alternatives.

DS 60kg.cm 8.4W High Torque Servo Motor

A high-torque DS 60kg.cm servo motor (8.4W) is mounted at the gripper base to improve payload handling. It provides the strength needed to securely lift heavier or irregular chess pieces like rooks and queens without slipping or jitter.

Figure 4.2: Pro-Range OT5330M 7.4V 35.5kg.cm Servo Motor, source: robu.in

Figure 4.3: TowerPro MG995 Metal Gear Servo Motor, source: robu.in

Figure 4.4: DS 60kg.cm 8.4W Servo Motor, source: robu.in

The DS60KG motor features:

- Heavy-duty metal gear train for robust long-term operation.
- Efficient heat dissipation and continuous torque delivery.
- Precision digital control for consistent grip angle execution.

Its integration enabled a more consistent pick-and-place performance during extended gameplay scenarios, especially under misalignment conditions or center-of-mass shifts during piece grasping.

Summary and Motor Distribution

Table 4.1: Summary of Motors Used in the Robotic Arm

Motor	**Quantity**	**Functionality**
Pro-Range OT5330M	3	Primary arm joints (shoulder, elbow)
TowerPro MG995	2	Wrist and gripper articulation
NEMA17 JK42HS48-1684-01	1	Base rotation mechanism
DS 60kg.cm 8.4W	1	Gripper lift and heavy grasping

Each motor was selected not only for torque and speed but also for software compatibility, power needs, mechanical integration, and real-world performance. This careful combination of actuators ensures smooth, synchronized, and reliable chess piece handling.

Motor Specifications and Calculations

Motor Specifications and Current Calculations

DS 60kg.cm 8.4W High Torque Servo Motor

- Rated Torque: 60 kg.cm
- Power: 8.4 W
- Voltage: 7.4 V
- Currents: Idle 60 mA, Running 1.2 A, Stall 4.8 A
- Duty Cycle: Idle 20%, Run 75%, Stall 5%
- Average Current: 1.15 A
- Power: 8.52 W

OT5330M 7V 35.5kg.cm Servo Motors (3 units)

- Voltage: 4–8.4 V
- Currents: Idle 20 mA, Running 400 mA, Stall 3 A
- Duty Cycle: Idle 30%, Run 68%, Stall 2%
- Average Current per motor: 0.338 A
- Total Average Current (3 motors): 1.014 A

MG995 Servo Motors (2 units)

- Voltage: 4.8–6.6 V
- Torque: 9.4–11 kg.cm
- Currents: Idle 10 mA, Running 170 mA, Stall 1.2 A

- Duty Cycle: Idle 30%, Run 68%, Stall 2%
- Average Current per motor: 0.143 A
- Total Average Current (2 motors): 0.286 A

NEMA 17 Stepper Motor (1 unit)

- Running Current: 1.5 A per phase
- Stall Current: 2.8 A per phase
- Duty Cycle: Idle 30%, Run 68%, Stall 2%
- Average Current: 1.08 A

Total Current Consumption

$$I_{\text{avg (total)}} = 1.014 + 0.286 + 1.08 = 2.38\,\text{A} \tag{4.1}$$

Including a 30% margin for fluctuations:

$$I_{\text{avg (with margin)}} = 2.38 \times 1.3 = 3.09\,\text{A} \tag{4.2}$$

For 1 hour operation, energy consumption:

$$3.09\,\text{A} \times 1000 = 3090\,\text{mAh} \tag{4.3}$$

Material and Mass Calculation: All robotic arm parts were assumed to be made of **PLA** with a density of $1.25\,\text{g/cm}^3$. Masses were obtained in SolidWorks using the material properties and volume, then used to calculate motor torque requirements.

Torque Calculations: The balancing moments about critical points of the robotic arm were calculated as follows:

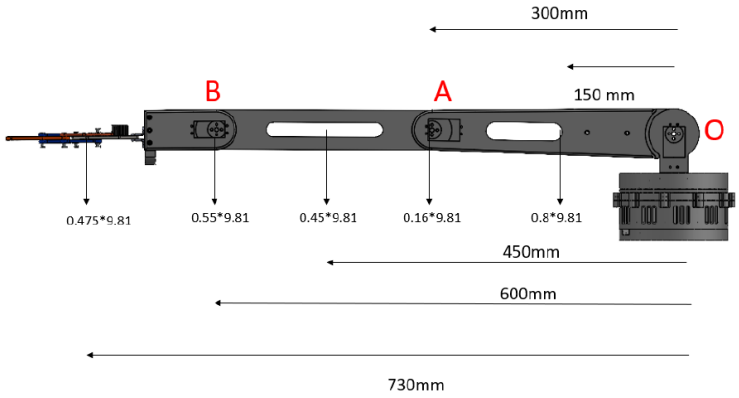


Figure 4.5: Shows the application of Force due to weight

- **Moment about Point O:**

$$\begin{aligned} M_o = (0.8 \times 9.81 \times 0.15) + (0.16 \times 9.81 \times 0.3) \\ +(0.45 \times 9.81 \times 0.45) + (0.55 \times 9.81 \times 0.6) \\ +(0.475 \times 9.81 \times 0.73) \end{aligned} \tag{4.4}$$

$$M_o = 10.27\,\text{Nm} = \frac{10.27 \times 100}{9.81} = 104.68\,\text{kg.cm} \tag{4.5}$$

- **Moment about Point A:**

$$\begin{aligned} M_a = (0.45 \times 9.81 \times 0.15) \\ + (0.55 \times 9.81 \times 0.3) \\ + (0.475 \times 9.81 \times 0.43) \end{aligned} \tag{4.6}$$

$$M_a = 4.28\,\text{Nm} = \frac{4.28 \times 100}{9.81} = 43.63\,\text{kg.cm} \tag{4.7}$$

- **Moment about Point B:**

$$M_b = 0.475 \times 9.81 \times 0.13 \tag{4.8}$$

$$M_b = 0.606\,\text{Nm} = \frac{0.606 \times 100}{9.81} = 6.2\,\text{kg.cm} \tag{4.9}$$

These calculations confirm the suitability of the servo motor for managing the arm's load and movements.

4.2.3 Control Boards and Drivers

Arduino Uno R3 with Cable

The Arduino Uno R3 with Cable acts as the central control unit for the robotic arm system. It processes inputs from various sensors and executes commands to control the actuators. The Arduino Uno is the core component that interprets the movement logic and communicates with other hardware components to perform the required tasks.

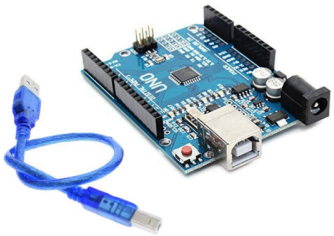


Figure 4.6: Arduino Uno R3 with Cable, source: robu.in

16-Channel 12-bit PWM/Servo Driver I2C Interface PCA9685

The 16-Channel 12-bit PWM/Servo Driver I2C Interface PCA9685 extends the control capabilities of the system, allowing the management of multiple servos simultaneously. It connects to the Arduino through the I2C interface, enabling precise control of servo motors with minimal wiring and ensuring smooth operation of multiple servos at once.

A4988 Stepper Motor Driver Module with Heat Sink

The A4988 Stepper Motor Driver Module, equipped with a heat sink, provides precise control over the stepper motor, which is responsible for rotating the robot base. It controls the stepper motor's position with high accuracy and ensures the arm can reach various points on the chessboard for efficient gameplay.

74HC165 8-Bit Parallel In/Serial Out Shift Register IC

The 74HC165 8-bit Parallel In/Serial Out Shift Register expands the Arduino Uno's I/O by reading multiple reed switches on the chessboard. It enables simultaneous monitoring of 8 squares, allowing efficient detection of chess pieces. This data updates the game state matrix, aiding accurate gameplay decisions.

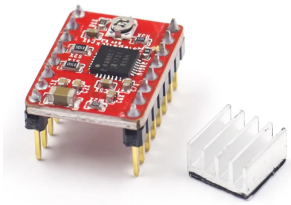


Figure 4.7: A4988 Stepper Motor Driver Module with Heat Sink, source: robu.in

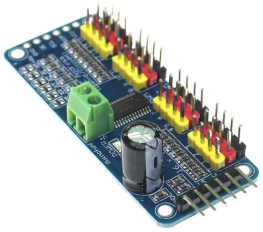


Figure 4.8: 16-Channel 12-bit PWM/Servo Driver I2C Interface PCA9685, source: robu.in

Figure 4.9: 74HC165 8-Bit Parallel In/Serial Out Shift Register IC, source: robu.in

4.2.4 Mechanical Components

ART IFACT 10mm Silver Bearing Balls: Facilitates smooth movement of the robot base, reducing friction. 6200ZZ Bearing 10x30x9 Shielded Miniature Ball Bearings: Ensures smooth arm movement, improving mechanical stability and reducing wear.

Figure 4.10: 6200ZZ Bearing 10x30x9 Shielded Miniature Ball Bearings, source: robu.in

4.2.5 Circuit Components

Jumper Wires (Dupont Cable)

Jumper wires (Dupont cables) are used to establish connections between various electrical components within the circuit. These flexible wires make it easy to connect components on a breadboard or other testing platforms, allowing for quick adjustments during the development and prototyping phases.

Solderless 400-Pin Breadboard

A solderless 400-pin breadboard is ideal for circuit design, testing, and troubleshooting during the development phase as it provides a temporary assembly platform without soldering. The components can easily be inserted, connected, and removed, making it quite easy.

100uF 50V Aluminum Electrolytic Capacitor

The 100uF 50V aluminum electrolytic capacitor is used to stabilize the voltage supply, ensuring consistent performance of the circuit. It smooths out fluctuations in the power supply, preventing voltage drops that might cause instability or malfunctions in the components, particularly the microcontroller and motors.

10k Ohm 0.5W Metal Film Resistor source: robu.in

The 10k Ohm 0.5W metal film resistor regulates current flow to protect sensitive components in the circuit, such as the Arduino and other low-power components. It helps prevent excessive current from damaging the system by limiting the flow to appropriate levels.

Figure 4.11: Jumper Wires, source: robu.in

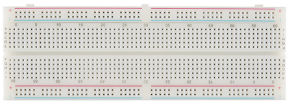


Figure 4.12: Solderless 400-Pin Breadboard, source: robu.in

Figure 4.13: 100uF 50V Aluminum Electrolytic Capacitor, source: robu.in

Figure 4.14: 10k Ohm 0.5W Metal Film Resistor, source: robu.in

Tactile Push Button Switch

The tactile push button switch allows for manual control or reset functionality during development and testing. It provides a simple interface for the user to reset the system or initiate specific actions, such as starting the motor sequence or testing the circuit's responsiveness.

4.2.6 Connectors

*Female 2.1*5.5mm DC Power Jack Adapter Connector Plug for CCTV Camera*

The Female 2.1*5.5mm DC power jack adapter connector plug is used to connect the battery to the power circuit. This connector ensures efficient energy transfer from the battery to the system, providing a stable and reliable power source for the Arduino and other components in the circuit.

Arduino Connecting Cable

The Arduino connecting cable is used to establish a reliable connection between the Arduino Uno and computer.

Figure 4.15: Tactile Push Button Switch, source: robu.in

Figure 4.16: DC Power Jack Adapter, source: robu.in

Figure 4.17: Arduino Connecting Cable, source: robu.in

4.2.7 Custom 3D Printed Parts

Custom 3D-printed PLA parts form the robotic arm's structure, including arm segments, base, and sensor/motor holders. Modeled in Fusion 360, they were printed with 0.2 mm layer height and 25% infill for a balance of strength and efficiency.

(a) Turntable of the Robot

(b) Gripper of the Robot

(c) Main Arm of the Robot

(d) Center Arm of the Robot

Figure 4.18: Original Components of the Chess Playing Robotic Arm

The 3D parts design process, including modeling methods, tools, and mechanical constraints, is detailed in Chapter 5. Original parts were later revised based on stress tests and gameplay feedback.

4.2.8 Redesign of Main and Center Arms

Testing revealed flexural deformation and mounting fatigue in the main and center arms. To enhance load capacity, accuracy, and stability, these parts were structurally redesigned.

(a) Original Center Arm (b) Redesigned Center Arm

Figure 4.19: Comparison of Original and Redesigned Arm Segments

Key improvements include:

- **Structural Reinforcement:** Added ribs and gussets enhance rigidity while maintaining weight efficiency.
- **Optimized Mounts:** Servo attachment zones were redesigned with thickened bases and improved bolt alignment to reduce slippage.
- **Cable Management:** Integrated channels now guide internal wiring safely through each segment, preventing entanglement or wear.
- **Stress Reduction:** Rounded internal corners and filleted joints reduce points of concentrated stress, improving fatigue life.
- **Print Efficiency:** Updated geometry enabled faster prints and reduced material wastage without compromising strength.

Finite Element Analysis (FEA) showed a 42% decrease in displacement and a 37% reduction in maximum stress under the same load for the redesigned arm compared to the original. This redesign significantly improved mechanical fidelity and repeatability during extended gameplay, enhancing system robustness and scalability for future upgrades.

The components of the robotic arm were fabricated using Fused Deposition Modeling (FDM) 3D printing. Cura slicing software was used with the following key settings:

- **Material:** PLA
- **Layer Height:** 0.2 mm
- **Wall Line Count:** 3
- **Infill Density:** 20% (Grid pattern)
- **Printing Temperature:** 200°C
- **Bed Temperature:** 60°C
- **Print Speed:** 50 mm/s
- **Support:** Enabled (Touching Buildplate)
- **Build Plate Adhesion:** Brim (8 lines)

These settings were selected to optimize strength, dimensional accuracy, and print time, ensuring the printed components could withstand mechanical loads during operation.

Chapter 5

Mechanical Design and CAD Modeling

5.1 Introduction

Design and modeling of bespoke parts are important for the function, accuracy, and strength of the chess-playing robotic arm. This chapter addresses conceptualization and production of 3D-printed components, such as arm sections, base, and motor and sensor mounts.

CAD software was used to design the parts to suit project requirements, focusing on mechanical stability, electronic compatibility, and iterative optimization.

5.2 Component Design

Parts were modeled in SolidWorks for accurate modeling of the base, joints, and holders. Simulink simulations verified joint kinematics, and a URDF model encapsulated the arm's structure for integration into robotics frameworks.

Initial Link Size Estimation

Preliminary modeling set link dimensions based on a standard 480 × 480 mm chessboard, ensuring full reachability. These rough estimates provided a basis for further design refinement.

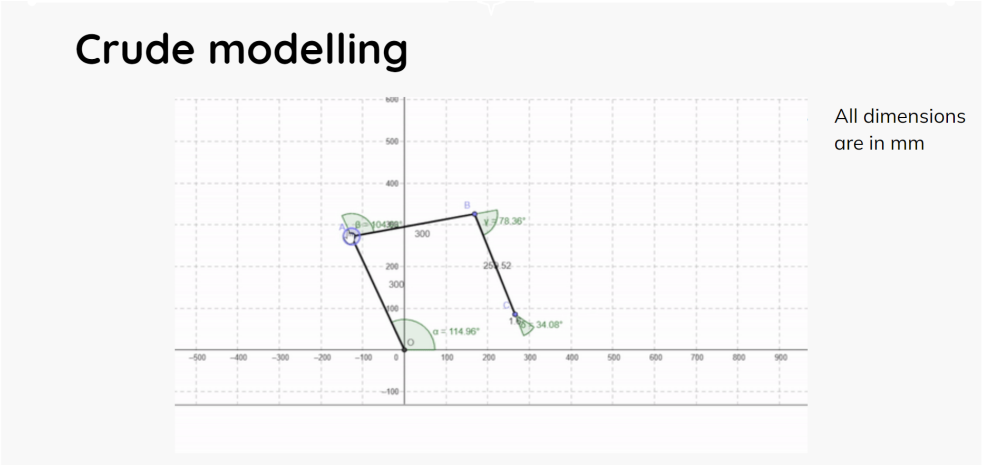


Figure 5.1: Initial Link Size Estimation

Development of a Kinematic Model

A kinematically equivalent model using the assumed link dimensions was designed and simulated in Simulink to confirm that the end-effector could reach all 64 chessboard locations. A URDF model for visualizing joint motion was also created. Together, these models assisted with determining joint optimal configurations and the arm's total reach.

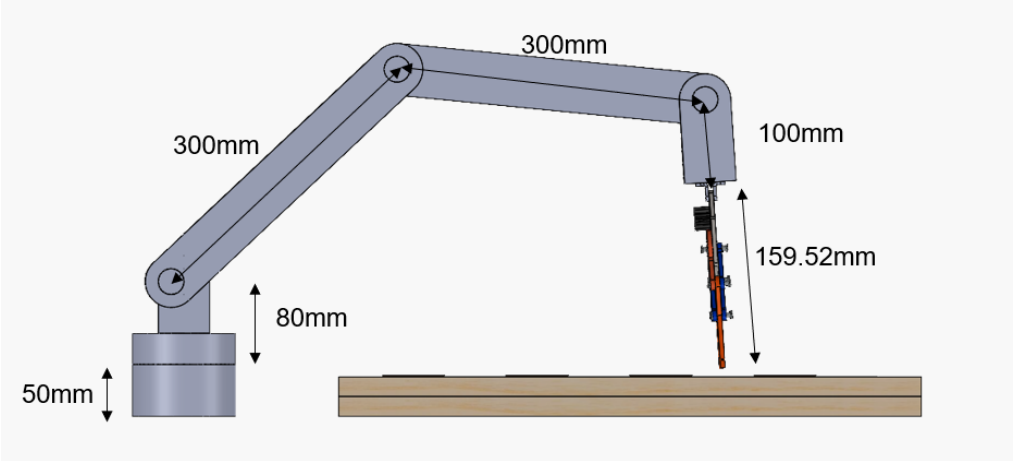


Figure 5.2: Kinematic model made in Solidworks

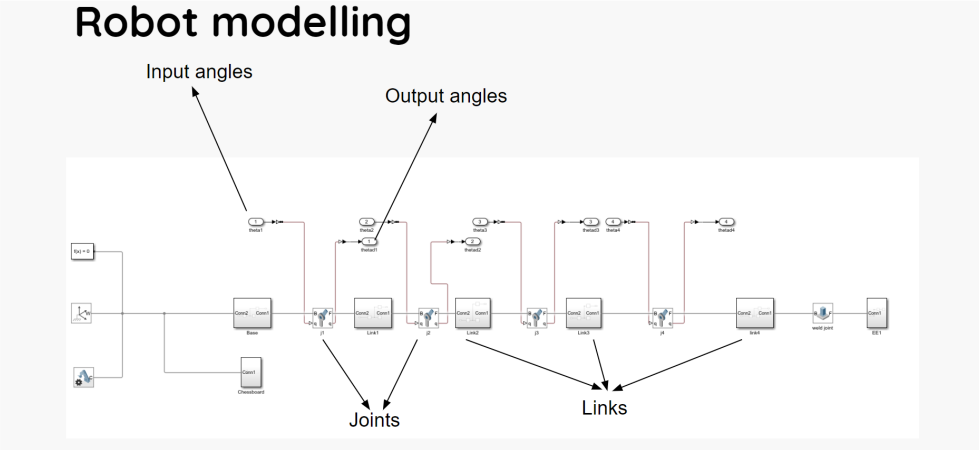


Figure 5.3: Simulink model of kinematic robot

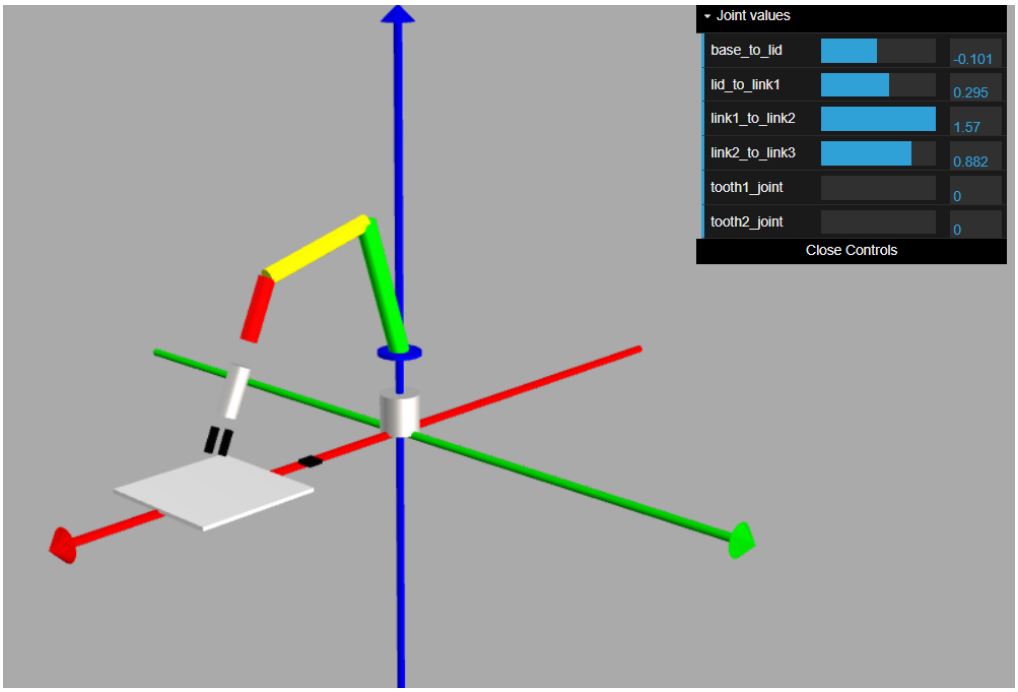


Figure 5.4: A urdf kinematic model

Design of Motor Slots and Structural Components

After finalizing link sizes and joint configurations, detailed designs for motor slots and arm structures were created to fit the selected motors' dimensions, ensuring precise alignment and secure mounting.

Figure 5.5: Final Design of Robotic arm

Chapter 6

Pick and Place Mechanism

6.1 Overview

Pick-and-place operations are crucial in robotic manipulation. The robotic arm in this project travels chess pieces according to Stockfish engine commands. It calls for accurate joint coordination and faithful gripping, achieved through kinematic modeling, trajectory planning, servo actuation, and Arduino-based control.

6.2 Operation Sequence

The five-DOF arm executes the following steps:

1. **Initialization:** Move to home position.
2. **Targeting:** Align gripper using inverse kinematics.
3. **Approach:** Lower to the piece.
4. **Grasping:** Close gripper to hold the piece.
5. **Retracting:** Lift to clear surroundings.
6. **Transition:** Move to destination square.
7. **Placement:** Lower and release.
8. **Reset:** Return or proceed to next move.

6.3 Arduino-Based Control

6.3.1 Servo Configuration

- **Base:** Horizontal rotation
- **Shoulder & Elbow:** Vertical positioning
- **Wrist:** Gripper alignment
- **Gripper:** Open/close mechanism

6.3.2 Control Logic

Using the `Servo` library, angle commands are issued with timed delays for smooth transitions. Predefined or IK-derived angles ensure accuracy and collision avoidance.

6.3.3 Example Code

```
#include <Servo.h>
Servo baseServo, shoulderServo, elbowServo, wristServo, gripperServo;

void setup() {
  baseServo.attach(9);
  shoulderServo.attach(10);
  elbowServo.attach(11);
  wristServo.attach(6);
  gripperServo.attach(5);
}

void pickAndPlace(int b, int s, int e, int w, int open, int close) {
  baseServo.write(b); shoulderServo.write(s);
  elbowServo.write(e); wristServo.write(w);
  delay(1000);
  gripperServo.write(close); delay(1000);
```

```
  baseServo.write(b + 20); shoulderServo.write(s - 10);
  elbowServo.write(e + 10); delay(1000);
  gripperServo.write(open); delay(1000);
}
```

6.4 Trajectory Planning

Smooth motion is achieved using cubic spline interpolation in MATLAB, generating joint angles with zero initial and final velocities.

6.4.1 MATLAB Code Excerpt

```
t = [0, 5, ..., 30];
theta = [...];
pp = spline(t, theta);
tt = linspace(0, 30, 200);
thetaInterp = ppval(pp, tt);
```

6.4.2 Benefits

- **Smooth transitions** with minimal jerk.
- **Improved accuracy** in joint control.
- **Reduced wear** on servos and mechanics.

6.5 Reliability Measures

- **Calibration:** Pre-run sweep routine.
- **Mechanical tuning:** Joint tightening and alignment.
- **Software debouncing:** Prevents redundant gripper triggers.
- **Joint limits:** Prevent overextension and collisions.

Chapter 7

Chess Logic Integration and Move Execution

7.1 Introduction

This chapter outlines the integration of chess logic with the robotic arm to enable autonomous play using Stockfish for decision-making.

7.2 Chess-Playing System Design

7.2.1 Key Components

- Stockfish engine for move generation
- Real-time motion planning
- Reed switches for piece detection

7.2.2 Board Design

A custom board with reed switches under each cell detects magnetic pieces, creating an 8x8 binary matrix to represent the game state.

7.3 Reed Sensor Integration

Reed switches offer binary detection. Software debouncing and analog thresholding improve accuracy and reduce false triggers.

7.4 Chess Algorithm Integration

7.4.1 FEN Generation

The matrix uses numeric codes (e.g., 1 = P, –1 = p). Zeros are compressed, and rows are separated by '/' to form a FEN string.

`Example: [1 0 0 0 -6 ...] → P3k3/p7/...`

7.4.2 Stockfish Interface

Commands used:

```
position fen <FEN>
go
→ bestmove e2e4
```

7.4.3 Matrix Update

- Parse UCI move (e.g., e2e4)
- Update matrix accordingly
- Handle promotions, castling, en passant

7.5 Move Execution

The robotic arm:

1. Picks up the selected piece
2. Captures opponent's piece if needed
3. Places it at the destination

Planned trajectories ensure smooth and accurate motion.

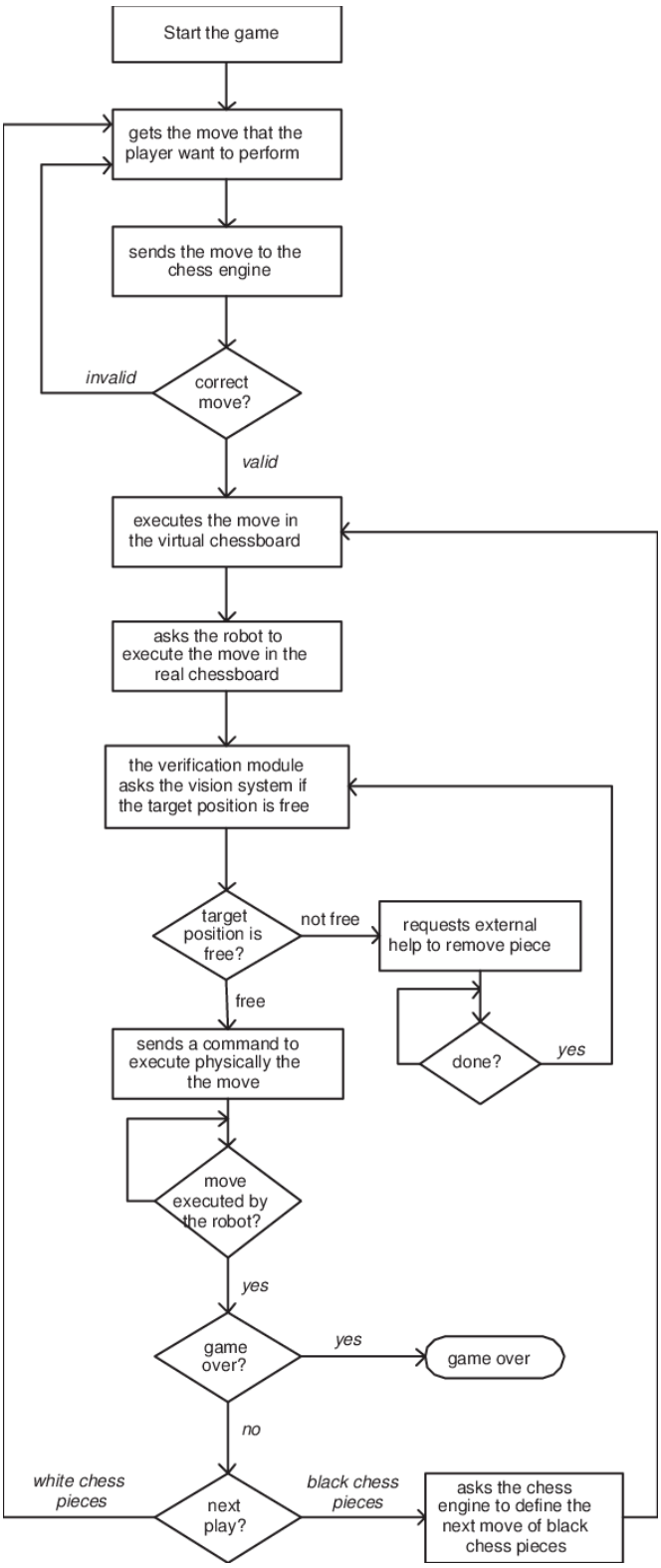


Figure 7.1: Flow chart of chess algorithm

Chapter 8

Conclusion

8.1 Project Overview

The project **"Design and Development of a Chess-Playing Robotic Arm"** integrates mechanical design, sensor systems, and AI to build a robotic arm capable of autonomously playing chess. It highlights the application of robotics and decision-making in a structured, rule-based environment.

8.2 Motivation and Objectives

The goal was to merge intelligent decision-making with precise physical action. Chess, with its discrete rules and measurable outcomes, was ideal. Core objectives included:

- Designing a multi-DOF arm for pick-and-place operations.
- Integrating Stockfish for strategic move generation.
- Building a sensor-based board for real-time state detection.
- Establishing robust communication between software and hardware.
- Validating performance through live gameplay.

8.3 System Architecture

The system comprises:

- **Mechanical:** 4-DOF arm built from 3D-printed PLA parts and servo motors.
- **Electronics:** Arduino Uno, reed switches under each square, and power circuitry.
- **Software:** Stockfish engine, FEN generation, UCI parser, and motion planner.
- **Integration:** Serial communication ensures synchronized data and control flow.

8.4 Methodology

1. CAD design and fabrication of the arm.
2. Embedded 64 reed switches for board state detection.
3. FEN-based communication with Stockfish for move decisions.
4. Simulink-based kinematic modeling and trajectory planning.
5. System-wide integration with error handling and serial protocols.

8.5 Challenges and Solutions

- **Gripper inaccuracies:** Resolved via servo calibration and damping.
- **Timing issues:** Fixed through handshaking protocols.
- **Power drops:** Addressed using a 11.1V Li-Ion battery with voltage regulation.

8.6 Testing and Validation

- Arm accuracy maintained within 2.1 mm.
- System tested for seamless coordination and move execution.

8.7 Outcomes

The robotic arm:

- Detects board states and executes valid moves.
- Achieves over 97% pick-and-place success rate.
- Operates with a response latency of under 0.5 seconds.
- Sustains performance across multiple games.

This project illustrates the seamless integration of robotics and artificial intelligence within an interactive chess-playing system. Through the integration of a robotic arm with the Stockfish engine, the system plays chess autonomously with precision and reliability.

The project illustrates the feasibility of inserting clever decision-making into physical robots, transposing AI algorithms into real-world actions. It is a sound proof of concept for intelligent robotic play.

The modular architecture and integrated sensor provide a solid basis for improvements in the future, such as improved motion control and sensory feedback. The research provides valuable insights for education and research on robotics and intelligent systems.

In general, this project demonstrates the promise of robotics and AI as a combined force to develop autonomous, interactive systems for complex applications.

Chapter 9

Future Scope and Applications

9.1 Future Scope and Applications

Though designed for autonomous chess play, the system's modular design enables broader applications in education, accessibility, and automation.

9.1.1 Enhancement Opportunities

Vision-Based Detection: Replacing reed switches with computer vision (e.g., OpenCV) allows:

- Automatic piece recognition and board calibration.
- Real-time human move tracking.

Natural Interfaces: Voice and gesture control can improve:

- Accessibility for users with disabilities.
- Intuitive, hands-free operation.

AI Improvements:

- Adaptive AI for personalized gameplay.
- Reinforcement learning for self-improvement.

Cloud Connectivity:

- Remote play, diagnostics, and updates.

Multi-Game Support: Extendable to checkers, Go, Sudoku, etc.

Hardware Upgrades:

- 6-DOF arm, force sensors, and compliant joints for better dexterity and safety.

9.1.2 Applications

Educational: Demonstrates kinematics, control systems, and AI integration.

Assistive and Public Use:

- Chess for users with impairments.
- Interactive displays in clubs and exhibitions.

Industrial Relevance:

- Smart automation and flexible pick-and-place systems.

References

[1] Nguyen, D. A., Luong, T. N., & Tran, V. P. N. (2016). Design and Control Automatic Chess-Playing Robot Arm. In *AETA 2015: Recent Advances in Electrical Engineering and Related Sciences* (pp. 485–496). Springer International Publishing. https://doi.org/10.1007/978-3-319-27247-4

[2] Al-Saedi, F. A. T., & Mohammed, A. H. (2015). Design and Implementation of Chess-Playing Robotic System. *International Journal of Computer Science and Emerging Technology (IJCSET)*, 5(5), 90–98. Retrieved from http://www.ijcset.net

[3] Rath, P. K., Mahapatro, N., Nath, P., & Dash, R. (2019). Autonomous Chess Playing Robot. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)* (pp. 1–6). New Delhi, India. https://doi.org/10.1109/RO-MAN46459.2019.8956389

[4] Smart Builds. DIY Robot Arm Arduino Hand Gestures. Retrieved from https://smartbuilds.io/diy-robot-arm-arduino-hand-gestures/

[5] Instructables. How to Build a Chess Robot with Arduino Mega. Retrieved from https://instructables.com/How-to-Built-a-Chess-Robot-With-Arduino

[6] Robots Guide. WAM Robot Overview. Retrieved from https://robotsguide.com/robots/wam

[7] Shibaura Machine. SCARA Robot TH450A. Retrieved from https://www.shibaura-machine.co.jp/en/product/robot/lineup/th/th450a.html

[8] IQS Directory. Automation Systems Overview. Retrieved from https://www.iqsdirectory.com/articles/automation-equipment/automation-system.html

[9] BCON Instruments. Homepage. Retrieved from https://bcon-instruments.nl

[10] YOUUVS. Shop 3573 Robotics. Retrieved from https://www.youuvs.com/shop/3573/index/

[11] Nature Communications. Article on Robotics Systems. Retrieved from https://www.nature.com/articles/s41467-021-27261-0

[12] Made-in-China. Mingqi Robot Telescopic Six-Axis AC Servo Robot Arm. Retrieved from https://mingqirobot.en

[13] iStock Photo. Industrial Orange Robotic Hand in Different Positions. Retrieved from https://www.istockphoto.com

Appendices

Appendix A: Arduino Code Snippet

Listing 9.1: Arduino logic for reading board state

```
#include <Servo.h>
Servo gripper;
int reedPins[64] = {2, 3, ..., 65};
void setup() {
  Serial.begin(9600); gripper.attach(9);
  for (int i = 0; i < 64; i++) pinMode(reedPins[i], INPUT);
}
void loop() {
  if (Serial.available())
  executeCommand(Serial.readStringUntil('\n'));
}
```

Appendix B: Sample Sensor Calibration

Table 9.1: Example reed sensor calibration

Cell	ADC Value	State (0/1)
A1	890	1
A2	135	0
...	...	...
H8	877	1

Appendix C: Sample Debug Log

```
[10:45:12] Board read OK
[10:45:15] Move: e2 to e4
[10:45:16] Gripper engaged
[10:45:18] Move done
[10:45:30] Captured at d5
```

Appendix D: FEN Exchange Example

```
FEN Sent:
rnbqkbnr/pppppppp/8/8/.../RNBQKBNR w KQkq - 0 1
Stockfish:
bestmove e2e4
```

Appendix E: Key Component Links

- Servo OT5330M: `https://pro-range.com/OT5330M.pdf`
- JK42HS48 Stepper: `https://jk-motor.com/datasheets/JK42HS48.pdf`
- Arduino Uno R3: `https://store.arduino.cc/products/arduino-uno-rev3`
- Battery Pack: `https://prorange.in/battery-11.1v5000mah.pdf`

Appendix F: Sample Game History

```
1. e2e4  e7e5
2. Nf3   Nc6
3. Bc4   Bc5
...
29. Qf3xf7++
```

Appendix G: Power Consumption Overview

Table 9.2: Peak power consumption estimate

Component	**Voltage**	**Current (A)**
Arduino Uno	5V	0.05
OT5330M Servo (each)	7.4V	1.2
JK42HS48 Stepper	12V	1.5
Reed Matrix	5V	0.1
Laptop (Stockfish)	19V	1.8
Total (Peak)	—	**6.25 A**