

Accurate Pick and Place Arm

By

Ian Chan

Nithin Durgam

Michael Talapin

Proposal For ECE445, Senior Design, Fall 2026

TA: Abdullah Allawad

September 18th,

Project No. 8

1 Introduction

1.1 Problem

Automation is becoming more common in manufacturing, laboratories, and many other industries. Robotic arms are often used for tasks that involve picking up and moving parts because they can work continuously and repeat the same motion many times. As these systems are used for smaller parts and more complicated tasks, accuracy and repeatability become much more important. Even a small positioning error can cause a part to be placed incorrectly, damage a component, or cause problems later in the manufacturing process.

Accurate pick-and-place robots are also becoming more important in laboratory environments. Labs may use robotic systems to move samples, handle containers, or perform repetitive procedures where consistency is important. In these situations, a robot needs to be able to return to the same position and handle objects without damaging them. Better accuracy can help reduce human error and make automated systems more reliable.

This problem also relates to safety and public welfare. Robots can take over repetitive or potentially dangerous tasks that would normally be done by a person, such as working around machinery, chemicals, or other hazardous equipment. However, the robot itself also needs to be accurate and predictable so that it does not damage equipment or create additional safety risks. In areas such as medical or laboratory automation, poor positioning could also affect the quality of testing or research.

There are also economic and environmental reasons to improve the accuracy of robotic systems. A more accurate robot can reduce defective parts, wasted material, and the amount of rework needed during production. This can lower manufacturing costs while also reducing material and energy waste. As labor costs increase and companies continue to automate more processes around the world, reliable robotic systems are becoming increasingly valuable.

1.2 Solution Overview

The goal of this project is to design and build an accurate pick-and-place robotic arm and demonstrate its performance using a chessboard. Moving chess pieces provides a simple way to test whether the arm can repeatedly identify specific locations, pick up objects, and place them accurately in a controlled environment.

A chessboard provides a challenging and intuitive test of the system's precision. To reliably pick up and place individual chess pieces without disturbing the surrounding pieces, the arm's end-effector must achieve approximately 3 mm of positional accuracy. Reaching this level of precision requires every part of the system (mechanical design, electrical components, computer vision pipeline, kinematics, and control algorithms) to work together accurately and consistently. Our primary demonstration will allow a user to select any chess piece or position on the board and specify a destination. The robotic arm will then identify the requested location, pick up the piece, and place it at the desired position while avoiding interference with nearby pieces.

If time permits, we would like to extend the project by allowing the robotic arm to autonomously play a complete game of chess against a human opponent. This would add another layer of complexity by combining the arm's existing perception and manipulation capabilities with board-state recognition, move planning, and a chess engine. Since accuracy is the goal and not speed, we will follow a 5+10 and 10 + 10 format.

At a high level, our goal is to avoid the complexity and cost associated with traditional multi-joint robotic arms, where every additional degree of freedom requires another motor, sensor, and control system. Instead, we plan to use a simpler and more elegant mechanical architecture designed specifically around the requirements of the task.

The arm will consist of two links that provide planar motion across the x-y plane, while the end-effector moves vertically along the z-axis to pick up and place chess pieces. On the electrical side,

custom PCBs will integrate sensors such as magnetic encoders and Hall-effect sensors to accurately determine joint position and provide reliable feedback for the control system.

The software stack will tie the entire system together. Closed-loop PID controllers will provide precise joint positioning, while a computer vision pipeline will identify chess pieces and determine their locations on the board. These positions will then be converted into robot coordinates and passed through the inverse kinematics system to determine the joint commands required to move the end-effector to the desired location.

1.3 Visual Aid

Pick-and-Place Robotic Arm System Concept

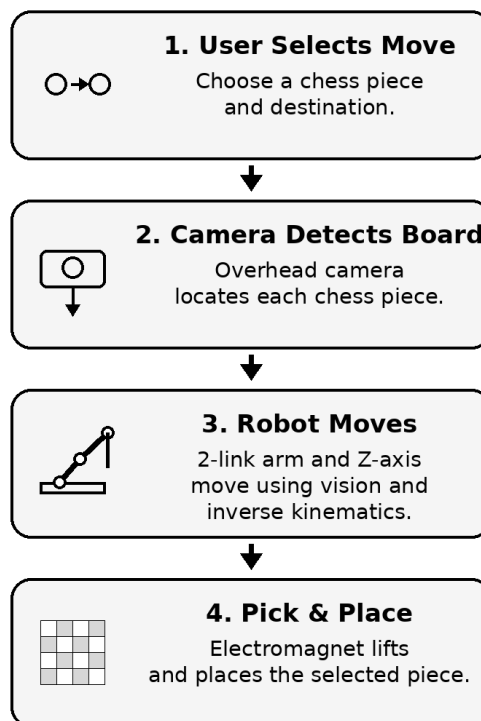


Figure 1: Project Concept Visual Aid (Created by Author(s))

1.4 High Level Requirements

- Achieve end-effector positioning accuracy +/- 3 millimeters.
- Correctly identify all chessboard pieces with at least 90% recall
 - $\text{Recall} = \text{True Positives} / (\text{True Positives} + \text{False Negatives})$
- The robotic arm must successfully pick up and place a chess piece in the intended square in at least 90% of attempted moves.

2 Design

2.1 Block Diagram & Preliminary Mechanical Drawing

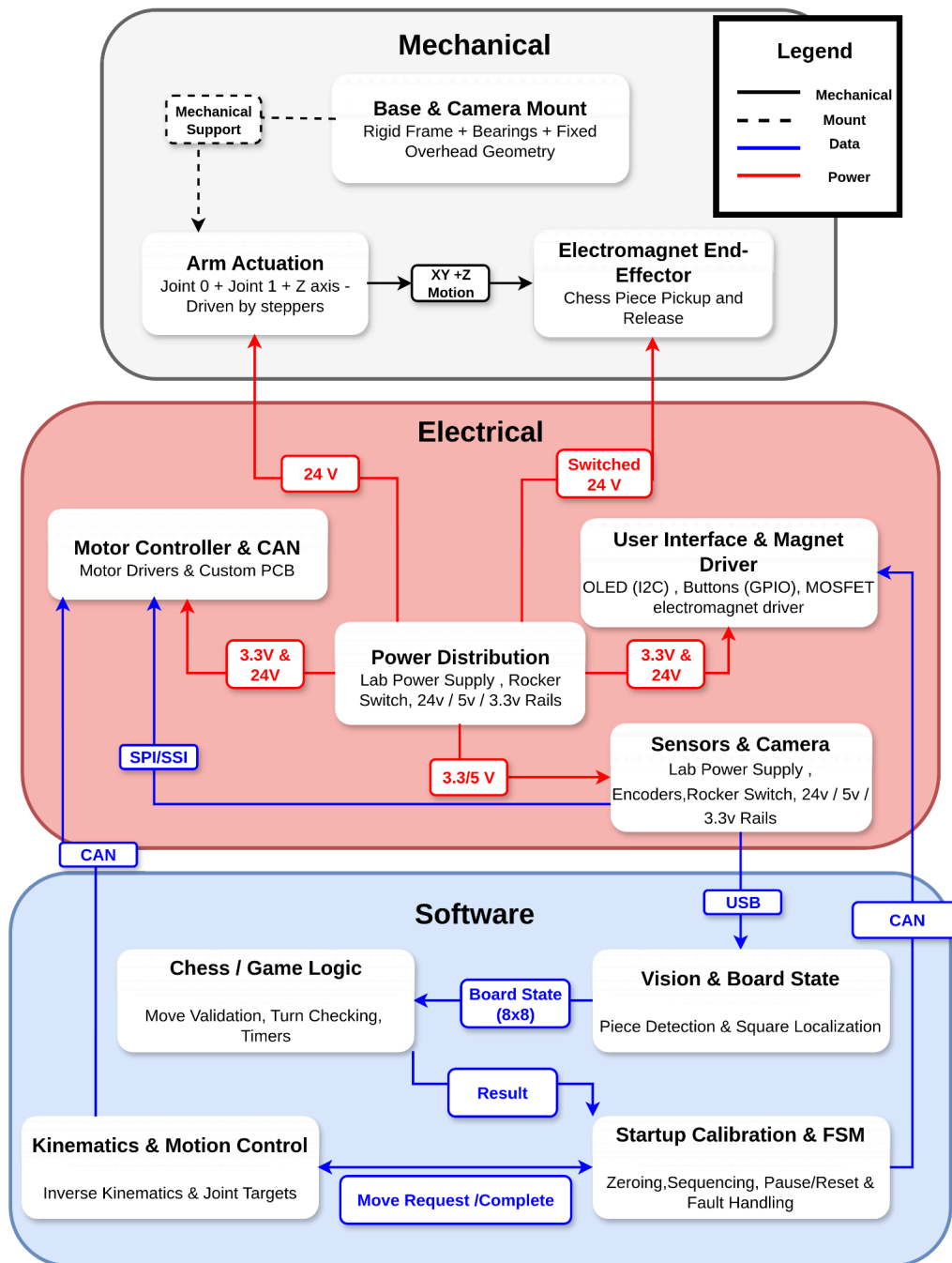


Figure 2: Block Diagram (Created by Author(s))

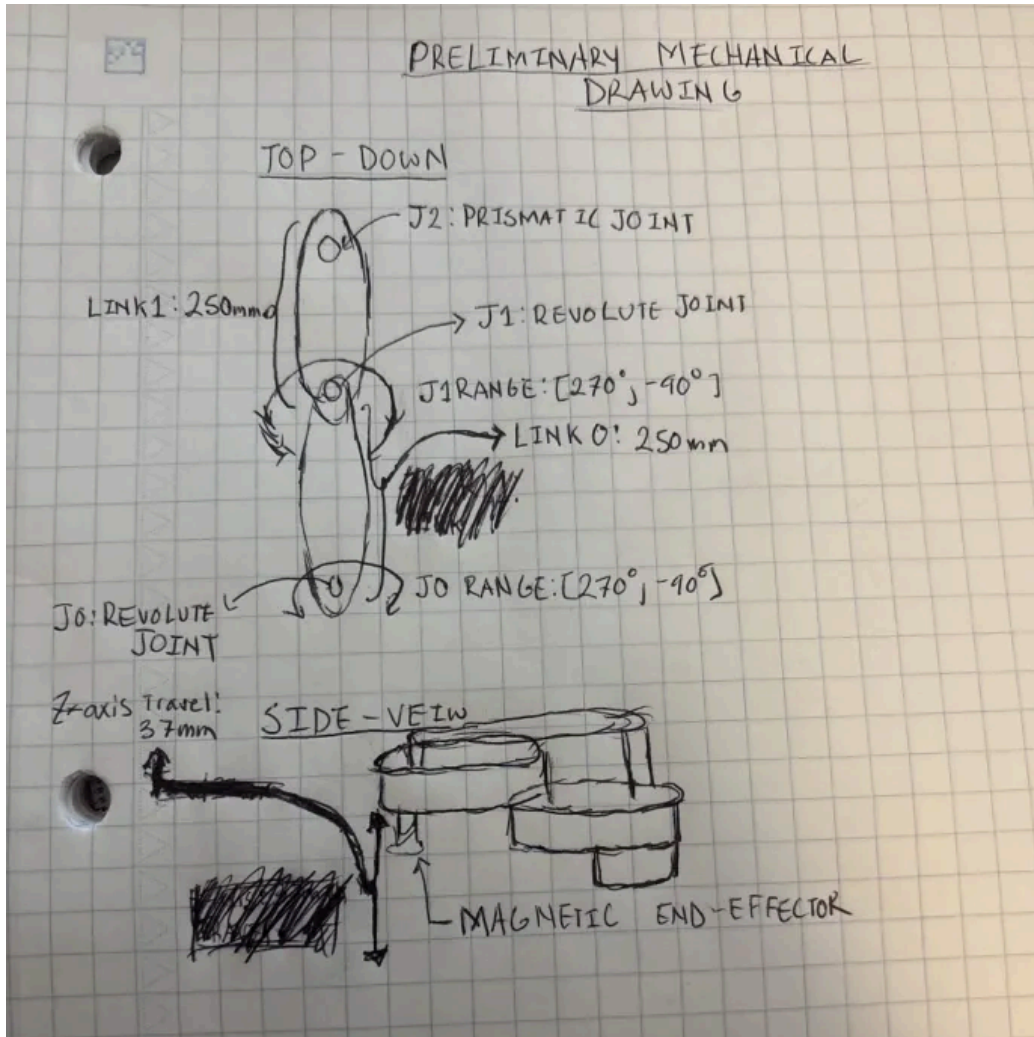


Figure 3: Mechanical Drawing (Created by Author(s))

2.2 High Level Block Diagram Explanation

The robot will use two rotational joints for x-y movement and a vertical Z-axis for picking up and placing pieces. An electromagnet mounted on the end-effector will be used to securely lift the chess pieces. The mechanical system is designed around stiffness, low backlash, and repeatability so that the end-effector can reach a commanded position within 3 mm. Joint 0, Joint 1, and the Z-axis will use stepper motors with encoder feedback to improve positioning accuracy. An overhead camera will be mounted above the board so the system can detect chess pieces and determine their locations.

The electrical system connects the mechanical hardware to the software. It includes motor drivers, absolute encoders, Hall-effect or reference sensors, power distribution, CAN communication, the electromagnet control circuit, an OLED display, and Start, Pause, and Reset buttons. The Jetson will act as the main computing platform and communicate with the rest of the robot through CAN.

The software controls the full sequence of operation. At startup, the robot performs a calibration routine to determine joint offsets and move to a known zero position. The vision system then identifies the board state and piece locations. During gameplay, the software tracks player turns and timers, validates moves, and determines the robot's next move. Inverse kinematics converts the desired chessboard locations into joint positions, while closed-loop control moves the arm to those positions. The project will ultimately demonstrate accurate piece recognition, reliable pick-and-place operation, and coordinated control of a complete robotic system.

Each of these subsystems will be described in more detail below, with tolerance analysis and subsystem requirements

2.3 Mechanical Subsystem

Mechanical Subsystem Explanation

The mechanical subsystem is responsible for positioning the end-effector over the chessboard and moving it vertically to pick up and place pieces. It consists of Joint 0, Joint 1, the Z-axis, the end-effector, and the camera mount. These components must work together to keep the end-effector within the project's 3 mm positioning requirement. Joint 0 is located at the base and provides the first rotational axis. It supports and moves the rest of the arm, so it will use a NEMA 34 motor.

Joint 0 receives commanded angles from the software subsystem and provides encoder feedback to the controller. It must maintain approximately $\pm 0.10^\circ$ positioning accuracy, have less than 0.10° of mechanical backlash, include an encoder and homing sensor, and use bearings to support the arm loads rather than relying only on the motor shaft.

Joint 1 provides the second rotational axis and works with Joint 0 to control the x-y position of the end-effector. It will use a NEMA 23 motor and will also interface with the controller through commanded joint angles and encoder feedback. Joint 1 must maintain approximately $\pm 0.10^\circ$ positioning accuracy, less than 0.10° backlash, and include an encoder and reference sensor. The structure must also remain stiff enough that deflection does not significantly affect positioning.

The Z-axis moves the end-effector vertically and will use a NEMA 14 motor with a lead screw, linear rail, or similar mechanism. It must provide enough travel to clear the tallest chess piece and maintain vertical positioning within approximately ± 3 mm. Lateral play should remain below 0.5 mm to prevent horizontal positioning error.

The end-effector physically picks up and releases the chess pieces. It must securely hold every type of chess piece while allowing some tolerance for small positioning errors. The gripper should tolerate at least ± 3 mm of horizontal misalignment and must not interfere with surrounding pieces. This directly supports the requirement of successfully completing at least 90% of pick-and-place attempts.

The camera mount holds the overhead camera in a fixed position relative to the chessboard and robot. Because the vision system depends on this relationship, the mount should limit camera translation to approximately 0.5 mm and angular movement to about 0.1° after calibration.

Mechanical Subsystem Requirements

- Joint 0 shall reach commanded positions within $\pm 0.10^\circ$.
- Joint 1 shall reach commanded positions within $\pm 0.10^\circ$.
- Mechanical backlash at each rotary joint shall be $< 0.10^\circ$.
- Z-axis positioning error shall be ≤ 3 mm.
- Lateral Z-axis play shall be < 0.5 mm.
- The end-effector shall tolerate within 3 mm pickup misalignment.
- Camera mount displacement shall remain < 0.5 mm after calibration.

Mechanical Subsystem Tolerance Analysis

A major risk is accumulated joint-position error. For a two-link planar arm,

$$x = L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2)$$

$$y = L_1 \sin(\theta_1) + L_2 \sin(\theta_1 + \theta_2)$$

For 250 mm links and $\pm 0.10^\circ$ joint error:

$$\text{Position Error} \leq (500)(0.001745) + (250)(0.001745) \approx 1.31 \text{ mm}$$

Adding approximately 0.5 mm of structural deflection and 0.5 mm of camera-calibration error gives:

$$\text{Total Error} \approx 2.31 \text{ mm}$$

This is below the required 3 mm positioning error, showing that the proposed mechanical design is feasible if the joints and structure meet these tolerances.

2.4 Electrical and PCB Subsystem

Electrical Subsystem Explanation

The electrical subsystem connects the mechanical hardware to the software and is responsible for power distribution, motor control, sensing, communication, end-effector control, and the user interface. These components must provide reliable feedback and control so the robot can meet the 3 mm positioning and 90% pick-and-place success requirements.

Power Distribution will take power from a laboratory power supply and distribute it throughout the robot. The primary bus will provide 24 V for the motor drivers and electromagnet, while regulators will generate 5 V and 3.3 V for sensors, PCBs, the OLED, and other low-voltage electronics. The Jetson will use a separate wall power supply and communicate with the robot through CAN. A rocker switch will provide main system power or enable control.

Motor and CAN Control will command Joint 0, Joint 1, and the Z-axis. CAN-capable motor drivers, where compatible with the selected motor, will receive target positions from the Jetson. The CAN bus will operate at approximately 500 kbps, which provides sufficient bandwidth for joint commands and sensor feedback while maintaining reliable communication. Since Joint 0 uses a NEMA 34 motor, its final driver must be selected to support the motor's required current and torque.

Position Sensing will use absolute optical encoders to measure Joint 0 and Joint 1 position. Magnetic encoders may be substituted if optical encoders become too expensive, provided they maintain the required resolution. Hall-effect sensors or limit switches will provide independent reference positions for homing and calibration. This feedback directly supports the 3 mm positioning requirement.

End-Effector Control will switch an electromagnet used to pick up the chess pieces. Because the electromagnet requires more current than a microcontroller output can provide, a MOSFET or similar switching circuit will control it from the 24 V power rail. The circuit must also include protection against inductive voltage spikes when the electromagnet is turned off.

User Interface will include an OLED screen and Start, Pause, and Reset buttons. The OLED will display system status, faults, and game time. The buttons will connect to PCB digital inputs and allow the user to control the robot without directly interacting with the Jetson software.

Electrical Requirements

- The 24 V rail should remain within $\pm 5\%$ and initially support at least 10 A continuously.
- The 5 V rail must provide at least 2 A and the 3.3 V rail at least 1 A.
- Motor drivers must be rated at least 20% above the expected operating current of their motors.
- Joint encoders must provide at least 12-bit absolute angular resolution.
- The CAN network must operate reliably at 500 kbps.
- The electromagnet switching circuit must be rated at least 25% above its measured steady-state current.
- Start, Pause, and Reset inputs must be electrically distinguishable and reliably detected by the controller.

Tolerance Analysis

Encoder resolution is a major electrical risk because angular measurement error becomes end-effector position error. A 12-bit encoder provides:

$$\text{Resolution} = 360^\circ / 4096 = 0.0879^\circ \text{ per count}$$

The maximum quantization error is approximately half a count:

$$\Delta\theta = \pm 0.0439^\circ = \pm 0.000767 \text{ rad}$$

Using two 250 mm links, the conservative end-effector error is:

$$\text{Error} \leq (500)(0.000767) + (250)(0.000767)$$

$$\text{Error} \leq 0.58 \text{ mm}$$

Therefore, encoder quantization contributes less than 0.6 mm of position error, leaving over 2 mm of the 3 mm requirement available for mechanical backlash, calibration, vision, and control errors. This shows that a 12-bit optical or magnetic absolute encoder provides sufficient resolution with reasonable margin.

2.5 Software Subsystem

Software Subsystem Explanation

The software subsystem coordinates the complete operation of the robotic arm from startup calibration through the completion of a chess move. It connects the camera, encoders, motor controllers, electromagnet, OLED display, buttons, and chess logic into one system. The software must support the three main high-level requirements by maintaining accurate robot positioning, correctly identifying chess pieces with at least 90% recall, and coordinating reliable pick-and-place operations.

The startup and calibration block establishes a repeatable robot coordinate system every time the system is powered on. Before normal operation begins, the software reads the absolute encoders and reference sensors to determine the offset between the measured motor positions and the defined Joint 0, Joint 1, and Z-axis zero positions. These offsets are stored in global variables and applied to future encoder measurements. Once the offsets are calculated, the robot moves to its defined zero or home position. While this is occurring, the OLED cycles between “Calibrating.”, “Calibrating..”, and “Calibrating...” to indicate that the system is not yet ready. The state machine cannot begin until calibration is completed successfully.

The user-interface block handles the Start, Pause, and Reset buttons and controls information displayed on the OLED. Pressing Start begins board-state verification and, if valid, starts the game. Pause stops the active player's timer and prevents the state machine from beginning another movement. If Pause is activated while the robot is carrying a piece, the electromagnet remains energized so the piece is not dropped. Reset stops the current game and returns the software to its startup state. The OLED displays the current game status, player timers, calibration state, and error messages. Button inputs should be detected within 100 ms so the interface feels responsive.

The computer vision block captures an image of the chessboard using the overhead camera and determines which pieces occupy each of the 64 board squares. The vision model must identify chess pieces with at least 90% recall. After classification, camera calibration and coordinate transformation are

used to relate locations in the image to physical locations on the chessboard. Since the dimensions and locations of the chessboard squares are known, the software can use the calculated center of each square as the robot's target instead of relying only on the detected center of a chess piece. The resulting board state is passed to the game-state block, while physical x-y coordinates are passed to the kinematics block when the robot needs to move a piece.

The game-state block keeps track of whose turn it is, the previous board state, the current board state, and each player's remaining time. At the beginning of the game, the software verifies that the detected pieces form a valid starting state. During the human player's turn, the camera captures a new board state and compares it with the stored state. The system must detect a valid change in pieces that corresponds to a legal move before changing turns. If the board changes in an unexpected way, the system displays an error and waits for the problem to be corrected. Each player has an independent software timer. Only the timer belonging to the active player counts down. The timer should update at least every 100 ms, giving a maximum displayed timing resolution of 0.1 seconds. If either player's timer reaches zero, the game immediately transitions to a game-over state and no additional robot moves are allowed. The system will initially support the planned 5+10 and 10+10 chess timing formats.

When the board changes after the human player's turn, the chess logic verifies that the move is legal and updates the stored game state. When it becomes the robot's turn, the current board state is passed to the chess engine. The chess engine returns a legal move containing a starting square and destination square. The software then sends these locations to the vision and kinematics blocks. The same block also detects game-ending conditions such as checkmate or other terminal game states and sends the appropriate result to the display.

The kinematics block converts the selected chessboard positions into robot coordinates and calculates the Joint 0 and Joint 1 angles required to reach them. The motion sequence moves the robot above the selected piece, lowers the Z-axis, activates the electromagnet, raises the piece to a safe clearance height, moves to the destination square, lowers the piece, turns off the electromagnet, and

returns to the camera-viewing position. Joint-position calculations must have enough numerical precision that software rounding contributes less than 0.5 mm of end-effector error.

The joint-control block sends position commands to the motor controllers and continuously receives encoder feedback. A closed-loop controller compares commanded and measured positions and corrects the joints until they reach their target. The control loop should operate at a minimum of 50 Hz, giving a new control update every 20 ms. The software should only consider a joint movement complete once the reported joint position is within approximately $\pm 0.10^\circ$ of the target for multiple consecutive control cycles.

The end-effector control block commands the electromagnet. The magnet is activated only after the robot reaches the pickup position and remains energized until the piece reaches its destination. If the motor controller reports an error or the robot fails to reach the required position, the state machine prevents the end-effector sequence from continuing.

The main finite-state machine connects all of the software blocks together. Each state has defined entry conditions, completion conditions, and fault conditions. Errors such as an invalid board state, calibration failure, motor fault, or failed movement prevent the software from advancing to the next state. The OLED displays the corresponding error so the user can identify the problem. This prevents one failure from continuing through the rest of the movement sequence and potentially causing an incorrect chess move or collision.

Software Requirements

- The computer vision system must identify chess pieces with at least 90% recall.
- Startup calibration must successfully determine and store all required joint offsets before the main state machine can begin.
- Joint-control calculations must update at a minimum rate of 50 Hz.
- The software must verify that each joint is within approximately $\pm 0.10^\circ$ of its commanded position before continuing to the next motion state.

- Kinematic and numerical calculations must contribute less than 0.5 mm of end-effector positioning error.
- Start, Pause, and Reset inputs must be detected within 100 ms.
- Player timers must update at least every 100 ms and detect timer expiration within one update cycle.
- Only the active player's timer may count down.
- An invalid board state or illegal human move must prevent the turn from advancing.
- Motor, calibration, or end-effector faults must prevent the state machine from beginning the next operation.
- The electromagnet must remain activated while a chess piece is being transported, including if the system is paused while holding a piece.

Tolerance Analysis

One major software risk is error when converting camera measurements into physical robot coordinates.

If this error is too large, the mechanical system could meet its own accuracy requirements while still moving to the wrong location.

For an initial feasibility analysis, assume a 400 mm-wide chessboard occupies at least 800 pixels across the calibrated camera image.

$$\text{Scale} = 400 \text{ mm} / 800 \text{ pixels} \quad \text{Scale} = 0.5 \text{ mm/pixel}$$

If board detection and camera calibration can locate a square center within $\sim \pm 2$ pixels:

$$\text{Vision Position Error} = 2 \text{ pixels} \times 0.5 \text{ mm/pixel}$$

$$\text{Vision Position Error} = \pm 1.0 \text{ mm}$$

This is below the overall 3 mm end-effector positioning requirement and leaves approximately 2 mm for mechanical backlash, encoder error, structural deflection, and joint-control error. Because the chessboard has a known fixed geometry, the software can also identify the board corners during calibration and calculate the center of each square mathematically. This reduces the robot's dependence on finding the exact center of an individual chess piece and provides additional margin for the vision system.

3 Ethics

Our project involves a robotic arm that autonomously moves chess pieces using stepper motors, encoder feedback, computer vision, and an electromagnet. The main safety concerns are unexpected robot motion, pinch or crush points at the joints, electrical faults, and software or vision errors that could command the arm to an incorrect location.

The IEEE Code of Ethics emphasizes protecting public safety and welfare, honestly communicating technical limitations, and identifying risks associated with engineering decisions. To follow these principles, we will accurately report the robot's measured accuracy and reliability rather than claiming performance beyond our test results. Safety-critical software will include joint and workspace limits, movement verification, and fault states that stop operation when unexpected behavior is detected. Intentional misuse, such as bypassing safety limits or modifying the robot to manipulate dangerous objects, will be reduced by restricting motion commands and requiring controlled operation.

For physical safety, OSHA 29 CFR 1910.212 requires guarding against hazards created by moving machine components. ISO 10218-1:2025 provides additional guidance for robot risk reduction and inherently safe design. Illinois OSHA also requires public workplaces to control recognized hazards, while UIUC requires machine-guarding assessments and appropriate safeguards for campus equipment. We will use an emergency stop, defined robot work area, electrical protection, controlled testing procedures, and power removal during maintenance.

Societally, accurate robotic systems can reduce repetitive human work, manufacturing errors, and material waste. However, increased automation may also reduce some manual jobs while increasing demand for robotics and technical skills. Improved accuracy can lower operating costs and waste, making automation more accessible while reducing unnecessary material and energy use.