ECE445 Design Document: JengaBot — A Robotic System for Playing Jenga with a Human

Team Members: Wangyihan Guo, Peiran Wei, Jiacheng Ye, Hengtie Zhu

Advisor: Prof. Pavel Loskot

Date: April 30, 2026

1. Introduction

1.1 Problem Statement

While technologies such as 3D printing and embodied intelligence have progressed significantly, precise object manipulation in constrained spaces continues to be a complex problem. Conventional robotic arms are too bulky for tight workspaces, yet such spaces are well-suited for customized 3D-printed frameworks. This project explores the feasibility of combining compact 3D-printed structural frames with precision robotic mechanisms to perform delicate manipulation tasks within confined volumes.

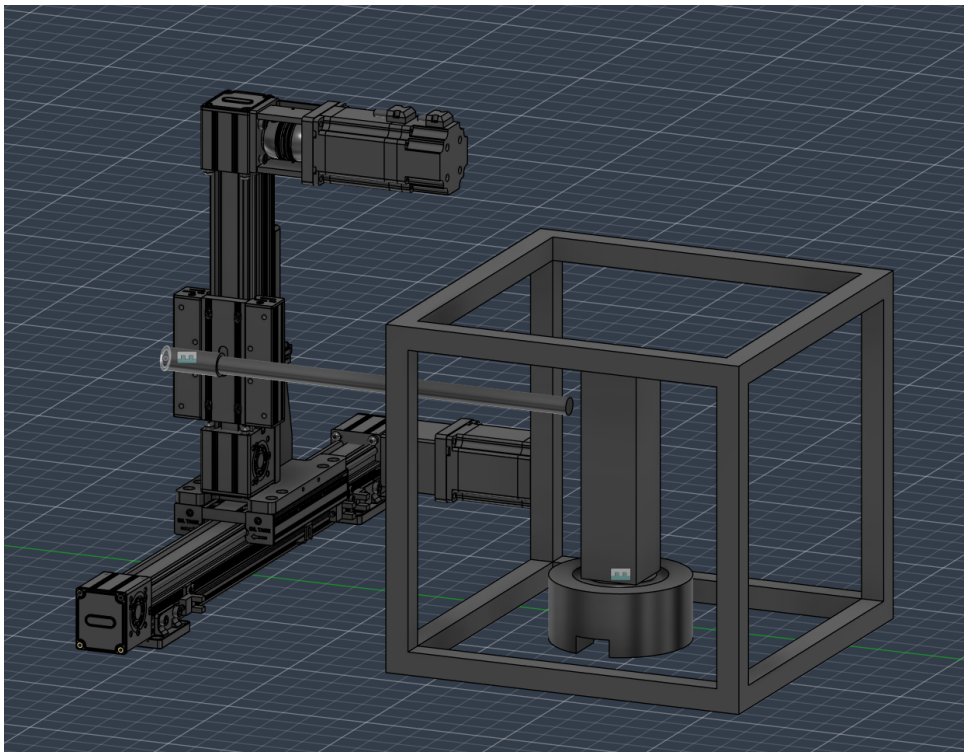

Figure 1. CAD model of the three-axis motion platform and end-effector arrangement.

1.2 Solution Overview & Visual Aid

To address this spatial limitation, we propose JengaBot: a system that integrates a custom 3D-printed enclosure with a three-axis motion control platform and a specialized end-effector. The system demonstrates its capability by playing Jenga interactively against a human opponent — a task requiring delicate block extraction without toppling the tower.

Four cameras mounted at the corners of the 3D-printed frame capture the real-time state of the Jenga tower from multiple angles. A Raspberry Pi processes these images, constructs a digital model of the tower, selects the optimal block to remove via predefined algorithmic strategies, and drives the three-axis system to execute the move.

Figure 1 illustrates the complete JengaBot system architecture in use context. The system consists of a custom 3D-printed structural frame, a three-axis Cartesian motion platform, four corner-mounted cameras, a Raspberry Pi for vision and strategy computation, and a Z-axis-mounted end-effector for block interaction. The mechanical frame constrains and stabilizes the workspace while also providing mounting positions for cameras, linear motion components, and electronics. The motion platform moves the end-effector to the selected block location, where the end-effector performs pushing, extraction, gripping, and placement actions. This integrated design enables the robot to perceive the tower state, make block-selection decisions, and execute precise manipulation within a compact footprint.

Figure 2. Initial CAD model of the end-effector arrangement.

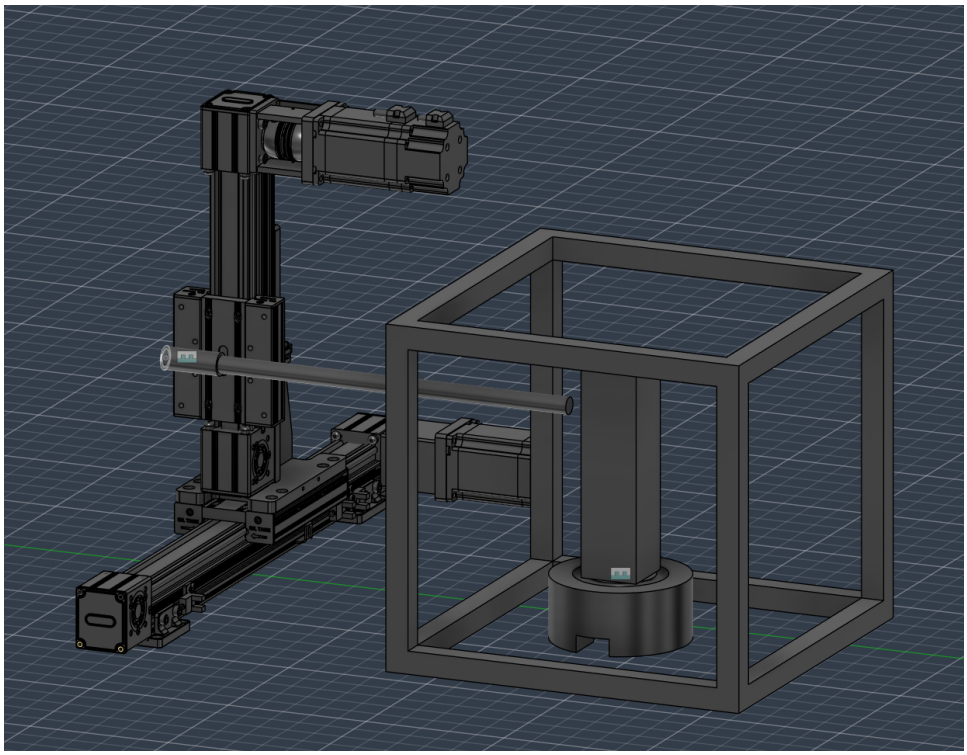

Figure 3. Updated CAD model showing the integrated motion platform, vertical axis, Jenga tower frame, and working space.

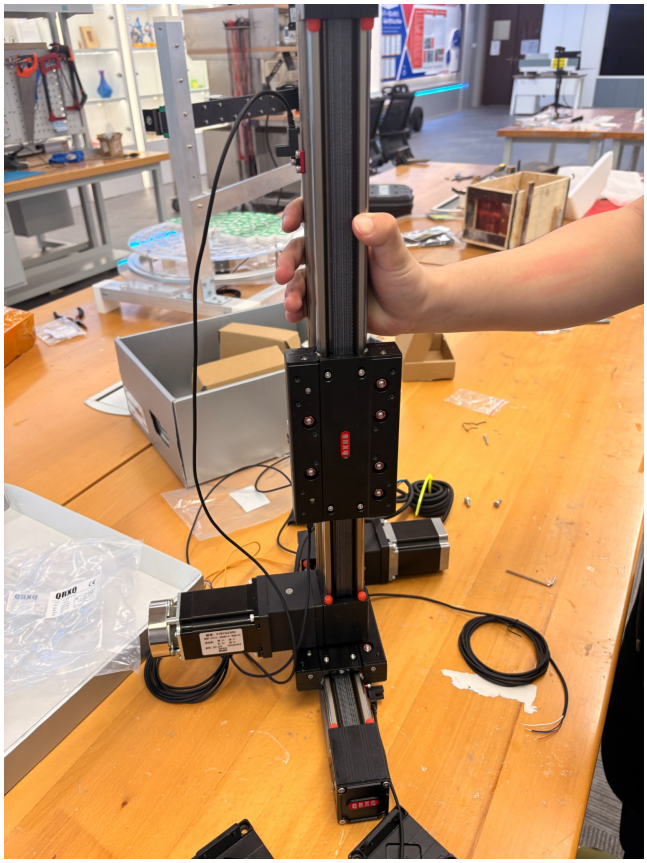

Figure 4. Physical assembly of the purchased two-dimensional motion platform and vertical linear axis.

Since the previous design stage, the mechanical subsystem has been updated from a conceptual CAD model to a partially assembled physical prototype. Instead of fabricating the entire motion base from scratch, the team purchased a commercial two-dimensional linear motion platform to serve as the X-Y stage. A vertical linear module is mounted on the moving platform to provide Z-axis motion, forming a compact three-axis Cartesian gantry around the Jenga workspace. This change reduces fabrication risk, improves rail alignment, and allows the team to focus more on end-effector integration, motor control, and system-level testing.

1.3 High-Level Requirements

1. **Gameplay Completeness:** The system shall autonomously complete the core Jenga interaction cycle of **detecting the tower state, selecting a target block, moving to the target, extracting the block, and placing it on the top layer** for at least **3 successful block operations per game** without causing tower collapse due to robot-induced error. Over **10 test games**, the system shall achieve a **successful game completion rate of at least 70%** under standard tower conditions.
2. **Motion Precision:** The three-axis motion platform shall achieve a repeatable positioning accuracy of **±1.0 mm** within the full operating workspace of approximately **300 mm × 300 mm × 250 mm**, which is sufficient for targeting individual Jenga blocks and aligning the end-effector with the selected extraction face.
3. **Response Time:** The system shall complete one full manipulation cycle, including **image capture, tower-state update, target selection, motion execution, block extraction, and top placement**, within **15 s** under normal operating conditions.

2. Design

2.1 Block Diagram

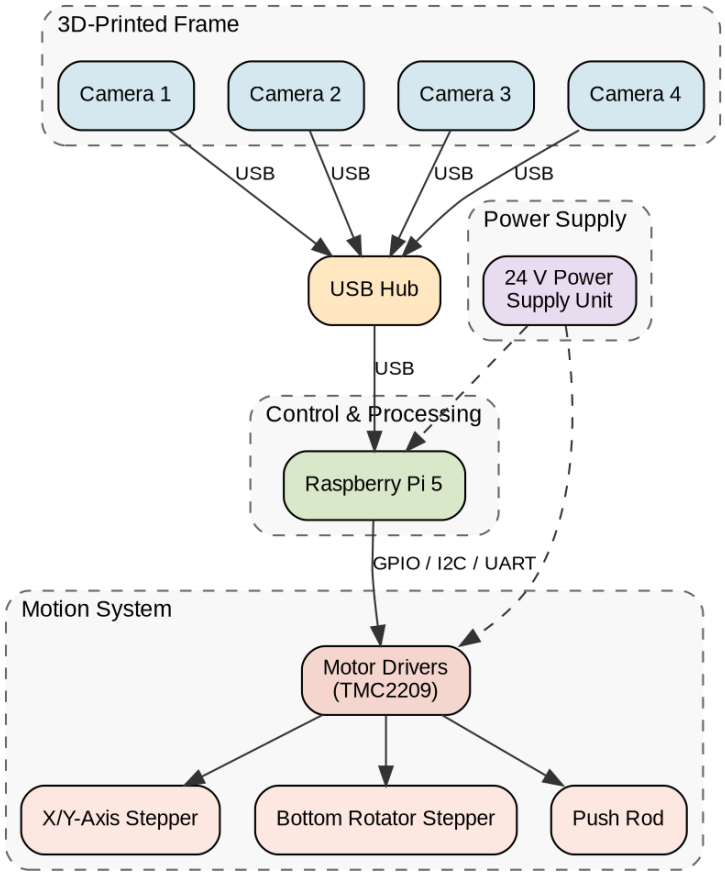

Figure 5. System block diagram showing data and power connections between the vision, processing, motion, and power subsystems.

The motion-control electronics use a layered control architecture. A Raspberry Pi 5 performs high-level processing, including multi-camera image acquisition, tower-state reconstruction, and decision making. An Arduino Uno R3 serves as the low-level real-time motion controller. The Raspberry Pi communicates with the Arduino through USB serial, and the Arduino generates STEP/DIR control signals for the motor drivers.

The motor drivers are based on the TMC2209 stepper driver. A custom PCB was designed around the TMC2209 module to simplify wiring and improve integration. The PCB provides a 24 V motor power input, motor terminal outputs, logic power input, and STEP/DIR control terminals. A local 220 µF capacitor is included on the motor supply side to improve supply stability during current transients.

2.2 Subsystem 1: Vision Subsystem (Cameras + Image Processing)

Description: Four OV9732 cameras (720p, 30 fps; operating at 1280×720) are mounted at the four corners of the 3D-printed frame and connected to a Raspberry Pi 5 via a USB hub. Each camera provides an independent viewpoint of the Jenga tower at a working distance of approximately 20 cm.

Each Jenga brick is affixed with 10 AprilTags (tag36h11 family), for a total of 300 tags coded from 1 to 300. Tags are distributed across the brick's faces to ensure visibility from multiple angles. Each camera runs an independent AprilTag detection pipeline using the apriltag Python library

(quad_decimate=2.0, nthreads=2), identifying each tag's ID, pixel center, and four corner coordinates. Detection results are served via a Flask HTTP API on each camera stream.

For calibration, we fixed four AprilTags (301–304) to the bottom turntable surface. Whenever the system starts, the cameras use these tags to automatically update the K-matrix and location information individually by capturing a fixed number of pictures and computing camera parameters in Python, ensuring the robustness of the vision system.

The system uses tag ID presence/absence across all four camera views to determine which bricks remain in the tower and at which layer. This information is fed into a MuJoCo simulation that maintains a real-time digital twin of the tower state, enabling the strategy algorithm to evaluate structural stability and select the optimal brick for removal.

Interaction with other subsystems: The vision subsystem outputs the reconstructed tower state (brick presence per layer) to the strategy algorithm running on the Raspberry Pi 5. The strategy module then computes target coordinates for the three-axis motion system.

Requirements & Verification:

#	Requirement	Verification
1	Each camera shall run the AprilTag detection pipeline at ≥30 fps at 1280×720 resolution.	Log per-frame processing timestamps over 1,000 consecutive frames; the average frame interval shall be ≤30 ms.
2	Each camera shall correctly identify ≥90% of visible AprilTags per frame at 20 cm working distance.	Place a fully tagged tower and compare detected tag IDs against ground truth across 500 frames per camera; detection rate shall be ≥90%.
3	The union of all four cameras shall achieve ≥99% tag coverage on an intact 30-brick tower.	Capture simultaneously from all four cameras on an intact tower over 100 cycles; the union of detected tag IDs shall include ≥99% of the 300 tags.
4	End-to-end latency from image capture to updated MuJoCo tower model shall be ≤50 ms.	Insert timestamps at frame capture and at MuJoCo state update; measure over 500 cycles; 95th-percentile latency shall be ≤50 ms.

2.3 Subsystem 2: Three-Axis Motion Control System

Description:

The motion subsystem now uses a purchased two-dimensional linear motion platform as the X-Y stage, combined with a vertically mounted linear module to form the full three-axis Cartesian gantry. The X-Y platform provides horizontal positioning around the Jenga tower, while the Z-axis adjusts the height of the end-effector relative to the selected block layer. Compared with a fully custom 3D-printed motion stage, the purchased platform provides better rail alignment, higher mechanical stiffness, and lower assembly uncertainty.

During recent hardware development, the team completed the initial assembly of the X-Y base platform and installed the vertical linear axis onto the moving carriage. The current prototype confirms that the main motion hardware can be physically integrated with the planned Jenga workspace. The next mechanical tasks are to complete cable routing, verify motor direction, check travel limits, measure repeatability, and mount the final pushing/gripping end-effector on the Z-axis carriage.

Interaction with other subsystems: The motion control subsystem receives target (x, y, z) coordinates from the Raspberry Pi after image processing and block-selection logic. It transports the end-effector to the designated block position and coordinates with the end-effector for the push, extract, grip, transport, and placement sequence. It also depends on the 3D-printed structural frame for stiffness, axis alignment, and mounting support for motors, rails, and cameras.

Requirements & Verification:

#	Requirement	Verification
1	The three-axis motion system shall achieve a repeatable positioning accuracy of ±1.0 mm over the full operating workspace.	Command the gantry to move to a set of known reference positions distributed across the workspace and measure the actual end-effector position using a ruler, calipers, or fixture-based reference points. Repeat for 10 trials and record the maximum positioning error.
2	The motion platform shall achieve a maximum travel speed of at least 80 mm/s on the X- and Y-axes under no-load or nominal-load conditions.	Command a full-range traversal over a known distance and measure the travel time using video timestamps or a stopwatch. Compute average speed over 5 trials and verify that the target speed is met without missed steps.
3	The frame and motion system shall support a combined moving payload of at least 0.8 kg (motors, carriage, and end-effector assembly) while maintaining static deflection below 2.0 mm at the end-effector.	Mount the nominal payload on the moving carriage, position the system at a representative extended location, and measure end-effector deflection relative to an unloaded baseline using a ruler, caliper, or dial indicator. Perform the test at 3 representative positions within the workspace.

2.4 Subsystem 3: Operation Unit (End-Effector)

Description: A specialized mechanism mounted on the Z-axis carriage capable of executing complex physical interactions with Jenga blocks, including pushing a block laterally out of the tower and grasping/translating it to the top.

Mechanical design of the end-effector:

The end-effector adopts a hybrid push-and-grip design mounted on the Z-axis carriage. It includes a slender pushing element for laterally displacing a selected Jenga block and a compact two-sided gripping mechanism for securing the partially extracted block and transporting it to the top of the tower. This hybrid approach was chosen because a Jenga move consists of two distinct mechanical tasks: first, overcoming static friction to push a block out of the tower, and second, stabilizing and relocating the block without slip or excessive tower disturbance.

Actuation method:

The end-effector uses electric actuation, with compact actuators integrated into the carriage assembly. A dedicated actuator drives the pushing motion, while a separate small actuator drives the gripping mechanism. This configuration avoids the complexity of pneumatic systems and is more suitable for a compact student-built prototype.

How it handles both pushing and grasping:

During operation, the end-effector first aligns the push rod with the centerline of the target block

and applies a controlled lateral push to partially extract the block. Once sufficient block length is exposed, the gripping mechanism closes on the block and secures it for transport. The gantry then lifts or repositions the block and places it onto the top layer of the tower. Separating the push and grip functions improves mechanical reliability and reduces the chance of tower collapse caused by a single overly complicated mechanism.

Force control strategy:

The current design primarily uses a conservative open-loop force strategy based on calibrated displacement, limited motor current, and controlled speed rather than active closed-loop force sensing. To reduce the risk of toppling the tower, the system uses a low-speed initial push, restricts the maximum commanded push distance per attempt, and avoids sudden acceleration. If resistance exceeds the expected range, the controller can stop the motion and reattempt or select another block. This approach is simpler to implement and appropriate for the current prototype stage.

Requirements & Verification:

#	Requirement	Verification
1	The end-effector shall provide a controlled lateral push force of up to 5 N, with repeatable operation within ±1 N under bench-test conditions.	Measure the pushing force using a force gauge or spring scale during linear actuation tests. Perform at least 5 trials and record the peak applied force and variation.
2	The gripper shall securely hold and transport one Jenga block (90 × 30 × 30 mm) without visible slippage or dropping during motion.	Conduct a grip-and-transport test over 10 cycles, including pick-up, short-distance transport, and release. Record whether any slip, drop, or misalignment occurs.
3	The end-effector shall complete the push–extract–grip–place sequence for one block within 8 s after reaching the target position.	Perform a timed functional test over 5 trials, starting when the gantry reaches the target block and ending when the block is placed at the top position. Compute the average completion time.

2.5 Subsystem 4: Central Controller (Raspberry Pi + Strategy Algorithm)

Description: The Raspberry Pi serves as the central controller. It receives multi-camera image data, runs the vision pipeline to build a tower state model, executes a predefined game strategy algorithm to select the optimal block, computes the corresponding (x, y, z) coordinates and motion sequence, and sends commands to the motor drivers. It also manages game-flow logic: detecting the human opponent's move completion, handling rule violations (e.g., the human collapsing the tower or attempting a second move), and signaling turn transitions.

Interaction with other subsystems: Receives data from the vision subsystem; sends motion commands to the three-axis system and operation unit; may include user-facing indicators (LEDs, buzzer, display) for game state.

The central controller of the system is a Raspberry Pi 5, which serves as the high-level processing and decision-making unit for JengaBot. It is responsible for receiving image data from the four cameras, reconstructing the current tower state, selecting a candidate block for manipulation, and sending motion commands to the low-level motion controller. To ensure reliable real-time motor actuation, the Raspberry Pi does not directly generate motor pulses. Instead, it communicates with an Arduino Uno R3 through USB serial, while the Arduino handles deterministic low-level

motor control.

From the software perspective, the Raspberry Pi runs the full perception-to-action pipeline. First, it collects multi-camera observations of the tower. These observations are processed to identify visible AprilTags and reconstruct the configuration of the Jenga tower. The reconstructed result is then used to build or update a digital model of the tower, which is used by the strategy module to determine a suitable next move. After a target block and motion objective are chosen, the Raspberry Pi converts the decision into motion commands and transmits them to the Arduino. The Arduino then drives the TMC2209-based motor control stage using STEP/DIR signaling and reports execution status back to the Raspberry Pi.

This split-controller architecture was chosen for both computational and reliability reasons. The Raspberry Pi 5 is well suited for image processing, state estimation, and strategy computation, but Linux-based systems are not ideal for precise real-time pulse timing. By delegating time-critical actuation to the Arduino Uno R3, the design reduces timing jitter and improves motion robustness while keeping the software architecture modular and easier to debug.

The controller also plays a coordination role across the entire system. It synchronizes the vision subsystem, maintains the tower-state model, issues motion targets, and monitors execution progress through serial status feedback. In this way, the Raspberry Pi acts as the system-level supervisor, while the Arduino functions as a dedicated motion execution unit.

Requirements & Verification:

#	Requirement	Verification
1	The Raspberry Pi 5 shall successfully transmit valid motion commands to the Arduino Uno R3 and receive acknowledgment/status feedback over USB serial for at least 30 minutes of continuous operation.	Run a scripted communication test for 30 minutes and log all transmitted commands and returned responses; verify zero communication failures or unhandled timeouts.
2	The central controller shall generate a complete command sequence for a selected manipulation action after receiving a reconstructed tower state.	Provide representative tower-state inputs and verify that the software outputs a valid target motion sequence for each case.
3	The controller architecture shall isolate high-level processing from low-level pulse generation such that motor execution remains functional during high CPU usage on the Raspberry Pi.	Run the vision/strategy pipeline while executing repeated motor commands and verify that the Arduino continues to execute and report motion correctly.

2.6 Subsystem 5: Power Supply

The motor drive stage is powered by a 24 V / 6 A switching power supply. This supply powers the TMC2209 driver boards and the associated stepper motors. The use of a regulated 24 V supply provides sufficient voltage headroom for stable stepper operation while maintaining a compact and practical laboratory power architecture.

A dedicated bulk capacitor is placed on the motor power input of the custom TMC2209 driver PCB to suppress transient voltage fluctuations caused by motor current changes. This improves electrical stability and helps protect both the driver and the rest of the control system.

Requirements & Verification:

#	Requirement	Verification
1	The motor power system shall provide 24 V to the driver stage under nominal operating conditions.	Measure the supply voltage at the PCB input terminals during motor operation.
2	The driver PCB shall correctly receive STEP/DIR control signals and produce corresponding motor motion.	Apply known pulse sequences from the Arduino and verify motor direction and displacement.
3	The motor driver power stage shall remain electrically stable during repeated motion cycles.	Run repeated start–stop motion tests and verify no reset, missed motion, or abnormal driver shutdown occurs.

2.7 Tolerance Analysis

To evaluate the feasibility of the mechanical subsystem, a tolerance analysis was conducted for the motion platform and end-effector. The analysis focuses on two critical questions:

4. whether the positioning resolution of the Cartesian gantry is sufficient for targeting individual Jenga blocks, and
5. whether the end-effector can provide enough pushing force to extract a block without introducing excessive disturbance to the tower.

(1) Positioning Accuracy Analysis

Each custom Jenga block used in this project has nominal dimensions of **30 mm × 30 mm × 90 mm**. Therefore, the motion system must position the end-effector with an error significantly smaller than **30 mm**, so that the pusher or gripper can align reliably with the target block face. The mechanical design target for this project is a repeatable positioning accuracy of **±1.0 mm**.

For a stepper-driven axis, the theoretical linear resolution can be estimated by:

$$\textit{Linear resolution} = \textit{travel per revolution} / (\textit{steps per revolution} \times \textit{microsteps})$$

For a standard NEMA17 stepper motor with a 1.8° step angle:

$$\textit{steps per revolution} = 360 / 1.8 = 200$$

If the driver uses 16× microstepping, the effective command resolution becomes:

$$200 \times 16 = 3200 \textit{ microsteps/rev}$$

Assuming a representative axis travel of 40 mm per revolution, the theoretical linear resolution is:

$$40 / 3200 = 0.0125 \textit{ mm per microstep}$$

This value is much smaller than the required system-level positioning tolerance of ±1.0 mm. In practice, the actual positioning error will be dominated not by step resolution, but by mechanical factors such as frame flexibility, backlash, rail alignment, print tolerance, and vibration. Therefore, although the theoretical resolution is very fine, a realistic working accuracy target of ±1.0 mm is appropriate for the current prototype.

Compared with a **30 mm** block width, a ±1.0 mm positioning error corresponds to only:

$$1.0 / 30 = 3.33\%$$

So the positioning error is about **3.3% of the block width**, which is acceptable for aligning the pusher or gripper with the center region of the target block.

In addition, if the target alignment region is taken as the central 10–15 mm portion of the block face, a ±1.0 mm positioning error still leaves sufficient margin for controlled contact without striking the block edge. Therefore, the motion subsystem remains mechanically suitable for the larger custom block geometry.

(2) Push Force Feasibility Analysis

The end-effector must push a selected block laterally out of the tower while avoiding sudden impact that could destabilize the structure. The required extraction force mainly depends on block-to-block friction and local contact conditions. A simplified estimate can be made using Coulomb friction:

$$F = \mu N$$

where μ is the effective static friction coefficient between wooden blocks and N is the normal force acting on the target block due to the weight of the upper layers and local contact compression.

Because the exact contact state varies from game to game, a conservative design range is more appropriate than a single exact number. For the current prototype, the end-effector is designed to provide a controlled push force of up to 5 N. This value is intentionally higher than the expected extraction force of a typical loose or moderate-resistance Jenga block, while still low enough to avoid aggressive impact loading.

To improve safety, the robot does not apply the full force instantly. Instead, the control strategy uses:

- a low-speed initial push;
- limited actuator current;
- restricted push stroke per attempt;
- reattempt/abort logic if the resistance appears too large.

This means the system relies not only on maximum available force, but also on conservative motion planning to reduce tower disturbance.

Conclusion of Tolerance Analysis

The tolerance analysis indicates that the motion subsystem has sufficient theoretical positioning resolution to target individual Jenga blocks, and the practical design goal of ±1.0 mm is reasonable for the current prototype. In addition, the end-effector's planned push capability of up to 5 N, combined with low-speed open-loop control, is adequate for experimental block extraction without requiring an excessively aggressive actuation strategy. Therefore, the mechanical subsystem is considered feasible for prototype-level Jenga manipulation.

3. Cost

Name	Description	Qty	Unit ($)	Total ($)
Raspberry Pi 5 (rental)	High-level processor for vision, reconstruction, and strategy	1	50.00	50.00
Arduino Uno R3	Low-level real-time motion controller	1	15.00	15.00

Name	Description	Qty	Unit ($)	Total ($)
TMC2209 Driver Module	Stepper motor driver module for axis control	4	8.00	32.00
Custom Driver PCB	Custom PCB for TMC2209 integration and wiring	4	5.00	20.00
24 V Switching Power Supply	Motor power supply, 24 V / 6 A	1	25.00	25.00
USB cables / interconnect wiring	USB serial link, signal wires, and general cabling	1	10.00	10.00
Connectors / terminal blocks / passives	Headers, screw terminals, capacitors, and small support components	1	15.00	15.00
Micro electric push rod	Push rod for displacing wooden blocks	1	145.80	145.80
2D motion platform	Purchased planar motion platform for X-Y positioning	1	2,500.00	2,500.00
Jenga blocks	Customized Jenga bricks	30	15.00	450.00
Cameras	OV9732 camera, 720p / 30 fps	4	60.00	240.00

4. Schedule

Week	Wangyihan Guo	Peiran Wei	Jiacheng Ye	Hengtie Zhu
3.2	Find correct simulator	Check camera arguments	Develop initial mechanical concept for frame and motion layout	Develop initial mechanical concept for frame and motion layout
3.9	Find platform	Build recognizing-system demo	Build CAD concept for 3D-printed frame and motion mounting structure	Build CAD concept for 3D-printed frame and motion mounting structure
3.16	Buy and test Pi 5	Validate system robustness on a single camera	Selection of XY-axis motion platform	3D-diagram modeling
3.23	3D reconstruction	Determine detailed Jenga brick size and label accuracy	3D-diagram modeling	Integrate end-effector with Z-axis carriage and verify basic mechanical motion

Week	Wangyihan Guo	Peiran Wei	Jiacheng Ye	Hengtie Zhu
3.30	Try to simulate tower	Validate system robustness on all cameras	Turntable design	Turntable design
4.6	Finish simulator building	Use 3D-printed substitute bricks to validate latency	Selection of push-rod actuator	Selection of push-rod actuator
4.13	Test motors	Develop demo control algorithm in simulation	Fixing-component design	Fixing-component design
4.20	Print PCB board	Develop demo control algorithm in simulation	3D printing	3D printing
4.27	Test 3-axis platform	Use 3D-printed substitute bricks to validate / tune algorithm	Align vertical axis and check mechanical mounting	Align vertical axis and check mechanical mounting
5.4	Apply running algorithm	Tune algorithm on real Jenga bricks	Measure travel range and preliminary repeatability	Measure travel range and preliminary repeatability
5.11	—	Tune algorithm on real Jenga bricks	First block-contact and extraction test	First block-contact and extraction test
5.18	Final report	Final report	Final report	Final report

5. Ethics and Safety

5.1 Ethics

A suitable ethical framework for JengaBot is the **ACM Code of Ethics and Professional Conduct**, which states that computing professionals should contribute to society and human well-being, avoid harm, be fair, respect privacy, and give proper consideration to the public good. The ACM states that the Code is intended to guide current and aspiring practitioners, instructors, students, and others who use computing technology in impactful ways. These principles are directly relevant to JengaBot because the system uses cameras, algorithmic decision-making, and autonomous physical interaction in a shared human environment.

For this project, the main ethical considerations are:

- **Privacy and image handling.** Since the system uses multiple cameras to observe the Jenga tower and nearby gameplay area, there is a possibility of capturing images of human participants. To reduce privacy concerns, the system should only collect images needed for tower-state recognition, avoid unnecessary storage of personal images, and

restrict recorded data to project development or demonstration use. These precautions are consistent with the ACM principles of respecting privacy and avoiding harm.

- **Responsible autonomous interaction.** JengaBot is not only a sensing system but also a robot that moves mechanical parts near human users. Therefore, the team must prioritize conservative motion behavior, predictable operation, and safe testing procedures. The ethical goal is not simply to make the robot work, but to make sure the robot does not create unreasonable risk during demonstrations or use. This aligns with the idea that the public good and harm prevention should take priority over technical performance alone.
- **Fairness and accessibility.** The system should interact with human players in a transparent and understandable way. For example, the robot's actions should be observable, the gameplay process should not rely on hidden intervention, and users should be able to understand when the robot is sensing, moving, or waiting. This improves fairness in interaction and supports accessibility in a mixed human–machine setting. The ACM Code's fairness and public-good principles support this design direction.

Overall, the project team treats ethics as part of the design process rather than a separate afterthought. In JengaBot, privacy-aware sensing, conservative robot behavior, and fair interaction are all necessary to ensure that the system is technically effective and ethically responsible.

5.2 Safety

The JengaBot prototype includes moving gantry axes, an end-effector, electrical power components, and a 3D-printed frame. Therefore, safety considerations must address both **mechanical hazards** and **electrical hazards**. OSHA states that moving machine parts can cause serious injuries and that hazardous machine parts must be safeguarded; OSHA's general machine-guarding requirements also state that one or more guarding methods should be provided to protect operators from hazards created by point of operation, ingoing nip points, and rotating parts. These principles are directly applicable to the JengaBot motion platform and end-effector.

For this project, the main safety considerations are:

- **Pinch hazards from the three-axis gantry and end-effector.** The linear axes, carriage interfaces, and end-effector create potential pinch points during motion. To reduce this risk, moving regions should be visually identified, unnecessary human access to the active workspace should be limited during operation, and the robot should run at conservative speeds during testing and demonstrations. Fixed covers or physical separation around the most hazardous moving areas are recommended where practical. These measures are consistent with OSHA machine-guarding guidance.
- **Electrical safety of the power supply and control system.** The project uses a 24 V switching power supply, motor drivers, and control boards. Good practice requires insulated wiring, secure terminal connections, strain relief for cables, and enclosure of exposed conductors. For electronic and ICT-related equipment, IEC/UL 62368-1 is a widely used safety framework for evaluating electrical hazards and related protection methods; while this student prototype is not a certified commercial product, the design should still follow the same basic safety mindset.
- **Emergency stop and controlled shutdown.** The robot should include a simple and accessible emergency stop or power-cutoff method so that motion can be stopped immediately if unsafe behavior occurs. Even in a prototype system, manual interruption capability is important because autonomous motion and actuator faults can otherwise create avoidable risk. This is especially relevant during subsystem debugging and

integrated demonstrations. OSHA's guarding philosophy emphasizes hazard elimination or control whenever accidental contact or operation could injure nearby users.

- **Structural integrity of the 3D-printed frame.** Because the frame supports motors, guide elements, cameras, and the end-effector, it must remain mechanically stable under load. Printed parts should be designed with reasonable tolerances, sufficient wall thickness, and reinforced mounting locations. Before full operation, the frame should be checked for looseness, visible deformation, and alignment drift. This is particularly important because structural deflection can reduce positioning accuracy and may also create secondary safety risks during motion.
- **Mechanical assembly and platform testing.** Since the purchased X-Y platform and vertical Z-axis contain exposed moving carriages, guide rails, and motor-driven components, the team shall keep hands away from the motion range whenever the system is powered. Mechanical adjustment, screw tightening, and cable routing should be performed only when motor power is disconnected. Cables must be routed away from moving rails and carriages to prevent dragging, entanglement, or accidental disconnection. During early tests, the platform should operate at reduced speed, and an accessible power cutoff should be available before full automatic motion is attempted.

References

[1] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, May 2011, pp. 3400–3407.