ECE445
SENIOR DESIGN LABORATORY

DESIGN DOCUMENT

JengaBot — A Robotic System for Playing Jenga with Human

Team Members:
Wangyihan Guo
Peiran Wei
Jiacheng Ye
Hengtie Zhu

Advisor: Prof. Pavel Loskot

April 8, 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.

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. 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.

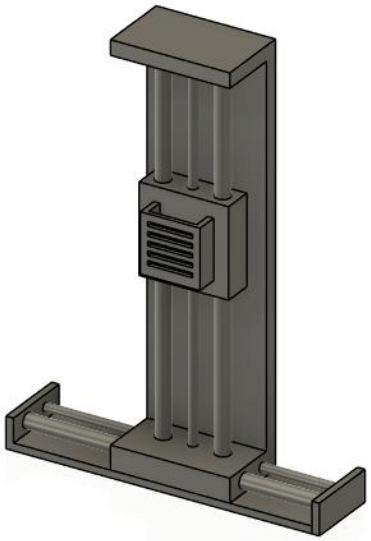

Figure 1 & 2. CAD rendering / system-level sketch of JengaBot

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

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 Vision Subsystem (Cameras + Image Processing)

Four OV9732 cameras (720p, 30 fps; operating at 640×480) 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.

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.

Requirement	Verification
Each camera shall run the AprilTag detection pipeline at ≥20 fps at 640×480 resolution.	Log per-frame processing timestamps over 1,000 consecutive frames; average frame interval shall be ≤50 ms.
Each camera shall correctly identify ≥90% of visible AprilTags per frame at 20 cm working distance.	Place a fully tagged tower, compare detected tag IDs against ground truth across 500 frames per camera; detection rate shall be ≥90%.
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.
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 Three-Axis Motion Control System

A three-axis (X, Y, Z) Cartesian gantry system moves the operation unit precisely within the constrained workspace defined by the 3D-printed frame. Each axis is driven by a 17HS4401 NEMA17 stepper motor and controlled through TMC2209 stepper driver modules mounted on a custom driver PCB. The system receives target coordinates from the Raspberry Pi after strategy computation, while the Arduino Uno handles low-level motion execution and synchronized point-to-point positioning. The Cartesian architecture was selected because it provides a simple mechanical layout, straightforward coordinate mapping, and sufficient positioning repeatability for Jenga block manipulation.

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.

Requirement	Verification
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 known reference positions and measure actual position using calipers. Repeat for 10 trials and record max error.
The motion platform shall achieve a maximum travel speed of at least 80 mm/s on the X- and Y-axes.	Command a full-range traversal and measure travel time. Compute average speed over 5 trials; verify target speed without missed steps.
The frame and motion system shall support a combined moving payload of at least 0.8 kg while maintaining static deflection below 2.0 mm.	Mount nominal payload on carriage, measure deflection at 3 representative positions within workspace.

2.4 Operation Unit (End-Effector)

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.

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.

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.

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.

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. The system uses a low-speed initial push, restricts the maximum commanded push

distance per attempt, and avoids sudden acceleration.

Requirement	Verification
The end-effector shall provide a controlled lateral push force of up to 5 N, with repeatable operation within ±1 N.	Measure pushing force using a force gauge during linear actuation tests. Perform at least 5 trials.
The gripper shall securely hold and transport one Jenga block (90×30×30 mm) without slippage.	Conduct grip-and-transport test over 10 cycles including pick-up, transport, and release.
The end-effector shall complete the push-extract-grip-place sequence within 8 s after reaching target position.	Perform timed functional test over 5 trials. Compute average completion time.

2.5 Central Controller (Raspberry Pi + Strategy Algorithm)

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, and signaling turn transitions.

The central controller of the system is a Raspberry Pi 5, which serves as the high-level processing and decision-making unit. An Arduino Uno R3 serves as the low-level real-time motion controller. The Raspberry Pi communicates with the Arduino through USB serial, while the Arduino handles deterministic low-level motor control.

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.

Requirement	Verification
The Raspberry Pi 5 shall successfully transmit valid motion commands to the Arduino Uno R3 and receive feedback over USB serial for at least 30 minutes of continuous operation.	Run a scripted communication test for 30 minutes; verify zero communication failures.
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 valid target motion sequence output.
The controller architecture shall isolate high-level processing from low-level pulse generation such that motor execution remains functional during high CPU usage.	Run vision/strategy pipeline while executing repeated motor commands; verify Arduino continues correctly.

2.6 Power Supply

The motor drive stage is powered by a 24V/6A switching power supply. This supply powers the TMC2209 driver boards and the associated stepper motors. A dedicated bulk capacitor is placed on the motor power input of the custom TMC2209 driver PCB to suppress transient voltage fluctuations.

Requirement	Verification
The motor power system shall provide 24V to the driver stage under nominal operating conditions.	Measure the supply voltage at the PCB input terminals during motor operation.
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.
The motor driver power stage shall remain electrically stable during repeated motion cycles.	Run repeated start-stop motion tests; verify no reset, missed motion, or abnormal driver shutdown.

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: (1) whether the positioning resolution of the Cartesian gantry is sufficient for targeting individual Jenga blocks, and (2) 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 has nominal dimensions of 30 mm × 30 mm × 90 mm. The motion system must position the end-effector with an error significantly smaller than 30 mm. The mechanical design target is a repeatable positioning accuracy of ±1.0 mm.

For a stepper-driven axis, the theoretical linear resolution can be estimated by: Linear resolution = travel per revolution / (steps per revolution × microsteps). For a standard NEMA17 stepper motor with a 1.8° step angle: steps per revolution = 200. With 16× microstepping: 200 × 16 = 3200 microsteps/rev. Assuming 40 mm per revolution, the theoretical linear resolution is 40 / 3200 = 0.0125 mm per microstep.

This value is much smaller than the required ±1.0 mm tolerance. In practice, actual positioning error will be dominated by mechanical factors such as frame flexibility, backlash, rail alignment, print tolerance, and vibration. A ±1.0 mm positioning error corresponds to only 3.3% of the block width, which is acceptable for aligning the pusher or gripper.

(2) Push Force Feasibility Analysis

The required extraction force mainly depends on block-to-block friction and local contact conditions. A simplified estimate uses Coulomb friction: $F = \mu N$, where μ is the effective static friction coefficient between wooden blocks and N is the normal force from upper layers.

The end-effector is designed to provide a controlled push force of up to 5 N. The control strategy uses low-speed initial push, limited actuator current, restricted push stroke per attempt, and reattempt/abort logic if resistance appears too large.

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. 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. The mechanical subsystem is considered feasible for prototype-level Jenga manipulation.

3. Cost

Name	Description	Qty	Unit Cost	Total Cost
Raspberry Pi 5	High-level processor for vision, reconstruction, and strategy	1	1300	1300
Arduino Uno R3	Low-level real-time motion controller	1	15	15
TMC2209 Driver Module	Stepper motor driver module for axis control	4	8	32
Custom Driver PCB	Custom PCB for TMC2209 integration and wiring	4	5	20
24V Switching Power Supply	Motor power supply, 24V/6A	1	25	25
USB Cables / Interconnect Wiring	USB serial link, signal wires, and general cabling	1	10	10
Connectors / Passive Components	Headers, screw terminals, capacitors, and small support components	1	15	15
Micro electric push rod	Push the Jenga blocks	1	145.8	145.8
2D Motion Platform	Purchased planar motion platform for X-Y positioning	1	2500	2500
Jenga blocks (custom)	Customized Jenga bricks	30	15	450
OV9732 Cameras (720p)	OV9772 camera with 720p30fps	4	60	240

4. Schedule

<table>
<tr><th>Week</th><th>Wangyihan Guo</th><th>Peiran Wei</th><th>Jiacheng Ye</th><th>Hengtie Zhu</th></tr>
<tr><td>3.2</td><td>Find correct simulator</td><td>Checkout camera arguments</td><td colspan="2">Develop initial mechanical concept</td></tr>
<tr><td>3.9</td><td>Find platform</td><td>Build recognizing system demo</td><td colspan="2">Build CAD concept for frame</td></tr>
<tr><td>3.16</td><td>Buy and test Pi5</td><td>Validate single camera robustness</td><td>Selection of XY-axis platform</td><td>3D diagram modeling</td></tr>
<tr><td>3.23</td><td>3D reconstruction</td><td>Determine brick size and label accuracy</td><td>3D diagram modeling</td><td>Integrate end-effector with Z-axis</td></tr>
<tr><td>3.30</td><td>Try to simulate tower</td><td>Validate all cameras</td><td colspan="2">Turntable design</td></tr>
<tr><td>4.6</td><td>Finish simulator building</td><td>Use 3D-printed bricks to validate latency</td><td colspan="2">Selection of push rod actuator</td></tr>
<tr><td>4.13</td><td>Test motors</td><td rowspan="2">Develop demo control algorithm</td><td colspan="2">Fixing component design</td></tr>
<tr><td>4.20</td><td>Print PCB board</td><td colspan="2">3D printing</td></tr>
<tr><td>4.27</td><td>Test 3-axis platform</td><td>Validate/tune algorithm with 3D bricks</td><td>Physical assembly</td><td>Physical assembly</td></tr>
<tr><td>5.4</td><td colspan="4" rowspan="2">Apply/tune algorithm and mechanical parts</td></tr>
<tr><td>5.11</td></tr>
<tr><td>5.18</td><td colspan="4">Final Report</td></tr>
</table>

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. These principles are directly relevant to JengaBot because the system uses cameras, algorithmic decision-making, and autonomous physical interaction in a shared human environment.

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. 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.

Responsible autonomous interaction: JengaBot is a robot that moves mechanical parts near human users. The team must prioritize conservative motion behavior, predictable operation, and safe testing procedures.

Fairness and accessibility: The system should interact with human players in a transparent and understandable way. 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.

Overall, the project team treats ethics as part of the design process rather than a separate afterthought. Privacy-aware sensing, conservative robot behavior, and fair interaction are all necessary to ensure 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. Safety considerations must address both mechanical hazards and electrical hazards.

Pinch hazards from the three-axis gantry and end-effector: The linear axes, carriage interfaces, and end-effector create potential pinch points during motion. Moving regions should be visually identified, human access limited during operation, and the robot should run at conservative speeds during testing and demonstrations.

Electrical safety: 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.

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.

Structural integrity of the 3D-printed frame: The frame supports motors, guide elements, cameras, and the end-effector and must remain mechanically stable under load. Printed parts should be designed with reasonable tolerances, sufficient wall thickness, and reinforced mounting locations.

References

[1] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pages 3400–3407, May 2011.