MaGomuku

Authors: Sizhe Chen, Shuailin Pan, Zipiao Wan
Affiliation: Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—An automated, networked gomoku set that is capable of automatically feeding and placing gomoku pieces onto a physical game board while playing with online players. While similar products focus on gameplay by detecting game pieces and incorporating AI for education purposes, our system aims to provide an automated gameplay experience to the user by integrating feeding, detection, locking and movement to automatically place the opponent's pieces during each turn.**

***Index Terms*— 3D Modeling, Gomoku, Hall Effect Sensors, Magnets, PCB Fabrication, Robotics, Web Application, X-Y Stepper Motor Gantry**

1 INTRODUCTION

In an era where digital interaction often replaces physical presence, the longing for tangible experiences, especially in the realm of games, has become increasingly pronounced. Gomoku, a traditional board game known for its simple rules yet deep strategic complexity, has been a source of enjoyment and mental challenge across cultures for centuries. However, with the rising trend of online gaming, enthusiasts of this classic game find themselves yearning for an experience that merges the tactile satisfaction of moving physical pieces with the convenience and global connectivity of digital play. This intersection of needs presents a unique opportunity for innovation—the development of "Magomoku."

Magomoku is an automated, networked Gomoku set designed to bring the best of both worlds to enthusiasts and casual players alike. The project transforms the traditional Gomoku board into a dynamic, interactive platform where players can engage with opponents from anywhere in the world through a physical board. At the heart of Magomoku lies an advanced x-y stepper motor movement gantry, equipped with magnets to pick up and place game pieces accurately on the board, simulating the opponent's moves in real-time. This system is complemented by hall effect sensors that detect and differentiate between game pieces, ensuring a seamless and intuitive gameplay experience.

Unlike existing technologies that focus primarily on the educational aspects of board games, utilizing LED indicators and AI to teach game strategies, Magomoku distinguishes itself by emphasizing the physicality of the gaming experience. Our approach not only caters to those seeking to learn Gomoku but more importantly, aims to satisfy the desire for a genuine, tactile gaming experience that online platforms alone cannot offer. By automating the movement of game pieces and integrating a web application for global play, Magomoku bridges the gap between the traditional and the digital, offering a novel solution to players worldwide. The primary goal of Magomoku is to recreate the authentic feel of playing Gomoku with a friend or family member across a physical board, without the constraints of geographical distance.

This project aspires to enhance the social aspect of gaming, enabling users to connect, compete, and share moments of joy and challenge through a beautifully engineered, interactive platform. By doing so, Magomoku aims to set a new standard for remote interactive board gaming, providing a unique solution that leverages the best of mechanical engineering, software development, and user experience design to enrich the world of board gaming.

2 USE-CASE REQUIREMENTS

MaGomuku, the physical, networked, automated Gomoku set is designed to provide to a diverse range of users, including Gomoku enthusiasts seeking an elevated gaming experience, individuals desiring to enjoy the game physically with online friends, elderly users unfamiliar with online Gomoku games, tech enthusiasts eager to explore magnetic levitation devices, and those seeking an engaging pastime.

Our requirements are mostly focused on ensuring an accurate and responsive game state detection system, an accurate and efficient automatic piece feeding, and a easy-to-use software interface to ensure that the user can have a interesting, care-free and consistent gameplay experience using our MaGomoku board.

To ensure that the recorded game state is always up-to-date and can quickly indicate any illegal move to the user, we need to ensure that:

- The hardware detection system can distinguish black and white pieces with high accuracy.
- The hardware detection system can reflect the actual game state with high accuracy.
- The software detection system can catch illegal moves with high accuracy.

To ensure that the game piece can be accurately and efficiently placed at the designated position, we need to ensure that:

- The feeding system can feed within 3 seconds.
- The movement system can move the piece to the required position in 10 seconds.
- The feeding system can feed to a position within 5mm of target location.
- The movement system can move the piece to a position no farther than 5 mm of the target position

Overall, we want to ensure the entire automatic turn finishes within 13 seconds.

3 ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

Our entire system comprises two main parts: the physical chessboard device and the web application. The physical chessboard device is responsible for interacting with local users, while the web application is responsible for exchanging information and ensuring the integrity of the game state.

The chessboard device comprises four main parts: the movement system, feeding system, and detection/locking system. The movement system is primarily responsible for moving pieces on the board. Upon receiving input, the processing unit, Arduino Uno WiFi R4, translates voltage information into human-interpretable XY location data and converts movement instructions into voltage signals. To control the movement system, a CNC shield and A4998 stepper motor driver are utilized for direct control based on existing library. The feeding system handles the dispensing of game pieces onto the board before each automatic movement. The detection/locking system is responsible for detecting the presence of pieces on the board and ensuring their secure placement.

In addition to the chessboard device, there is the web application responsible for checking board integrity and serving as a bridge between the physical device and remote users online. The web application, fronted with HTML and backed with Django, should be able to receive, process, and send game-related information with the processing unit on board to ensure the a seamless game play.

3.1 Principle of Operation

3.1.1 Local Player's turn

When a player places a game piece onto the board, the detection system senses the change in magnetic flux caused by the positioning of the new game piece. The circuit then reflects this information by providing voltage information for the processing unit to read. Once the information is received, the processing unit forwards it to the Web Application. Subsequently, the Web application will determine if the move is legal. If deemed legal, it updates the game state and notifies the remote user.

3.1.2 Remote Player's turn

When a remote player makes a decision, it must be legal as our online platform only accepts legal moves from remote users. The Web Application processes the move into instructions and sends the information to the processing unit. When it's the remote player's turn, the processing unit initiates the instruction by alerting the feeding system to feed a game piece onto the electromagnet component. Then, the movement system places the piece onto the desired location. The detection/locking system verifies the move by detecting whether the game piece is delivered to the desired location. In case of error, it alerts the player for manual correction. Once the remote player's move is completed and verified, the chessboard alerts the player for their turn.

4 DESIGN REQUIREMENTS

4.1 Detection & Board Integrity

The goal of MaGomuku's design is to provide a networked gameplay experience that closely simulates playing against a physical player. Therefore, it is important for our system to accurately detect game pieces and ensure a smooth gaming experience by consistently checking the board's integrity. We aim for a 95% accuracy rate in distinguishing between black and white pieces, as well as in detecting the presence of pieces on the board. Furthermore, we will utilize sensors and software to regularly monitor board integrity and alert players for manual correction.

4.2 Movement

For movement, our primary objective is to ensure that our system is reliable and can deliver game pieces accurately and consistently, with minimal error. We hope that our system can achieve a stable piece transportation with 90% success. By stable, we mean that the system is able to automatically place the piece onto the the desired location without human interference. We also hope that we can achieve an accurate piece landing within 5mm of the center of designated placement location.

Beyond reliability, the system should also be fast as long transportation time will make the game frustrating and the fore less entertaining. The longest travel instance of a game piece should not exceed 13 seconds.

4.3 User Experience

In order to provide a smooth and wholesome experience for the users, the system should be easy to use. The goals of our system include: Easy setup, Easy to play, and Low latency. For easy setup, the user should do nothing but connect to the web application to start a game of Gomoku. For ease of play, the player should only be responsible for his/her move. For low latency, excluding the factor of the

movement system, the whole system should be able to process the game state with a maximum latency of 1 second.

5 DESIGN TRADE STUDIES

5.1 Magnetic Levitation vs. Magnet Locking

Initially, we explored the possibility of using magnetic levitation technology for moving game pieces. This approach would levitate a piece using a magnet inside, drag it to its designated location, and then lower it onto the board. However, our research and experimental tests highlighted several feasibility issues: Size and Interference: Magnetic levitation kits are large (around 3 inches) and could interfere with nearby pieces. Additionally, the magnetic field from the levitated piece could disturb other pieces, leading to instability and potential dropping of the piece. Weight and Cost of Game Pieces: To achieve stable levitation, the pieces would need to be significantly larger and heavier than standard Gomoku pieces (2 cm in diameter), increasing costs and deviating from traditional game standards. Placement Precision: Positioning a piece on the levitation kit requires delicate manual adjustments due to the sensitivity to slight deviations from the center, making automated placement impractical without sophisticated control systems. Given these challenges, we opted for a magnet locking system. This system employs an electromagnet at the top of our gantry. When positioned beneath a piece, the electromagnet is activated, attracting and securing the piece. It then drags the piece to its destination before deactivating to release it. This method ensures the piece remains in contact with the board, sliding to its new position, thereby avoiding the complications associated with magnetic levitation.

5.2 Arudino Uno Wirless R4

The Arduino Uno Wireless R4 was selected among other potential processing units for several reasons. Firstly, it presents a more cost-effective option, as all current team members already possess one. Additionally, the team has prior experience with Arduino, making programming relatively easier. We opted for the wireless version because it provides wireless communication capabilities, potentially serving as an add-on to our system beyond traditional serial communication. Lastly, it has an existing library that can directly communicate with the A4998 stepper motor driver component so that we can introduce less complexity into the project.

6 SYSTEM IMPLEMENTATION

MaGomuku consists of five main subsystems: Detection and Locking System, Feeding System, X-Y Movement System, Board Processing Unit, and Web Application. The first four systems are integrated into the physical chessboard device for physical interaction with user, while the Web Application subsystem ensures the network connectivity of MaGomuku.

6.1 Detection and Locking System

The detection and locking system is responsible for go piece detection and location locking. In a game of Gomuku, there are two color players - typically represented by black and white pieces. Our system is intended to successfully distinguish game pieces placed by different player and lock game pieces in place to protect the integrity of the game state.

6.1.1 Detection

A matrix consisting of 10 x 10 radiometric hall effect sensors will be employed to precisely determine the location of pieces on the game board. The hall effect sensors will be mounted on a custom printed circuit board. Each individual circuit unit for the hall effect sensing will include the power source, a 0.1uF bypass capacitor, and the hall effect sensor itself. The Hall effect sensor will have a base output 0.6 voltage and will increase proposal to the flux density.

For the actual game piece detection, each black and white piece will contain a different size of magnet and therefore different strength of magnetic field. The hall effect sensor will measure such difference and therefore perform differently in response to different game piece placement. In the case of a sensor output that exceed the threshold for either white or black pieces, to efficiently manage the sensor outputs, a 16:1 multiplexer will be used to determine which sensor is outputting the value. The processing unit receiving this voltage information will then send this information along to the web application.

6.1.2 Locking

In the case of inadequate positioning by the player, we will implement a straightforward locking mechanism. Under the physical chessboard, we will position 10 * 10 Carbon Steel round pieces at each intersection of the grid, which is the desired placement location for each game piece. As the game pieces are magnetic, the Carbon Steel piece will magnetically attract them to the intended location, ensuring proper placement.

6.2 Feeding System

The feeding system plays an important role in the automatic system as it delivers the game piece of the online player to the designated location.

The feeding system utilize a spring-loaded plywood container integrated into the Go board itself. This magazine serves as the repository for the magnetic pieces, which are

stacked vertically inside. The player can also reload the plywood container by stacking the pieces vertically and place them

To facilitate the controlled release of pieces from the magazine, the system utilizes a stepper motor, which drives a flag to physically push the pieces out of the opening of the magazine.

Robustness is the primary consideration in the design of the feeding system. Since electromagnetic forces can attract magnetic pieces within range, the feeding system must deliver game pieces to the designated location with a maximum error of 5mm between each game piece delivery.

6.3 X-Y Movement System

To implement movement, we are using an X-Y gantry system, primarily based on two V-Slot NEMA 17 Linear Actuator Bundle. This system consists of beams, linear rails, wheels, and a stepper motor. Controlled via a CNC shield and A4998 stepper motor driver, the movement system enables precise control over the electromagnet component within a 500mm x 500mm range on the board. This range extends beyond the designated board size of 450mm x 450mm to accommodate the feeding system's operation.

To ensure that the game piece follows the movement of our gantry system accurately, we will utilize an electromagnet component. The electromagnet component mainly consists of magnetic coils that can be activated by a 5V DC voltage input. The electromagnet can exert a powerful magnetic force capable of securely holding the game pieces in place during movement. The magnetic attraction ensures that the game piece will move smoothly along the desired path throughout the movement process. To minimize the risk of displacement or interference with existing pieces on the board, the movement system will take advantage of the space between each grid cell, as game pieces are placed on the interaction of the grid. By doing so, the pathing algorithm will be straightforward as it does not need to account for the existing game state.

6.4 Board Processing Unit

The board processing unit is responsible in controlling various subsystems within the chessboard device, enabling the recording of user inputs and facilitating interaction between local and remote users. The primary task of the processing unit is motor control and location sensing. To effectively control the XY-gantry and gather relevant data, a board processing unit is essential. The Arduino UNO R4 WiFi is chosen for this purpose due to the fact that Arduino already features an existing library that allows direct communication with the A4998 stepper motor driver, enabling precise control of the XY gantry system. The adruio will also recieve voltage input from the hall effect sensors for gane piece sensning. Moreover, it provides wireless communication beyond serial communication. For our final product, we hope that we can achieve wireless communication and therefore the physical chess board will no longer be required to connected to a PC.

6.5 Web Application

6.5.1 Django (Python) Framework

The Django backend is responsible for handling game state control. This includes starting/ending game sessions, updating game state, validating player moves, determining game outcomes, and provide reponese accordingly. Game state control will ensure the integrity and consistency of game data.

Django will also kickstart a separate process for communication with the Arduino board with the PySerial package. This process will enable communication between the backend architecture and physical hardware, allowing for real-time interactions with the game board.

6.5.2 HTML/CSS

HTML will be used to structure the content of web pages, while CSS will be used for styling and layout. The HTML/CSS will present the game interface to users, including the game board, player information, and any other interactive contents.

JavaScript may be used for client-side interactivity and simple user input logic. It will handle user input from the non-physical player, and update the user interface accordingly. It will not be reponsible, however, for any game state handling.

7 TEST & VALIDATION

In order to guarantee the resilience of our entire system, we have chosen to establish and verify unit testing metrics for each individual subsystem, as discussed below. Our emphasis will then shift towards ensuring user experience post-integration.

7.1 Tests for Movement Subsystem

Accuracy

To evaluate the precision of the movement system, we measure the average distance between the position of a fed Go piece and its target position on the game board. The objective is to ensure that the actual final position is within 5 millimeters of the target position for consistent activation of the locking subsystem.

Speed

To evaluate the efficiency of movement system, we measure the time delay between the reception of a signal and the placement of a Go piece in its designated position, considering the longest possible path. The objective is to ensure that the movement time less than or equal to 10 seconds so that each overall turn time is within an acceptable range.

7.2 Tests for Detection Subsystem

Hardware Accuracy

Distinguishing Go Piece:

To evaluate the accuracy of the detection system in distinguishing between black and white Go pieces, we compare the detection HE results for both white and black pieces through iterative testing and ensure that given a threshold of distinction, the system achieves a detection accuracy of at least 96%.

Board State:

To ensure that the hardware system can accurately detect the state of the physical game board, we compare the output of the detection system with the actual state of the board in real time by introducing and removing new go pieces of random color, with the target minimum detection accuracy being 90%.

Software Accuracy

To evaluate the software's ability to detect and prevent various types of illegal moves on the game board, we perform different kinds of prohibited moves and verify that the software accurately detects them. The test aims to achieve a minimum detection accuracy of 96%.

Latency

To evaluate the software's responsiveness in detecting and raising warnings for illegal moves on the game board, we perform various illegal moves and measure the time delay until the board raises a warning. The test aims to ensure that the latency between the occurrence of an illegal move and the warning notification is within the threshold of 50 milliseconds.

7.3 Tests for Locking Subsystem

Range

We evaluate the locking mechanism's effectiveness by determining the distance within which a Go piece can be securely locked without the need for external force. The objective is to ensure that the locking range is less than or equal to 25 millimeters to achieve minimal interference with the movement system.

7.4 Tests for Feeding Subsystem

Accuracy

We evaluate the precision of the feeding mechanism by measuring the average distance between the position of a fed Go piece and its target position on the game board. The objective is to ensure that the actual fed position is within 5 millimeters of the target position for smooth pick up by the movement subsystem.

Speed

We evaluate the efficiency of the feeding mechanism by measuring the time delay between the signal reception and the placement of a Go piece in its designated position. The objective is to ensure that the feeding speed meets the specified requirement of being less than or equal to 2 seconds.

7.5 Tests for User Experience

Timing

To ensure the overall turn time is within an acceptable range, we evaluate the overall efficiency of the automated turn process by measuring the time delay between when a move is played in the software and when the corresponding game piece is actually placed on the board. We aim to limit the overall time taken to be within 13 seconds, which is the sum of the feeding subsystem time and the movement subsystem time.

User Feedback

We will find actual players to play Gomoku using our game board and asks them to rate our product mainly on the aspect of: Ease of use, Timing (is it taking too long), and the overall impression, while taking notes of possible bugs and feedback during user testing.

8 PROJECT MANAGEMENT

8.1 Schedule

We mainly structured our schedule according to the modular subsystems, tailored to accommodate the strengths and weaknesses of each team member. In the early stages, we decided to conduct research and perform feasibility testing on each member's respective subsystem individually, but as we progress along the pipeline, integrating the whole system becomes more critical. We have decided to focus on feasibility and unit testing for individual hardware subsystems (feeding, detection, movement) before Spring Break, and switch to hardware integration and software development after Spring Break. Additionally, we have allocated two weeks of slack time as mitigation in case of unforeseen changes or issues.

The schedule is shown in Fig. 2.

8.2 Team Member Responsibilities

Each member in our team is responsible for a main modular subsystem (Feeding, Detection, Movement), some secondary tasks and the final integration. While some responsibilities may intersect, this approach enables us to leverage the strengths of each member to the fullest.

Sizhe Chen

- Go Piece Detection & Locking System
- PCB Design and Fabrication
- HE Sensor and Mux Integration

Shuailin Pan

- Go Piece Feeding System
- 3D Modeling & Printing & Laser Cutting
- Electromagnet Testing

Zipiao Wan

- Go Piece Movement System
- X-Y Stepper Motor Gantry System
- Web Application Development

8.3 Bill of Materials and Budget

The largest portion of our budget is going towards the X-Y gantry system. In order to save manufacturing cost, we are planning to utilize TechSpark's 3D printers to make custom components and use laser cutters to make custom housing for the entire system. To further drive down the budget, we are using personal Arduino and also borrowing certain singular component from Ideate for testing. To see the full BOM you can refer to Table 1.

8.4 Risk Mitigation Plans

PCB Design & Manufacturing

Moving forward, our primary concern lies in the design and production of the custom PCB necessary for the HE matrix. Within our group, only Chen possesses familiarity with PCB design, and none of us have prior experience in PCB manufacturing. We anticipate collaborating to develop the circuit design and exploring viable manufacturing solutions together.

MUX Communication & Latency

Our secondary challenge as we progress is determining the optimal method and timing for utilizing the 16:1 MUX to accurately receive data from the HE sensor matrix while minimizing latency. To address this, we propose implementing a comprehensive testing regimen where we simulate various data input scenarios and assess the responsiveness of the system under different conditions.

RESOLVED: Movement Method

One of the risks we have successfully addressed involves determining the method for physically guiding the Go piece along the board. Initially, we contemplated utilizing a magnetic levitation device due to its impressive appeal. However, after two weeks of testing, we concluded that while maglev technology is robust during transportation, loading the Go piece onto the existing maglev device requires an impractical level of precision and minimal disturbance. This level of precision is unattainable using pure robotics and is highly error-prone even with human intervention. Consequently, we have opted to progress with our project and transition to utilizing electromagnets for movement.

9 RELATED WORK

Our project is similar to an existing product called Phantom Chess[2]. Phantom Chess is an automated, networked chess board that allows users to play online chess on a physical board. The following comparison is based on solutions recorded in Phantom Chess's development logs[1].

Similarities and Differences

Use Case

Similar to MaGomoku, both project utilizes physical boards to ensure a tangible gaming experience and has network connectivity so that each user can not only play against an AI but also against other players online. Both products want to achieve the same experience, only that Phantom Chess implements the game chess and Magomoku tackles the game of Gomoku.

Feeding

Due to the difference between the game of Chess and the game of Gomoku, our project requires the development of a feeding system while Phantom chess does not.

For Gomoku, since the Go board is empty at the beginning, Magomoku requires a system to feed in pieces to put on the board.

For Phantom Chess, since all the pieces are already on the chess board from the start of the game, it has no need for a feeding system but instead need to worry about removing captured pieces from the game board during gameplay.

Detection

Different from Chess, since Gomoku has only two different kinds of pieces (black & white) compared to six in chess. We only need one HE sensor in each position to accurately distinguish different pieces compared to Phantom Chess's solution of using 9 HE sensors in each grid.

However, since the board space of Gomoku (15 by 15 official, 10 by 10 in our implementation) is much larger than that of Chess (8 by 8). Magomoku has a higher requirement for parsing the location data instead of distinguishing different pieces.

Locking

For Phantom Chess, since each piece may be moved again after each turn, it has no implementation of a locking system to limit each piece to its current position.

Instead, to ensure each Chess piece can stably stay in place after each turn, it adds a metal counter-weight to the bottom of each Chess piece. On the contrary, since the go piece after each turn cannot be further moved, and each go piece is minute compared to a Chess piece, a locking system is necessary for Magomoku.

Movement

Both Phantom Chess and Magomoku uses an X-Y gantry as the movement solution.

Data Processing and Networking

Phantom Chess processes the game data on an ESP32 microcontroller and communicate over the network via an Arduino Nano 33 IoT. For Magomoku, we uses and Arduino Uno R4 to collect all the data and process the game data on a separate processing device (mobile phone, PC).

10 SUMMARY

Magomoku is a networked Gomoku game system with automation features, allowing users to play against remote opponents using a physical board. This system achieves its functionality by employing a magazine to dispense Go pieces onto the board, an X-Y stepper motor gantry to position the game pieces accurately, a matrix of Hall effect sensors for board state detection, and an integrated matrix of steel discs to secure the pieces in position.

Foreseeable challenges in the remaining time of the project would be PCB design and manufacturing, testing for detection and movement accuracy, and modular hardware subsystems integration. We have full confidence in our ability to surmount these challenges and fulfill our use-case and design requirements, ensuring the delivery of a quality final product.

Glossary of Acronyms

- API - Application Programming Interface
- BOM - Bill of Materials
- IoT - Internet of Things
- HE - Hall Effect
- MUX - Multiplexer

References

[1] HACKADAY. *Automatic Chessboard Logs*. Last accessed 1 March 2024. URL: `https://hackaday.io/project/179268/logs`.

[2] KickStarter. *PHANTOM. The Robotic Chessboard Made of Real Wood*. Last accessed 1 March 2024. URL: `https://www.kickstarter.com/projects/wondersubstance/phantom-the-most-advanced-chess-board-in-the-world`.

Table 1: Bill of materials

Description	Model #	Manufacturer	Quantity	Cost @	Total
V-Slot NEMA 17 Linear Actuator Bundle	500mm	OpenBuilds	2	$153.99	$307.98
V-Slot 20x40 Linear Rail	500mm	OpenBuilds	1	$7.99	$7.99
V-Slot Gantry Plate - Universal	-	OpenBuilds	1	$11.99	$11.99
Spacer Block	-	OpenBuilds	2	$11.99	$11.99
Xtreme Solid V Wheel Kit	-	OpenBuilds	4	$6.99	$27.96
Low Profile Screws M5 (10 Pack)	15mm	OpenBuilds	1	$1.19	$1.19
Low Profile Screws M5 (10 Pack)	40mm	OpenBuilds	1	$1.69	$1.69
NEMA 17 Stepper Motor	-	OpenBuilds	1	$17.98	$17.98
Stepper Motor Driver Module	A4988	Ideate	3	$0	$0
Stainless Steel Compression Spring	1 * 10 * 305 mm	METALLIXITY	1	$6.49	$6.49
Carbon Steel Round Flat Washers (100 Pcs)	M5	Still Awake	1	$7.99	$7.99
Round Neodymium Magnet Disc (50 Pcs)	10 * 2 mm	BC MAGNETS	1	$5.49	$5.49
Round Neodymium Magnet Disc (50 Pcs)	10 * 3 mm	VSKIZ	1	$5.79	$5.79
Electromagnetic Coil	-	Ideate	1	$0	$0
Hall Effect Sensor (20 Pcs)	A3144	EPLZON	1	$6.99	$6.99
Capacitors	-	-	-	$0	$0
Resistors	-	-	-	$0	$0
16-Channel Analog Digital MUX (12 Pcs)	CD74HC4067	DORHEA	1	$11.99	$11.99
Arduino Uno R4 WiFi	ABX00087	Arduino	1	$0	$0
					$433.51

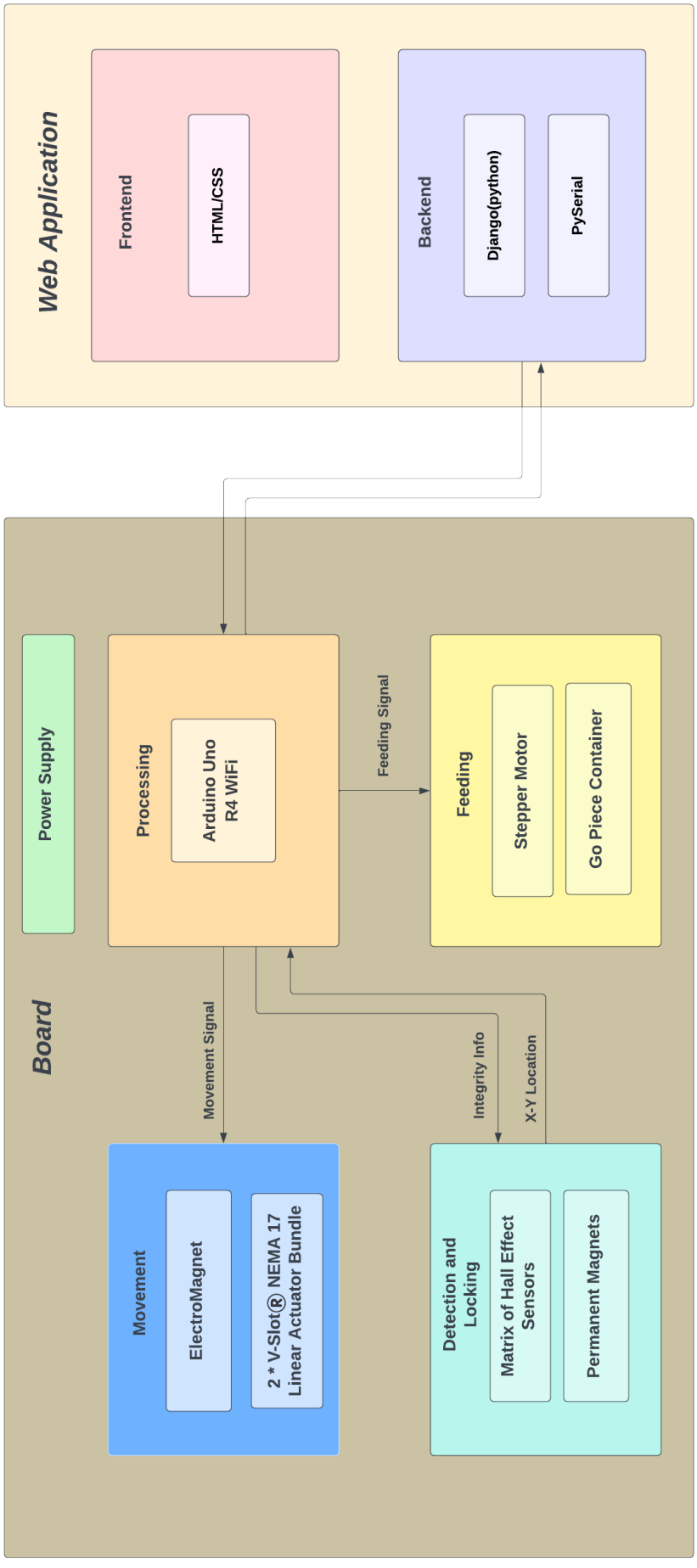

Figure 1: A full-page version of the same system block diagram as depicted earlier.

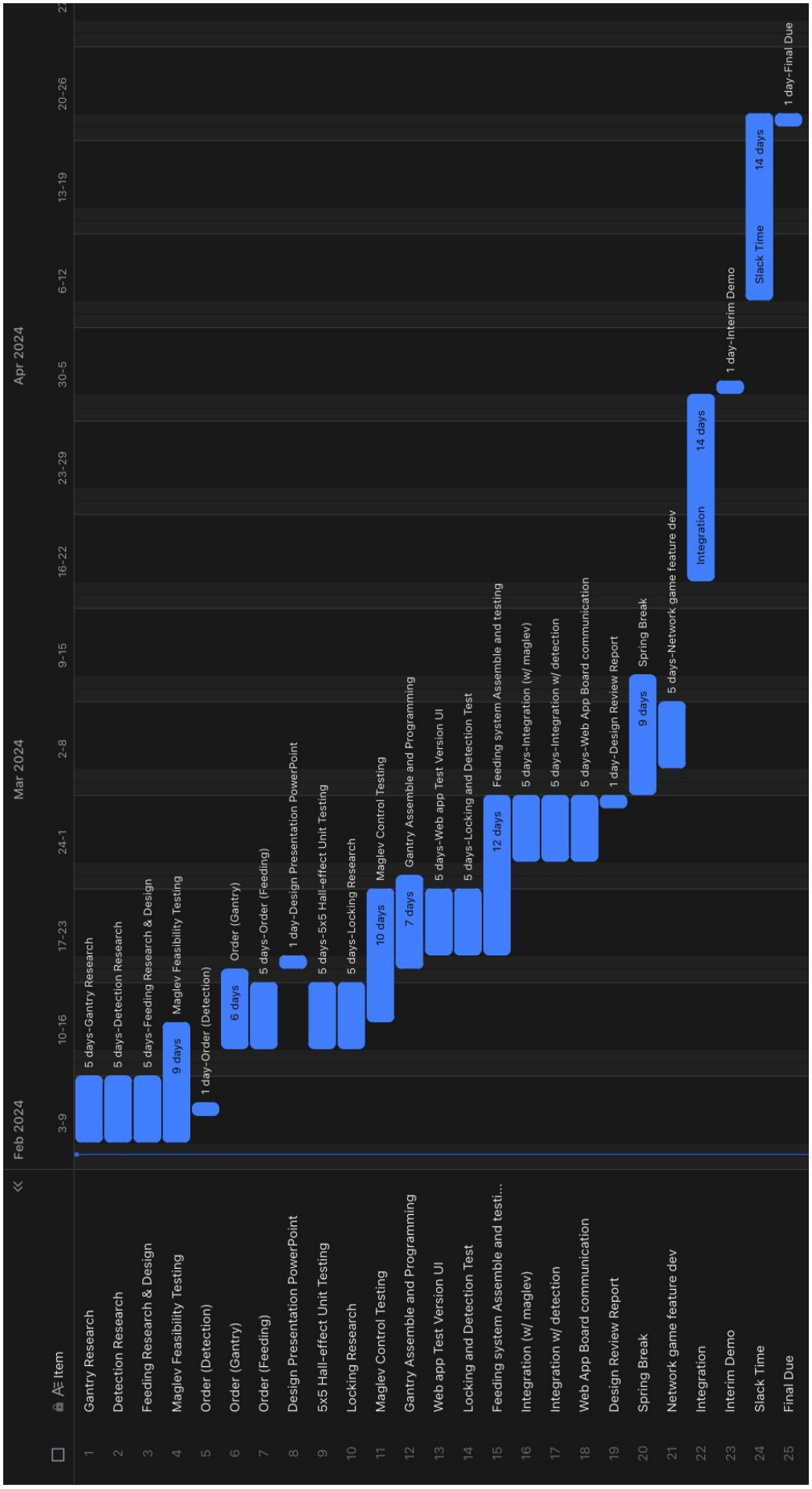

Figure 2: Gantt Chart