

CMPE 325 – Assignment 5

WIZARD CHESS

Tom Heyssel (20000838)
Allison Christensen (10211533)
Ryan Kinsella (10194574)
Matthieu Roux (20013052)
Dennis Grajo (20017666)

Table of Contents

Idea	2
Literature Review	2
Implementation	4
Board Construction	4
Step by step prototype construction	5
Construction considerations and decisions	5
Materials Used and Assembly.....	5
Construction Process and Issues.....	6
Game Logic.....	6
Movement Control Logic	7
Audio Input	8
Testing.....	9
Changes Since Project Demo	11
Prototype to Showcase	11
HCI Relation.....	11
Learnability.....	11
Ease of Use	12
Flexibility	12
Satisfaction.....	13
Timeline.....	13
Future Work.....	13
Challenges	14
Ryan Kinsella:	14
Dennis Grajo:.....	14
Allison Christensen:.....	14
Tom Heysel:.....	14
Mattieu Roux:	15
Appendix A: Contributions.....	16
Appendix B: Video Permission	17
Appendix C:	18
Works Cited.....	18

Idea

This project was inspired by Harry Potter and the Philosophers Stone. At the end of the book, the three heroes must save the day by completing a series of dangerous tasks, one of which is winning a violent game of wizard chess. In the Harry Potter series, wizard chess is played exactly like the traditional game, with the exception that, when instructed, the pieces come alive and move on their own. The goal of the group's project is to build a voice-controlled chess board to allow fans of the series, avid chess players and those with limited upper-body mobility to play the game hands-free and feel as if they are in the magical world of Harry Potter.

Literature Review

The goal of our project was to design a chess board that is more accessible to users with limited mobility, using unconventional, speech inputs. The more general topics concerning the project include how the study of human computer interaction (HCI) can improve accessibility with a focus on its application in board games. The purpose of this literary analysis is to review some of the notable contributions made by experts in the field who are working towards the development of technology that meets the needs and enhances the experiences of a wider user base. When it comes to improving accessibility through the study of HCI, it is not only about what we can do to advance technology, but also about how technology can be used to advance ourselves as a society.

An important question to address in relation to the presented topic is, how do designers enhance board games with technology while preserving the core mechanics of the game? In a publication from the Interactive Institute Game Studio, the authors introduce a technique of performing task analyses to gather functional requirements [1]. Several board games were used for the study, selected based on their relatively complex set of game mechanics. What is novel about their approach is that, instead of having the purpose be understanding how a general activity should be supported, the purpose of their study was to play the game to analyze what the specific tasks and the options available to the players were. This consideration allows gameplay design to be the initial consideration while supporting the optimal use of the chosen technology.

Applying a similar analysis to a game of chess, we can identify our task distribution which may include, taking an object from a position, placing an object on a position, or skill-based action. From there, to improve our project we would consider augmenting each specific task before we consider the entire system. This allows the design team to assess whether specific tasks involved in chess profit from enhancement with electronics. In summary, the approach serves to identify areas where we can streamline the technological growth.

To unite the topics of electronic enhancement and accessibility, a conference paper from the Foundation for Research and Technology and the University of Crete's Department of Computer Science introduces the notions of computer and game accessibility, along with extensive background information and related approaches [2]. It then goes on to discuss software implementation and how accessibility is supported in universally accessible chess for different user categories through the game's interface, its adaptation capabilities, and the available alternative input and output modalities.

This research very closely relates to our project in that the authors set out to achieve the same goal; to create a more accessible chess game. It differs in that we tried to preserve a physical board game while

still improving accessibility through technology. This publication instead presents a multi-modal Web-based chess game, which can be played between two players, including people with disabilities (low-vision, blind, and hand-motor impaired), either locally on the same computer or remotely, over the internet. Where the authors succeeded in achieving near universal-accessibility, our project was only designed to assist those who are hand-motor impaired- and even then, our initial iteration needs to implement a way of resetting the board for the prototype to be truly hands-free.

What we can take away from this publication is a functional example of how we may change our design to also include low-vision, blind, and other users to make the product more universally accessible. These alterations to the team's current design, however, will be no easy feat. This raises the question: if the software platform offers more accessibility, is it worth it to preserve the physical game board?

An alternative solution could be a board game designed with a tabletop display interface. In a report from the Dalhousie University Faculty of Computer Science, author Tara Whalen proposes that board game designs can be adapted for tabletop use, allowing the creation of interfaces that are usable by multiple users working on concurrent tasks [3]. The aspects of tabletop displays to which board game designs can be applied include, recognizability of display for interaction; demarcation of individual and shared space; and the creation of public and private display areas.

While all these elements may not be applicable for a relatively simple game like chess, the tabletop interface presents a functional mid-way point between a physical board game and a web-based application. Its implementation offers more potential for creating an immersive gameplay experience while eliminating the challenges of hardware design.

One argument in favor of the physical board design states that, although immersive graphics, mundane task automation, saving and restoring the state of the game have been greatly improved by technological enhancements to games, the transition to a digital environment leads to a loss of the flexibility that makes traditional board games inherently popular.

This argument is supported by the research published in the "2010 Third International Conference on Advances in Computer-Human Interactions" [4]. The authors claim is that it is difficult for the user to achieve same level of customization in the digital domain. A conceptual model is introduced for the design of digital board games that the authors call the "flexible rules framework" which enables and facilitates flexible and extendible game design and development. While we are not designing a digital game, the authors survey results validate the previously stated claim and supports the team's argument for choosing to preserve the physical board, even if this approach presents challenges of its own.

Expanding our design, a step further towards achieving universal accessibility, the team must also consider how to make the existing design accessible to users with hearing loss. A paper on eliciting accessibility requirements for people with hearing loss, from the International Conference on Human-Computer Interaction, identifies the perceptual, cognitive, and social barriers that people with singular impairments face [5]. The idea of the authors' investigation was to elicit a set of requirements and recommendations for the design of web content and application, but also of assistive technology in general. Based on the conclusions of this investigation, our project should be modified to allow a player to enter key board inputs to move their game pieces. A user could not simply move a piece by hand unless the internal game logic had a way of tracking this movement. There must also be a visual element, like a monitor, to provide text prompts as an alternative to audio cues.

The topic of accessible board games in HCI is an abundant field of study. A key element in designing a technologically augmented game without diminishing its core mechanics, lies in the task-analysis and enhancement optimization. While the advantages of a software-based implementation of accessible chess is discussed, the team will be moving forward with the preservation of the physical board. A tabletop display will be considered as a viable alternative in future work and other HCI game-based projects. Before the prototype can be considered universally acceptable, the barriers faced by individuals of varying ability must be considered and the design must be modified accordingly.

Implementation

Board Construction



Figure 1- Stepper motors with axis directions labelled



Figure 2

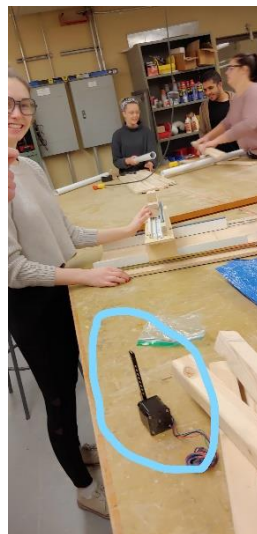


Figure 3



Figure 4



Figure 5

Step by step prototype construction

1. Mounted the two X-axis drawer sliders parallel to each other, onto the base.
2. Mounted the pieces of wood where the Lego track will lie, onto the base.
3. Created the mounts for the stepper motors (construction in Figure 4, visually seen well in Figure 1), mounted onto the X-axis.
4. Created Y-axis: Mounted drawer sliders to Y-axis, mounted wood where the Lego track will lie to Y-axis, mounted Y-axis on top of the X-axis stepper motor mount so the Y-axis rests on top of the X-axis (seen in Figure 2 or Figure 3), mounted Y-axis stepper motor holder onto the Y-axis drawer slider (Figure 1).
5. Hot glued on Lego tracks to X and Y-axes.
6. Hot glued stepper motors onto their respective mounts on X and Y-axes (the Z-axis stepper motor rests upon the top of the mount on the Y-axis).
7. Cut whole out of the centre of the thickest top board (Figure 5).
8. Screwed in the four tabletop legs to the top and bottom base.
9. Cut whole in thin Z-axis arm, pushed into stepper motor armature (Figure 5).
10. Glued game board on top of the top base board.

Construction considerations and decisions

Before beginning construction, the team understood that the prototype will be designed to be able to move (using stepper motors) a magnet to any X/Y co-ordinate of the game board. For future reference in this document, Figure 1 above shows what is being referred to by the X, Y, and Z-axis. As seen in Figure 1 above, Z-axis rotates the magnet 90 degrees to grab/release the chess pieces. Each axis will be mounted on the previous axis, with the lower most axis being the X, then Y, then Z. This will enable the Z axis (which contains the magnet) to be freely carried to any position that the X and Y positions declare through the game logic. The chess pieces would need to be magnetic, and the magnetic attraction between arm and piece needs to be strong enough to move a singular chess piece, while not so strong as to affect other pieces on the board.

The team agreed before construction that using a permanent magnet would be favourable over an electromagnet, due to the cost, team inexperience and complications that may arise using electromagnets and integrating it with game logic. The user experience would be the most magical if after they give a move command, the user won't see the innerworkings of what is going on underneath the board, which the team can cover up using a curtain.

Materials Used and Assembly

The design of the prototype heavily revolved around using recycled materials due to budget constraints. The team used assorted pieces of recycled wood, Lego, and dresser drawer sliders, best seen in Figure 2 or Figure 3. The drawer sliders were sprayed with WD40 to reduce friction and give a smoother glide feel.

The team ordered a set of compatible Lego gears and tracks online, and with a rough eye estimation on how big the gears would be they came in and were the same size as the stepper motor armature, and were able to be pushed firmly in. This made it easy for the team to guarantee the gears to stay on without error. The tracks were placed on a thin Lego board that was glued to another set of wood, designed for the exact height where the stepper motors will rest. The stepper motors were given square resting spots to sit in, where they were secured in place using hot glue.

Construction Process and Issues

The first design feat the team needed to figure out how to move the magnet on a set of axes, which took a great deal of brainstorming and research before attempting an implementation. Figure 2 shown above was how the team was able to make the Y-axis completely independent of the X-axis. This was necessary so that the Z-axis could be mounted onto the Y-axis.

The sliding drawers and their tracks were mounted sideways. This is because the drawer tracks are angled to slide slightly downwards towards the base (bottom most part) of the prototype. The sideways wheels made friction a larger problem than the team anticipated but was overcome by applying WD40 to reduce lubricate the parts in motion. The horizontal facing wheel of the Y-axis is clearly seen in Figure 3.

The permanent magnet initially was far too powerful and often attracted multiple pieces in close proximity. Instead of adjusting the height of the Z-axis arm, the team broke the magnet into a smaller size to reduce its strength.

Since the wood we used was recycled and was slightly warped in certain areas, this cause the track to be higher in certain areas than others. How the team dealt with this was by taking off sections of the track and using extra hot glue to these sections as this would add a few millimetres which was needed to make the track level.

Figure 3 above shows a Lego piece we intended on using on the stepper motor as the magnet, however it wasn't until once the board was almost fully constructed the team realized that the magnet attached will be larger than the width of the Lego piece. Instead the team used a section of a thin wooden pole which has a diameter of the magnet, that was drilled using a thin drill bit the same diameter as the armature of the stepper motor to attach, as seen in Figure 5. Despite all these challenges, a prototype was constructed and fully functioning.

Game Logic

Structure

The game logic was constructed in a single python file called brain.py. The program declares two classes, Board and Piece, then starts the infinite game loop.

The Piece class has two attributes: type and color. An example of a piece's attributes would be (queen, white).

```
class Piece:
    def __init__(self, type, colour):
        self.type = type
        self.colour = colour
```

The Board class' constructor instantiates the 12 Piece objects required in a chess set and populates a 2-D list with them such that it resembles a top-down view of a chess board.

Game Loop

Inside the game loop, the program instantiates a Board object then enters a nested loop broken only by a checkmate. Inside the inner loop, the system prompts the user to state a move to and waits for a

response. Upon receiving user input, the system translates the audio to a string (see Audio Input), then from a string to a set of co-ordinate pairs called a command. For example, "Delta two take delta 5" would become {(3, 2), (3, 5)}. The system repeats the interpreted phrase to the command and asks the user to confirm or try again. When a command is confirmed it is passed via serial communication to and an Arduino program, the board toggles its turn variable, and the inner loop ends. When a king is taken, and the inner loop is broken, the players are asked if they want to play again or exit.

Enforcing Chess Rules

Before the command can be sent to the Arduino, it first must go through five processing checks to determine if the move is legal. The check system is set up such that if the command passes the first test, it moves on to the next, if at any point a condition fails it will immediately exit and ask user to repeat.

The tests are:

1. **Occupancy:** Is there a piece at the starting co-ordinate?
2. **Turn:** Is the piece at the starting position of the correct colour based on whose turn it is?
3. **Direction:** Is the path and length of move legal for the type of piece being moved?
4. **Clear Path:** Are there any pieces in between the start and end point?
5. **Capture:** Is there a piece at the endpoint and is it an opponent (*trigger a capture), or, is end of the path empty?

A capture is the intermediate move that is automatically called to remove a piece from the board when it is being taken. This occurs before the attacking piece arrives at its destination so that its entire path is clear of any obstacles.

In the first draft of the program, each type of piece had its own function determining if the Direction and Clear Path conditions are met. These functions all consisted of a series of loops and conditions that were highly repetitive between pieces. The logic has since been optimized and condensed, reducing the brain.py line count by approximately 200 lines of code, or 25%.

One example of an area of optimization is in the functions that check for a valid queen or king move. Inside the check_queen() function, it will in turn call the functions check_bishop() and check_rook(), if it is legal for one of them, it is also legal for the queen.

The king is determined to be a valid move if the check_queen() function returns true and the command's start position is one tile away from its end.

Movement Control Logic

Preparing Command for Transfer

Once a command has been determined to be valid, it must be translated into the exact sequence of motor controls to successfully execute the movement. This process must be completed twice if the move is a capture - the first translation being to move the defending piece to the graveyard and the second for the attacking piece to take its place.

The translation results in a packet consisting of six chars to be transferred serially. The packets contents are:

1. X Distance (0-7 tile positions)
2. X Direction (0 clockwise/1 counter clockwise)

3. Y Distance (0-7 tile positions)
4. Y Direction (0 clockwise/1 counter clockwise)
5. Magnet State (0 off/1 on)
6. Packet Info (single char encoded to dictate control in Arduino program, more in Arduino Control, below)

Moving one piece requires sending four packets:

1. Move arm to start position
2. Turn magnet on
3. Move arm to end position
4. Turn magnet off

To build the first packet, `brain.py` subtracts the current position of the magnetic arm's (X, Y) from the start position of the move's (X, Y). The subtraction determines the X and Y distance from the magnitude of the difference and the direction from the sign of the difference. It does the same for the third packet.

The second and fourth packet order a change in state of the magnet. Based on magnetic arm's current orientation, `brain.py` must determine a safe way to rotate the arm. If either the start or the end X position is $X = 0$ or $X = 9$, the arm's rotation is at risk of catching the edge of the board due to a slight lip in construction. This risk is mitigated by ensure the arms rotation is directed inward. If the arm's X position remains in the middle of the board, any direction is considered safe.

After all 4 packets are formed they are written to the serial output and are considered completed. The transfer assumes no error and is therefore unidirectional. There is no acknowledge sent from the Arduino.

Arduino Control

The Arduino program `loop()` function continuously polls the serial buffer, doing nothing if it is empty and breaking the polling loop if a character is detected. Once there is data available, the system fetches the packet and facilitates motion. The precise controls required (i.e. X/Y order, diagonal motion, knight move specifications, capture specifications etc.) are encoded in the last char of the packet and ensures the movement does not interfere with any of the other board piece's current states. Other than the Arduino `setup()` function that initializes the pinModes of the three stepper motors, and the function to step the motors the program has no other components. It was designed to be as lightweight as possible with the majority of computation kept on the raspberry pi.

Audio Input

The team originally planned only to use the Google speech-to-text API, in conjunction with the `SpeechRecognition` python library to facilitate audio input. An unforeseen complication with this configuration was that, by design, Google's API continuously sends all microphone activity to be translated. This not only disrupted the flow of the game, with users talking in between moves being confused for commands, it also quickly expired the amount of translation time that came with our free trial of Google's software.

Re-evaluating this aspect design, the team pivoted to incorporate a python hotkey library called `Snowboy` in the system. This library specializes in offline keyword recognition, allowing short 5 second audio clips to Google only after `Snowboy` recognizes the custom wake phrase "Wizard Chess". Using

Snowboy allowed the team to continue with our free subscription to Google's API as well as allowed users to converse freely during their game.

Inherent Speech Input Difficulties

COMPLICATION	SOLUTION
INPUT CORRECTION	Before executing a move, system repeats the command and asks for user confirmation.
LEXICON	Clear explanation of command syntax at start of game (start column, start row, "take", end column, end row). Analogies to known game Battleship.
DEIXIS	Originally designed to take input similar to Harry Potter books ie "knight to E4". This designed was changed so that the user must explicitly state start and end position. Removes possibility of ambiguity in command when there are two pieces of the same name that can move to the end position.

Testing

Testing was divided into three sections: software, hardware, and integration. All three sub-systems were further broken down into the smallest testable pieces, a technique known as unit testing. In isolated conditions, each unit was tested under specific functional requirements, all errors and bugs encountered were recorded to be fixed in the ongoing iterations of design. This process was repeated until no errors remained.

Software

Software components were broken down into piece units, board configuration units and game configuration units. Piece units were tested to ensure each piece was able to correctly identify itself. Then each piece was tested in an empty board configuration, being given a series of legal and illegal moves. Once a piece's simulated motion was confirmed to be error-free on an empty board, pieces were added to test clear path and capture recognition. Finally, game configurations were tested to ensure turns are enforced, and checkmate was a functional game-over trigger.

Speech recognition was unit tested in an isolated program, ensuring that the microphone was properly configured, and that speech could be accurately translated.

Hardware

Hardware testing began with adjusting the stepper motor driver potentiometers to yield desired current flow for smooth motion. Once established, motors underwent testing to determine the correct number of steps per tile were required. Then, individual pieces were tested to ensure magnet strength and arm speed were appropriate to handle movement. Finally, the board underwent perimeter and line testing. Where a single piece was moved around the entire perimeter of the board, across the length of every row and column sequentially, and through the diagonal X of the board. Although, not realistic for gameplay, these tests were deemed necessary to ensure the motors had full range of motion.

Integration

Integration testing started by replacing the command input from keyboard to audio. Individual letters and numbers were tested for recognition, and then full commands. These tests required numerous iterations and resulted in the decision to change the original design from English alphabet to radio alphabet to better distinguish the rhyming letters B, C D, and E. Additionally, these tests indicated the

need to change the command structure from “alpha one to alpha two” to the final version “alpha one *take* alpha two”. The middle word was changed from “to” to “take” so as not to be confused with the number “two”

Once audio input was firmly established in the program, the hardware/software communication element was added. Preliminary testing consisted of creating a sender and receiver program to ensure serial communication was successful. Once the connection was established it was easy to implement into the game logic. Complete testing of the integrated system consisted of playing full games with group mates and peers while bug tracking, fixing and iterating.

Evaluation

Due to integration issues, the team was unable to implement audio and visual output into the system. Initial designs incorporated backlighting to inform the user when the board was listening and processing commands. This feature that was not implemented as the prototype had already exceeded its budget and the feature was deemed to be nice but not necessary. Additionally, the board is supposed to inform the player using audio output if they had made an invalid move or input, as well as which player's turn it is, and whether it was able to register the user's commands. This was implemented via command line text output, however, when attempting to shift to audio, the Pi's operating system's audio configuration files were unintentionally affected in such a way that it no longer recognized the microphone as an input device, essentially breaking the systems audio input. After significant attempts to fix the issues, the team had to back up their files and reconfigure the Raspbian OS on the Pi in order to fix the input. Afraid of breaking the system again during filming of the promotional video, audio output was left behind.

This led to decreased satisfaction during the user study. Text output on a monitor, while functional, was not as seamless and a frequent comment from users was that it was not as fun to read the system response on the command line. If the team had the chance to implement things differently, we would have rearranged the project timeline to focus on output earlier giving more time to fix the problems when they arose.

During the study, it was observed that the users enjoyed using the activation phrase “Wizard Chess” as it reminded them of smart home assistants that they were familiar with that also used an activation phrase. Despite a few minor issues, overall, the users enjoyed using and playing with the chess board. They expressed a sense of wonder and curiosity upon watching the chess pieces move automatically after giving it a command. The users were also very intrigued at the inner mechanics of the board and wanted to know how it worked. The board was able to inspire a sense of awe and give the illusion of magic to the users.

Changes Since Project Demo

Prototype to Showcase

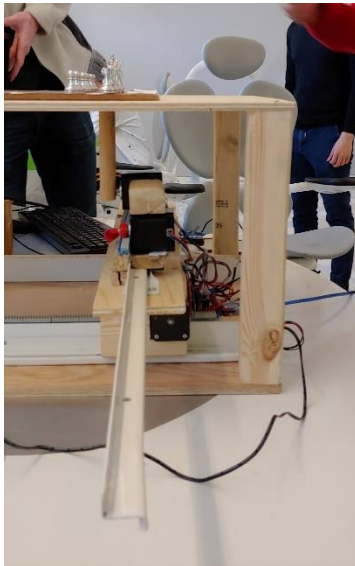


Figure 6- Initial Prototype



Figure 7- Showcase design

The prototype as seen above in Figure 6 was functional but lacked integration the speech and game logic. The aesthetic appearance had not been considered up to this point and the team intended to improve this aspect for the showcase.

Since the construction of the prototype, improvements have been made to the track's functionality by raising its height to reduce the number of gear slips. Speech input was implemented successfully and aesthetics improved.

Inspired by Harry Potter and the Philosophers Stone, and Figure 7 above shows the final version of the chess game as the team intended it mimicking the room in which the protagonists find themselves at the end of the book. This includes the door our heroes have walked through to get to the chess game, and the door they still must pass through once winning the game, with torches lighting the room. The wood frame has been painted black, the game board is painted white and black, with the numeric and radio alphabet neatly imprinted along its edge to assist the user. Curtains along the side were set up to hide the internal mechanics of the system and give a more magical feel.

HCI Relation

The key principles of system usability, as established by Shackel in 1990, include learnability, ease of use, flexibility, and satisfaction. These principles allowed the team to focus early in the project on the goals we aimed to reach and on the criteria by which we would evaluate our success. As well, having quantifiable measures by which we were able to evaluate whether goals were met, assisted in assessing our progress throughout the semester.

Learnability

To assess the learnability of a device is to assess the time and effort it takes to reach a certain level of task performance. This is achieved through the support of rule-based behavior. The topic of learnability

can be broken down into several elements including synthesizability, predictability, familiarity, generalizability, and consistency.

Familiarity and generalizability were relatively simple to achieve as the rules of our board game remain consistent with the rules of traditional chess. The prototype is designed with the assumption that the user already knows how to play the game, thus offering complete instructions was deemed out of the unnecessary. Interactions, such as using the hot word “Wizard Chess” will be familiar to many users who have seen or used similar such as “Hey Siri” or “Alexa” from more popular devices. Through these features, the user can apply existing rule-based knowledge rather than resorting to unfamiliar knowledge-based problem solving.

The synthesizability is achieved through audio and visual responses to user input. For example, if a user attempts to make an illegal move with a pawn, the prototype will communicate that the move is not valid, as the piece will not be moved. The user can therefore synthesize that when they attempt to move a piece in a way that is against the rules of chess, the board will not accept the command as a valid input. Based on this information the user can then reasonably assume that if they try to move another type of piece illegally, such as a knight, they can expect a similar response. Moreover, with chess in the title of the device, the user should be able to reasonably assume that the device is consistent with the traditional rules of chess.

Ease of Use

Ease of Use, as a metric of success, refers to the degree to which a task can be completed efficiently and effectively (without error). Avoiding and recognizing skill-base and rule-base slips relies on observability, and responsiveness. It was essential for the response of the board to be quick enough for the user to observe the change of state resulting from their command. Responsiveness presented a challenge initially due to the communication between the Pi and the Arduino but was achieved after significant testing and optimizing of the code.

Avoiding rule-base and knowledge-based mistakes relies on recoverability, and task conformance. Task conformance is achieved through the game logic - again, the board will not accept invalid entries. Recoverability could be implemented in to the prototype through audio output, by repeating the input of the user as the computer has interpreted it and asking the user to confirm that this interpretation matches their desired move.

Flexibility

Flexibility refers to the ability of a system to adapt to different users and uses. This can be achieved through implementing dialog initiative, parallelism, task migratability, substitutivity, customizability, and/or measuring flexibility. Applying this principle, the team focused on substitutivity mapping interpreted homonyms such as ‘two’ ‘to’ ‘too’ and ‘2’ or ‘for’, ‘four’ and ‘4’ all to one result.

An initial concern with our design development was how to accommodate English speakers with varying accents in our audio input. We were also concerned that using a letter by number coordinate system, where the user would say ‘B5 to B7’ would present several difficulties. For one thing, the letters B, C, D, and E all sound very similar phonetically, thus the computer struggled to distinguish between different commands involving these letters. This resolved thanks to the use the radio alphabet. Secondly, using the directive word ‘to’ would not be possible as it can too easily be confused with the number 2. Thus, the directive word was changed to be ‘take’ instead.

In instances where using audio input may not be feasible, like in a busy hall with substantial ambient noise, our prototype can be adapted to handle keyboard inputs. This ensures performance and handles error in different situations. This adaptability also offers the user some customizability. Customizability, as a sub-element of the usability principles, refers to the extent to which a system can be adapted to user's needs. The prototype can be used with keyboard input when internet is not available for speech-to-text conversion, or it can be used as a classic manual chess board if both internet and a source of power are unavailable.

Satisfaction

Satisfaction as a principle of system usability measures the degree to which a system is perceived by a user as pleasant and comfortable. This can be accomplished through aesthetics, environmental factors, or by creating a product that offers emotional value, beyond the use of the system.

As the prototype was designed to be functional rather than attractive, much of our materials are recycled or repurposed. Thus, initially the appearance of the device was lacking, but in considering the importance of aesthetics, the following changes were made; paper curtains were fixed to the top board to hide the electrical components below- this was to keep the user's eye focused solely on the movement of the pieces on top of the board. While the sliding drawer tracks still protruded from the frame on some side, the curtains assisted in preserving a sense of illusion and "magic". All other visible areas of the device were painted black, aside from the board which kept a traditional black and white chess board pattern. Along the chess board were painted the radio alphabet and number coordinates to assist the user in stating their move. Additional elements, such as paper torches and doors provided a nod to the original wizard chess scene in the Harry Potter films while cultivating a cohesive aesthetic.

Timeline

One significant and unforeseen bottleneck in development was only having one system to design and develop on. Having to co-ordinate between groupmates who would have the Raspberry Pi when, and carrying the heavy, delicate and awkward prototype in-between teammates' houses, the shop and computer labs significantly slowed down the work flow.

This bottleneck, combined with shipping delays, resulted in the team being consistently about a week and a half behind schedule during the middle of the term. Starting the project early when we had low workload and putting more focus on the design at the end of the term helped the team get back on track and finish within the allotted time.

Future Work

If more time and especially more financial resources were allotted for this project, several changes and additions would be implemented into our current design.

During the Creative Computer Science Showcase on April 4th, the first hour of demonstration went smoothly. However, after about an hour of the system being powered on, the Z-axis' motor began to malfunction, twitching or not rotating at all, when commanded to. By comparing the relative temperatures of motors and stepper driver's heat sinks, it was predicted that the Z-axis driver had overheated and burnt out due to an excess of current flow as a result of being powered on for a long period of time.

This hypothesis was confirmed after the showcase by replacing the stepper driver with a spare one that came with the CNC kit. If this project was continued in future iterations, the team would replace the 24V stripped and soldered camera charging cord with a 15V one specifically designed to be used with CNC Shield.

The main issue with the tracks not functioning at 100% accuracy was their reliance on using the drawer sliders as tracks, and the tracks not being equally level due to wood inconsistencies. Instead of using the drawer sliders, the X and Y-axes stepper motor mounts could use wheels and follow a straight path created by two parallel pieces of wood/metal as a guiderail. The guiderail that contains the section of the track should be ensured that it is completely level before gluing on, and the glue should be a consistent height. For some additional aesthetics, the team would have liked to replace the large drawer slider protruding out of the game board with something that can stay within the boundaries of the four legs of the tabletop, with this protrusion seen much earlier in Figure 6.

Finally, as described in the Evaluation section above, the team would want to implement audio and lighting output queues to enhance the overall user experience.

Challenges

Ryan Kinsella: The biggest challenge of this project for me was constructing the prototype using such limited resources and materials. The team was very constricted because of the budget/ access to a close hardware store that wasn't an hour bus ride away, so we all gave everything that we had on hand and made it work. We made it work using the drawer slides and mounting them sideways, but in testing this caused additional track problems that would not be an issue if we used a sturdy set of tracks that are the same height and were consistent with each other.

Dennis Grajo: The biggest challenge for me when working on this project was constructing the track. As part of the track and board construction team, we were responsible for the construction of the hardware aspect of the project. However, I am more comfortable with the software aspect of things, as it is what I'm specializing in during my Computer Engineering studies at Queen's and I have more experience with it. So, having to work on hardware, a field in which I do not have as much experience in, was very challenging. We also worked on the board in the ILC workshop, working with woodworking tools. I have not worked with woodworking tools in a few years so having to use them again after all this time proved quite challenging.

Allison Christensen: As part of the board design team, I faced similar challenges to Ryan and Dennis during construction. Having been heavily involved with integration testing, the biggest challenges I faced was in fine-tuning the gear movements and reducing friction along the track. Additionally, the stress of building and integrating the prototype created some friction between team members, especially in the days leading up to the demonstration. This presented a challenge in that I had to exercise some mediation skills to ensure that the team remained intact, and that the development timeline continued to progress.

Tom Heyssel: One of the biggest challenges I faced was establishing communication between the Raspberry Pi and Arduino. Initial attempts to use the Nanpy python library was successful for controlling small features on the Arduino, such as LEDs, but when trying to drive the stepper motors, their movement was slow and unpredictable. After several days of testing and debugging, I pivoted to using

plain serial communication and allowed the stepper motors to be controlled in the Arduino code. This implementation was more difficult in design but led to a more elegant control system.

The second biggest challenge I faced was in fixing the board's track. The system is designed so that every movement is relative to its current position, and its current position based on the last movement. Therefore, if the gear slips off the track at any point in time, the entire system is compromised and requires a reset. I spent numerous hours refitting and resetting the motors and the track teeth to ensure that all components were secured in place.

[Mattieu Roux](#): Working with Snowboy wasn't as easy as one would like it to be. First, Snowboy only works on Unix system and the only Unix system the team had was the raspberry pi. This means that the team couldn't work on integration between the pi and the Arduino and speech recognition at the same time. Then, when Snowboy recognized the keyboard, it wouldn't want to leave the microphone access to the SpeechRecognition library. For that, some edits had to be made to the library.

Another issue where homonyms, or numbers being give in letters: When someone would say "Two", the API would sometimes give "2", "two", "too" or "to". To solve the problem, the team created a function that would process the strings given by SpeechRecognition; the function would capitalize all words in the string, then turn "two", "too" or "to" into "2". The chess logic would then be given the processed string.

Appendix A: Contributions

Assignment 5



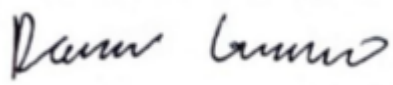

Name	Contributions
Allison Christensen	<i>Written:</i> Literature Review, HCI Relation, Co-edited Report <i>Physical:</i> Board Aesthetics
Tom Heyssel	<i>Written:</i> Game Logic, Movement Control Logic, Testing, Co-edited Report. <i>Physical:</i> Painting/Board Aesthetics, Unit testing - Bug tracking - Bug fixing – Iteration, Refitted motors and track to allow for full range of movement, Configured local area network to establish SSH connection between Pi and laptop so laptop could be used as a monitor during showcase.
Dennis Grajo	<i>Written:</i> Evaluation, Timeline, Challenges
Ryan Kinsella	<i>Written:</i> Board Construction, Future work
Matthieu Roux	<i>Written:</i> Audio Input <i>Physical:</i> Filmed/Edited promotional video

Project Summary

Name	Contributions
<i>Track & Board Design/Construction Team</i>	<ul style="list-style-type: none"> o Discussed/designed physical prototype mechanics o Collected materials (wood, chess pieces, gears/track, drawer sliders, etc.) o Cut and sanded base board, legs, motor support, and top board o Screwed parallel drawer sliders for lower axis to base board o Mounted upper axis drawer slider and support o Fixed circular gears to x and y axis stepper motors and glued motors to the drawer sliders o Glued track attachments at appropriate height to meet the gear teeth (both axes) o Cut and sanded dowel for magnet arm attachment o Secured magnet/dowel to third stepper motor and mounted the unit to the upper axis slider o Cut top board to fit square chess board o Attached legs, top board, and chess board to base o Drew 10X10 chess grid on board for testing

	o Resolved upper axis instability issue
Allison Christensen	o <i>Track & Board team member (see above)</i> o Unit testing o Painting/board aesthetics o Co-edited all reports
Tom Heysel	o Game logic design and implementation o Movement logic design and implementation o Established serial communication between Raspberry Pi and Arduino o Configured Arduino CNC Shield for 32-microstepping mode o Installed stepper drivers and configured potentiometers o Unit testing - Bug tracking - Bug fixing – iteration o Resetting motors and track o Established LAN network to establish SSH connection between Pi and laptop so laptop could be used as a monitor during showcase o Painting/Board aesthetics o Co-edited all reports
Dennis Grajo	o <i>Track & Board team member (see above)</i> o Wrote Assignment 3 report
Ryan Kinsella	o <i>Track & Board team member (see above)</i> o Audio output
Matthieu Roux	o Audio Input o Filmed/Edited promotional video

Appendix B: Video Permission

Name	I hereby grant consent to Queen's University to share my CMPE 325 Project Video	Signature
Allison Christensen	Yes	
Tom Heysel	Yes	
Dennis Grajo	Yes	
Ryan Kinsella	Yes	

Appendix C:

Works Cited

- [1] PEITZ, J., ERIKSSON, D., AND BJÖRK, S. 2005. Enhancing board games with electronics. In *Proceedings of the 2nd International Workshop on Pervasive Games-PerGames*.
- [2] GRAMMENOS, D., SAVIDIS, A., AND STEPHANIDIS, C. 2005. UA-Chess: A universally accessible board game. In *Universal Access in HCI: Exploring New Interaction Environments-Proc. 11th Int. Conf. on Human-Computer Interaction*, 7.
- [3] CHANG, Y.-L. B., SCOTT, S.D., AND HANCOCK, M. 2014. Supporting Situation Awareness in Collaborative Tabletop Systems with Automation. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces (ITS '14)*. ACM, New York, NY, USA, 185-194. DOI: <https://doi.org/10.1145/2669485.2669496>
- [4] FRAPOLLI, F., BROCCO, A., MALATRAS, A., AND HIRSBRUNNER, B. 2010. FLEXIBLE RULES: A Player Oriented Board Game Development Framework. In *2010 Third International Conference on Advances in Computer-Human Interactions*. IEEE. 113-118.
- [5] FERREIRA, M.A.M., BONACIN, R. 2014. Eliciting Accessibility Requirements for People with Hearing Loss: A Semantic and Norm Analysis. In *Kurosu M. (eds) Human-Computer Interaction. Applications and Services. HCI 2014. Lecture Notes in Computer Science*. Springer, Cham. 8512. DOI: https://doi.org/10.1007/978-3-319-07227-2_27