Chessbot: A voice-controlled chess board with self-moving pieces

Cite as: AIP Conference Proceedings **2502**, 040001 (2022); https://doi.org/10.1063/5.0108986
Published Online: 26 October 2022

Nino U. Pilueta, Honeylet D. Grimaldo, Moises F. Jardiniano, et al.

Chessbot: A Voice-Controlled Chess Board with Self-Moving Pieces

Nino U. Pilueta[1,a)], Honeylet D. Grimaldo[2, b)], Moises F. Jardiniano[1, c)]and Manuel Garcia[3, d)]

[1]*College of Engineering, FEU Institute of Technology, Manila, Philippines*
[2]*College of Engineering, FEU Alabang, Muntinlupa, Philippines*
[3]*College of Computer Studies, FEU Institute of Technology, Manila, Philippines*

[a)] *Corresponding author: mfjardiniano@feutech.edu.ph*
[b)] *hdgrimaldo@feualabang.edu.ph*
[c)] *nupilueta@feutech.edu.ph*
[d)]*mbgarcia@feutech.edu.ph*

Abstract. Automated chess is an emerging challenge from the field of robotics to human interaction. Customarily, chess playing robots use video camera for detecting the state of the board and robotic arm to manipulate the pieces, which makes it either expensive or too fragile to move. In this study, an automated chess board called "*Chessbot*" was built by combining a multitude of technologies and techniques such as voice command recognition, *x* and *y* plotting and recognition of chess pieces in a Cartesian plane, artificial intelligence algorithm, and Android mobile application. To properly execute chess moves either from a human player or computer via voice commands or a mobile game interface, the microcontroller provides motors the current and landing squares of chess pieces on the board. For the evaluation, several matches were simulated to perform various testing procedures such as voice command recognition, move log accuracy, display and user input acceptance accuracy, self-arrangement, and chess move reliability. Upon testing, *Chessbot* performs all the necessary tasks efficiently and accurately, which indicates the possibility of using this automated device for chess games at a hobby-level or professional matches.

Keywords: Automated Chess, Auto-arrange, Chessbot, Google Speech, Voice Command

INTRODUCTION

Since the ancient times, board games (e.g., Mehen of Egypt, Ludus duodecim scriptorium of Roman Empire, and Royal Game of Ur of Mesopotamia) have been an integral part of all cultures and a prevalent practice of entertainment.Masukawa [1] explored the invention of board games and discovered that a board with a row of squares or small holes was used by our ancestors to determine oracles. In present times, chess is one of the most popular board games as it has not been confined to any demographic aspects (e.g., age, gender, race, cultural background). This two-player technique tabletop game is a mimic battle fought upon a field of 64 squares orchestrated in an 8×8 network. In a paper written by Cleveland [2] on the psychology of chess more than a century ago, he asserted that game of chess is so widely popular at all times in all places because it appeals to the fundamental instinct of combat and that this game is a competitive game of skill – an intellectual one – as opposed to other sports that require bodily dexterity. Rather, the game of chess requires a comprehension of rules of chess, awareness of pieces, analysis of movements, and evaluation of positions. Because of its association with intelligence, chess has been employed for intellectual and social-emotional enrichment in the field of education [3]. Although playing chess does not have enough evidence of a significant effect towards academic outcomes, attention, and creativity, a field experimental evidence confirmed that it can improve scores in mathematics and reduces the incidence of time inconsistency[4]. In the field of technology, chess automation has been getting more popular due to the recent advancements of the field.

The concept of an automated chess presented in this study is not new in the literature and has been accomplished by several authors using different technologies and techniques. Two of the most common mechanisms for automating

The 3rd International Conference on Automation, Mechatronics, and Robotics 2021: ICAMEROB 2021
AIP Conf. Proc. 2502, 040001-1–040001-9; https://doi.org/10.1063/5.0108986
Published by AIP Publishing. 978-0-7354-4400-3/$30.00

the game of chess are the robotic arm that lifts and positions the pieces, and a magnetic slider underneath the board in accordance to an *x-y* Cartesian coordinate system. For instance, Angelkov, et al. [5] built an automated chessboard with a robot arm that uses a computer vision system to continuously monitor and track the movements on the board. Another similar work is a chess-playing robot build by Ómarsdóttir, et al. [6] called *Chessmate* where four elements were combined such as an electrical components, a software component, and two physical components. Both of these studies used robotic arms to automate the game. On the other hand, Mendes, et al. [7] built an automated physical platform called *Chess.Automated* through the use of Arduino Mega ATmega 2560, servo motors, sensing membrane keypad, and hybrid stepper motor and *x-y* plotter mechanism. Similarly, Jariyavajee, et al. [8] built an interactive chess board through the use of microcontroller to provide players a new experience to the game of chess. All of these existing studies were used as an inspiration for *Chessbot* and several improvements were made to further its kind.

In an attempt to contribute to the game of chess as well as to the field of automation and technology, *Chessbot* was built through the combination of various technologies and techniques such as voice command, *x* and *y* plotting and recognition of chess pieces in a Cartesian plane, artificial intelligence algorithm, and Android mobile application. This automated chess game follows three different modes of gameplay such as a one-player mode (a human player plays against a computer that can physically move the pieces on its own), two-player mode (a human versus another human player), and a special game mode for blind players made possible by the built-in voice command feature. In the case of two-player mode (or multiplayer game format), *Chessbot* automatically assigns the first player to use white chess pieces, and the second player to use black chess pieces. After the match, *Chessbot* will automatically restore the chess pieces to their starting position in preparation for the next one. It will also prompt the user if they want to save the moves they have made during the game - although the maximum limit will depend on the storage space availability. To evaluate *Chessbot*, several matches were simulated to perform various testing procedures such as voice command recognition, move log accuracy, display and user input acceptance accuracy, self-arrangement, and chess move reliability.

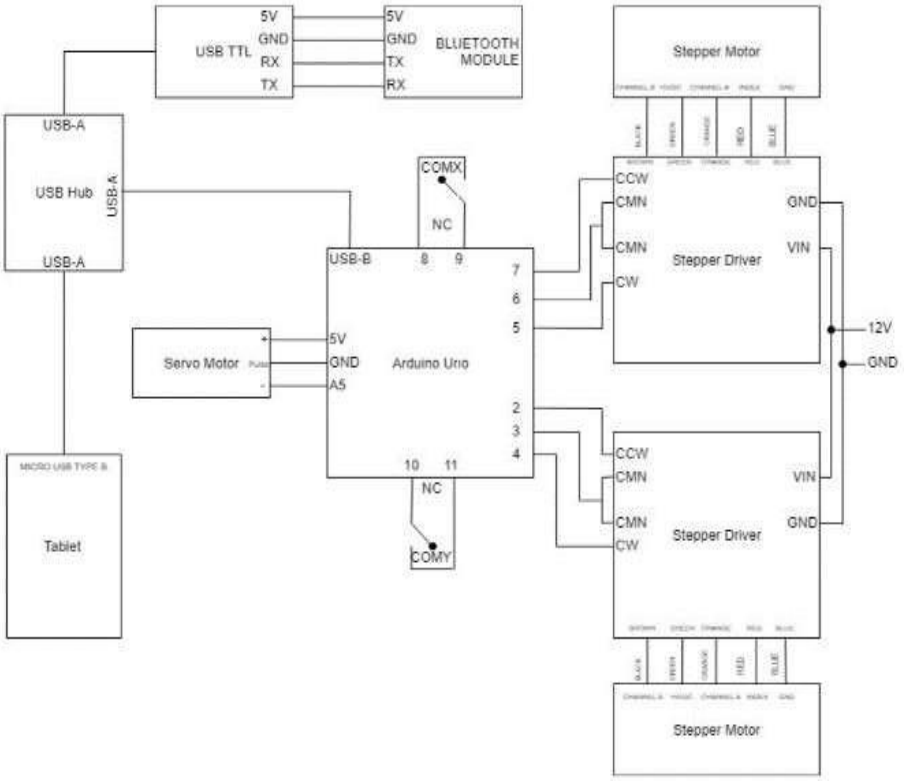

FIGURE 1. Schematic Diagram of *Chessbot*.

HARDWARE AND SOFTWARE COMPONENTS

The main goal of this research study was to build an automated chess board that accepts voice commands from a player and performs self-moving of pieces as initiated by both player and computer. Apart from revolutionizing the way people play chess, the inspiration of this project came from the idea of allowing disabled people to enjoy and be competitive without limitations. Garcia and Pilueta [9] had the same goal when they built *VISIMP* – a portable communications device for visually impaired individuals. For *Chessbot*, several hardware (Figure 1) and software components were developed, utilized, and combined. The first step was to build the automated chess platform using the following:

- **Microcomputer**: For this specific prototype, a tablet computer acts as the microcomputer and display module where the source code is installed and all input and output signals are processed.
- **Microcontroller**: The microcontroller, Arduino Uno ATmega 328, provides the instruction to the positioning module. It converts the instructions sent by the microcomputer to a language that the positioning module will understand in terms of the movement that needs to be done to get to the right position.
- **Bluetooth**: This module is responsible for ensuring that the voice commands issued by human players are received and processed by the microcomputer. HC-05 was used to add full-duplex wireless functionality.
- **USB-TTL UART**: Connected to the Bluetooth via wire connectors, both hardware works together to support voice command recognition that allows microcontroller to process voice commands and perform actionable events.
- **Stepper Motor**: Together with the stepper motor driver, this component is responsible for executing precise movements to position the pieces to the right landing squares on the board. Stepper motor driver, on the other hand, receives and transmits electric signals to drive these stepper motorswith precise position control.
- **Servo Motor**: This rotary actuatoris responsible for coupling and de-coupling the magnet underneath, which enables to position pieces over the board that represents physical movement of each move played.The magnet underneath is placed in a platform that raises or lowers depending on the movement it has to perform.
- **Linear Guide**: The linear guide moves the positioning system through *x-y* axis in reflection of the chessboard above it. It positions the magnets along with the stepper motor and driver to the correct position.
- **Limit Switch:** This electromechanical device detects the presence and position of all the components while ensuring that any object does not exceed whatever commands were given and executed.

(a) (b) (c) (d)

FIGURE 2. Illustration of game user interface *i*n the Android application with focus on the chessboard and (a) the initial state, (b) when a player taps a piece, (c) after selecting the landing square, and (d) the opponent's' action.

After setting up the hardware components, the next step was to develop the mobile game chess application (Figure 2). As for the Algorithm used, pipe programming was used. This is a technique for passing information from one program process to another. Basically, a pipe passes a parameter such as the output of one process to another process which accepts it as input. It is also a connection between two processes, such that the standard output from one process becomes the standard input of the other process. The researchers used Java programming language using Android Studio to implement pipe programming and several custom Java classes were coded such as (1) *spot* to represent one block of the 8×8 grid, (2) *piece* that represents the basic building block of the game, (3) *board* that represents an 8×8 set of boxes containing all active chess pieces, (4) *player* that represents either a human player or a computer, (5) *move* that represents the starting and ending spot, and the (6) *game* class that controls the overall mechanics of the application

that is based on the rule of chess [10]. This mechanics includes subroutines under different modules such as the (1) *positioning* module for moving one's pieces, capturing opponent's pieces, and self-arrangement, (2) *gameplay* module for setting up a one-player, two-player, and a special game mode for disabled individuals, (3) *opponent* module that integrates a chess artificial intelligence framework that uses alpha-beta algorithm to search for the next best move, and (4) *voice* module which is for processing commands via Google Speech. Python programming language was used to bind the C language from the microcontroller.

In building the said research study, the researchers followed the engineering design process which includes: (1) State the problem, (2) Generate ideas, (3) Select a solution, (4) Build the item, (5) Evaluate and (6) Present results.

For step 1, the researchers stated that in building this project will allow disabled people to play chess with enjoyment and be competitive without limitations. For step 2, the researchers were able to generate ideas through the different RRL's, brainstorming, data gathering and interviews. For step 3, Organizing all gathered raw facts, the researchers were able to select the best solution to build the prototype. For step 4, Building the prototype is one of the hardest part, wherein the researchers chose the best components and applications that fits the requirement of the project. For step 5, After careful step in building the prototype comes the evaluation, in the evaluation, the researchers let the target respondents to try and test the prototype several number of times in terms of the following parameters: voice command recognition, self-arrangement, recording of movement of chess pieces, capturing opponent's pieces and other subroutines. And eventually for step 6, The researchers were able to present the results with the projects' reliability.

FIGURE 2. Hardware Decomposition of Chessbot.

TESTING PROCEDURES

For the evaluation of *Chessbot*, ten matches (five games of human-human and five games of human-computer) were simulated to perform various testing procedures such as voice command recognition, move log accuracy, display and user input acceptance accuracy, self-arrangement, and chess move reliability. The procedures were as listed below:

Voice Recognition Test

1 Switch on the prototype
2 Choose the two-player mode
3 Move white from A2 to B3
4 Move black fromA7 to B6

Self-arrangement and Move Log Accuracy Test

1 Switch on the prototype
2 Choose the two-player mode
3 Move white from F2 to F3
4 Move black from E7 to E5

5	More white from F2 to F3	5	Move white from G2 to G4
6	Move black from E7 to E5	6	Move black from D8 to H4
7	For white's turn, declare surrender	7	Wait for the prototype to arrange the pieces
8	Wait for the pieces to be arranged	8	Compare logs with the prototype
9	Record the results	9	Record the results
10	Repeat procedure with random actions	10	Repeat procedure with random moves

Display and User Input Acceptance Accuracy Test

1. Switch on the prototype
2. Choose one-player mode
3. Choose any difficulty
4. Choose Human vs. Human mode
5. Play randomly using the game app
6. Switch off the Prototype
7. Record the results
8. Repeat procedure with random action

Chess Move Reliability Test

1. Switch on the prototype
2. Choose two-player mode
3. Move a piece using the app
4. Move a piece using voice command
5. When all the pieces are move declare surrender.
6. Wait for all the pieces to be arranged
7. Record the results
8. Repeat procedure with random moves

TESTING RESULTS

After ten chess matches, findings show that *Chessbot* performed the tasks correctly with 90% in voice command recognition, The researchers used string as commands for better recognition instead of just characters. Example of which were Alpha, Bravo, Charlie, Delta, Echo, Foxtrat, Golf and Hotel.100% in move log accuracy, 100% in display and input acceptance accuracy, 90% in self-arrangement, and 90% in chess move reliability. The one test wherein *Chessbot*failed to determine the voice command was due to the presence of a noise interference. It affected the accuracy of the system's interpretation of the voice command issued by the user even after Google Speech tried to autocorrect the given command. In the case of self-arrangement, the failed attempt occurred when the platform holding the magnet underneath the chessboard was not able to lower itself and carried the chess piece that was connected to it. Meanwhile, the chess move reliability failed on one chess match as well where the opponent's piece was moved in accordance to the player's previous movement. Upon checking, the issue was with the platform as it failed to lower itself when it went to position itself to the next chess piece. To correct the issue, the proponents placed drops of oil on the platform mechanism to assist in the mechanical movements of the platform.

4.1. Voice Recognition Testing

Objective:

- To test the systems functionality to recognize voice patterns

Table 2—Voice Recognition Testing Results

Trial Number	A	B
1	✓	
2	✓	
3	✓	
4	✓	
5		✓
6	✓	
7	✓	
8	✓	
9	✓	
10	✓	

Formula:

$$R = \frac{\text{Successful number of test}}{\text{Number of test}} \times 100$$

$$R = \frac{9}{10} \times 100 = 90\%$$

Table2 shows there sultsf the voice recognition testing.ColumnAareforresultsthat were successfully recognized by the system while column B are for results that wereunsuccessfully recognized. The one result wherein the system failed to determine the command was due to the presence of a noise interference. It affected the accuracy of thesystem'sinterpretationofthevoicecommandissuedbytheuserevenafterGoogleSpeechtriedtoautocorrectthegivencom

mand.However,asshownonthetableabove,itwasstillabletorecognize90%oftheuser'svoicecommands,whichalsoshowsthatthesystemhas90%accuracywhichisdependentonthelevelofnoiseinterferingwiththevoicecommandsissuedbytheplayer.Thevoicecommandsweregiveninaclearvoiceandwithminimalnoisetoavoidsoundinterferencethe mobile phone device mightpickup.

4.2. Move Log Accuracy Testing

Objective:

- To test the accuracy of the move logs displayed by the prototype

Table 3—Move Log Accuracy Testing Results

Trial Number	A	B
1		✓
2		✓
3		✓
4		✓
5		✓
6		✓
7		✓
8		✓
9		✓
10		✓

Formula:

$$R = \frac{\text{Successful number of test}}{\text{Number of test}} \times 100$$

$$R = \frac{10}{10} \times 100 = 100\%$$

Table3showsthenumberoftestsconductedand theresultsofeach testsperformed to determine the accuracy of the move logs of the system. Column A is forresultsthatshowsthesystemwasnotabletodisplaythecorrectmovesthatwereperformedduringthegame.ColumnBisfortheresultsthatshowsthesystemwasabletodisplaythecorrectmovesexecutedduringthegame.Table10showsthesystemhasnodifficultydisplayingandrecordingthemovesthatwereexecutedduringthegame.Basedontheformulaforreliability,thesystemwasabletoreach100%reliabilityforthemove logs.

4.3. Display and User Input Acceptance Accuracy Testing

Objective:

- To test the display of the system
- To test if the prototype can accept the input from the user after choosing from the display

Table 4—Display and User Input Acceptance Testing Results

Trial Number	A	B	C	D	E
1	✓	✓	✓	✓	✓
2	✓	✓	✓	✓	✓
3	✓	✓	✓	✓	✓
4	✓	✓	✓	✓	✓
5	✓	✓	✓	✓	✓
6	✓	✓	✓	✓	✓
7	✓	✓	✓	✓	✓
8	✓	✓	✓	✓	✓
9	✓	✓	✓	✓	✓
10	✓	✓	✓	✓	✓

Formula:

$$R = \frac{\text{Successful number of test}}{\text{Number of test}} \times 100$$

$$R = \frac{10}{10} \times 100 = 100\%$$

Table4showstheresultsofthe 10trialsdoneonthe systemforthemenudisplayand the acceptance of user input by the system. Column A is the result of the menu forSingle player and Two players displayed on the screen. Column B is the result fordisplaying the levels of difficulty; Easy, Medium, and Hard. Column C shows the resultfor the display of color selection after choosing Human vs. AI option. Column D showstheresultforthereturnoptiondisplayedonthescreenoneverysub-menu.ColumnEshowstheresultforwhentheuserprovidedaninputtothesystem.Onthetestsconducted,10outof 10 times the system was able to show the display for the menu and their correspondingsub-menus when the user selected from the

display. This means that the system was abletorecognizethecommandsissuedbytheuserwhenselectingfromthemenuandwasthenable to show the next option after choosing. In short, the system was able to reach 100%reliability on its display and the recognition of the user's input after choosing from thedisplay.

4.4. Self-Arrangement Testing

Objective:

- To test the self-arranging functionality of the prototype

Table 5—Self-Arrangement Testing Results

Trial Number	A	B
1		✓
2	✓	
3	✓	
4	✓	
5	✓	
6	✓	
7	✓	
8	✓	
9	✓	
10	✓	

Formula:

$$R = \frac{\text{Successful number of test}}{\text{Number of test}} \times 100$$

$$R = \frac{9}{10} \times 100 = 90\%$$

Table 5 shows the results of the testing conducted to determine if the prototypewas able to return the chess pieces to their original position. Column A shows the resultforwhenthesystemwasabletoreturnthepiecestotheirstartingposition.ColumnBshowsthe result for when the system was not able to return all chess pieces to their startingposition. There was one instance during the testing that the system was not able to returnall the chess pieces to their starting position. That incident occurred when the platformholding the magnet underneath the chessboard was not able to lower itself and carried thechess piece that was connected to it via magnet also moved to the next chess piece forarrangement. The issue was corrected when the proponents put oil on the mechanism toassisttheplatform'smovements.Outofall10experimentsconducted,thesystemwasabletoreturnallchesspiecestotheirstartinglocation9timesshowingthatthesystemhas90%reliability in self-arrangement testing. However, the results also show that the positioningsystem has 100% accuracy as it was able to position the chess pieces to their respectiveposition. The 90% was achieved because of an issue encountered on the hardware of thesystemwhich wascorrected.

4.5. Chess Move Reliability Testing

Objective:

- To test the reliability of the chess moves

Table 6—Chess Move Reliability Testing Results

Trial Number	A	B
1	✓	
2		✓
3		✓
4		✓
5		✓
6		✓
7		✓
8		✓
9		✓
10		✓

Formula:

$$R = \frac{\text{Successful number of test}}{\text{Number of test}} \times 100$$

$$R = \frac{9}{10} \times 100 = 90\%$$

Table6showstheresultsofthetestingperformedtotestthereliabilityofthechessmoves. Column A shows the result for when the system was not able to place the chesspieces to their correct position. Column B shows the result for when the system was ableto place the chess to their correct position. 10 test runs were performed to determine thereliabilityofthesystemintermofthemovementofthechesspiece.There wasoneinstancewhere the system placed the chess piece to its correct position however when the systempositioned the opponent's chess piece, the previous piece it placed went with it. Whencheckedthe issuewaswiththeplatformasitfailedtoloweritselfwhenitwenttopositionitselftothenextchesspiece,becauseofthisconcernt hegroupdecidedtomarkthisasthesystem not able to position the chess piece to its correct position. To correct the issue, theproponents placed drops of oil on the platform mechanism to assist in the mechanicalmovements of the platform. The remaining results showed that the system was able toposition the chess pieces to their intended position therefore displaying 90% reliability inpositioning the chess pieces. It also shows that the prototype has 100% accuracy in themovementofthechesspiecesasitwasabletoremovecapturedchesspiecesofftheboardandposition thechesspiecesto their respectivelocation.

CONCLUSION

This study aimed to build an automated chess board by combining a multitude of technologies and techniques to work as one in performing the rules and mechanics of the game of chess. To determine whether these mechanics were achieved, *Chessbot*was put through a series of testingsuch as voice command recognition, move log accuracy, display and user input acceptance accuracy, self-arrangement, and chess move reliability. In an actual match, players who play with real chess set often get pressured. Therefore, towards a complete chess telepresence experience, it would be interesting to measure and analyze both quantitative and qualitative feedback of players and disabled individuals on how they play and perceived an automated chess board. Because this is a preliminary study, it would be beneficial as well to improve the mobile game application and leverage chess game matchups between players with similar level of skills. Nevertheless, *Chessbot* is already a functional, automated chess board that any player can enjoy.

ACKNOWLEDGEMENT

The authors would like to thank FEU Institute of Technology for funding the research presentation and publication.

REFERENCES

1. K. Masukawa, "The Origins of Board Games and Ancient Game Boards," in Simulation and Gaming in the Network Society, Singapore, 2016, pp. 3-11: Springer Singapore.
2. A. A. Cleveland, "The Psychology of Chess and of Learning to Play It," The American Journal of Psychology, vol. 18, no. 3, pp. 269-308, 1907.
3. R. Aciego, L. García, and M. Betancort, "The benefits of chess for the intellectual and social-emotional enrichment in schoolchildren," (in eng), Span J Psychol, vol. 15, no. 2, pp. 551-9, Jul 2012.
4. A. Islam, W.-S. Lee, and A. Nicholas, "The Effects of Chess Instruction on Academic and Non-cognitive Outcomes: Field Experimental Evidence from a Developing Country," Journal of Development Economics, vol. 150, p. 102615, 2021/05/01/ 2021.
5. D. Angelkov, N. Koceska, and S. Koceski, "Automated Chess Playing with a Robot Manipulator," International Journal of Engineering Issues, vol. 2, pp. 45-51, 2015.
6. F. Y. Ómarsdóttir, R. B. Ólafsson, and J. T. Foley, "The Axiomatic Design of Chessmate: A Chess-playing Robot," Procedia CIRP, vol. 53, pp. 231-236, 2016/01/01/ 2016.

7. A. R. Mendes, A. M. Mehta, and B. H. Gohil, "Implementation of the Automatic and Interactive Chess Board," IOSR Journal of Electrical and Electronics Engineering, vol. 9, no. 6, 2014.
8. C. Jariyavajee, A. Visavakitcharoen, P. Sirimaha, B. Sirinaovakul, and J. Polvichai, "A Practical Interactive Chess Board with Automatic Movement Control," in 6th Global Wireless Summit, 2018.
9. M. B. Garcia and N. U. Pilueta, "The VISIMP Portable Communications Device for Visually Impaired Individuals – Development and Feasibility Study of an Assistive Technology," Journal of Critical Reviews, 2020.
10. International Chess Federation. (2018). Laws of Chess. Available: https://handbook.fide.com/chapter/E012018.