Chess automation System using GRBL and XY plotter system

Madhu Palati[1], S Balajee Devesha[2], Anand Venkitaramana[3], Haripriya S[4], Raksha Bowade[5]
[1]*Assistant Professor, Department of Electrical and Electronics Engineering, BMS Institute of Technology and Management, Bangalore, Karnataka, India*
[2,3,4,5] *B.E Students, Department of Electrical and Electronics Engineering, BMS Institute of Technology and Management, Bangalore, Karnataka, India*

***Abstract*- Automated chess is a physical platform that enables the game of chess between humans and Artificial Intelligence. In this paper various designs for the mechanism of automation and control are analyzed, the pros and cons of each method are discussed. In this paper a methodology of automation of a game of chess by the use of CoreXY plotter and magnetic piece actuation with a Console-based chess engine for validation and G-code based motion control is presented.**

Index Terms—Chess automation, Core XY plotter, G-code, Magnetic actuation, Minimax, Alpha-Beta Pruning, Chess engine, GRBL

I. INTRODUCTION

Chess is a strategic game played on an 8x8 chequered board for about 1500 years all over the world. With the development in the fields of automation and artificial intelligence over the last decade, various conventional mechanisms have been automated. Chess, being one of them. In this paper, the game of chess is automated using a core XY plotter interfaced with control softwares. The system is equipped with a chess engine, which enables a game of player versus Artificial Intelligence. A push button keypad acts as an input device, which sends the data serially to the processor, which could be a computer or a raspberry pi. The move is validated, converted into coordinates, and further into distance measurements. This instruction converted into g-code facilitates the physical movements via GRBL compiler. This paper focuses on creating an economical Automated Chess Board (ACB) which is an effective tool for the visually impaired with the addition of braille script to the keypad.

1.1. Literature Survey

In this section, different methods used for movement of chess pieces are discussed. Robotic arm, Computer Vision Based Chess Playing Capabilities for the Baxter Humanoid Robot, X-Y Cartesian table and Interactive chess board are some of the methods for movement of chess pieces [1-5]. The main movement based methods are discussed in this section

1.1.1 Methods for movement of chess pieces

A. Robotic Arm based approach

A robotic arm can be equipped with software and hardware manipulations to modify it to play chess against a human opponent. A robotic arm manipulator can be programmed with some open source chess engine to function. Image processing techniques are used to monitor and record the moves.

The projects that have implemented robotic arm based automated chess are [3-4]:

a) An Self-directed Chess-Playing Robotic System-Gambit

b) Baxter Humanoid Robot with computer-vision based Chess Playing Capabilities

1. Gambit

It is a robot manipulator that is constructed to

i. Discern the board and the pieces

ii. Understand the game state (real time monitoring)

iii. Physical manipulation of the pieces

iv. Coordinate with the opponents moves

The robotic arm of Gambit is built to have 6 DOF which enables position and orientation control. A hollow gripper jaw is used with a rubber tip joint to pick up the pieces. A camera is fitted at the gripper such that covers the entire workspace [4][10].

Fig.1 Gambit an Autonomous Chess-Playing Robotic System

2. Baxter Humanoid Robot with computer-vision based Chess Playing Capabilities

Baxter is a two-armed with an animated face industrial robot designed and manufactured by Rethink robotics. Baxter is programmed to play against a human opponent under dynamic environment with changing lighting conditions and board positions. Baxter is capable of playing two simultaneous game with each arm if programmed accordingly [3].

Fig.2 Baxter humanoid robot

The working methodology of Baxter humanoid robot is controlled by three subsystems:

- Computer-vision subsystem for observation
- Chess-engine subsystem for calculation
- Mechatronics subsystem for triggering

B. X-Y Cartesian Table

X-Y Cartesian table is used for the movement of chess pieces on the board and is shown in fig.3. Entire chess board can be covered by a two-dimensional coordinate system. Movement along both the axes is done by the pinion and rack mechanism using dc motors. The speed and direction of rotation of the dc motors are controlled to get rotation angle as per the requirement [7].

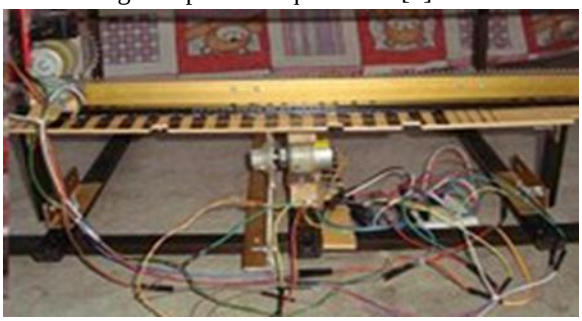

Fig.3 X-Y Cartesian table

C. Enactment of the instinctive and collaborating chess board- square off

The Model of automatic and interactive chess board- SQUARE OFF is shown in fig.4. In this method, the entire process in divided into the input and the output cycle. In input cycle consists of setting the origin, detecting the coordinate of the played move and processing it. The output cycle consists of Stepper motor movements, servo action and Voice and Visual feedback mechanism. When a move is made, a voice feedback is given indicating the initial and final position of the chess piece. Square off, can be connected to app and can be physically played from anywhere in the world [2].

Fig.4 Model of automatic and interactive chess board- SQUARE OFF

1.1.2. Methods for move detection

i) Reed switch- wizard chess

Wizard chess uses X-Y Cartesian table for movement of chess pieces and reed switches for its detection. By applying magnetic field, the reed switch is operated. The switch may be activated by an electromagnetic coil or by getting a permanent magnet near to the switch. The switch returns to the original position When the magnetic field is removed. Reed switches can be used for the detection of moves made on the chess board and is shown in fig.5

+2	+3	+4	+6	+5	+4	+3	+2
+1	+1	+1	+1	+1	+1	+1	+1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
-1	-1	-1	-1	-1	-1	-1	-1
-2	-3	-4	-6	-5	-4	-3	-2

Fig. 5 Reed switch-Wizard chess

ii) Image Processing

This can be performed in four discrete stages:

- First stage is to recognize the board, light condition and supplementary initial parameters.
- Second stage includes knowing the positions of pieces.
- Third stage to spot and find moves.
- Fourth stage involves to transform moves into algebraic notation.

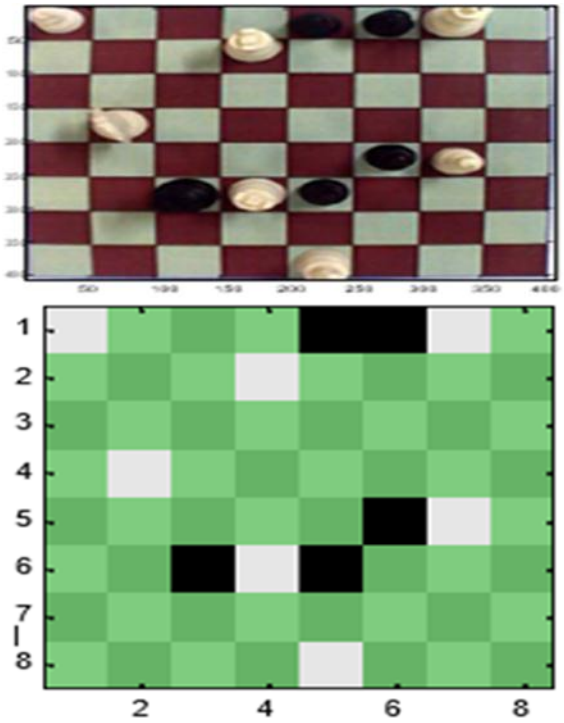

Fig.6 Deduction and move recording based on intensity of pieces recorded

1.2 Objectives

The objectives of the present work is

i) To analyze various techniques used in automation of chess board in terms of design and control of movements and monitoring mechanisms.

ii) To understand the methodology and working of various methods and come up with best combination of technologies to establish a commercially viable product.

iii) To explore the application of automated chess for the visually impaired.

II. METHODOLOGY

The general flow chart is shown in fig.7

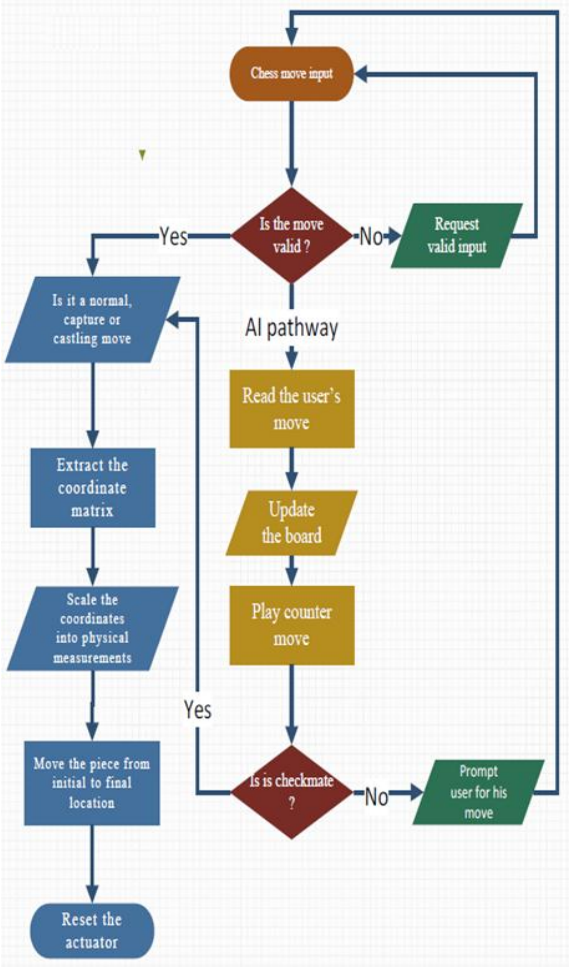

Fig.7 Flow chart of the proposed system

Step by step procedure for making the movement of pieces is explained in this section.

Step 1- Input from the user.

In case of player verses AI game, player is always assigned white as default. Usually in the game of chess white always starts first. The game begins by requesting a move from the user. The move is represented by using standard chess notations.

The move is indicated by mentioning initial position of the piece to the destination to which it has to be

moved. though in standard practice the move is indicated by mentioning the letter to a particular piece, it is not done so here to make things simpler.

Letters A to H and numbers 1 to 8 are used to label the Squares in chess board. The left most square of the white player is labeled as A1. From to format is used to record the move.

Using regular expression, the user's input is first separated as initial and final expressions. This data is separated into rows and columns for the positions. According to standard notations, rows are already indicated by numbers so data is processed directly. As the columns are designated by alphabets, they are assigned corresponding numerical value for processing the data.

Step 2- Processing the move

1) After the user inputs the move, the validity of the move is tested. If the move is valid, it is executed by the following steps. In case of an invalid move, the chess engine requests a valid move from the user.

2) Once the move is validated, the program determines the nature of the move.

a) A normal move is indicated by initial and final positions only. Example a2a4.

b)A capture move is indicated by a numeral (1 is set as default)

c) A castling move is detected by four set moves, they are

i) White side : Right castling — e1g1
Left castling — e1b1

ii) Black side : Right castling — e8g8
Left castling — e8b8

3) After the nature of move is determined, the distance and route to be moved by the piece is sent to the plotter in the form of gcode.

a) The plotter calculates all distances with respect to the origin. Once to move is received by the gcode, the distance to be moved in X and Y direction is calculated by subtracting the initial position from the final position of the piece.

b) In case of capture, first the captured piece is removed from the board and placed at a distance of about one and half times of dimension of each square.

c) For a castling move, the king is first moved and then rook is taken to its position.

d) To understand the method of movement, let us consider an example of moving bishop from c1 to g4(validated move).

i.As the initial position is c1, the rod latches on to the piece at c1 and moves it half of the reference square value (say, 'd') to the right (default first step).

ii.Then the row difference is calculated, in this case its 3 and 'd' is subtracted from this value and the piece is moved 21/2 steps forward in this case. Now the column is calculated, in this case its 4 and 'd' is subtracted from this value and the piece is moved 41/2 steps right in this case.

iii.As the final position is g4, the piece moves it half of the reference square value forward and the rod unlatches (default last step), placing the piece at its destination.

iv.The subtraction of the reference is done to ensure that the movement if the piece takes place only along the lines of the board and not in between the squares.

v.The magnetic rod always reverts back to the origin position after the move.

Step 3- AI/ Player 2 response

Once the AI's move is generated, similar calculations are done by the gcodes and sent to plotter. This process continues until one of the sides or one of the player forfeits presents checkmate.

2.1 Software based aspects

This implementation uses Python language [8] for major operation processing then the

STEP I: Initialisation of AI and the chess board. Obtaining the movement string which is in the form of a Packed Move e.g. a2a4 for regular moves and 1a4a5 for capture moves, either from Keyboard directly or by using the Push button matrix. The main source code allows the option to use either the computer's keyboard or attach a serially connected Push-button keyboard using arduino as the piloting serial controller and validation of the move.

STEP II: Conversion of the Packed moves into initial and final position coordinates

Step III: Correlation of the coordinates into physical measurements and writing the g code file. Now, this is the main step for the functioning of the program, as in this step the program differentiates the move from Capture move > Castling Right > Castling Left > Regular move.

STEP IV: After the calculations are done too obtain the measurements, they are then stored in variables which in turn are used to create the actual G Code instructions based on the type of move.

STEP V: Analysis of the move by ai and physical implementation of the move on the chess board and ai updating the user move. Then the AI plays a suitable counter move and the same process is repeated from the beginning.

STEP VI: The G code is streamed serially to the XY plotter by a function

2.2 Chess engine setup

Chessnut is the chess engine being used for all the moves and chess logic. In this work for finding the best move, the possible chessboards 3 levels deep and depth first search, minimax and alphabeta pruning are generated by utilizing a tree based on the heuristics [9] material (total piece count for each player), number of possible legal moves with emphasis on centre squares ,check/checkmate status and pawn structure. The recursive function uses a lot of computing power so calculating heuristics on board states more than 4 levels deep takes a lot of time. With a depth of 3 levels, our AI makes pretty good moves but also makes a lot of ill-advised ones as well. The AI's chess intelligence is estimated to be at a level 3 out of 9. The AI file uses back ends based on the following chess move optimization algorithms:

2.2.1. Minimax Algorithm

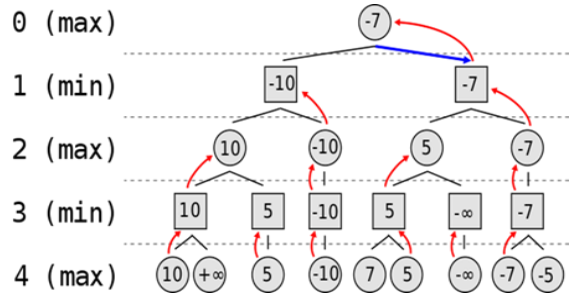

Fig.8 Minimax Algorithm

With respect to chess, the player to act is the maximizer, whose move would be met with an adversarial response from the opponent (minimizer). The minimax algorithm shown in fig.9 assumes that the opponent is competent and would respond by minimizing the value (determined by some heuristic) of the maximizer.

2.2.2 Alpha-Beta pruning

Because of the number of board states possible in chess (estimated to be 10^120), using a layer of alphabeta pruning minimax can be improved. By keeping track of beta (the lowest value guaranteed to the minimizer) alpha (the highest value guaranteed to the maximizer), avoids calculating the heuristics of certain board states that cannot improve the situation for the current player. The greyed-out leaf node with a heuristic of 5 is never explored because the maximizer, at that point, is guaranteed a 5 by going left. The minimizer would choose the 4, if the value of the greed-out leaf node is larger than 4. If it is less than 4, the minimizer would choose it instead. From the maximiser's viewpoint, there is no reason to investigate that leaf node. The prototype model of chess board along with XY-plotter is shown in fig.9

Fig.9 Prototype model of chess board

III. RESULTS AND DISCUSSIONS

The main program converts a given string literal move into physical measurements and implement it on the physical and the model output after the movement of chess piece is shown in fig.10 and the program was successful. The program was tested for different moves, and the results were as expected.

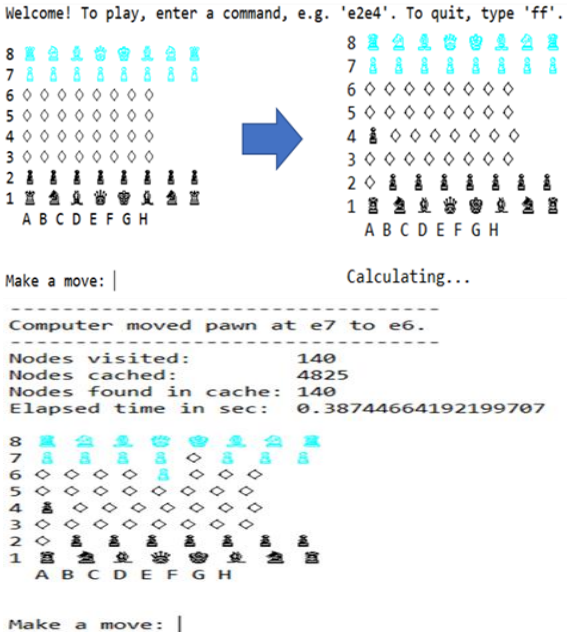

Fig.10 Output displayed after the movement of chess piece

IV.CONCLUSION

In the present work, an attempt has been made to design and develop a low cost, low weight Chess automation system with improved aesthetics that enables the user to play chess without direct interaction. After a lot of trial and error and referring various research papers, one of the most effective methodology of chess automation was found to be the use of AI based piece tracking mechanism with magnetic pieces on the board moved by a Core XY plotter with a metal rod acting as the magnetic latch actuated by servo motor. AI method is used to assess effectively the pieces on the chessboard, can be manipulated by the XY plotter and thus providing an efficient system.

ACKNOWLEDGEMENT

The Authors gratefully acknowledge the encouragement of Principal, EEE HOD and Management of BMS Institute of Technology, Bangalore in carrying out this work.

REFERENCES

[1] S. Sarker, "Wizard chess: An autonomous chess playing robot," IEEE conference proceedings of International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), Dhaka, 2015, pp. 475-478.

[2] Allen R. Mendes, Atur M. Mehta, Bhavya H. Gohil, “Implementation of the Automatic and Interactive Chess Board”, IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE), Volume 9, Issue 6, 2014, pp 01-04.

[3] A. T. Chen and K. I. Wang, "Computer vision-based chess playing capabilities for the Baxter humanoid robot," Conference proceedings of 2nd International Conference on Control, Automation and Robotics (ICCAR), Hong Kong, 2016, pp. 11-14.

[4] C. Matuszek et al. “Gambit: An autonomous chess playing robotic system”, conference proceedings of IEEE International Conference on Robotics and automation, China,2011.

[5] G.D.Illeperuma, “Using Image Processing Techniques to Automate Chess Game Recording”, Proceedings of the Technical Sessions, 27 (2011) pp.76-83, Institute of Physics – Sri Lanka

[6] J. Balata, Z. Mikovec and P. Slavik, "Problems of blind chess players," 6th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), Gyor, 2015, pp. 179-183.

[7] G. Du, S. Bi, Y. Xiao and W. Li, "The compliance control study of Chinese chess robot in Cartesian coordinate system", Proceedings of the 2013 International Conference on Advanced Mechatronic Systems, Luoyang, 2013, pp. 31-35.

[8] Wrutz, T., Golz, J., & Biesenbach, R. (2017). “An internet platform for open-source robot offline programming interface RoBO2L.” 14th International Multi-Conference on Systems, Signals & Devices (SSD), 2017.

[9] M. Bikos, Y. Itoh, G. Klinker and K. Moustakas, "An Interactive Augmented Reality Chess Game Using Bare-Hand Pinch Gestures," International Conference on Cyberworlds (CW), Visby, 2015, pp. 355-358.

[10] Cynthia Matuszek, Brian Mayton, Roberto Aimi, Marc Peter Deisenroth, Liefeng Bo,Robert Chu, Mike Kung, Louis LeGrand, Joshua R. Smith, Dieter Fox, “Gambit: An Autonomous Chess-Playing Robotic System”, IEEE International Conference on Robotics and Automation Shanghai International Conference Center,May 9-13, 2011, Shanghai, China