2015 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)
19-20 December 2015, BUET, Dhaka, Bangladesh

Wizard Chess: An Autonomous Chess Playing Robot

Sunandita Sarker
Mechanical Engineering
Bangladesh University of Engineering and Technology (BUET)
Dhaka, Bangladesh
sunanditask@gmail.com

***Abstract*— Robotic chess is an emerging problem domain in the field of human-robot interaction. In this paper, Wizard Chess, an autonomous chess playing robotic system is introduced, which is capable of recognizing all possible chess board states, generating subsequent moves as well as executing those moves. Existing chess playing robots use camera for the detection of game states and articulated robotic arm for the movement of pieces on the board which make these robots particularly complex and highly expensive. Compared to these prior works, Wizard Chess is architecturally simple, low-cost and feasible in case of personal use, educational and recreational purposes. In this system, an X-Y Cartesian table is used for moving chess pieces along the board. Additionally, sixty-four magnetic reed switches are used for determining the current game state. A prototype of the proposed robotic chess is implemented and an extensive performance evaluation is presented in this paper which shows the feasibility of the system in real-life situations.**

Keywords— robotic chess, autonomous system, x-y table, multimedia system, entertainment and games

I. INTRODUCTION

Robotic board games pose to be excellent research topics in the field of human-robot interaction. Among various board games played autonomously by robots, chess has always been on top of interest for the researchers. Recently it has been included in several school curricula. Moreover, it has also turned out to be a popular source of recreation among elderly and disabled persons as it requires deep attention, but minimum physical strength and fitness. However, in many cases, elderly people have to play chess alone in the absence of a companion. Therefore, in this paper, design and development of an autonomous chess playing robot, Wizard Chess is discussed. It has three main functionalities; recognizing chess board states, computing subsequent moves and executing these moves by means of an X-Y Cartesian table.

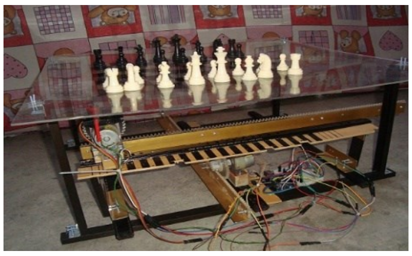

Fig. 1. Wizard Chess ready to start a new game

Playing board games like chess presents several challenges that include observation of the board and game pieces, reasoning about the game and the game states, handling of the chess pieces, as well as coordinating with the human opponent. Wizard Chess is an electro-mechanical device that consists of an X-Y Cartesian table and a sensory chessboard. It is a fully autonomous, compact and low-cost robot that can serve educational and entertainment purposes. This paper is focused on the algorithms used to recognize the moves done by the human player and to control the X-Y table to perform a machine-generated move.

This paper is organized as follows. Section II shows the related works done in this sector. In Section III, main components of the robot are described. In Section IV, the detailed workflow of the proposed system is presented. Performance analysis of a prototype of the robot is shown in Section V. Finally, in Section VI, we conclude the paper with future research directions.

II. RELATED WORK

In previous years, a large number of chess playing machines were planned or constructed. The first chess program was developed by Alan Turing [1] in 1950 as an example of what a computer can do. However, it was a simplified paper version of the game. The first full-fledged game of chess by a computer was written by Alex Bernstein in 1957 at MIT [2]. However, IBM first introduced a chess program named Deep Blue that was considered to be better than a human player [3]. This chess program managed to defeat chess master Garry Kasparov.

Currently chess programs are accessible to any regular consumer. There are many chess engines that can be downloaded for free from the internet e.g. Stockfish, Crafty, Fruit, GNU Chess etc. These engines can run on a personal computer and surpass even world champion caliber players at blitz and short time controls.

In addition to the research on chess engine in the AI region, enthusiasts and researchers were also working on the development of automated chess playing machines. The first attempt was taken in late eighteen century by building "Mechanical Turk" [4]. It was created by Wolfgang von Kempelen and was exposed in the early 1820s as an elaborate hoax. In 1910, Leonardo Torres first created a chess-playing machine which was not based on fraudulent concept [5]. In later years, he built a second one with his son and both machines are still in working order in the Torres Quevedo Museum of the Technical University of Madrid. Furthermore, "Boris Handroid" was a commercially advertised chess playing machine introduced in 1980. 'Phantom', 'Novag' are some other dedicated chess playing robots. In 2010, "The Gambit Robot" was built in collaboration between Intel, UW, and Alium Labs [6]. It uses a manipulator arm designed for small-scale use. In recent years at K.J Somaiya College of Engineering, Vidyavihar, Mumbai-India [7], a team implemented an automated interactive chessboard. A 8X8 Sensing Membrane Keypad is used in their prototype to simulate the 64 boxes on a chess board which is relatively complicated and expensive.

Most of the prior works on robotic chess-playing systems involve a robotic arm for piece movement and a camera for perception of the board. However, systems using such an articulated robotic arm having 5 or 6 joints with servomechanism are significantly complex and intricate. To control all the servo-motors simultaneously, more computational power and memory capacity are needed. Besides, these systems are also noticeably large and expensive as the body links used in these robotic arms are milled from aluminum or alloy steel.

Compared to these approaches, our proposed Wizard Chess differs in the aspect of simplicity and feasibility. It is a low cost and compact chess playing robot that shows reliable and consistent performance. For instance, in Wizard Chess only two DC motor with feedback sensors are controlled whereas movements of the motors are independent of each other. Mechanical simplicity makes it practical in case of personal use for educational and recreational purposes.

III. Components of the Robot

In this section, the main components of Wizard Chess (WC) Robot is described. It consists of four essential parts; a sensory chessboard, magnetic chess pieces, an X-Y Cartesian table and a chess program. The Chessboard is attached on top of the X-Y Cartesian table. Electric circuit on the robot is connected to the computer via USB connection.

A. The Chessboard and The Chesspieces

Generally, a common game of chess is played with a wooden chess board and 32 chess pieces. The length of each side of a standard board varies from 32 to 40 centimeters, whereas the side length of each small square section is between 40 and 50 millimeters. However, in Wizard Chess, the chessboard is custom designed in order to integrate it with the mechatronic system.

As discussed earlier, we instrumented the chess board instead of using camera and pattern recognition algorithms for the perception of the game. The instrumented chessboard of WC consists of three layers. 64 black and white square sections are drawn on a 2 millimeter thin acrylic board at the top layer. Beneath the top layer, a sensor layer of magnetic reed switches with wires and pull up resistors are placed and connected to the controller board. Another sheet of acrylic is placed at the bottom to protect the sensor layer. These three layers are bolted together on top of the mechatronic X-Y Cartesian table.

Reed switch is a sensor which needs a magnet for actuation. This sensor may operate in a normally open mode, a normally closed mode or a latching mode. In the normally open mode, when a magnet is brought towards the reed switch (or vice versa), the reed blades are closed. Similarly, when the magnet is withdrawn, the reed blades get opened. We used in total 64 normally open mode reed switches, each beneath a small square section.

In our system, small sized magnets are attached to all the chess pieces. The length of each square section is restricted by the magnetic field area of the magnets used in the chess pieces. In the prototype we developed, the length of each square section was reduced to 25 millimeters resulting in the 20 centimeters long chessboard. We ensure that magnet underneath one chess piece activates only the reed switch beneath the square it occupies, not any other nearby reed switch.

We used magnetic chess pieces available with travel chess box in our prototype. However, the orientation of each magnet was made uniform so that the North Poles of all the magnets were faced upwards. It was necessary for the proper operation of the magnetic gripper attached to the X-Y table (details in Section III-B).

B. X-Y Cartesian Table

Chess is a turn taking game between two opponents. Normally, in a robotic chess game one opponent is human while the other is a machine. Regardless of the opponents being human or machine, a board game like chess needs frequent piece movements for turn completion. There are several options in choosing the means of manipulation such as an articulated robotic arm, over hanging pick and place robot etc. In Wizard Chess (WC), we introduced an X-Y Cartesian table with magnetic gripping system beneath the chess board. The design choice was made to build a robotic system with minimum complexity and low-level framework.

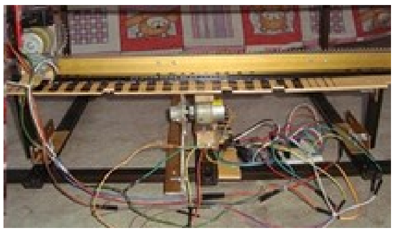
Fig. 2. Mechanical structure of the X-Y table

In WC, we need to move chess pieces on a plane surface. Therefore, we considered a 2 dimensional Cartesian co-ordinate system. The X-Y table has two degrees of freedom movement in two horizontal layers. The bottom layer is designed for longitudinal (Y-axis) movement while the upper layer completes movements in transverse (X-axis) direction. Both of these movements are done using rack and pinion mechanism. A channel bearing carries a DC motor that drives a pinion over the rack and moves the structure. The rack and pinion set in the X-axis moves along the Y-axis with its motor. Both the rack and channel bearings are supported on aluminum beams. These beams are light-weight but strong due to their M-shaped cross section.

In commercially available X-Y table, we frequently notice the use of stepper motors and rotary encoders for precise movement along an axis. Each step of the signal provided to the stepper motor gives a certain amount of rotation. If a rotary encoder is connected to the motor, it can give any error feedback. However, in our system we employed a different approach using low cost DC motors. We know that the rotation angle of the DC motor cannot be controlled directly, although the speed and the direction of rotation can be controlled. When power is given to the motor, it moves the pinion on the rack and covers a certain distance. In our system, we used a sensor module and a feedback control loop to maintain the distance covered by the pinion.

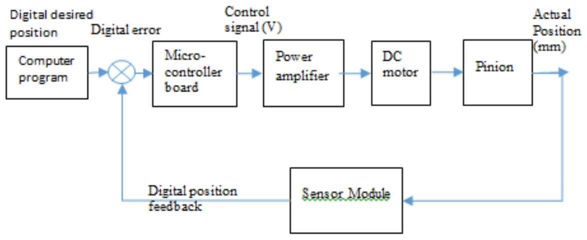

Fig. 3. Feedback control loop of the motor movement

The sensor module moves above a paper strip which is covered with black lines. Each of these black lines is 2 millimeters thick and placed at a uniform distance of 2 millimeters. LDR, LED, and resistors are used to build the sensor modules. The circuit diagram of a sensor module is given in Figure 4. When a sensor module passes over a black line, it gives a high analog value. This analog value is read by the ADC pin of micro-controller and the travelled distance is calculated by counting the total number of black lines passed.

Distance travelled (millimeters) = 2 * Number of black lines passed (millimeters)

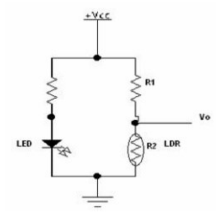

Fig. 4. Circuit diagram of the sensor module

This system gives us movement with precision of 2millimeters which is enough for our operation. Above the DC motor of X-axis, we attached a servo motor that rotates at a certain angle when it is commanded by the controller. It carries a stick that bears a magnet on one end. This stick with the magnet is the end effector of the table. Both the DC motors cover a particular distance commanded by the computer program and then the servo motor flips the stick. The intended chess piece gets attached to the magnet with the servo motor and then the motor moves the piece to the desired square section.

C. *Chess Computer Program*

The Wizard Chess is controlled with an Arduino Mega. The controller is connected to a computer running on Windows 7 operating system over USB connection. We built a program in Matlab that handles the entire collaboration operation of the robot. It analyses the board states, recognizes moves by the human player, passes the move to the chess engine and passes machine generated moves to the Arduino Mega for execution.

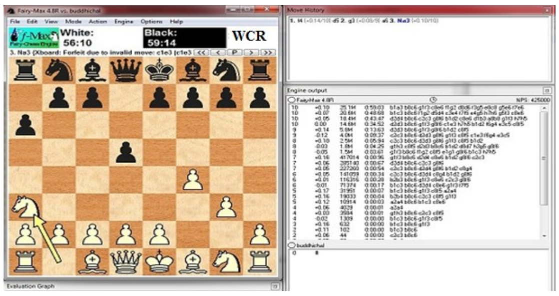

Fig. 5. GUI of Winboard software

We did not build the chess engine; instead we used an open-source software named Winboard. Winboard interfaces two chess engines. It is commonly used by developers to test their chess engine against other established chess engines such as Crafty, Fairy-max, Pulser etc. Our Matlab program operates as a pseudo chess engine and provides a move of the user as input to Winboard in a specific text file. Selected chess engine then generates a machine move and passes it to the Matlab program. This move is finally executed by the X-Y table.

IV. Proposed System

A complete game of chess with Wizard Chess Robot is composed of four important tasks. In this section, these tasks are explained elaborately.

A. *Analyzing The Chessboard*

An important part of the system is to analyze the chess board and to recognize the current game state. For this reason, two things are needed to be determined; whether there is a chess piece on each of the squares and what kind of piece it is. In Wizard Chess, 64 reed switches are fixed under each of the 64 squares of the chess board and connected to the controller board. As we discussed in Section III-A, each chess piece contains magnet underneath it. Thus, when a chess piece occupies a square, the reed switch under that particular square closes the circuit and sends a high value to the controller board. In this situation, that particular square is said to be activated. At the start of the game, all the squares of the first two rows on each side are activated. In this way, we detect the presence of a chess piece on any square of the board.

+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. 6. Assigned values to the 2D array in the initial state

Now, we need to identify the type of the chess piece present on different squares of the board. An 8 by 8 two dimensional integer array is used to save the current game state. Each array element corresponds to each of the 64 squares. The zero value for an array element denotes that the corresponding square is unoccupied. On the other hand, different integer values are used to recognize different chess pieces. For example, king, queen, bishop, knight, rook, and pawn are enumerated as 6, 5, 4, 3, 2, and 1 respectively. White pieces are given positive values whereas black pieces are given negative values. Figure 6 shows the values assigned to the 2D array at the initial state.

B. Move Detection

After each move by the user (or the machine), the 2D array is updated to record the new game state. The reed switches are scanned continuously for stable values to recognize if the user has finished his/her move. Two cases can arise after each move by the user (or the machine). One, the user (or machine) moves one of his/her pieces to an unoccupied square and two, the user (or machine) captures a piece of the opponent.

In the former case, values of two array elements are changed. The reed switch corresponding to the previous square position of the moved piece gives low value to the controller board and thus the respective array element is assigned a zero value. However, the previous value of this array element is saved in a variable. Let this variable be *movedPiece*. The new square position of the moved chess piece is detected by the high value of the corresponding reed switch. Consequently, the array element of this square which was previously zero valued is now assigned the value of *movedPiece*. The change in the array is interpreted and saved in a specific text file, which Winboard can access.

In the second case, the capture of an opponent's piece by the user is done in three steps – removing the captured piece of the opponent from the board, taking out the piece used for capturing from its current square position and placing it in the square of the captured piece. At first, the user removes the opponent's piece from the board. Thus, the reed switch under the square of the removed piece gives low value to the controller board. Consequently, the corresponding array element is assigned a zero value and the previous value of this element is saved in a new variable called *capturedPiece*. After the first step, an unoccupied square position is created in the board which will be occupied by the capturing piece of the user in the next two steps. Thus, a scenario similar to the former case is produced where a piece is moved to an unoccupied square and the 2D array is updated accordingly to record the new game state.

This procedure is continued until the game state corresponding to a checkmate position is reached or the game is stopped willingly by the user.

C. Move Execution

After a move is generated by Winboard, it is saved in a text file. The main Matlab program reads this file and decodes the move in a simple co-ordinate manner. Now these co-ordinates are sent to Arduino via serial communication and moves are executed accordingly by means of the X-Y table.

V. Performance Analysis

The Wizard Chess was able to play complete games with several human opponents and an extensive analysis was made on the overall performance. Our prototype showed relatively less problems in detecting user moves and generating machine moves accordingly. However, we note that most of the problems encountered by our system were caused by off-center placement of a piece by the mechatronic system. Using a PD controller in the motor feedback system this error was reduced significantly.

In our analysis, we evaluated our system in three parts. Firstly, we checked the problems encountered in user move detection. We took 40 trials and analyzed the data. Secondly, we analyzed success rate in picking a piece and placing it in the right destination. We took 20 trials without the PD controller and 40 trials with the PD controller. Lastly, we analyzed the prototype playing an entire game of chess for ten times. It was able to finish the game uninterruptedly 6 times. Most of the errors encountered during the full games were caused by the serial connection between PC and Arduino.

TABLE I. Satistical Data of the Performance of WCR

Name of the test	No. of trials	Success rate
Human move detection	40	87.5 % (35 times)
Piece pick up with P controller	20	45% (9 times)
Piece pick up with PD controller	40	77.5% (31 times)
Piece drop with P controller	20	35% (7 times)
Piece drop with PD controller	40	82.5% (33 times)
Complete chess game	10	60%

These results suggest that our system presents a reliable and robust chess playing robot. Although the data set is small the overall result suggests a high potential for further improvement.

VI. Conclusion and Future Work

This paper gives an overview of the design and implementation of an autonomous chess playing robot, Wizard Chess which was built from locally available, simple and low cost material. The software part is constructed in a separate module so that it can be easily changed or upgraded.

In future, we intend to use a single board computer such as Raspberry Pi inside the mechatronic system to make our proposed robot more user-friendly and robust. This will eliminate the use of a personal computer to run the chess program and make it an independent gaming console. We also intend to include voice recognition system to provide user moves so that disabled persons can also be benefited from our system.

References

[1] A. Hodges, "Alan Turing: The Enigma" Vintage edition 1992.

[2] A. Bernstein, M. Roberts, T. Arbuckle, and M. Belsky, "A chess playing program for the IBM 704", in the Proceedings of the 1958 Western Joint Computer Conference,1958.

[3] Hsu, Feng-hsiung,"Behind Deep Blue: Building the Computer that Defeated the World Chess Champion",Princeton University Press, 2002.

[4] S. Simon, "Enlightened Automata", in "The Sciences in Enlightened Europe, Chicago and London" The University of Chicago Press, 1999.

[5] "Torres and his remarkable automatic devices", Issue 2079 of Scientific American, 1915.

[6] C. Matuszek et al. "Gambit: An autonomous chess playing robotic system", in IEEE International Conference on Robotics and automation, China,2011.

[7] A. R. Mendes et al. "Implementation of the Automatic and Interactive Chess Board", IOSR Journal of Electrical and Electronics Engineering, Volume 9, Issue 6 Ver. I , 2014.