Chess Brain and Autonomous Chess Playing Robotic System

Hafiz Muhammad Luqman
Faculty of Electrical Engineering and Information Technology, Technical University Chemnitz
Chemnitz, Germany
hafiz-muhammad.luqman@s2013.tu-chemnitz.de

Mubeen Zaffar
Department of Electrical Engineering
COMSATS Institute of Information Technology
Lahore, Pakistan
mubeenciit@gmail.com

***Abstract*—In this paper, a 4 degree-of-freedom (DOF) chess playing robotic manipulator and computer vision based chess recognition system are presented. The robotic system is capable of playing chess game autonomously against human or another robotic system. The logical system consists of anti-glare camera mounted on the chessboard, which acts as an eye of robot, personal computer for run-time implementation of computer vision algorithm, an open-source chess engine used as chess brain of robotic manipulator, which plays chess algorithmic ally on behalf of robot, and of course robot itself. On development side, a simple image segmentation based computer vision algorithm was developed to find the legitimate chess move and human hand motion detection, a software system was developed that enables the robot to pick and drop chess pieces from prescribed chess boxes, lastly communication channel was exploited for interfacing computer vision algorithm with chess engine. The whole robotic system is formulated from recycled machine parts and open source tools. The convincing performance of computer vision algorithm and robotic manipulator testifies by its 1st position in IEEE Final Year Project Competition Lahore 2012.**

Keywords- Computer Vision; Chess Engine; Autonomous Robot; Human Robot Interaction.

I. Introduction

Chess is an ancient, intellectual and noble game in which two players separated by chessboard challenge the mental capacity of each other. While playing chess, we utilize all of our cognitive resources. The chess game starts with 20 possible moves and about 400 possible moves after first move each and this number exponentially increases with each move. After a few moves the possible moves becomes so complex that neither can imagine the all possible moves [1]. No one thought that one day fast speed hardware clocks and sophisticated software will be able to solve such complexity level and even surpass the human intellectual capability.

The chess game has been used as a model in the field of artificial intelligence since 1950s. Developing a chess program is consider to be the first discrete problem in the field of artificial intelligence and computation. This problem has discrete rules and clear goal (capturing King). The computer handles this complexity as tree problem. The root of the tree describes the current position of the game and each branch of the tree leads to the next possible legal move. It was assumed that if we solve the chess problem computationally, then we would be able to solve any other problem too. The strategy to find the best move (best tree) is called minimax [2]. The minimax strategy is used with alpha-beta pruning [3] which prevents minimax from going into branches that cannot produce better result than previous branches have already produced. The efforts to develop chess playing machine was even started before the existence of computer. First most, an American mathematician Claude Shannon proposed computing routine for general purpose computer which was theoretical algorithm of chess program [4]. In 1950A British Mathematician and computer scientist Allan Turing transformed this idea into a probably the first ever chess program 'TURBOCHAMP' which could not run on a computer. In 1957, a Mathematician and IBM employee wrote first complete computer decode-able chess program which ran on IBM 704. It took about 40 years to build a computer chess program which could beat chess champions. In 1997, IBM created first best and brightest chess playing engine called Deep Blue [5]. It was the first chess program that beat the Chess Champion Garry Kasparov in a 6 game match held in New York.

Though the realization of autonomous chess laying robot seems to be of last decade, but astonishingly it dates back to 18th century. The chapter of autonomous chess playing robot was started in 1770, when a Hungarian engineer, Baron Wolfgang presented first so called chess playing robot, “The Turk” or “Chess Playing Automan”. The Turk had privilege to beat the statesman of that time including Napoleon Bonaparte and Benjamin Franklin. The Turk remained center of fascination for 86 years until its destruction by fire in Chinese museum in 1854.

Several serious efforts have been made in the last decade to realize chess playing robot on industrial and academic level. Some used the magnetic sensor beneath each box to detect move and while some used digital chessboards [6-7]. Such techniques, exclude the image processing part and decrease the complexity level of whole project. The current work is an effort to develop a robotic system which can play

978-1-5090-2255-7/16 $31.00 © 2016 IEEE
DOI 10.1109/ICARSC.2016.27

board games. In this work, chess game was focused to challenge the dignity of human brain. The process control flow is very simple and shown in Fig. 1. First of all, a camera visualizes the board and sends the capturing frames to computer vision algorithm. It saves the current state of the board and then goes to sleep mode until a move is detected. When a move is made, it is fed to chess engine using Chess Engine Communication Protocol. Chess engine processes input move and proposes its move to the robot in algebraic chess notation. The proposed move is then sent to customized controller (consists of 4 stepper motor drivers and PIC micro-controller) of robot over a serial port. Controller yields the directions to the actuators to execute this proposed move mechanically.

The remainder of this research paper is as follows: Section II describes the computer vision algorithm for chess move recognition and its implementation. Section III discusses the development of the robotic arm. Section IV describes the robot controller and Octave interfacing with chess engine. Section V describes system performance and proposes possible future enhancement.

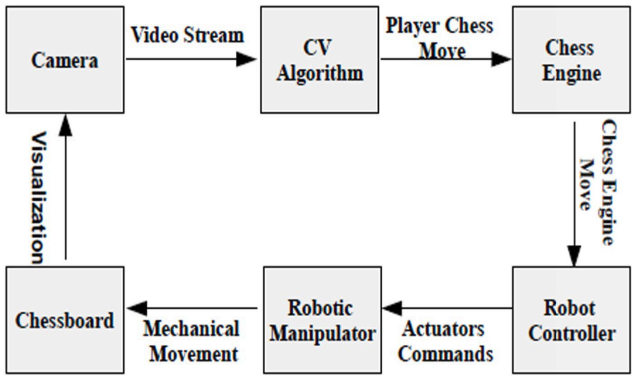

Fig. 1.Process Flow

II. Image Processing Part

The recognition and analysis of visual activities in diverse environment is a challenging task. In this study, a real-time computer vision algorithm is proposed for chess moves recognition and hand motion detection. The series of basic image processing operations are performed to accomplish the desired goal. The consistent chessboard design simplifies the task.

Several complex approaches have been adopted to analyze the chess game [8-12]. But the methodology we followed is quite straightforward. It does not use complex or computational expensive visual detection algorithms to abide by the real-time constrain of our computer vision algorithm. Firstly, a background registration technique [11] is used to construct a base for background subtraction in later stages and its position will also be used as coordinate system for positioning of moving pieces. Then the moving object regions are separated by implementing change

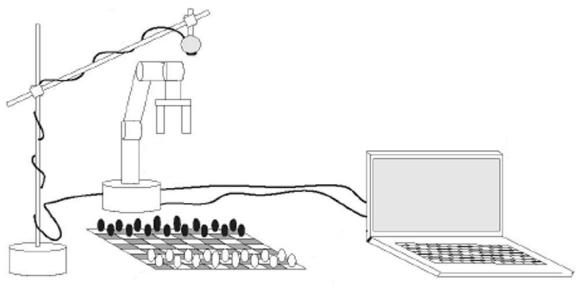

Fig. 2. Camera and Game Setup [7]

detection based segmentation algorithm which yields moving objects(chess pieces). Hand motion detection is implemented by computing the absolute difference of two consecutive images to measure the interval of move making process by human. Eventually, a post-processing step is applied on the object image to remove noise regions and to the smooth the object boundary for positioning.

A. Camera Setup

A USB web-cam is used to keep the optimal quality and economic. It provides the best quality images in low light conditions and avoids reflections. The pixel resolution of video is 640x480 in RGB format. The camera is mounted on chessboard at height about 18 inches to visualize the board from top-view. The frame is cropped to see only board, it saves our computation power and to avoid processing extra detected objects. The color-map of frames are converted from RGB to grayscale so that basic image processing operation can be applied.

B. Image Distortion

The image is a 2D digital signal whose values are arranged in a matrix. Each value (pixel) gives information about the environment it captured. In practical imaging, some factors alter these values or distort the image [13-14] which cannot be avoided and have to be encountered. The initial experiments were performed on animated images which gave ideal results. As animated images do not have any kind of noise, contrast imbalance, low intensity or distortion. When the actual images were captured in real environment, all these factors become bottleneck for attaining successful results. Octave gives us very handy tools which helped us to implement the complex operations with a single command.

C. Image Enhancement

In chess game scenario, chess pieces are positioned close to each other that causes shadows in the neighboring boxes. A per-processing gradient filter is applied on the input frame to reduce the shadow effects. Sometime, standing people around the board occluded the light from their side which makes brightness of board non-uniform and also affects the contrast which is critical for the detection of chess pieces on the board. Image contrast enhancement is a crucial problem in image processing [14-15]. There are several image processing techniques for the contrast enhancement in Octave (histogram equalization, histogram stretching, image sharpening, image intensity adjustment) which can be applied. In our case, image adjustment function (imadjust) worked best. It changes the intensity values of input grayscale image in such a way that only 1% values saturated around high and low intensities of the real image.

D. Chessboard Detection

The first step in chess game analysis is to detect the chessboard. The edge detection based segmentation technique is used to segment the whole chessboard into 65 segments. This segmented frame will be used in background subtraction in later stages. It makes the system computation faster and accurate. The chessboard was printed on a flex banner. The colors selection of boxes was made on the basis of their grayscale values in such a way that, they give maximum contrast. It helps us to extract edge information. Two sets of color was chosen, In first case, White and Orange colors were chosen for chessboard boxes and Black color for outside region of the board. In second case, conventional White and Black colors were chosen for boxes and Gray color for outside region of chessboard. In later case, the black chess pieces were marked with different color on the head to make them detectable on black boxes. The chessboard has a consistent and unique structure which

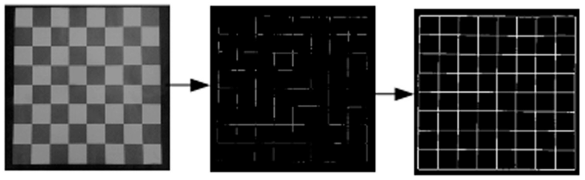
Fig. 3. Chessboard Detection

simplifies board detection problem. The first step before placing the pieces on the board is to extract the position of corners of each box on the board and of the board itself and assign a label to each box (65 segments). The canny edge detection algorithm [16] is used to detect boxes boundaries and pieces following by dilation operation [17]. Now the camera position should be fixed until the end of game.

E. Move detection

When the initial process of setting a game has been completed and the camera has saved the background of the board. It sets to wait mode until hand motion is detected. As soon as it detects the motion in the video stream, it is set to alert state and keeps on checking the video frames unless motion is vanished. Now it captures another image and subtracts it from the image that was saved before motion and calculates the resultant image which shows change in the game situation. After performing some image processing operations the resultant image is shown in Fig. 4. The detected piece coordinates are calculated and examined in

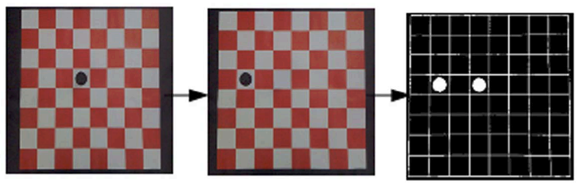
Fig. 4. Chess Move Detection

which box these pieces lie. After determination of box number, chess move was assigned according to Algebraic Chess Notation. As the piece location was found by its position coordinates so the pieces should be placed in the center of box. Otherwise, it will not be able to find the box accurately. The overall process flow is described in Fig. 6.

III. Chess Engine Interface

Chess Engine is a piece of software which is capable of battling against humans on chessboard. It is a beautiful creation of Artificial Intelligence [19-20]. Fundamentally, chess engine does not come with GUI. It comes as a console application and a separate GUI (Winboard/Xboard) is interfaced to play interactively. Chess engine and GUI application communicate according to a protocol. The latest protocol is Universal Chess Interface, which is being used in this research. Chess moves are described, sent and saved in Algebraic Chess Notation [20]. It is the standard chess notation of FIDE[1]. In Algebraic Notation each box is represented by unique alphanumeric number as shown in Fig. 5. The columns are named from a to h and rows are named from 1 to 8. Each piece is named by uppercase letter of its name such as King by **K** and Queen by **Q.** Moves are represented by source box to destination box such as if King's black pawn moves one step, this move is represented by e7e6. If a move is capturing a piece then a 'x' is inserted between source and destination box, which indicates that move is of capturing a piece. Every move has own its specific representation. Chess Engine sends moves in Algebraic Chess Notation to GUI applications. They

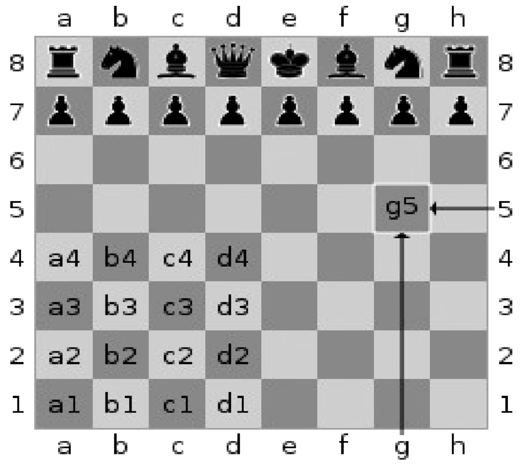

Fig. 5: Algebraic Chess notation [21]

1 Federation Internationale des Echecs

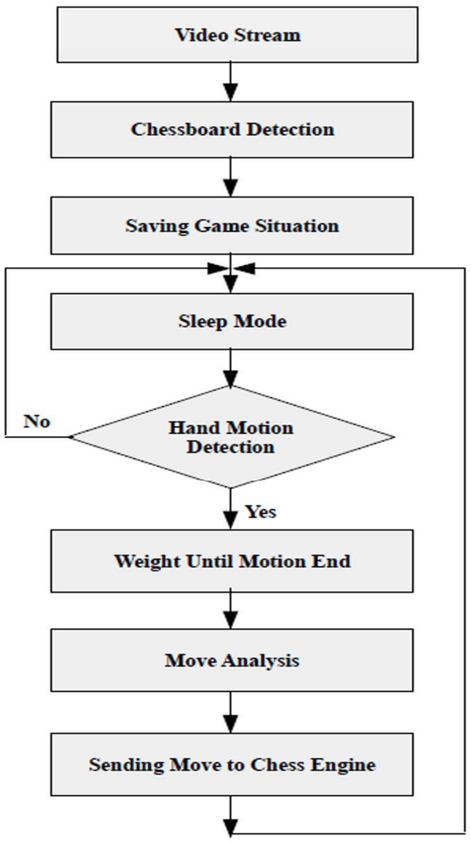

Fig. 6. Image Processing Control Flow Diagram

interpret text and exhibit it in the form of graphical changes. In this case, the chess engine output move was the input of robot controller software . A communication channel had to be developed between chess engine and controller software. For this interfacing problem, the pseudo-chess engine logic was used developed by a Nepali student [22]. A fake chess engine was used to extract chess engine output in the form of algebraic chess notation. It works through file handling. Image processing algorithm writes its moves in the input file for chess engine. Chess engine reads the input file and writes its output on the output file which is red and sent to the robotic arm controller.

IV. Robotic Manipulator Structure

The designing, construction and trajectory planning of a robotic arm without blueprint of its structure is a challenging task specially if you do not have prior experience and knowledge about kinematics and inverse kinematics [23]. The designing part needs attention as you have to take care of work space limitations, movement speed and minimum degree of freedom is required to implement task with smooth flow and high precision and accuracy. Trajectory planning is one of challenging tasks as it requires lot of direct and inverse kinematics calculations [24]. Your whole solution depends on the structure of the robot. Industrial robotic manipulators could also be the option to accomplish chess playing task. But mostly robotic manipulators are designed with respect to specific automation purposes and for heavy material handling. Their motion trajectory and serving capability is restricted by their design. The chess playing robot is playing in humanly environment, it requires to be sophisticated and lightweight structure and human arm like motion while playing chess. While picking and placing chess piece, it should not collide with neighboring pieces. It requires accuracy of about one-third of width of a chess box. Therefore, robotic manipulator was not a choice for us. The proposed robotic manipulator can execute pitch motion only. It is made from aluminum scrap pieces. Every component of robot was recycled from dis-assembled machine parts. The structure of the robot is an assortment of the already developed chess playing robots[25-26]. The sliding robotic arm was designed to avoid programming complexity of yaw motion. The intended column is accessed by the sliding movement of the robot and intended row is accessed by robotic arm joints movement. The robotic arm is sliding on an aluminum rail driven by two stepper motors. The aluminum was used to keep the weight of robot minimum. The four stepper motors were used as actuator of robot, while a DC motor was used for the opening and closing of the robot end effector to decrease the complexity of the electronic circuit and program. The stepper motors were driven at half-step mode to attain the accuracy in millimeters. The mechanical structure of robotic arm is as follows;

Upper arm is connected with the base of the robot through shoulder joint. The other end is connected to the elbow (arm between shoulder and elbow). Shoulder joint is responsible for moving the whole upper arm. Shoulder’s actuator is connected with the upper arm via timing belt.

Fore arm connects elbow and wrist just like a human arm. The elbow motor is responsible for the fore arm’s motion. The length of fore arm is 30cm.

Wrist plays crucial role in lifting the piece from desired box without disturbing the neighbouring pieces. When robotic arm reaches the desired box, it's upper and forearms reach above that box and then wrist moves downward for the accurate piece picking and placing.

End effector was specifically designed in such a way that its movement does not disturb the neighboring pieces. Its structure is made from aluminium and rubber inside to hold the piece firmly. The motion of gripper is controlled by DC motor. A screw is used to control the opening and closing of gripper jaw. When wrist reaches at specific height, the screw opens the gripper jaw to pick the piece and closes when it holds the piece. The joint movement of robot is programmed to make the wrist movement at 90 degree upwards so that it does not collide with neighboring pieces. The robotic arm and its assembly are fixed on a wooden board and chessboard is placed in-front of robot at fixed position to keep the every box of chessboard reachable for robotic jaw.

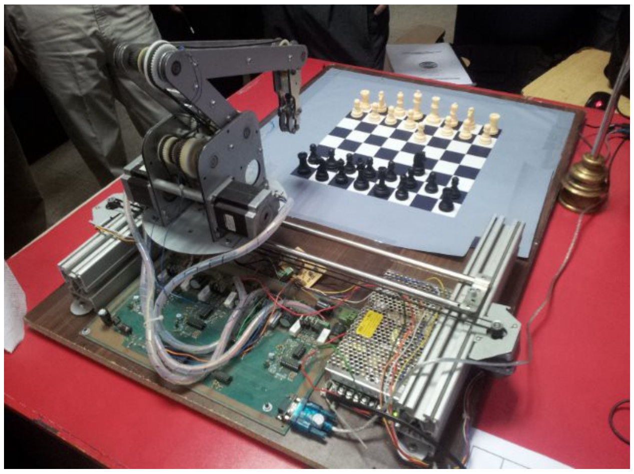

Fig. 7. Robot playing chess in IEEE Final Year Project Competition Lahore, 2012

It makes the robotic system robust against the position of chessboard.

V. Controller and Working

The scope of this section is to present the architecture of autonomous robot controller (ARC) based on architectural construction approach [27]. The aim was to develop an adaptable controller, so it was indispensable to establish architecture of software who can respond to changes easily later when required. The software is divided into application specific and robotic-arm part. All the basic robotic arm related functionalities especially actuators controller speed, power supply and communication part control by this part. Application part handles specific application related task. Development of both parts is independent from each other. This allows to deploy this robotic system in various applications.

The 8-bit PIC16F877A microcontroller was used as primary controller of the robot. It is based on modified Harvard architecture. It is a 40 pin powerful microcontroller with 14kB flash memory, 356 Kbyte RAM. The low cost (2 Dollars) and powerful PIC architecture made it the best choice for this kind of application [28]. The hardware also includes four stepper motor drivers on the same PCB board. The control unit is connected with computer through serial bus.

To implement chess move mechanically was one of the complex and hectic part of robot playing chess. The chess engine proposes the next move for robot. Octave is used for serial communication with robot controller. Chess engine writes its output on a text file. Octave reads the text file and sends it to controller of the robot over serial bus operating at 9600 baud rate. The control program parse the move which is sent in algebraic notation and calculate the number of cycles for each stepper motor to execute this move mechanically. There are two possibilities, either capturing a piece or simple placing a piece from one place to the other. In capturing case, robot first lifts the captured piece and place it out of chessboard at prescribed position and then placed the capturing piece on new box. In case of simple move, it transports the piece from one box to the other box. The castling, en-passant and all other special moves are possible with simple parsing but at the moment could not implement yet, which is drawback of this system. Holding the any piece(from queen to pawn) from a fixed height is decided at which gripper is closed. The length of the each piece below the jaw grip is same in either case e.g. pawn or queen. The length of piece varies above the jaw grip. At the moment, the chess piece recognition functionality has not been implemented yet. So, it cannot detect to which piece it is going to play with.

The chessboard image is segmented into 64 boxes and the centre position of each box is saved in a 2D matrix. When a box number is given to controller software, it finds box's column and row number and then starts its journey towards the box. All robot parameters to perform any move have been authenticated by successive experiments. The symmetrical design of chessboard simplifies the process of finding steps required for each actuator to reach any box. To perform any move, robot first recalls its current position, then it calculates each actuators movement required to reach the desired box. It slides over the relay and reaches calculated column, leans towards the calculated row and lift the piece. It repeats the process again, finds the destination box where it has to place the piece in a box or out of chessboard. Again it calculates the destination column and row and places the piece there. The last position of the robot is saved which will be used next time at the start of move implementation.

VI. Conclusion and Future Enhancement

An autonomous robotic system is described in this paper which is capable of autonomously playing chess against human or another robotic system without human assistance. The non-sensory chessboard and pieces are used to keep the ingenuity of the game. The primary objective of this work was to develop a prototype for research work and to enhance practical experience by implementing learned theoretical knowledge. The game situation analysis is entirely based on computer vision part. In a pre-set-up environment (Lab) robot can play game with quite good efficiency but in an unknown environment the light changing and shadowing affects the performance of the system. To tackle this issue, the threshold value of edge detection algorithm is adjusted manually to keep the performance of the robotic system optimal. The robot can join the play from any point of game, unless the Portable Game Notation (PGN) file is loaded in the chess engine. In case, human player try to cheat or the image processing algorithm feeds wrong move to the chess engine, the chess engine will send back the notification of wrong move and will ask the player to undo the move and make legal move. The time of making a move is the function of distance of targeted chess piece from the current of position of robotic arm and the type of chess move (capturing or simple). The whole system was developed from open source tools. It does not use any sensory chess pieces or sensory chessboard. The system was designed in such a way that to play in any environment and with normal chess piece and chessboard. The robotic manipulator is entirely made from recycled machines parts. The stepper motors are driven on half-step mode to achieve the accuracy in millimetres.

The possibility of improvement is a fundamental property of every research work. The improvement can be done at computer vision algorithm to improve its efficiency in diverse environment. The real-time adaptive threshold techniques can be applied through which the parameters can be adjusted automatically. In this way code can be run on a DSP/embedded system. The illegal move detection functionality can be added; if human player makes a wrong move, chess engine alarm the wrong move and lets the human to undo that move and play the right move. The Unix pipes can be used to interface chess engine with computer vision algorithm. On hardware side a complete embedded system can be developed which can process all tasks (CV algorithm, chess engine and robotic arm control) on a single system on chip.

VII. Acknowledgment

The project was done under the department of Electrical Engineering of COMSATS Institute of Information and Technology Lahore. We are greatly thankful to our parents who greatly supported us. We are also thankful to Mr. Antonio Cebollas and Mr. Khawar Zia for helping us in completion of this work.

References

[1] Allen Newell, J. C. Shaw, and H. A. Simon, "Chess-Playing Programs and the Problem of Complexity," *IBM Journal of Research and Development*, vol. 2, pp. 320-325, 1958

[2] Michel Willem, "Linking Theorem", in *Minimax Theorems: Progress in Nonlinear Differential Equations and Their Applications* , Berlin; Birkhäuser, 1996, pp. 41-43

[3] M. Tim Jones, "AI and Games", in *Artificial Intelligence, A Systems Approach*, Hingham: Infinity Science Press, 2009, pp. 101-106

[4] Claude E. Shanon, "Programming a Computer for Playing Chess," *Philosophical Magazine*, vol. 41, pp. 256-275, 1949

[5] Feng-Hsiung Hsu IBM Thomas J. Watson Res. Center, "IBM's Deep Blue Chess grandmaster chips," *Micro, IEEE*, vol. 19, pp. 70-81, 1999

[6] Şükrü Ozan, Şevket Gümüştekin, "A Case Study on Logging Visual Activities: Chess Game", in *Artificial Intelligence and Neural Networks*, Berlin: Springer , 2006, pp. 1-10

[7] Emir Sokic, Melita Ahic-Djokic, "Simple Computer Vision System for Chess Playing Robot Manipulator as a Project-based Learning Example", in *IEEE Int. Symp. on Signal Proc. and Infor. Tech.*, Sarajevo, 2008, pp. 75-79

[8] 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", in *in Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, 2011, pp. 4291 -4297

[9] M. Piskorec, N. Antulov-Fantulin, J. Curic, O. Dragoljevic, V. Ivanac, L. Karlovic , "Computer vision system for the chess game reconstruction", in *Proc. of the 34th International Convention (MIPRO)*, Opatija, 2001, pp. 870-876

[10] J. Yubo, D. Yuntao, W. Dianjun, X. Long, L. Zhanmin and W. Wei, "Pieces Identification in the Chess System of Dual-Robot Coordination Based on Vision," *Int. Conf. Web Information Systems and Mining (WISM)*, vol. 2, pp. 248-251, 2010

[11] Shao-Yi Chien, Shyh-Yih Ma ; Liang-Gee Chen , "Efficient moving object segmentation algorithm using background registration technique," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 12, pp. 577-586, 2002

[12] Bernd Jähne , "Image Formation", in *Digital Image Processing*, Heidelberg: Springer, 2005, pp. 201, 487

[13] Sebastian Montabone, "Distortion Correction", in *Beginning Digital Image Processing Using Free Tools for Photographs*, Apress, 2010, pp. 185-204

[14] Yeong-Taeg Kim, "Contrast enhancement using brightness preserving bi-histogram equalization," *IEEE Trans. Consum. Electron.*, vol. 43, pp. 1-8, 1997

[15] Y. Wang, Q. Chen and B. Zhang, "Image enhancement based on equal area dualistic sub-image histogram equalization method," *IEEE Trans. Consum. Electron.*, vol. 43, pp. 68-75, 1999

[16] John F. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, pp. 679-698, 1986

[17] Rafael C. Gonzalez, Richard E. Woods, "Morphological Operations", in *Digital Image Processing*, Prentice Hall, 2001, pp. 523-525

[18] Diego Rasskin-Gutman, "The Complete Metaphor: Chess and Problem Solving", in *Chess Metaphors artificial Intelligence And The Human Mind*, MIT Press, 2009, pp. 85-89

[19] Barbara J. Huberman, "A Program to Play Chess End Games", *Stanford University, California*, Tech. Rep CS-106, 1968

[20] Staunton Howard, "Endings of Games", in *The Chess-Player's Handbook A Popular And Scientific Introductory To The Game of Chess*, London: Harrison And Sons, 1847, pp. 500-503

[21] FIDE, "Algebraic Chess Notation", Feb 12. 2016, [Online]. Available: https://en.wikipedia.org/wiki/Algebraic_notation_(chess)

[22] Rajendra Adhikari, "Interfacing MATLAB with Chess Engine", Feb. 12 2016, [Online]. Available: https://sites.google.com/site/therajendraadhikari/publications/chess-playing-robot/interfacing-matlab-with-chess-engine

[23] Guo-Shing Huang, Chiou-Kou Tung, Hsiung-Cheng Lin and Shun-Hui Hsiao, "Inverse kinematics analysis trajectory planning for a robot arm", in *IEEE 8th Asian Control Conference (ASCC)*, , 2011, pp. 965-970

[24] Wong Guan Hao, Yap Yee Leck and Lim Chot Hun , "6-DOF PC-Based Robotic Arm(PC-ROBOARM) With Efficient Trajectory Planning And Speed Control", in *International Conference On Mechatronics(ICOM)*, , 2014, pp. 1-7

[25] T. COUR, R. LAURANSON and M. VACHETTE , "Autonomous Chess-playing Robot", February 02 2016, [Online]. Available: http://www.timotheecour.com/papers/ChessAutonomousRobot.pdf

[26] Patrick McCabe, "Chess Robot V2", February 02 2016, [Online]. Available: http://patrickmccabemakes.com/hardware/Chess_Robot_V2/

[27] R. Brooks, Massachusetts Institute of Technology, "A robust layered control system for a mobile robot," *IEEE Journal on Robotics and Automation*, vol. 2, pp. 14-23, 2003

[28] Microchip Incorporation , "Datasheet PIC-16F877a", Feb. 08 2016, [Online]. Available: http://mech.vub.ac.be/teaching/info/mechatronica/PIC16F87XA.pdf