Chess Competition with an Industrial Robot

Aleksandrs Pridoroznijs[1], Zigurds Markovics[2], [1-2] *Riga Technical University*

***Abstract* – This article describes an industrial robot application in a chess competition with a human. The article provides information on the aims, objectives and achievements. The system consists of three units – the technical vision, chess logic and robot control. Basic methods and techniques applied to the design and implementation stages, as well as the error prevention techniques and pre-match calibration procedures are described in this paper**.

***Keywords* – Chess, industrial robot, robot control, technical vision.**

I. Introduction

At Riga Technical University, the Faculty of Computer Science and Information Technology (FCSIT), Institute of Computer Control, Automation and Computer Engineering there is an industrial robot ABBIRB1600 (Fig. 1) available for the implementation of new ideas in computer science. Although the given robot model is built as the welding task automation tool, both teachers and students transformed it into a multi-purpose device that can be applied to robot control, adaptation and technical vision tasks.

Fig. 1. ABB industrial robot IRB1600 with a welding tool attached.

One of the robot applications is the two party board games with the human opponent. These kinds of solutions require technical vision, adaptation and control task solving technology interaction. These kinds of robot applications are appropriate and have been successfully used in bringing up students' knowledge and experience, as well as practical skills throughout research, analysis, solution design and implementation phases.

These kinds of robot applications require the number of criteria that must be satisfied:

1. to perceive the state of the game field and analyse it;
2. to recognize human's moves without additional interaction (no move descriptions are input from the keyboard);
3. to produce the effective move;
4. to transmit the selected move into control signals for autonomous execution;
5. to operate in a strictly defined area without touching the opponent and game pieces.

The aim of these applications is to create a system capable of providing people with the real world game experience against the industrial robot without manual move description input.

II. Applied Technologies and Tools

Chess competition with industrial robots is used as the task of combining the implementation of the technical vision, artificial intelligence and robot control techniques all together. Each of these application areas is divided into individual units that are combined together producing the final solution.

As the base for the technical vision unit OpenCV [10], the open-source computer vision library developed by Inteland and its wrapper for .Net programming environment – EMGUCV – are used. The library allows capturing a video stream from a web-camera connected to the robot and apply both global and small area image modifications [3]. This improves the overall image quality, image characteristic identification and conversion into non-graphical description,

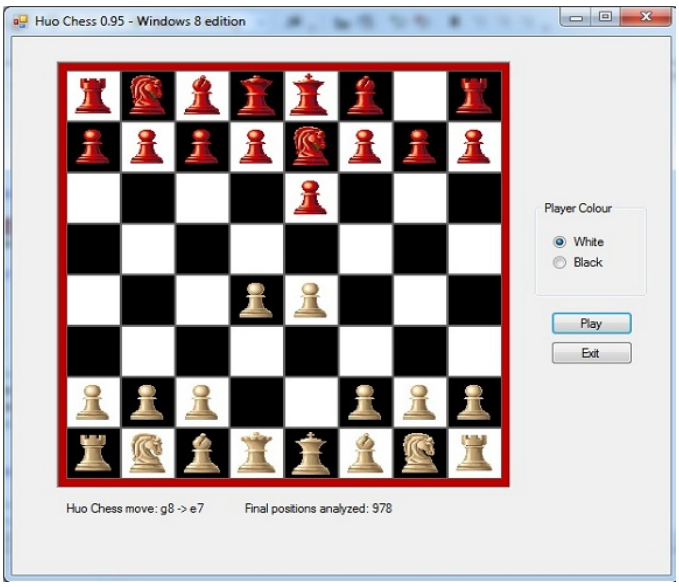

Fig. 2. Huo Chess interface.

which is further used for the robot control signal generation. It is also used for automatic game field recognition and calibration tasks, which minimize human intervention in the robot's field analysis process.

Artificial Intelligence unit includes chess rule description and decision logics. Open source chess engine Huo Chess [9] (Fig. 2) is used for this application. It utilizes the Minimax algorithm with a depth of 8 steps in the game decision tree, i.e., four its and four opponent's next moves. Engine includes possibility to play both with white and black game pieces.

The robot control unit includes the area definition and control signal generation, focusing on autonomous robot move execution. The given control unit, according to the central unit functionality, allows the robot to play both types of game pieces. It is designed for both normal and hit procedures. This unit is created in ABB Robot Studio development environment using RAPID programming language.

III. Solution Structure

The solution consists of three units – technical vision, chess logic and robot control. Central unit is based on a chosen chess engine. During the analysis stage, the chess engine operational data types used for move calculation and decision-making processes were identified. All other units were designed based on the central node properties in order to be compatible.

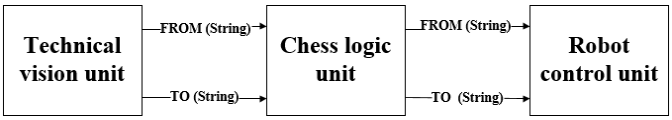

Fig. 3. Solution structure.

Fig. 3 shows that the human's move parameters should be passed to the chess engine input as two string variables, where the first variable determines FROM which field the human's move was executed, and the second, accordingly, TO the field (e.g. FROM "B2" TO "B4"). The same rules apply to the chess engine output resulting in the robot's move description as the FROM-TO expression.

Technical vision unit includes a primary image quality enhancement and transmission to the non-graphic description as a FROM-TO expression. This block implements an automatic chequer board calibration, as well as human's move recognition by using only the video stream from the camera, without any description input from the keyboard.

The robot control unit includes the chess engine output conversion into the robot's control system understandable form, the definition of the performance area and robotic control signal generation, as well as autonomous move execution.

IV. Technical Vision Unit

Based on the analysis phase, the technical vision unit defines the following tasks:

1. to automatically find and calibrate the game field;
2. to treat each of 64 fields as separate areas;
3. to recognize human made moves;
4. to output data in "FROM" and "TO" string type variables.

A. Automatic Game Field Calibration

Steps:

- automatic detection and identification of all chequer square intersections;
- square field sequential indexing;
- real-time display of recognition result;
- perspective warp and rectification;
- calculating the centre points of square fields for the robot control unit;
- dividing the warped image into 64 separate areas.

Description:

OpenCV library offers the "FindCessboardCorners()" [4] procedure, which is designed for the camera calibration tasks – a rectangular chessboard type calibration map is placed before a camera, the procedure finds all intersections in the area and returns the coordinates of each intersection as X and Y values. These values are stored and subsequently used in the description of the spherical deformation of the image, which are related to the lens structure imperfections. This provides possibility to correct spherical distortion in the next step. The result is a rectificated image.

In the process of the chess competition with the industrial robot, a given procedure is used as a tool for solving the task that is outside the defined range of this procedure – to find all the square field intersections and calculate their centre coordinates. Each square field is enumerated with its own sequential index (Fig. 4):

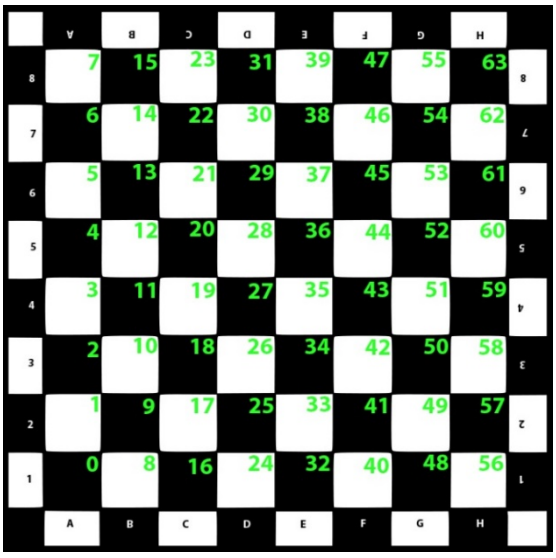

Fig. 4 . Sequentially indexed square fields.

As a next step it is necessary to represent the order, in which the indices are distributed, as in the starting element can be automatically placed in any of the four corners of the game field. This problem is solved by

"cvDrawChessboardCorners()" [5] built-in function, which connects all intersection coordinates with the coloured line (Fig. 5). The starting point is always marked with a red colour. In this way one can easily identify the current indexed sequence and correct it in real time by adjusting the camera or game field position.

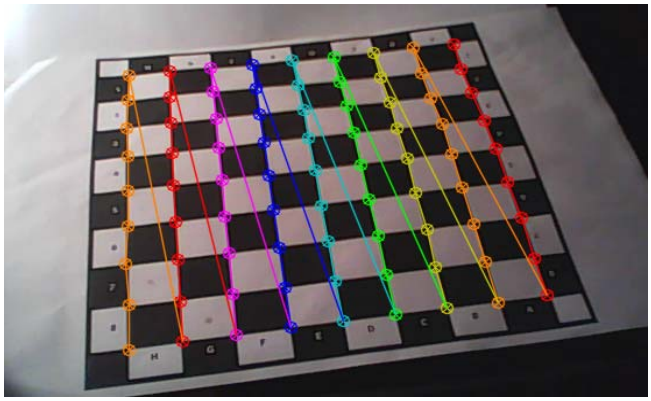

Fig. 5. The real time visual representation of the found intersections.

The obtained coordinates from the found intersections are passed to "GetPerspectiveTransform()" [8] method that generates 3x3matrixwhich describes the perspective transformation parameters and it is passed to the "WarpPerspective()" [8] method, which produces the image warp and rectification (Fig. 6).

Fig. 6.Calibration procedure result – warped chessboard.

As a final step, the resulting image is divided into the 64 separate processing areas of the same size with the help of OpenCVROI (Region of Interest) method [10]. Each area corresponds to a square field.

B. Human's move recognition

Steps:

- primary enhancements of the overall quality of the image;
- stationary state difference method known as background cut out;
- noise and error reduction;
- independent processing of each of the 64 areas;
- non-graphical description acquisition;
- passing of human made move parameters to the central unit – a chess engine.

Description:

Human's move detection task is divided into three stages – primary image preparation, recognition, and the non-graphical description acquisition.

The first stage begins with the choice of the main method that is to be used as a basis for further manipulations. Considering the central block functional properties, the selected method is able to exhibit areas, where any changes took place. These changes describe a human's move. The next factors in selecting the base method are the game board and playing piece properties – they are of the various shapes and heights, which can cause overlapping in video stream images, thus complicating their cognition procedure.

Taking into account these properties, the background cut out method was selected. It is widely used in video surveillance applications [5]. The operating principle is based on a static reference image, which is taken when the frame does not contain any moving object, and its comparison with each of the next captured frame using mathematical subtraction of the pixel values. If there are no changes, the subtraction returns the pixel value of 0, which corresponds to black colour. Thus, if something new appears in the frame, then the value differs from zero, forming two bright spots that indicate areas that participated in the human's move (Fig. 7).

This method is adopted for the chess competition task by enrolling both reference and comparison shots in a cycle after the robot's move and the human's move, respectively. The resulting picture consists of a black background and two objects, which represent the human's move. Since the pixel value of the mathematical subtraction process may be negative the absolute value is calculated.

Binarization procedure takes place in the following step, where black background remains black, but the object is filled white. Binarization sensitivity can be adjusted in the user interface.

This is followed by image enhancement procedures – mathematical morphology that separates converged objects and reduces the possibility of errors. The use of the Gaussian blur filter allows eliminating noise. Consequently, the image processing and preparation phases are accomplished.

During the recognition phase, objects' contours are obtained by using "FindContours()" [8] method. For fault detection reduction purposes, the outline dimensions of the contour are calculated and filtered by size. These values are

adjusted in the user interface before the start of each game to reduce influence of the different lighting levels. This way, objects are derived from the noise (Fig. 7).

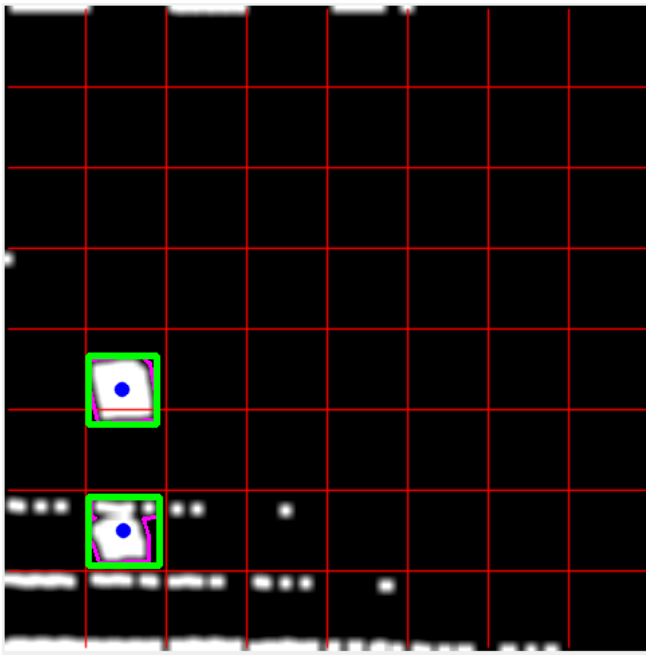

Fig. 7. Move recognition error prevention demonstration (noise is produced on purpose).

As the chess playing pieces are not flat and the camera is placed with the angle to the game field, the objects produced as a result of the move event can occupy more than one of the 64 areas simultaneously. This requires reducing the outcome to the single area. It is achieved by finding the object centre coordinates and outlining them in the resultant image with a blue colour (Fig. 7). This image gets binarized once again, but this time by the blue colour. This results in two new central objects. The obtained centre object is reduced to the size of one pixel, allowing it to be associated with only one of the 64 areas.

During the last recognition phase, each of the 64 areas is treated separately. Areas are screened for the presence of a reduced central pixel. Once a pixel is found, it brings the "flag" up for the area index, suggesting that it participated in the human's move. This action initiates non-graphic information acquisition procedure.

During the final stage, the data about the human's move involved areas are converted in a suitable format for chess engine input data (FROM-TO). To achieve this, the new data structure is introduced. It includes a description of each individual field – the coordinates of the centre in the video stream images, warped image field centre coordinates, field names, a flag that identifies if a field is occupied or free, and a flag that identifies whether or not it is occupied by the human game piece. Each indexed element of the structure corresponds to each indexed game field element (Fig. 4). This structure is propagated with initial values before the start of every game during the calibration phase. The given initialization procedure provides a possibility for the human to play both black and white pieces.

The comparison of the structure indexed element description provides the corresponding fields for FROM and TO variables, as well as the hit event. Element property values are overwritten with regard to the current situation of the game field.

The identified element field names coincide with the central block input data format. Technical vision unit completes its operation by transferring the data to the central unit. Then the chess logic unit produces a decision and passes new data to the robot control unit for the control signal generation and autonomous move execution.

V. Chess Logic Unit

Huo Chess [9], the open source project, is used as the chess logic unit. It utilizes the Minimax algorithm for the decision making and next move selection processes.

Minimax algorithm with the limited depth consists of the following steps (Fig. 8):

1. game tree generation, where each node corresponds to a situation on the game field;
2. when reaching a defined depth, the tree is split into MIN and MAX levels where MAX corresponds to the robot possible moves;
3. dead-end nodes are propagated with heuristic values that describe the quality of an outcome;
4. heuristic values are transferred up to a higher level. If there are more than one value to be moved to a higher level, depending on if the MIN or MAX level is engaged, respectively, the minimum or maximum value of heuristic measure is transferred up.

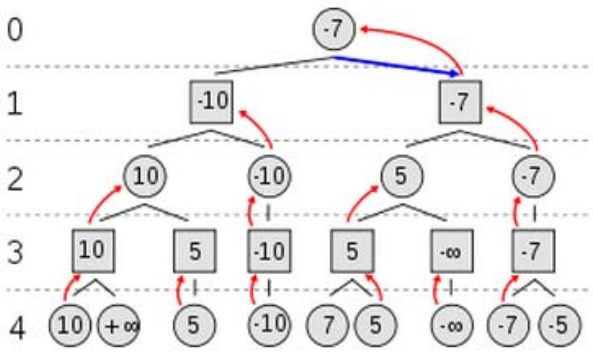

Fig. 8.Representation of a Minimax tree.

The selected chess engine utilizes the depth of 8 steps, which means that it calculates all possible outcomes for both own and human's next four moves.

As the game continues, the tree depth remains constant, but the width grows drastically. This means that each next move increases the count of the nodes involved in the decision process, making this process significantly longer.

This is a reason why in the future it is planned to replace the chess engine with a more efficient one.

VI. Robot Control Unit

The mission of robot control unit is to receive data from a central unit and convert into control signals. In addition, care has to be taken for human opponent and game pieces to remain intact. The control unit executes the robot move and gives control back to the technical vision unit for the next human's move recognition cycle.

Steps:

- to get the data from the chess logic unit;
- to convert received data into movement offsets;
- to define the robot movement working area restrictions;
- to define the robot authorized movement schemes;
- to generate control signals;
- to play both black and white pieces;
- to perform both regular and hit moves.

Description:

Central unit returns a string description of the move in a FROM-TO form. This data is compared with the structure of the indexed element names, which also identifies those fields from which and to which the robot will execute the move.

In order to reconcile the robot and game field coordinates, the reference point is defined at the chess board bottom right-hand corner (the human's side). The real world and warped image field dimensions are well known. Thus, the coefficient K is calculated.

$$K = \frac{A}{B} \qquad (1)$$

where K – the resulting conversion factor;

A – the real world field size in mm;

B – the virtual warped image field size in pixels, pix.

The distance is calculated between the FROM and TO virtual field centres in X and Y Cartesian coordinate directions. Both values are summed with the distance from the defined reference point. The outcome is multiplied by the conversion factor K, resulting in the robot control signal offset values in the horizontal plane.

Robot's movement area from which it is forbidden to leave, as well as move execution schemes are defined for sake of human intactness and game piece properties. (Fig. 9).

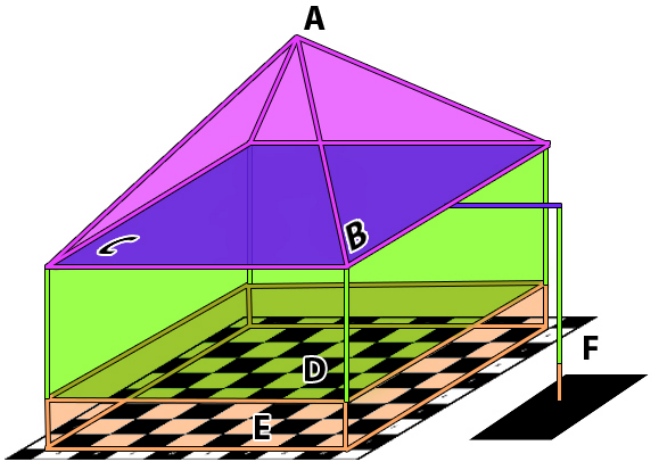

Fig. 9. Defined robot movement area.

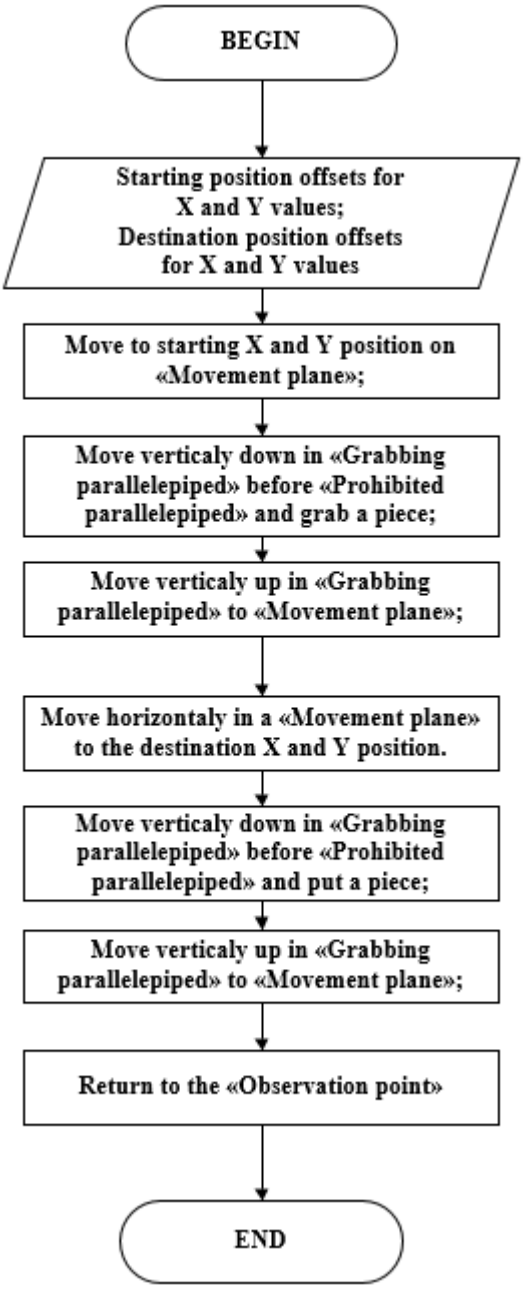

Fig. 10. The robot regular move execution scheme.

The following terms are defined in the robot's movement area:

A – "Observation point";

B – "Return pyramid" (Violet colour);

C – "Movement plane" (Blue colour);

D – "Grabbing parallelepiped" (Green colour);

E – "Prohibited parallelepiped" (Orange colour);

F – "Hit tray" (Black colour).

where:

- "Observation point" A – a point, which is defined as the robot's initial state, from which a video stream is taken for the computer vision unit;
- "Return pyramid" B – within this pyramid only linear motions are allowed to return to the "observation point" or to take a position in the "movement plane" above the game field;
- "Movement plane" C – the plane in which the robot is allowed only to perform horizontal linear hoover from one to another game field. While performing tasks in this plane no game pieces can be accidentally touched;
- "Grabbing parallelepiped" D – a parallelepiped within the robot is allowed to make only vertical linear motion, to perform the game piece grabbing or placement procedures;
- "Prohibited parallelepiped" E – the robot is forbidden to enter this parallelepiped;
- "Hit tray" F – a defined point where the robot drops opponent's pieces.

Depending on the central block produced data, the robot control unit calls one of two procedures:

- regular move (Fig. 10);
- hit move.

This provides the complete chess logic functionality associated with the robot move execution.

VII. Additional Units

Real-life chess competition requires additional units, which interact with a human player:

1. before the game the calibration;
2. human's move execution;
3. wrong move acknowledgement and repeated execution.

The first procedure is related to the technical vision unit and the general preparation tasks:

- connecting to the robot controller;
- game piece colour selection;
- game board placement before the camera and running the technical vision calibration procedure;
- game piece placement on the board;
- binarization and contour size threshold adjustments;
- starting the game.

In order to ease the competition process, the chess clock in imitation is introduced. By using it a human announces his/her move to be accomplished. The given imitation participates in the wrong move repeated execution:

- the wrong move notification is displayed;
- a human places his/her piece to the previous position;
- acknowledgement that the mistake is corrected by pressing a chess clock imitation button;
- Making his/her move.

VIII. User Interface

The user interface includes a number of control units, as well as video stream display windows. In addition, Huo Chess game field representation is included (Fig. 11).

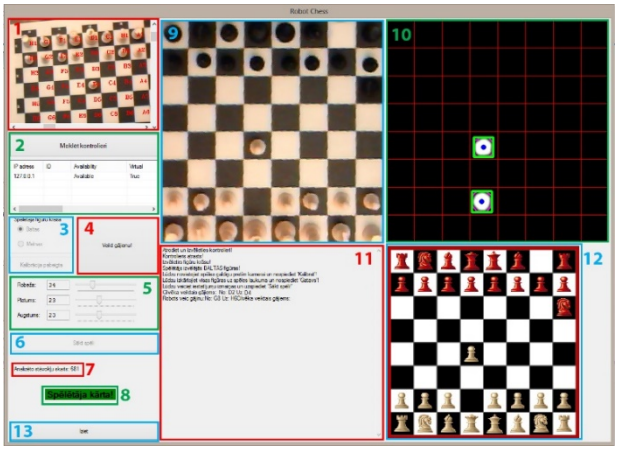

Fig. 11. User interface.

The application interface consists of the following elements:

1. video streams from a web camera – they help to verify the correct placement of the chessboard and display recognized field intersections. After the calibration process is completed, field names are displayed;
2. connecting to the robot controller – the panel consists of a network scan button and the found controller's list;
3. piece colour selection panel – a human selects piece colour by selecting the corresponding menu item;
4. multifunction control button – depending on the state of the game it can display the following values:
   - "Confirm" – it acknowledges the wrong move;
   - "Execute move" – chess clock imitation, a human confirms his move;
   - "Ready" – during the calibration process it acknowledges that all game pieces are arranged on the playfield;
5. computer vision value adjustments for binarization and contour size thresholds;
6. start button – calibration process termination of this button linking field and game initiation;
7. the chess engine statistics – it is used for informational purposes of how many nodes participated in the decision making process;
8. game status indicator – depending on the state of the game it can contain the following values:
   - "Calibration process" (yellow) – it indicates that the calibration process is in progress;
   - "Processing …" (blue) – it indicates the robot's turn to execute a move;
   - "Player's turn" (green) – it indicates that it is player's turn to make a move;
   - "Illegal move" (red) – it indicates that an illegal move was made and the application is awaiting for acknowledgement from the user;

9. warped board video stream – it displays the video stream after the board calibration procedure;
10. recognized move display window – it visually displays move recognition results;
11. log window – it outputs messages and hints for the calibration procedure and records user's activities. During the game it displays both human's and robot's made moves;
12. Huo Chess game field representation – the standard representation of the game field, which is supplied with the chess engine;
13. exit button – the program can be terminated at any time and closed by pressing this button.

IX. Conclusions

The robot is able to recognize human's moves and to autonomously oppose him/her. It is able to play both white and black pieces.

The selected solution structure makes it a flexible-central unit that can be replaced by another chess or even checkers engine. The solution is not associated with either piece form or their colour.

Solution interface is intuitive and user error safe.

It has been experimentally found that natural light with variable intensity can produce significant amount of noise, which can sometimes lead to recognition errors. In addition, the solution should not be used under daylight lamps, since they create a ripple noise effect in the video stream output that causes recognition errors.

Solution testing phase defined improvements for the pleasurable feel of the game:

- to replace the chess engine in order to increase the processing speed – due to the game tree growth, the Minimax algorithm is executed quite slowly;
- to modify the hit procedure – a newly hit piece can sometimes be placed right above the previous hit piece;
- to make a solution less dependent on the lighting quality.

References

[1] T. Cour, R. Lauranson, Autonomous Chess-playing Robot, - EcolePolytechnique, 2002 p. 27.
[2] S. Blunsden, Chess Recognision, - Bsc (Hons) Computing and Informatics, 2003 p. 110.
[3] L. G. Shapiro, G. C. Stockman, Computer vision, - Prentice Hall, 2001 p. 755.
[4] G. Bradski, A. Kaehler, Learning OpenCV, - O'REILLY, 2008 p. 571.
[5] R. Laganiere, OpenCV 2 Computer Vision Application Programming Cookbook, - PACKT Publishing, 2011 p. 298.
[6] J. A. Coens, Taking TekkotsuOut of The Plane, - School of computer Science at Carnagie Mellon University, Pittsburg 2010 p. 49.
[7] K. Kovaļovs, Dambretes spēles realizācija ar industriālo robotu, RTU 2012.
[8] EMGU CV documentation [Online] Available: http://www.emgu.com/wiki/files/2.4.0/document/Index.html.
[9] Huo Chess documentation [Online] Available: http://www.codeproject.com/Articles/20736/C-C-CLI-Micro-Chess-Huo-Chess.
[10] OpenCVdocumentation [Online] Available: http://docs.opencv.org/.

Aleksandrs Pridoroznijs, B.sc.ing., 2013.
He is a Master's degree student at Riga Technical University, the Faculty of Computer Science and Information Technology (FCSIT).
Research interests: technical vision, robot control systems.
E-mail: aleksandrs.pridoroznijs@gmail.com

Zigurds Markovics holds Dr.habil. sc.ing. degree, Professor (since 1993) at Riga Technical University, the Faculty of Computer Science and Information Technologies, Institute of Computer Control, Engineering and Technology.
He is the author of more than 150 publications.
Research interests: computer control system, artificial intelligence system, robotics.
He is a Member of the Latvian Society of Professors and the Latvian Association of Scientists.
Address: Meza Str. 1/4, LV – 1007, Riga, Latvia.
E-mail: Zigurds.Markovics@rtu.lv

Aleksandrs Pridorožnijs, ZigurdsMarkovičs. Šaha spēle ar industriālo robotu
Šajā rakstā ir aprakstīta industriālā robota šaha spēle ar cilvēku. Rakstā sniegta informācija par mērķiem, uzdevumiem un sasniegumiem. Spēles vadības sistēma sastāv no trim blokiem – tehniskās redzes, šaha loģikas un robota vadības. Tiek aprakstītas šo bloku pamatfunkcijas, metodes un paņēmieni, kas pielietoti projektēšanas un realizācijas posmos. Aprakstīti arī kļūdu novēršanas paņēmieni un pirms spēles kalibrēšanas procedūras.
Piedāvātais risinājums ļauj automātiski atpazīt un kalibrēt šaha spēles laukumu, atpazīt cilvēka veiktos gājienus, izmantojot *Huo Chess* šaha dzini – veikt nākošā gājiena izvēli, ģenerēt vadības signālus un vadīt robota gājienu izpildi. Izmantotās metodes ir pielāgotas spēles laukumu un figūru īpašībām. Piedāvātās metodes pozitīvā īpašība ir tāda, ka tā nav piesaistīta nedz spēles figūru krāsām, nedz formai, kas nozīmē, ka risinājums neprasa specifisku inventāru. Metodi var izmantot citu analoģisku uzdevumu risināšanai, piem., dambretes spēlei.

Александр Придорожный, Зигурд Маркович. Игра в шахматы против индустриального робота
В данной статье описывается применение промышленного робота в игре в шахматы с человеком в роли оппонента. В статье представлена информация о целях, задачах и достижениях. Система управления состоит из трех основных блоков - технического зрения, шахматной логики и управления роботом. В данной статье описаны основные методы и технологии, применявшиеся на этапах разработки и реализации проекта. Так же описаны процедуры избежания ошибок и калибровки.
Предлагаемое решение позволяет автоматически распознавать и калибровать изображение шахматной доски, распознавать ход, сделаный человеком, используя код HuoChess, выбрать следующий ход и формировать сигналы управления для исполнения роботом собственного хода. Используемые методы формированы при учёте особенностей игрового поля и формы фигур. Положительной особенностью предлагаемого метода является то, что он не привязан ни к цвету, ни к форме фигур. Это означает, что решение не требует специального оборудования. Метод может быть использован для выполнения и других аналогичных задач, таких, как игра в шашки.