# Simple computer vision system for chess playing robot manipulator as a Project-based learning example

Emir Sokic, Melita Ahic-Djokic
Faculty of Electrical Engineering, University of Sarajevo
Zmaja od Bosne bb, 71000 Sarajevo, Bosnia and Herzegovina
esokic@etf.unsa.ba, amelita@etf.unsa.ba

***Abstract*** *— This paper presents an example of project-based learning (PBL) in an undergraduate course on Image processing. The design of a simple, low-cost computer vision system for implementation on a chess-playing capable robot is discussed. The system is based on a standard CCD camera and a personal computer. This project is a good tool for learning most of the course material that would otherwise be mastered by homework problems and study before an exam.*

*An algorithm which detects chess moves is proposed. It compares two or more frames captured before, during and after a played chess move, and finds differences between them, which are used to define a played chess move. Further image processing is required to eliminate false readings, recognize direction of chess moves, end eliminate image distortion. Many Image processing problems and solutions can be introduced to students, through the proposed algorithm.*

*The results are encouraging - students without any previous knowledge in image processing and advanced topics, such as artificial intelligence (neural networks etc.), may attain a chess move recognition success rate greater than 95%, in controlled light environments.*

***Keywords*** *— image processing, project-based learning, robot manipulator, chess moves recognition*

## I. Introduction

Project-based Learning (PBL) is a teaching approach that is gaining increasing interest within the engineering educational community [1]. In contrast with traditional learning methods, where student course motivation is based on homework and test results, PBL tries to spread the learning process throughout the course and to shift away from learning for the sake of the exam to one of learning in order to use the knowledge [2]. The ability to solve problems and think critically and proactively is more valued in engineering than the ability to remember facts. This is very difficult to attain through the traditional approach, which involves giving students knowledge and engineering tools throughout the term, and then evaluating their mastery of the subject at finite times by way of examinations [3]. This environment usually leads students to focus on ”Whats going to be on the exam?” rather than more important questions, such as What is this new knowledge, how do I use it, and what can it do for me in the future?”.

The basic approach that the authors take in project-based learning is to present the students very early on in the course with a set of tasks that must be accomplished. The tasks are presented in the form of an engineering project that embodies the specific learning objectives of the course. Students must seek out and master the course material that enables them to complete the projects. In a traditional course, this same course material is learned through homework problems and study prior to an exam. Evaluation of learning is based on project solutions, the method and presentation of solutions, and possibly a final examination [2].

This paper proposes an adequate example of Project-based learning intended to be part of an Image processing undergraduate course. The example is interesting because it incorporates most of the topics discussed in the course: image representation and display, image histograms, contrast and brightness enhancement, histograms equalization, color transformations, image resizing, interpolation, affine and higher-order spatial transformations, edge detection, region of interest processing, filtering, noise elimination, some elements of convolution and correlation etc.

The proposed algorithm doesn’t use any advanced recognition algorithm (for example, neural networks), so it leaves a lot of space for students with limited previous knowledge to easily learn new methods and techniques, and successfully experiment with different approaches in Image processing. The ability to choose between different algorithms and algorithm parameters develops critical thinking by students. Image processing is done via the software package MATLAB, so this project may be used in combination with a beginner’s course for MATLAB as well.

An example of a computer vision system for a chess playing robot manipulator is proposed in a similar manner in the work of Uyar et al. [4], but in this paper we will use slightly different approaches, especially in the domain of image calibration and false reading elimination.

The paper is organized as follows. In Section 2 we define the computer vision system, its main components, and discuss its usability as a project-based learning example in an Image processing undergraduate course. Applicability to other courses is presented in Section 3. An original solution to a simple computer vision system from the authors of this paper is proposed in Section 4. Section 4 also summarizes the main results, and ends with a conclusion and some comments on future work.

## II. Problem description

The computer vision system for implementation on a chess-capable playing robotic manipulator explained in this paper is based on processing of images acquired with a CCD camera, connected to a PC via standard USB port. The camera

is positioned as close as possible above the chess board, orthogonally to it, as shown in fig 1.

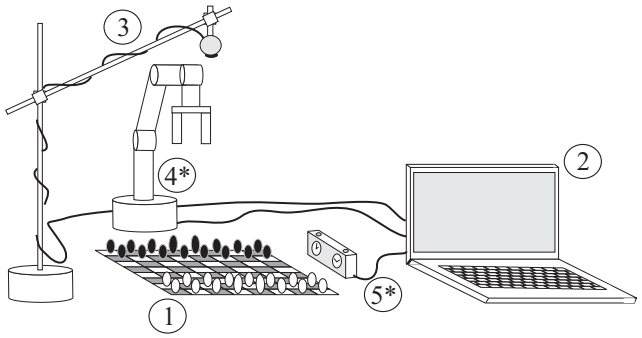



Fig. 1. Physical configuration of the described computer vision system - main parts: 1) chess board, 2) PC, 3) camera , optional parts: 4) robot manipulator, 5) chess clock as a triggering device.

The more important part of the problem is the algorithm detecting chess moves. The algorithm's inputs are frames captured at specific time points defined by a chess move. The first output of the algorithm is the numeric notation (also called "coordinate chess notation") of a chess move (for example, "a2 a4"). If a chess move is not to be played by a human, a specific parameter (Chess move descriptor - CMVD) which describes what type of move is played is required, and this creates a second algorithm output. In this paper, we assigned the following values to CMVD: 1-regular move, 2-capturing piece, 3-kingside castling , 4-queenside castling, 5-en passant. With this setting, a minimal set of information needed for planning a robot manipulator trajectory is acquired, and may be evaluated for generation of a following move.

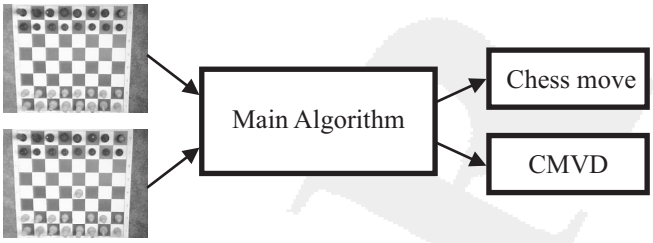



Fig. 2. Inputs and outputs of the algorithm detecting chess moves.

There are several theoretical topics in the domain of Image processing included in this project that can be presented to, and discussed with students. Some of them are described in the following text.

*Brightness and contrast adjustment*

Captured frames usually do not have equal brightness and optimal contrast, due to environmental light changes. Brightness refers to the overall lightness or darkness of the image [5]. Contrast is the difference in brightness between objects or regions. Increasing the brightness makes every pixel in the image become lighter. In comparison, increasing the contrast makes the light areas become lighter, and the dark areas become darker.

Before comparing frames, it is crucial to equalize captured frames by brightness, and enhance their contrast. Several Image processing themes might be discussed at this stage. This may be a good place to introduce students to output transform (Gamma curve), grayscale transform, and histogram equalization, even if the direct use of the latter method will not produce satisfactory results in this project.

*Colormap conversion and resolution change*

Grabbed frames (typically RGB-colormap, 640x480 pixels) contain sufficient information about a chess move, usually much more information than we actually need. For the sake of processing speed and computer memory requirements it is advisable to change the colormap and resolution. It can be easily calculated that memory requirements are changing exponentially in respect to an image resolution change. At this stage, students might be asked to find an optimal resolution and colormap, in respect to a chosen criterion (for example processing time versus maximum error rate). They should experiment with different resolutions and colormaps, and create corresponding conclusions.

*Comparation of selected frames*

This can be done by simple absolute subtraction of matrices that represent grabbed frames. We will further refer to this difference as Difference Image (DI). If we assume that frames are converted in a grayscale colormap, it is obvious that the difference in value of every pixel on DI may vary from zero (no difference) to 255 (maximum difference). It is crucial to empirically evaluate the threshold which will define the origin of the difference - is it a result of a chess piece move, or a nonevitable noise. In respect to the chosen threshold, in this paper we propose that the difference image is converted to a binary, as shown in fig. 3. This is a simple way to nonlinearly filter noise.

*Eliminating image distortion*

There are several ways that distortion of images may appear either we want a photograph of a section of a rounded object (for example, photographing a planet's surface), or we photograph a plain object which is relatively close to the camera (lens), and because of that it appears rounded. Both ways, we get an oblique view of the object, and often before using it, it should be transformed (using spatial transformations - warping). In our physical setup those problems could but must not be solved, because their elimination is not crucial for picture evaluation. In this paper we do not discuss recognition of chess pieces, but the recognition of chess moves. A more important problem is the effect of projective transformation that appeared because of the impossibility of exact centering of the camera above the playing board, as in fig. 4, c).

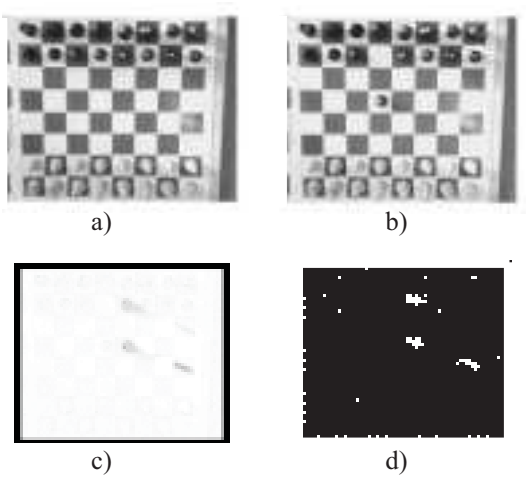



Fig. 3. a) Image captured before a chess move, b) image captured after a chess move, c) DI (inverted, for easier viewing), d) DI converted to a binary image.

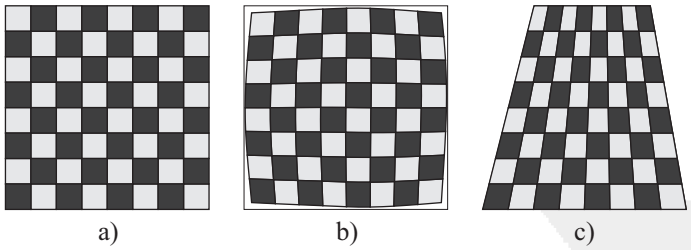



Fig. 4. a) Original image, b) image distortion that appears because of a round camera lens, c)image distortion that appears because of an inadequate camera position.

At this stage, students may be introduced to different spatial transformations (warping), and methods of interpolation, and solve problems stated above.

*Image segmentation*

After converting it to a binary image, the DI should be spatially transformed, and segmented (8x8 segments, according to chess board squares). Each segment is uniquely associated to a number that describes the probability of participation in the current chess move. Luckily, we should have two identified chess board squares whose numbers are at least ten times greater than the other 62 - it means those are the fields that participated in a chess move. The experiments showed that often on DI appears noise, as a result of a shadow, or environmental light change. The value of the noise is often comparable to the actual chess move playing squares, which makes it difficult to eliminate. Another problem is how to distinguish these results from a three and four squares chess playing moves (en-passant and castling). The only way to solve this is to directly implement searching moves pattern in proposed algorithm, as shown in fig. 5.

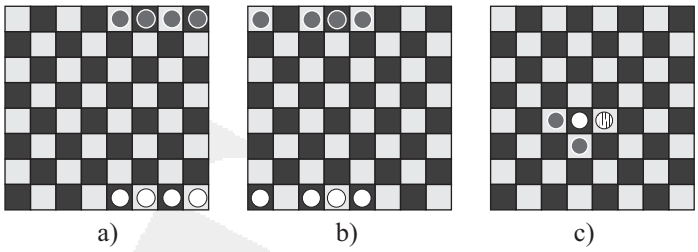



Fig. 5. Three and four squares chess moves patterns: a) kingside castlings, b) queenside castlings, c) example of en-passant.

Students' assignment in this stage should be to propose a chess moves detection algorithm, taking three and four squares moves and noise into consideration.

*Chess move direction*

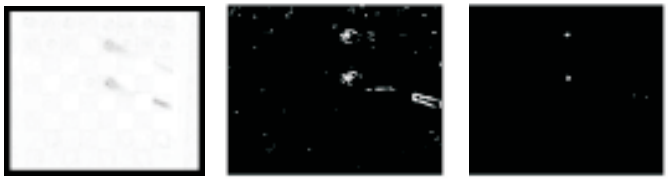
If we assume that the correct chess squares are already identified (for example: d7 and d5), it is impossible to know if the move is made from d7 to d5, or otherwise, without any additional information. In the work of Uyar et al. [4], a way to find out the direction of a chess move is proposed by comparing it to an earlier captured frame with no chess pieces on it. In this paper we used a different approach, which assumes that the initial frame is taken in initial chess pieces position, so the comparison could be made only with a different empty square - which will produce incorrect result, or by defining colors of board and pieces, upfront - hence getting some additional information.

Observation of a chess game will lead to a conclusion that the square which is left empty (in the second frame) is the one from which the chess piece moved. Using edge detection, and a low threshold, it is very easy to locate an empty square, as shown in fig. 6.

Fig. 6. An empty chess square contains no edges.

*False reading, or error, elimination*

Experiments with edge detection on several frames have shown that edge detection processed frames are less sensitive to shadow and environment light changes, which can be successfully used for reducing errors in detection, as shown in fig. 7.



a) b) c)

Fig. 7. a) DI (inverted, for easier viewing), b) edges calculated on DI (threshold parameter is 20), c) edges calculated on DI (threshold parameter is 40).

A group of students may experiment with edge parameters, and segmentation. Here a ROI (region of interest) processing may be introduced to students. The segmentation should not be same as made for the moves detection, because significant edge values will be created at the borders of chess squares (because of sudden color change), and could lead to a false conclusion.

*Frames triggering*

Frame grabber function (or frames triggering) may be software or hardware implemented. Students may also be introduced to Video processing as a consecutive frame processing, using this example. As presented in [4], timing moments for capturing frames may be defined using the difference amount between consecutive frames, which clearly identifies a chess move. Although this approach is very comfortable for a human user, it is very computer memory and processor consuming, because it requires analysing a video stream continuously. We proposed a different solution which is perhaps more appropriate for students as it incorporates some knowledge from digital systems creating a replica of a chess clock, so that user (or robot) presses it every time it finishes its move. The processing time is reduced, frames are taken only when needed, and even precise chess play time can be measured and indicated.

## III. Applicability to other courses

This project not only involves Image processing, but is also applicable to many other research areas, and as such it can be extended to other courses as well, from which we well mention a few.

First, this project should be used as a computer vision sensor system for a computer controlled robot manipulator. Minimal number of parameters and information needed for manipulator trajectory planning is available. Depending on the construction of a robot manipulator, a three DOF semiautonomous robotic manipulator should be sufficient. A similar project-based learning course on Robotics and Mechatronics may be organised, and a group of students should develop a robot manipulator. This will be explained in our future papers.

Digital systems should be included too. The proposed hardware triggering solution, together with an indication of time and chess player turn, could be developed to interact with the rest of the application. Additional simple circuitry and sensor units could be made, to improve successful chess moves detection rate.

Intermediate computer-programming course students can be involved. Chess playing engine, if not implemented directly on a chess playing robot, should be implemented, or at least an interface between an image processing software (MATLAB in this case), and some chess engine, should be developed. Interprocess communication is a very interesting topic that can be easily discussed through this example, in frames of such a course.

## IV. Proposed solution and results

*Proposed solution*

We will briefly present some of the main concepts of a solution that can be used as a model for students' work, and is mainly an original contribution of the authors of this paper.

It is possible to process grayscale colormap frames, converted from a RGB image acquired from a camera. Those images, captured in a native camera (used Logitech QuickCam Express WEB camera) resolution (352x288), are subtracted. The difference image (DI) was then converted to binary image, due to predefined threshold. Before starting a chess game, a camera calibration is made, using a graphical user interface (GUI), from which the user selects chess board corners, points that later could be used as reference for dewarping the image, as shown in fig. 8.

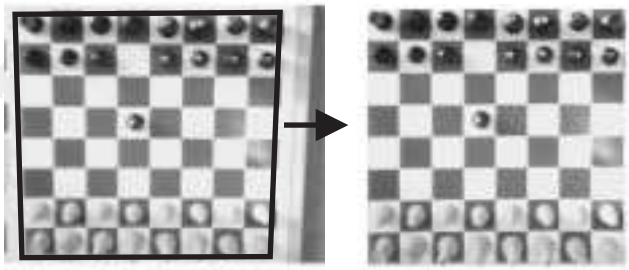


Fig. 8. Dewarping an image using a projective transformation with bilinear interpolation.

The binary image of differences is now transformed to a 240x240 resolution image, which is used as a starting point for image segmentation. Numerical "integration" over chess squares on segmented board (8x8) is now made, and chess squares with great probability of participating in a chess move are located. Numerical "integration" consists of simple calculation of an arithmetic sum of Boule variables (derived from a binary image) on a chosen chess square. The direction of the move is evaluated using edge detection with Sobel operator, as explained before.

The probability of error elimination is increased using edge detection of difference image (DI). Segmentation of the binary

image created from edged DI gives us a new 8x8 matrix which can be used to calculate the weight factors of every square on the board, in respect to improbability of noise. This is done by multiplying this new matrix element-wisely with the matrix that represent the segmentated image. Correctly chosen parameter of edge detection algorithm (threshold which decides if a change in intensity is an edge or not) could eliminate the noise almost completely, but the problem is the estimation of this parameter. This problem is solved by using a sort of adapting algorithm, which iterately changes the edge detection parameter and repeats the process automatically until only two possible chess squares are left (during that process, algorithm checks for en-passant and castling). This algorithm increases the successful chess moves recognition by 20 percent.

The solution is further improved. A hardware triggering device, a replica of a chess clock, is created. This device is able to trigger a snapshot (grab a frame) either pressing a button, or sending a command from a PC or MCU. It is based around a RS flip-flop, and it connects to a PC through a standard parallel port. It can also trigger on a time-event basis (for example, every 10 seconds), to compensate for environment light change during human/PC thinking. Even more, a chess moves presentation and generation is created using a chess engine (GNU Chess), and a communication protocol. Most of the newer chess programs communicate with each other through a Chess Communication Protocol (CCP), but we propose using an older chess communication protocol AutoRS232 because students might lack some programming knowledge required for establishing interprocess communication using CCP. AutoRS232 is a serial protocol used for communication between two chess playing computers, serially connected. By emulating a serial null-modem cable (free software package), it is possible to establish communication between a chess engine and MATLAB, using simple serial communication principles. This approach has even more advantages - connection to multiple computers is much easier, and so is a connection to a MCU.

*Results*

The results of the proposed solution were outstanding. The chess moves recognition success rate were up to 99%, in good light conditions. Lower rates (down to 75%) were mostly because of low quality camera, low brightness, and exceptionally unfavourable shadows. The results do depend on the choice of playing chess board, and even better results can be acquired with high contrast and low gloss playing boards. The chess move detection algorithm cycle conducted in MATLAB, on a notebook with Pentium 1.6GHz Centrino Processor, and 512MB RAM memory, finishes in less than half a second (for one move) with over 35 consecutive edge parameters changes and process repeats, which may be characterised as a real-time chess playing vision system.

During a term, the described Project-based learning system is conducted over a group of students, at our Faculty of Electrical Engineering, University of Sarajevo, on Signal and Systems Analysis undergraduate course. The results are encouraging - the students finished their task in supposed time, greatly motivated, and with exceptional knowledge skills - in contrast to the traditional method of learning.

## V. Conclusions and Future Work

The Project based learning system presented here for implementation on an Image processing course has successfully completed its purpose. The students have actively participated in learning process, developing active learning and self learning, and have enhanced communication skills, critical and proactive thinking, as well as learned some ”soft skills” demanded from engineering graduates (effective teaming skills, project management, ethics, engineering economics, etc.)

Even though this vision system is intended mostly for education purposes, in future work we plan to improve the computer vision system, mostly by giving some more information to an image processing computer, either by a higher quality camera, or using additional cameras, or sensors, but yet holding the simplicity and usability in the educational environment. The other interesting topic we plan to discuss is the evaluation of many empirically gained parameters, and try to set up some constraints in the choosing of parameters. The robot manipulator is to be discussed, designed and implemented together with our computer vision system.

## References


[1] E. S. Hadim H.A, “Enhancing the engineering curriculum through project-based learning,” in *Frontiers in Education*, 2002.

[2] H. R. Clark W.W., “An example of project-based learning using a laboratory gantry crane,” in *Frontiers in Education*, 1997.

[3] e. a. Golob M., Tovornik B., “Project based learning using a magnetic levitation control system,” in *Southeastern Europe, USA, Japan and European Community Workshop on Research and Education in Control and Signal Processing, REDISCOVER*, 2004.

[4] e. a. Uyar E., Gumustekin S., “A computer controlled vision oriented robot manipulator for chess game,” in *Southeastern Europe, USA, Japan and European Community Workshop on Research and Education in Control and Signal Processing, REDISCOVER*, 2004.

[5] S. Smith, *The scientist and engineer’s guide to DSP*. California Technical Publishing, 1997.