Chess State Detection

Bhavani B S
Department of Electrical Engineering
Stanford University
Stanford, CA 94305
Email: bhavani@stanford.edu

***Abstract*—Chess detection algorithm is an implementation capable of recognizing chess board, locate the squares and detect chess pieces from an image. This algorithm will work for a couple of viewing angles between 30 to 60 degrees. This is constructed using image processing techniques. Firstly the chess board is segmented from the input image, and then Hough's transform is used to detect crosses. Next, each square, with some vicinity around it, is extracted, parsed and checked to see if it contains chess piece. Then in each of the segments, pieces in the vicinity margin are removed, leaving only the chess piece of the main square. The test piece is oriented vertically and compared with the pre-defined training set. The area score is later calculated by taking difference of scaled training pieces and the test piece. The one resulting in the lowest score is the best matching piece.**

Keywords—chess detection; chess piece recognition

I. Introduction

Chess is one of the most intellectually challenging games. People have been playing this since 6th century. This makes people smarter, build critical thinking and prediction skills. However, it is also a complex game for the masses with difficulty even in following the rules of the game. Thus, the algorithm explained in the report is a technical solution for a new person to not only get to know the basics of the game, but also to predict the future moves for a given state of a chess piece.

This algorithm can also be used in robots to play chess [1]. Most of the existing chess playing robots are designed to play for a given viewpoint. These robots can be made more robust by making it work for different viewing angles, and with further improvement, this algorithm can be made independent of color of chess pieces. These robots can be even contested on the basis of chess engine.

II. Image processing

A. *Detection of squares in the chess board*

First, the chess board is extracted from the image. This helps in detecting the grid of cross lines more accurately and filter out the lines that don't correspond to these cross lines.

To generate the mask to extract out the chess board, input image is binarized using Otsu's method, then detect edges using Canny's edge detection followed by Hough's line detector. Each line is then processed to obtain left, right, top and bottom mask. Left mask is a binary image that has value '1' for all positions that lie to the right (below) of every line having positive slope. Likewise, right mask will have all '1s' to the positions lying to the left side (above) of all lines with positive slope. Top mask will have all '1s' below (left) the lines having negative slope and bottom mask will have all '1s' above (right) lines having negative slope. Performing AND operation on these 4 masks will get the mask of the chess board. Close filtering is performed to remove lines on the mask which don't correspond to the chess board edges. Using this mask the chess board is extracted (Figure 1).

To the extracted chess board Otsu's binarization, Canny's edge detection and Hough's line detection is applied once again. Now the noisy lines will not be detected (Figure 1). Morphological operations like close filter are reused to extract out the regions that correspond to 64 squares of the chess board (Figure 1).

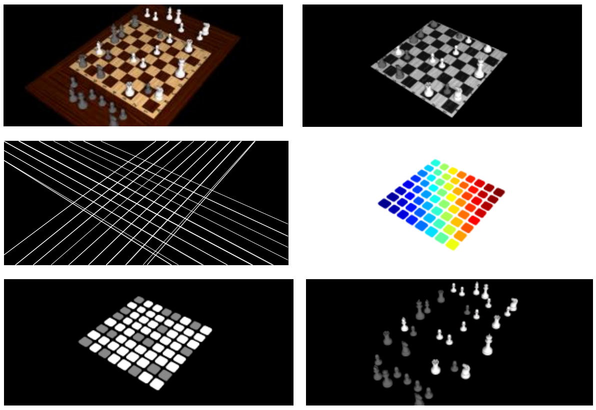


Fig. 1. Top row (left to right) input image; segmented chess board; second row (left to right) Hough's transform on the segmented board after applying Otsu's binarization and Canny's edge detector; Labelled squares of the chess board; Bottom row (left to right) squares showing detected pieces; Extracted pieces of the input image by using a mask obtained by subtracting two color channels

B. *Detection of pieces on the chess board*

First, squares containing pieces need to be detected. The input images used in this scenario have pieces with shades of gray on a board with color and texture. A useful property of gray image is that all the three color components have same value. On subtracting two color channels, the result will have value 0 at all locations where pieces are found and a non-zero value to other [1]. By implementing binarization and inversion we will get a mask which can be used to extract out the pieces on the chess board (Figure 1). In case of images taken at varying light conditions, MAP detector will be useful to extract the pieces. For only gray images, line detection on small segment on images can be done and remove the cross lines. There after basic image processing techniques can be used to refine the result.

To check if there is a piece in the region, area occupied by the suspected piece in the region is noted. With sufficiently good area detected, presence of piece can be concluded (Figure 1). Once the piece is present, the color of the piece is noted to check if the piece is black or white.

Every square, containing a piece, is extracted along with small vicinity around it. This segment can have neighboring pieces, which needs to be eliminated. Distance transform can be applied on this image [4], and then is eroded it to get small regions at every piece. The region closest to the centroid of the square is picked and dilated and used as mask to extract the piece. Most of the images taken at any angle will have the pieces looking almost straight with slight tilt. For erosion, a rectangular structuring element whose height is bigger than width can be used resulting in elimination of overlap on sides. But the result should be further refined to perfectly extract out the piece (Figure 2).

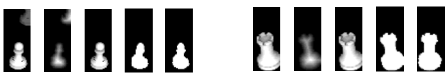


Fig. 2. Two sets of images showing extraction of piece (left to right) Test piece pawn; test piece rook; (sub images 1 to 4) small segment of image taken from the square which contained a piece; results of applying distance transform on image 1; segmented image after removal of noisy piece; mask of piece; mask of the piece after fixing the orientation

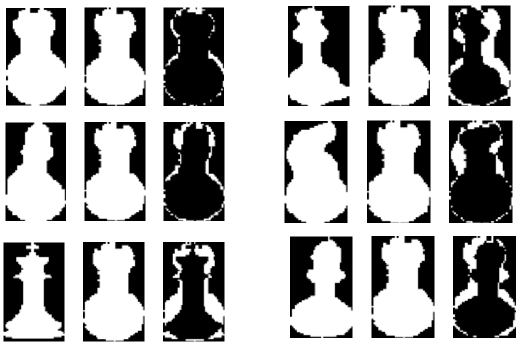


Fig. 3. Computation of area score of the test piece Rook on training piece; first row (left to right) Training rook; training Queen; second row (left to right) training pawn; training Knight; bottom row (left to right) training king; training Bishop; sub images (1 to 3) scaled mask of training piece; test piece; difference if training and test piece

The orientation of the extracted piece is fixed in a way so that the comparison with training set will be easy. Major axis that passes through the mask of the extracted piece is detected, its orientation with respect to y-axis is calculated and this angle is used to rotate the piece to make it straight. Transformation of the piece on z-axis produced results very similar to scaling. Hence for comparison, the mask training set is scaled to match the test piece. For each training piece, the difference of the scaled training mask and the test mask is calculated - the number of '1's is counted in the resulting image and this corresponds to the area score for the given training image. The training piece that corresponds to the lowest area score is the matching piece. This process is repeated to all the pieces present on the chess board to recognize the pieces (Figure 3).

III. Experimental results

Figure 4 summarizes results obtained for chess images taken at different viewing angles. The results are pretty good for angles round 45 to 60 degrees. On an average 70% to 80% of pieces can be detected correctly. This algorithm fails to detect pieces for top view and side view.

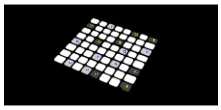
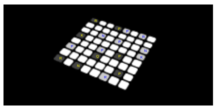
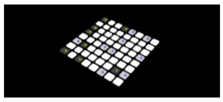
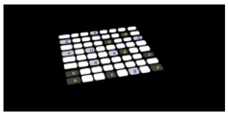
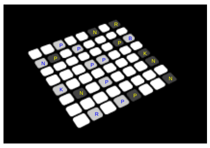
INPUT IMAGE AT DIFFERENT VIEWING ANGLES	RESULT	PERCENTAGE OF CORRECTLY DETECTED PIECES
		12/16 = 75%
		15/16 = 81%
		17/17 = 100%
		8/16 = 50%
		15/19 = 79%

Fig. 4. Table showing results on images of chess piece taken at different angles. (Notation K: King; Q:Queen; B:Bishop; N:Knight; R:Rook; P:Pawn)

IV. Improvements

There is no perfect algorithm. This algorithm too has a lot of areas of improvement.

A. *Shape matching*

Experiments with couple of image matching techniques to match the pieces are carried out. However, keypoint detectors fail to match the pieces exactly because most of the pieces have similar keypoints. Likewise if the moment on the edge of the piece is computed, the resulting numbers are also similar. The distance plots of the edge from the major axis too had similar trend for a couple of orientations.

To have better results, a mechanism to compare shapes is needed. One solution is to find the orientation of the chess board in reference to a flat chess board. For this two images are needed or the dimension of the flat chess board with respect to the given viewpoint needs to be found. Using this, matrix of the affine transform can be generated. Another solution is to match shapes using shape context by which similarity of shapes can be measured by solving corresponding points between two shapes and then estimate an align transform. The dissimilarity between the two shapes is computed as a sum of matching errors between corresponding points [5].

B. *Overlap of pieces*

The pieces need to be segmented out well when there is overlap with the same colored piece [2] [3]. In case of different colored pieces, it is easy to segment them out. Then, these results also need to be refined. Another challenging task is to properly detect and segment out pieces when the pieces and chess squares are of similar color.

C. *Add intelligence to the algorithm*

The two kings are always present till the end of the chess game. The king is generally the tallest among all pieces. Using height measurements, it can be checked if the pieces were correctly detected or not. The pieces can be categorized based on heights and only those templates can be used that belong to that category for detection. Perspective height of all present pieces can be used by applying affine transform for a given location.

Sometimes a given piece could be detected more than the valid number of times. In such cases, analysis can be done on the basis of the area score or height and recursively apply shape matching algorithm to refine the results.

D. *Different light conditions*

Since virtual chess images were used for the experiment, varying light conditions are not tested. As mentioned earlier it is safer to use MAP detector to deal with variations in light conditions.

E. *Top and side viewing angle*

This algorithm can produce good results for images taken at angles around 30 to 60 degrees. Detection of chess piece from top and side view could be improved.

F. *Chess engine*

Once the state of the chess board is known, a chess engine can be implemented to predict the next best move.

Acknowledgment

Thanks to the teaching staff and mentors for all their support in helping me decide and execute this project. Also thank my manager and my organization who gave me an opportunity to learn this amazing course.

References

[1] Wang, V; Richard Green, “Chess move tracking using overhead RGB webcam”, pp. 299-304, 27-29 Nov 2013

[2] Dhara, B.C.; Chanda, B., “A fast interactive image segmentation to locate multiple similar colored objects”, Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), 2011 Third National Conference, pp 25 – 28, 15-17 Dec. 2011

[3] Arteta, C. ; Lempitsky, V. ; Noble, J.A. ; Zisserman, A., “Learning to Detect Partially Overlapping Instances”, Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pp 3230 – 3237, 23-28 June 2013

[4] Hinterstoisser, S. ; Lepetit, V. ; Ilic, S. ; Fua, P., “Dominant orientation templates for real-time detection of texture-less objects”, Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference, pp 2257 – 2264, 13-18 June 2010

[5] Serge, Belongie; Jitendra, Malik; Jan, Puzicha, “Shape Matching and Object Recognition Using Shape Contexts”, IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 24, April 2002