Visual Chess Recognition

Cheryl Danner, Mai Kafafy
Stanford University
cdanner@stanford.edu, mkafafy@stanford.edu

***Abstract*—In this paper, we correctly detect and identify a chessboard and the configuration of its pieces through the application of image processing. While more technically challenging, the use of image processing to detect and identify a chessboard and the configuration of its pieces avoids the need for a digital chess set. Furthermore, image-based detection of chess pieces is a vital step in building chess-playing robots.**

I. Introduction

Over the years, the task of electronically tracking a game of chess has been undertaken by many people. A digital chess set can easily record a game automatically, but the specialized chessboard and pieces can cost hundreds of dollars. While more technically challenging, the use of image processing to detect and identify a chessboard and the configuration of its pieces avoids the need for a digital chess set. Furthermore, image-based detection of chess pieces is a vital step in building chess-playing robots, as the playing strategy of the robot depends on its knowing the locations of its own chess pieces and the pieces of its opponent. Such robots can be used for fun and have been considered as an interactive toy that helps in developing the learning abilities of children.

A review of existing work on visual recognition of a chessboard and pieces provided more information on recognizing the board rather than the pieces. In addition to chessboard detection in the context of playing chess, there is work on using chessboard patterns to perform automatic camera calibration [1]. Chess-playing robots such as MarineBlue [2], Gambit [3], and others [4],[5] classify pieces based on the initial positions at the start of the game and track piece movements as the game progresses. Additionally, most of the projects reviewed use a direct overhead view of the board, with a few exceptions [3] [6] [7].

A goal of this project is identification of both the board and pieces using image processing. Therefore, a direct overhead view was ruled out, as the different types of pieces are nearly indistinguishable from this angle. The resources cited earlier provide some applicable approaches to identifying the chessboard but almost no information on approaching the piece identification part of the problem.

A. Chessboard

Tam, Lay, and Levy's paper on Automatic Grid Segmentation of Populated Chessboard [6] categorizes methods for detecting the grid into corner-based and line-based, opting to use a line-based method. Tam et al. use edge detection and the Hough transform to detect lines, followed by using the geometry of a single square to extrapolate locations of points in the local neighborhood. The algorithm described in this paper is similar but uses a different method for inferring the geometry of nearby squares.

The de la Escalera paper on camera calibration [1] uses both corner detection and line detection in conjunction. Although corner detection could potentially improve results, it was not used in the final algorithm for this project because initial testing showed a high proportion of corners detected that were not relevant to the board. Also, chess pieces may obscure a number of corners, further lowering the yield.

B. Chess pieces

Most of the current chess playing robots projects [4] and [5] assume the initial positions of chess pieces to be entered manually by the player at the begining of the game, and only the piece movements on the board are tracked during the game. Identifying chess pieces is a challenging task, as they don't have texture. Therefore, matching pieces through SIFT/SURF descriptors doesn't give good results. Figure 1 shows that few inliers comes out of matching the profiles of two queen pieces using the SIFT algorithm.

Pattern or shape recognition is more suitable for identifying the pieces as it depends on their contour or silhouette shape. A good survey on shape representation algorithms was conducted in [8]. Shape representation can generally be divided into

1) Polygonal approximation.
2) Spatial interrelation feature.
3) 1-D shape signature.

As its name indicates, polygonal approximation simplifies each shape to a polygon. This method captures the overall shape of the object and ignores the fine details. Therefore, it is expected to work well for objects with high geometrical dissimilarities. Spatial interrelation features represent objects by their geometrical features and the relations between points on their contours. One of the well known spatial interrelation feature representation techniques is shape context. Shape

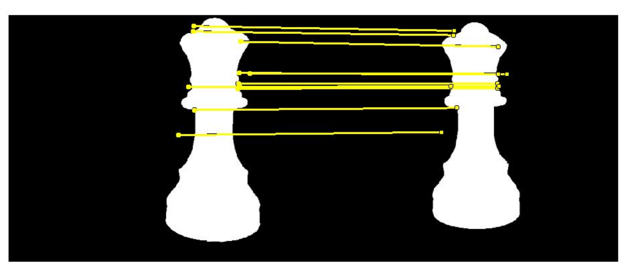

Fig. 1. Inliers from sift matching.

context takes a reference point on the object contour and makes it the center of a polar plane that is divided into bins. The histogram of the contour points in each bin is the shape representation of the object. The 1-D shape signature represents the 2-D object with a 1-D function. Chess pieces have a similar elongated shape, and the difference among them is the fine variations in their boundaries. This makes polygon approximation and the shape context not the best shape representation method.

Fourier descriptors are one of the most famous and successful descriptors in the field of pattern recognition. The main idea is to represent the 2-D shape as a 1-D function (shape signature) that depends mainly on the shape contour, then get the Fourier coefficients of this signature. The Fourier descriptors of a curve are translation, rotation, and scale invariant. The most common shape signatures are the centroid distance function and the cumulative angular function [9][10]. The centroid distance function determines a number of points on the shape boundary and gets the distance between each point and the shape centroid. This signature is translation invariant. The cumulative angular signature simply calculates the tangent angle at the boundary points as tangent angles track variations in the object contour/pattern as shown in Figure 2. The function $\phi(t)$is modified as follows to be periodic with period 2π.

$$\phi(t) = \phi\left(\frac{Lt}{2\pi}\right) + t, \quad t \in [0, 2\pi] \tag{1}$$

The addition of t also makes $\phi^8(t)$ of a circle equal zero, which indicates that a circle is shapeless or a reference shape.

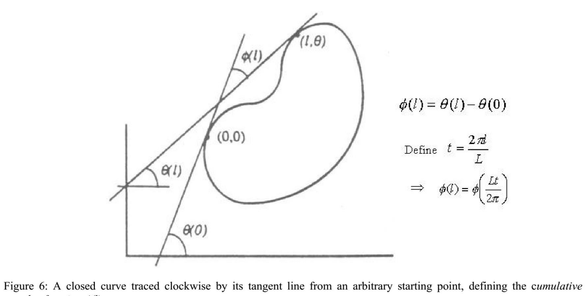

Fig. 2. Cumulative angular function [9].

II. ALGORITHMS

Initial work and testing was performed using a standard chessboard and pieces. However, segmenting the images with the default coloring was extremely challenging. To aid in distinguishing a light piece from a light square and a dark piece from a dark square, a board with red and green squares was substituted.

III. CHESSBOARD RECOGNITION ALGORITHM

The objective with regard to processing the chessboard is to find a mapping from the original image to an ideal chessboard. Having this mapping simplifies analysis of the color of each square and also associating a piece with the square that it is occupying. At a high level, the approach consists of the following steps: 1) detect edges, 2) use the Hough transform to detect lines, 3) select two pairs of lines to locate a single square, 4) extrapolate out to locate remaining squares, and 5) apply a perspective transformation to the completed board.

The first step of the process is edge detection with the goal of producing a suitable input for the Hough transform. From the original image with red and green squares, binary images for red and green channels are obtained by subtracting the other color channels from the desired channel, applying a median filter to reduce noise, and binarizing the result using Otsu's method. If the image does not contain other red and green objects, the two binary images have a foreground of only red (or green) squares (Figure 3). Edges are created using morphological edge detection on the combined red and green images, specifically, dilating the image and subtracting to get the difference (Figure 5).

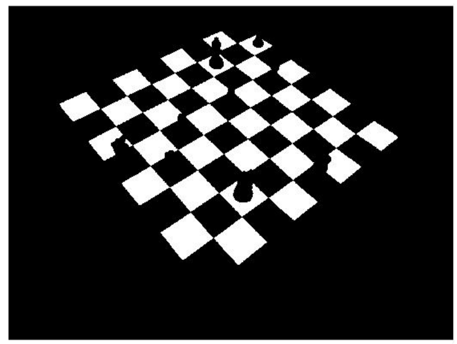

Fig. 3. Binarized image of red squares.

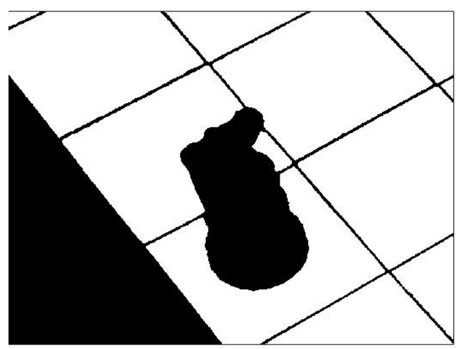

Fig. 4. Binarized red and green images combined (close-up).

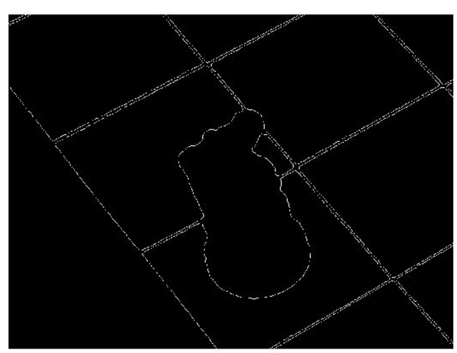

Fig. 5. Edges of binarized board image (close-up).

The Hough transform is applied, and a limited number of peaks (eighteen) are selected. The number of peaks was chosen to allow identification of all lines on the chessboard while minimizing extraneous lines. Although some extraneous lines may be identified and true lines on the board missed, a number of these errors can be handled with no negative impact. The resulting lines are categorized into sets based on the distribution of theta values. Two peaks in the theta distribution are identified (Figure 6), and all lines near a peak are selected for membership in that set (Figure 7).

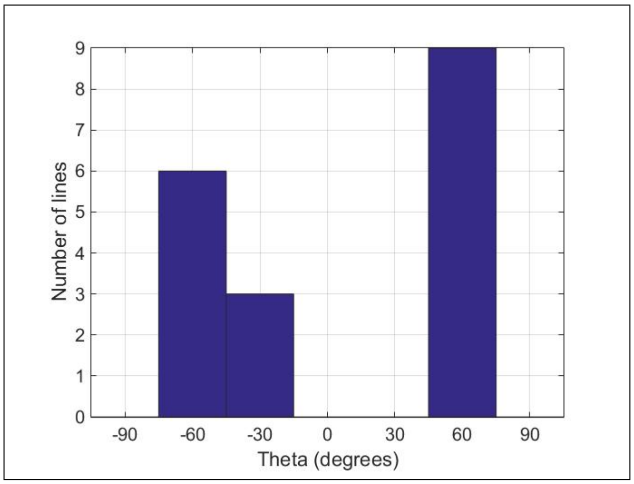

Fig. 6. Distribution of theta values associated with lines.

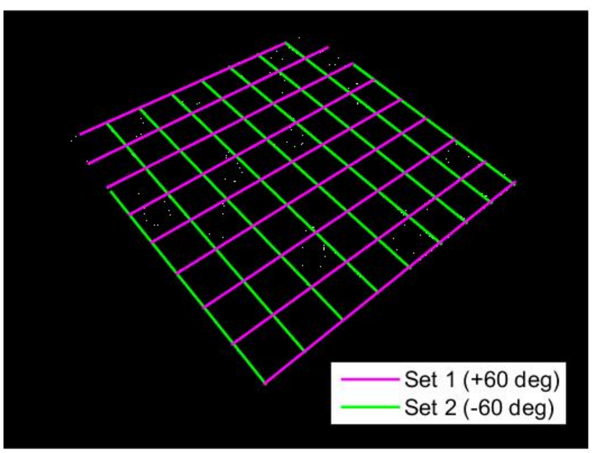

Fig. 7. Lines separated into sets by orientation.

Locating the first square on the chessboard is a critical step because the remaining squares rely on this initial square. Within each set of lines, pairwise intersection points are calculated. If many lines do not intersect, it is assumed that the lines are close to parallel, and those with theta values equal to the mode of the set are selected as the best candidates. If the set is not approximately parallel, members that intersect at points relatively far away from the other pairwise intersections are discarded, based on the fact that the true lines should all converge to a single point. Of the remaining candidate lines, the largest consecutive subset is chosen, and from that subset, the two central lines are determined to be the edges of the first square. The corners of this square are found by calculating the intersections of the final two pairs of lines.

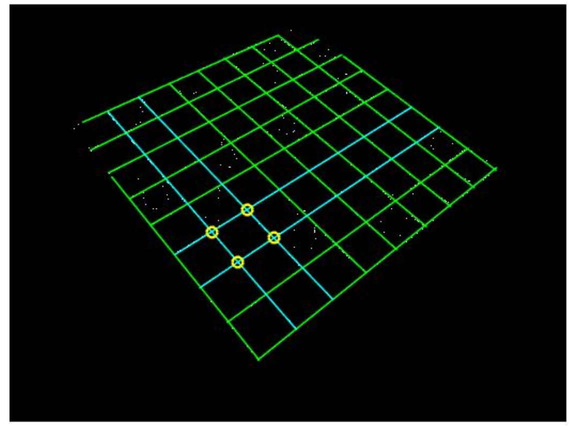

Fig. 8. First square with selected lines in cyan and intersections in yellow.

The homography transformation matrix is estimated using the four corners of the initial square and setting the last entry of the matrix (h_{33}) equal to 1 [11]. The ideal board corner x and y coordinates and the image coordinates are used to solve for the best estimate of values h_{11} through h_{32}.

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}$$

The transformation is applied to predict the locations of the two far corners of an adjacent square. Then, the set of lines with the appropriate orientation is searched for a member that approximately intersects both corners. If no line is found that meets the criteria, the Hough transform is applied to a corridor within the image where the line is expected to be, and the resulting lines are checked. In this manner, adjacent squares are located by searching to the right, left, up, and down from the initial square, until either no further squares are found, or the current row/column has eight squares. At each step, the homography matrix is recalculated using all squares that have been found so far. When a full row and column have been filled in, all horizontal and vertical lines have been determined, so the corners for the remaining squares are calculated by finding the intersections.

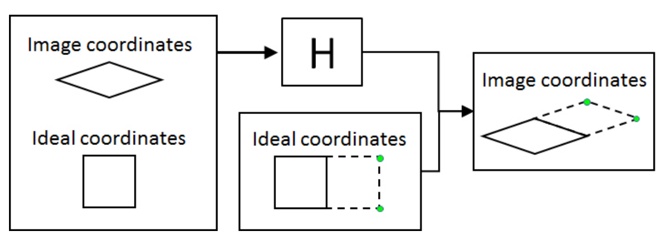

Fig. 9. Process of locating corners of an adjacent square.

Finally, an inverse transformation is applied to the completed set of squares to reshape the board area into the idealized form. The same method described above is used with all 81 corners mapped to an ideal board with squares 100 pixels wide used as input to estimate the transformation matrix. From the transformed red and green binary images, it is easy to count the ratio of red/green pixels to total red and green pixels in each square. Normalization is required because the sum of the red and green pixels in a square may be significantly less than 10,000 if a chess piece appears within the boundaries. The inverse transformation is also used to associate the detected chess pieces with the respective squares that they are occupying. A point at the base of each connected region is transformed into board coordinates, which indicates which row and column the piece is in or if it is outside of the playing area.

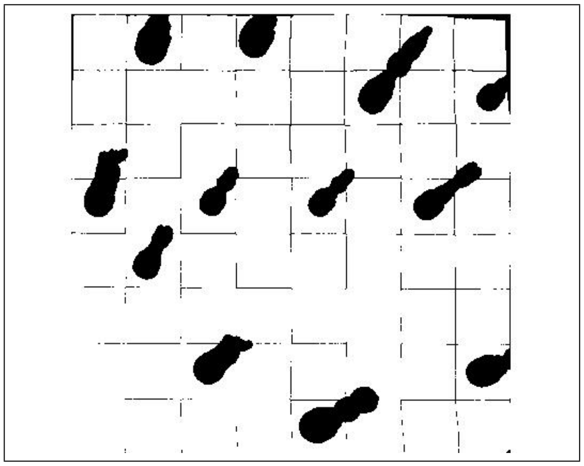

Fig. 10. Combined red and green squares after transformation.

IV. Chess piece recognition algorithm

As explained in I-B, chess pieces share much similarity with each other, for example they all have same base shape. This similarity makes choosing the centroid distance as a shape signature not very accurate (especially if we are going to normalize the distance to make it scale invariant). For this reason, using the cumulative angular function as a shape signature along with its Fourier descriptors is a better choice for detecting chess pieces as they track the finest details of the shape contour and thus track all possible variations among pieces. The following subsections explain the steps of identifying the chess pieces.

A. Constructing the database

The first thing to do is to capture reference images for each chess piece from different view angles, and to get the binary pattern (silhouette) of each piece. Some azimuthally asymmetric pieces like the knight and the king need more reference images than symmetric pieces like the pawn, rook, queen, and bishop. Figures 11 and 12 show the reference images for the queen and the king respectively. The number of reference images for the bishop, king, knight, pawn, queen, and rook are 30, 46, 49, 26, 31, and 20 images respectively.

The database has some deliberate redundancy to reduce chances of piece misidentification. The optimum number of images in database can be obtained by removing images and comparing recognition rates after additions/removals. This database was refined twice to remove ambiguous shapes. For example, pawns were mistaken for rooks because of the ambiguous rook in Figure 13. Therefore, removing this misleading image from the rook database was important to refine the database.

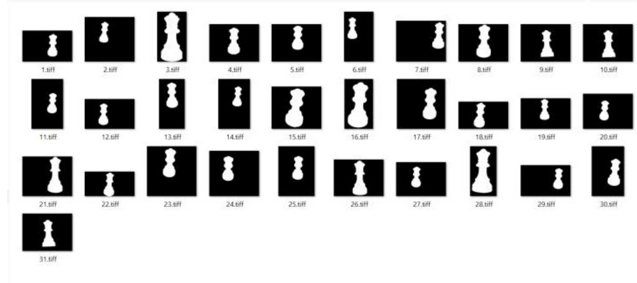

Fig. 11. Queen database.

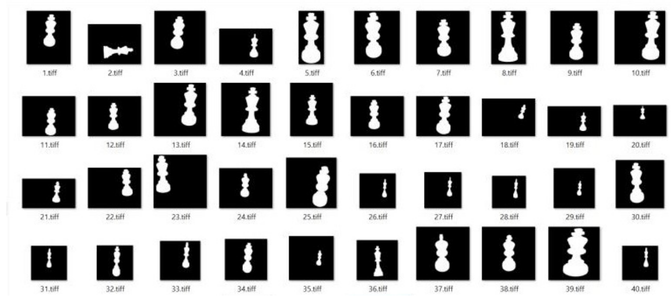

Fig. 12. King database.

B. Identifying pieces

The Fourier descriptors were calculated for each reference image by first getting the shape signature (cumulative angular function) and then getting normalized Fourier coefficients. Test images are matched to the nearest neighbor in the database in the descriptors domain.

As explained before, getting a neat contour is a vital step in piece identification. Since lighting conditions are not always controllable, glare on pieces can make the contour appear chopped or ragged. Three different methods are used to get the contours of the pieces. The first one is segmentation using unsupervised thresholding (Otsu segmentation), the second is detecting the edges of pieces using the Canny edge detector, and the third is detecting pieces by their colors.

Segmented pieces from each of the above three methods are identified independently. Then, two selection methods are used to determine a final recognition. Each connected region from each image is matched to its corresponding connected region in the two other images by comparing the centroid locations of each connected region. Also, a base point for each connected region is calculated to help identify the position of the piece on the board. This base point is taken at the base of the piece to make sure it lies in the correct square. (For example, the top point of a king will not reflect its position correctly.) The first selection method is a majority vote. For example, if two methods agreed on the same class ”pawn” of one piece, then this piece is finally determined to be from class ”pawn”. The second selection method is the minimum distance. According to this method each initial decision from (segmentation, edge detection, and color detection) is reported along with its error, then the final decision selects the initial decision with minimum error.

Also, knowing how many pieces of each class are present on the board can be used as side information to help improve

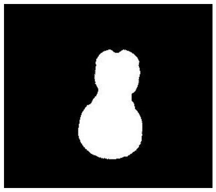

Fig. 13. Ambiguous rook.

decisions. If class ”king”, for example, has extra pieces and class ”queen” or ”pawn” has missing pieces, the kings with highest errors are reclassified and set to any of the other classes from the three initial votes.

V. Results

One challenge of our approach is finding a suitable camera angle, which becomes a trade-off between 1) obtaining a more distinctive shape profile for the pieces and 2) minimizing occlusions and simplifying board geometry. For recognizing the chessboard, an overhead angle is optimal because in the ideal case, there is no perspective transformation, and the edges of the board are not occluded by pieces (assuming pieces are placed fully within their respective squares). For recognizing the pieces, the shape profiles are most distinctive from a direct side view but indistinguishable from an overhead view. In addition, at lower angles, pieces may occlude each other to a greater extent.

To assess this trade-off space, pictures of the chessboard and pieces were taken using different camera angles, with images including the board alone, the pieces alone, and the chessboard populated with varying numbers of pieces. Four categories were created for camera angles of approximately 20 degrees (angle 1), 30 degrees (angle 2), 45 degrees (angle 3), and 60 degrees (angle 4). The following sections assess the the algorithm to identify the board, the robustness of the Fourier descriptor to recognize the pieces, and the full algorithm with discussion of the challenges.

Overall, a viewing angle of approximately 45 degrees is favorable for correct identification of both the board and the pieces. As expected, the the steepest angle category is most favorable for recognizing the board but less favorable for recognizing pieces.

A. Chessboard

Images from each of the four angle categories were processed to locate the corners of each square on the chessboard and the results scored manually to determine the number of errors. Within each angle category, images have varying board orientations and varying numbers of pieces populating the board. The total number of images tested in each category are 33 (angle 1), 39 (angle 2), 57 (angle 3), and 40 (angle 4). A success rate of 90 percent or better was achieved for the two steepest angle categories, as shown in Figure 14. The second lowest angle appears to be a breaking point, and the lowest had a very poor recognition rate.

A selection of images with poor results in the first angle category were analyzed to determine the failure mode, and the most common issue was identification of the initial square.

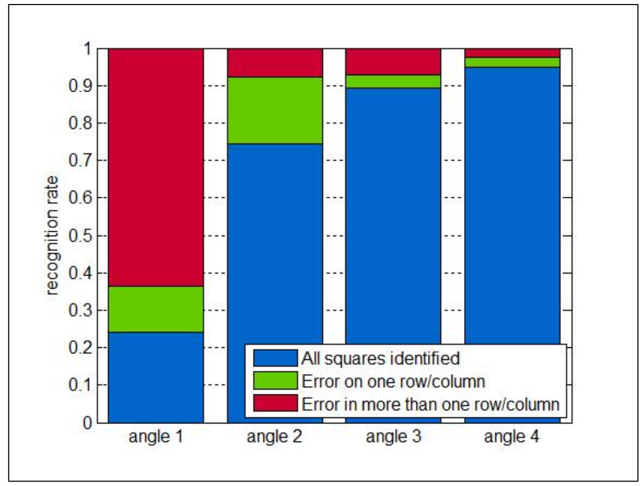

Fig. 14. Chessboard success rates at different angles.

When orientations of the two sets of grid lines are not well-separated, the two sets can be defined incorrectly, resulting in a pair of lines that are not true lines on the board. Another contributing factor is that the lines defined by the peaks of the Hough transform included a greater number of erroneous lines at the lower viewing angle.

Due to the limited scope of this project, it is assumed that the scene containing the chessboard does not contain other red and green objects. Although the algorithm was not designed to handle extraneous red objects, a small number of images were taken to test performance with red objects in the scene. In three of three images, the board had multiple errors due to interference with the other red objects.

B. Chess pieces

The piece detection algorithm was tested with 42 test pieces (with neat contours) from three different views. The algorithm is able to correctly detect all pieces of the for view 1 and view 2. However, it mistakes the rook, bishop, and sometimes the king in view 3 as shown in Figure 15. It is worth mentioning that the database doesn’t have enough images for these three pieces in the top view, as they cause confusion for the algorithm when testing pieces from views 1 or 2 with ragged contours.

It is worth noting that these test pieces have an almost perfect contour, so they do not show how robust the detection algorithm is to variations and changes in the piece’s contour.

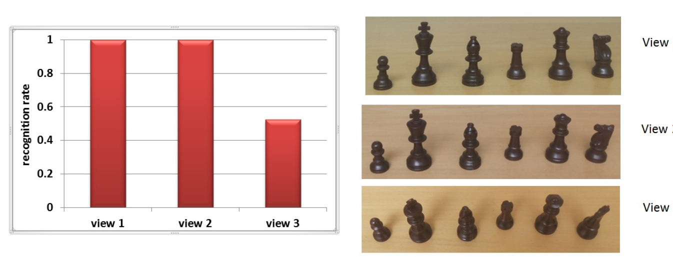

Fig. 15. Piece recognition rate

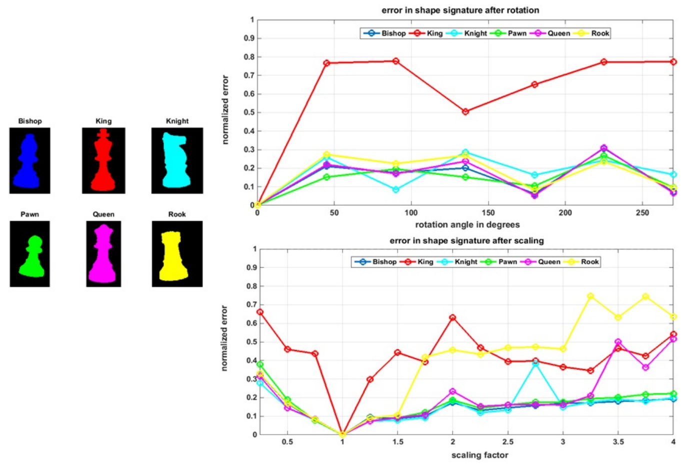

Fig. 16. Error in Fourier descriptors after rotation/scaling

Figure 16 shows a reference image for each piece and the distance between the rescaled/rotated image and the original image in the descriptor domain. As shown, all pieces (except the king) have normalized error below 30 percent for 360 degree rotation with 45 degree steps. The king and rook images have higher error for scaling. All pieces are identified correctly over the whole scaling range [0.25-4] except for the king, which was misidentified at scaling factors 0.25 and 4. All pieces were identified correctly for all rotation angles, except for the king, which was identified correctly only at the 135 degree angle. The most obvious reason for this unique behavior of the king is the high discontinuity in its contour due to the cross shape on its top.

To check the robustness of the algorithm to error in the shape signature, a zero mean Gaussian noise is added to the shape signature $\phi^*(t)$ of each piece and the recognition rate versus the error variance is sketched in Figure 17. Also, a zero mean Gaussian noise is added to the x and y positions of the boundary points and the recognition rate for each piece is sketched versus the variance in Figure 18.

C. Full algorithm (chessboard and pieces)

Figure 19 shows the recognition rate of pieces on the chessboard for images taken from angles near 30, 45, and 60 degrees. The chessboard must necessarily be recognized successfully in each image, since the full algorithm cannot run to completion otherwise. A total of 36 pieces at each angle are tested. The algorithms show a good overall performance with recognition rate between 70-80 percent at 60 degrees view angle, around 78-94 percent at angle 30 degrees and 82-100 percent at angle 45. It is worth mentioning that the lighting conditions are not exact for all cases and the testing sample is not very big, so more testing might be more informative. Figures 20-24 show steps of piece identification.

VI. Challenges and Possible Improvements

Chess pieces and the original chessboard were the same color, which made identifying the board and pieces a hard task, therefore we used a printed board with two primary colors red-green to be easily distinguished from the pieces.

Lighting conditions affect segmentation, therefore three different methods were used to extract pieces contour, voting and side information were used to improve decision if possible.

Pieces are usually well identified at low angles while the board is well identified at high angles. Angle 45 degrees is a good angle for detecting both the board and pieces.

Further improvements can be done to the project

1) Use images from different view angles to resolve occlusions.
2) Use 3D object recognition and try other shape signature and compare their results with the currently used cumulative angular function.
3) Use size variations between pieces (taking into account scaling from perspective transform) to help identify pieces.

Acknowledgment

The authors would like to thank Bernd Girod and Gordon Wetzstein for giving us the background to enable this class project, to our project mentor Jean-Baptiste Boin for his valuable guidance, and to a fellow Stanford student Jay Hack, who worked on the a project called CVChess in 2014.

References

[1] A. De la Escalera and J. M. Armingol, "Automatic chessboard detection for intrinsic and extrinsic camera parameter calibration," *Sensors*, vol. 10, no. 3, pp. 2027–2044, 2010.

[2] D. Urting and Y. Berbers, "MarineBlue: A Low-cost Chess Robot." *Robotics and Applications*, 2003. [Online]. Available: http://www.researchgate.net/publication/220785821_MarineBlue_A_Low-cost_Chess_Robot/file/50463522e1d99a1e6c.pdf

[3] C. Matuszek, B. Mayton, R. Aimi, M. P. Deisenroth, L. Bo, and D. Fox, "Gambit: A robust chess-playing robotic system," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2011.

[4] T. COUR, R. LAURANSON, and M. VACHETTE, "Autonomous chess-playing robot," *Ecole Polytechnique, July*, 2002.

[5] N. Banerjee, D. Saha, A. Singh, and G. Sanyal, "A simple autonomous robotic manipulator for playing chess against any opponent in real time."

[6] K. Y. Tam, J. A. Lay, and D. Levy, "Automatic grid segmentation of populated chessboard taken at a lower angle view," in *Digital Image Computing: Techniques and Applications (DICTA), 2008*. IEEE, 2008, pp. 294–299.

[7] J. E. Neufeld and T. S. Hall, "Probabilistic location of a populated chessboard using computer vision," *Midwest Symposium on Circuits and Systems*, pp. 616–619, 2010.

[8] M. Yang, K. Kpalma, and J. Ronsin, "Shape-based invariant feature extraction for object recognition," in *Advances in Reasoning-Based Image Processing Intelligent Systems*. Springer, 2012, pp. 255–314.

[9] K.-J. Lu and S. Kota, "Compliant mechanism synthesis for shape-change applications: Preliminary results," in *SPIE's 9th Annual International Symposium on Smart Structures and Materials*. International Society for Optics and Photonics, 2002, pp. 161–172.

[10] D. Zhang and G. Lu, "A comparative study of fourier descriptors for shape representation and retrieval," in *Proc. 5th Asian Conference on Computer Vision*. Citeseer, 2002.

[11] A. Criminisi, I. Reid, and A. Zisserman, "A Plane Measuring Device," in *Proceedings of the 8th British Machine Vision Conference, Colchester, England*, vol. 2, no. 0, 1997, pp. 699–708.

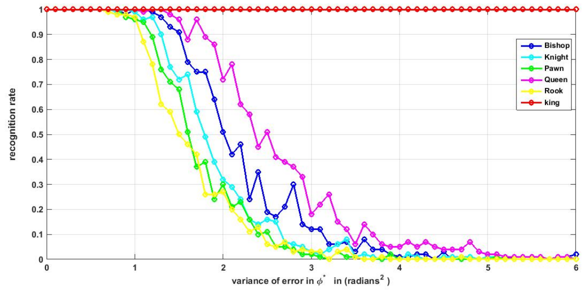

Fig. 17. Recognition rate vs. variance of error in shape signature

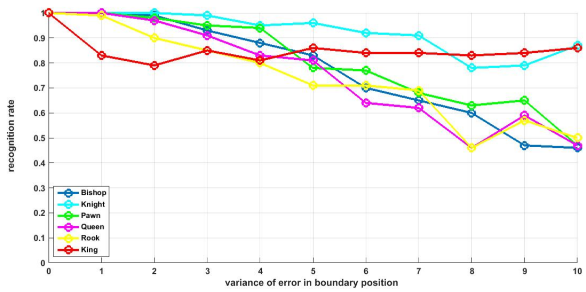

Fig. 18. Recognition rate vs. variance of error in shape contour.

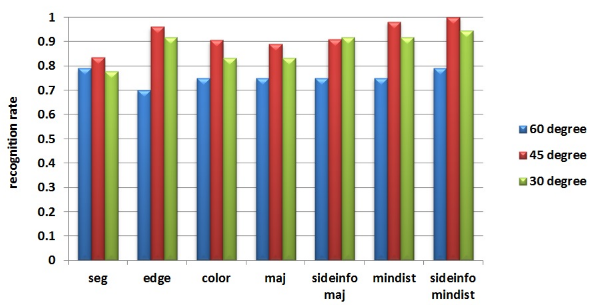

Fig. 19. Recognition rate for different viewing angles

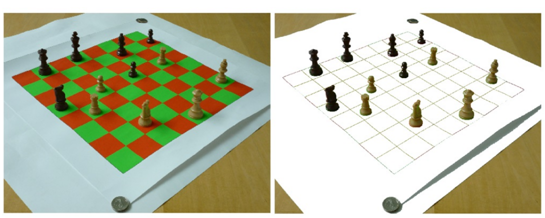

Fig. 20. Chess board before and after setting red ang green squares to white

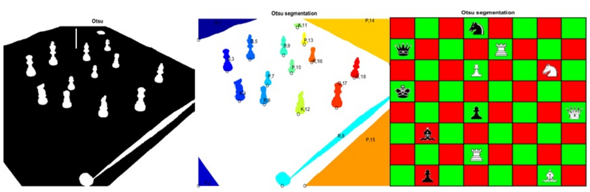

Fig. 21. Piece identification using segmentation.

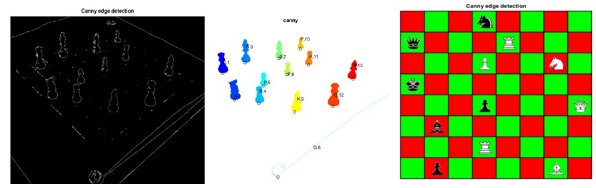

Fig. 22. Piece identification using edge detection.

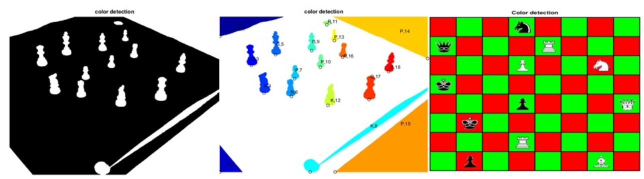

Fig. 23. Piece identification using color detection.

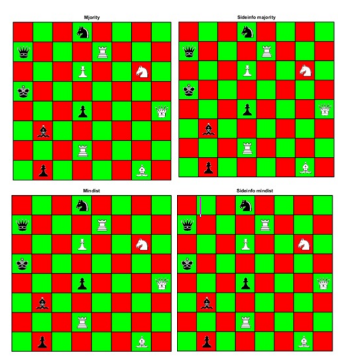

Fig. 24. Piece identification after voting.