Chinese Chess Character Recognition

Project of Signal, Image and Video:

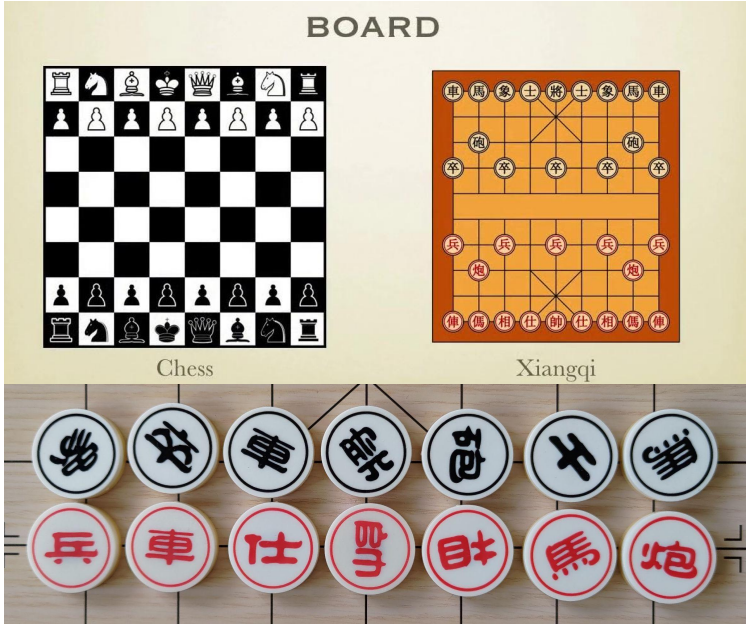
Silvano Vento Maddonni

Department of Information Engineering and Computer Science

Academic year 2023-2024

Introduction

Chinese chess, also known and referred to as **Xiangqi**, is a very popular strategic board game in many Asian countries. It is played between two players, black or red, on a board that has 9 horizontal lines and 10 vertical lines. As in the game Go, the pieces are placed on the intersections. Pieces are flat circular disks labeled or engraved with a chinese character identifying the piece type.

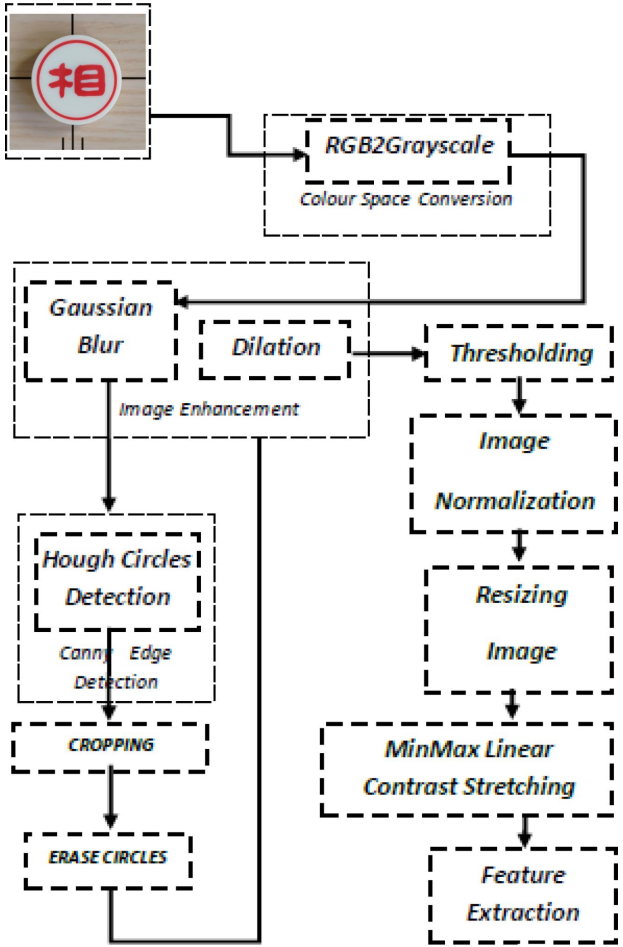


Procedure

In this project we develop a procedure that is able to detect and locate pieces on the board using various image processing techniques.

Some major steps of the algorithm include image processing, transformations, edge detection, feature extraction and template matching.

Xiangqi pieces are circular, so the main idea of the procedure is to detect circles in the image and once individuated, extract the features inside them. This revolves around the Hough Circle Detection operation (using OpenCV).



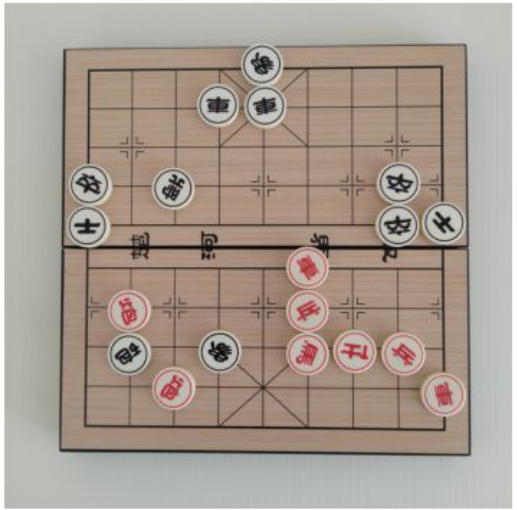
Pre-Processing

Starting from the input image, several pre-processing methods have to be used in order to correctly prepare the image for the circle detection step and to be able to extract the features.

Grayscale

The first step in the procedure is the conversion to grayscale. After this, the grayscale image is used as a starting point for many other modifications.

Original



Grayscale

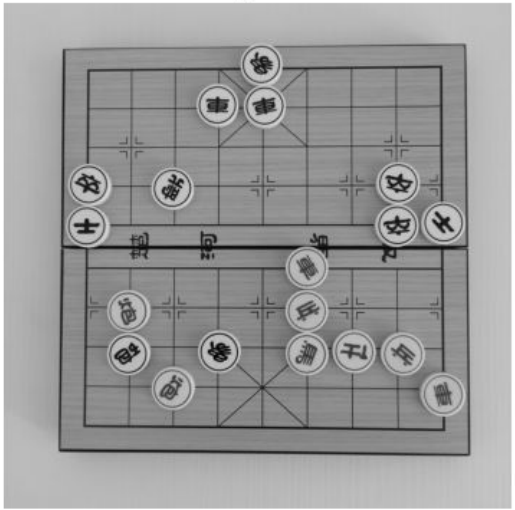
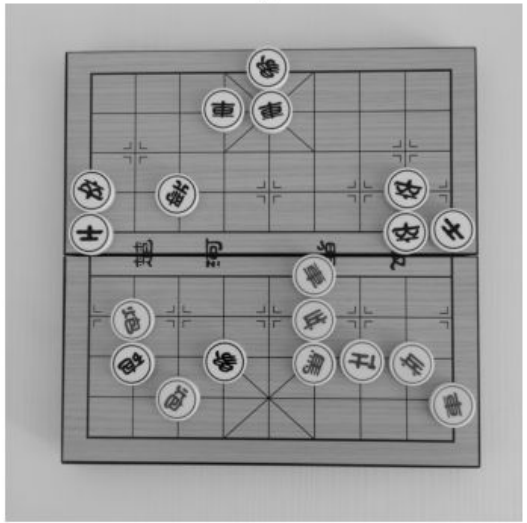


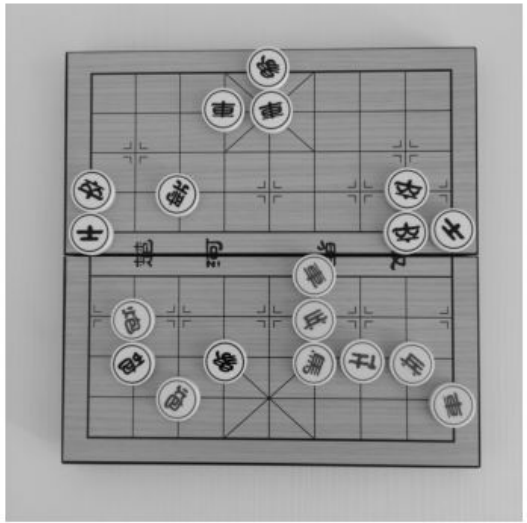
Image Smoothing (Blurring)

Image blurring is achieved by convolving the image with a low-pass filter kernel. It is useful for removing noise. In practice it removes high frequency content from the image, so the edges are blurred a little bit in this operation.

Average blur



Gaussian blur



Median blur

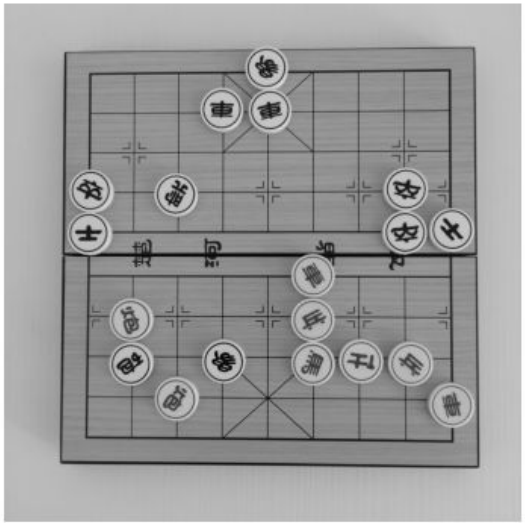
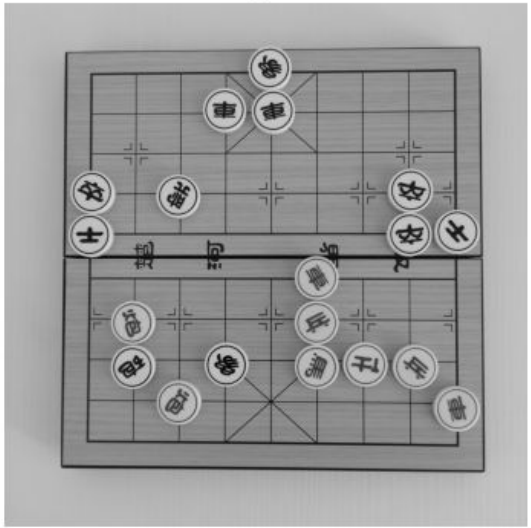


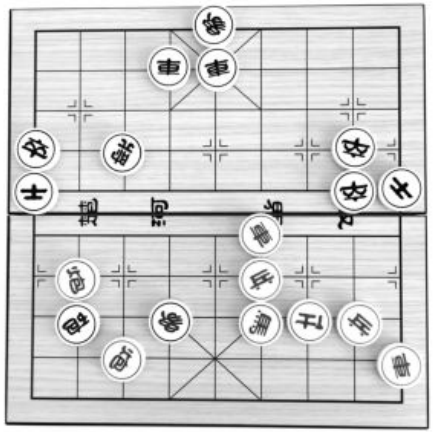
Image Enhancement and Histogram Equalization

These two techniques are used to better highlight some characteristics of the image by working on brightness and contrast. In the histogram equalization what we do is stretch out the intensity range.

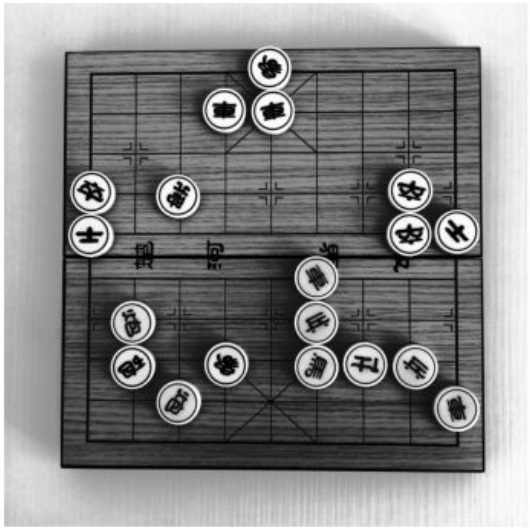
Original



Enhanced



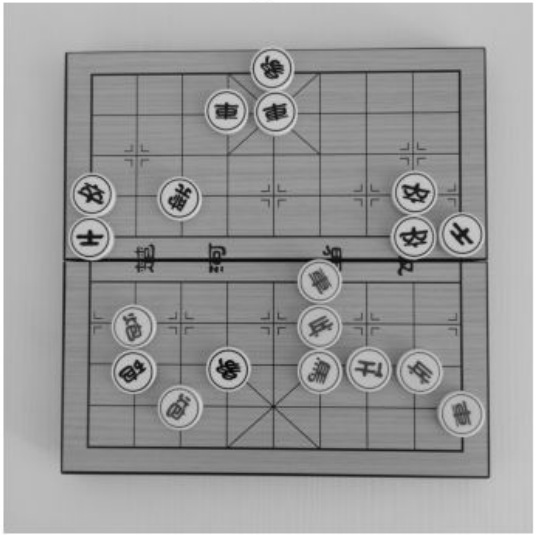
Hist. Equalization



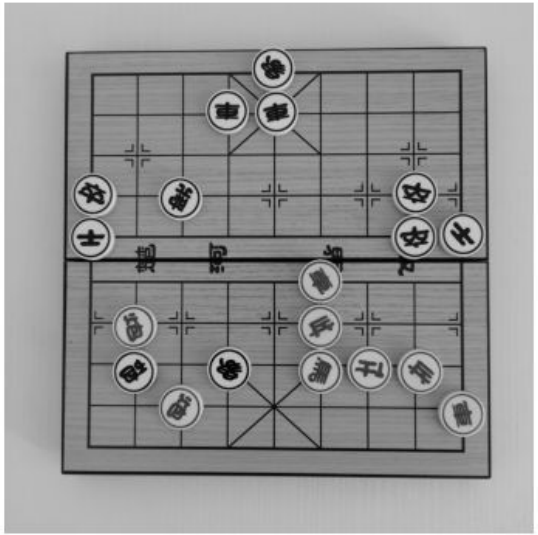
Morphological Transformations - Erosion and Dilation

Morphological operations are a set of operations that process images based on shapes by applying a structuring element to an input object. Common used operations are: Erosion and Dilation. They are used to remove noise, isolate individual elements or join separate components (e.g. fill little holes in the image). These operations consist of convolving an image A with some kernel (B) and then compute the maximal pixel value overlapped by B for Dilation, minimal pixel value overlapped by B for Erosion.

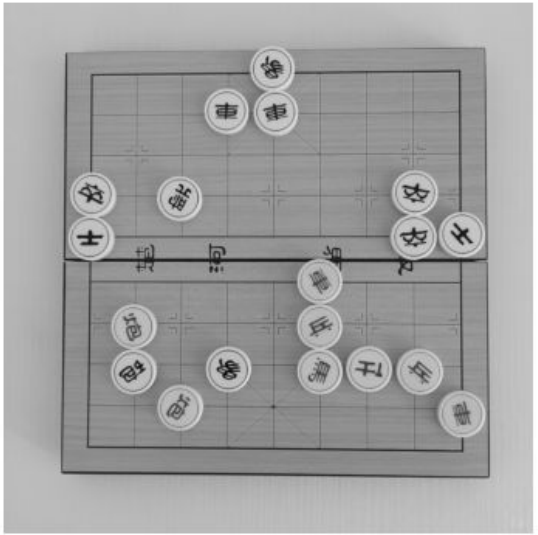
Original



Erosion



Dilation

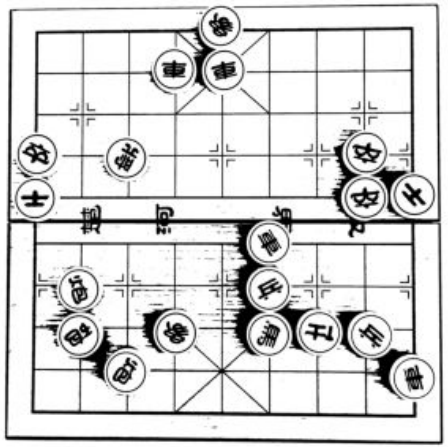


Thresholding

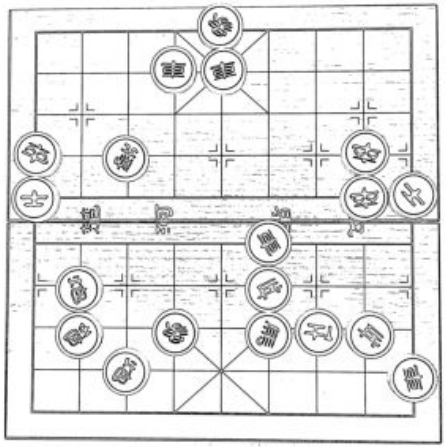
Thresholding is one of the most common and basic segmentation techniques in image processing and it allows us to better separate the objects that we are interested in from the rest of the image.

In cases (like ours) where an image has different lighting conditions in different areas, it could be better to use adapting thresholding where the threshold for a pixel is based on a small region around it rather than only on the pixel.

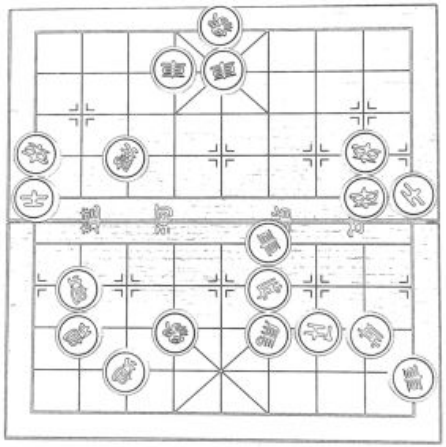
Binary



Adaptive Mean Thresholding



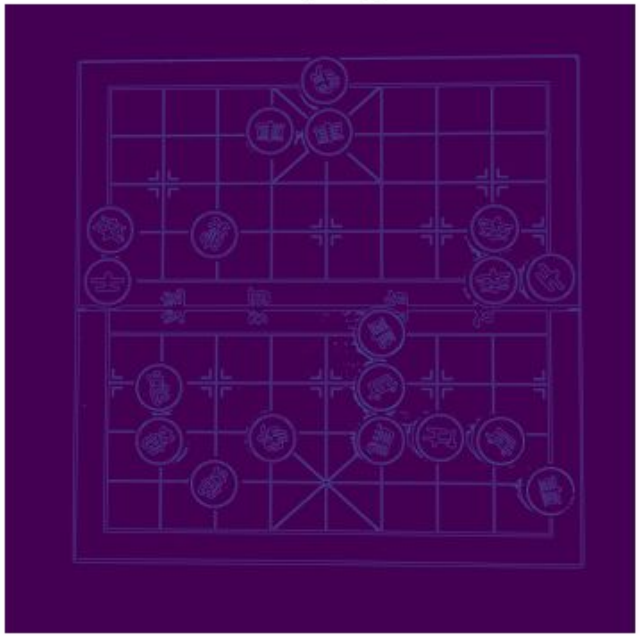
Adaptive Gaussian Thresholding



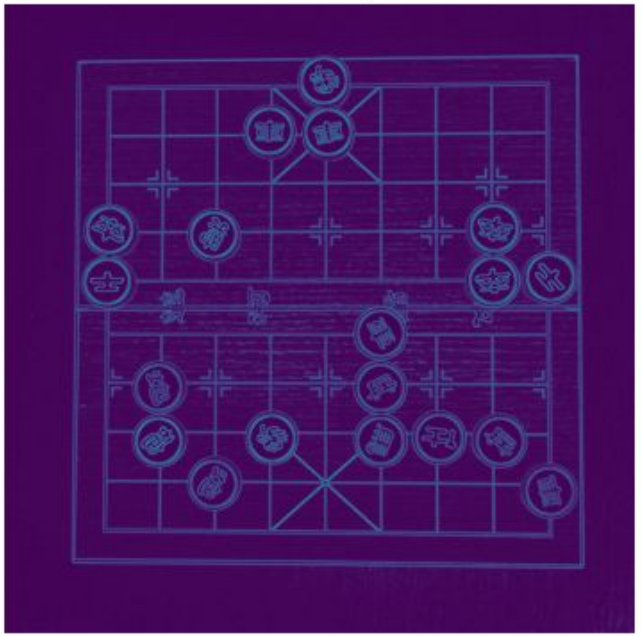
Edge detection

Edge detection is an image-processing technique that is used to identify the boundaries (edges) of objects or regions within an image. Sudden changes in pixel intensity characterize edges. We need to look for such changes in the neighboring pixels to detect edges.

Canny Edges

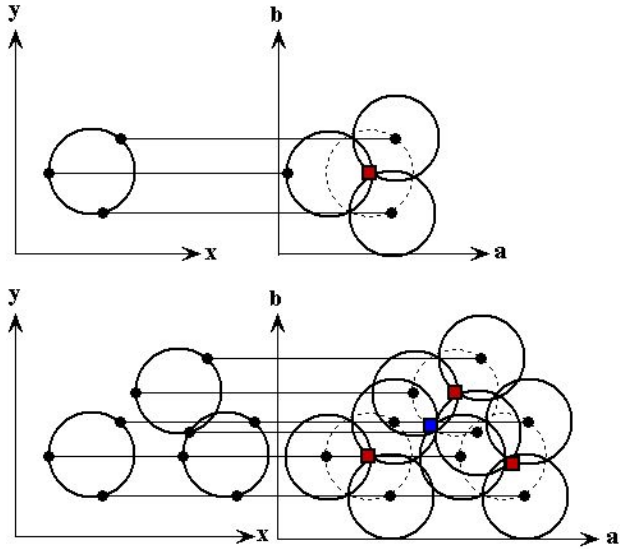


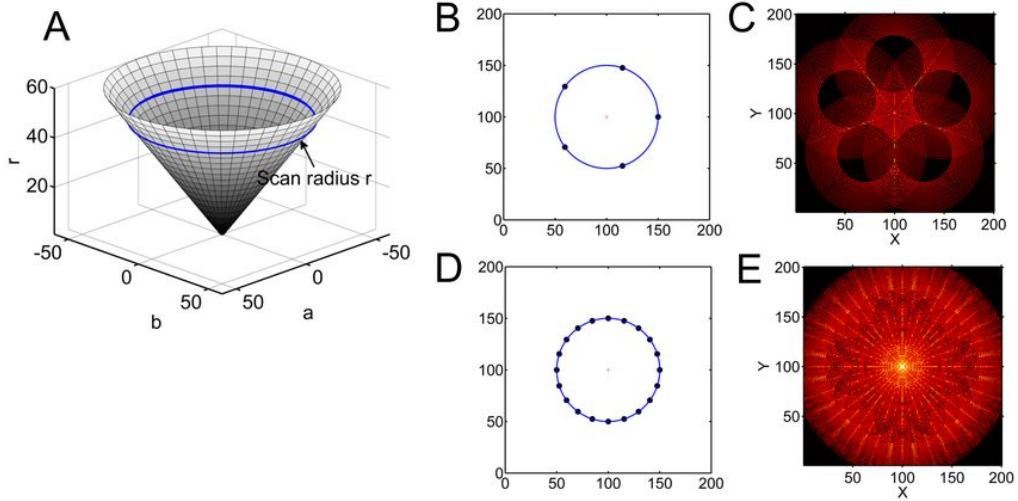
Sobel Edges



Hough Circle Transform

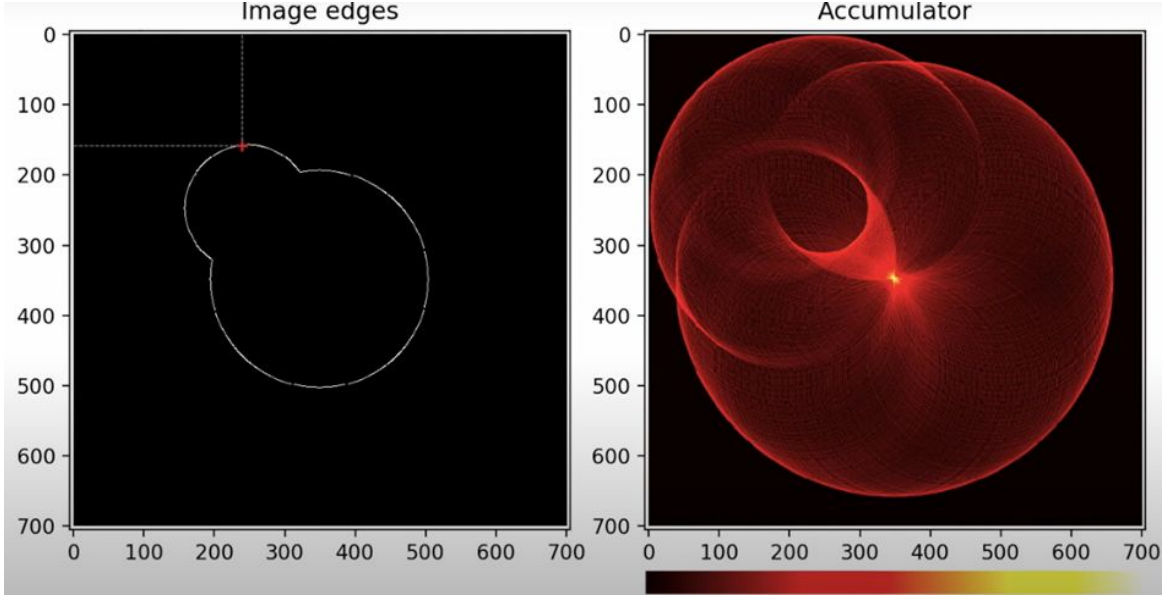
The Hough transform is a feature extraction technique that is used to find imperfect instances of objects within a certain class of shapes by a voting procedure. This voting procedure is carried out in a parameter space, from which object candidates are obtained as local maxima in a so-called accumulator space that is explicitly constructed by the algorithm for computing the Hough transform.





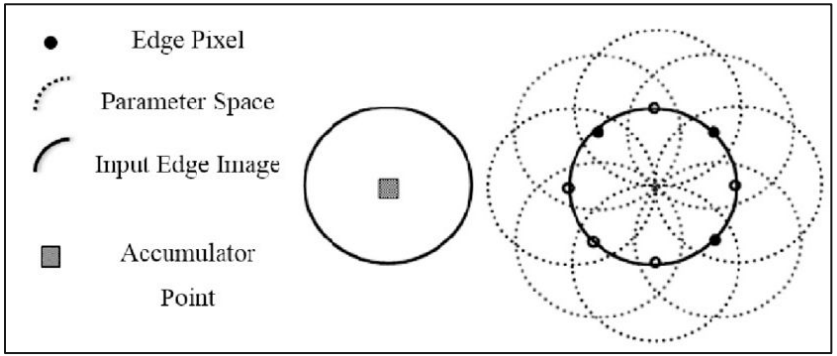
Hough Circle Transform

Hough circle transform draws circles at a certain radius by traversing the edges of the input image with the help of an accumulator with the background. The point where all these circles intersect gives us the center of the circle.



Hough Circles Parameters

cv.HoughCircles(img, **method**, **dp**, minDist, **param1**, **param2**, minRadius, maxRadius)



method

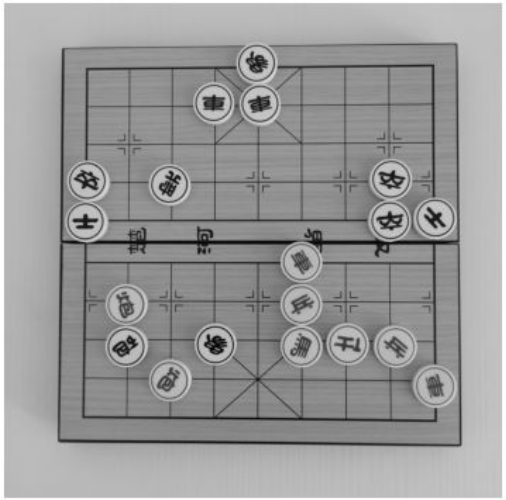
The formula used is cv2.HOUGH_GRADIENT:
this method will increment the accumulator
cells in the gradient direction of each edge pixel.

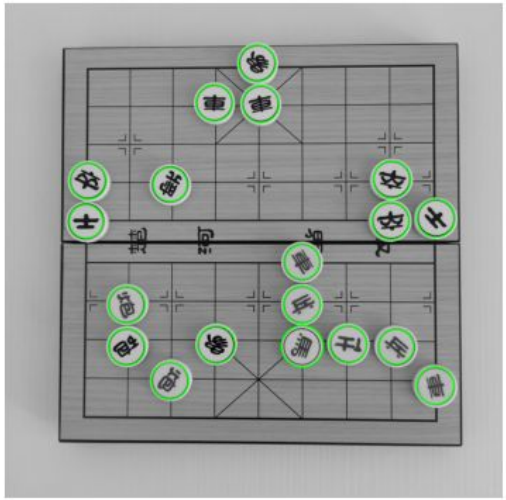
dp

dp represents the Inverse ratio of the accumulator resolution to the image resolution. During Hough transformation the input image is transformed into the Hough space where each edge pixel votes for all possible circles on which the pixel could lie by increasing multiple values within a 3-d matrix.
The bigger the matrix, the smaller your *dp* and the higher the resolution of your voting. The higher the resolution, the more accurate the circle detection.
However, the more accurate the detection, the more likely it is to miss slightly degenerated circles or detect multiple circles instead of one with a big edge.

param1 and param2

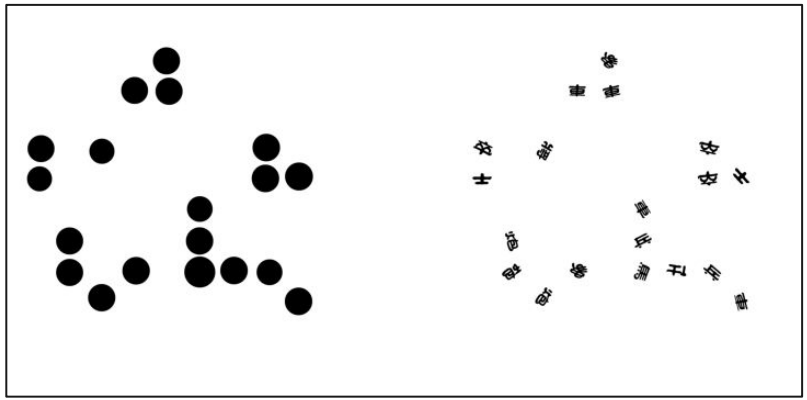
HoughCircles() internally uses Canny() function. These parameters specify how aggressively you want to detect the edges.
param1 = it is the higher threshold of the two passed to the Canny edge detector.
param2 = it is the accumulator threshold for the circle centers at the detection stage. The smaller it is, the more false circles may be detected.





Mask and extraction

In our program we pass to the HoughCircles function our image pre-processed and enhanced. After the circle detection, using a mask, we extract all the single characters. Having precedently applied binarization and blurring/correction on the image, we are able to get a clean output of the extracted black character on white background.



CLOSING and OPENING on single extracted characters

Closing

It is obtained by the dilation of an image followed by an erosion. Useful to remove small holes (dark regions).

dst=close(src,element)=erode(dilate(src,element))

Opening

It is obtained by the erosion of an image followed by a dilation. Useful for removing small objects.

dst=open(src,element)=dilate(erode(src,element))

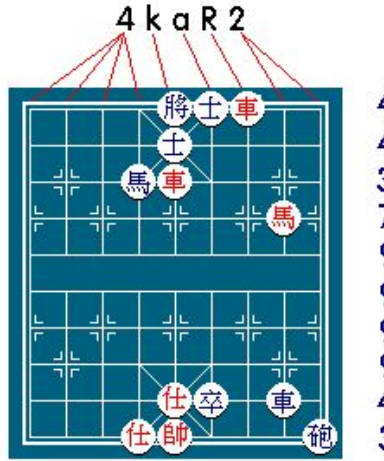
Character Identification

After we detect the circles and we extract each single character, we need a way to identify it. Ideally at the end what we want is to be able to assign to each extracted fragment its relative corresponding letter, in order to convert the position in FEN string. FEN is "Forsyth-Edwards Notation"; it is a standard for describing chess (and Chinese chess) positions using the ASCII character set.

馬 ⇨ H - (Horse)

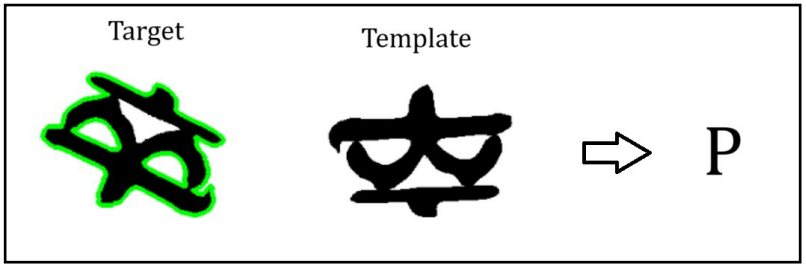
仕 ⇨ A - (Advisor)

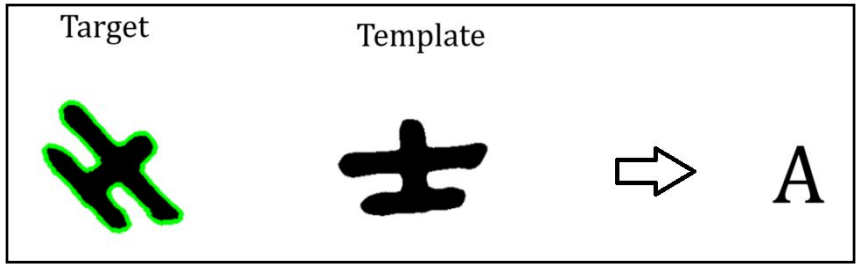
相 ⇨ E - (Elephant)



4kaR2/4a4/3hR4/7H1/9/9/9/9/4Ap1r1/3AK3c

To do this we use a composite method of findContours and matchShape. It is generally a weak approach for recognition but in this case it works because we have clean characters and also one important thing is that it is rotation invariant.
For each fragment (each single char) we use this contours-matchShape function to see if it matches one of the reference characters in our characters' database. The function works in this way:
1) Identify contours in the target image.
2) Identify contours in the template images.
3) For good candidates the template's contours are drawn over the input image and the area inside it gets erased.
4) On the erased image we calculate the percentage of black pixels that has been removed by this operation. Over a certain threshold we declare the char recognized.





Thank You
For the attention!

Silvano Vento Maddonni