Suit Counting in Playing Cards

Marcus Wong Yew Hon, Wong Shen Yang

BSc Computer Science

Abstract— **In this paper we present an algorithm for playing card detection and suit identification. This algorithm is robust enough to detect and count cards and its respective suits with moderate card rotation and photograph angle, and is achieved through thresholding, perspective correction, and template matching. The system is tested using a variety of computer generated images as well as camera captured images and return a satisfying accuracy rate of more than 90%.**

Keywords— template matching, perspective correction, thresholding, suit detection, card counting

I. Introduction

We were tasked to develop a system that would be able to detect playing cards from an input image and determine the number of playing cards in the image as well as counting the number of suits from the cards. The system should be able to distinguish between different suits and should be able to do so regardless of card orientation.

This system could prove to be useful for industries requiring identification of an object from a top down perspective. Casinos in particular will benefit greatly from systems similar to ours as there is a need for casinos to monitor playing cards on a gambling table in real time to prevent cheaters and scammers. Banks could also utilize similar systems to a degree such as monitoring and identifying currency bills in counters.

Early testing of the proposed system showed promising results whereby the average accuracy rates for card counting is at a consistent 100% for both computer-generated images and captured images. On the other hand, the average accuracy rates for suit detection and counting exceeded 90% in average accuracy rates for both computer-generated images and captured images.

II. Literature Review

Reading through several prior works in regards of card detection, the most common method for suit recognition in these works is a matching technique known as template matching. In [1], [2] and [3], the authors proposed algorithms for card recognition while implementing template matching with great success and accuracy.

Before template matching process, M. Paulo, P. R. Luís and T. Luís [1] proposed a method of pre-processing an image with grayscale conversion, Gaussian blur and using edge detection, specifically canny edge detector, before finding the contour of the cards in an image. Perspective correction and enlargement is used later, but the end result presents with an extremely blurry image of the card. This may be due to the enlargement process and may affect detection accuracy. The perspective correction is achieved by obtaining the corner coordinates of each card. The perspective transform matrix is then calculated using the obtained corner points and a newly created destination point. The top left and bottom right corners of the card are cropped and used for template matching operations. This is a simple but efficient way for obtaining sufficient card rank and suit information. Lastly, template matching is performed using a set of pre-existing template images, where the match returning the lowest value is determined to be a match. The authors have set a matching threshold whereby any match exceeding the threshold will be disregarded as a match. This method could significantly reduce the occurrence of false positives.

B. Dan and W. Hugh [2] have a similar proposed method with [1], having a slight difference in image pre-processing procedures. In [2], instead of using edge detectors, the authors proposed using binary thresholding and region filtering for card segmentation. The end result is excellent, with each segmented card being clearly distinct from one another and the image having minimal noise. However, edge detection was implemented during template matching in [2] on the cropped card corners as direct template matching without edge detection resulted in poor results. This seems to be an issue only to the authors of [2] as template matching without edge detection worked well for both M. Paulo *et al* [1] and C.B Zheng's [3] proposed algorithm. B. Dan and W. Hugh [2] have also highlighted the challenges they have faced implementing other template matching algorithms such as SIFT and SURF. Not only were the end results using these algorithms poor, the algorithms are significantly more computationally heavy compared to regular template matching as well.

Lastly, C. B. Zheng and G. Richard [3] proposed an algorithm with similar core concepts to [1] and [2], but with a very different approach in implementing and executing the perspective correction process. While the algorithm for perspective correction in [3] is similar to [1] and [2], C. B. Zheng and G. Richard proposed for each detected card to be rotated at 10 degree clockwise increments, up to 180 degrees, while performing template matching. The images are scaled to multiple sizes ranging from 90% to 40% as well. The images are matched every 10-degree rotation increment when the size of the images are scaled for each iteration. In total, 108 matches will be performed for each card. The end results for [3] prove to be rather impressive with almost flawless rank detection for images at full scale. However, the suit detection is rather poor for J, Q and K cards as the suits in these cards are generally represented by an alternative suit pattern, thus harder to be recognized by the algorithm.

.

III. Aims & Objectives

We aim to develop a system that would be able to detect and identify the number of playing cards in the image as well as counting the number of suits from the cards that is robust enough to perform well with moderate rotations of cards.

Several key steps are required to achieve our aim. Our first objective is to extract and isolate the cards from an input image. This will allow us to identify the total amount of playing cards within an image. The next objective is to obtain the suit information of each individual card within an image. Orientation correction and cropping will be necessary to obtain suit information from the card in a consistent manner. Last objective is to count the number of suits by matching the extracted suit information with a set of existing suit training images.

IV. Methodology

In this segment, we will discuss in detail about the methods that we have implemented to achieve the objectives we have set in the previous segment. In a nutshell, the chronological implementation methods of our system are as follows: image preprocessing, card segmentation, perspective correction, suit extraction, template matching and suit counting. Figure 1 illustrates the steps mentioned and would be further discussed in its respective sub-section.

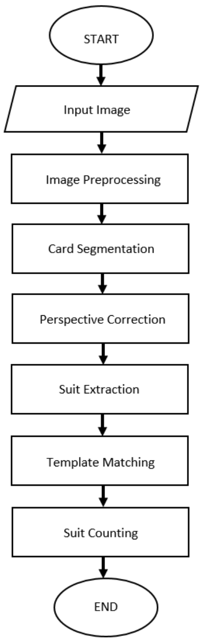

Fig. 1 Main steps in playing card suit counting.

A. Image Pre-processing

Typical in a casino table setting, where card detection and recognition are commonly implemented, playing cards are usually presented on a table where there exists a stark difference of contrast between the cards and the table surface. This brings us to utilizing thresholding to segment the cards from the background to find the card contours. However, pre-processing of the image must first be done prior to segmentation.

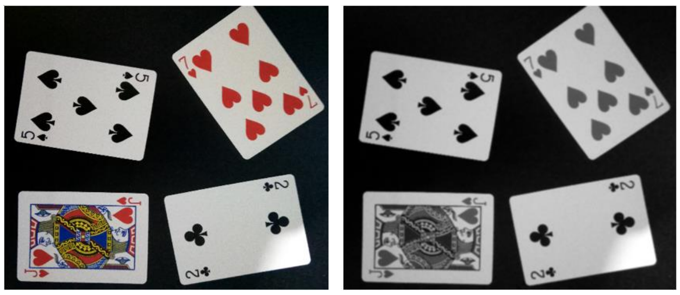

Fig. 2 Before and After Applying Gaussian Blur

The image is first converted from a RGB image to a grayscale image. Then, a Gaussian blur is applied to the grayscale image before being converted into a binary image as shown in Figure 1. This is to reduce the occurrence of false positives such as noise or light reflections to be detected as contours in later stages.

B. Card Segmentation

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) \geq T \\ 0 & \text{otherwise} \end{cases} \qquad (1)$$

If g(x,y) is a thresholded version of f(x,y) at some global threshold T

The blurred image is then converted into a binary image through a simple global thresholding method denoted in (1). There is no need for a complex thresholding method as the images are assumed to be well perceived and contrast strongly with its background. Edge detection is not implemented as we find thresholding to be sufficient in card segmentation operations.

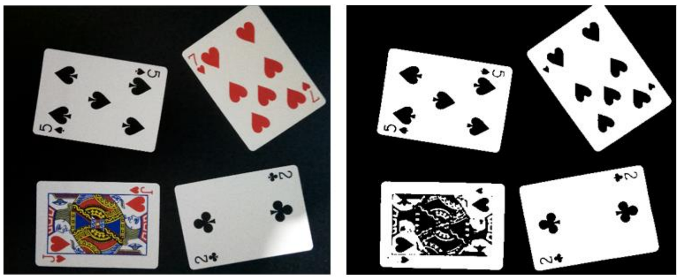

Fig. 3 Before and After Thresholding

After thresholding, each of the cards in the binary images would be extracted into individual components to be processed. The thresholded image is then subjected to a contour finding algorithm provided in the extensive OpenCV library called cv2.findContours. The detected contours will then be drawn by implementing the cv2.drawContours function to provide visual representation of the detected cards as shown in Figure 4.

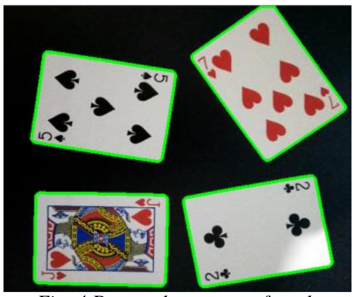
Fig. 4 Detected contours of cards

The contours not only serve as a base for future card processing operations such as perspective correction and suit extraction but also allow us to obtain the total card count as each detected contour is regarded as an individual card. However, the method of card detection using contour detection is only used because it is assumed that the given images will have well illuminated objects (cards) against a dark background, as is the case in casino tables. If images are subjected to light background with little contrast between the target objects, other methods such as edge detection will have to be implemented instead.

C. Perspective Correction

As the cards will need a consistent vertical orientation to proceed with suit extraction via cropping, perspective correction would have to be done on the card contours. Corner points of respective card contours will have to be calculated first. This can be achieved by obtaining the perimeter of each contour and use it to approximate the shape of the contour. Once the approximate shape has been gathered, the vertices coordinates of the card can be extracted. A bounding rectangle formed from the contours will provide dimension information as well (e.g width, height). With these coordinates and dimensions, perspective correction operation can then commence. Figure 5 shows the bounding rectangle of the dimensions for each contour.

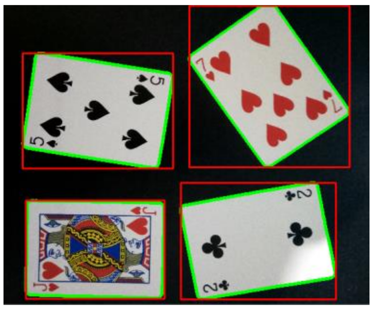
Fig. 5 Bounding rectangle for each contour

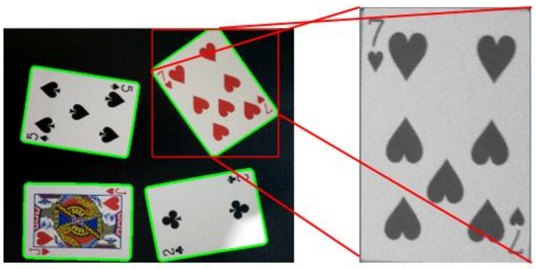
Fig. 6 Perspective corrected view of card at 200x300 resolution

For perspective correction, the goal is to convert each card's orientation into a perfectly vertical 200x300 resolution top-down orientation view as illustrated in Figure 6. The vertices points of the cards are stored into an array before doing perspective transform. The order of vertices points storage is dependent on the orientation of the card. Using the coordinates and dimensions obtained earlier, we can determine the orientation of the card by comparing between the width and height ratio of the bounding rectangle shown in Figure 5.

Once the orientation is determined and the array of corner points (c) stored, a destination array (d) is created to define our intended top-down vertical view. Then, a perspective transform is calculated by collecting the corresponding points from (d) and (c) using the cv2.getPerspectiveTransform function. The result of the perspective transform is then used to warp the perspective of the card into our intended orientation and view. This is achieved using the cv2.warpPerspective function. Lastly, the perspective corrected card is converted to grayscale for easier processing in later stages.

D. Suit Extraction

This segment details on how the suit of the card will be extracted from the perspective corrected card. We have found that the corners of a playing card would provide sufficient information regarding a card's suit and even its rank. As the card is now at a constant resolution of 200x300 pixels, we can safely crop out the corners of the card at a fixed size without worrying about information loss due to the crop. Figure 7 illustrates the extracted corner of a card after having its perspective corrected.

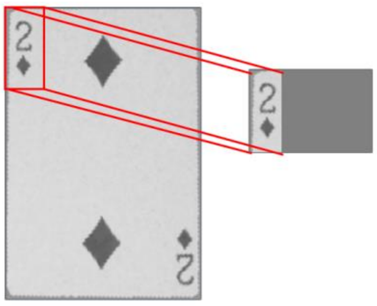
Fig. 7 Extracted corner of a card

Besides providing ample suit information, the card corner will also significantly reduce the computational time required for template matching operations later as the area of the image is now on a much smaller scale. It will also increase accuracy as there are lesser elements within the corner of the card, thus reducing the likelihood of false positives to occur in template matching.

E. Template Matching & Suit Counting

Template matching is a method of finding small parts of an image by comparing it with a matching image of similar size, known as template image. We have decided on using template matching as it is a simple, straightforward method that is more than enough for our system needs. This means reduced computational time especially when compared to other matching algorithms such as SIFT and ORB.

We implemented template matching by matching the extracted corner of each card with an array of suit training images. The template with the best match will return the highest max value. Figure 8 shows the template images used for our system's template matching operation.

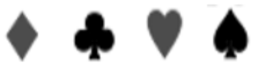

Fig. 8 Suit training images

For counting the suits, a threshold for the max value is set so only matches exceeding the threshold would be counted as a match for its respective suit. This reduces the rate of false positives and increases the accuracy of our system.

F. Testing

Our system will have to go through two testing phases with two distinct categories for each phase. The categories being computer generated images and pictures captured with mobile phone camera. The steps for the tests conducted will be the same for each category. We used a total of 10 images comprising of 5 computer generated images and 5 mobile phone camera images for the tests.

Pre-requisites:

The images that are to be used in testing must have its target objects (cards) be sufficiently illuminated with a high contrasting background to mimic a casino table environment. The target objects also must not be partially occluded and be placed separately with adequate spacing.

Phase 1 – Counting total number of cards:

The system will run and return the number of detected cards within an image. The results will be presented in percentage form and the accuracy rate will be calculated as follows:

$$Card\ counting\ accuracy = \frac{total\ cards\ detected}{total\ cards\ in\ image}\ x\ 100\%$$

Phase 2 – Suit Counting

The system will run and return the count of cards based on its suits from an image. However, since the accuracy rate of suit recognition corelates directly with the accuracy of suit counting, the test will calculate the accuracy of suit recognition in an image instead. The results will be presented in percentage form and the accuracy rate will be calculated as follows:

$$Suit\ recognition\ accuracy = \frac{total\ correct\ suit\ detection}{total\ cards\ in\ image}\ x\ 100\%$$

The results for the tests will be tabulated and discussed in detail in the next segment.

V. Results and Discussions

The test results are recorded and tabulated into two tables, each representing its respective category. Each of the tables are further split into two tables, one for total card number count results and another for suit counting accuracy results, amounting to 4 tables in total.

A. Computer Generated Images Results

TABLE I. CARD COUNTING ACCURACY RATE

Input Image	Card Counting Accuracy		
	Total Cards Detected	Total Cards In Image	Results
1	10	10	$\frac{10}{10} * 100\% = 100\%$
2	15	15	$\frac{15}{15} * 100\% = 100\%$
3	2	2	$\frac{2}{2} * 100\% = 100\%$
4	9	9	$\frac{9}{9} * 100\% = 100\%$
5	8	8	$\frac{8}{8} * 100\% = 100\%$
		Average:	100.00%

TABLE II. SUIT DETECTION ACCURACY RATE

Input Image	Suit Detection Accuracy		
	Total Correct Suits Detected	Total Cards In Image	Results
1	10	10	$\frac{10}{10} * 100\% = 100\%$
2	15	15	$\frac{15}{15} * 100\% = 100\%$
3	2	2	$\frac{2}{2} * 100\% = 100\%$
4	9	9	$\frac{9}{9} * 100\% = 100\%$
5	5	8	$\frac{5}{8} * 100\% = 62.5\%$
		Average:	92.50%

B. Captured Images Results

TABLE III. CARD COUNTING ACCURACY RATE

Input Image	Card Counting Accuracy		
	Total Cards Detected	Total Cards In Image	Results
1	4	4	$\frac{4}{4} * 100\% = 100\%$
2	6	6	$\frac{6}{6} * 100\% = 100\%$
3	4	4	$\frac{4}{4} * 100\% = 100\%$
4	5	5	$\frac{5}{5} * 100\% = 100\%$
5	6	6	$\frac{6}{6} * 100\% = 100\%$
		Average:	100.00%

TABLE IV. SUIT DETECTION ACCURACY RATE

Input Image	Suit Detection Accuracy		
	Total Correct Suits Detected	Total Cards In Image	Results
1	3	4	$\frac{3}{4} * 100\% = 75\%$
2	6	6	$\frac{6}{6} * 100\% = 100\%$
3	4	4	$\frac{4}{4} * 100\% = 100\%$
4	4	5	$\frac{4}{5} * 100\% = 80\%$
5	6	6	$\frac{6}{6} * 100\% = 100\%$
		Average:	91.00%

C. Discussion

Based on tables I and III, the results show that our system's ability to distinguish card objects from an input image, be it a computer-generated image or a captured image, is at a 100% rate of accuracy. This is evident that our system is extremely reliable in the aspect of card detection and card counting.

Now to specifics, the suit detection test for computer generated images gives us an average accuracy rate of 92.50%. There appears to have a large drop in accuracy for input image 5 at only 62.5% accuracy. There are many factors that would cause the dip in accuracy for these test cases. One of them being the quality of the source image itself. If the source image presented are of low resolution or low quality, the cropped corner of the extracted card will yield an inconclusive match during template matching, which contributes to the lowered accuracy rate. Another factor that may have affected the accuracy is the implementation of template matching itself. As our template matching algorithm requires a high matching rate to register a match, coupled by the fact that the computer generated images are from several different sources, the slight deviation of the source image's suit design might cause the system be unable to detect it as a match. In the case of input image 5, none of the spades cards have been detected as a match. Upon closer inspection, it is found that the pattern of spades in input image 5 is vastly different from our predefined spade training image as shown in Figure 9.

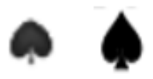
Fig. 9 Spade pattern of source image (left) and spade pattern of training suit (right)

Moving on to the suit detection test for camera captured images, at 91.00%, there is only a 1.5% dip in average accuracy percentage compared to the computer-generated images test. It is interesting to note that a significant number of cards in the captured images have moderate to major rotations in them. This makes the accuracy rate for this test to be surprisingly decent and this proves that our system is robust enough to perform well even with moderate to major rotations of cards. Test cases for input image 2, 3 and 5 yielded a 100% accuracy rate whereas input image 1 yielded a 75% accuracy rate with 1 undetected suit while input image 4 yielded an 80% accuracy rate with 1 undetected suit. While some suits go undetected in computer generated images due to template mismatch attributed by variations in suit patterns, the reason is different for captured images. For some cards that have extreme rotation in excess of 70 degrees from the perpendicular of an upright card, the perspective correction algorithm will warp the card horizontally instead of the intended vertical view. This error in perspective correction have in turn affected the corner extraction for suit information. Without the suit information, a match will not be registered for that card during template matching operations. Figure 10 illustrates an example of an erroneous perspective correction of a card.

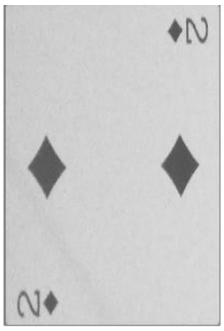
Fig. 10 Erroneous perspective correction of a card

VI. CONCLUSION

In this paper, we have proposed an algorithm for a system to be able to extract and count cards from an input image reliably. We have also proposed a suit recognition algorithm capable of identifying and counting card suits in an image with an accuracy rate exceeding 90%. However, there are flaws in our algorithm that have led to occasional decrease in accuracy for some cases due to the nature of the methods used, e.g template matching, in the algorithm as well as the inconsistency of the perspective correction algorithm when faced with cards of extreme rotations. Thus, future enhancements to the matching algorithm and perspective correction method must be done in order for the system to approach higher accuracy rates for suit recognition.

REFERENCES

[1] M. Paulo, P. R. Luís and T. Luís, “Poker Vision: Playing Cards and Chips Identification based on Image Processing,” [Online]. Available: https://pdfs.semanticscholar.org/0077/07dded08ae36b301e35734d8136c13e821fc.pdf. [Accessed 3 May 2018].

[2] B. Dan and W. Hugh, “Texas Hold’em Hand Recognition and Analysis,” [Online]. Available: https://stacks.stanford.edu/file/druid:yj296hj2790/Brinks_White_Texas_Hold_Em_Hand_Recognition_and_Analysis.pdf. [Accessed 2 May 2018].

[3] C.B.Zheng and G. Richard, “Playing Card Recognition Using Rotational Invariant Template Matching”, [Online]. Available: https://pdfs.semanticscholar.org/d064/6c9b7c8111e1c7ed5b34827324c322c52638.pdf. [Accessed 3 May 2018]