Playing Card Detection and Identification

Daniel Snyder, Stanford, CS 232 Winter 2019

***Abstract*— Playing cards are a typical item in households. They are used in a variety of games such as poker and blackjack. In some of these cases it can be helpful to algorithmically identify the playing cards from camera images. This paper explores two approaches to detect and identify cards in a static image. An approach using SIFT is first tried, but due to highly similar features, fails to produce meaningful results. Secondly, a template matching based approach is explored and analyzed. For ideal cases, it reaches a 98% success rate for correct identification. Finally, the non-ideal case of perspective distortion is explored and tested.**

I. Data Collection

Two sets of data were collected for this project. One set is the training images used to develop the templates. The other set is the images used to test the implementation. In this document, the first set is referred to as training or template images and the second set is referred to as target images.

Both sets of images were collected with a Pixel 3 camera using the standard settings. In order to get a consistent performance, dark backgrounds were used in all of the images to ease in the card detection process. For the training images, 52 pictures were taken, one for each of the 52 different cards. A set of cards were randomly chosen for each of the testing images. An example training image is shown in Figure 1, and an example of a testing image is shown in Figure 2.

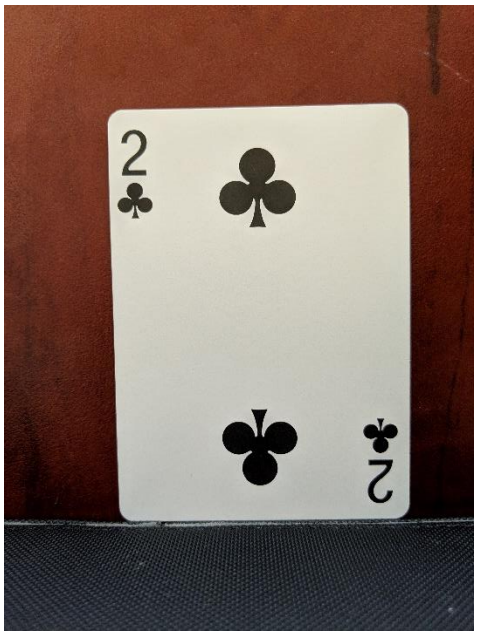

Figure 1: Example Training Image

Figure 2: Example test image

II. Approaches

For this project, two approaches to detecting and identifying playing cards were explored. The first method uses feature detection with SIFT to try and find matches between the keypoints in the target and the training images. The second approach uses thresholding, edge detection and template matching to compare each card in the target image to a set of pre-trained images.

A. Approach #1: Scale invariant Feature Transform

This approach is designed around the Scale Invariant Feature Transform [1](SIFT) algorithm (provided by the vl_sift library [2]). The idea is to match SIFT descriptors in the target image with SIFT descriptors that have been precomputed for each type of card.

The precomputed descriptors are obtained by applying SIFT keypoint detection to each of the 52 training images. The location of each keypoint and the corresponding feature descriptor are then stored for later use.

For detecting the cards in a given image (the target image), SIFT keypoints and descriptors are first found using the vl_sift function. Then for each of the 52 cards, vl_ubcmatch is used to match the feature descriptors to the precomputed descriptors. Finally, RANSAC [6] is used to find the transform that best maps the features to the matches found by vl_ubcmatch. The number of features matched is then stored. If for a given card, more than n of the features are found in the target image, the card is considered to be in the image.

B. *Approach #2: Template Matching*

In this approach template matching is used to score how similar each card in the target image is to each of the 52 cards in a standard deck of cards using pre-trained data. To get the set of training data, the image is thresholded to identify the white card against the dark background. The edges of the region are then found along with the corners. The corners are arranged in a pre-defined order, and the template is stored.

For detection, the image is thresholded, to isolate each of the cards in the target image. Then for each large white region in the thresholded image, the edges and corners are found and arranged. Then, for each template image, a transform is found to map the template onto the card in the target image, and each pixel of the template is mapped onto the target image. The sum squared difference between the mapped template and the target image is then taken as the score of the match. The lowest score of all the templates is selected as the best match.

1) Card Detection

Card detection is done by via thresholding and the use of MATLAB's built-in regionprops function.

In order to apply a threshold to the image, the r, g, and b channels are added up, and a hand-picked threshold is used to binarize the image. Because this project is focused on known backgrounds, the threshold can easily be picked to remove a dark background and leave only the white cards.

On the thresholded image, MATLAB's regionprops function is used to get region statistics. Areas of a small size are removed to prevent spurious bright regions from causing false positives. This creates one region for each card in the image. An example region detected by this step can be seen in Figure 3.

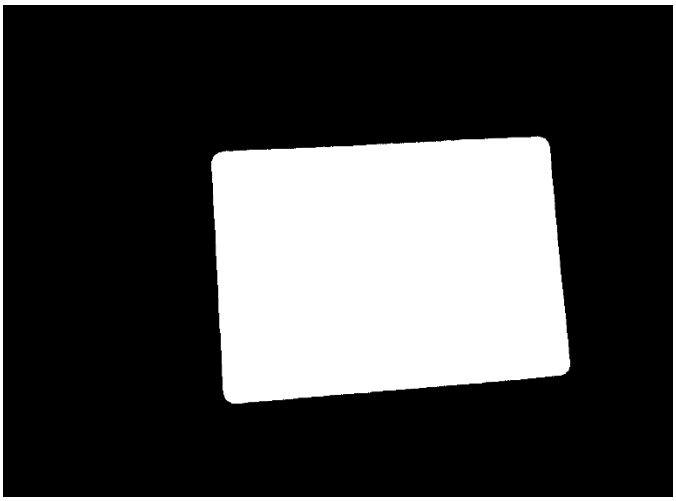

Figure 3: Example region detected by Card Detection

2) Corner Finding

For each region, the outside corners are found. A canny edge detector is used to find the outside edges of the card. A Hough transform is then used to get the equations of the four dominant edges in the image, which represent the outside of the card. A transform from ρ, θ to x, y is then used to get the equation of each line. From the line equations, the 4 intersections that represent the corners are found and stored.

3) Corner Arrangement

To increase the accuracy of finding the correct transform, it is important to arrange the corners in a standardized order. This prevents the card from being distorted when calculating the transform. The predefined order makes the first corner one with a number in it, and then goes along the short edge to the second corner, continuing along the edges marking the third and fourth corners.

The corner with the number was identified by finding the dark region inside of the card, whose centroid was closest to a corner. In all the standard cards, the dark region closest to a corner is the number or letter identifying the rank of the card. So the corner closest to the dark region's centroid can then be marked as the first corner. From there, Euclidean distance between the corners is used to find the short edge and the second corner.

Figure 4 shows an example of a card that has had its corners detected and arranged.

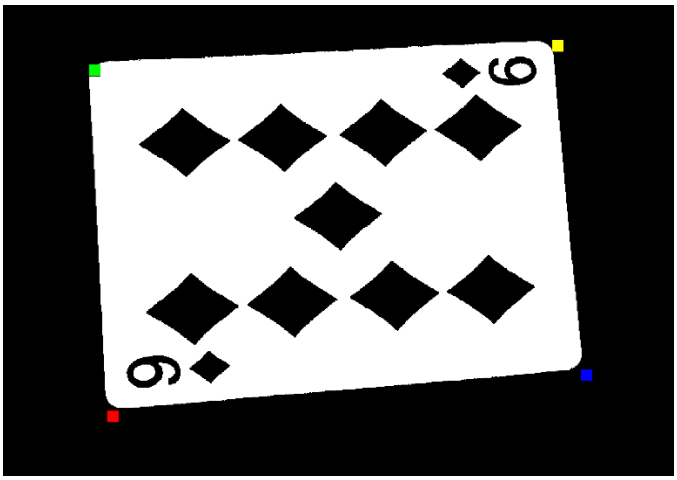

Figure 4: 9 of Diamonds with the corners detected and arranged. The red corner represents the first corner. The green corner is the second corner. The yellow corner is the third corner, and the blue corner is the fourth corner.

4) Transform creation

Using the four ordered corners in both the template and the target image, the homography is found that maps the corners on template onto the corners of the target. Equation (1) describes the linear system that uses the transform matrix A that we would like to find. From (1), (2) and (3) can be derived through matrix multiplication and normalization of the resulting vector so that the third element is a 1. Since we have 4 corners, and 2 equations per corner, this gives the needed amount of equations to find all the elements of the transform matrix.

$$x' = A\,x, A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & 1 \end{bmatrix},\ x = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (1)$$

$$x_i' = \frac{a_{11}x_i + a_{12}y_i + a_{13}}{a_{31}x + a_{32}y + 1} \quad (2)$$

$$y_i' = \frac{a_{21}x_i + a_{22}y_i + a_{23}}{a_{31}x + a_{32}y + 1} \quad (3)$$

5) Scoring

Using the transform found in the previous step, the area of the template image containing the card is mapped onto the target image. The sum squared difference is then taken for all pixels within the card to get the score of the match (a lower score is a better match).

In order to account for the fact that not all of the cards are rotationally symmetric, each template is scored twice. One time with no rotation and a second time with a 180-degree rotation. This allows corner arrangement to select either of the two numbered corners as the first corner despite the cards not being rotationally symmetric.

After the two scores are calculated, the lesser of the two is reported as the match score for the card and the template.

6) Repetition

The transform creation and scoring are repeated for each of the 52 pre-trained templates. The lowest score is selected as the best match and the template with the lowest score is then considered to be the correct identification.

7) Training

The template images for this approach were obtained as described in the Data Collection section. The Card Detection, Corner Finding and Corner Arrangement steps are performed on the training images, same as they are on the target image. The resulting corners and image are then stored in a database for use in identifying cards. This process is repeated for each of the 52 cards in a standard deck of cards to get a template image for each of them.

III. Results

A. SIFT Method Results

The SIFT method failed to produce any meaningful results. In some cases, it was able to correctly identify a single card, but would fail to do so reliably. This is suspected to be because of the highly symmetric nature of playing card features and is explored more in the Analysis section.

B. Template Matching Results

The results of the template matching approach for a few images can be seen in Table 1. To test the algorithm, one picture was taken for each suit, where the picture contained every card of that suit. So "13hearts" contained all 13 cards with the heart suit.

As can be seen from Table 1 the algorithm has a very high level of accuracy for images without perspective distortion. Of all 52 cards, 51 of them were correctly identified, which is approximately a 98% accuracy rate. In the "13spades" image, it was unable to correctly identify the King of Spades. While attempting to identify the card that is the King of Spades, an invalid transform was calculated, causing an error in the program. This is most likely due to an incorrect detection of the area of the image that contains the King of Spades.

Image	#Cards in Image	#Correct Identifications
13hearts	13	13
13diamonds	13	13
13clubs	13	13
13spades	13	12

Table 1:Results of Template Matching on all 52 cards

IV. Analysis

A. Why the SIFT approach failed

The foremost reason why the SIFT method may have failed to accurately identify cards is that it suffered from inaccurate feature matching. As can be seen in Figure 5, it was not uncommon for vl_ubcmatch to match multiple features in the training image to the same feature in the target image. This means that if N keypoints in the training image are matched to the same point in the target image, then at least N-1 of those matches are incorrect matches. Furthermore, it means that RANSAC often find that the best transform maps all points in the template onto a single point. This leads to an incorrect number of matching features and therefore incorrect detections.

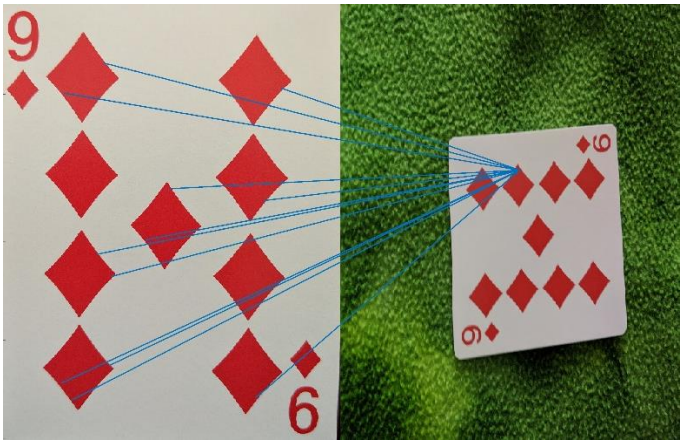

Figure 5:Example of vl_ubcmatch matching multiple features in the template to the same feature in the target image.

At one point, I attempted to compensate for this by preventing RANSAC from creating the homography with features that were matched to the same point. This failed to improve the results, possibly due to the fact that most of the points are incorrectly matched to begin with.

The reason why SIFT has this issue is because of the highly symmetric nature of playing cards. Playing cards often have the same symbol repeated multiple times on the card. Due to the symmetry within the card many features have similar feature descriptors. Furthermore, because SIFT aligns the feature descriptors by their dominant orientation, some feature descriptors (such as the corners of a diamond) can become almost impossible to distinguish from each other. An example of this can be seen in Figure 6. In this example, the top and bottom corner of the left-hand diamond are both matched with the bottom corner of the right-hand diamond. This is because when rotated to along lie on their dominant orientation, the top and bottom corners of the diamond look identical and have similar SIFT feature descriptors.

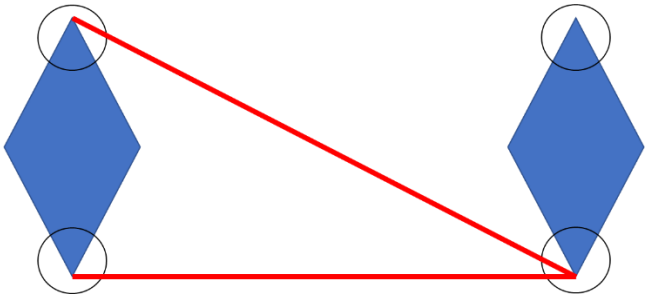

Figure 6: Example of how Sift features may be improperly matched.

B. Results of Template Matching

Template matching worked exceedingly well to detect and identify the cards. However, it is important to note that these results are in the ideal case. The performance of the algorithm in a non-ideal case is covered in Section V.

Despite the high accuracy of the algorithm it is not without its flaws. Overall, the template matching proved to be computationally expensive, taking upwards of a minute to identify a single card. The main expense is the scoring part of the algorithm, as it is run 52 times for every card (once for each template) and requires a matrix multiplication for every pixel that makes up a card. This is an area for further improvement and techniques such as subsampling can reduce this time, possibly at the expense of accuracy.

V. Non-Ideal Case: Perspective Distortion

So far, the images analyzed have had a top down perspective where the cards are not distorted. However, this is not always the case and it is of interest to examine the effects of perspective distortion on the accuracy of the algorithm.

An example of the template matching algorithm operating on cards with perspective distortion can be in Figure 7. The algorithm does not seem to encounter any difficulties with this level of distortion and is able to correctly identify the card as the Ace of Hearts.

However, for more extreme forms of distortion as seen in Figure 8, the template matching algorithm encounters an error as the transformation between the template and the target image is ill formed. This is most likely due to errors in isolating the card region properly, as the constants chosen for things such as minimum region size and thresholding may need to be changed for perspective distortion.

Figure 7: Image of a card with perspective distortion that is able to be correctly identified.

Figure 8: Image of cards with perspective distortion that could not be correctly identified by template matching.

Acknowledgment

Thank you to Jean-Baptiste Boin for his help on this project. He was instrumental in helping find the define the direction for the project.

Thank you to Dr. Girod for his clear explanations in lecture, they were very helpful when developing both approaches.

References

[1] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, vol. 60, no. 2, pp. 91-110, 2004. Available: 10.1023/b:visi.0000029664.99615.94.

[2] "VLFeat - Home", Vlfeat.org, 2019. [Online]. Available: http://www.vlfeat.org/.

[3] C. Zheng, R. Green, 'Playing Card Recognition Using Rotational Invariant Template Matching', Proceedings of Image and Vision Computing New Zealand 2007, pp. 276-281, Hamilton, New Zealand, December 2007.

[4] J. Pimentel and A. Bernardino, “A Comparison of Methods for Detection and Recognition of Playing Cards.”

[5] Martins P., Reis L.P., Teófilo L. (2011) Poker Vision: Playing Cards and Chips Identification Based on Image Processing. In: Vitrià J., Sanches J.M., Hernández M. (eds) Pattern Recognition and Image Analysis. IbPRIA 2011. Lecture Notes in Computer Science, vol 6669. Springer, Berlin, Heidelberg

[6] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography", Communications of the ACM, vol. 24, no. 6, pp. 381-395, 1981. Available: 10.1145/358669.358692.