***Playing Card Detection and Identification***

Münire Bacanlı 1, Gamze Dilmaç1,

1 Department of Computer Engineering, Eskişehir Technical University, Eskişehir

munirebacanli@eskisehir.edu.tr, gamzedilmac@eskisehir.edu.tr

*Abstract*—In this project, we explained how to detect and identify images of playing cards(standard 52-card deck). The project is created by first detect the location of the pictures in various sizes and directions and then matching the images with different templates. output is images, which are predicted with high accuracy. *(Abstract)*

Keywords-detection; identification; playing cards; templates; (key words)

# Introduction *(Heading 1)*

It is difficult to detect and identify playing cards in a correct background and in the right direction. For these stages, the correct algorithms should be chosen and applied. 58 images used in our project. Each image shows a different playing card. Each image is at different angles and in different directions. In order to match these cards with the correct methods, we divided them into two folders as rank and suit. Then we aimed to reach the images in datasets and use correct image processing techniques. We used the MATLAB platform when we performed all these steps.

# ALGORITHM

Our algorithms implemented the following processes:

1. Image Resizing

2. Binary image thresholding

3. Crop and Orientation image

4. Template matching

## Image Resizing

We used this method to equalize the size of our images due to the different dimensions of the images we use in our project. We enlarge our images that are below a certain size.

*B.. Binary Image Thresholding*

Some of the first steps we need to consider related to image processing in Matlab are the conversion to gray level and binary level. Binary thresholding was performed using Otsu’s method. It determines the brightness threshold of the image and then converted into binary.

## Crop and Orientation Image

We cropped the images according to the regions we set, and then resized again. We determined the orientation of the images and make calculations according to that angle. Then we rotated the cards at the right angles. In this way, card was detected.

## Template Matching

Card identification was achieved using template matching.

# Implementatıon

First of all, we created our ranks and suites template that we will be using for 'template matching'.

Then, it was necessary to reduce the size of the image. The dimensions of the images in our dataset are close to 3000\*3000 in general. Since we want the program to run faster, we tried to reduce the size of the pixel it was reading, and firstly, we reduced the size of the image to half the size by using the bilinear interpolation[[1]](#footnote-1). If the size of the image being read is smaller than 1500 \* 1500, then enlarged the size of the image by 3 times with the 'bilinear interpolation function again. The reason we're doing this is because some of the images in the dataset are very small. We tried to find these small images and make them the same size as other images, so we tried to reduce the margin of error.

Since some images are rotated in our given dataset, we had performed the rotation process to make the images straight.

Then we loaded our templates and went through pre-process separately for rank and suit.

In template pre-process, we made resize templates.

Using the Otsu's Threshold method[[2]](#footnote-2), we converted the templates binary level. Otsu’s Threshold [1] method determine the best efficient threshold related to image and then use this threshold to make the image binary. We did this steps also the uploaded image.

After the image and templates come to the shape we want, we started processing each card. First we detected the card and then tried to identify what that detected card is by using template matching. [2]

After the pre-process step applied to both image and templates,we divided image into labels. Each label of the card is exposed with the 'regionprops'[[3]](#footnote-3) method. It takes 'convexhull', 'area', 'centroid', 'orientation', 'boundingbox' as parameters. In this way, it extracts the features of the region like area, orientation, center points in the area etc.

If the area of the region is smaller than 70000, we made it perceive it as a 'noise' and bypassed it and look at the other label. The number of 70000 is actually the approximate total pixel number of each image. In this way, we prevented the detection process from perceiving non-object locations as objects. With BoundingBox, we take the smallest box containing the area and use it for cropping. Before proceeding to the template matching process, we wanted the image and templates to be as close as possible to each other in size. So we did the resize process again.

We rotated the image again. After rotating, we cropped the card again and calculated the top, bottom row and left, right column of the card. Since the smallest value represents the highest point of the image and the largest value means the lowest point of the image, we made the adjustments accordingly:

*topRow = min (rows);*

*bottomRow = max (rows);*

*leftColumn = min (columns);*

*rightColumn = max (columns);*

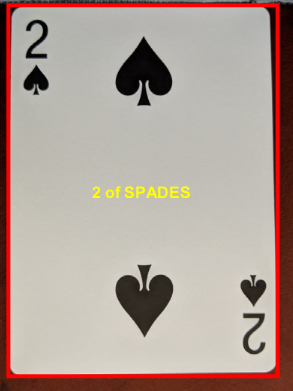
*croppedImage\_og = uprightImage(topRow:bottomRow, leftColumn:rightColumn);*

And this way the card was detected. Time for the identification.

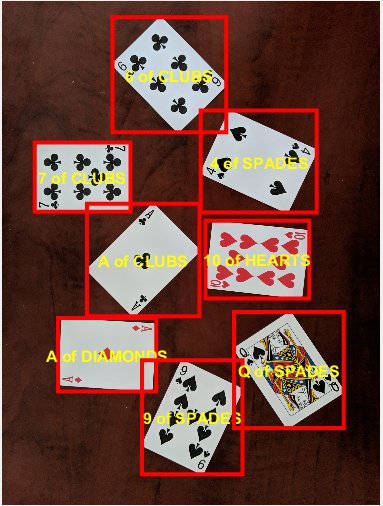
To match the image with the template, we take the top-left and bottom-right parts of the image as an angle. We set a size\_template to get the small and large suit in the label and then we started the template matching process.

Here, in general, using the normxcorr2[[4]](#footnote-4) function, there is a similarity ratio between the image and the template (separately for suit and rank). If this ratio is more than 90 percent, we concluded that the suit and rank was found. After the template matching process, we found the middle point of the card we detected and printed what the card is in the middle point.

At the same time, we framed the detected object and displayed it.



*Figure 1*



*Figure 2*



*Figure 3*

CONCLUSION

The playing card identification and detection project included many image processing steps. We tried to choose the most appropriate steps in our project and completed our project. In our testing, we found that our application identifies cards with an accuracy of % 89. As a result, we can accurately identify and detection the very high proportion of images given to us in our project.

REFERENCES

[1] Zhang, Jun, and Jinglu Hu. "Image segmentation based on 2D Otsu method with histogram analysis." *2008 International Conference on Computer Science and Software Engineering*. Vol. 6. IEEE, 2008.

[2] 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.

1. <https://en.wikipedia.org/wiki/Otsu%27s_method> [↑](#footnote-ref-1)
2. [↑](#footnote-ref-2)
3. <https://ch.mathworks.com/help/images/ref/regionprops.html> [↑](#footnote-ref-3)
4. <https://ch.mathworks.com/help/images/ref/normxcorr2.html> [↑](#footnote-ref-4)