COMS 4036A: COMPUTER VISION

# BLACK J TOURNAMENT

Bill Seota: 1043377
Francis Mudavanhu: 1233925
School of Computer Science and Applied Mathematics

October 23, 2018

**Abstract**

Blackjack has never been an easy game for machine vision models to play. The core of this concerning issue lies in the machine's ability to localise and identify the cards that it has been dealt, while ignoring everything else. For card detection, we fit all contour parameters using a normal distribution. For the same purpose, we regress to manual tweaking of some parameters to accommodate the feature detectors used: SIFT, SURF and ORB. The game logic is defined, and an optimal machine is chosen based on the findings between the methods surveyed. The authors believe that this cutting-edge paper will be trailblazing in the approaches to methodology of detection and identification of cards in an image, taken at very particular angles at a lecturer or professor's desk.

# Contents

# Introduction

The aim of this project is to survey a series of methods and their combinations for playing Blackjack, with the hopeful goal to identify an optimal model. The rules are specified in [1]. Various methods are surveyed and are put through a series of tests to determine if they perform well enough to form part of the the strongest model. The optimal model will then be used as a weak[1] proxy of the models considered[2].

[1] Weak because it does not capture the number of litres of blood, sweat and tears poured into the project
[2] Read this report like a story :)

# Methods

The methods used at each step of the project were motivated by how well subsequent steps involved in compiling the machine could handle the images in the build-up to card identification and playing of the game (POG). The images [2] used consisted of 228 1920x1440 images taken under various lighting conditions and angles.

## Model 1: Card Detection 1 (CD-1)

Contours [4] were used as a means of extracting the cards needed for valuation: they were separated from the rest of the image using probabilistic methods defined below. The training images comprised of masks that were used as labels for the position of cards in the image. The properties of contours we used [3]:

1. Contour Length
2. Contour Area
3. Contour Angle against horizontal plane (orientation)

These three properties were trained for noisy images (with background contours that we didn't need) as well as for the mask images with properties of contours we wished to extract from the any image during POG. *Card* was defined as any contour enclosing cards that were part of any game at any point in any image, while *Background* was taken to be any other contour.

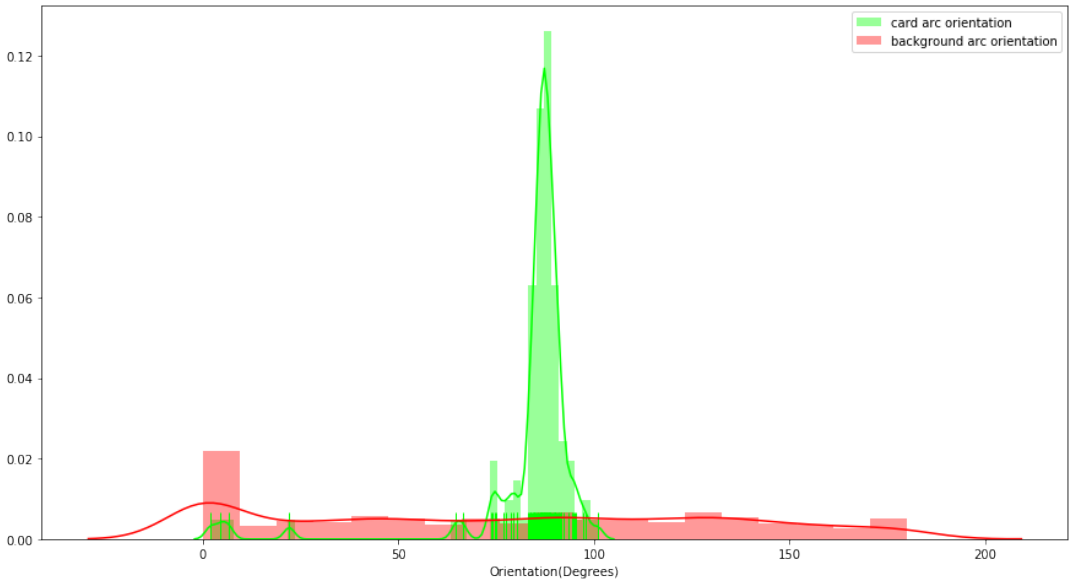


**Figure 1:** Angle of card contour (tall with small standard deviation) vs angle of background contours (flat with large standard deviation)

The following summary statistics were computed:

**CARD: 283 contours**

| | **Length (Px)** | **Area (Px)** | **Angle (Px)** |
|---|---|---|---|
| **Mean** | 624.37 | 20561.53 | 82.66 |
| **Standard deviation** | 144.70 | 7973.68 | 14.45 |

**Table 1:** Card summary statistics with small standard deviations

**BACKGROUD: 5294 contours**

| | **Length (Px)** | **Area (Px)** | **Angle (Px)** |
|---|---|---|---|
| **Mean** | 137.86 | 6058.62 | 88.66 |
| **Standard deviation** | 720.34 | 49880.34 | 51.78 |

**Table 2:** Background summary statistics with very large standard deviations

The separability of the distributions between the card and background contour orientation are shown in *figure 1*.
As demonstrated by *figure 1*, it was by no mistake that we chose the separability of arc angle, for instance, as one of our features. The length and area separability can be inferred from the summary statistics in the above table.
Once the card and background properties were found, we did a hard classification of *card* vs *background* contours: We tried different ranges and *if* statements that would partition the two classes.
The hard classification of contour types (card and background) proved a to be a great deal of manual labour; so we used a normal distribution to fit independently the same parameters; the *angle* parameters produced the normal distribution in *figure 1*. However naive, we assumed independence, then used a Naive Bayes [8] classifier to identify the contours which enclosed cards, and those which did not. Here, the probability of a contour being part of a binary class (card or background) was conditional on the parameter values obtained from the normalised product of the three independent distributions (length, area, angle).

## CD-1 Findings and Issues

From the outset of the experimentation phase, it became quickly clear that there would be an issue with defining the region of interest. details on how that affected our assumptions, models and adaptations thereof follow:

1. Single card and stacked card detection: *CD-1* worked perfectly for the detection of singular cards and stacked cards that were part of the game. That is, its precision

$$\frac{TP}{TP+FP} = 1, \tag{1}$$

and recall

$$\frac{TP}{TP+FN} = 1, \tag{2}$$

where TP stands for the number of contours drawn over appropriate cards (cards that part of the game), FP stands for the number of contours drawn over background objects and FN stands for the number of appropriate cards which had no contours enclosing them.

2. Separately dealt cards: *CD-1* worked poorly where it had to detect cards that were placed near each other It produced a larger false negative and false positive rate). This was, in part due to the distance from the camera. To solve this *region of interest* issue, We went as far as fitting for ***top, bottom, left, and right-most*** points of the cards; but without much happiness in the end. The precision score after this modification became far less impressive. In fitting these four latter parameters to a normal distribution, rather than tweaking the regions via trial-and-error, we hoped to achieve a softer and perhaps more sound learning method for finding cards within a region.

   Before wiping the sweat invested in this back-stabbing model, we tried our last lifeline: Add all types of noise distortion to the background training images and hope the model would pick up skewed means and disgusting variances. This should have worked better. It did not.

   Clinging on to the model, adaptations to the *CD-1* were made so that it worked for the separate images. This worked with a close-to-one precision rate, but not for free. The the adaptation reduced dramatically its perfection in detecting stacked and singular cards. This trade-off really seemed like it was here to stay, so we went back to the brute force tweaking of regions of interest.

We later abandoned the *CD-1* model.

## MODEL 2: HISTOGRAM OF ORIENTED GRADIENTS WITH SUPPORT VECTOR MACHINES

The Histogram of Oriented Gradients (HOG) model was used to learn the number of occurenes of gradient orientations between the pixels of an image, and label these points as edges. The distribution of various gradient magnitudes and directions were used to segment regions (similar-looking regions would have similarly distributed magnitudes). These distributions were fed to the Support Vector Machines classifier.

*Model 2* did not work as we would have liked, since the distribution of the card distances from the camera in different images is not uniform (i.e the HOG as a descriptor does not capture scale very well (compared to, say, SIFT described in the final model).
We shelved this classifier method but extracted interesting elements from the HOG which were used, discussed in the SIFT model.

# MODEL 3: FEATURE DETECTION ALGORITHMS

Like with *CD-1*, contours were also the ball of the game; but these contours were used a little differently. Since a sift descriptor is invariant to scale and multiple kinds of translations, the fitting done on the contours in CD-1 were hypothesised and proved to be less useful.
Details on the use of contours, transformations and preprocessing in card identification are given below.

## Contours

1. Length and area [3]: These were specified as the maximum and minimum contour lengths and areas we felt would give us a sound card detection system. We found that the average card contour length within what we deemed to be our region of interest was roughly 614 pixels.

2. Order points [5]: Order points were extracted from images so that the proceeding affine transform would produce the appropriate orientation.

3. Manually-chosen centroids: In order to reduce the false negative and ipmrove the true positive rate, we sought to create three clusters based on region of interest (ROI) for each image:

   (a) Cards that are part of the game

   (b) Far background

   (c) Near background

   For *a)*, Centroids near the centre of the image were placed with radii drawn around them. We found that doing so reduced the number of false detections. A centroid was placed in *b)* and *c)*, to ensure that any objects too far or too near were ignored by the descriptors.

### Preprocessing and Transformations

1. Gradients: As promised, we now discuss how HOG [6] helped reveal some insights into interpixel gradient parameters (magnitudes and directions). The gradient parameters between each pixel were computed using a Sobel edge detection filter. Then, grayscaling and gaussian blurring were used to smooth the resultant image; reducing its level of noise. This allowed for only the key contours to be drawn. That is, the microscopically small contours would not be drawn over objects that are negligible, in the idenification process.

2. Affine transform: This function's output had the card orientated up-side up. The Affine transform [7] was used in order to feed as input for the extraction of key points done by the SIFT detector.

## CARD IDENTIFICATION

Using the valid contours identified for each image, we commenced the feature matching process.
If contours are the ball of the game, then we would say that feature detectors are the game's pitch. As such, three feature detectors were used and each exhibited a wide range of results.

### Card Identification using ORB and SURF

Both the ORB and SURF descriptors for card identification were explored mainly for experimentation. Conducting this part of the project helped us survey the efficiency and effectiveness of each of the three methods discussed below. The ORB (Oriented FAST and Rotated BRIEF) feature detection method over all images returned a true positive rate of 27%. With SURF (Speeded-Up Robust Features) feature detector, we only managed an accuracy of 33% with the preprocessing done up to that point. We suggested that this accuracy was a little low, and would remain low even with a little more preprocessing. One cause of the poor accuracy in both could have been major mismatches between labeled key-points and those in validation images caused by sporadic correlations, as shown in the *figure 2*.

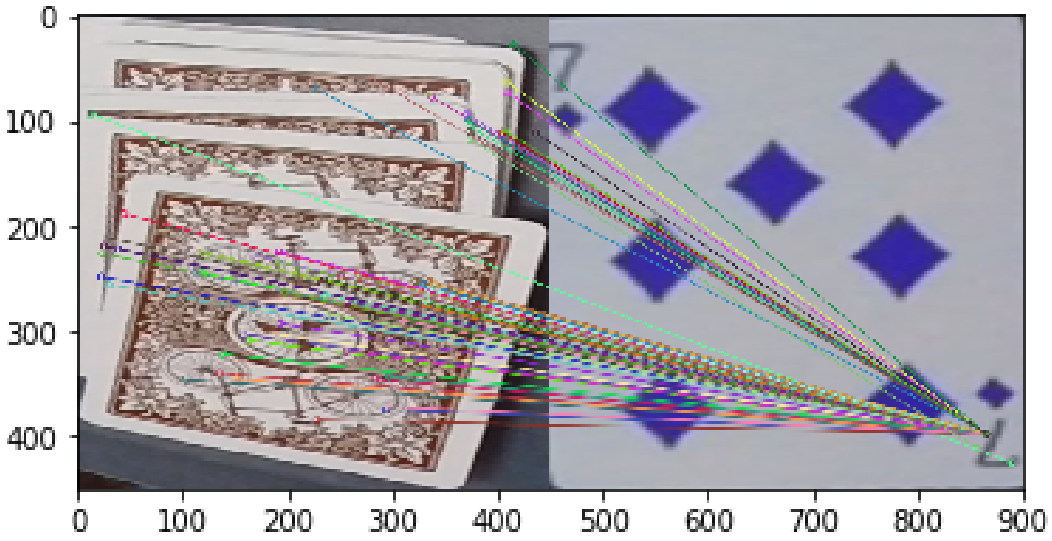


**Figure 2:** The right hand side shows key-points extracted by the descriptor in the training phase. The left hand side shows the (mis)matched key-points identified by the descriptor.

## Card Identification using SIFT

SIFT (Scale Invariant Feature Transform), like SURF, describes local features in an image. With SIFT, we found that considerably more key-points were being matched to the appropriate cards than in SURF or ORB; especially for the Jack, Queen and King cards.

Trade-off between true positives and false negatives.. Having a high number of specifid key-points.

Using the above preprocessing methods, the model managed to achieve an identification (true positive) rate of 77%.

To avoid the false positive issue demonstrated by *figure 2*, we used contour moments (we will describe how in a *moment*).

Some of the biggest issues underpinning the model up to so far:

- Agressive responsiveness: This happens when more cards are recognised than were actually part of the game(many false positives). Probable cause: We may have too specified too many card key-points as hyper-parameters to the SIFT model.
- Unresponsiveness: This happens when cards were being incorrectly identified as other cards (false negatives), or some cards part of the game not being recognised at all. Probable cause: Not enough key-points specified.

## GAME LOGIC

The greatest exploitation of the data (or game) that we used was that the sequence of images were taken from sequential frames of a video. This became the strongest requirement of the game for the machine to work. The aim of the model was thus to determine the state of the game, or *(value,hand)* pairs that describe the state, where *value* represents the value of the cards in the frame and *hand* distinguishes between a *winning* or *losing* hand.
For games that did not end in an empty frame, we added our own empty frame; as we believed that this was a true representation of casino standards and regulation (really, this was for our convenience).
Moments of contours were also used here to determine the centroids of the card contours. These centroids were used to determine the distance from each contour, and furthermore and identify whether that contour's card was in the set of cards in the previous frame of the game.

## RESULTS

With our best (SIFT) model against our validation set of 122 cards, 156 cards were detected, 94 of which were true positives, leaving $122 - 94 = 28$ cards that were false negatives. $156 - 122 = 34$ cards were false positives.

## REFLECTION

A fair criticism of our approach would be that we could have used a multivariate normal approach to capture the correlation (or dependence) between the properties in *CD-1*; or, better yet, use a Gaussian mixture mode to fit its parameters. Thereafter, we would be able to amalgamate that detection model with the *SIFT* descriptor model. This would have resulted in higher precision scores in detection and identification.
Instead of using manually-chosen centroids described in the *Model 3* section, we could have explored the clustering method surveyed in [9]: Determining contour clusters using dynamic distances between cluster centroids and other contours. These centroids would be learned by a KMeans with a K=3 for instance and the centroids would be

1. Cards that are part of the game
2. Far background
3. Near background

Our model relied heavily on the fact that the frames were in the same sequence in which the game was played. As a result of using this to our advantage it also causes a plausible limitation to our model. A more robust model would be one that finds all cards within a given frame and determine the *(value, hand)* pairs at any given time, regardless of preceding frames or steps.

## CONCLUSION

We aimed to explore methods of card identification and select the most robust one during POG. For detection, we explored the use of Gaussian models and a Naive Bayes classifier. For detection and proceeding identification; contour specification, HOG with SVMs, ORB, SURF and SIFT. We found that given our model specification, the SIFT classifier performed best.

# List of Figures

# List of Tables

# Bibliography


[1] Blackjack, howpublished = https://wizardofodds.com/games/blackjack/basics/, year=2018, note = Accessed: 2018-09-15.

[2] COMS4036A - Computer Vision - 2018 , howpublished = https://moodle.ms.wits.ac.za/moodle/course/view.php?id=8, year=2018, note = Accessed: 2018-07-15.

[3] Contour Properties, howpublished = https://docs.opencv.org/3.4/d1/d32/tutorial_py_contour_properties.html, year=2018, note = Accessed: 2018-09-15.

[4] Contours : Getting Started, howpublished = https://docs.opencv.org/3.3.1/d4/d73/tutorial_py_contours_begin.html, year=2018, note = Accessed: 2018-09-15.

[5] Ordering coordinates clockwise with Python and OpenCV, howpublished = https://www.pyimagesearch.com/2016/03/21/ordering-coordinates-clockwise-with-python-and-opencv/, year=2018, note = Accessed: 2018-09-15.

[6] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[7] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International journal of computer vision*, 60(1):63–86, 2004.

[8] Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46. IBM New York, 2001.

[9] Xiao Zhang. Dynamic based contour clustering. 2014.