COMS30121 - Image Processing and Computer Vision
The Dartboard Challenge

Joshua Van Leeuwen & Karim Allaouat

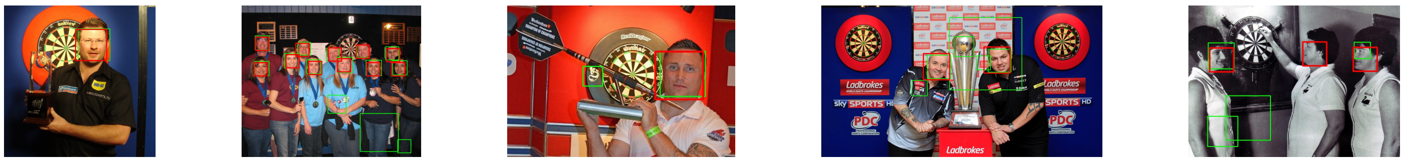
Introduction

This task introduces the ability to detect and locate instances of an object class in images. This is important as this ability is used in many computer vision applications. The task explores the Viola-Jones object detection framework (an "off the shelf" face detector) and combines it with other detection techniques to improve it. The image set used is from the popular sport, darts.

The Viola-Jones Object Detector

The Viola-Jones object detection framework is the first object detection framework to provide competitive object detection rates in real time. The algorithm was used with a strong classifier trained using AdaBoost for detecting human faces from the front.

Using the Detector on Human Faces



(a) dart4.jpg. (b) dart5.jpg (c) dart13.jpg (d) dart14.jpg (e) dart15.jpg

Figure 1: Result of Viola-Jones Algorithm on Human Faces.

Assessing How the Detector Performs

The TPR or True Positive Rate measures the proportion of relevant items that are correctly identified. In this case it is the fraction of successfully detected faces out of all valid faces in an image. The TPR of dart5.jpg and dart15.jpg are 100% and 67% respectfully.

A practical difficulty of computing the TPR accurately is that the hits and misses have to be manually counted. Also errors can occur when faces are side profile because they become ambiguous as to whether they are valid. It is always possible to get 100% TRP because you can detect everything in the image and so will always get all possible hits. It will however, get all the misses too. A better way of evaluating the detector would be to calculate the F_1 score. It takes into account the detectors precision (PPV - Positive Prediction Value, how many selected items are relevant) and recall (TPR). A set of rules were created to evaluate whether a face was valid:

- Two eyes and a mouth must be within a boundary to be counted as a hit.
- Two eyes and a mouth must be visible to us in order for it to be counted as valid.
- The F_1 score will be calculated by:

$$\frac{2 \times R \times TPR}{TPR \times R}$$

Where

- True positive rate $TPR = truepositives/(truepositives + falsepositives)$.
- Recall $R = truepositives/predictedpositives$.

TODO: Talk about how we are going to calculate the F1 score automatically by using comparisons of pre determined boxes and comparing the distance (and size?).

Building and Testing the Detector

Interpreting TPR vs FPR

Figure 2 shows the training of the detector over the 3 stages. The TPR always remained as 1 therefore, it was successful in detecting all dartboards. The decreasing FPR portrays that the detector firstly detects as much as it can, then reduces the number of objects it detects. As a consequence, it is clear that the detector is improving.

TODO: Talk about Parameter tuning. Used 500:1000 Positive to Negative ratio and a 0.4 max false alarm rate.

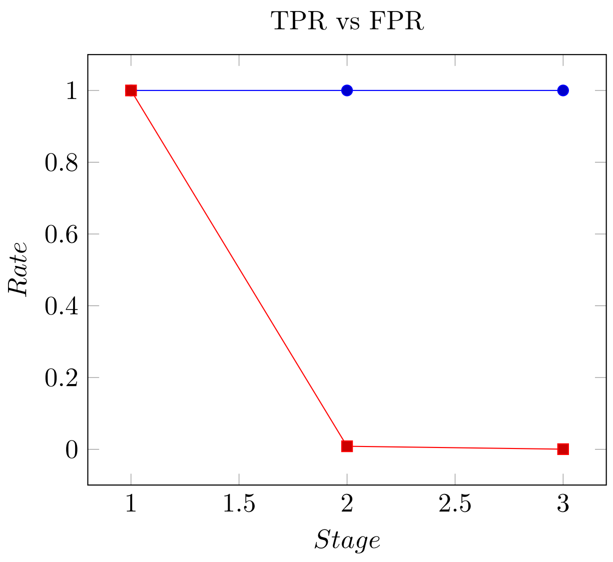
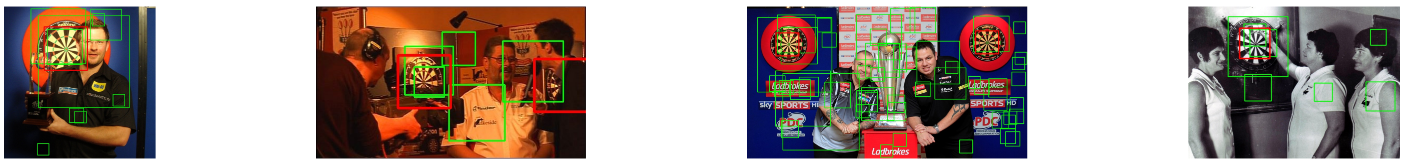


Figure 2: TPR(blue) vs FPR(red) across the 3 stages.

Testing on images



(a) dart4.jpg. (b) dart5.jpg (c) dart13.jpg (d) dart14.jpg

Figure 3: Result of the trained dartboard detector.

TODO: make this look beter? The F_1 of the images are:

- dart0.jpg - 0.14.
- dart1.jpg - 0.13.
- dart2.jpg - 0.12.
- dart3.jpg - 0.20.
- dart4.jpg - 0.20.
- dart5.jpg - 0.10.
- dart6.jpg - 0.17.
- dart7.jpg - 0.09.
- dart8.jpg - 0.13.
- dart9.jpg - 0.13.
- dart10.jpg - 0.11.
- dart11.jpg - 0.29.
- dart12.jpg - 0.33.
- dart13.jpg - 0.14.
- dart14.jpg - 0.07.
- dart15.jpg - 0.25.

As the F_1 score was calculated by the third rule and the TPR is 1, the new equation becomes:

$$\frac{2 \times hits}{detected + hits}$$

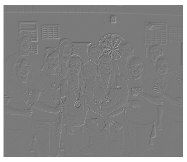
. Also, the F_1 score is relatively low therefore there denominator is much bigger and can conclude that the was a high number of detections. This means that there were a lot of misses. The usefulness of the plot (Figure 2) is... I dunno..

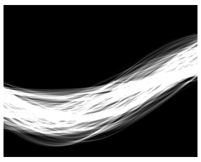
Not sure we should use a new formula. We are using our own code to test if there is a hit.

TODO: Talk about how it was shit but we can use the under fitting to out advantage to combine with other classifier detectors.

Integration with Shape Detectors

Image Results


(a) Threshold Gradient Magnitude.


(b) Hough Space

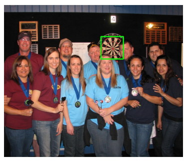
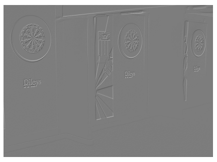
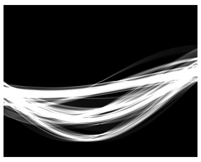

(c) Result

Figure 4: dart5.jpg shows the merits of the detector.


(a) Threshold Gradient Magnitude.


(b) Hough Space

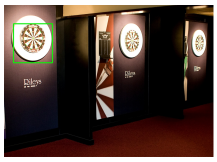
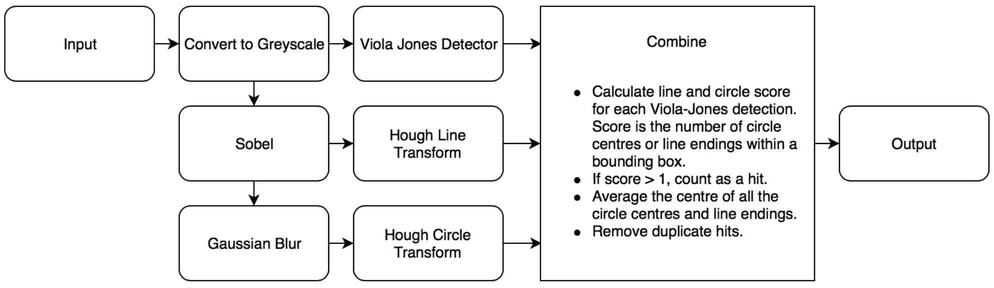

(c) Result

Figure 5: dart10.jpg shows the limitations of the detector.

Merits and Limitations

Our new dartboard detector did considerably better than the previous, achieving an overall F_1 score 0.767 with the previous being $0.163 == \textit{checkthis}$. This detector adds more classifiers when analysing images meaning that the large set of detections with many negative hits, is able to be reduced by combining each classifier result. This detector works optimally with images where dartboards are in good lighting and are facing straight at the camera in the scene. As shown in the dart10.jpg image, the detector failed to detect two dartboards that are at an angle. This is because of the circle and line Hough transformation being used, struggle to detect these. TODO: Talk about how we get the line detections - intersections etc.

Combination of Detectors


Input
Convert to Greyscale
Viola Jones Detector
Sobel
Hough Line Transform
Gaussian Blur
Hough Circle Transform
Combine
Calculate line and circle score for each Viola-Jones detection. Score is the number of circle centres or line endings within a bounding box.
If score > 1, count as a hit.
Average the centre of all the circle centres and line endings.
Remove duplicate hits.
Output

(a) Flowchart representing the detector.

As shown, the Viola Jones detector results have a positive detection of every dartboard in each image however have a large amount of false positives. Our approach was to create new classifiers which would refine these hits down to true positives by accepting Viola Jones hits that were also observed by our line and circle detectors. This involves iterating through the set of Viola hits, comparing whether any line or circle hits are contained in the Viola bounding box, accepting if so, rejecting otherwise. If accepted, this hit would change its location based on the average position of itself, along with its combined detections. With the use of the circle detections, this also gave us the ability to estimate the size of the dartboard also, taking the average radius of included circles and including this in the approximation. Once we had achieved the new set of accepted detections we recognised that multiple bounding boxes overlapped with one another. We thus combined all bounding boxes that were overlapping, again averaging the coordinates and size, ensuring one positive hit per dartboard.

Improving the Detector

- ellipses
- find a better viola jones classifier
- adjust the parameters to fit better
- triangles
- make out3 brighter??
- otsu

References

Picture	Actual	Detected	Hit	Missed	F_1 Score	
dart4	1	1	1	0	0	1
dart5	11	14	11	0	3	0.88
dart13	1	2	1	0	1	0.67
dart14	2	6	2	0	4	0.5
dart15	3	4	2	1	2	0.57

Table 1: Comparing the F_1 Score of different images.

TODO: make this bullet points

In order to improve our dartboard classifier, we considered further shapes which help to classify a dartboard being present. As such, we identified two shapes to consider - ellipses and triangles. We realised that some dartboards had been not been identified by our classifier due to not being detected by circles with dartboards that are at an angle. By using ellipse shapes we would be able to capture this shape of valid dartboards. We also chose to consider triangle shapes inside images to detect dartboards. This is because all dartboards have distinct triangles contained that will give more detections that can be combined in our detector.

Detecting both triangle and ellipse shapes is achieved by first applying the Canny edge detector on the grey scale input image. The Canny edge detector, developed by John F. Canny in 1986. [add citation] and works by calculating the gradient magnitude of each pixel. To determine whether each pixel is part of an edge two thresholds are applied, pixels higher than the larger threshold are considered pixels on an edge, pixels below the lower threshold are discarded and pixels in-between the two thresholds are considered edges only if they are connected to a pixel that is above the higher threshold. In order to select an appropriate threshold for each image, the Otsu method can be applied. [https://en.wikipedia.org/wiki/Otsu

The Otsu method is used to determine the largest threshold input for the Canny edge detector. TODO: Talk about how the method works. The lower threshold value is one third of the returning value from the Otsu method.

If a pixel gradient is higher than the upper threshold, the pixel is accepted as an edge If a pixel gradient value is below the lower threshold, then it is rejected. If the pixel gradient is between the two thresholds, then it will be accepted only if it is connected to a pixel that is above the upper threshold. Canny recommended a upper:lower ratio between 2:1 and 3:1.

F1 score = .8666667333333333333333

TODO: talk about surf We use a scoring system with surf We cluster surfs together togther that count as the same object