The Object Detection Challenge

Aikaterini Baous

ab16651@mybristol.ac.uk

Ankit Seal

as16877@mybristol.ac.uk

I. Introduction

This report features an object detection algorithm designed using OpenCV in C++. In this report, an analysis and discussion on the Viola-Jones object detection framework is presented by using mathematical models. It also explores and implements various approaches that can be used to improve the detection process of Viola-Jones framework."

II. First Task: The Viola-Jones Object Detector

A. *Running the code for a selection of images*

B. *Interpreting the results*

The TPR is calculated using the following formula:

TPR= TP/(TP+FN)

True Positive (TP) → number of correct matches

True Negative (TN) → non-matches that were correctly rejected

False Positive (FP) → proposed matches that were incorrect

False Negative (FN) → matches that were not correctly detected

Applying the equation for dart5.jpg and dart15.jpg we get:

→ dart5.jpg TPR=11/(11+0)=1

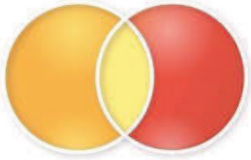
→ dart15.jpg TPR=2/(2+1) = 2/3 =0.67

1. The true positive rate does not always provide the most accurate parameters for assessing the performance of the Viola-Jones detector. To access the TPR in a picture successfully it is necessary to form a set of features that would be invariant to geometric (i.e. rotation) and photometric (i.e. brightness) transformations.
2. It is always possible to achieve a TPR of 100% since this measure isn't sensible to the false outcomes of the detection algorithms. The biggest drawback of the TPR function is that it doesn't account for false positives. For instance, if the features we want to detect are not strictly defined or if the data set describing these features is not big enough, the algorithm will detect the faces and it will also detect other objects in the image that are not faces. As observed from the image, dart5.jpg, it is possible to achieve a 100% TPR even though it had detected a few False Positives. In that case the TPR will be 100% because the detection algorithm would have detected all the faces in that picture even though the FP rate will be high
3. The measure that combines the precision and recall is the harmonic mean of precision and recall, the F_1-score:

   F_1=2*(precision*recall)/(precision + recall)

   Precision= TP/(TP + FP)

   Recall=TP/(TP + FN)



With the previous figure we represent:

- A (orange set) = Relevant Objects
- B(red set)= Retrieved Bounded Boxes
- A∩B= True Positives

With that being said we can calculate:

Precision= |TP|/|B|

Recall=|TP|/|A|

These types combined with the type for F1-score allow us to calculate the F1-score for dart5.jpg:

Precision= 11/14 Recall=11/11

F_1= 2 * (11/14)/(11/14+1)= 22/25= 0.88

III. SECOND TASK:TRAINING CASCADE FOR DART BOARDS

A. Training Procedure:TPR vs. FPR diagramm

HR and FA stand for hit rate and false alarm. Conceptually, HitRate = % of positive samples that are classified correctly as such. FalseAlarm = % of negative samples incorrectly classified as positive.

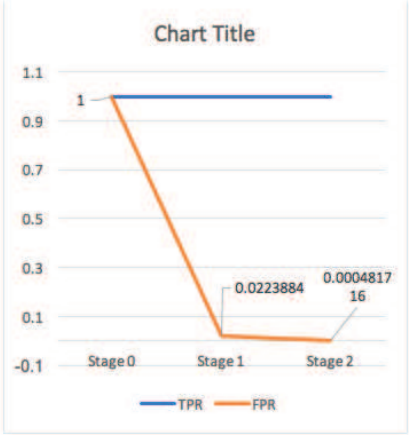


Figure 3- graph plotted for the TPR and FPR obtained at different stages of the training process.

By analyzing the graph, we observe that at the TPR remains constant at 1 (1000:1000), whereas the FPR steeply declines down from 1 (1000:1000) to 0.00048 (1000:0.000481716) over the three stages of training. Essentially, a TPR of 0.99 and a FPR of 0.05 means that any stage must detect 99% of the positive samples as such while its allowed to classify 50% of the negative samples as positive. At stage 0, it initialises by identifying all images as dartboards. As more Haar-like features are learned using AdaBoost, the FPR decreases dramatically, thereby increasing the accuracy of the classifier.

B. Abbreviations and Acronyms

Figure 4-Dartboard Candidates

The results for the F1-score of all 16 dartboard images are the following:

- F1(dart0.jpg)=0.285714
- F1(dart1.jpg)=0.666667
- F1(dart2.jpg)=0.285714
- F1(dart3.jpg)=0.5
- F1(dart4.jpg)=0.571429
- F1(dart5.jpg)=0.285714
- F1(dart6.jpg)=0.4
- F1(dart7.jpg)=0.266667
- F1(dart8.jpg)=0.315789
- F1(dart9.jpg)=0.444444
- F1(dart10.jpg)=0.285714
- F1(dart11.jpg)=0.857143
- F1(dart12.jpg)=0.857143
- F1(dart13.jpg)=0.363636
- F1(dart14.jpg)=0.297872
- F1(dart15.jpg)=0.666667

All the images, for which the F1 score was calculated, have a TPR of 1. A TPR score of 1 implies that all dartboards were identified accurately, which in turn implies that the classifier is working as excepted. But, the variation in F1 score implies that the detection was not precise and the classifier is not well trained.

IV. ANALYSING THE CODE RESULTS

A. *Images retrieved from the code execution*

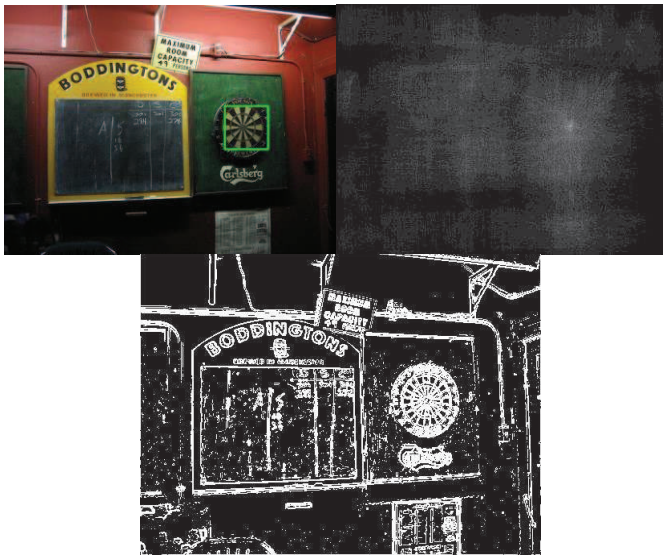


Figure 5-Output retrieved for image dart3.jpg

Figure 6-Output retrieved for image dart9.jpg

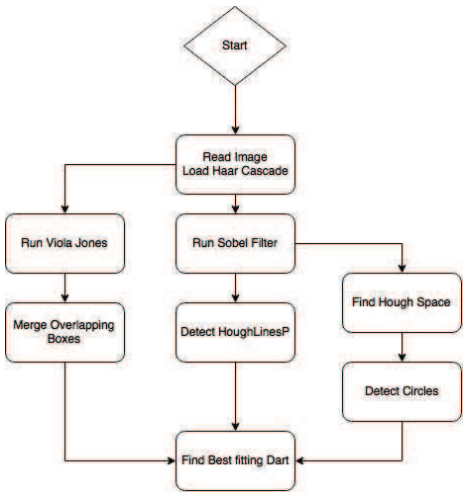
B. *Providing the F1-score for the images*

Implementing the combination of the Viola-Jones detection and the Hough Circles and Hough Line detection really intriguing. In our code we successfully managed to reduce the number of bounded boxes created by the Viola-Jones detection algorithm by presenting only the bounded boxes that contained lines and circles due to the appearance of the dartboards. We managed to achieve high line concentration inside the dartboards and therefore it was easier to distinguish the most interesting part of the pictures and limit the bounded boxes/results that are contained inside this area. In order to create even more accurate bounded boxes we searched for the ones that not only contain lines but also the circles produced by Hough Lines.

During this implementation we noticed that if the image is poorly lighted the detection becomes really difficult and the result contains many false positives. Furthermore the code requires a lot of time and memory to run ,especially in the case of the Hough Circle detection algorithm.

The following table and picture show the F1-score of our results and the flow chart of the algorithm:

Image	Recall	Precision	F1
Dart0.jpg	1	1	1
Dart1.jpg	1	2/3	4/5
Dart2.jpg	1	1/2	2/3
Dart3.jpg	1	1	1
Dart4.jpg	1	2/3	4/5
Dart5.jpg	1	1/3	1/2
Dart6.jpg	1	1	1
Dart7.jpg	1	1	1
Dart8.jpg	1	1	1
Dart9.jpg	1	1/3	1/2
Dart10.jpg	1	4/5	8/9
Dart11.jpg	1	1	1
Dart12.jpg	1	1	1
Dart13.jpg	1	1/3	1/2
Dart14.jpg	1	7/13	14/20
Dart15.jpg	1	1	1
AVERAGE			0.75



V. IMPROVING THE DETECTOR

As observed previously, by incorporating hough lines and hough circles, the code had improved a lot. The accuracy of the results though can further improved. To do so, this section incorporates another feature detector called the SURF. SURF: Speeded Up Robust Features" which introduced a new algorithm called SURF. It is an algorithm which extracts some unique key points and descriptors from an image. SURF uses an intermediate image representation called Integral Image, which is computed from the input image and is used to speed up the calculations in any rectangular area. It is formed by summing up the pixel values of the x,y

Unlike Haar cascade, SURF is capable of detecting objects features that would be invariant to geometric (i.e. rotation) and photometric (i.e. brightness) transformations.

- F1(dart0.jpg) = 0.8
- F1(dart1.jpg) =0.666667
- F1(dart2.jpg) =0.5
- F1(dart3.jpg) =0.5
- F1(dart4.jpg) =0
- F1(dart5.jpg) =0.5
- F1(dart6.jpg) =0.5
- F1(dart7.jpg) =1
- F1(dart8.jpg) = 0.571429
- F1(dart9.jpg) =0.66667
- F1(dart10.jpg) =0.5
- F1(dart11.jpg) =0.75
- F1(dart12.jpg) =0.9
- F1(dart13.jpg) =0.5
- F1(dart14.jpg) =0.66667
- F1(dart15.jpg) =0.666667

As observed from the F1 scores mentioned above, The SURF detection algorithm did improve the performance of the code further. Despite the shortcomings of the code for integration, the results are quite conclusive for this test. Further improvements can be made to the detection algorithm by training the Haar-like classifier further with a wider variety of data set images. Using a better edge detector such as Canny edge detector, would have further improved the results.