# Computer Vision - Project 4 - Darts Detection

Mihail Feraru - 407

September 2023

## 1 Introduction

In this paper we will present the approaches in solving the task of detecting darts arrows on multiple types of boards from images and videos.

For the first 2 tasks, where image detection had to be made in images, a unified approach was used. Only a few parameters were fine tuned specifically for each kind of board, but the algorithm was broadly the same. Roughly, the algorithm has the following steps:

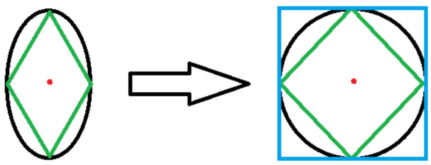
1. prepare a nice template image of the board without any perspective transform;
2. use ORB keypoint matching algorithm to extract the board from images;
3. detect the arrows in the image using a fine-tuned version of YOLOv8 model;
4. detect the contour of the arrows using MOG2 background subtraction;
5. computing the direction to which the contours point and establish where the arrowhead is;
6. check the position of the arrowhead according to the table format.

For the 3rd task, an entirely separate algorithm for processing the video frames was implemented. The steps are listed below:

1. use Hough circles transform to find the center of the dartboard;
2. detect which kind of video perspective is used using the position of the center;
3. detect the contour of the newly placed arrow using the difference of the first and last frame;
4. find the smallest triangle encompasing the contour and detect the arrowhead;
5. check the position of the arrowhead according to the table format.

## 2 Detecting darts arrows in images

To extract the board position from each image we used the provided template for each board type. The templates were perspective transformed, so they had to be reversed. To apply a perspective transform we need to select 4 points, thus we selected 4 points on the template's ellipse which should be moved to an equal distance of the center. Considering $L$ to be the length of the rectangle at destination, a transformation was applied to move the corners of the quadrilateral to



(a) Transformation applied to the template board to cancel the perspective. The corners of the green quadrilateral of the left ellipse represent the points selected in the perspective transformed template. On the right side is their destination after applying a homography crafted to cancel the perspective.


(b) Template board after getting rid of the perspective transform.

$(0, \frac{L}{\sqrt{2}}), (\frac{L}{\sqrt{2}}, 0), (2\frac{L}{\sqrt{2}}, \frac{L}{\sqrt{2}}), (\frac{L}{\sqrt{2}}, 2\frac{L}{\sqrt{2}})$. In figure 1a can bee seen the transformation and in 1b there is the result.

To extract the position of the arrows, we annotated the given dataset with the bounding boxes in which each arrow head is found as you can see in figure 2. After annotation, we trained the extra large version of the object detection model YOLOv8 on the dataset (6/2/2 train/validation/test split) with augmentation enabled. We trained it for 100 epochs with early stopping.

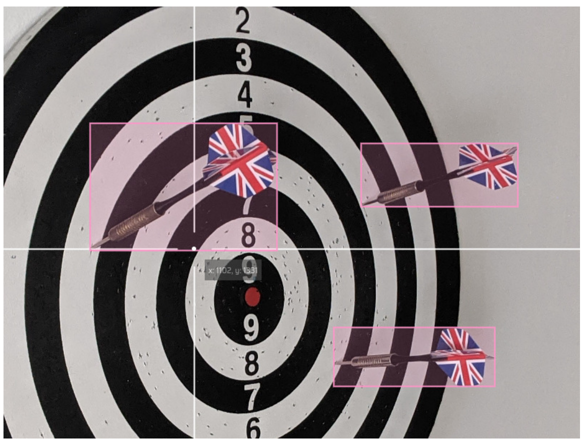


Figure 2: Annotated image of arrows on a dartboard for training YOLOv8.

Now, with a a good enough template and an object detection model, we proceed to the extraction of the arrowheads. First, we use ORB keypoint matching to obtain a perspective free look of each image with the focus on the dartboard. Then, we use the YOLOv8 model to detect the bounding boxes of each arrow. The result of such a detection can be observed in figure 3a. The confidence for each detection was thresholded at least 60%.

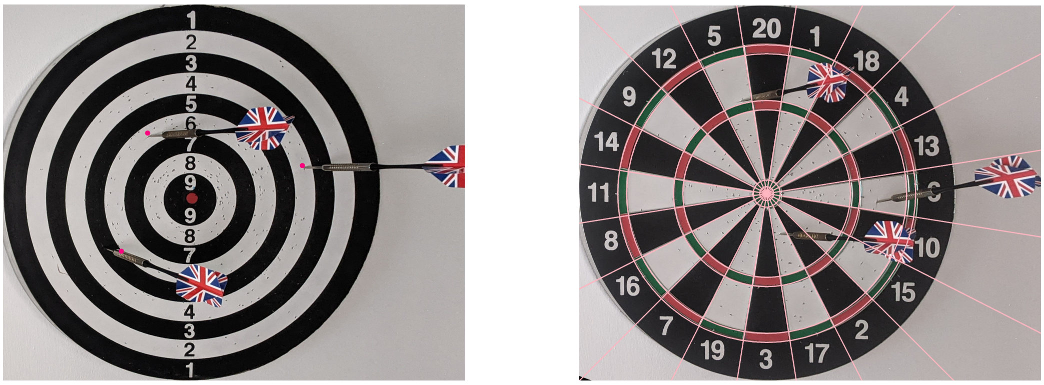
After detecting the arrows, we subtract the template image from the current image to obtain the


(a) Arrows detected using YOLOv8 on the board without perspective.

(b) Detection of smallest triangle around the contour of an arrow.

contours of the arrows on the dartboard using a background subtraction algorithm, namely MOG2. We inspect only the contours contained in the bounding boxes of arrows. Using this approach we are able to usually obtain one or two big contours, either the entire arrow or the head and the tail.

When we detect just one big contour, we use a triangle to approximate the position of the arrow head. If we detect two contours, we compute their gravity center and intersect the line traversing both of them with the bounding box of the arrow. The bounding boxes and triangle detection can be seen in figure 3b.



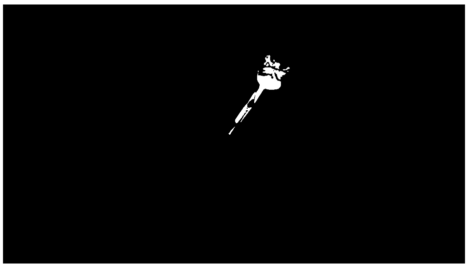
(a) The detection of the arrow heads on a given image.

(b) The sectors and rings of a dartboard.

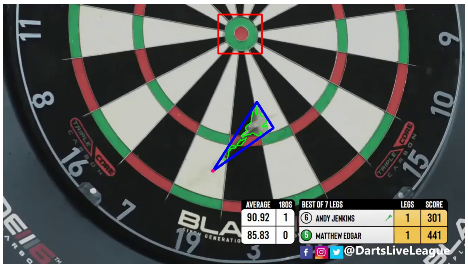
Finally, we detect the position of the arrowheads as seen in figure 4a and determine scoring using an approximation of the rings and sectors of the dartboard as seen in figure 4b.

## 3 Extending the detection to videos

The approach for detecting the arrows in video is similar, but we did not need to use a object detection model for this one, as we were looking for only one arrow and there was little noise in the images as the camera was static. First, we take the first and the last frame and we use their difference and a few image filters to obtain the contour of the arrow as seen in 5a.



(a) The contour of an arrow obtained by image absolute difference and other filters.




(b) The sectors and rings of a dartboard.

Second, we use a Hough circles transform to find the center of the board. We filter the circles based on area and their perfectness. Then, we check the known locations of the dartboard center and find which one contains a single circle. Thus, we know which perspective of the three ones we are working with.

We again find the contours in the images and fit the tightest triangle to them. We use the shortest side and its opposite corner to determine the position of the arrowhead as seen in the figure 5b.

Finally, as for the previous task, we find the angle and the distance from the center of the board to the arrowhead and obtain its ring and sector, thus having the final solution.