Detecting Players: Possible Determination of Offside During Soccer Match

Vish Panagari, Hamza Kakeh, Richard Park
Department of Computer Science, University of Virginia, Charlottesville, VA 22904
[vp5zg, ahk8fy, rp5zp]

Abstract

Computer Vision techniques have become increasingly prevalent in modern sports in recent years. Popular line detection software, VAR, has been around for nearly 5 years but has recently been adapted to the mainstream by the English Premier League. The objective of this move by the Premier League was to give referees and viewers at home a computer-generated automated decision on close calls in a soccer match.

As avid soccer fans, we have found rash and incorrect calls by the referee to be extremely disappointing in the context of the sport. One undeserved free kick in an intense competition could turn the flow and have a drastic effect on the outcome of the entire match. Therefore, in our project we decided to apply computer vision techniques we learned this semester like Hough Transform, Color Similarity, and Vanishing Point ideas to create our own model that identifies players and attempts to delegate an offside call.

1. Introduction

An offside position in soccer match is defined as a player who is in the opposing team's half of the field and is closer to the goal line than both the last opponent (excludes goal keeper) and the ball. The offside is determined by a sideline referee who views the game from his/her perspective only. It is a difficult task to continuously track the last defender, the ball, and the player that could be offsides. The referee can signal the call and stop the game at any moment, which means the accuracy of these decisions must be very high in order to avoid controversial calls.

Here we show some sample input images of FIFA screenshots that we are using in our work. Our objective is to identify the players given these input images and apply line detection for offside. The images are publicly available across google. In our project, we use still images rather than videos to make the offside call. We analyze the still-image frames of FIFA matches in order to make these calls. In current technology, a VAR system analyzes real time video feeds to determine the call, but for the sake of this project, we are analyzing a single frame to detect the accuracy of a controversial offside call.

Figure 1. Offside (penalty)

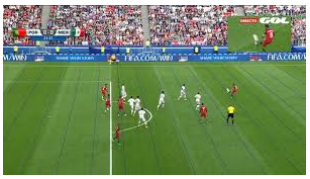

Figure 2. Onside (no penalty)

2. Related Work

Machine learning techniques implementing pattern recognition have become increasingly popular and successful in the field of visual detection. What began as basic object detection on still images has now evolved into dynamic video recognition of players, jerseys, and lines.

Recent work, in other sports like Basketball and Football has elevated the presence of augmented reality and computer vision in the public eye. Starting from technology like the virtual "first-down" line in American football invented by engineer Stan Honey of Sportsvision. The company, Sportsvision, would also go on to implement cutting edge CV techniques in sports like baseball, NASCAR, and even sailing.

3. Model

After taking in an image of a soccer match, we first perform a grayscale conversion of the image. Our model also assumes that the image given to detect an offsides is a bird's eye view of the match, where the goals are located on the

left and right of the image. This allows the model to use the canny edge detector and blob detection. For the purpose of the project, we use the OpenCV class to implement our edge detector (see Figure 3). This includes a noise reduction of the image using a 5x5 Gaussian filter. The image is then filtered with Sobel kernels to detect horizontal and vertical edges and then a non-max suppresion is applied to the result. We also used hough transformation to plot the most horizontal lines in the image in order to detect the boundaries of the field. (see Figure 4)

Figure 3. Canny Edges

Figure 4. Hough Transform (max horizontal line detection)

The image is then cropped at the maximum horizontal lines detected from the Hough transform. Once the image is cropped, we will apply a blob detector in order to identify the players on the field. The thresholds for the image are first tuned, then we will tune the blobs to detect by area, circularity, convexity, and inertia. We tune all of these parameters to find the optimal thresholds for detecting player outlines. Once detected, the players are then sorted by a color similarity detector that will partition the referee and the players of one team from another (see Figure 5).

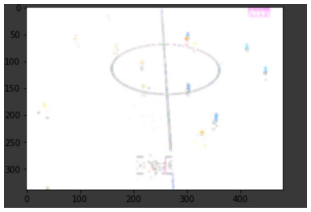

Figure 5. Color similarity for player detection

Then, the next step would be to identify the position of the ball and the nearest players to it, and determine which team has possession using a proximity measure. If we know who has possession, we can determine which direction is forward. Therefore, once we have found the:

1. position of the ball
2. possession
3. farthest forward attacking player
4. farthest backward defending player

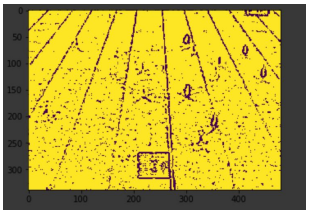

Figure 6. Adaptive Thresholding

The algorithm marks a line where an offside should be called, if a player from the other team has crossed that line.

4. Experiments and Results

We used pytorch and openCV to implement our model, but unfortunately did not have a publicly available implementation in code to build upon. Therefore, we first created the part of the model that identifies the players. The images were transformed and manipulated in order to extract only the player outline and an identifying color blob. We then built out our model to actually determine offside by implementing the vanishing point principle and computing the line from the vanishing point to the last defender/ball in order to draw out a virtual offside line.

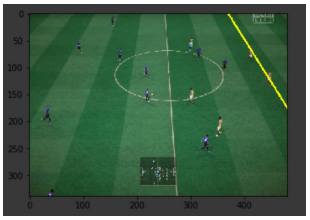

Figure 7. Offside Line

In our final report we will additionally include sample outputs from our model showing side-by-side input images with metadata, and offside labels obtained from our final model. Given our preliminary experiments we are confident that we can implement a robust offside provided a sizable data-set of pitch images.

References

[1] Open CV: Canny Edge,
https://docs.opencv.org/trunk/da/d22/tutorial$_p y_c anny.html$

[2] Detection and Tracking the Vanishing Point on the Horizon,
https://pdfs.semanticscholar.org/ba19/1fa3a3adb372384b3079ed5287fb5a14880f.pdf

[3] Blob Detection Using OpenCV,
https://www.learnopencv.com/blob-detection-using-opencv-python-c/

[4] Simple and Adaptive Thresholding,
https://opencv-python-tutroals.readthedocs.io/en/latest/py$_t utorials/py_t utorials.html$

[5] Background Image Processing,
https://docs.scipy.org/doc/numpy/reference/generated/numpy.ones$_l ike.html$