DIGITAL IMAGE PROCESSING

SOCCER VIDEO ANALYTICS

Introduction

The problem is an interesting application of a combination of image processing and semantic soccer logic in order to generate information about a broadcast soccer video. When watching a game on TV, we can automatically perceive certain information based on what we see (goals, shots etc). A state of the art analytics system should be able to process this information and also compute other information which is not very obvious (like possession, number of passes, shot to goal ratio etc). In this work, an input soccer video is taken and processed to determine the locations of players and the ball in every frame and be able to track them across frames. Statistics such as ball possession, game momentum are also generated, and as well as an attempt to detect events such as goals, passes and shots.

Parts of the Project

1. Player/Ball Detection

   Frame processing, Background subtraction, player detection, ball detection, Mapping current view to actual pitch location

2. Player/Ball Tracking

   Tracking player movement, Identifying a particular player, tracking ball movement.

3. Event Detection

   Goals, Passes, Shots

4. Statistic Generation

   Momentum, Possession, Player Specific

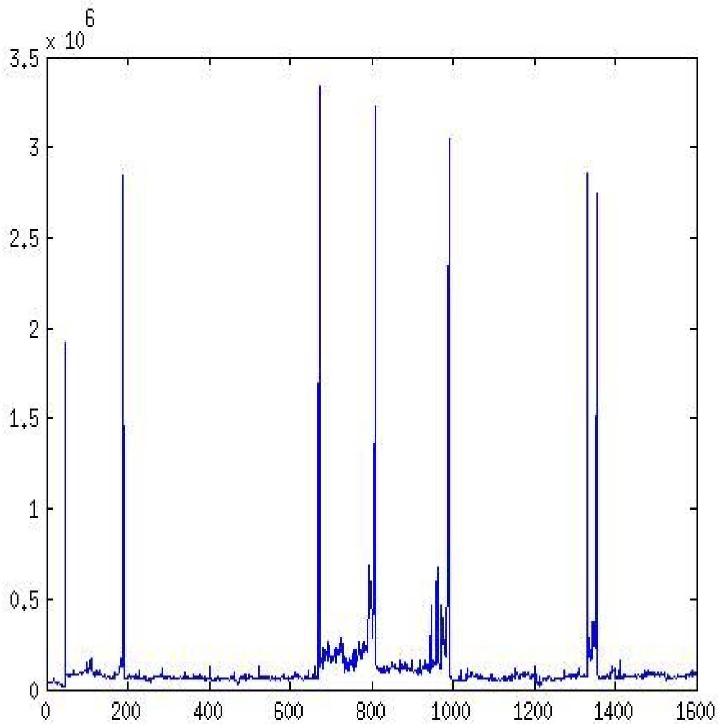
Video Used For The Project

The video from the 2014 FIFA World Cup semi final match between Brazil and Germany was used. Only the long view was considered (from the central camera which is used most frequently during the normal course of the came). Close-ups, camera changes, replay angles etc were not considered and the individual parts were done only in the long view which was determined using a histogram of differences.

Part 1 - Player/Ball Detection

Frame Processing

Every frame in the video was processed to determine if it was the long view or not based on the assumption that the video started out in the long view. A Histogram of Differences was used to detect changes in the view. Sharp peaks in the histogram of differences indicates a hard cut (transition between long view and any other view).



A plot of the histogram of differences for the first 1600 frames of the video (from 20:34 in game time) show the following sharp peaks which represent transitions from the long view to any other view.

Other views could be close-ups, or out of context audience views, or hard cuts which lead to replays being shown from different angles after a goal is scored and so on.

Background Subtraction

Background subtraction is necessary in order to limit the region of interest to within the playing area. It gets rid of all of the non-playing areas such as the audience. Two different kinds of background subtraction techniques were used to get the required feature images. The first kind is in the RGB color space, and is done by first masking out the green color of the field and then applying morphological operations such as the top-hat transform, opening and closing to get only the field region, with the players and the field lines as the output feature image.

Figure 1 - The input image

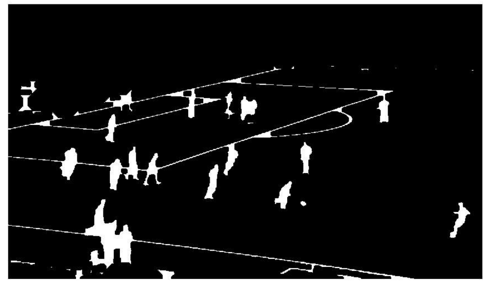


Figure 2 - background subtracted feature image

Figure 3 - Extracted pitch using HSV

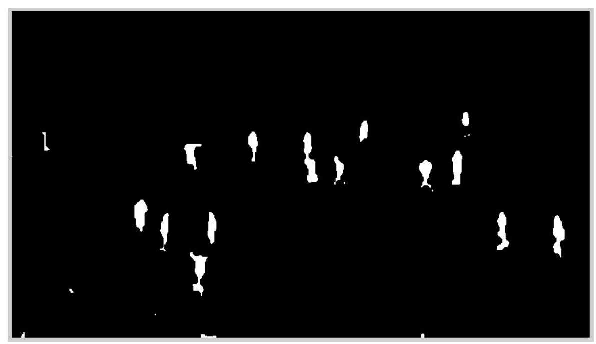


Figure 4 - Feature image having player locations

The second kind of background subtracted feature image is determined using the HSV color space. This is used for player detection as it does not have any field lines.

Player Detection

Player detection is done by first subtracting the audience from the HSV image using the audience mask, following which the resultant image is eroded to remove any possible noise.

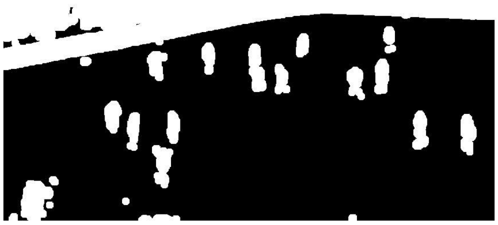
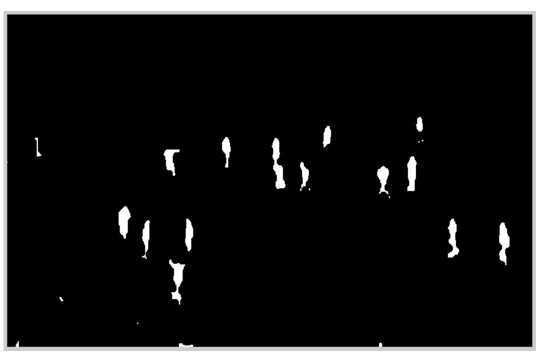


Figure 5 - Input Feature Image

Figure 6 - Feature image after applying audience mask

In the final feature image, all connected components are found and a bounding box is drawn around each of them. To avoid small speckles in the feature image from being detected, the minimum area of the connected component is constrained to be 200 pixels. The entity within each of these bounding boxes is assumed to be the detected player. All these detected players are marked and their coordinates are passed on to the other parts of the pipeline for further processing.



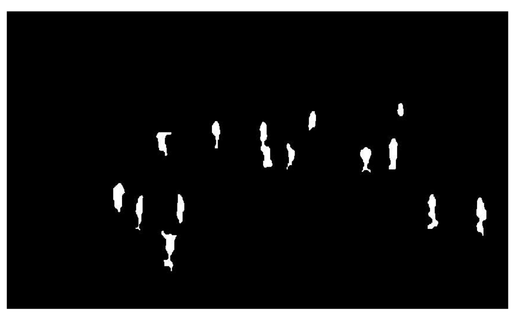


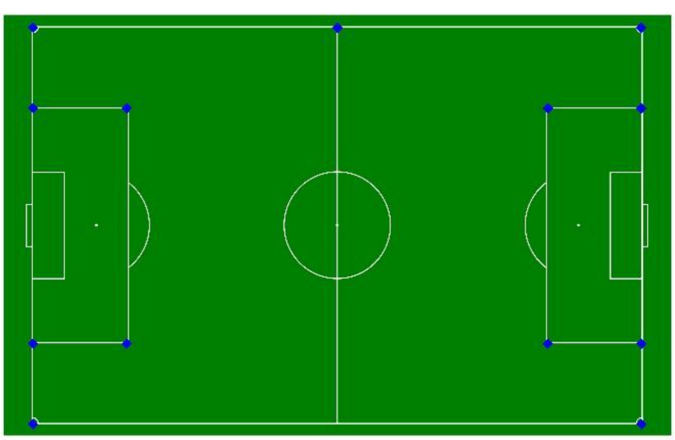
Figure 7 - Eroded Feature Image

Figure 8 - Final Feature Image

The results of the steps described above are as shown. We do get some inaccurate detections such as clusters or when two players are very close to each other and get detected as one etc, but for the general case, the process works fine.

Mapping Current View to Pitch Location

The view in the current frame was mapped to the actual pitch location by using the Hough Transform and morphological operations to generate the required feature images. The first step is to extract the white field lines and then determine which field line we are looking at in the original frame. Based on the locations of the detected field lines, we determine if any of 13 possible keypoints are part of our current view.



For images in which no keypoints were detected, the change from the old angle was seen. If the camera was between the box and the centre (no points), depending on left or right, the angle was set to +22.5 or -22.5 which was a reasonably okay estimate.

Figure - Input Pitch Image

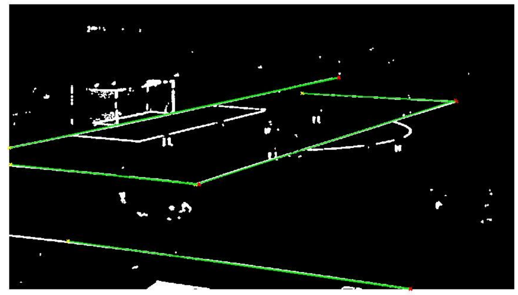
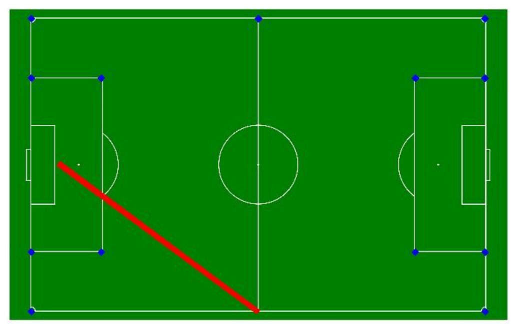


Figure - Hough Transform lines

The input pitch image is generated using HSV based background subtraction. We then convert this image to the LAB space and then apply thresholding on the Luminance component to get the feature image on which we apply the Hough transform. Depending on the positions of the lines detected using the transform, we identify keypoints.

(Detected keypoints marked by blue dots)



(Detected viewing angle marked)

The detected viewing angle was marked and its value was stored to be used in further parts of the pipeline.

An attempt was made to stretch the contrast of the detected pitch in order to detect the angle of

the mowing lines, which would help accurately map any view angle. Unfortunately, this failed.

Part 2 - Player/Ball Tracking

Tracking Player Movement

The output of the 'player detection' section gives us a structure of connected components which we assume to be players. A 'history' field is added to this structure which saves the label of the previous connected component to which it matches.

	1	2	3	4	5	6	7	8	9	10
1	1	1	1	1	1	1	2	2	3	1
2	2	2	2	2	2	2	3	1	2	2
3	3	3	3	3	3	3	4	2	4	3
4	4	4	4	4	4	4	5	3	5	4
5	5	5	5	5	5	5	6	4	6	5
6	6	6	6	6	6	6	7	5	7	6
7	7	7	7	7	7	7	8	6	8	7
8	8	8	8	8	8	8	9	7	9	8
9	9	9	9	9	9	9	10	9	10	9
10	10	10	10	10	10	10	11	10	11	10
11	11	11	11	11	11	11	12	11	12	11
12	12	12	12	12	12	12	13	12	13	12
13	13	13	13	13	13	13	14	13	0	0
14	14	14	14	14	14	14	0	0	0	0

In the first frame the history field is populated with the corresponding component number. There are two arrays which store the centroids of the previous frame's connected components and the current frame's. For every connected component in the current frame, the distance between itself and the centroid of all the connected components of the previous frame is computed and stored in the distance matrix. The component number which returns the least distance is stored in the connectivity map. We use a distance metric for tracking as we can say that the displacement of a player is very small between consecutive frames. If none of the distances are within a threshold, a new player has entered the scene, and thus it is not given a component number. The history field of the current structure of connected components is populated with the values of the history field of the corresponding component number saved in the connectivity map. Every component for which the connectivity map cell is empty is assigned the next highest number which isn't already existing in the history field of the current connected component structure. The drawback to this method is that we cannot identify whether the same player has re-entered the field. The main issue lies in the player detection where some of the components are not actually players, or when some players are removed due to the morphological operations.

Player Team Classification and Mapping

From each of the detected sub-images, the player was classified into one of three categories (Brazil, Germany, Referee) depending on the average color value of the non-green pixels in that sub-image.

The classification was done based on the Euclidean distance of the average (R,G,B) value from the following predefined values and the assignment was made to the one with the minimum distance.

```
bra_color = [195,190,30];
ger_color = [50,35,45];
ref_color = [30,50,110];
```

The location of each detected player was mapped to the pitch coordinates. For the scope of this project, player identification was not done, although that would immensely enrich the entire pipeline.

Figure - Detected and marked players

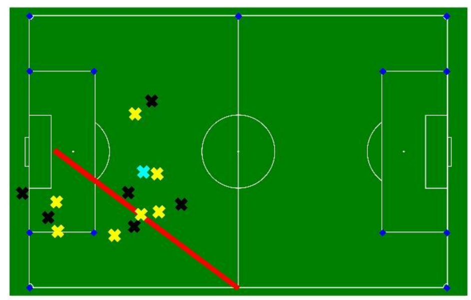


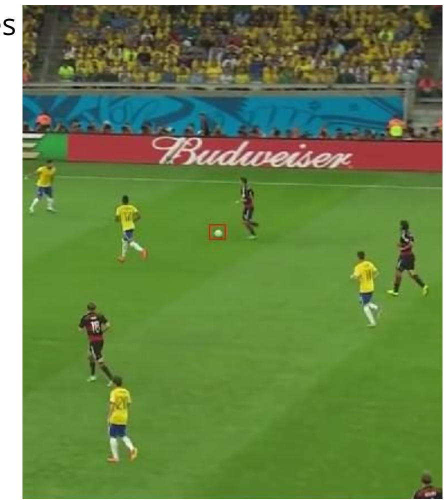
Figure - Players mapped to pitch

Tracking Ball Movement

The background which consists of the crowd is subtracted from the from the image , this is output of the background subtraction step and this mask is applied on the the grey-scale of the original image. This is done so that the noises in the crowd and outside the fields are reduced.

Initially the ball is annotated from a frame , then for the coming frames block matching (done using correlation) is used to obtain the possible location of the ball. This is done in spatially constrained with the previous location of the ball.

First, a ball is annotated ,in our case it is saved. For the subsequent frames this annotated image of the ball is used to detect the part of the image which best correlates with it .This gives the most likely position of the ball.



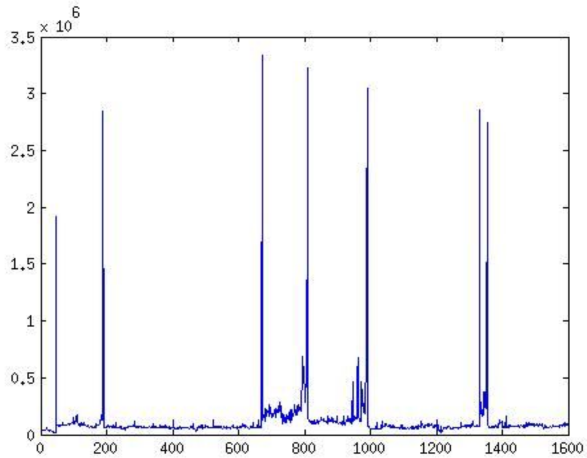
This is always not a good indicator of the ball as in few cases the some other part of the image can also show a better result .To improve the search for the ball , the previous location of the ball is saved and if the distance to the correlated part is far from the previous location then the new location is decided in the proximity of the previous location and ball is detected using the property that it is white.

The location of the ball can be further used to find passes, shots, possession etc.

Part 3 - Event Detection

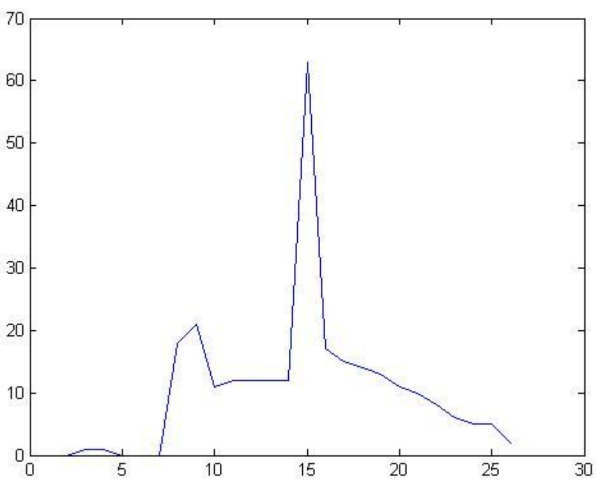
Goals

Goals cause very sharp changes in the histogram of differences of the current view (which is caused due to there typically being a sequence of replays being shown for a certain time immediately after a goal has been scored). Using the histogram of differences, we can detect possible goals and differentiate between a goal and a different hard cut in the video depending on the amount of time for which we were out of the long view.

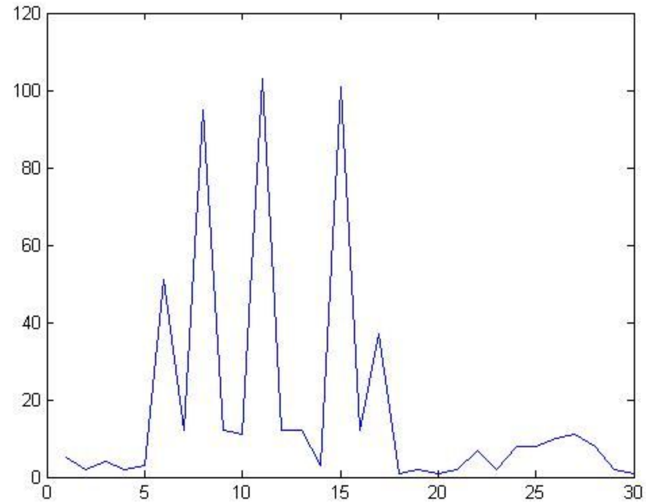


Passes

A pass is detected as a sharp change in velocity of the ball. A peak in the plot of ball velocity vs frame number refers to the instant that a pass was made.



Shots



A shot is also detected as a sharp change in velocity of the ball.

However, a shot is different from a pass in the fact that it has higher velocity and is near the goal.

The multiple peaks in the shot vs frame number graph are due to uncompensated camera motion.

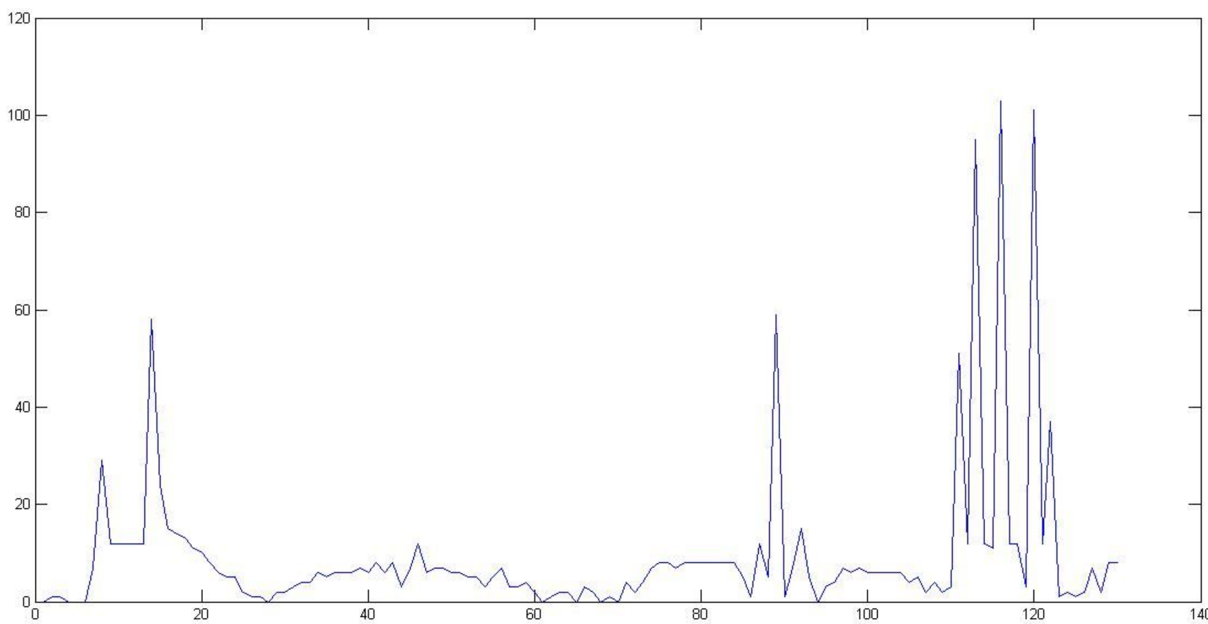
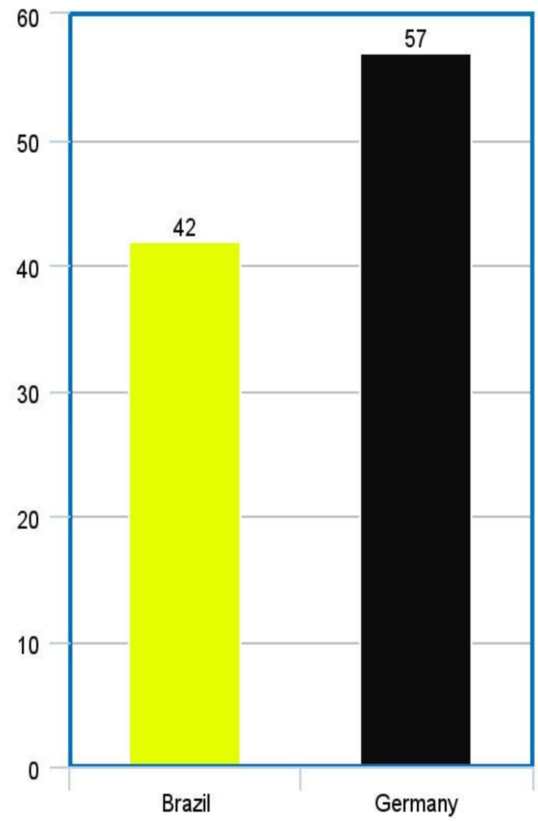


Figure - A plot of ball velocity vs frame number for a sequence of 2 passes and a shot

Part 4 - Statistic Generation

Ball Possession



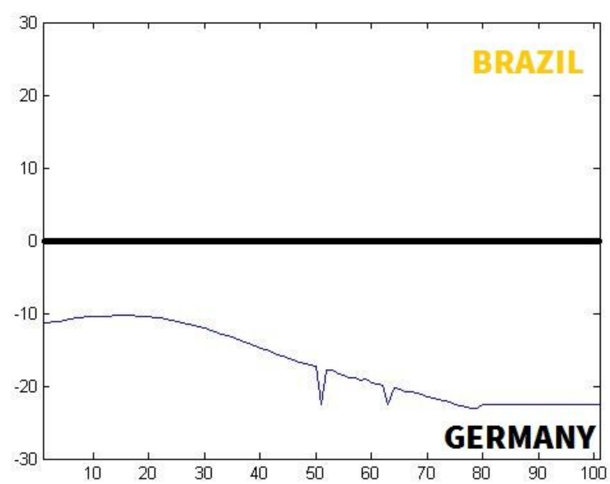
The location of the ball and the players was approximated. Using these positions, rough estimate of which team has possession of the ball was made.

A simple assumption was (although it is not very robust) that the team of the player who is closest to the ball has possession of the ball.

Possession over time can be represented as a percentage, as is done in a conventional TV broadcast of a soccer game.

Trying to determine possession this way was somewhat successful, because our results detected that Brazil had 42% and Germany 57% (with 1% error for the referee).

Game Momentum



An estimation of game momentum using the angle of view and depending on which team had possession of the ball.

For instance, if Germany had the ball and the camera was facing the Brazilian goal, then game momentum is highly in favour of Germany.

The result for a small sequence of a German attack seemed to give a somewhat accurate representation of this.

Player Specific Statistics

Player specific statistics could not be determined because an exact player could not be identified. If an individual player could have been determine, then relevant detected events could be associated with that particular player. Player specific heat-maps could not be generated either, due to the lack of identification.
The detection of a specific player would involve some training and classification which would be a very good extension to this project, as it would pave the way for a lot more interesting features to be added and more infographics to be generated such as the visualization of every pass a player has made or every shot that he has taken (similar to the infographics which we see in halftime shows of modern European soccer broadcasts).

Possible Future Work

- Make the whole system more robust by intelligently detecting parameters from a random input video instead of having to specify individual parameters.
- Improve player classification from just splitting between two teams to actually being able to identify a particular player.
- Optimize the implementation which as of now is very slow.