Fariborz For the Win!

Mohammad Amin Parchami

Spring 2021

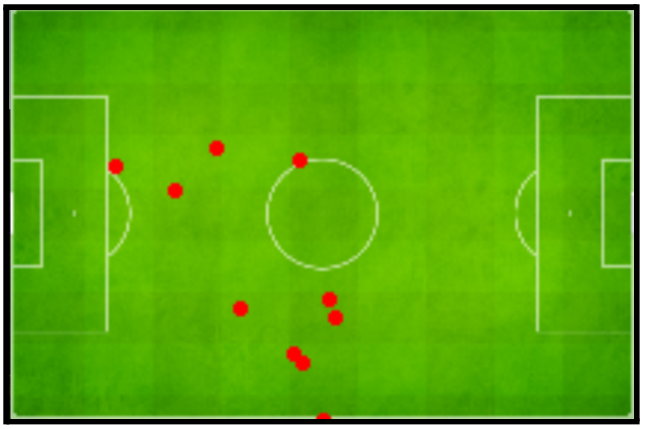
As we head towards the FIFA World Cup 2022 in Qatar, Fariborz has recently joined the national team as a coaching assistant. His job is to suggest effective formations (where players stand on the field) to the coach. He has decided to start by studying the recent games of the team. Since watching the entire match can be time-consuming, he asks you to develop an application to generate player's per-frame positions from a video sequence.

In short, given an input video, we would like to locate the players in the scene, identify their team, and create a 2D map of their location from the bird's eye (top) view.

The project is divided into two main parts: 1- Detection and mapping, 2- Classification.

Part 1 - 70 points

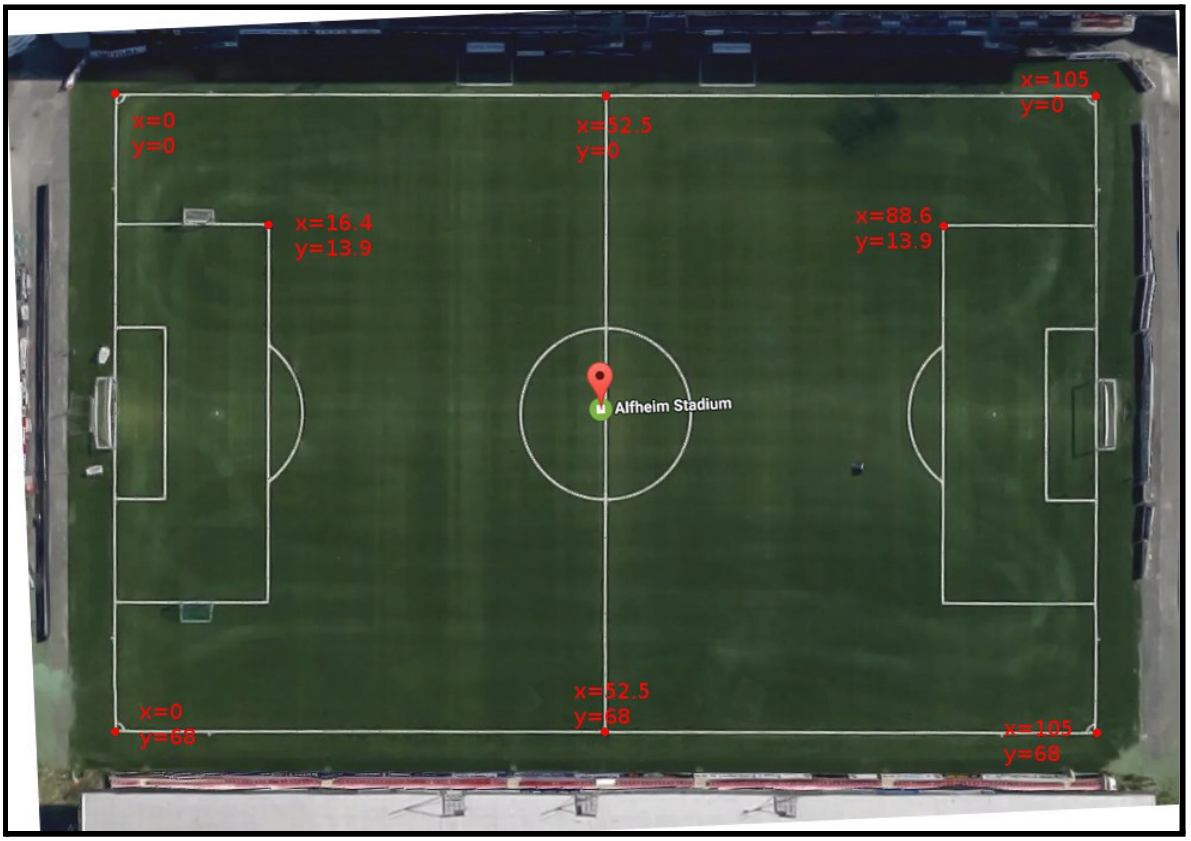
In this part, our primary focus is to detect everyone (including the referee) in the scene. We also want to generate a 2D map of the field.



Method

The detection method is of your choice. However, if you choose to use neural networks, you cannot use them for classification (part 2).

You can use the following map of the stadium for accurate map generation:



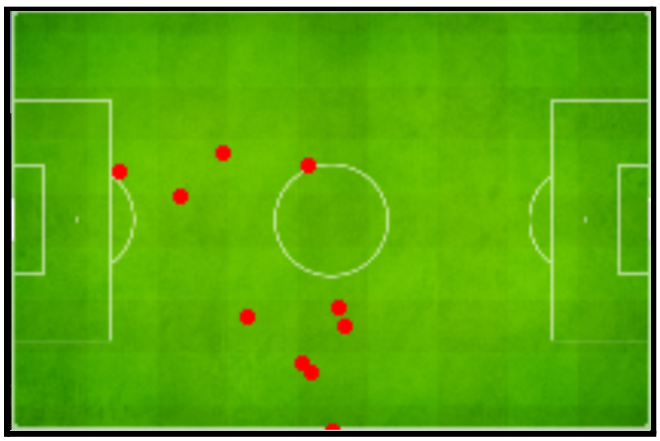
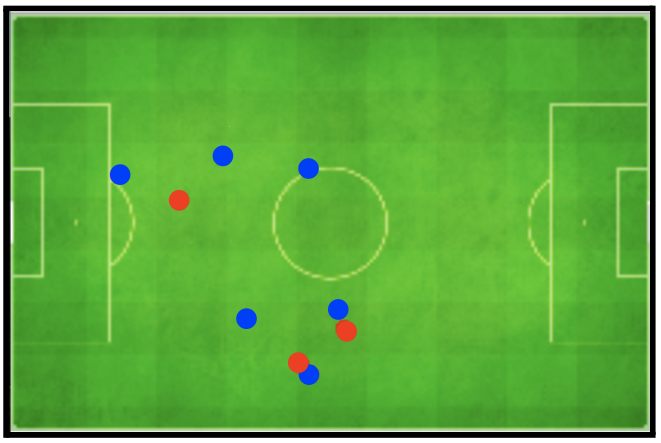
Hint 1: You can choose to detect players using either the original image or the projected image.

Hint 2: You can assume that all players are standing straight on the ground (ignore jumps and slide tackles)

Useful Links: **YouTube(Link1, Link2, Link3, Link4), Link5, Link6**

Part 2 - 30 points

In this section, we assume we have the positions of the players. Now, we want to classify them. Keep in mind that the 2D map colors do not have to match the actual team's color. The only goal is to separate players.

The final 2D map does not have to locate the referees; detecting them has 10 extra points.

Method

The classification method is of your choice. However, if you choose to use neural networks, you cannot use them for detection (part 1).

Hint 3: We recommend using neural networks for classification and use other methods for detection.

Hint 4: For fast data acquisition, we suggest developing a simple labeling tool. First, save the patches of detected players from part 1. Then, write a code that opens every saved image and waits for two keys, such as R and L. Assign each key to a team and label the images based on the user's input.

Hint 5: You can add some pre-processing to the patches. For instance, you can try lowering the green intensity in the patch.

Note: You must use neural networks at least once for either part 1 or part 2, and you cannot use them for both parts.

Part 3 - 25 points (Extra points)

You can start working on this part **only** if you have finished the last two.

As you know, consecutive frames are often very similar. That is, the objects' motion is relatively low. With this in mind, we can avoid doing detections for every frame and do it in intervals instead. For the intermediate frames, we can track the bounding boxes. For instance, we can choose to run a detection method once every ten frames and track the existing detections for the remaining nine frames.

For this part, we ask you to add player tracking to your project and avoid doing detection in every frame.

Hint 6: A general workflow can be: extracting features from the region of interest, tracking them in the next frame, and estimating a transformation.

Useful Keywords:

Dense Optical Flow, Sparse Optical Flow.

Method

Any method of your choice.