Project Name: SoccerNetPlus

AiJing Wu (awu53, 9081879174)
Michael Vanden Heuvel (mmvandenheuv, 9079226842)

Current Progress (UPDATED)

Google Cloud Directory: SoccerNetPlus

1. Finalized our direction of implementing the idea without using the existing source code
2. Finished player detection
3. Almost finished the 2D spotting on a digital field
4. Finished the first version of Anvil frontend webpage
5. Integrated with Google Colab for both frontend and backend

Project Background and Motivation

We will address the problem of accurately detecting players on a soccer field in real time in videos. In addition, we will also estimate the localization of each player to create a top down 2D representation of the field with markers for each player.

Soccer is one of the world's most popular sports. As a result, the revenues associated with the sport are astronomical. Improvements to the way fans view soccer have the potential to increase revenues, affect the way billions of people enjoy the sport, and attract even more people to the sport. Utilizing a player detection and camera calibration model in unison could be used in scenarios such as auto-generating highlights, improving live graphics, and strategy analysis. Successfully recreating such a system would be rewarding both technically and personally because of our passion for soccer. We are passionate about this project because we are interested in the usefulness and range of applications of tracking people in video, enhancing user experience, and improving people's understanding of soccer.

Major Reference (UPDATED)

Camera Calibration and Player Localization in SoccerNet-v2 and Investigation of Their Representations for Action Spotting
How to track football players using Yolo, SORT and Opencv. | by C K | Towards Data Science

The Current State-of-the-art (UPDATED)

1. OpenCV object tracking
   OpenCV is really popular and is extensively used in the field. It has a bunch of built-in functions to perform various object-tracking activities. OpenCV is the best practice for us to do a simple tracking system on our own.
   We have used it to detect players with the help of Yolo. Also, add annotations to the original video with the result of detection.
2. Yolo v3 Object Detector

YOLO (You Only Look Once) is one of the most popular series of object detection models. Its advantage has been in providing real-time detections while approaching the accuracy of state-of-the-art object detection models.
We are using its pre-trained model to help detect players between teams (and separate players from the judge) along OpenCV.

3. DeepSort object tracking
DeepSort computes the bounding boxes around the objects in the videos. This is a very cool algorithm with an identification model and estimated tracks (see https://arxiv.org/pdf/1703.07402.pdf).
This might be in use for us if we want to split all players and find their individual statistics.

Goals for new implementation (UPDATED)

After multiple attempts and due to hardware limitations (GPU), we transitioned from using the existing source code from SoccerNet-v2. Instead, we built our own model with the help of OpenCV and YOLOv3 on Google Colab with Anvil as the frontend. SoccerNet-v2 relied on the use of CUDA cores, but neither of our local machines or many free resources have dedicated Nvidia GPUs. Therefore, we used an OpenCV implementation that is able to utilize a GPU or CPU to complete tasks.

New goals for updated implementation:

1. Detecting players
2. Splitting players between players & roles (yeah there is a judge on the field)
3. Spotting the players' coordinates on a digital field from the birds-eye view
4. Tracking players with their paths
5. Built a web app with new frontend UI

Changes between our current proposal and our original proposal with SoccerNet-v2:

1. We cannot deal with the moving camera cases
2. We need the user to enter coordinates of the field in their video to get a better spotting result
3. The time taken for the whole process will be longer

Our contribution and creative aspects (UPDATED)

1. Developer Perspective:
    a. Eliminate the hardware restrictions (NVIDIA GPU)
    b. Interaction with the model is more portable, as the model's inputs and outputs will be online resources. Specifically, the inputs will be handpicked videos available in Google Drive or user-chosen YouTube videos. Output will be saved in Google Drive, available to any user with access to the folder
    c. Easier to collaborate with the use of Colab
    d. Provide detection with additional markings on the original video for people's reference
    e. Provide 2D birds-eye spotting, which could be especially useful for analyzing gameplay, such as when a possible offside happens

    f. Provide analysis on different players, and statistics display
2. User Perspective:
    a. Easier to use – don't need to have any CS background
    b. More accessible – as long as they have a Google account, they could use this app
    c. More interactive with our UI presented
    d. Could directly use a YouTube link to get analysis, so they don't even need to know how to transform the video to be used by the model
    e. Could directly see the statistics of the players

Measurable Results

Quantitative results correspond to the qualitative questions:

- What is the success rate of identifying players within an image?
- What is the success rate of locating players within an image?
- How many seconds can the program track a player?
- What is the frame rate of the simulated field (can it keep up in real-time)?

Qualitative results will be multi-layered, including:

- Can our program identify player(s) within an image?
- Can our program identify where in the image the players are located?
- Can our program track players from frame to frame within a video?
- Can our program correctly map player locations in an image to player locations on a field?
- Can our program be interacted with effectively via some sort of user interface (command line, web app, etc)?

Estimated Timeline (UPDATED):

Week Range	Tasks	Deadline
Oct. 2 - Oct. 8	Get familiar with existing technologies Decide the main technology to use Write proposal	Proposal (5%)
Oct. 9 - Oct. 15	Gather video with steady camera angle Learn SoccerNet-v2 source codes Split tasks within the group	
Oct. 16 - Oct. 22	Experiment with existing models Try to set up the environment and make the code work locally Dig down to learn how to interact between backend and frontend	
Oct. 23 - Oct. 29	(Direction changed from now on) Learn OpenCV and Google Colab Learn to use Google Colab to make an app	

	via Anvil Start to do player detection with OpenCV and Yolo	
Oct. 30 - Nov. 5	Finish player detection Start on 2D spotting	
Nov. 6 - Nov. 12	Finish 2D spotting Built a prototype of UI with Anvil Link Anvil to Colab - Make python functions callable from Anvil app Write report	Midterm Report (5%)
Nov. 13 - Nov. 19	Start player tracking using SORT Work on backend calculation for general statistics Continue adding more components to Anvil	
Nov. 20 - Nov. 26	Finish individual player tracking Add the functionality of directly using YouTube link to get video Take in user input to allow only analyzing specific timecodes of videos (instead of first 10 seconds)	
Nov. 27 - Dec. 3	Try to detect corners automatically if possible Finish up all minor issues Make sure the whole backend and frontend connection is stable for (limited) public use	
Dec. 4 - Dec. 15	Prepare final demo Finish webpage	Final Presentation (10%) Project Webpage (15%)