Analyzing the Detailed Metrics for Soccer Penalty Shots: CSC420 2019 Project Report

Sean Tong, James Bao

Abstract

As of today, there's been numerous examples of an impressive soccer Penalty shots and attempts. In the real world, no two shots are exactly the same. For any single video footage of a soccer penalty shot attempt in interest, we would like to extract as much data as possible in regard to the soccer ball being kicked. This includes the direction, trajectory, speed, curvature (if any) and as well as the total distance the ball has travelled during the whole duration of kick.

1. Introduction

Video footages of worthwhile soccer penalty kicks can be easily found via the internet. While the videos can be entraining look at, there does not exist a tool that provides the detail stats of the kick unless the information is embedded in the video. Generally, the footages come in variety of quality, video framerate (FPS), and angle of view.

We built a tool that provides generalized and objective analytics over the kick that is somewhat invariant to all of these variations. A user may extract video of a plenty shot from the web, fed into our program, and be returned the statistics that the user like to see in a very reasonable time frame.

1.1 Related Work

For this project we taken knowledge from many sources, both inside and outside of the course content. The concepts introduced from class, in a board sense, includes the following:

- **Convolution:** A quick way to detect if a window is similar of that of a soccer ball
- **Machine Learning:** The heavy lifting of detecting of a soccer ball. Convolution or Hough circle detector is used if the resulting network returns multiple regions with similar confidence.
- **Homography:** knowledge to generate perspective transformation matrix M and to perform perspective transformation. Therefore calculating the relative distance travelled by the ball per frame. Subsequently with FPS information of the video speed of the ball can be calculated.

There are concepts and procedures find elsewhere. This includes the alternative template matching methods from CV2's documentations[1]. A deep learning model of soccer ball detection, for modeling reference, and tutorials in how to make it work (Chansung, 2018). C code of Hough circle detectors and Hough transform. (Hough Transform, 2014)

1.2 Expectations and Hypothesis

After looking though the related work we are able to find as guidance, we expected a few challenges. The following list gives an overview of the potential limiting factors we may face when building the software.

- **Lack of soccer training data**: In order to build a successful machine learning model, we need to give sufficient data so that the network may produce desirable outcome. We anticipate that the process of gathering the training data will be for the very least, troublesome. We plan to also look if other implications of network of similar purpose have

[1] The alternative matching also includes convolution (Alexander Mordvintsev & Abid K, 2013)

listed their source or use transfer learning so that our network could (hopefully) work on a much smaller dataset. If both resolutions fail, we plan to generate footages from in the PC game FIFA 19. We expect the generated footages should be similar enough to generalize onto boarder use.

- **Error in ball distance calculation due to the ball being in the air.** When the ball is in air, it is sometimes mistaken as the ball has gone further away from the camera. However, depending on the viewing angle, this may bring a drastic difference in the distance calculation. As we calculation the soccer balls' traveling distance by seeing how much it has traveled on a field after reversing its perspective transformation. Unless we identify the flying height of the ball, the distance will all be calculated as the ball traveling north, relative to the camera. For now, we will try to avoid calculations of footages where the ball is significantly high above ground.

2. Methods

Using the related work as guidance, the workflow of our software will execute in a few general steps.

1) Use various techniques to detect the pixels that represents the soccer ball, for each frame of the video.
2) Find the center of each pixel that presents the ball, save the information.
3) Perform perspective transformation onto the field (goal zone) so that it looks alike a top down view of itself. The saved location of the ball should be transformed respectively as well. By now we should know where exactly the soccer ball is in respect to the field, for each frame of the video.
4) Workout ball speed, trajectory, angle, total displacement and various other data by taking in consideration of the size of the soccer field goal zone 40.3 × 16.5

2.1 Parts from Partner

My partner has spent time tuning and testing to build a custom neuro network that performs soccer ball (and players) detection. The results, combined with some traditional template matching techniques, is far superior than just using template matching on its own.

The algorithm works the best on video clips generated from FIFA 19, but works fine with all other cases. Usually, soccer ball detection works despite visual blockages (for example the player's leg) as illustrated in **Error! Reference source not found.**.

Figure 1. Ball Tracking given different states of the same soccer ball.

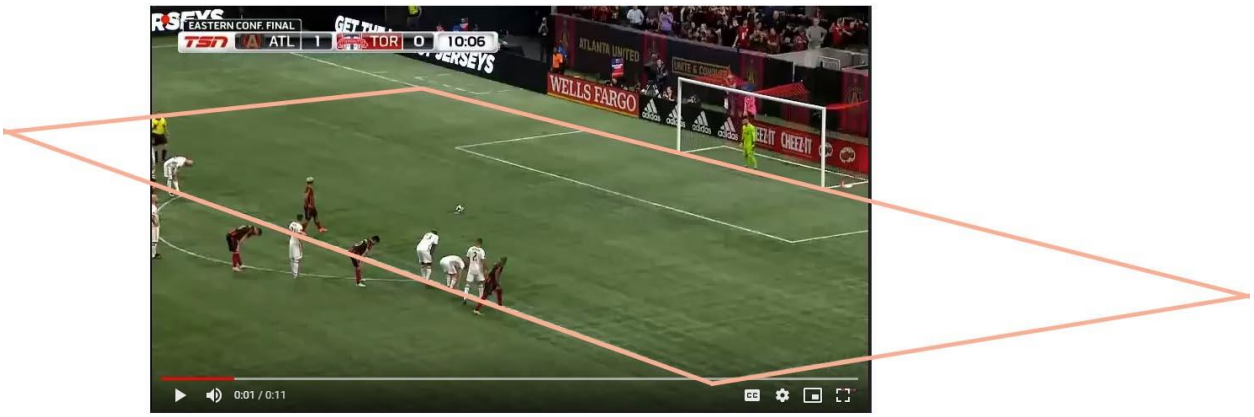
From the proposal, I mark out that I'll be responsible in creating solutions homographic transformations, so that the field of each frame can be converted into a bird's eye view for analytical purposes.

Thereon, calculations of the statistics become easy as we measure the distance of the ball's travel in respect to the field/goal size.

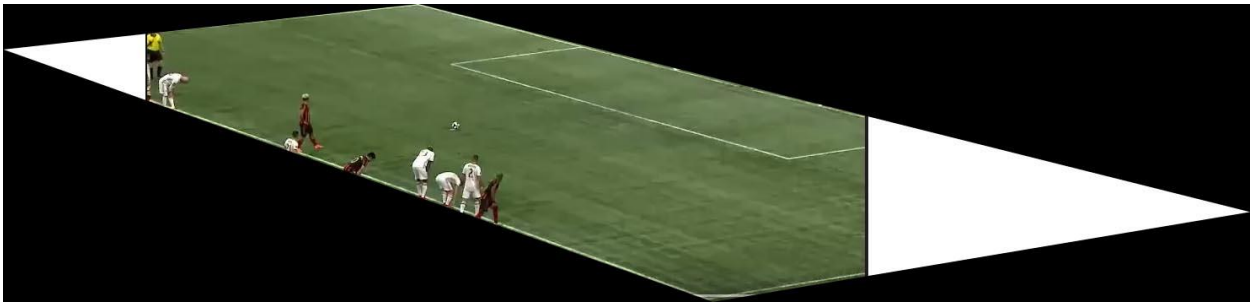
I developed a procedure that enables automatic transformation of fields into bird-eye's view, steps are overviewed as below.

Step 1: take any frame from a penalty shot video

Step 2: Recognize the lines and therefore create a bounding box that predicts the entire filed of image. This is the hardest step for a few reasons. One is that edge detectors will always return two lines for every white line on the image, secondly the entire soccer fields is not visible, so precaution needs to be taken in how much the image needs to be resized so that the whole field can be estimated

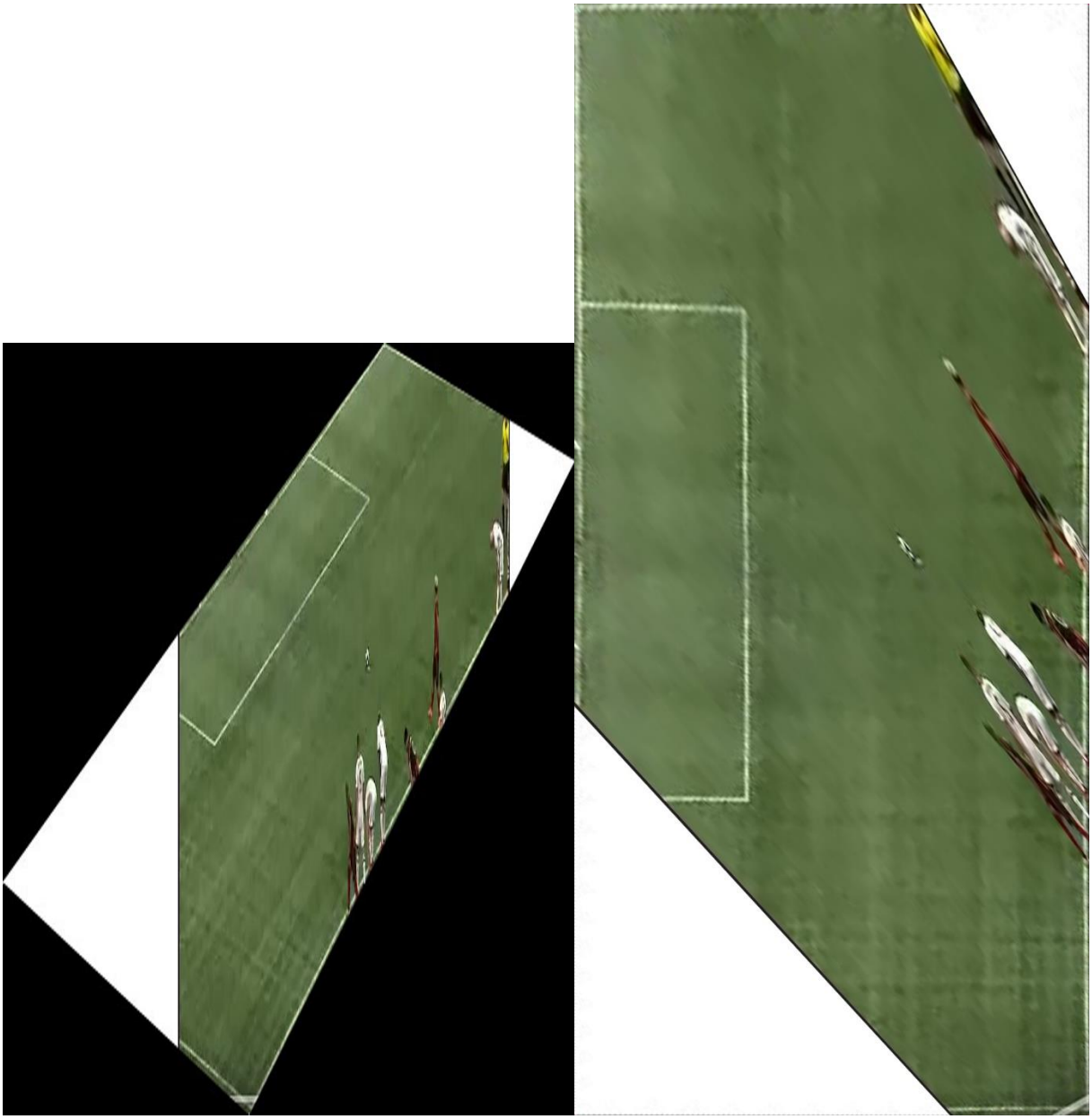


Step 3: Create a cutout image of the region



Step 4: Scale the cutout so that it is proportion to the actual proportion of the goal box **2**, left.

Step 5: The final result of the field, **2**, right.



2 The last two steps of automatic perspective transformation

3. Conclusion :

In conclusion, the overall idea worked, how-ever to make this work in a very general case, a lot of our parameters need to be retuned or generalized. The homography transformation was successful in a controlled video (points were guaranteed to be there) but performed poorly in a random video. Using a simulated soccer game allowed us to control more variables but took away from the generality of our project.

4. Further Discussions

In the future for this project, there are many things that can be done. We descoped player analysis as it is very similar to the code using the ball however identifying the player took too much time in our implementation. This would be a beneficial next step as providing information on the player is more useful than just the ball. Somethings I would want to improve on in the future for my part would be locating the ball with more detail. More specifically, in the homography transformation, I would have liked to be able to spend more time trying to figure out a reliable way to judge how high the ball is elevated from the ground. Mapping the x and y coordinates from the 3D image to my 2D representation was doable, but the calculations did not ac-count for the height. I considered various ide-as such as detecting the size of the ball in relationship to the box, with the assumption that a higher ball would be closer to the camera therefore larger. This did not work well as the size change was almost negligible. Another future step would be to map segment of the video unto the entire soccer field, instead of just a controlled portion of the penalty box. One last future step that would improve the overall quality of this project is being able to analyze an entire soccer game, not just a penalty kick. The code should not be too different than what we currently have, it would just have to be more robust. One issue I had was finding enough match points to correctly identify the location but towards the end of the project I noticed quite a few points that I did not consider early on.

5. References

Alexander Mordvintsev & Abid K. (2013). *Template Matching*. Retrieved from https://opencv-python-tutroals.readthedocs.io/: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_template_matching/py_template_matching.html

Chansung, P. (2018, July 26). *YOLOv2 to detect your own objects using Darkflow*. Retrieved from Towards Data Science: https://towardsdatascience.com/yolov2-to-detect-your-own-objects-soccer-ball-using-darkflow-a4f98d5ce5bf

Park, C. (2019). *https://github.com/deep-diver/Soccer-Ball-Detection-YOLOv2/blob/master/sample_img/test_image1.jpg*. Retrieved 2019