Statistics generation through automatic soccer video analysis

Riccardo Catalini, Davide Abba, Sebastiano Aloscari

Università degli Studi di Modena e Reggio Emilia

***Abstract*—Automatic interpretation of a soccer game is a very difficult challenge, due to the number of elements, information and events occurring during a match. Many modern techniques face the problem with very deep neural networks and using very big datasets. In this paper we focused on a more heuristic approach in which we tried to manage the task with limited datasets and constrained resources, creating a useful tool to extract some salient information of a soccer match.**

***Keywords*—soccer analysis, object detection, color segmentation, lines detection.**

I. Introduction

Analysing videos of Soccer matches is a major challenge in Computer Vision. This project goal is to understand match situations, trying to detect events occurring during a soccer game, and providing some useful statistics to help soccer clubs and coaches improve their game strategies. For instance, coaches can recognize usually adopted tactics to understand if an opposing club is more offensive or defensive and use this knowledge in order to train his players in the best manner to face the opposing club in the future matches. In a soccer match video, images are taken from different angles, light conditions, and perspective (wide or zoomed views). A lot of research has been carried out for this soccer task. In [1], they focus on using both audio and video to detect, for example, the goal (based on the crowd). Another interesting approach focuses on logos and information on the graphics [2]. Most of the currently available approaches have utilized object-motion tracking [3], automatic camera calibration [4], recurrent neural networks [5], 3D convolutional networks [6] and much more. These approaches usually require expensive computations and very big datasets.

SoccerNet [7] is one of the major researchers on this topic, for action spotting in soccer videos. They make use of action recognition models in videos. The dataset is composed of 500 complete soccer games from six main European leagues. Such a dataset weighs hundreds of gigabytes and it is not a good fit for our limited resources. To provide this automatic analysis of a video we tackle different tasks: ball and players detection, shot classification, line detection, team clustering, heuristic extraction of statistics (like game situations, ball possession, penalties and corners), collecting at the end all this information in a dataframe. We analyze videos frame by frame. First, we focus on the detection of the players and the ball, the main information of the soccer game. We have trained different object detection models with two different datasets for images and compared their performances for this task. It's very important to discriminate between the teams and so we clustered them based on the color of the shirts. We enhance strategies to recognize the camera shot, and consequently, the portion of the field, a very important information that was used to improve the next steps. After that, we apply many different transformations on the wide view frames to preprocess the image and then detect field lines with Hough Transform and the results are refined with geometrical considerations. With all the information collected with the previous steps, we are able to extract some relevant statistics. At the end of this paper, we will also confront the problems we found and provide some possible solutions for future work.

II. PLAYERS AND BALL DETECTION

We start from the detection of the players and ball in each frame, the fundamental features of a soccer game. We prefer to use YOLO because it's more suitable for video analysis w.r.t. other object detection architectures. It is also well maintained and continuously updated.
Specifically, we have trained three different models: Yolov5s, Yolov5m, Yolov8m. The differences between the two Yolov5 [8] models are mainly in the number of parameters (7.2 M and 21.2 M respectively), while Yolov8 [9] presents differences in the architecture. Each of these models were trained with a dataset of soccer game images (1914 for train, 502 for validation, and 250 for test) with labeled players and ball, covering shots of all kinds (i.e.: side view, midfield view, close up). We created this dataset combining these two found online [10] [11]. All of these networks were pre-trained on a COCO dataset and then fine-tuned with our one. The number of epochs was 100, setting early stopping with a patience of 5. In order to evaluate the performances after training we have made a quantitative analysis considering the mean average precision metric

(specifically mAP 0.5 and mAP 0.95) to choose the best network.

	Best mAP50	Best mAP95
YOLOv5s	0.7902	0.5096
YOLOv5m	**0.8329**	**0.5575**
YOLOv8m	0.779	0.5343

Table 1. Yolo results showing the best mAP achieved during training.

Based on the results shown in Table 1, we have decided to use the YOLOv5m network. An example of detection is shown in Figure 1. For some more details of the training and validation of these network see the Section A of the Appendix.

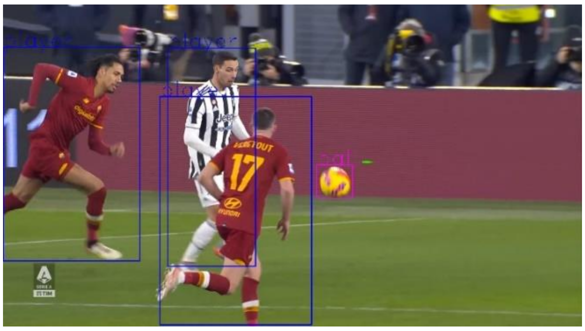


Fig. 1. Example of detection (blue boxes for players, pink box for the ball)

III. TEAM CLUSTERING

The idea behind recognizing which team the player belongs to is inspired by real life. Players have different shirts, with different colors. To this extent, we look at the shirts of the players. In principle, we apply a Gaussian Blur to remove noise. Starting from the detected Bounding Boxes, the first step is to extract the specific part of the bounding box that we assume is the shirt, starting from the center of the box [figure 1]. We compute the three histograms with OpenCV [12], concatenate them and then normalize for all the players detected. To compute the "distance" between the histograms, we decided to use the bhattacharyya distance.

$$D_b = -\log \sum_i \sqrt{p_i \cdot q_i}$$

A K-Means clustering is performed based on this metric with a number of cluster centers that is 3 by default (to recognize team1, team2, and referee). Since the goal is to apply this model to videos it's important that the cluster centers found in the first frame are saved for the future detections. This means the clustering has to start in a situation with all the 3 classes visible (to this extent it's best that the video starts with the kick off, but it's not mandatory).

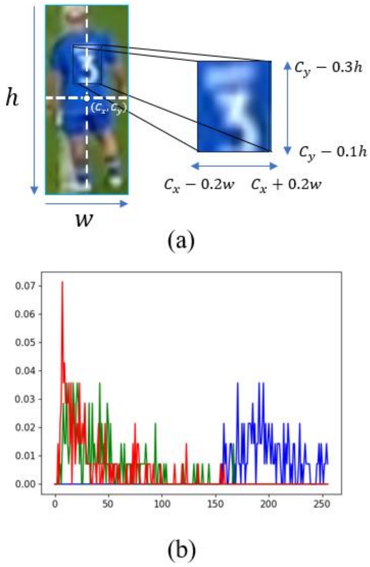


Fig. 2. (a) Extraction of the shirt window with measures (b) Histograms of the window

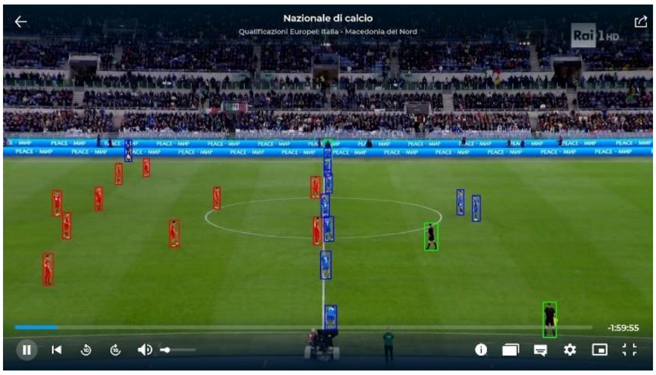


Fig. 3. A good example of clustering at the Kick-off

For the subsequent frames, the histograms are compared with the cluster centers saved before, and each box is assigned to the cluster with the minimum distance. We took 40 different images of kickoffs and we have decided to evaluate this method in this way: if more than three players are misclassified, the prediction is wrong, otherwise it's correct. There is also another distinction, between acceptable clustering (the teams were clustered correctly but poor detection of the referee) and perfect (all the roles matched perfectly).

TABLE I. KICK-OFF CLUSTERING

Perfect	Acceptable	Bad
14	14	12

The method was good enough in 70% of the cases. We noted that the clustering performs better when the images have high quality and shots are near to the pitch compared to others. An Example of initial clustering is in Figure 3.

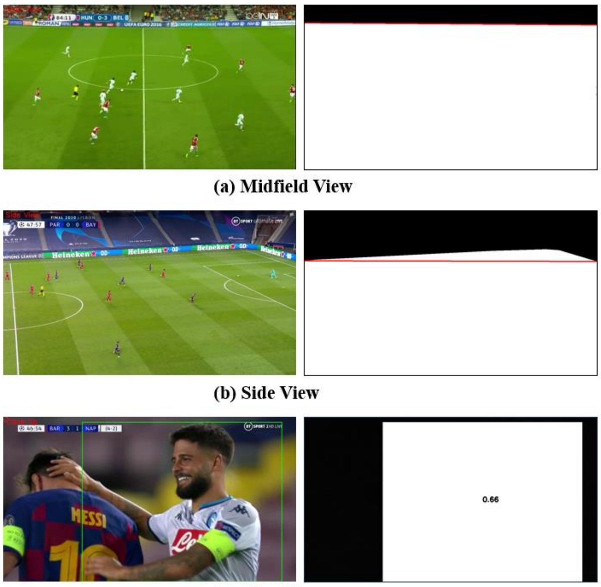
We also insert a control in our code. If two clusters have not at least 6 elements, the initial clustering will be discarded and repeated. During the videos, the clustering remains mostly consistent, in wide angles, while in close ups the accuracy of the clustering is lower because in this kind of shot often the body isn't fully displayed or the player is in strange positions (like for example running fast, bent over etc…)

IV. SHOT CLASSIFICATION

The next step is to understand which portion of the field we are looking at based on the camera view. For this task we took the fantastic work of [13] and applied some personalization.

The first important thing to do is to recognize which part of the image represents the pitch. Knowing that the dominant color of the soccer pitch is green we blur the image using a Gaussian blur, we convert the color space of the image from RGB to HSV, and then compute the color histogram to find the dominant color. In particular, the algorithm looks at the peak of the histogram and also considers the neighborhood bins, which allows us to understand firstly which is the relevant color in the image and secondly, as a consequence, to distinguish a wide camera angle to a close up. If the main color is different from green it surely is not a wide view shot, so the image is classified as close up. Differently, when green is the main color, the algorithm proceeds to perform a color segmentation of the field and keeps only green pixels filtering out other colors, usually from the fan stands. In some images, the audience can occupy a big part of the image, but their colors usually have a high variance, so even in this case, the peak will be that of the field, which is more homogeneous. The image is thresholded to obtain a binary map, with white pixels belonging to the field. Some post-processing operations are applied, including morphological opening and closing operations, computing then the contours choosing the one that encloses the greatest area. Then the contour is approximated with cv2.approxPolyDP and cv2.convexHull.

With the field segmentation map we are able to distinguish if we are looking at a Midfield View (camera centered on the field), or a Side View (camera displaying the left or right part of the pitch) by computing the presence of a corner. We start by computing the line that joins the two uppermost intersections between the black and white separations at the edges of the image. To know if there is a corner in the field segmentation map, we compare the number of white pixels on each part of that line; if the ratio is greater than a threshold or the slope of line is greater than another threshold (that we manually choose by looking at the results in our test dataset), we classify the image as Side View. Otherwise, we classify it as a Midfield View. The slope of the red line also tells us if we are on the left or right side of the pitch.



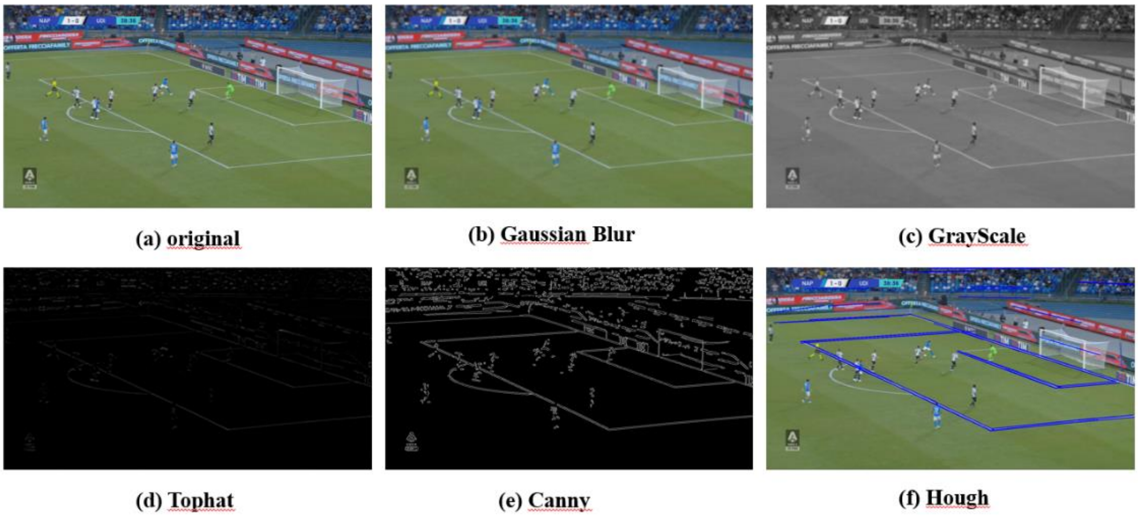
(a) Midfield View

(b) Side View

(c) Close Up

Fig. 4. (a) Midfield View Frame and (b) Side View frame, with on the right their respective field segmentation and red line, (c) a Close Up, on the right an image displaying the ratio of the bounding box of the player with respect to the area of the image

The work presented in [13] only determines the camera view of the images as described before; however, we used this method also to provide, as mentioned before, the close-up camera view detection. In particular, if green is not the main color, the frame is classified as close-up, but also, before the shot analysis steps, we compare the bounding box area of players detected with the all image area, and if the ratio is greater than a threshold (tested exactly like the previous one), the shot is classified as a close up. Three examples of this shot classification are shown in Figure 4.


(a) original

(b) Gaussian Blur

(c) GrayScale

(d) Tophat

(e) Canny

(f) Hough

Fig. 5. Preprocessing Pipeline and Hough Application

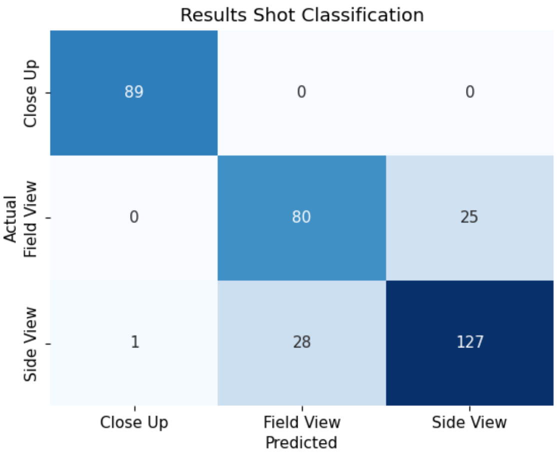


Fig. 6. Confusion Matrix representing the results of this algorithm on a test of 350 manually labeled images

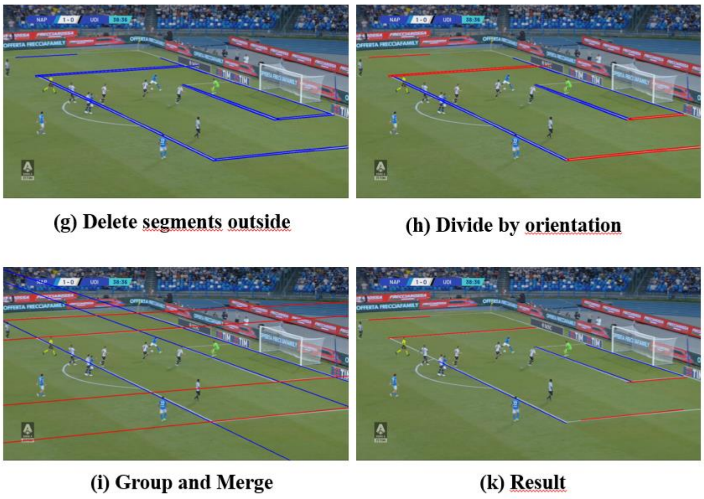
V. LINES DETECTION

Field lines detection is a difficult task to achieve because, as mentioned in the introduction, our dataset consists of frames taken from real game matches, meaning images with a lot of noise and depend also on the quality of the video itself.

The knowledge of the field lines is important for action recognition and some event detection, because it allows us to provide geometric considerations to apply to the next steps regarding statistics, like the discrimination of events and the classification of only the players inside specific areas of the pitch.

Since our strategy is to recognize the field lines using the Hough Transform technique, which is good at detecting simple lines in images, we previously performed some pre-processing techniques to improve the input images and obtain better results.

The preprocessing pipeline is shown in Figure 5. We apply a Gaussian blur to remove some noise, we convert the image to grayscale and then the Top-Hat transform to highlight brighter objects relative to their background, in our case the white lines of the field. After that, Canny is applied to detect edges. Lastly, we use Hough transform to get the principal lines.


(g) Delete segments outside

(h) Divide by orientation

(i) Group and Merge

(k) Result

Fig. 7. Geometrical Refinement of Hough Detections

It is visible after the Hough Transform that some lines are detected outside the field, and some others are overlapped. To refine our detections, we apply some geometrical filtering, shown graphically in Figure 7, to maintain only the segments we are interested in.

Thanks to the color map segmentation mentioned in the previous section, we remove the segments that have the starting point or the ending point outside the field mask. Knowing the slope of those segments we can also distinguish them by orientation (vertical or horizontal) (h). The overlapped lines are filtered to get only one line with a simple strategy: first we extend the segments to become lines. Then, if two lines are very near each other and they have the same slope, we maintain only the longest one (i). Lastly, to know intersection points that define the areas (like the penalty area), we extend the segments until the borders of the image. Assuming that the maximum number of vertical lines should be three, we know in a Side View that the space inside the first and the last is the penalty area region.

VI. Statistics

From all the information that we have collected we can now extract some interesting statistics in a heuristic way.

A. *Ball Possession*

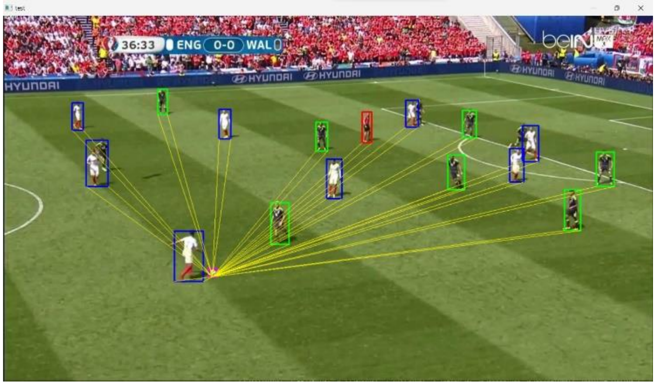


Fig. 8. Image showing graphically how the distance is computed. Basically the ball is assigned to the player which is connected with the shortest segment.

Ball possession statistics is one of the main aspects studied by coaches and soccer clubs to relate the team's performance to the game produced. In soccer, usually, the team with the highest ball possession during a match is the most likely team to win the game; also, it reveals the players' harmony with the coach's playing style.

To avoid the problem of overlapping of players, which generates many potential errors when the ball possession is computed, we take the bounding box of the ball and the bounding box of the players. We calculate the median point of the base of the rectangle of the ball and then we check which player has the external base point nearest to the ball median point. Even using this method, in an overlapping situation it’s difficult to understand which is the real team in ball possession, so we based our consideration on the bounding box distance. The possession is assigned to the team of the correspondent player. Due to perspective, this method works mainly when the ball is on the ground, and it looks at the feet of players.

We tested this method with 250 test images and the possession of the ball was correctly assigned in 243 of them.

B. *Game Situation*

Knowing how many times a team is offensive or defensive during a match can be useful for coaches to understand how comfortable his players are with his style of play and take countermeasures if this shouldn’t be their game approach.

To develop the understanding of which team is attacking or defending, some preliminary knowledge is needed. During the initial team clustering phase, computed at the kick-off, we need to know how the clusters are disposed of in the field and we just look at which cluster is prevalent in the right and in the left (they will be named left_team and right_team). With this information, we know which team should be attacking on both sides. Thanks to the shot classification, we also know which side of the pitch is captured by the camera. Figure 9 depicts the decisional process.

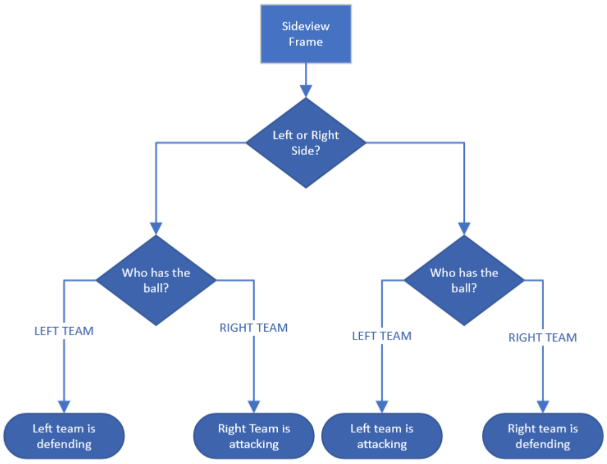


Fig. 9. Decision tree of the game situation description

C. *Penalty*

Penalty represents an important event during a soccer game, it is very distinguishable from all the other events because it has a well-defined pattern on which we define our considerations. In fact, the penalty frame is characterized by two players and the referee inside the penalty area; the ball is clearly visible in the middle of the area. All other players are outside the most external line of the penalty area. Usually, the referee doesn’t get into the area if there is a running offensive play, so it represents important information to avoid potential false positive penalty detections.

Knowing that, the penalty statistic is computed considering only *side view* frames. We compute the distance of the bottom-further corner of the bounding box of each player (i.e. the bottom-left corner if the frame is a *right view* or the bottom-right corner if the frame is a *left view*) from the detected vertical line of the area (that we call area line), further from the side of the frame, i.e. where the goal is. We classify as penalty only those frames where there are only two players inside the area line and if there is also the referee inside it. An example of penalty is in Figure 10, where the area line is drawn in green.

In addition, when the ball is correctly detected, it becomes another discriminant check to do; only frames with the ball inside the area line are classified as penalty. If the ball is not found, it isn't evaluated for classification.

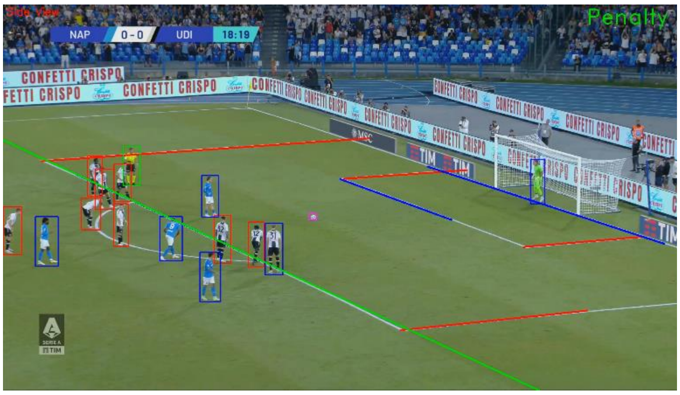


Fig. 10. A Penalty frame well detected. The ball, the referee and only two players are inside the green line (that delimits the area).

D. Corner

Corner statistics is similar to the penalty detection, because the algorithm uses the shot classification, the line detection and the distance of the bounding box of players from the area lines to determine if they are inside it. Differently from penalty, we consider as potential corner frames only those that have a high number of players inside the area (defined by the green line) and one player, different from the linesman, outside the furthest horizontal line, assuming that he's the one kicking the ball in the corner. A corner frame is shown in Figure 11, with the furthers horizontal line displayed in yellow. After some quantitative tests shown in Figure 12 done to understand the best prior threshold value, we decided to classify as corner the frames with at least 10 players over the area line. For now, this method works only if the corner is on the opposite side of the camera position.

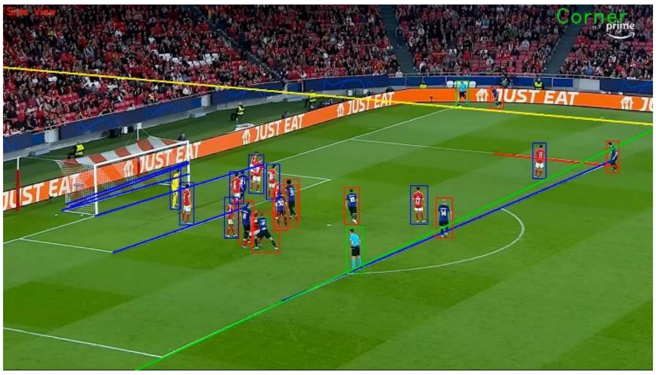


Fig. 11. Corner example. More than ten players are inside the area and one player different from the linesman is over the yellow line.

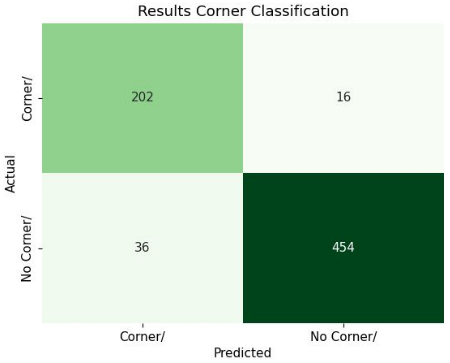
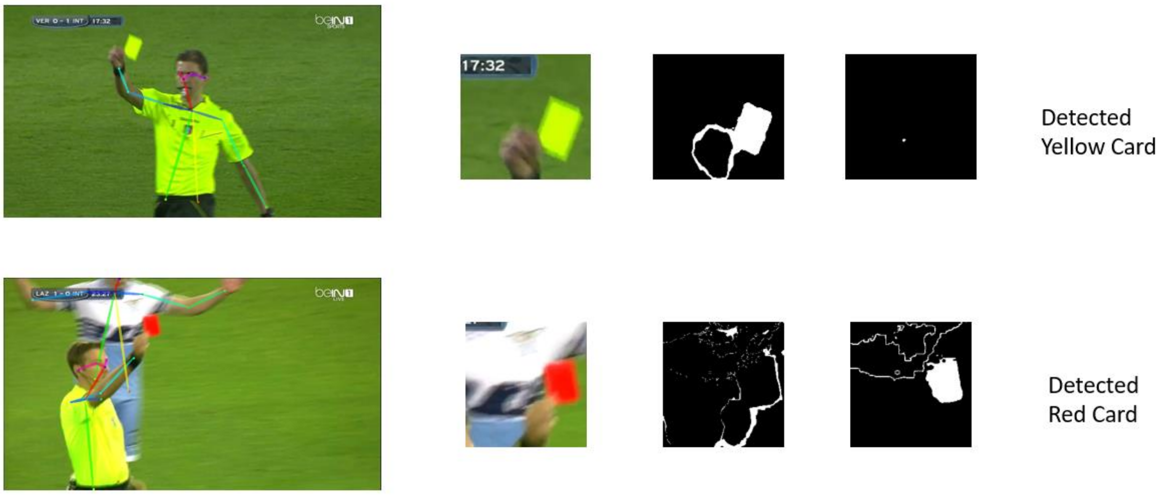


Fig. 12. Confusion matrix of this algorithm on a test set of images with 218 Corners and 490 images of other game situations.

E. Yellow/Red Card

When a referee shows a card, it cannot be seen very well in a wide-angle view. Due to the lack of good datasets for yellow and red cards (these are very rare events too), we tried to use a pose estimation-based approach shown in Figure 13, focusing on the referee gesture. We know that if the referee has the hand over the shoulder, that means that he could be showing a card, or he has just communicated outplay. If we are in the first case, we could analyze the immediate area over the hand and see if the color is mostly yellow, or red, doing a color segmentation, and decide based upon a threshold. Due to the fact that the OpenPose [14] doesn't detect the hand, but only the wrist, it is very difficult to establish a priori which is the area over the wrist that highlights the card in the hand, some assumption could be done based on the knowledge of the size of the bounding box, but we couldn't reach an accurate enough detection, so we didn't integrated it in our final algorithm.



(a) Pose Estimation (b) Wrist Region (c) Yellow Segmentation (d) Red Segmentation

Fig. 13. Yellow/Red card pipeline

F. Goal

The idea of the goal was mainly to assign a goal to one team when the ball crosses the goal line. Despite the simple idea, this method faces some very big problems. First, the ball is usually not detected by our yolo model when it's inside the goal. Moreover, we tried to look at the frames before the event, i.e. when the ball is close to the goalkeeper and still detected, but also in this case we faced some difficulties; we are not able to distinguish when the ball is on the ground and when is flying, so there might be frames where the ball is very close to the goalkeeper and the soccer goal but, depending on a perspective vision captured by the camera, this event might be classified as a *goal event*, when it is not. Figure 14 is an example of this problem.

Detecting the goal remains a difficult challenge with only the information that we collected.

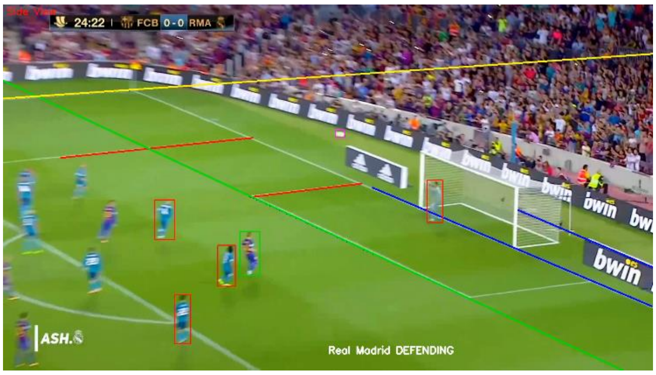


Fig. 14. Perspective problem. The ball seems to be outside the pitch but it’s in the air, going towards the goalkeeper.

VII. FINAL PIPELINE

Putting all the pieces together, we ran the on 15 manually trimmed videos of 25fps with different resolutions of about one minute each, ten of which showing one corner or one penalty. On each frame we have what position of the field we are in, if the ball is detected we have information of ball possession, what is generally happening, and if the right conditions are met, if there is a particular event occurring. The information extracted of each frame with at list one statistic found is saved in a pandas Data frame. The final algorithm pipeline is shown in Figure 15.

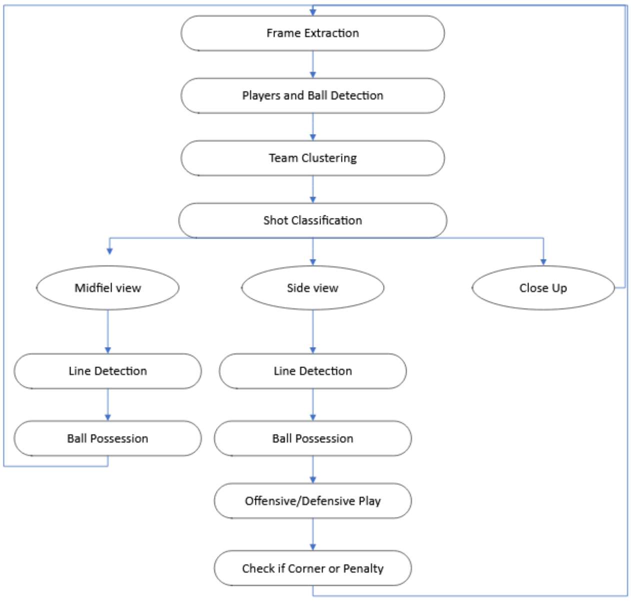


Fig. 15. Final Algorithm pipeline.

To analyze the performances on assigning the major ball possession, we just watched the videos and labelled them with the name of the team that had carried the ball the most. From the data frame we take all the rows assigning the possession on each team and we divide them by the total number of frames showing clearly a possession, finding the possession percentage of each team (discovering also if the team who had the major percentage is the one that we labeled).

To analyze the results on the detections of events (corner and penalty), we manually defined the window of frames in which the event was happening; in both cases the event starts with the first Side View frame showing the positioned players and it ends with the frame where the ball is kicked. The correct event detections were considered as true positives if they were detected inside their specific window, false negatives if the event was not detected inside the window. Every event detected outside the window was considered as a false positive. Results are shown in Table 2.

TABLE II. RESULTS

Possession Accuracy	Corner Precision	Corner Recall	Penalty Precision	Penalty Recall
0.8	0.271	0.703	0.653	0.583

The ball possession was assigned well in most of the cases, even if, when the ball is not on the ground, there could be false classifications.

When there is an event, there is not one case in which there is not a true positive. This means that the events are detected, even if only for just few frames. The rules for the penalty are discriminative enough, most of the time the false negatives are for the referee not well clustered or lines that are not detected well. The rules of the corner instead are not so well discriminative looking at the high number of false positives.

VIII. CONCLUSIONS

Providing a method that covers all the possible situations is a very challenging task. First it is important to notice that the final analysis of a frame strictly depends on the sub-module's accuracy, like for example in Figure 16. Also, the thresholds chosen for Hough Transform aren't generalizing enough. There are also game situations or camera shots that we cannot consider for our algorithm, like the one in Figure 17.

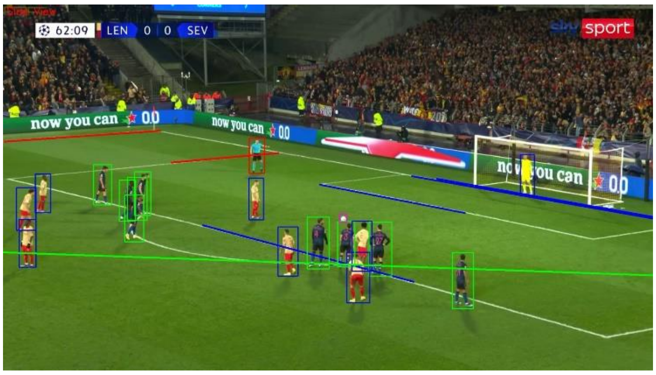


Fig. 16. Hough transform found a line that belongs to the semicircle, the penalty is not detected.

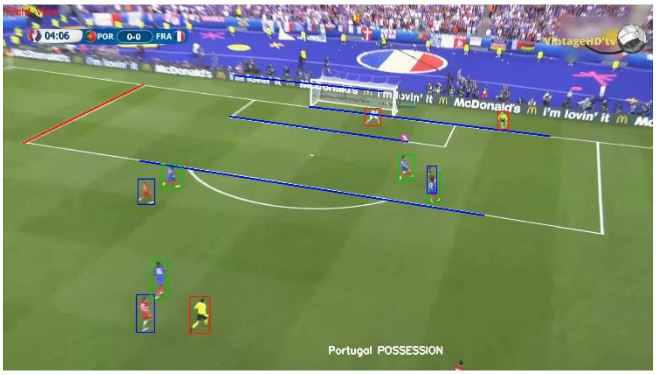


Fig. 17. A shot that is not covered by our algorithm

Another problem that we encounter is the goalkeeper, that has a completely different shirt and that can be of the same color of the one of the referees, and it is not shown at the kickoff. This can lead to some errors, like in Figure 18.

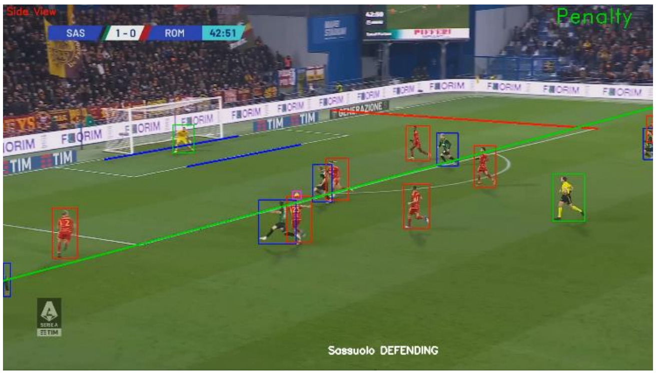


Fig. 18. The shirt of the goalkeeper is yellow as the one of the referee. This frame is wrongly classificated as penalty, because if the goalkeeper is considered as a referee the penalty conditions are met.

Despite these problems, based on the results we can say that this is a good example of the potential of what we can achieve with a bottom-up approach, applying many different computer vision techniques, geometrical considerations, and main knowledge about one of our favorite sports.

REFERENCES

[1] B. Vanderplaetse, S. Dupont. Improved Soccer Action Spotting using both Audio and Video Streams. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2020, pp. 896-897

[2] J. O. Valand et al., "Automated Clipping of Soccer Events using Machine Learning," 2021 IEEE International Symposium on Multimedia (ISM), Naple, Italy, 2021, pp. 210-214, doi: 10.1109/ISM52913.2021.00042.

[3]https://openaccess.thecvf.com/content/CVPR2022W/CVSports/papers/Cioppa_SoccerNetTracking_Multiple_Object_Tracking_Dataset_and_Benchmark_in_Soccer_Videos_CVPRW_2022_paper.pdf

[4] arXiv:2207.11709v2 [cs.CV] 1 Oct 2022

[5] M. Z. Khan, S. Saleem, M. A. Hassan and M. Usman Ghanni Khan, "Learning Deep C3D Features For Soccer Video Event Detection," 2018 14th International Conference on Emerging Technologies (ICET), Islamabad, Pakistan, 2018, pp. 1-6, doi: 10.1109/ICET.2018.8603644.

[6] H. Jiang, Y. Lu and J. Xue, "Automatic Soccer Video Event Detection Based on a Deep Neural Network Combined CNN and RNN," 2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI), San Jose, CA, USA, 2016, pp. 490-494, doi: 10.1109/ICTAI.2016.0081.

[7] Adrien Deliège et al., “SoccerNet-v2: A Dataset and Benchmarks for Holistic Understanding of Broadcast Soccer Videos” 2020

[8] https://github.com/ultralytics/yolov5

[9] https://github.com/ultralytics/yolov8

[10] https://universe.roboflow.com/augmented-startups/football-player-detection-kucab

[11] https://universe.roboflow.com/fyp-v3pnw/fyp-amjew

[12] https://opencv.org/

[13] A. Cioppa, A. Deliège and M. Van Droogenbroeck, "A Bottom-Up Approach Based on Semantics for the Interpretation of the Main Camera Stream in Soccer Games," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, USA, 2018, pp. 1846-184609, doi: 10.1109/CVPRW.2018.00229.

[14] arXiv:1812.08008v2 [cs.CV] 30 May 2019

APPENDIX

Section A

This section shows the training, validation graphs of the YOLO models and the results. The augmentations used were the default one of Yolo:

- Blur (p = 0.01, limit= (3,7))
- Median Blur (p = 0.01, limit= (3,7))
- To Gray (p=0.01)
- CLAHE (p=0.01, clip_limit=(1,4))

The class loss is computed based on the binary cross-entropy loss for the confidence scores of each predicted bounding box.

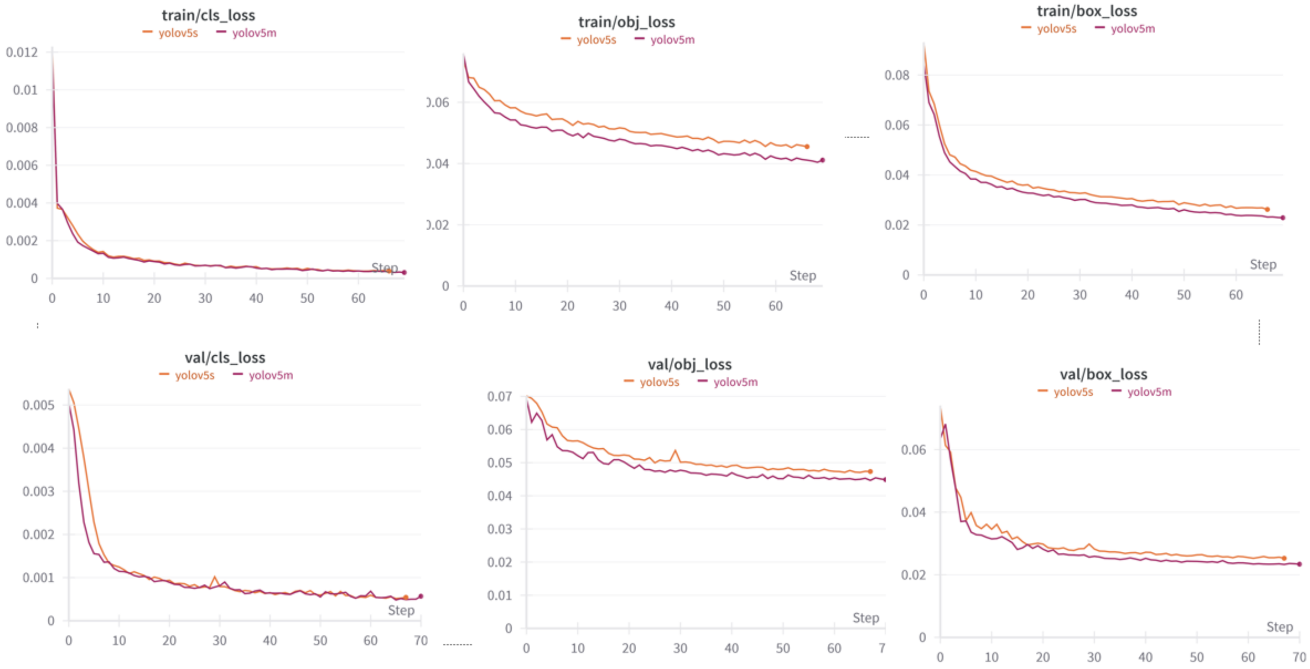
The box loss is summed up over object spatial locations, object shapes and different aspect ratios and is computed as the mean squared error (MSE) between the predicted bounding box parameters and the ground truth ones.

The main difference between the losses of the two models is that the Yolov5 models compute the loss objectness (also binary cross entropy), while the Yolov8 model compute the Distribution Focal Loss

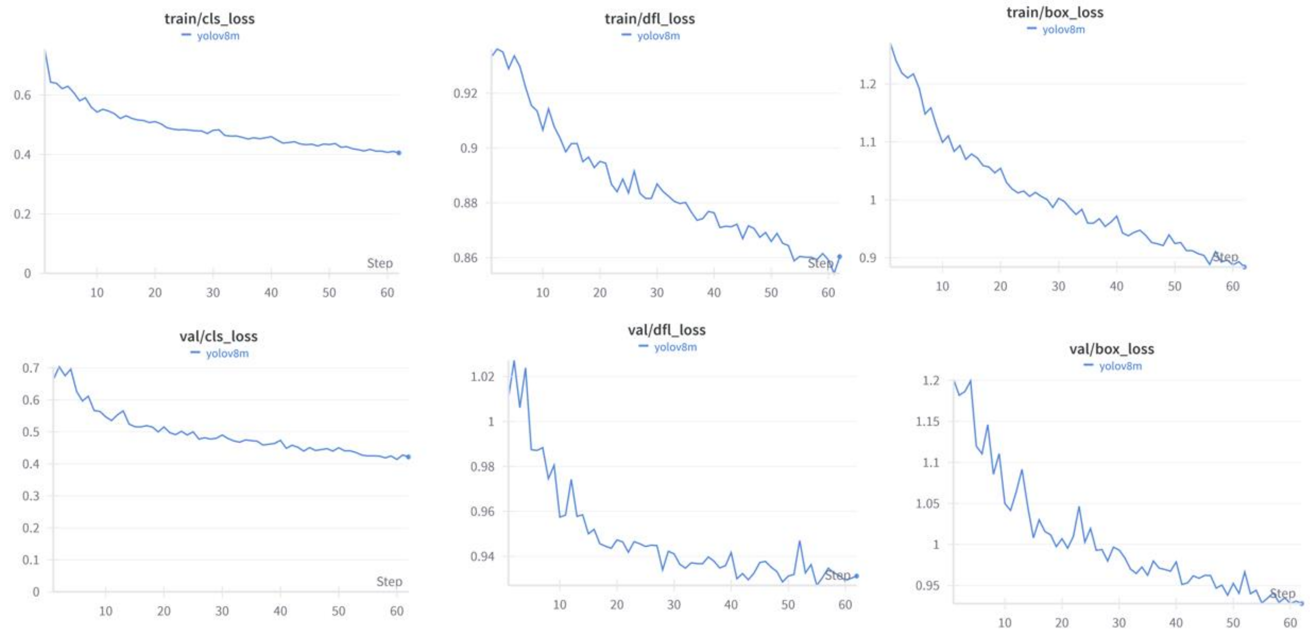
TABLE III. YOLOV5M RESULTS ON THE TEST SET

Class	Precision	Recall	mAP50	mAP50-90
all	0.939	0.777	0.81	0.544
player	0.972	0.971	0.985	0.747
ball	0.906	0.584	0.636	0.341

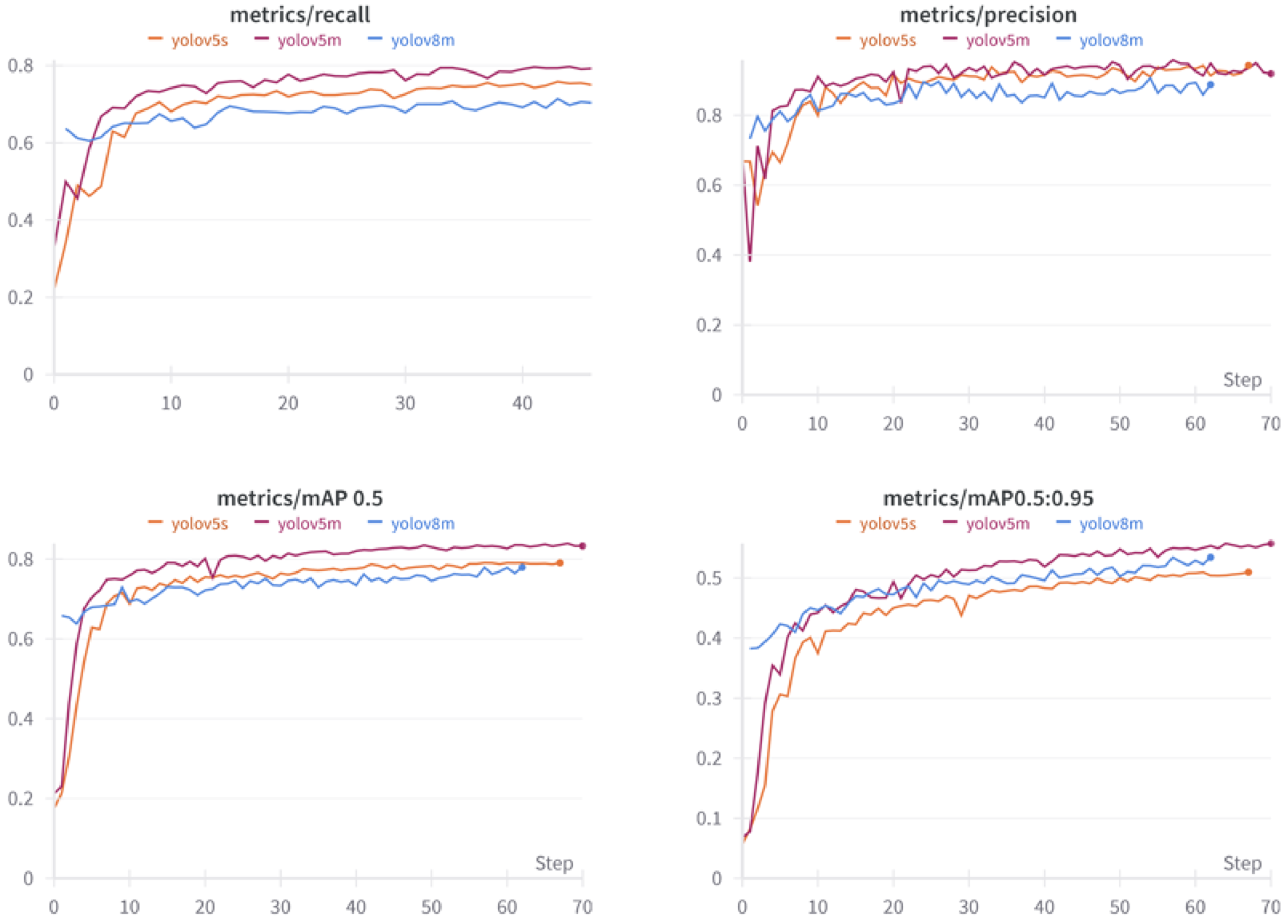
Validating the test set shown that the network performs way better in detecting players than balls.



Training curves of Yolov5s and Yolov5m



Training curves of Yolov8



Metrics Comparison of all the networks tested on the validation set during training.