Virtual Advertising in Soccer Videos

L.W.J. Kanger, *l.w.j.kanger@student.utwente.nl, s1931318, Applied Physics*

***Abstract*—In this work a new method is proposed for virtual advertising in soccer videos. To achieve this, we make use of the known properties of the field lines to construct a white field line pixel detection algorithm. Combined with a Hough transform the field lines are detected and optimised to produce high quality field lines. The homography from a model to the world coordinates is determined based on a manually paired set of intersecting field lines of the initial frame. With this, the proposed method is able to autonomously project an advertisement sign onto all consecutive frames with high precision.**

***Index Terms*—homography, Hough transform, line refinement, virtual advertising**

I. Introduction

IN the current digital era, personal advertising is rapidly increasing and starting to become the dominant advertising strategy. This kind of marketing is really effective and it is then to no surprise that new methods are being developed to improve this or implement this in new areas. One of these new areas, where personal advertising is popping up, are the soccer games that are broadcasted. The video transmitted to all viewers is modified to show them personal advertisements, for example based on the country they live in.

The questions is, how do we place these virtual advertisement signs, because these signs cannot simply be placed on top of the video frames. First of all, placing it inside the field lines will distort the viewing experience and thus create a negative impact on the viewer. Hence, the sign should be somewhere outside of the field lines, but it must also be visible for a significant part of the match. One place that is often used is next to the goal along the back field line, since there is a lot of dead space next to the goal. Secondly, the sign should look like if it was a real advertisement sign placed in the real world in the same spot. This again minimises the distraction of the viewer. To achieve this, the sign should adhere to the real world coordinate system such that it has the same perspective as the soccer field and shows no geometrical differences to a real banner of the same size that is placed in that spot.

The problem, however, is that there is no specific calibration object in the scene (image of the soccer field). Additionally, no information about the camera is given to us unlike in real world examples one might encounter. Therefore we need to somehow find points in the image that correspond to points in a model. This then leads to a projection matrix called a homography that allows us to transform from the image coordinates to the model and vice versa. Fortunately, we do know the dimensions of the soccer field lines, which allows us to find the real world coordinate system. This then allows us to transform advertisement signs from a model onto the frames of the broadcasted videos. Hence, we need a good field line detection system in order to produce high quality virtual advertisement signs.

The outline of the rest of the report is as follows. It starts by stating what the materials used are as well as an comprehensive and in-depth explanation about the different algorithms that make up the proposed method. After that, the results obtained with this method are shown. These results will be discussed as well as the limitations and possible improvements. Finally, a conclusion will be drawn based on the results and discussion.

II. Methods and Materials

A. Materials

In order to test the proposed algorithm, videos of a soccer match are needed. In this case a summary video of the semi-finals of the Champions League 2018-2019, AJAX versus Tottenham, published by Ziggo Sport is used [1]. This video is of high quality, 1080 HD, as well as the soccer field itself. The video contains multiple clips of the goal area where a virtual advertisement sign can be placed. We also need an image of the advertisement sign that needs to be placed next to the goal. This image should not have too much details, since the transformed image will be much smaller than the initial image. Therefore, most fine structures will be destroyed in the process.

The algorithm is written in MATLAB version 2018b, see [2] for the GitHub repository where all the code is posted including examples. Therefore, if one wants to test the algorithm on their own machine, one needs to have a proper MATLAB installation. The code is not tested for any other version and thus it is not advised to use any other version, especially older versions. Some of the image processing and computer vision functions used might have been removed or been reworked in newer versions.

B. Methods

1) Analysis and overview: The goal is to place an image onto the frame in a certain position with the correct perspective projection. To achieve this, we need to find points in the image that correspond to points in the model. This model is a 2D top-down view of the field lines of the soccer field, see Fig. 7. The dimensions of the soccer field are taken from the IFAB laws of the game and are the dimensions for international matches. Given the fact that the model and the real world field lines are both 2D planes, we are looking for a planar perspective mapping. To simplify this, we can set both planes, in their coordinate system, to be the xy-plane, i.e. set $z = 0$. When we represent the selected points in both the world and model in homogeneous coordinates [3] instead of Cartesian coordinates and define $\mathbf{p} = [x, y, 1]^T$ and $\mathbf{q} = [u, v, w]^T$ to be the points in

Fig. 1: The initial frame of an example video. This example video is taking from the video published by Ziggo Sport [1].

the world and model respectively, we get a projection matrix called the homography that is defined as

$$\mathbf{H} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix}. \tag{1}$$

Depending on which way we want to transform, from the world to the model or the other way around, we thus have

$$\mathbf{q} = \mathbf{H}\mathbf{p} \quad \text{or} \quad \mathbf{p} = \mathbf{H}\mathbf{q}. \tag{2}$$

Hence, the transformation direction determines the values of the elements of the homography. From the definition of the homography we can see that we have 8 unknowns, while a point in homogeneous coordinates only has 2 known values. Therefore, we need at least a set of 4 corresponding points to find a homography. However, the more points we have, the better we can fit/estimate the homography. To determine the homography, the MATLAB function *fitgeotrans* with *projective* as transformation type is used.

From this it follows that we need to find a set of corresponding points. This is done by using the field lines. As mentioned, the model is a 2D top-down view of the field lines of the soccer field. For each vertical line, we can calculate its intersection with each horizontal line. The same is done in the image. The field lines are detected and a Hough transform is applied to obtain the individual lines. These lines are divided into vertical and horizontal lines, after which the line parameters are optimised to better fit on the image. Just like for the model field lines, the intersections are calculated of the vertical with the horizontal lines. With these two sets of intersections, one in the image and one in the model, we need to somehow find the corresponding points. There are possible approaches to do this automatically, see for example [4], [5] and [6]. However, due to limited time and resources, these are not tested/implemented and the pairing of the points is done manually. This is only done for the first frame and all consecutive frames are done automatically.

After a set of corresponding points is obtained for the first frame, the initial homography is determined. With this homography we can transform a virtual advertisement sign onto the initial frame. For all the following frames, the lines detected in the previous frame are optimised to fit on the current frame. Using the set of corresponding intersection points, which is still the same set of points but their position in the new frame is slightly different, the homography of the frame is determined. With this transformation matrix, the advertisement sign is warped onto the frame and the result is stored in the form of a video. In Algorithm 1 the developed method is written in pseudo code. To evaluate the method, we simply look at the produced video and determine the quality of the placed virtual advertisement sign.

Algorithm 1: Overview of the developed method.

```
Load the model of the soccer field lines, advertisement
  sign and the video.
Perform the line detection on the first frame of the
  video.
Divide lines into horizontal and vertical lines and
  calculate their intersections.
Manually select pairs of corresponding intersections of
  the model and detected field lines.
for all frames in the video do
    Find the pixels in the frame that correspond to the
      white field lines.
    Refine the lines found in the previous frame to
      match the current frame.
    Calculate the homography using the new lines and
      the known relation between the intersections of
      the detected and model field lines.
    Transform the advertisement sign from the model
      coordinates to the world coordinates.
    Place the advertisement sign onto the frame and
      write it to a video file.
end
Close the video file and write it to the disk.
```

2) Field line pixel detection: To detect the field lines we can make use of standard line detection algorithms. However, we have a lot of information about where the field lines are, what their colour is and on what they are placed, namely the green grass. By using this information we can significantly improve the field line detection.

The first and most obvious piece of information is that the lines are on the playing field. Combining this with the fact that the playing field is the green grass, we can conclude that we are only interested in the grass part of the image.

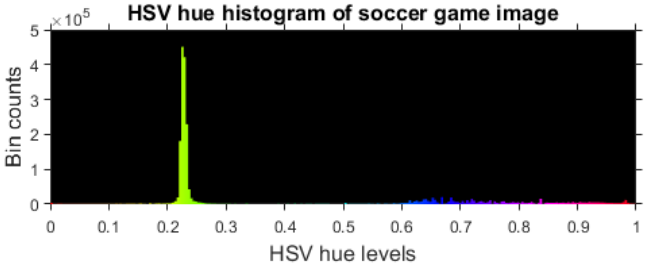
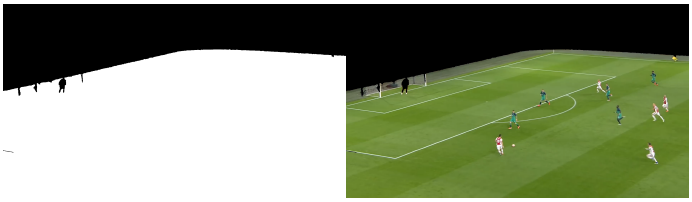
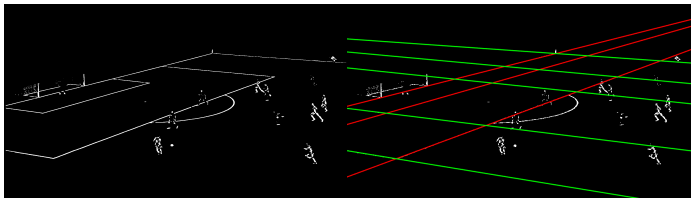


Fig. 2: Colour histogram of the frame shown in Fig. 1 using the hue channel of the HSV colour space. The bars of the histogram have the colour that represents their hue value.



(a) field mask (b) applied to frame

Fig. 3: In (a) the field mask is shown, which is a result of the colour histogram shown in Fig. 2. In (b) the field mask is applied to the frame and shows that indeed only the playing field is left.



(a) Line pixel detection. (b) Optimised field lines.

Fig. 4: In (a) the result of the white field line detection algorithm developed by [4] applied on Fig. 1 is shown. All of the field lines are detected, but there are some false classified pixels. In (b) the detected and optimised field lines are shown. The vertical lines are in marked in red, and the horizontal lines are in green (color online).

Therefore, before any line detection is done we first create a field mask of a frame of the video. This field mask will be used to remove parts of the stadium and boardings that are visible in the frame, such that only the playing field is left. To construct such a field mask we use the dominant colour in the colour histogram of the frame. The reason for this is that we expect most of the frame to be filled with the green grass, since we only consider wide angle shots and no close ups. Therefore, using only those pixels with or close to the dominant colour should give us the desired field mask. The colour histogram is made by looking at the hue level of the HSV colour space. An example histogram is shown in Fig. 2. From this we can see that the dominant colour (the green grass) corresponds to the highest peak in the histogram. After determining the position of the peak, a small range around this peak ($\Delta\lambda$) is picked to produce the field mask with. The field mask is then created by only keeping the pixels that have a hue level in the specified range around the dominant colour, while the other pixels are made black. However, these hue levels are not solely the playing field pixels. In some of the boardings and on the tribune there will also be some pixels that have a hue level in the accepted range. This thus gives some unwanted pixels or noise. To solve this, we make use of the fact that the playing field is one fully connected area and should be the largest. Therefore, when we only keep the largest area, we have filtered all the unwanted noise and non-playing field pixels. After doing this, there are still holes left in the mask that could potentially be on a field line and interfere with the line detection algorithm. These holes are the players that are present on the field due to their non green jerseys. Note, this is something that could be used to place the advertisement sign behind the players instead of on top of them. However, this is not a very good player detection method and thus would not give great results. Nevertheless, the remaining holes are filled and the resulting field mask is then applied on the frame. An example is shown in Fig. 3 to illustrate the method and the result.

Now that we have used the information about the position and background colour of the field lines, we can work on using the information about the colour of the field lines. We know that the colour of the field lines is white. A possible approach would then be to just simply applying a threshold that filters out most of the non white pixels. However, we can enhance this by also using the fact that the lines have a certain width. To do this, we use the method developed by Farin *et al.* [4]. Their method is based on the luminance or brightness of a pixel compared to that of surrounding pixels at a certain distance in the horizontal and vertical direction. In short, consider a pixel of the frame that has a brightness larger than σ_l, which we call the candidate pixel. Next, check the brightness of the horizontal pixels at a distance τ from the candidate pixel on both sides. If the difference in brightness is larger than σ_d, mark the candidate pixel as a field pixel. Next, do the same for the vertical pixels at a distance τ. If both conditions fail, mark the candidate pixel as a non field line pixel. An example result of the algorithm applied on the field masked frame is shown in Fig. 4a. As one can see, all of the field line pixels are detected, however there are some wrong pixels marked as field line pixels (mostly the players). Nevertheless, the result is considerably easier to use for standard line detection methods and thus will produce high quality results.

3) Line detection and optimisation: With the detected the field line pixels, we can apply standard line detection algorithms to obtain a mathematical representation of the lines. For this we used the Hough transform line detection algorithm [7]. After specifying a number of Hough peaks, minimum line length and maximum gap between two line segments, this method gives a list of lines where each line is specified by its distance from the origin (top left corner of the image), ρ, and its angle in the coordinate system of the image, θ. Before the lines are divided into vertical and horizontal, the duplicate lines are removed. These lines are detected as being two individual lines, since they have different values for ρ and/or θ. However, it is possible that these lines represent the same field line and thus when we optimise both lines they will be identical. Therefore, we define two lines to be similar when difference in θ is less than $\Delta\theta$ and difference in ρ is less than $\Delta\rho$. Once two identical lines are detected, only the longest of the two is kept and the other(s) are removed. The longest is kept, since it better represents the detected field line pixels and thus requires less iterations to optimise its parameters.

Once the duplicate lines are removed from the set, the lines are divided into being either a vertical or horizontal line when viewed from above. This is done by grouping the lines based on their value for θ. The Hough transform assigns a negative θ value to horizontal lines and a positive θ to the vertical lines.

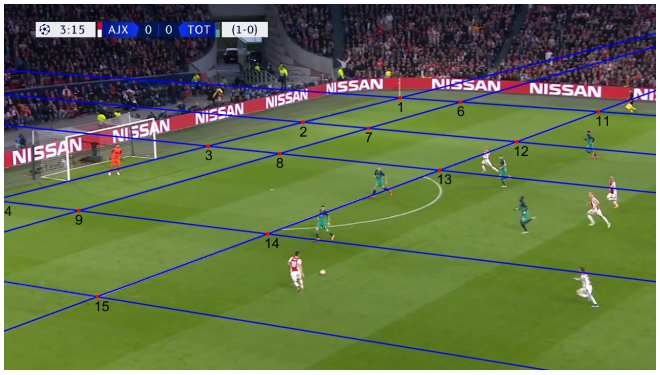


Fig. 5: Result of the line detection algorithm applied on the initial frame shown in Fig. 1. The intersections of the horizontal with the vertical lines are indicated. These points are numbered to be able to manually pair these points with the intersections of the model field lines.

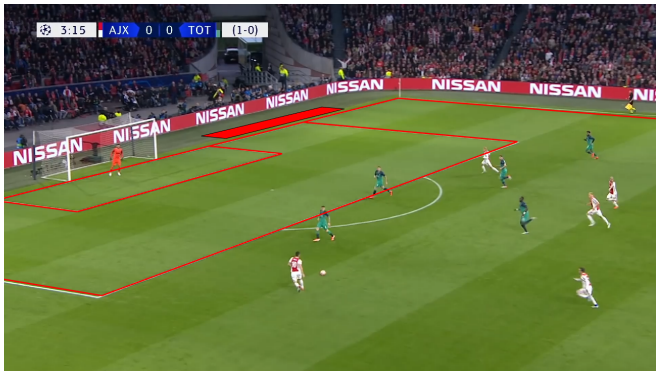


Fig. 6: Model and a virtual advertisement sign warped onto the initial frame shown in Fig. 1. The model lines are indicated in red, while the advertisement sign is the red patch.

Hence, this is used as the filter criterion. After the grouping, the lines are sorted. The vertical lines are sorted from left to right, increasing ρ, and the horizontal lines are sorted from top to bottom, decreasing ρ. Unfortunately, the Hough transform also returns lines that are not actual field lines, for example a goal post or part of the arc on the edge of the goal area. Instead of ignoring these outliers when pairing up the intersection points of the detected lines with the model lines, we made a function that removes these non field line detected lines. This is done by noting that for the vertical lines, the θ values should be decreasing. Any outlier can therefore be detected by a sudden increase in θ. For the horizontal lines, the same holds but for increasing θ values and thus outliers are the sudden decrease in θ. After the sorting, grouping and filtering, the lines in the resulting sets are converted to homogeneous coordinates. This is done, since this is a easier format for the line optimisation algorithm.

After all the processing of the lines, the line parameters of each line are optimised to better fit the detected white field line pixels. The idea of refining the line parameters is taken from [4], however the method we use is more straight forward. For each detected line, either horizontal or vertical, we find all pixels that are within a distance σ_r from the line. Next, we use a linear fit on these pixels to find the new line parameters. This is then repeated multiple times, however, most of the time 3 iterations is sufficient. The result of the whole line detection process on the example frame is shown in Fig. 4b. In the process of grouping the lines, the lines are also sorted based on the location. The vertical lines are sorted from left to right and the horizontal lines are sorted from top to bottom. This is done to be consistent with the numbering of the field lines that is used in the model of the field lines.

4) Advertisement sign placement: The line detection algorithm as described above gives us a set of optimised horizontal and vertical lines. For these lines, the intersections of all horizontal with the vertical lines are calculated. Then, for the first frame, we manually pick the intersections of these lines that correspond with the intersections of the field line model. This gives us a set of corresponding points as well as a set of indices that will later be used to automatically pair the intersections. These corresponding points are used to find the initial homography that describes a transformation from the model coordinates to to the image coordinates of the initial frame.

With the homography, a set of corresponding intersection indices and the detected field lines of the initial frame, we can now find the homography for each following frame automatically. To do this, for every frame in the video, we first detect the white field line pixels using the method described before. Then, instead of doing the whole line detection again, we use the detected field lines from the previous frame and optimise them for the current frame. The reason for this, is that the camera movement between two consecutive frames is very small. Therefore, the difference between two frames is small and the lines detected in the previous frame are close to the actual field lines of the current frame. After that, the intersections are calculated and using the set of corresponding intersection indices we can determine the homography of the current frame. This homography is then used to transform the advertisement sign onto the frame. The resulting frame is then stored in a video file and after all frames have been processed, this video file is stored on the disk.

5) Performance evaluation: To measure the performance of the proposed method, it is tested on several different video clips taken from [1]. Unfortunately, there is no quantitative measure for the produced videos and no ground truth to compare with. The only way to evaluate the performance is qualitatively, which is done by observing the video produced by the method. As mentioned in the beginning, one of the requirements would be that the sign should adhere the same perspective as the soccer field lines in the image, which is something we can only qualitatively check.

A more quantitative measure is the computation time per frame, with the exception of the first frame since it is done manually. In an ideal case, this should be less than the time between two frames to able to broadcast it live. On the other hand, the first frame is done manually. Therefore, the speed is only of interest when this too can be automated.

TABLE I: A list of the parameters including their settings use to produce the example results shown in Fig. 8. The number of Hough peaks, fill gap and minimum length are all parameters used for the line detection using a Hough transform. The number iterations refers to the line optimisation process.

Parameter	$\Delta\lambda$	σ_l	σ_d	τ	#HoughPeaks	FillGap	MinLength	$\Delta\theta$	$\Delta\rho$	σ_r	#Iterations
Value	30	128	20	10 px	11	50 px	125 px	3.5°	50 px	6 px	3

III. Results

In Fig. 5 the result of the line detection algorithm applied on the initial frame of an example video is shown. In this figure the intersections of the vertical and horizontal lines are indicated by the red dots. The points are also numbered to be able to manually pair these points with the intersections of the model field lines. Note, that intersection 4 is not visible in the image, but this intersection is actually calculated and can be used. When we manually select the corresponding intersection points in the model, the homography can be determined. In Fig. 7 the model of the field lines is shown as well as all the possible intersections. Pairing up these intersections results, for this particular example frame, in a homography equal to

$$\mathbf{H} = \begin{bmatrix} 2.2514 & 0.1202 & -0.0003 \\ -2.1982 & 0.1826 & -0.0007 \\ 923.6660 & 162.4660 & 1.0000 \end{bmatrix}. \tag{3}$$

With this homography, we are able to transform a advertisement sign from the model onto the image. Doing so gives the new frame shown in Fig. 6. Here both the model lines as well as a virtual advertisement sign (the red patch for simplicity) are projected onto the frame. In Fig. 8 more examples are shown, this time without the field lines but with a real advertisement sign. All these examples are produced using the same parameter settings shown in Table I.

The computation time of the whole method, except for the manual selection part, is measured for the example videos on the linked GitHub in [2]. This resulted in an average computation time 1.3 seconds per frame using a laptop with Intel(R) Core(TM) i7-7700HQ CPU at 2.80 GHz and with 16 GB RAM.

IV. Discussion

Comparing the placement of the red patch in Fig. 6 with the placement in the model in Fig. 7, we can clearly see that the correct homography is found. The red patch starts and end at the correct places and has the correct perspective. This patch is, in the model coordinates, designed to by 15 meters long and 2 meters wide and is placed 0.75 meters behind the goal line. Comparing these dimensions in the frame to other known distances, such as the goal or field lines, we can conclude that it shows no geometrical differences to a real banner of this size placed in this spot.

The model lines also nicely fit onto the actual field lines, especially near the virtual banner. However, on the bottom left and top right we can see that the projected model lines start to deviate from the actual field lines. This is likely due to having less corresponding points in that region, which introduces errors. Also, when we zoom in on the model line of the goal area that is attached to the arc, we can see that the actual field line starts on one side of the projected model line. Then moves to the other side in and eventually moves back again. This is only possible when either the field line is not perfectly straight in the real world, which I doubt due to the high quality of the field. Or there is a weak, but noticeable, nonlinear lens distortion, which is the most reasonable cause. Either way, this causes errors in the homography, since the intersections of the field lines in the image do not correspond with the actual intersections in the real world. However, from the result we can conclude that these are very small deviations and are not noticed in the projected advertisement sign.

Looking at the example results shown in Fig. 6 we can conclude that the proposed method is working very well. In the first example we can see that the sign stays in the correct position and has the correct projection after 150 frames. This example includes both rotation and zooming of the camera. In the second example, the camera does not move or zoom in, but there are a lot of players that disrupt the visibility of the field lines. As one can see, even with all the players blocking parts of the lines, the method is still able to produce a high quality projective transformation of the virtual advertisement sign after 285 frames. In the third example the camera rotation and zooming are much higher than in the first example and the video is much longer, 225 frames instead of 150. As one can see, this situation is also no problem for the proposed method. The fourth and last example video contains 690 frames and a section of rapid camera movement and zooming. Even for this large amount of frames and these camera events the method produces high quality results no drift is observed.

The examples shown in this report are all videos of the left goal area. This is done due to the fact that the playing field is symmetric. Placing a sign on the right side of the playing field only requires the sign to move in the model coordinates to the desired location. One could also mirror every frame to look like the left goal area and no changes further changes are needed. Hence, we opted not to test the method on this side.

One of the things that could be added would be a player detection algorithm. This method must be able to mark and track all the player pixels in real time. This could then be used to place the advertisement sign behind the players instead of on top of them to make the virtual advertisement sign even more realistic. Currently, the field mask that is obtained by only looking at the dominant colour, so no filling of the gaps, is used to somewhat create this effect. However, this is a rather rudimentary approach and there are way more sophisticated algorithms to achieve much better results, see for example [8].

The main drawback of this method would be the manual selection of the corresponding intersections. However, as mentioned there are solutions to make this part also automatic, see

[4], [5] and [6]. Other options such as automated key-feature detection do not work for this method, since we specifically need the intersections of the field lines to be able to transform from the model to the frames. This would therefore be the main area for further research and focus of development to bring this method to a higher and more competing level.

An other weakness of this method is that only specific video sequences can be used. First of all, the video should contain the goal area at all times. It cannot be a close up of a player or rotate away to the middle part of the field. The reason for this, is the fact that only video clips can be used in which the set of field lines in the initial frame are visible in all frames. This is caused by using the corresponding set of intersection points in the initial frame in all consecutive frames in order to automate the sign placement. However, this can be fixed when the manual selection is automated. This method could then also be used to update the intersection pairs after a few frames are processed.

The proposed method requires a lot of time to process a single frame, excluding initial frame. Having a computation time of 1.3 seconds per frame makes it not competitive to other methods, let alone being able to use this for real time broadcasting which typically is at 60 frames per seconds nowadays. The main time consumer is the line optimisation algorithm. This is caused by the fact that it has to loop through all pixels in the frame (size 1080 by 1920 pixels) for all the vertical and horizontal lines. We have already decreased this by doing the vertical and horizontal lines simultaneously, but further optimisation is possible such that we only have to loop through all the pixels once per frame. However, due to a lack of time and computation time not being our main objective, this is not implemented and/or tested.

Although the method is quite slow and is not fully automated, it can still be used for a highlight/summary video of a soccer match. These videos are not broadcasted live and thus a low computation time per frame is not of high interest. In this case, high quality placement of the virtual advertisement sign is the priority, which is something that this method is able to do. The method could also be used for its high quality and robust line detection algorithm. This is indeed a computationally heavy process, but it does not require any manual labour, apart from setting the right parameters.

Unfortunately, this method is currently is only build for soccer fields and is not tested for different type of sport fields due to limited time and resources. However, the developed algorithms can also be used for other type of sport fields as long as the field line markings are white. The field mask uses the dominant colour in the image to find the sport field. Combining this with the fact that nearly all sport fields use a single colour, this method will still work. The homography calculations, however, do need to be changed. Namely, a different model is needed that reflects the correct sport field instead of a soccer field. Also, the idea of vertical and horizontal field lines may not work for all sport fields, e.g. a baseball field. On the other hand, this idea is only used to reduce the number of possible intersections and reduce the calculations. Hence, the method could in theory also be used for other sports, but this needs some tweaking.

V. Conclusion

The goal of this work was to develop a method that was able to place virtual advertisement signs onto the frames of a video. This sign has to be placed in such a fashion that it has no geometrical differences to a real banner of the same size in that spot in the real world. Based on the results shown in this report we can conclude that this goal is achieved. By using a field mask, detecting the white field line pixels and optimising the line parameters, we were able to build a robust and high quality field line detection algorithm. After manually pairing the intersections of the detected lines with the model field lines for the initial frame of a video, the developed method is able to autonomously project the advertisement sign onto all consecutive frames with high quality. No drift or other artefacts are seen for videos with a large amount of rotation and zooming. The same is true for videos without any rotation and zooming, but with a lot of players disturbing the visibility of the field lines. All the examples shown have used the same parameter values, which indicates the robustness of the algorithm.

Although the quality of the transformed advertisement signs is very high, the computation time of 1.30 seconds per frame makes it not competitive with current methods. Combined with the fact that a manual selection of corresponding points is needed, makes this method not suitable for live broadcasting of soccer games. However, there are options to reduce the computation time, since the main time consuming part can be optimised and even be executed in parallel. Further research and development has to be done to make manual selection also automated. There are studies that show possible implementations, though these are yet to be tested in combination with this work. If successful and the computation time can be reduced to match the frame rate of the videos, this method can serve as a great tool for virtual advertising in broadcasted soccer games.

References

[1] Z. Sport. (2019, May) Ajax vergooit finale in slotseconden — ajax vs tottenham — champions league 2018/19 — samenvatting. Youtube. [Online]. Available: https://youtu.be/w1E8amFdQs0

[2] L. Kanger. (2021) Ipcv-virtualadvertisment. [Online]. Available: https://github.com/LouKanger/IPCV-VirtualAdvertisment

[3] A. F. Möbius and B. A. v. Lindenau, *Der barycentrische Calcul*. Leipzig: Verlag von Johann Ambrosius Barth, 1827.

[4] D. Farin, S. Krabbe, P. With, and W. Effelsberg, "Robust camera calibration for sport videos using court models," vol. 5307, 01 2004, pp. 80–91.

[5] A. Bozorgpour, M. Fotouhi, and S. Kasaei, "Robust homography optimization in soccer scenes," 05 2015.

[6] N. Homayounfar, S. Fidler, and R. Urtasun, "Soccer field localization from a single image," *CoRR*, vol. abs/1604.02715, 2016. [Online]. Available: http://arxiv.org/abs/1604.02715

[7] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, p. 11–15, Jan. 1972. [Online]. Available: https://doi.org/10.1145/361237.361242

[8] P. L. Mazzeo, P. Spagnolo, M. Leo, and T. D'Orazio, "Visual players detection and tracking in soccer matches," vol. 0, 09 2008, pp. 326–333.

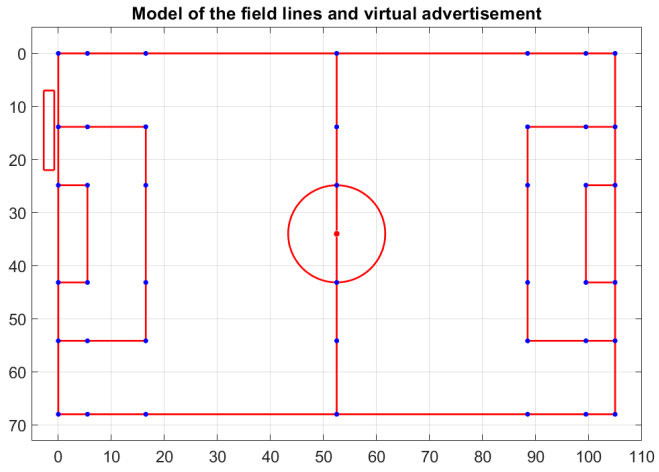


Fig. 7: Model of the field lines using the international standards for the dimensions. On the top left the outline of a virtual advertisement is placed, which later will be transformed onto the image in the same spot. In this image, the blue dots mark all the possible intersections of the field lines. The centre circle is not used and is only for visualisation.

APPENDIX A
MODEL OF THE FIELD LINES

A model of the field lines is constructed according to the international standards. The resulting lines are shown in Fig. 7. All the possible intersections of the lines (expect of the centre circle) are indicated in blue. The outline of a virtual advertisement sign is also shown in the top left. This is a sign of 15 meters long (in the vertical direction), 2 meters wide and is placed 0.75 meters from the closest field line.

APPENDIX B
MATLAB CODE LISTINGS

In this report I did not include the MATLAB code. This is due to the fact that the main code file is 416 lines and uses 11 self-made functions ranging from 10 to 60 lines of code each. Listing all the written code in the appendix does not do the reader nor me any good. Therefore I have uploaded the code to GitHub, which is linked in [2]. Here not only the MATLAB code is listed, but also the example videos used, the produced results and figures used in this report.

Fig. 8: Here the start and end frames of four example videos are shown. Each row is a different video and the left and right column are the start and end frames respectively. Final frames shown are frame numbers 150, 285, 225 and 690 respectively. The settings of the method, the values of the parameters, are identical for all examples and are listed in Table I. In each example the virtual advertisement sign is placed in the same position and and with the same dimensions in the model coordinate system. The sign, in the model, is 12 meters long (in the vertical direction), 3 meters wide and is placed at 0.75 meters from the closest field line.