# Virtual Advertising on Football videos

Image Processing and Computer Vision - 202200103 (2024) - Group - 9

Animesh Jain - s3204472
University of Twente
The Netherlands
a.jain-2@student.utwente.nl

Bambang Muharto - s3268462
University of Twente
The Netherlands
bambangmuharto@student.utwente.nl

Gabriel Gausachs - s3436012
University of Twente
The Netherlands
g.gausachsfernandezdelosronderos@student.utwente.nl

Saurabh Joseph - s3150968
University of Twente
The Netherlands
saurabhjoseph@student.utwente.nl

## ABSTRACT

This paper presents a framework on virtual advertising projection for football videos focusing on techniques from image processing and computer vision. We implemented two different methods for placing a virtual advertisement. First, intersection points in the initial frame were manually selected to compute the homography. Second, we explored the possibility of automatically detecting these points to eliminate the need for human intervention, aiming for a more efficient and less tedious process. After this step, the ad is inserted and a maintained during the video. Our results demonstrate the potential effectiveness of these techniques in enhancing viewer engagement with minimal visual distraction, offering insights for both technical development and advertising strategies.

## KEYWORDS

Homography, Hough transform, virtual advertisement, prespective projection

## 1 INTRODUCTION

The integration of computer vision and image processing techniques has significantly advanced in sports, enabling various impactful applications in sports like football, basketball, cricket, and more. Leveraging these technologies provides deep insights, including the ability to accurately track player positions, extract ball trajectories, animation, and virtual content insertion, bringing sports fans closer to the action with enhanced visual experiences [3]. Among these, virtual advertisement is one of the most impactful applications.

Virtual advertising involves inserting digital ads directly into live broadcasts or recordings. This technique overlays advertisements onto specific areas of the broadcast feed, such as existing ground signage or open spaces on the field or in the crowd. Notably, these ads are only visible to viewers watching on television, spectators at the venue do not see the digitally imposed signs [13]. Virtual advertising reduces the logistical costs and limitations of physical ads, making it a cost-efficient way to integrate advertisements. One major benefit of virtual advertising is its ability to target specific audiences effectively. For instance, ads can be customized for individual country broadcasts during international sporting events. This allows advertisements to align with regional consumer preferences and purchasing habits, as well as comply with country-specific advertising laws, such as those governing alcoholic beverages or tobacco products [12].

In this paper, we aim to find and test a suitable method to place a virtual ad on the World Plane of a football field, and then project this ad onto the Image Plane of each video frame. To follow current industry trends, we positioned the virtual advertisement along the back field line near the goal area, ensuring it appears naturally within the field's perspective without obstructing the viewing experience. Our goal was to make the advertisement look integrated with the World Plane while maintaining an angle of 60 degrees, providing a realistic perspective to viewers on the Image Plane. What's more, in this case, there was no specific calibration object in the scene and no information about the camera is given to us unlike in real-world examples one might encounter.

Taking all this into account, several subtasks were necessary to achieve the main goal. First of all, we had to identify calibration points to estimate the homography matrix, which maps points between the World Plane and the Image Plane. Moreover, the virtual advertisement had to be shown during a video, so we had to find a way to track these intersection points to update our homography in each frame continually. Finally, we had to transform the ad coordinates to align with the perspective of the football field, so that it appeared as if it were present in the scene.

The outline of the report is as follows: it begins with a description of the materials used, followed by a comprehensive and in-depth explanation of the proposed methods. Next, we present the results obtained from these methods, which will be discussed along with their limitations and potential improvements. Finally, we conclude with insights based on the results and the preceding discussion.

## 2 METHODOLOGY & MATERIALS

### 2.1 Data Used

For this project, we sourced videos from a variety of sports channels on YouTube, including those from FIFA [2] and the EFL [1]. The Jumbo logo, obtained from an online source, was used consistently as the advertisement banner throughout our methodology and results sections.

### 2.2 Algorithms and Techniques Used

*2.2.1 Gaussian Blur:* The Gaussian blur technique is used to smooth an image by applying a Gaussian function, which helps decrease

noise levels. This method acts as a nonuniform low-pass filter that retains low spatial frequencies while minimizing noise and insignificant details within the image. Typically, Gaussian blur is implemented by convolving the image with a Gaussian kernel.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

Here, $\sigma$ represents the standard deviation of the Gaussian distribution, with x and y as the positional indices. The value of $\sigma$ influences the distribution's variance around its mean, which in turn dictates the degree of blurring applied around each pixel.[10]

*2.2.2 Canny Edge Detection:* The Canny edge detector is a popular tool in computer vision for identifying abrupt intensity changes and detecting object boundaries within an image. It marks a pixel as an edge if the pixel's gradient magnitude is greater than that of neighboring pixels along the direction of the highest intensity change[5]. Canny edge detection implements hysteresis thresholding which applies two thresholds—high and low—to distinguish between strong and weak edges. Strong edges are preserved, while weak edges are retained only if they connect to strong edges, effectively reducing noise and ensuring continuous edges[4].

*2.2.3 Morphological Opening and Closing:* The opening and closing processes are image enhancement techniques that rely on erosion and dilation to improve visual clarity [9]. Both methods depend on the characteristics of a structuring element—a small, predefined shape that is used to probe the image. This structuring element is systematically placed at various points across the image, where it is compared with the surrounding pixels to influence the processing [11].

Erosion and dilation form the basis of these morphological transformations. Erosion removes pixels from object boundaries, effectively shrinking object sizes and eliminating small-scale noise, while also helping to separate closely positioned objects. Conversely, dilation expands object boundaries by adding pixels, which fill in small gaps and connect nearby elements. Opening and closing utilize these processes in sequence: opening first applies erosion and then dilation to smooth out noise without significantly altering the size of larger objects, while closing performs dilation followed by erosion to fill in small holes and gaps [9].

*2.2.4 Hough Transform and DBSCAN clustering:* The Hough Transform is a fundamental feature extraction method used to detect basic geometric shapes such as lines, circles, and ellipses in images. It accomplishes this by mapping points from the image space into a parameter space, where shapes are represented as intersections or clusters.[8]

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a clustering algorithm specifically developed for detecting clusters in large spatial databases. It employs a density-based method, enabling it to identify clusters of any shape while requiring minimal domain knowledge and simplifying the input parameter selection [6].

*2.2.5 Optical Flow Tracking (Lucas-Kanade):* The Lucas-Kanade algorithm is a method for computing optical flow that assumes neighboring pixels share the same flow values, utilizing a patch of pixels for calculations. It relies on the assumptions of brightness constancy and small motion, making it a sparse optical flow technique that calculates flow at selected pixels, allowing for faster computations. The algorithm can also be adapted for dense optical flow.[7]

*2.2.6 Homography:* The homography matrix, denoted as $\mathbf{H}$, is a $3 \times 3$ transformation matrix that establishes a projective relationship between points in the World Plane $W$ and the Image Plane $I$. It is utilized to map points from one plane to another through perspective transformations, accounting for variations in camera angle, position, and scale.

Mathematically, if $\mathbf{p}_W = [x_W, y_W, 1]^T$ represents a point in the World Plane and $\mathbf{p}_I = [x_I, y_I, 1]^T$ represents the corresponding point in the Image Plane, the relationship can be expressed as:

$$\mathbf{p}_I = \mathbf{H} \cdot \mathbf{p}_W \tag{2}$$

where $\mathbf{H}$ is defined as:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

Here, $h_{ij}$ are the elements of the homography matrix, which are determined based on corresponding point pairs between the two planes. The homography matrix enables the transformation of points $\mathbf{p}_{A_W}$ in the World Plane to their respective points $\mathbf{p}_{A_I}$ in the Image Plane, facilitating accurate overlay and alignment of objects, such as advertisements, within the video frame.

*2.2.7 Perspective Transformation:* Perspective transformation involves mapping 3D world coordinates, such as those on a football field, to a 2D frame, like an image or video. By defining control points on both the 3D space and the 2D projection, it accurately simulates depth and perspective, ensuring that objects like banners appear correctly aligned and proportioned from the viewer's perspective.

## 2.3 Preprocessing and line detection

The first step we had to take was implementing a good line detection to identify the intersection points. To achieve this, we used preprocessing techniques.

*2.3.1 Field mask.* First of all, we knew that the lines are in the grass. The football grass is green so we created a mask to get only the grass from the image and remove parts of the stadium that we weren't interested in. To construct a field mask, we computed the histogram of the hue channel of the HSV colour space and found the maximum value which corresponds to the most dominant color in the image, in our case, the colour green. After this, we picked a small range around this peak and we kept the pixels that had a hue level in the specified range and the others in black. However, these pixels were not only in the grass, there were also some pixels in the stands. To remove these outliers, we kept only the biggest connected area in the image, as it can be seen in Figure 1.

**Figure 1: Field mask**

*2.3.2 Edge detection:* We already know that on a football field, the grass is green and the lines are white. We wanted to detect these lines to then get the intersection points, so after only keeping the area that we were interested in, we implemented a method for edge detection. Firstly, we applied two different Gaussian Blur separately to the grayscale image, one with a kernel size of $5x5$ and $\sigma = 1$ the other of $7x7$ and $\sigma = 2$. Then we used Canny Edge detector with different lower and higher thresholds in each blurred image. With this approach, we achieved to detect small-scale edges in the first blurred image and large-scale edges like field boundaries in the second blurred image. Then, we combined both binary images using the operation *OR* to capture all the relevant features in the image. Finally, we applied closing in the combined binary image to fill gaps and broken edges that were not perfectly detected. The output can be seen in the Figure 2.

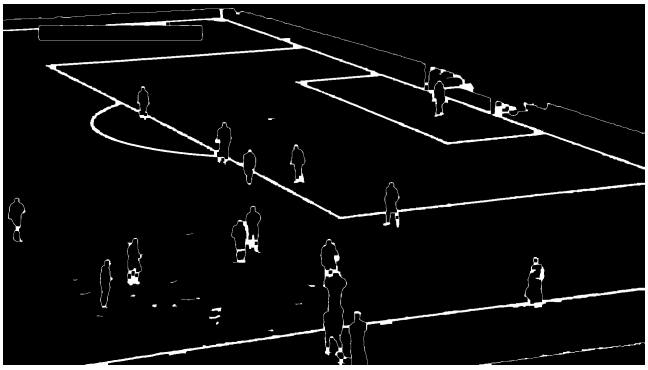


**Figure 2: Frame after edge detection**

*2.3.3 Line detection:* After the preprocessing, the next step was detecting lines in the frame. For that, we used Hough transform.[!!!!] This algorithm is a feature extraction technique used primarily to detect straight lines. For each edge point detected previously, it defines the family of lines that goes through that point as: $r_\theta = x_0 \cdot \cos\theta + y_0 \cdot \sin\theta$, meaning that each pair $(r_\theta, \theta)$ represents each line that passes by the point. After doing this for each point, if the curves of two different points intersect in the plane $\theta - r$, that means that both points belong to the same line. we can define a threshold of the minimum number of intersections needed to detect a line. If it is above the threshold, then it considers it a line. Apart from this parameter, there are others very useful. In our implementation, this threshold was 120, the minimum length that a line had to have was 100 and the maximum allowed gap between points of the same line was 8.

*2.3.4 Intersection points detection:* After detecting lines in the frame, we had to find intersection points. To achieve this, firstly we distinguished horizontal and vertical lines using the angle of the line between the horizontal axis. Then, we found the intersection points between the horizontal and the vertical lines. However, the edges detected in the preprocessing step were thick enough so that the Hough transform detects multiple lines on the same edge. This led to duplicate intersection points. To solve this problem, we grouped nearby intersection points using DBSCAN clustering and kept the mean intersection point. In Figure 3, the intersection points (red cross) and the vertical and horizontal lines (green and blue lines) are shown.

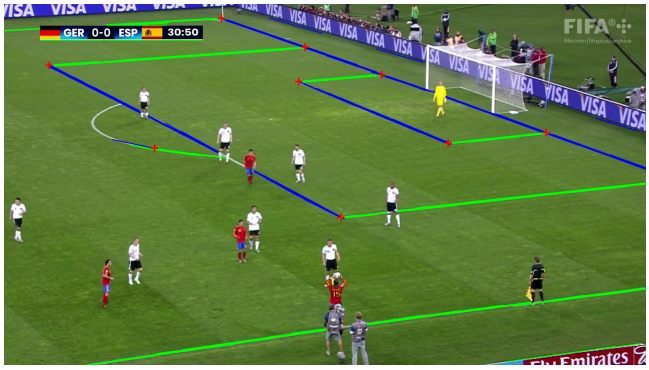



**Figure 3: Intersection points detected**

## 2.4 Key-points selection

Let $P_W = \{p_{W_1}, p_{W_2}, \ldots, p_{W_n}\}$ represent the calibration points on the World Plane $W$, where $n$ is the number of points selected, and $P_I = \{p_{I_1}, p_{I_2}, \ldots, p_{I_n}\}$ denote the corresponding points on the Image Plane $I$. The selection of calibration points was focused on easily identifiable intersections, such as field line corners and penalty box lines, which are consistent across both planes.

We have implemented two processes to get the points on the World Plane $W$ and the Image Plane $I$ to get the points $P_W$ and $P_I$ respectively.

*2.4.1 Manual process:* After preprocessing, we selected a set of calibration points manually on both the World Plane $W$ and the Image Plane $I$ to enable accurate overlaying of virtual advertisements on the football field in the video. The World Plane $W$ is a 2D model of the football field drawn with the accurate standard dimension of an international football field, and the Image Plane $I$ is the plane in the football video clip.

To facilitate precise alignment, we displayed both planes side-by-side, as shown in Figure 5, visually marking corresponding points. We used red dots for $P_W$ and blue dots for $P_I$ in the Figure

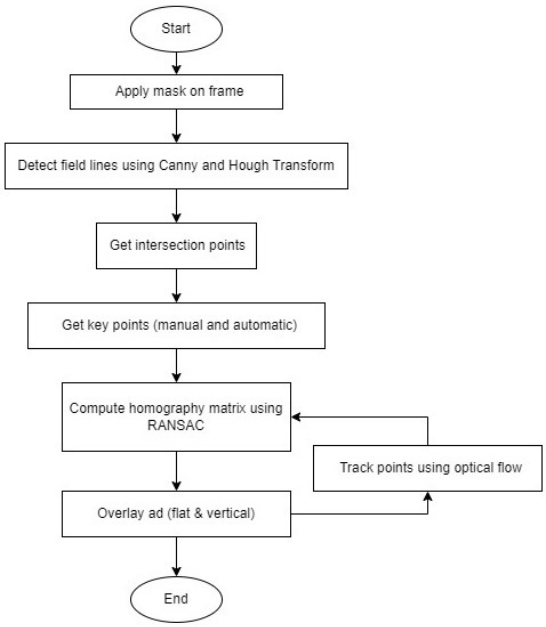



**Figure 4: Flow chart**

5. Following the manual selection process, we refined the Image Plane points to subpixel accuracy, transforming the set $P_I$ into a refined set $\hat{P}_I$ to enhance alignment precision.

To calculate the homography matrix, a minimum of four corresponding points are selected, and can be adjusted to select more points. This requirement arises from the fact that homography is defined by eight degrees of freedom, necessitating at least four pairs of corresponding points to solve for the matrix uniquely.

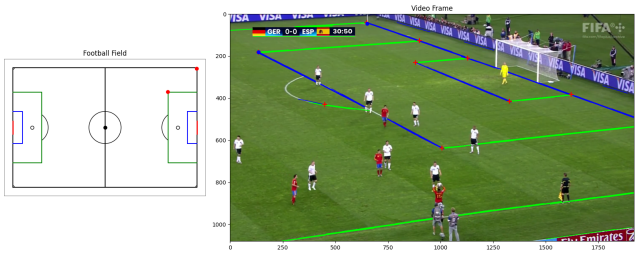
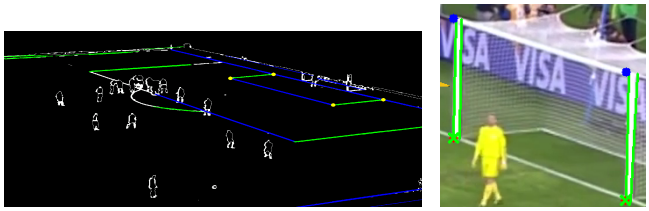



**Figure 5: Manual points selection process**

*2.4.2 Automated process:* After performing the preprocessing and line detection step, we get several intersection points. We filter them out so we get our 2D reference object from the frame which in this case we use a goal line. To achieve this, we use a neighbor-checking approach that examines the spatial proximity of the detected intersection points to ensure they are closely grouped. By setting maximum allowable distances in the x and y directions, we retain only those points that are likely to belong to the goal line.

Once the potential goal line points are filtered, the next step is to order these points to match a specific configuration for further processing. Since the goal line is a straight line with a known orientation, we expect four points: two on the bottom (closer to the camera) and two on the top (farther from the camera). To establish this order, we first sort the points by their y-coordinates, which roughly separates the points into bottom and top pairs based on their vertical positioning. Among the bottom points, we further sort by x-coordinates to distinguish the left and right points, and repeat this for the top points.

After that, we identify four key points on the goal post: two bottom points, obtained using a homography matrix based on known real-world coordinates of goal line, and two top points. We crop the image around the goal post using the bottom points as boundaries. Within this cropped region, we filter for white areas (since the goal post is predominantly white) and use line detection to locate vertical lines. For each bottom point, we find the nearest vertical line and determine the top point by following the line upwards. This setup allows us to align the ad accurately along the goal post structure.



**(a) Goal Line Points** **(b) Goal Post Points**

**Figure 6: Automatic Goal Line and Goal Post Points selection**

## 2.5 Homography calculation

We use either the manual process or the automated process to obtain the key points from both the World Plane and the Image Plane. After obtaining the key points, we computed the homography matrix $H$ as defined in Equation 2, which enables the transformation between these two planes.

To compute the homography matrix, we utilized the OpenCV function cv2.findHomography‘, which employs the Random Sample Consensus (RANSAC) algorithm to robustly estimate $H$ despite potential outliers in the selected points. The function takes as input the corresponding points $P_W$ and $P_I$ and returns the estimated homography matrix $H$.

The use of RANSAC is critical in this context, as it iteratively selects a random subset of the point pairs to estimate the homography, subsequently evaluating the fit of the remaining points to this model. By doing so, RANSAC effectively discards outlier points that could adversely affect the accuracy of the homography calculation.

## 2.6 Advertisement Position in the World Plane

The initial step in overlaying the virtual advertisement within the video frame involved defining the optimal ad position, denoted by

$P_{A_W}$ as a set of points in the World Plane. We selected specific coordinates within the World Plane, opting for either the top-left or top-right region of the grassed area. These locations were strategically chosen to minimize potential interference from players and other on-field activity and based on the standard dimension of an international football field.

## 2.7 Overlaying Advertisement

*2.7.1 Perspective transformation in flat position:* We transformed ad position, $P_{A_W}$, in the world plane into the corresponding position in the image plane, $P_{A_I}$, using the previously computed homography matrix $H$. This matrix $H$ serves as a transformation tool that maps each point in the world plane to the video frame's image plane by accounting for perspective, scale, and camera orientation differences. We applied $H$ to $P_{A_W}$ to obtain $P_{A_I}$, the exact location in the image plane where the ad should be overlaid in the video frame. Notably, this transformation positions the ad flat on the ground, ensuring it appears naturally integrated into the scene and aligns seamlessly with the field's visual elements.


**Figure 7: Video frame used in methodology with advertisement overlay. We used automatic points detection. In this advertisement is laid flat to the ground.**

By computing $P_{A_I}$ through this perspective transformation, we ensured that the advertisement's placement remained consistent with the real-world perspective of the field. This approach allowed the ad to dynamically adjust within each frame, maintaining its intended position and scale as the video progresses while preserving the appearance of lying flat on the ground. We show the overlay ad for the video frame used in methodology as shown in Figure 7.

*2.7.2 Perspective transformation in vertical position.* We first compute new homography matrix from vertical plane using goal post points as reference. With this new homography we get transformation coordinate in vertical plane and it makes us able to place ad horizontally aligned to the surface of the field. Additionally, we also try to get projection matrix ($P$) calculated from 3D-2D point correspondences to make ad vertically aligned to the ground. We tested both of them to transfrom World Place to Image Plane coordinate and place the ad vertically aligned.

## 2.8 Tracking points for recalculating homography

To maintain accurate ad placement across $k$ frames, we tracked a set of manually selected points in the image plane, initially defined as $p_{I_i}^{(0)}$ for each point $i$ in the first frame. As the video progresses, these points move due to changes in camera angle and perspective, making it necessary to continuously track and update them to recompute the homography matrix $H$ and ensure stable ad positioning.

We employed the Lucas-Kanade optical flow algorithm, which updates each point $p_{I_i}^{(t-1)}$ to its new position $p_{I_i}^{(t)}$ in the current frame $t$, based on the intensity patterns around each point. This method is well-suited for tracking sparse, manually selected feature points, preserving their positions frame-by-frame. After updating the points for each frame, we verify that sufficient points (at least four) remain to compute a reliable homography. If so, we use the new points $p_{I_i}^{(t)}$ to recalculate $H$, ensuring that the ad overlay accurately follows the shifting perspective of the video. By dynamically recalculating the homography, this approach preserves the ad's intended perspective, alignment, and visibility throughout the video sequence

# 3 RESULTS

We show the results obtained with the techniques discussed in Section 2 for another video. In Figure 8, the original video frame without the advertisement placed is shown.


**Figure 8: Original video frame without any advertisement**

From this video, we show the advertisement overlayed using manual points detection in figure 9. The homography matrix $H_{manual}$ which was obtained for the manual points selection process is shown in Equation 3.

$$H_{manual} = \begin{bmatrix} 38.52 & 59.13 & 962.15 \\ -2.81 & 14.69 & 484.17 \\ 0.003 & -0.036 & 1.00 \end{bmatrix} \quad (3)$$

**Figure 9: Video frame with advertisement overlay using manual points selection. In this advertisement is laid flat to the ground.**

We tracked the manually selected points to re-calculate the homography matrix for better placement of the ad using optical flow tracking discussed in Section 2.

On the other hand, Figure 10 illustrates the result of overlaying ad with our automated point detection and tracking approach, as outlined in Section 2.

**Figure 10: Video frame with advertisement overlay using automated points selection. In this advertisement is laid flat to the ground.**

The homography matrix $H_{auto}$ which was obtained for the detect points is shown in Equation 4.

$$H_{auto} = \begin{bmatrix} 18.84 & 48.40 & 968.00 \\ 4.92 & -4.65 & 486.00 \\ -0.01 & 0.003 & 1.00 \end{bmatrix} \quad (4)$$

Since optical flow tracking is applied on the detected points, the homography matrix will also be re-computed on different frames.

Finally, Figure 11 shows the result of overlaying a vertically aligned ad using automatically detected points on the goal post as a reference. As described in Section 2, we used the goal post as a vertical plane to align the ad at a 90-degree angle. By applying color masking and edge detection to isolate goal post points, we established reference points, computed the transformation matrix, and ensured the ad appeared upright.

**Figure 11: Video frame with advertisement overlay using automated points selection. This advertisement is vertical to the ground.**

The homography matrix $H_{auto\_vertical}$ is shown in Equation 5.

$$H_{auto\_vertical} = \begin{bmatrix} 8.0807 & -6.2625 & 1403.1327 \\ 2.1193 & -28.0777 & 510.8365 \\ -0.0042 & -0.0033 & 1.0000 \end{bmatrix} \quad (5)$$

The parameters used for different parts of the methodology such as Gaussian blur, line detection, Hough line transform, and DBSCAN clustering are shown in the Appendix A.2.

## 4 CONCLUSIONS

Analysing the results obtained in the Section 3, we achieved to overlay the advertisement in a different video, showing that our implementation is generalized. Firstly, we showed that the ad is placed using manual points selection in Figure 9 correctly in the side of the goal line. With this implementation we can prove that the method worked well.

Then, in Figure 10 we showed that automatically detecting points we achieved also our goal. We proved that our other method worked well.

Finally, in Figure 11 we went further and achieved to overlay the advertisement vertically to the ground. We find this results very positive because we accomplished to provide a realistic perspective to viewers on the Image Plane.

Although the main goal of this project is achieved, we found some limitations during the process due to varying video quality and conditions. Issues such as distortion, zoom, varying camera angles, and the uniqueness of each video made it more difficult. First, the visibility of detection lines diminished when they were far from the camera, as white lines on the field often lacked clarity. Also, camera calibration presented difficulties, as inconsistent camera setups and angles across videos hindered our ability to achieve reliable spatial measurements and accurate line detection. Moreover, since each video is different, using the Hough Transform was challenging because it lacked a universal set of parameters, requiring custom tuning for different videos to ensure accuracy. Finally, we also introduced a method for automatic detection of goal area points; however, in certain videos, these points were either too far from the camera or entirely out of frame, complicating reliable detection. These limitations impacted the consistency of our method across various video sources.

In conclusion, this project successfully demonstrates the integration of virtual advertising into football videos through a combination of advanced computer vision and image processing techniques. By implementing edge detection, Hough Transform, DBSCAN clustering, and homography calculations, we accurately detect and track field boundaries and key points for stable ad placement. The use of both manual and automatic methods for selecting calibration points allows flexibility and improves efficiency, while optical flow tracking ensures dynamic ad stability across frames.

## REFERENCES


[1] [n. d.]. EFL — theEFL. https://www.youtube.com/@theEFL. [Accessed 04-11-2024].

[2] [n. d.]. FIFA — fifa. https://www.youtube.com/@fifa. [Accessed 04-11-2024].

[3] Thulasya Banoth, Mohammad Farukh Hashmi, Neeraj Bokde, and Zaher Yaseen. 2022. A Comprehensive Review of Computer Vision in Sports: Open Issues, Future Trends and Research Directions. https://doi.org/10.48550/arXiv.2203.02281

[4] John Canny. 1986. A Computational Approach To Edge Detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* PAMI-8 (12 1986), 679 – 698. https://doi.org/10.1109/TPAMI.1986.4767851

[5] Lijun Ding and Ardeshir Goshtasby. 2001. On the Canny edge detector. *Pattern Recognition* 34, 3 (2001), 721–725. https://doi.org/10.1016/S0031-3203(00)00023-6

[6] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*. AAAI Press, 226–231.

[7] Vedant Gaur. 2022. Lucas-Kanade Optical Flow Machine Learning Implementations. *Journal of Student Research* 11 (08 2022). https://doi.org/10.47611/jsrhs.v11i3.2957

[8] P. V. C. Hough. 1962. Method and Means for Recognizing Complex Patterns. Ser. No. 17,7156 Claims.

[9] Khairul Anuar Mat Said, Asral Jambek, and Nasri Sulaiman. 2016. A study of image processing using morphological opening and closing processes. *International Journal of Control Theory and Applications* 9 (01 2016), 15–21.

[10] Siddharth Misra and Yaokun Wu. 2020. Chapter 10 - Machine learning assisted segmentation of scanning electron microscopy images of organic-rich shales with feature extraction and feature ranking. In *Machine Learning for Subsurface Characterization*, Siddharth Misra, Hao Li, and Jiabo He (Eds.). Gulf Professional Publishing, 289–314. https://doi.org/10.1016/B978-0-12-817736-5.00010-7

[11] Priya M.S. and G.M. Kadhar Nawaz. 2017. Effective Morphological Image Processing Techniques and Image Reconstruction. *International Journal of Trend in Research and Development (IJTRD)* Special Issue | ASAT in CS'17 (March 2017).

[12] Matthias Sander and Claudia Fantapié Altobelli. 2010. Impact of virtual advertising in sports events. In *Proceedings of the 9th International Marketing Trends Conference*. 21–23.

[13] Paul Turner and Sam Cusumano. 2000. Virtual Advertising: Legal Implications for Sport. *Sport Management Review* 3, 1 (2000), 47–70. https://doi.org/10.1016/S1441-3523(00)70079-9


# A APPENDIX

## A.1 Code

The code to overlay the ad using manually selected and automatically detected key points can be found here: https://github.com/GabrielGausachs/Project_IPCV_UT

## A.2 Parameters

| Technique | Parameters |
|---|---|
| Gaussian Blur | kernel_size = (3, 3)<br>sigma = 1 |
| Canny Line Detection | low_threshold = 35<br>high_threshold = 80 |
| Dilation | kernel_size = (3, 3)<br>iterations = 2 |
| Erosion | kernel_size = (3, 3)<br>iterations = 1 |
| Hough Line Transform | rho = 1<br>theta = $\frac{\pi}{180}$<br>threshold = 120<br>minLineLength = 100<br>maxLineGap = 8 |
| DBSCAN Clustering | eps = 20<br>min_samples = 2 |
| Filtering Goal Line Points | max_x_dist = 450<br>max_y_dist = 50 |

**Table 1: Parameter Settings for Image Processing Techniques**