Football offside tracking

by

Ishrat Jahan Mitu
19101087
Mohammad Hafiz-Ul-Islam Chowdhury
19101067
Alif Rahman
19101256
Sabbir Rahman Bhuiyan
19101012

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

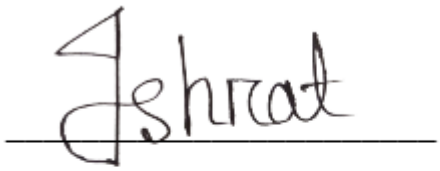
Department of Computer Science and Engineering
BRAC University
September 2022

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

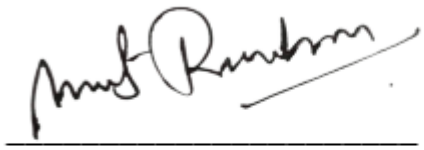


Ishrat Jahan Mitu

19101087

Mohammad Hafiz-Ul-Islam Chowdhury

19101067



Alif Rahman

19101256

Sabbir Rahman Bhuiyan

19101012

Approval

The thesis/project titled "Football Offside Tracking" submitted by

1. Ishrat Jahan Mitu (19101087)
2. Ishrat Jahan Mitu (19101087)
3. Alif Rahman(19101256)
4. Sabbir Rahman Bhuiyan (19101012)

Of Summer, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on September 2022.

Examining Committee:

Supervisor: (Member)

Dr. Md. Khalilur Rhaman

Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

MD Golam Rabiul Alam

Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Sadia Kazi Hamid

Chairperson and Associate Professor
Department of Computer Science and Engineering
BRAC University

Ethics Statement

We now state that this thesis is based on our research findings. All other materials used have been appropriately documented. This work has never been submitted, in whole or in part, to any other university or institution for the award of a degree

Abstract

There haven't been many attempts in football, compared to other sports, to deploy cutting-edge technology to help referees make accurate calls. Modern football tries to use VAR (Video Assistant Referee) technology to correct wrong calls like offside, but this is being delayed because the equipment can't make decisions in real time. The researchers consequently chose to base their determination of offside on tracking. The researchers employed a few algorithms and detection techniques to carry out this tracking. In this paper, we suggest a brand-new real-time offside tracking system using footage from actual football games.. The advantage of our system is that it will check offside and if an offside is found in real-time in the game it will send a notification to the referee's smartwatch. Moreover, no one needs to worry about VAR decisions and waste a match's crucial time. In order to keep the game flowing smoothly, our system will be able to deliver real-time offside judgments. The referee's decisions on violations of the offside rule have a significant impact on the outcomes of football games. The outcome of a football game should be decided by the players, not the officials. Technology should be utilized throughout the game to improve the accuracy of the offside call, not to detract from the ability of the officials. A suggested system uses player monitoring technologies to quantify player positions and an algorithm to spot players who are offside. The likelihood of algorithm mistakes is influenced by the player tracking system's accuracy. One main aspect of our system is to help the referee to make a call by utilizing all the detection, and data that are needed to make the "offside" call. The referee will always make the decisions and our program is going to assist it.

Keywords: OpenCV; VAR; tracking;

Dedication

We wish to devote all of our educational efforts and sacrifices to our wonderful parents, without whom we are useless.
We likewise dedicate our thesis report to Dr. Md. Khalilur Rhaman, sir, who filled in as our supervisor and who mentored us, taught us in the advancement of our abilities and characters as proficient experts

Acknowledgement

All praise to Almighty, without whom our thesis wouldn't have been finished.

Second, we thank our honorable and conscientious supervisor, Dr. Md. Khalilur Rhaman sir, for his help, direction, and advice. He motivated us to do this research and explains it thoroughly.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

BLE Bluetooth Low Energy

ROI Region of Interest

$RSSI$ Received Signal Strength Indicator

TOF Time Of Flight

UWB Ultra-Wide Band

Chapter 1

Introduction

Around the world, there are numerous football competitions at local, national, and international levels as well as in amateur and professional leagues. The world's top competitions include the FIFA World Cup, UEFA Champions League, Copa America, Europa League, Premier League, La Liga, Bundesliga, etc. But there is no bigger and greater competition than the FIFA World Cup. The significance of football games in sports marketing has a big economic influence in Europe and other numerous nations. The key to a football game is scoring a lot of goals Offside is a key rule for maintaining fair play in the contest. Fair rivalry between offensive and defensive players to take control of defending the goal. The linesman, who runs alongside the sidelines, judges' goals, offsides, and fouls. The referee makes the final decision. However, offside is a strategy used by the offensive team to try and score a goal, therefore the lineman must be very attentive to observe it. As a result, it is necessary to provide the linesmen with autonomous technology for detecting offside, as linesmen are often unable to notice vital instances for suggesting judgment to the referee due to limitations in human vision (such as memory, blockage, etc.).

Several sports leagues have recently adopted cutting-edge technology to assist referees, and as a result, complaints about bad calls have decreased. A technology known as VAR (Video Assistant Referee) is currently in use in tennis, baseball, and other sports. Football is one of the most popular team sports in the world because of its straightforward rules and need for little equipment. Football video analysis has received a lot of attention lately, and a variety of prospective applications have been taken into consideration.

Offside is a rule specified by the Association of Football (soccer) that states a player is offside if any body part other than his hands and arms is in the opposing half of the field and closer to the opposing goal line than the ball and the second-to-last opponent. Offside calls are still very subjective for human referees, and they are one of the most difficult regulations to master. Technology in sports has enabled numerous advancements in terms of increased quality, accuracy, and better decision-making across a wide range of sports, from ball tracking to virtual simulations. For this topic of offside detection, computer vision techniques can be used effectively.

Computer Vision techniques can be used efficiently to develop an end-to-end system for automatically finding offsides or to assist referees in making informed calls in

almost real-time regarding offside identification. It will help the linesmen in such a way that, the referee will check the offside situation and then match his decision with the system. In this way, referees can provide better offside calls which will save some crucial seconds or even some minutes in all matches. The combination of humans and technologies can give the best results in this area.

1.1 Research Problem

There is a widespread consensus that football is one of the most popular sports on the world. Football's most crucial component is the goal, and offside is connected to goals. It may alter the outcome of a game. Therefore, finding a match result is quite important. In football, the linesman is responsible for enforcing the offside rule and greatly assists the referee from the sidelines in determining when the ball is out of play. Algorithms for image processing or computer vision are used in a number of projects to determine where specific players are on the pitch and to identify off sides. In addition, numerous sophisticated image processing and computer vision techniques must be used, necessitating a substantial amount of training data that is often only accessible for games of a professional level. In such situations, the processors or projects can come in handy but what about the games that take place in the countryside or in small communities? In the case of this particular problem, the solutions are scarcely practical. If the program is very heavy then it will require a highly configured computer to process everything There won't be any significant difficulties in setting them up for major events like The World Cup, Champions League, Premier League, LA Liga, etc., but it won't be the ideal option for smaller competitions.

One study used the concept of several kinds of wearable sensors to enhance athletes' performance. The study also provided a thorough review of the many kinds of sensors used to record movement, posture, and physiological information, all of which can later be used by athletes to enhance their performance. Offside tracking cannot be employed since offside can be recognized by any portion of the body. Although sensors can be used to determine a player's position on the field, they cannot be used to determine if a player is offside because doing so would interfere with their ability to perform on the field. Consequently, it is not the finest option available either.

Nowadays football is very fast as most of the players are fast enough. Standing in an offside position is not an offense. But players move all the time. As they are good sprinters as well and they not only stay onside but also move into the offside position so it is very difficult to track them all the time. Moreover, the last touch is a key part of offside, so it is very important to track the ball as well. So, mistakes can happen by anyone although they are very professional. Offside is related to goal scoring and it is the most important part of football, so helping the linesmen can play a crucial role in football. As linesmen most of the calls are accurate, if they could get help from the technologies then these offside calls will be almost perfect. As per FIFA, any part of the body, head, arms, or feet in an offside situation will be considered as offside on the opponent's half. So, tracking all these things for a full 90 minutes is very difficult for a person as the games are very fast right now.

Technologies can be a blessing for the linesman as it can help them to assist the full match.

1.2 Research Objective

The purpose of our research is to assist the linesmen throughout the football match and minimize the false call and time delays that can occur by double-checking after the side referee's call and the VAR in big matches. Though our main objective is to help linesmen in small or any type of football matches so that any football match can be played fairly in any circumstances. Because, if it requires a high-resolution video camera and highly configured computer to process information then most of them will try to avoid this technology. For this reason, we tried to make the program simpler and decent video cameras that can capture smooth video will do the work. If we can change the format of the phone camera then phone cameras video can also work here from a very close view. Moreover, many times in football matches wrangles between the playing teams are seen due to offside play and the wrong call by the referee for offside can be seen to lead to serious disputes and fight in rural areas. Although the technology may not be currently used in rural areas due to budget problems but where people can afford it should be able to provide fair decisions for offside and inhibit disagreement and brawls where the referee is unable to do so as after all they are only human. Here, our technologies can be beneficial for them as they will be affordable for them. Advance and improved technology will surely be more accurate, and now football fans do not like the marginal offside decisions. So, a balance must be hit between technology and the decision-making of referees. "Football is looking to the future because this is all about Artificial Intelligence (AI) and data analysis. But, most importantly, it's quick for the fans." [13].

When a goal is scored, every fan applauds in excitement. However, it was deemed offside right away once the VAR was checked, which was quite annoying. Even with the relaxation of VAR offside rules in season 20-21, offside was a factor in one-third of the 48 VAR withdrawals in the English Premier League. And according to the ESPN website, 10 of the 16 goals that were rejected were for offside. As we know, after a call of an offside-by-side referee if a goal scores VAR checking is a must nowadays. VAR is also operated by four to five members of the referee. Here comes our research's main purpose, where we want to try to improve these calls with AI which can give us almost perfect calls instantly without delay, or by VAR which is operated by a group of members. Also, VAR is not a hundred percent accurate as it is checked by humans. Though VAR has made a great impact in Football it cannot be ignored. A huge number of mistakes are reduced after using VAR. But everything is changing with technology. We were introduced by Goal-Line Technology and the latest addition was VAR. Here, Goal-Line Technology is almost perfect yet VAR is a little bit slow and not perfect as it is checked by humans and so there is room for human error and also space for improvement.

In semi-automated offside technology" It's a camera-based system, Holzmuller explained. "These cameras follow the players and track up to 29 data points 50 times per second, which is then analyzed and computed virtually in real-time by the software, using Artificial Intelligence, and provided to the VAR and replay operator."

And "It is based on limb-tracking technology, or as some call it skeletal-tracking technology," Johannes Holzmuller, Football Technology Innovation director, told FIFA's Living Football show. " We term it semi-automated offside because the VAR still has to evaluate and accept the suggested offside line and kick point that comes out of the program before informing the referee on the field." [13]. So, our research can lead us to better technology that can provide a near-instant decision, with improved accuracy, and clear most doubts about a goal being scored while almost fully eliminating the interruption of the game flow caused due to false or time-consuming calls.

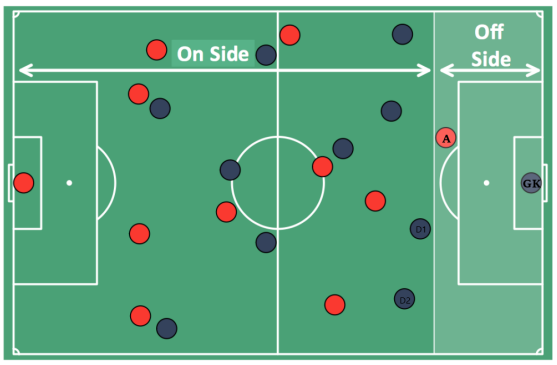
Chapter 2

Related Work

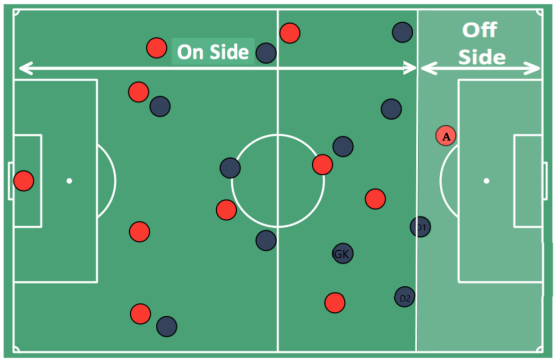
2.1 Problem Defination

One of the rules of association football is offside, which is codified in Law 11 of the rules. According to the rules, a player is considered to be offside if any of their body parts, other than their hands and arms, are on the other half of the field and closer to the opposing goal line than both the ball and the opponent who is in second place (the last opponent is usually, but not necessarily, the goalkeeper). When a player is in the offside position while receiving a pass from a teammate or engaging in active play, it is considered an offside infringement.

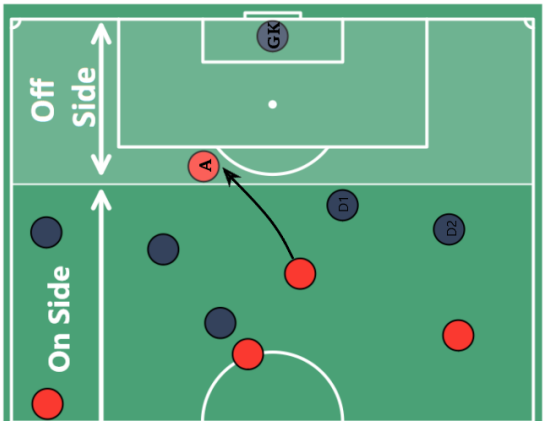
One of the rules of association football is offside, which is codified in Law 11 of the rules. The rule states that a player is offside if any of their body parts—other than their hands and arms—are farther from the opposing goal line than the ball and the second-last opponent (the goalkeeper is typically the final opponent, but not always) [19]. When a player is in the offside position while receiving a pass from a teammate or engaging in active play, it is considered an offside infringement.



Here, we can see in this picture that, team red is in attacking position and team blue is in defending position. In this situation, player A of red team is in offside position as he has crossed the last line of blue team's second last player who is D1. Here, GK(Goalkeeper) is last player of blue team but it is not necessary that a goalkeeper will be the last defender.



In this picture we can see that GK(Goalkeeper) has change his position for some reason, therefore, D1 of blue team is now the last defender and D2 is the second last defender. D2 player's body will be now the last line for offside.



Now, while any attack by team red if player A receives any ball from is team mate it will be consider as offside. Rather than that if other players of team red pass the ball between themselves then it will not consider as an offside.

This is an actual footage of football offside where players of Liverpool are attacking and players of West Hum United are defending. Here we can see that 3 Liverpool players are in offside position.

2.2 Literature Review

The paper [6] used Bluetooth 4.0 BLE technology to conduct research and experiments for the development of an autonomous offside judging system. A Bluetooth-based technology called Beacon, developed by Google is used as a measuring device. Taking advantage of Bluetooth 4.0 BLE multiple of these beacons are paired. Two Arduino-based Bluno receiver devices are used on either side of the goal line. The location of each Beacon is calculated using RSSI signal values created by the connection between the Beacon and the receiving devices. Paired Beacons constantly send RSSI values to each receiver which are used to determine the Beacons' distance from each Beacon, the constant distance between the receivers is considered as the base value and then Heron's formula is applied to determine the height or the orthogonal distance from the goal line to each Beacon. These values are compared to determine offside occurrence which is then relayed to the main referee through a vibration signal to their device. A margin of error of around 50cm has been observed in the case of measuring distances over 5m while testing due to factors such as battery capacity, electronic wave interference and so on which makes this method less reliable for professional football matches and more effective in amateur football matches.

The researchers [5] used computer vision and image processing methods such the Hough transform, color similarity (quantization), graph-connected components, and vanishing point theories to construct a vision-based dynamic offside line marker for soccer games. The play area must first be identified in order to track and identify participants properly. This is done by using Hough transform and taking advantage of the line boundary to differentiate the play area from the audience, signboards, and everything else that is irrelevant to making offside calls. Then using the color information of different team jerseys and image filling, clear localization of the players is obtained. Image opening morphological operation and image dilation is applied to improve accuracy and the strength of player detection. Graph-based connected component labeling is done to find the bounding box for each player. The bounding boxes of each player are tracked using the Kanade-Lucas-Tomasi tracker, which also correlates to minimal motion. The vanishing point is then found by using lines already present on the field. The X intercept of the lines going through each player and the vanishing point is then used to monitor player positions and call an offside when it occurs.

The research work [9] proposes a Deep Learning process for Ball Tracking in football Sequences. To take advantage of deep learning's robustness, the ball detection technique used in this study is based on a deep convolutional neural network. They introduced the creation of trajectories to overcome the difficulties of differentiating the ball from other ball candidates with great confidence. The targeted sequences are long-shot views captured with the primary camera. When a candidate is detected on successive frames, the tracking module begins by using the Kalman filter to create candidate trajectories.

This research paper [2] proposes a machine learning-based event detection and summarization method for soccer matches (ML). They divided the popular video into manageable chunks and utilized neural network (NN) and support vector machine (SVM) techniques to highlight key sections with logo appearance. Additionally, the caption region, which provides data about the game's score, was found using the SVM technique. Using the Hough transform, the two goal posts are detected.

The author of this paper[7] uses contour detection to detect the offside. The author's goal is to determine the position of the farthest defense and attacker while the attacker is receiving a pass from a teammate. To determine the final results, the horizontal positions of the players are contrasted with one another. The authors employed Frame Acquisition, dividing the area into two halves with four cameras and capturing 5 frames per second. In the Image processing methodology, the Authors converted images into HSV and then converted images into grayscale images Because contours can only be found on binary pictures in OpenCV. Contours for the hand are ignored since only goal-scoring components are examined for offside. The x coordinates of these extreme points are compared. If the goal is on the right side of the field, the x coordinate of the attacker is larger than the x coordinate of the defender then the attacker is in an offside position. On the other hand, if the goal is on the left side and if the x coordinate of the attacker is smaller than x coordinate of the defender, then the attacker is in an offside position. The values of Hue, saturation, and intensity should be matched exactly with the color of the jersey which is the only limitation of this methodology.

For football offside identification in this research [12] the authors employed numerous cameras in an automatic linesman assistance system. In this method, a hardware system consisting of numerous cameras and offside detection units is deployed, and offside is notified whenever at least one of the offside detection units detects offside. Vertical lines that depict the attackers, defenders, and ball are used to determine offside. To obtain accurate findings in offside identification, the authors divided the offside problem into a few components and a few conditions. Every time the ball is passed during the competition, it is tracked. The position of the ball is indicated by a vertical line on the half-side boundary nearest to the goal line. When a ball is moved from one attacker to another during an offside situation who is closer to the opposition's goal line than the most-back defender and ball, the offside is immediately recognized.

The feasibility of a multiple camera system for automated offside detection is examined by the writers in this research paper [1]. The author suggested placing 6 fixed cameras on either side of the field (3+3) to reduce perspective and other issues. Consistent images from actual soccer matches were used to support the complete system, and quantitative disclosures on structure precision were attained by separating system responses and ground truth data that was actually gathered on various

deleted relevant plans. Only once, throughout the framework formation, were the six cameras adjusted to process the change grids for the homograph assessment using some appropriately located reference focuses on the field. When the system erroneously offers passive and uncertain events, more effort is required to improve performance in challenging situations. The proposed approach fails when forced to make decisions in crowded situations. To separate players in these situations, either more cameras in various locations on the field or more thorough feature analysis may be used.

A computational offside determination method for soccer match photographs is described in this research work [14]. The method for building a quantitative representation of soccer match images using this offside algorithm has also been laid out as a pipeline of Computer Vision tasks. We provide a fresh dataset for testing this idea, which was gathered from the view made available to the VAR. The authors focused on player identification, team categorization, projection of player body parts, and vanishing point identification. Research can enhance the fundamental computer vision tasks of this system, such as posture estimation and team categorization. These two sub-tasks could be completely automated in addition to having their accuracy increased.

A computational offside determination method for soccer match photographs is described in this research work [10]. The method for building a quantitative representation of soccer match images using this offside algorithm has also been laid out as a pipeline of Computer Vision tasks. We provide a fresh dataset for testing this idea, which was gathered from the view made available to the VAR. The authors focused on player identification, team categorization, projection of player body parts, and vanishing point identification. Research can enhance the fundamental computer vision tasks of this system, such as posture estimation and team categorization. These two sub-tasks could be completely automated in addition to having their accuracy increased.

This study [4] offers a saliency-based automatic offside detecting method. Four cameras are set up on either side of the centerline to capture views of players holding a ball. In the situations, saliency is used to evaluate the movement of offensive players who are carrying the ball, and the estimated motion is compared to the locations of the defense players to detect offside. Three recommended processes—the preprocessing process, the primary proposed method, and the decision process—are included in this essay. The photos used in the proposed technique are in grayscale. The primary approach step for computing the map of saliency characteristics recovered by motion estimation is the construction of a saliency map. In the decision-making process, the saliency map model from the training saliency map database is used. This method uses the training saliency map database to determine offside. Salience is a heuristic that, like most others, is quick and effective but not entirely correct. Depending on the circumstance, this may be superior to accuracy. This can be an issue with their job.

2.3 Existing Algorithms

(UWB) signals have a wide operating bandwidth and operate between 3.1 and 10.6 GHz. The devices can exchange more accurate timestamps, which in addition to TOF principles, can provide more accurate ranges, thanks to the signals' short transmission time, which makes them less sensitive to multi-path effects. (TOF) determines the separation between the transmitter and the receiver by multiplying the signal propagation time by the speed of light. To provide correct results, this method necessitates exact timekeeping synchronization between the sending and receiving devices. The notion is used in time-of-flight sensors, which have a tiny laser source and a corresponding sensor to measure the distance.

Beacon is a Bluetooth 4.0 based technology developed by Google. It supports pairing with multiple other beacons. A Bluno or Smartphone can also be paired to calculate a Beacons' location using RSSI signal values created by the connection between Beacon and the receiving devices. Herons' formula is a formula that takes the length of the three sides of a triangle as input and outputs the area of the triangle.

If, a,b,c are the value of the sides of the triangle.
s = (a+b+c)/2.0;
area = (s*(s-a)*(s-b)*(s-c));

In image analysis, the Hough transform is a feature extraction approach. The Hough transform may be used to isolate aspects of any regular curve, such as lines, circles, ellipses, and so on. Although this model form is relatively simple, it is significantly problematic in the case of complex forms owing to noise and shape imperfection, as well as the challenge of identifying vertical line slopes In the realm of computer vision, the Kanade-Lucas-Tomasi (KLT) tracking method is a popular feature tracking algorithm. To achieve precise and reliable tracking findings, it is essential to choose the appropriate warping parameters for the calculation of optical flow between neighboring image frames. KLT algorithm in its' most simple form tries to find a shift that a point of interest might have taken.

Convolutional neural networks (CNN or ConvNet) are a type of Artificial Neural Network (ANN) used most frequently in deep learning to interpret visual data. Because of how closely the connectivity pattern between neurons mirrors the structure of the animal visual cortex, convolutional networks were inspired by biological processes. The network's ability to automatically learn to optimize the filters without human intervention is a significant advantage that requires relatively little pre-processing. One of the most significant and popular estimate algorithms is the Kalman Filter. The Kalman Filter generates hidden variable estimates based on unreliable and erroneous measurements. The Kalman Filter additionally forecasts system state based on estimates from the past.

Chapter 3

System

3.1 Selected Procedures

The cornerstone for computer vision enthusiasts is the open-source computer vision library known as OpenCV. OpenCV offers a number of functions that are particularly useful in constructing computer vision projects. In this paper, we have used OpenCV and we emphasized all our ideas and combined them into a set of ideas where we decided to detect all the players inside the field and set every player a specific number. So that, we can work with those specific numbers in our working process. We will mark the players into a rectangular-shaped box where their whole bodies will be covered into the rectangular box. It will help us to separate all the players as individuals. Moreover, we will separate the players according to their jersey color as the jersey color is always different in a match. Using these features, we can later detect offside by the player that touched the ball last. In this situation, we need to be aware of both the location of the football players on the field as well as which player is in possession of the ball or actively engaged in the play at the time the off-side has occurred.

After that, we will detect the ball to track it as it is very necessary to track the last touch of a player. The last touch is an important part of offside tracking. Similar to player tracking inside a rectangular box we will track the ball inside a square-shaped box. Then, we will track player positions where there will be our region of interest. We will check whether the offensive players are in an offside position or not while checking the last touch of the ball was by which player or which team player. Then we will pass the call whether it is offside or not. Moreover, two kinds of situations need to be studied in real-time constantly and they are the onside position and the offside position. Firstly, player position concerning both the on-side line and the offside line is to be thoroughly studied or monitored all the time. Similarly, ball position concerning the on-side line or the offside line and most importantly the touch of the player according to the position needs to be studied at a time.

Chapter 4

Methodology

The purpose of the proposed offside tracking method using OpenCV and its Hough transform using the ROI method is described in this chapter.The algoritms are detailed described how those works and functions. In order to do so, the proposed model requires designing the process whether it will be offside or not. Fig 1 shows the view of the designed model:

Basically We will try to detect the player and after that the ball and if all of them work fine we will go for the line drawing, pose detection, and based on them we will determine the offside and after offside, we will calculate the the other things mentioned below in the workflow.

Workflow

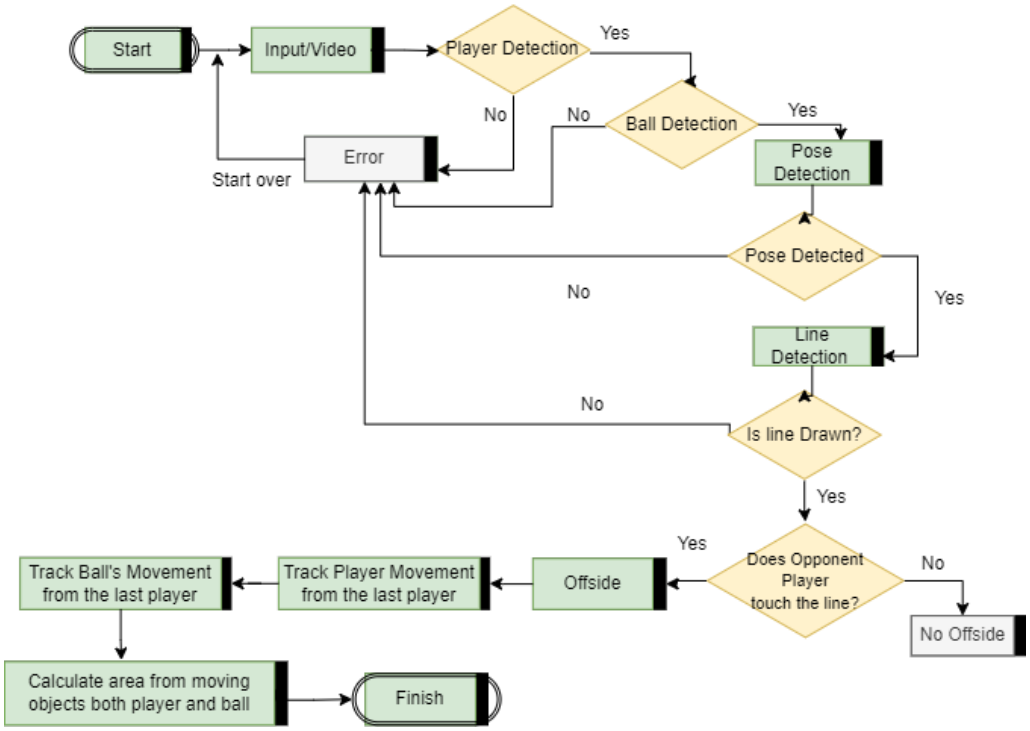


Figure 4.1: WorkFlow Diagram

4.1 Computer vision

The visual system of a human being is both complex and stunning. As an added bonus, Earth is teeming with many forms of life, many of which use strikingly comparable virtual systems. Receptors in the brain, visual cortexes for analyzing visual information, and the eyes for capturing visual data all fall into this category. Genetically engineered and well calibrated system components that improve our productivity. The truth is, though, that this is just the beginning. Over the past 30 years, we've come a long way in our quest to give machines and humans alike the ability to appreciate art with their eyes. The earliest camera, which consisted of a little box containing a piece of paper coated with silver chloride, was invented in 1816. Two centuries later, we have vastly improved versions of the technology that allow for instantaneous digital photography.[15]
The goal of computer vision (CV) is to create artificial systems with similar capabilities to human vision. Data extraction from digital media entails the use of tools like computers to parse through media files. It is also possible to compare and contrast complex images and spot discrepancies.
In several fields, computer vision has become increasingly important. Computer vision is used in many different fields, including optical character recognition, visual bio-metrics, object detection, special effects, 3D scanning and digital photography, sports, intelligent cars, medical imaging, and many more. Ultimately, the goal of computer vision is to make our lives simpler. Users and programmers alike are elevated to the status of computer super heroes as a result of these tools.
Three basic steps how Computer vision works are:

- Acquiring an image

  Real-time visuals can be captured using films, photos, or 3D technology,

- Image Processing

  The procedure can be automated with the help of deep learning models, but only after they have been trained on thousands of photographs that have already been categorized or identified.

- Image understanding

  The last step in the process of interpreting an image is to identify or classify something inside it.

4.2 OpenCV:

An Open-Source collection of programming functions, known as OpenCV, focuses mostly on real-time computer vision. It works wonderfully in image processing and other computer-vision-related applications. The simplicity and readability of the code are the major factors in its appeal. A free library of software for machine learning and computer vision is called OpenCV.[20] In order to speed up the integration of machine perception into commercial products, OpenCV is a software library that was created to standardize the development of computer vision programs. Since OpenCV is a program, it is straightforward for companies to use and modify the code, making it accessible to anyone who wants to obtain the raw data and alter it to their needs. Over 2500 efficient algorithms, including both established and novel machine learning and computer vision methods, are available in the library. These techniques can be used for a wide variety of tasks, including but not limited to: object recognition, characterizing human behavior in movies, face recognition and differentiation, tracking moving objects with a camera, retrieving 3D models of objects, and even creating new 3D models of objects. Examples of 3D modeling include making 3D images of objects, creating 3D point clouds using stereo cameras, combining images to make high-resolution panoramas, removing red eyes from photos of lightning, tracking eye movements, spotting scenery, and overlapping markers, among many other applications. It may be used to carry out operations like face detection, motion detection, location detection, and much more, and it supports many programming languages like Python, Java, and C++. Around 18 million downloads and over 47 thousand users' makeup OpenCV's user base. Governmental organizations, enterprises, and research teams frequently use the library. Intel was the company that created it first. The library is widely used by various startups, including Advanced Minds, VideoSurf, and Zeitera, as well as by major tech companies. It supports C++, Programming languages, Java, and MATLAB languages and includes Windows, Ubuntu, App for ios, and Mac OS. OpenCV is significantly biased towards actual vision applications when the MMX and SSE commands are enabled. Fullfeatured CUDA and OpenCL interactions are currently being created. There are more than 500 algorithms and 10 times as many units that make up or support them. The C++ program OpenCV offers a templated approach that seamlessly interfaces with STL containers. For doing the detection much better in this paper OpenCV is used.

4.3 Hough transform:

In image analysis, the Hough transform is a feature extraction approach. The Hough transform may be used to isolate aspects of any regular curve, such as lines, circles, ellipses, and so on.. Although this model form is relatively simple, it is significantly problematic in the case of complex forms owing to noise and shape imperfection, as well as the challenge of identifying vertical line slopes.

The circle Hough Transform (CHT)'s primary goal is to locate circles in images. Ball is also a circle type that's why in this paper Hough transform is used.

If r is known in CHT,
Hough transform of Circle draws all potential positions of circle center in Hough space for each edge point (x, y) in image space. Given an edge point (x, y) with a given radius r, the collection of all potential circle centers, which is a Hough Transform output, is a circle in Hough space once again.[8]

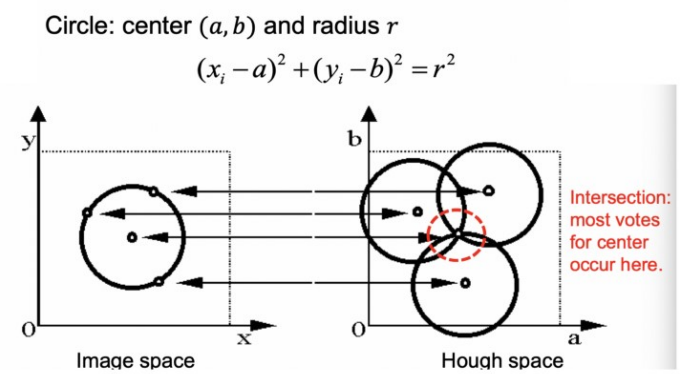


Figure 4.2: known r

If r is not known in CHT,
Hough space becomes 3D space.

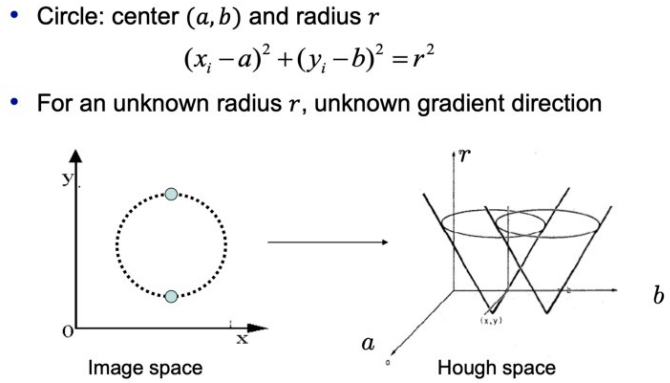


Figure 4.3: Unknown r

Taking the circle characteristic and gradient direction at point (x, y) into consider-

ation, we may conclude that the circle center is most likely located on the line that begins at (x, y) and continues in the direction θ.

Algorithm of Hough circle Transform:[16]

- Initialize the accumulator (H[a,b,r]) to all zeros
- Find the edge image using any edge detector
- For r=0 to diagonal image length
- For each edge pixel (x,y) in the image
- For = 0 to 360
    - a = X -r*coso
    - b = y - r*sino
    - H[a,b,r] = H[a,b,r] +1
- Find the [a,b,r] value(s), where H[a,b.r] is above a suitable threshold value

4.4 Region of Interest(ROI):

ROI basically means Region Of Interest in OpenCV. In OpenCV, Region of Interest is used to execute several common-picture operations. We can work on a rectangular subset of the image with an ROI. To use ROI, one has to follow these steps: construct an ROI on the image, perform the operation one wants to on this subregion of the image, and then reset the ROI. Even if openCV and ROI are similar in some aspects, in the case of OpenCV the origin is at the top-left corner and the ROI is constructed using a correct structure.[11]

In OpenCV ROI or ROIs can be selected by two functions and they are:

1. cv2.selectROI: used for selecting a single ROI on the image which is shown as a bounding box
2. cv2.selectROIS: used for selecting multiple bounding boxes of ROI from an image

Single ROI comes in handy when one needs to specify or crop a single object from the image whereas multiple ROIs work when one selects multiple objects from an image. The bounding box of ROI is by default created starting from the top-left corner. However, if one seeks to have the bounding box from the center then one needs to use the argument “fromCenter = true”

4.5 Kalman Filter

Kalman filtering, also known as linear quadratic estimation, is an algorithm used in statistics and control theory that employs a series of metrics noted over time to

produce estimates of unknown factors that are typically more precise than those based on a single unit of measure alone, despite the presence of statistical noise and other factual errors. Rudolf E. Ka'lm'an, who contributed significantly to the development of the theory, is honored with the naming of the filter.

The Open CV Kalman filter method is implemented or utilized using the following syntax:[18]

```
<KalmanFilter object>= cv.KalmanFilter(dynamParams, measureParams[,controlParams[, type]]
cv::KalmanFilter::KalmanFilter (int dynamParams, int measureParams, int controlParams = 0, int type = CV_32F)
```

When interfacing and operating with extremely noisy systems, the Kalman filter is a very effective tool. The holistic approach of the Kalman filter involves processing noisy input as it enters the system and processing less noisy data as it exits. The Kalman filter has many uses, including algorithm design for tracking devices and objects, data pointing to detect large noisy patches, use in economics, primarily for motion detection and navigation, widely used in many computer and technology-based vision applications, stabilizing measurements for profundities and heights, and feature tracking with widespread adoption and development of stereo-cameras.[17]

Chapter 5

Implementation

5.1 Input Data

For taking input in this model, a video was used of the England VS France game, and some picture was also used. The authors have used different videos close view and run those to make sure to run this system properly.

5.2 Implementation

5.2.1 Player detection

Player detection is one of the most important things for offside detection. The offside basically depends on the players of both team members. For offside tracking, it's obvious to specify the players by their teams. Without specification, it would be hard if one tries to decide whether offside happened or not. Player detection is a must to do the rest of the implementations.
For player detection, The authors decided to specify the team by their jersey colors. To accomplish this part, the authors have used a simple way to detect the players using OpenCV. As input, a video is used. Later on, the morphological operation is used for better player detection

```
thresh = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE, kernel)
```

Then authors find contours of the threshold images. Also, some parameters were used to detect the players from the video. Now, the contours on each frame are detected, and only contours with a greater height than width are evaluated for player recognition. The previous masking technique is repeated, but this time to identify the player jersey colors, and then a rectangular box is built around each player with appropriate text to indicate which team they are in. Masking is used to identify white color while verifying for suitable contour height and breadth while detecting the football.

5.2.2 Ball detection

One of the key elements is to use ball tracking so that one can know the position of the ball in that particular moment where they want it to be seen. Using OpenCV and Hough Transform ball tracking is completed.

The ball is specified by a green circle all over it. Whenever a ball is shown on the screen it will detect the ball by using the hough transform in OpenCV.

```
circles = cv2.HoughCircles(blurred, cv2.HOUGH_GRADIENT, 1, minDist, 100,
param1=param1, param2=param2, minRadius=minRadius, maxRadius=maxRadius)
```

Here, Hough transformation is used to detect the circle or the ball.

```
If circles is not None:
    circles = np.uint16(np.around(circles))
    For i in circles[0, :]:
        cv2.circle(img, (i[0], i[1]), i[2], (0, 255, 0), 2)

cv2.imshow('img', img)
cv2.waitKey(0)
cv2.destroAllWindows()
```

Later on, the circle is drawn after detecting the circle from the image using the parameters of cv2.circle(img, (i[0], i[1]), i[2], (0, 255, 0), 2). Lastly, the result is shown by the cv2.imshow('img', img)

5.2.3 Line Drawing

In the case of line drawing, the task was to identify the goalpost white line outside the d-box. To calculate offside, player detection, ball detection, and line detection need to be included in a package. Only by comparing these 3 can one conclude that an offside occurred in the match or not.

For line detection, the authors emphasized creating a region of the image which needs to be focused on. Other than our necessary line, nothing else matters in the image. As input, both image and video can be used by tweaking the source code a little. First of all Region of interest needs to be defined as a method and in the source code the defined method is:

```
def region_of_interest(IMG, vertices):
    mask = np.zeros_like(img)
    match_mask_color = 255
    cv2.fillpoly(mask, vertices, match_mask_color)
    masked_image = cv2.bitwise_and(img, mask)
    return masked_image
```

First, the image and its vertices are passed to a method called "region of interest," which returns a region of interest based on the image and its parameters. Secondly, np.zeros like() is used to return an array of zeros that are with a similar shape and type as the provided array, afterwards the mask color is set at 255. Next cv2.fillpoly is used to draw a filled polygon similar to the triangle, rectangle, or pentagon of the image and in the parameters are the image and endpoints of the polygon and previously defined color. Mask is a set of ways of merging. If an area of an image contains 0 color in OpenCV or black color it will not be merging the image with the masked image but for the exact opposite case, it will. At last, the newly created masked image is returned to the method.

```
gray_image = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)

canny_image = cv2.Canny(gray_image, 100, 200)


cropped_image=region_of_interest(canny_image, np.array[region_of_interest_vertices],
np.int32)

lines = cv2.HoughLinesP(cropped_image, rho=6, theta=np.pi/55, threshold=600,
lines=np.array([]), minLineLength=45, maxLineGap=800)

image_with_lines = draw_the_lines(image, lines)

plt.show(image_with_lines)

plt.show
```

All of these above-shown codes are used to draw the line using our previously defined region of interest method. Firstly, the image color is changed to RGB2GRAY and then a canny image is created from the gray_image using the parameters (gray_image, 100, 200). Secondly, a cropped image is created using the previously defined region_of_interest method and the parameters include the canny image, the vertices of selected region_of_interest, and np.int32 are used to represent a 32-bit signed integer.

Thirdly, to detect lines HoughLinesP is used to detect the shape in a mathematical form using the given parameters which are shown as rho, theta, threshold, minLineLength, macLineGap, etc. Next, using the given parameters and by creating the shape from the Hough transformation the lines are drawn on the image.

Lastly, using plt.show(image_with_lines) we can see the result of our detection which is shown in the results below.

5.2.4 Object detection Path/Trajectory Tracking

There are several technological uses for Kalman filtering. Popular applications include vehicle guidance, navigation, and control, particularly for ships, aircraft, and spacecraft. Moreover, signal analysis and econometrics typically use Kalman filtering, which is a time series analytic tool. One of the most important parts of autonomous motion planning and control is the application of Kalman filtering to the optimization of trajectories. Modeling the regulation of movement by the brain's central nervous system can also be done using Kalman filtering. The use of Kalman filters offers a realistic model for estimating the present incarnation of a motor system and delivering updated commands because of the latency between delivering control signals and receiving sensory feedback.

There are two stages to the algorithm's operation. For the prediction phase, the Kalman filter generates estimates of the current state variables and their uncertainty. Since the outcome of the subsequent measurement will inevitably be tainted with some error, including random noise, these estimates are adjusted to utilize a weighted average, giving much more weight to estimations with more certainty. The process is cyclical and follows the same method. It can function in real-time using only the most recent input measurements, the known state, and its ambiguity matrix, without requiring any further background information.

Firstly, multiple obstacles to track present extra difficulties, including:

- The proper tracks must be linked to several detections.
- Users have to deal with new objects that occur in a scene.
- When many items combine into a single detection, object identity must be preserved.

The assignDetectionsToTracks function and the vision.KalmanFilter object can help to resolve the issues of:

- Making detection available to trackers
- determining whether a detection is associated with a new object, or track creation
- Prediction can be utilized to assist segregate objects that are adjacent to each other, just as it can in the instance of an occluded specific object.

A background subtraction approach using Gaussian mixture models is used to detect moving objects. Noise is removed from the generated foreground mask using morphological procedures. Blob analysis also finds clusters of linked pixels that are most likely to represent moving things.

Only motion is used to connect detections to the identical object. A Kalman filter estimates each track's motion. The filter is utilized to forecast where the track will be in each frame and to calculate the probability that each detection will belong to a specific track.

This example emphasizes the need for track maintenance. Some occurrences may be allocated to tracks in a given frame, whereas other observations and tracks may not be assigned. The corresponding detections are used to update the allocated tracks. The undesignated tracks have an undetectable marking. A new track is started by an unassigned detection.

Each track maintains a note of how many consecutive frames it went without being assigned. one such example is that it deletes the track and concludes that the object has left the field of view if the count rises over a predetermined threshold.

showing the ball's trajectory by superimposing all of the video frames one on top of the other.

```
function showTrajectory()
 % Close the window which was used to show individual video frames.
 uiscopes.close('All');

 % Create a figure to show the processing results for all video frames.
 figure; imshow(utilities.accumulatedImage/2+0.5); hold on;
 plot(utilities.accumulatedDetections(:,1), ...
  utilities.accumulatedDetections(:,2), 'k+');

 if ~isempty(utilities.accumulatedTrackings)
  plot(utilities.accumulatedTrackings(:,1), ...
   utilities.accumulatedTrackings(:,2), 'r-o');
  legend('Detection', 'Tracking');
 end
End
```

To display the ball's trajectory, assemble video frames, identify locations, and track locations.

```
function accumulateResults()
 utilities.accumulatedImage      = max(utilities.accumulatedImage, frame);
 utilities.accumulatedDetections ...
   = [utilities.accumulatedDetections; detectedLocation];
 utilities.accumulatedTrackings  ...
   = [utilities.accumulatedTrackings; trackedLocation];
end

-Pick the starting position that the Kalman filter will use as an example.

function loc = computeInitialLocation(param, detectedLocation)
 if strcmp(param.initialLocation, 'Same as first detection')
  loc = detectedLocation;
 else
  loc = param.initialLocation;
 end
End
```

These are the some lines from the main code to make the things more clear and show how the functions are working.

5.3 Pose Detection

MediaPipe Posture is a ML answer for high-devotion body present following, construing 33 3D tourist spots and foundation division cover all in all body from RGB video outlines using our BlazePose research that likewise drives the ML Unit Posture Recognition Programming interface. Present status of-the-craftsmanship approaches depend essentially on strong work area conditions for derivation, though our technique accomplishes continuous execution on most current cell phones, work areas/PCs, in python and, surprisingly, on the web.

A two-step locator tracker AI pipeline is utilized in the arrangement, which has been effectively applied to our MediaPipe Arms and MediaPipe Look Cross section items. The pipeline begins by locating the region-of-interest (ROI) for the person or position within the picture using a detector. Through using ROI-cropped frames as input, the tracker then predicts the posture markers and segment mask within the ROI. Be aware that in video use scenarios, the detector is only used in the first frame and when the tracker is unable to identify a body position in the frame before it. For all other frames, the procedure merely creates the ROI using pose markers from the previous frame.[21]

Blaze Pose Detector: The detector is based on our own, portable BlazeFace gadget, which we use in MediaPipe Face Detection as a stand-in for a human detector. It explicitly anticipates two more virtual keypoints, rotation and scale, which firmly define the human body's center as a circle. In the style of Leonardo da Vinci's Vitruvian man, we make predictions about a person's hip midpoint, the diameter of a circle encircling their complete body, and the slope of the line connecting their shoulder and hip centroids.
Pose Landmark Model:

The MediaPipe Posture milestone model forecasts the location of 33 posture tourism attractions.
This is an actual footage of football offside where players of Liverpool are attacking and players of West Hum United are defending. Here we can see that 3 Liverpool players are in offside position.
STATIC IMAGE MODE

If false is selected, the program processes the supplied pictures like a video stream. After attempting to identify the most noticeable individual in the initial photographs, it will proceed to localize the pose markers. To save time and reduce dormancy, it then just watches those milestones in subsequent images without requiring extra identification until something loses track. A series of stagnant, maybe unrelated photos can be processed with ease if the person detection option is set to true. False by default.

- **MODEL COMPLEXITY:** Pose landmark model complexity: 0, 1, or 2. As models become more complicated, benchmark accuracy and inference latency typically increase as well. By default, 1.

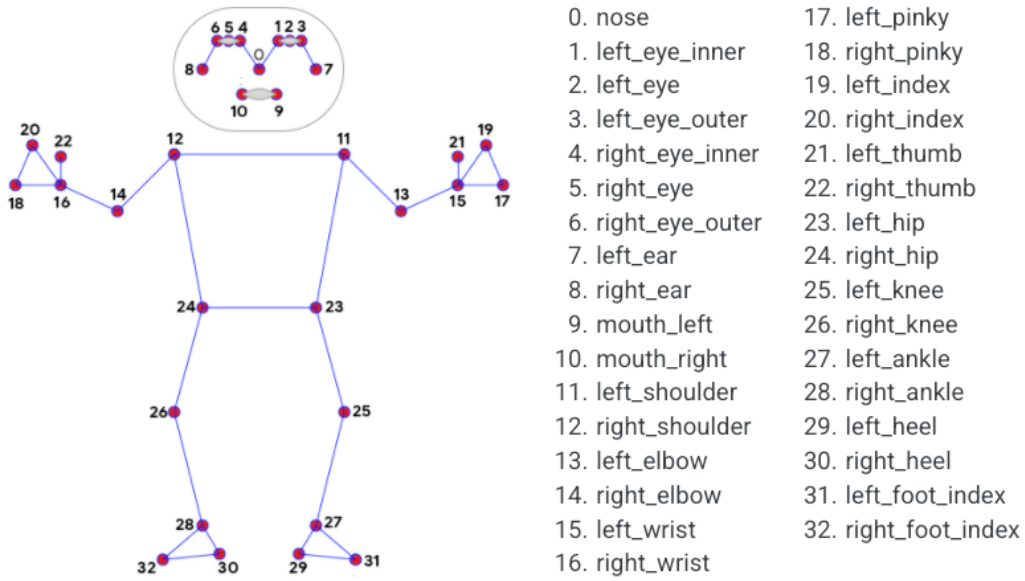


Figure 5.1: Pose landmarks

- **SMOOTH LANDMARKS:** If configured correctly, the solution ignores pose landmarks if static image mode is also configured correctly, but does so to reduce jitter. Usually, true.
- **ENABLE SEGMENTATION:** If set to true, the solution additionally constructs the segmentation mask on top to pose landmarks. False by default.
- **SMOOTH SEGMENTATION:** If true is specified, the approach reduces jitter by filtering segmentation masks across several input images. ignored if static image mode is true or enable segmentation is false. Usually, true.
- **MIN DETECTION CONFIDENCE:** The person-detection model must produce a minimum measure of confidence ([0.0, 1.0]) before the detection can be deemed successful. 0.5 is the default.
- **MIN TRACKING CONFIDENCE:** The pose landmarks must have a sufficient measure of confidence ([0.0, 1.0]) from the landmark-tracking algorithm to be regarded successfully tracked; otherwise, figure detection will be activated automatically on the following input image. By increasing it, the solution's robustness can be improved at the cost of an increase in latency. If static image mode is set to true, person detection is ignored and every image is processed as normal. 0.5 is the default.

Chapter 6

Results and Analysis

6.1 Work Results

6.1.1 Player Detection

In this detection we used England vs France Video as an input and then the author detected the players according to their colors. Figure 2 shows that the two teams are differentiated by their jersey colors.

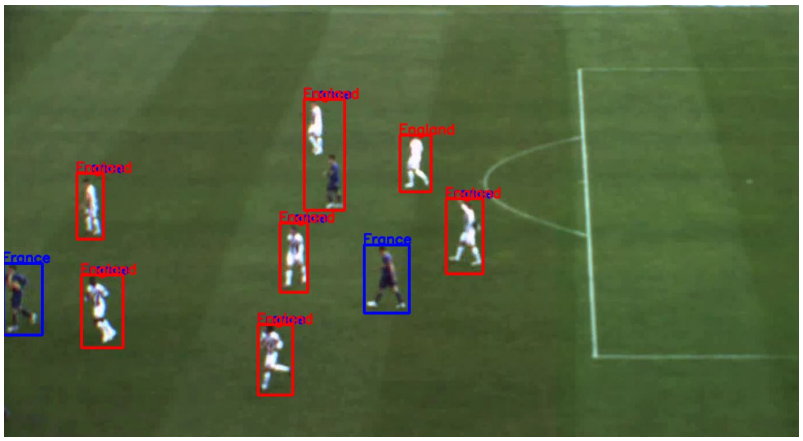


Figure 6.1: Player detection

6.1.2 Ball Detection

Here, an image of a ball is used to detect the ball as an input. Figure 3 shows the before image of the ball and figure 4 shows the image while it is being detected by the detector. After Applying the algorithm on Video :

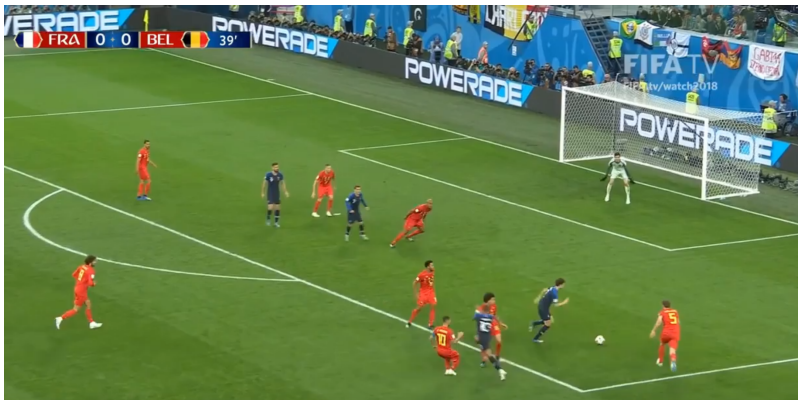


Figure 6.2: Raw image of the ball

Figure 6.3: detected image of the ball

6.1.3 Line Drawing :

Here, an image of France vs Belgium is used as an input to detect the white line and it is marked green by the line detector. Figure 5 shows the raw image that is given as input and figure 6 shows the line detected image by the detector.

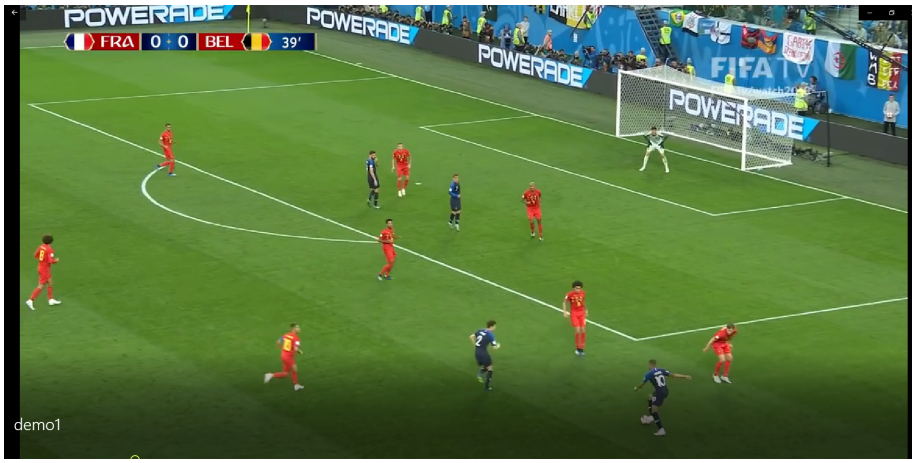


Figure 6.4: Raw input to detect line

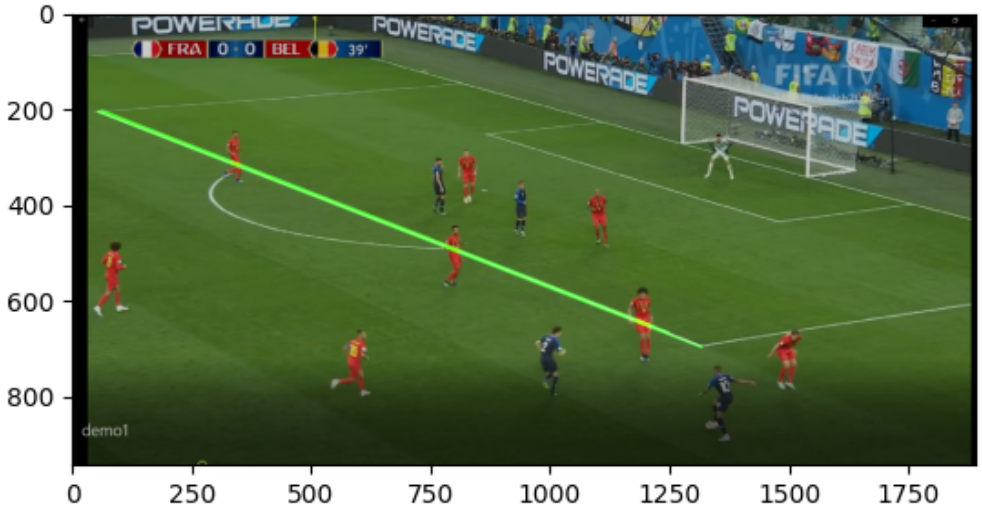


Figure 6.5: Detected image from the input

6.1.4 Object detection Path/Trajectory Tracking:

Here two images from a video input are taken and the tracker tracks the paths of the moving player football both, tracks the trajectory also, and the path of the moving objects is shown with dots along the line.

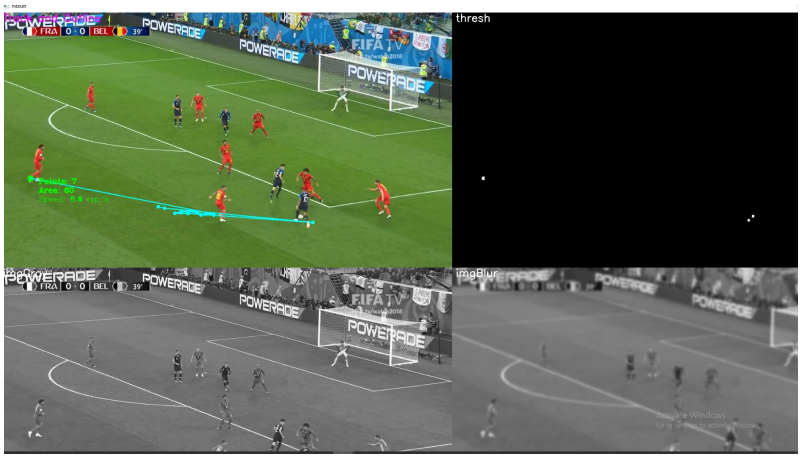


Figure 6.6: Object Detection1

Figure 6.7: Object Detection2

6.1.5 Pose Detection:

In the below pictures we have shown that the players position were detected using TensorFlow so that if in that match offside is happening or not for any players.

Figure 6.8: Pose Detection1

Figure 6.9: Pose Detection2

6.2 Result Analysis

From all of our detections, including players, ball, player poses, lines, and trajectory, whenever an offside situation occurs, the algorithm will start all of these algorithms and analyze the data to figure out every possible solution. If there is a probable offside situation, the referee can make the offside call; otherwise, no offside call will occur.

In our thesis work, These datasets were taken from random videos played on youtube on different matches. So different videos play at different rates and different resolutions. There is no specific dataset here. Also as it is not fully auto, mated so using some videoes we have calculated our accuracy rates.

Logistic Regression:

A probability of a value between 0 and 1 is generated via logistic regression. If the model calculates a value of 0.932 for a certain email message, it means that there is a 93.2% chance that the email message is spam.

```
from sklearn.datasets import make_classification

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score

from sklearn.linear_model import LogisticRegression

nb_samples = 1000

x, y = make_classification(n_samples=nb_samples, n_features=2,
n_informative=2, n_redundant=0, n_clusters_per_class=1)

xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.2,
random_state=42)

model = LogisticRegression()

model.fit(xtrain, ytrain)

print("Accuracy Score:",accuracy_score(ytest, model.predict(xtest)))
```

Mean Absolute Error:

MAE(Mean Absolutae Error was also used in our proposed system.

```
from sklearn.metrics import mean_absolute_error as mae

from sklearn.model_selection import cross_val_score

X, y = load_boston(return_X_y=True)

dtr = DecisionTreeRegressor()

cross_val_score(dtr, X, y)


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)

dtr = DecisionTreeRegressor()

dtr.fit(X_train, y_train)

print('MAE on train:', mae(y_train, dtr.predict(X_train)))

print('MAE on test:', mae(y_test, dtr.predict(X_test)))
```

```
Accuracy Score: 0.985
MAE on train: 0.0
MAE on test: 2.9157894736842107
```

```
Accuracy Score: 0.965
MAE on train: 0.0
MAE on test: 3.2756578947368418
```

```
MAE on train: 0.0
MAE on test: 2.838815789473684

Process finished with exit code 0
```

Confusion Matrix

After developing the model, its performance should be tested to establish its applicability. Confusion matrix helps to measure the model performance. The below table describes the the categorization problem's characteristics, see the table below.

	Predicted Value Positive (P)	Predicted Value Negative (N)
Actual Value Positive (P)	True Positive (TP)	False Negative (FN)
Actual Value Negative (N)	False Positive (FP)	True Negative (TN)

The Classification Rate or Accuracy can be calculated using the relationship below.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$F - measure = \frac{2 \times \text{ Recall } \times \text{ Precision}}{\text{Recall } + \text{ Precision}}$$

In our work we have used Confusion Matrix.

```
from sklearn import metrics
y_pred = ["a", "b", "c", "a", "b"]
y_act = ["a", "b", "c", "c", "a"]
print(metrics.confusion_matrix(y_act, y_pred, labels=["a", "b", "c"])
print(metrics.classification_report(y_act, y_pred, labels=["a",
"b","c"]))
```

Using the confusion_Metrics, We have created a table below:

Table 6.1: Ball Detection Accuracy Table

Table Head	**Table Column Head**			
	Precision	***recall***	***F1-Score***	***Support***
a	0.50	0.50	0.50	2
b	0.50	1.00	0.67	1
c	1.00	0.50	0.67	2
accuracy			0.60	5
acro avg	0.67	0.67	0.61	5
weighted avg	0.70	0.60	0.60	5

In the below table we got accuracy rate using Logistic regression.A sample video used on our working codes.

Table 6.2: Accuracy Table

Player Detection	***Ball detection***	***Path Detection***
0.835	0.915	0.935

6.2.1 Analysis

From Paper [10], the precision score of player detection is 0.87 ±0.05, the Recall score is 0.91 ±0.02 and in our case, the highest precision score is 1.00, and the Recall score is also 1.00.

From Paper [4], the highest accuracy score is 93.33% whereas our highest achieved score is 98.50%.

From paper [14], we have similar precision scores, however, we have higher recall score than them and their recall score is 0.9 and our one is 1.00.

From paper [3], the precision score and recall score of both the party is pretty much closer to one another, the opponent paper has a precision score of 0.99 and a recall score of 0.99 also whereas our one is 1.00 on both the precision and recall scores.

Chapter 7

Conclusion And Future Works

7.1 Conclusion

In this paper, we performed offside tracking using input from video streams. Even many researchers are trying to accomplish this work using different methods to make the process of offside calling more smooth and more accurate yet we are trying to make the process more accurate by using algorithms and computer vision. There has been a lack of accuracy in some processes, we are planning to minimize the issues. We are detecting offside from the live video where we detect player, ball, and player position using different algorithms. After analyzing all the data we send the results to the linesmen. It will assist the referees in many ways. By doing this research our main objective is to lessen human error and make football games smooth and more attractive by helping the linesmen. Thus this research represents an attempt to identify the ball position in most of the encountered situations and to improve the precision of ball tracking, player tracking, and offside calling as much as possible while also minimizing the time required.

7.2 Future scope

The popularity of football is huge. With the help of technology football game can be more attractive. By using more dedicated camera, if we can convert the whole stadium into a 3D model and track everything then it will help us to detect all players and the ball in more convenient way. Which will lead to a result with better accuracy. Moreover, if we can use dedicated tracking cameras instead of normal camera then we can use each players' exact position on the field which can give far better results. And for ball tracking, if we can implement a chip inside a ball which can give us exact point location on the field along with the results of ball being touch or kicked or not then that also can help us to provide better results. By combining the player and ball tracking data and using AI(artificial intelligence), the cutting-edge technology, this technique can be totally automatic and generate results that are almost accurate without the need of human interference.

Bibliography

[1] T. D'Orazio, M. Leo, P. Spagnolo, *et al.*, "An investigation into the feasibility of real-time soccer offside detection from a multiple camera system," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 12, pp. 1804–1818, 2009. DOI: 10.1109/TCSVT.2009.2026817. [Online]. Available: https://ieeexplore.ieee.org/document/5159423.

[2] H. Zawbaa El-Bendary and Kim, "Machine learning-based soccer video summarization system," vol. 263, 2011, ISBN : 978-3-642-27185-4. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-27186-1_3.

[3] M. Tavassolipour, M. Karimian, and S. Kasaei, "Event detection and summarization in soccer videos using bayesian network and copula," *IEEE Transactions on circuits and systems for video technology*, vol. 24, no. 2, pp. 291–304, 2013.

[4] S. Siratanita, K. Chamnongthai, and M. Muneyasu, "Saliency-based football offside detection," 2017, pp. 1–4. DOI: 10.1109/ISCIT.2017.8261213.

[5] K. Muthuraman, P. Joshi, and S. Raman, "Vision based dynamic offside line marker for soccer games," Apr. 2018. [Online]. Available: https://www.researchgate.net/publication/324600045_Vision_Based_Dynamic_Offside_Line_Marker_for_Soccer_Games.

[6] T. Park and J. Park, "Automatic offside judgment system based on position information in a soccer game," *International Journal of Advanced Culture Technology*, vol. 6, no. 3, pp. 216–225, 2018, PMID: 2288-7202. DOI: 10.17703//IJACT2018.6.3.216. eprint: https://doi.org/10.17703//IJACT2018.6.3.216. [Online]. Available: https://doi.org/10.17703//IJACT2018.6.3.216.

[7] P. N. Patil, R. J. Salve, K. R. Pawar, and M. M. Atre, "Offside detection inthe game offootball using contour mapping," *International Journal of Research in Engineering and Science*, vol. 6, no. 4, pp. 66–69, 2018, ISSN Online: 2320-9364.

[8] jun94, "Hough circle transform," *jun-devpBlog*, 2020. [Online]. Available: https://medium.com/jun94-devpblog/cv-6-structure-extraction-with-hough-transform-line-circle-aaf8be62f169.

[9] A. Lhoest, "Deep learning for ball tracking in football sequences," 2020. [Online]. Available: http://hdl.handle.net/2268.2/9074.

[10] N. Panse and A. Mahabaleshwarkar, "A dataset & methodology for computer vision based offside detection in soccer," in *Proceedings of the 3rd International Workshop on Multimedia Content Analysis in Sports*, 2020, pp. 19–26. DOI: 10.1145/3422844.3423055. [Online]. Available: https://doi.org/10.1145/3422844.3423055.

[11] O. Singh, "Select roi or multiple rois bounding box in opencv python," 2020. [Online]. Available: https://blog.electroica.com/select-roi-or-multiple-rois-bounding-box-in-opencv-python.

[12] S. Siratanita, K. Chamnongthai, and M. Muneyasu, "A method of football-offside detection using multiple cameras for an automatic linesman assistance system," *Wireless Personal Communications*, vol. 118, no. 3, pp. 1883–1905, 2021. [Online]. Available: https://link.springer.com/article/10.1007/s11277-019-06635-0.

[13] D. Johnson, "Semi-automated var offside is here," *ESPN Fc*, 2022. [Online]. Available: https://www.espn.in/football/blog-fifa/story/4355005/semi-automated-offside-why-arsene-wenger-thinks-it-can-fix-var.

[14] M. Y. K. Tani and L. F. K. Tani, "An offside soccer detection system using ontology and deep learning," *Turkish Journal of Computer And Mathematics Education (TURCOMAT)*, vol. 13, no. 03, pp. 377–387, 2022. [Online]. Available: https://www.turcomat.org/index.php/turkbilmat/article/view/13000.

[15] "Computer vision," [Online]. Available: https://medium.com/mlearning-ai/computer-vision-fundamentals-and-opencv-overview-9a30fe94f0ce.

[16] "Hough transform," [Online]. Available: https://theailearner.com/tag/hough-circle-transform-algorithm.

[17] "Kalman," [Online]. Available: https://www.mathworks.com/help/control/ref/ss.kalman.html?searchHighlight=kalman&s_tid=srchtitle_kalman_1/.

[18] "Kalman filtering," [Online]. Available: https://www.educba.com/opencv-kalman-filter/.

[19] "Laws11 of the football offside," [Online]. Available: https://www.theifab.com/laws/latest/offside/#offside-position.

[20] "Opencv," [Online]. Available: https://opencv.org.

[21] "Pose detection," [Online]. Available: https://google.github.io/mediapipe/solutions/%20pose.html?fbclid=IwAR0G7Rgd82UsG3roKAuqUO%5C%5C7COBYH3DBcukD4Wvwibg9WfFWT8JNfk0gkScc.