AIE206 PROJECT

DR. Amr ABDELBARI

Topic: **OBJECT DETECTION BY YOLO: CASE OF FOOTBALL ANALYSIS SYSTEM**

GROUP NAME : SURVIVORS

Members:

Full name:	Student number:
Christian Katshiza MALONGA	20226324
Rafay Ahmed QURESHI	20226992
Bi Doué Modeste TIAN	20215127

Table of Contents

Introduction

Purpose of the Program

The Football Analysis program is designed to help analyze football (soccer) matches by tracking two important aspects:

- How the camera moves during a match recording
- Which team each player belongs to based on uniform colors

This information is valuable for coaches, analysts, and students who want to understand player movements, team formations, and game strategies. By accounting for camera movement and automatically identifying player teams, the program provides more accurate and efficient analysis than manual methods.

System Overview

The Football Analysis program consists of two main components that work together:

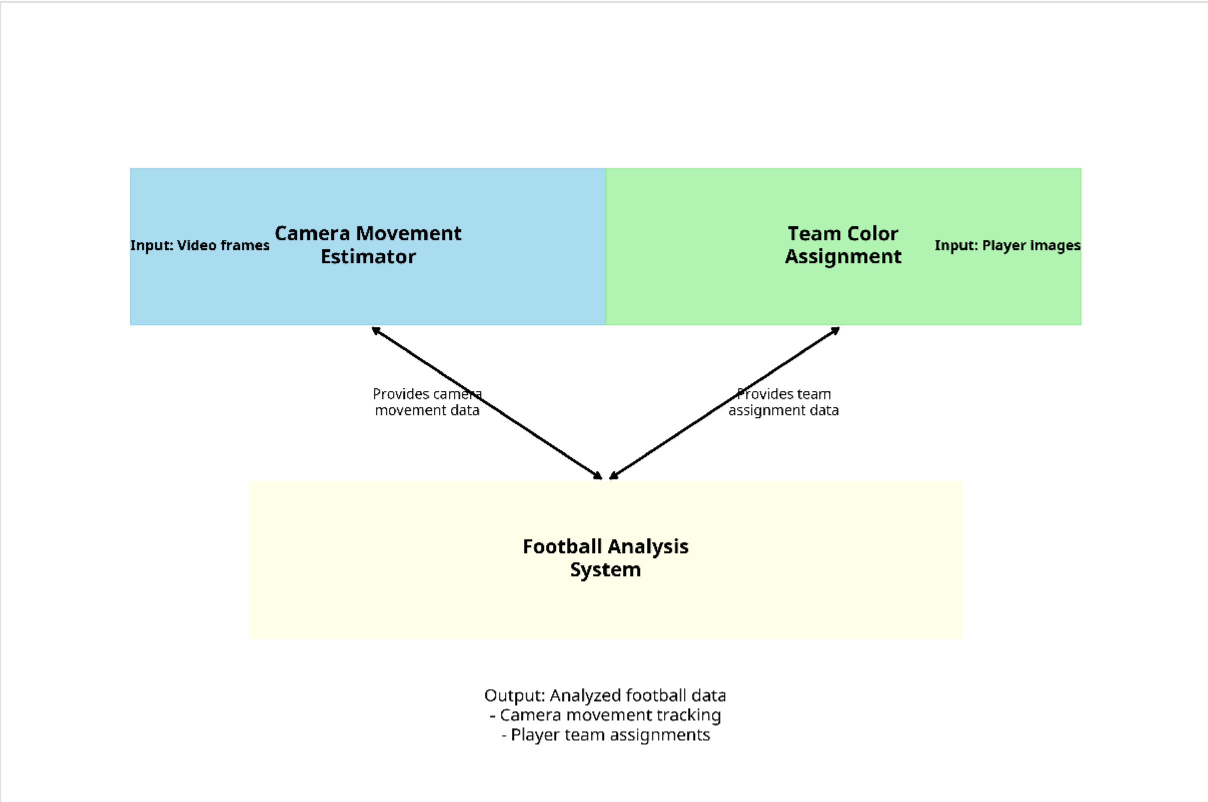

Figure 1: Overview of the Football Analysis System

As shown in Figure 1, the system takes two types of input:

- **Video frames** from football match recordings, which are processed by the Camera Movement Estimator
- **Player images**, which are analyzed by the Team Color Assignment component

The outputs from both components are combined to provide comprehensive football match analysis data.

> **Why this matters:** When analyzing football matches from video, two common challenges are accounting for camera movement and identifying which team each player belongs to. This program solves both problems automatically, saving time and improving accuracy.

1 - Camera Movement Estimator

1.1 Why Track Camera Movement?

When analyzing football matches from video recordings, the camera typically follows the action by panning, tilting, and zooming. This camera movement can make it difficult to accurately track player movements on the field.

For example, if a player appears to move to the right in the video, it could be because:

- The player is actually moving to the right on the field, or
- The camera is panning to the left, making the player appear to move right

The Camera Movement Estimator solves this problem by calculating how much the camera moves in each frame of the video. This information can then be used to adjust player position data, resulting in more accurate analysis of actual player movements.

1.2 How It Works

The Camera Movement Estimator works by tracking distinctive points (called "features") in the video frames and measuring how these points move from one frame to the next.

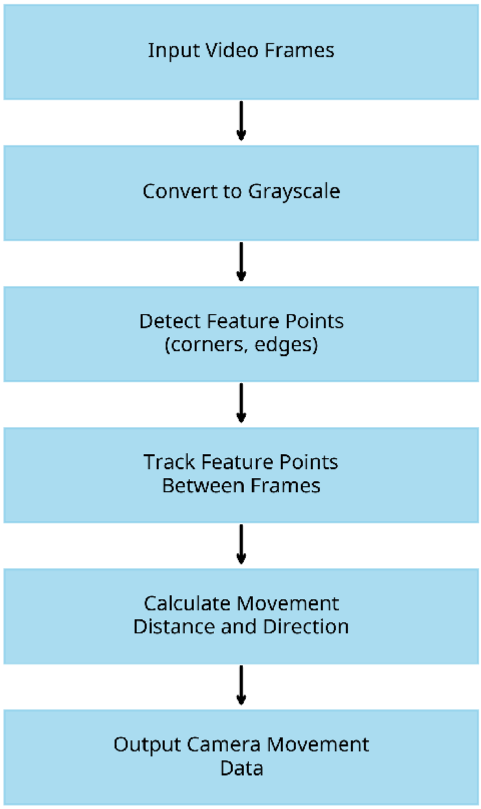

Figure 2: Camera Movement Estimation Process

As shown in Figure 2, the process involves several steps:

1. **Input Video Frames:** The system takes a sequence of video frames as input.
2. **Convert to Grayscale:** Each frame is converted to grayscale to simplify processing.
3. **Detect Feature Points:** The system identifies distinctive points like corners and edges in each frame.
4. **Track Feature Points:** It tracks how these points move from one frame to the next.
5. **Calculate Movement:** Based on the movement of these points, it calculates the overall camera movement.
6. **Output Data:** The system produces camera movement data that can be used to adjust player positions.

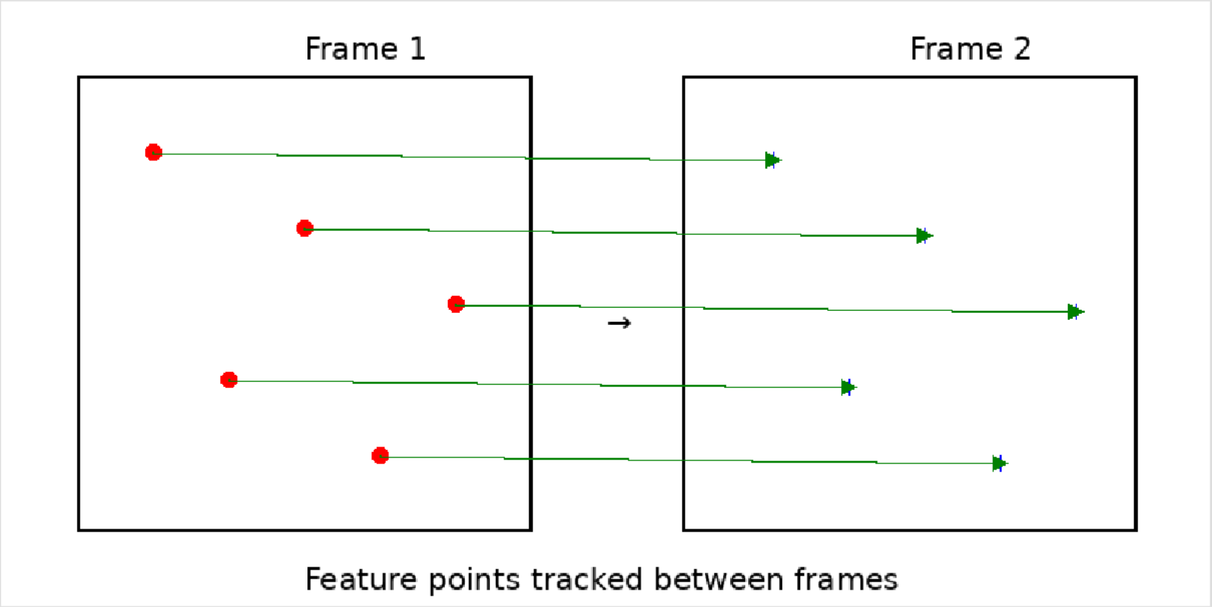

Figure 3: Feature Point Tracking Between Frames

Figure 3 illustrates how feature points (red dots in Frame 1) are tracked to their new positions (blue dots in Frame 2). The green arrows show the movement of each point. By analyzing these movements collectively, the system can determine how the camera has moved.

1.3 Technical Explanation

The Camera Movement Estimator uses a technique called "optical flow" to track feature points between frames. Here's a simplified explanation of how it works:

Feature Detection

The system uses the "goodFeaturesToTrack" function from the OpenCV library to identify corners and other distinctive points in the first frame. These points are chosen because they are easy to track reliably.

Feature Tracking

The system then uses the "calcOpticalFlowPyrLK" function (Lucas-Kanade method) to find these same points in the next frame. This function calculates the optical flow, which represents how pixels move from one frame to the next.

Movement Calculation

By comparing the positions of feature points in consecutive frames, the system calculates:

- How far the camera has moved (distance)
- In which direction the camera has moved (x and y components)

Position Adjustment

Once the camera movement is known, player positions can be adjusted to account for this movement. For example, if the camera moves 10 pixels to the right, all player positions need to be shifted 10 pixels to the left to represent their actual positions on the field.

> **Example:** Imagine a player standing still on the field. If the camera pans to the right, the player will appear to move to the left in the video. By calculating that the camera moved 15 pixels to the right, the system can adjust the player's apparent position by 15 pixels to the right, correctly showing that the player didn't actually move.

2 - Team Color Assignment

2.1 Identifying Players by Team

In football analysis, it's essential to know which team each player belongs to. While humans can easily distinguish teams by their uniform colors, automating this process allows for faster and more consistent analysis of large amounts of video data.

The Team Color Assignment component analyzes images of players and determines which team they belong to based on the dominant colors in their uniforms. This automation saves analysts from having to manually label each player in every frame.

2.2 Color Analysis Process

The Team Color Assignment works by analyzing the colors in player images and comparing them with known team colors.

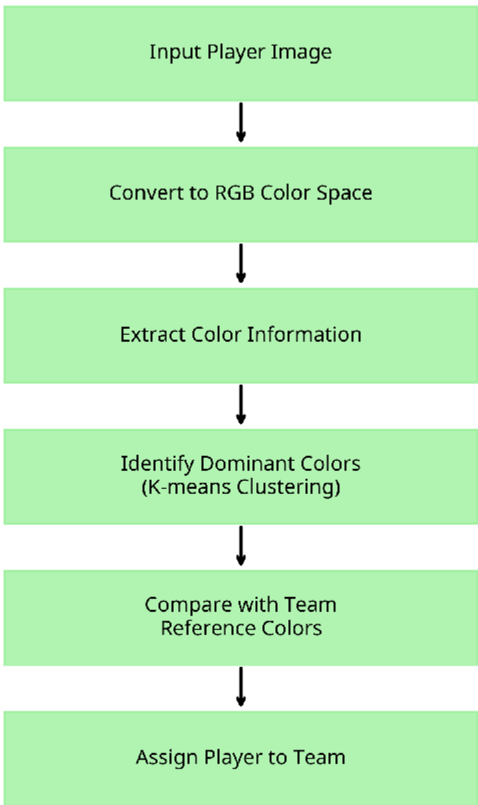

Figure 4: Team Color Assignment Process

As shown in Figure 4, the process involves several steps:

1. **Input Player Image:** The system takes an image of a player as input.
2. **Convert to RGB:** The image is converted to RGB color space for analysis.
3. **Extract Colors:** Color information is extracted from the image.
4. **Identify Dominant Colors:** The system identifies the most prominent colors using clustering.
5. **Compare with Team Colors:** These dominant colors are compared with reference team colors.
6. **Assign Team:** The player is assigned to the team with the closest matching colors.

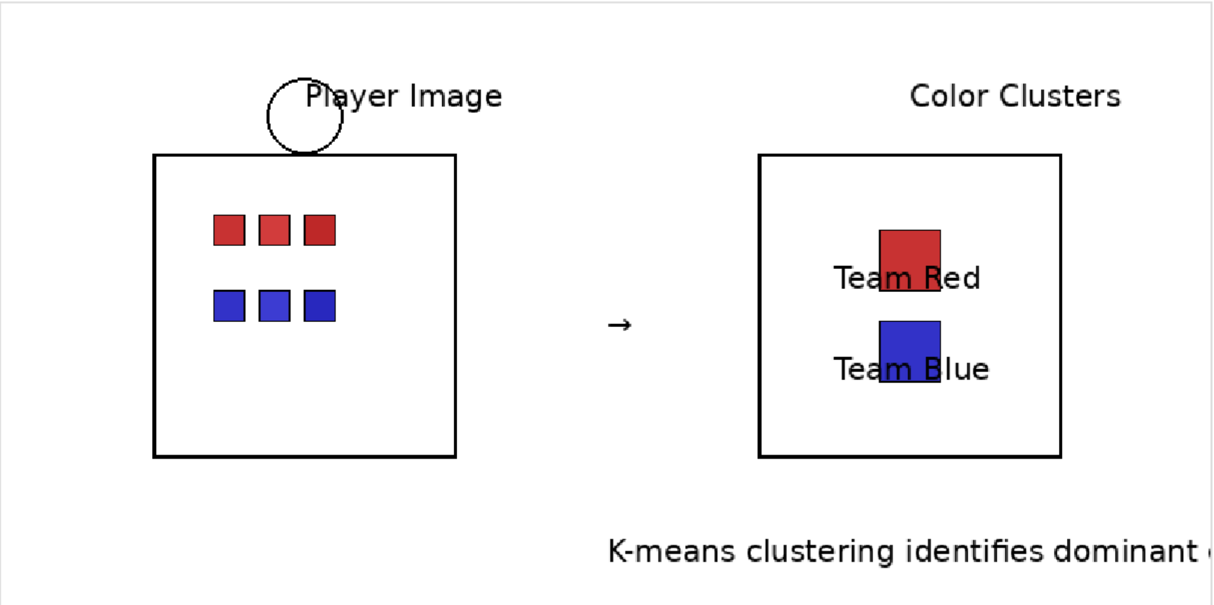

Figure 5: Color Clustering for Team Assignment

Figure 5 illustrates how colors from a player image are clustered to identify dominant colors, which are then matched with team reference colors to determine which team the player belongs to.

2.3 Technical Explanation

The Team Color Assignment uses a technique called "K-means clustering" to identify dominant colors in player images. Here's a simplified explanation of how it works:

Color Extraction

The system extracts color information from each pixel in the player image. Each pixel is represented as a point in 3D color space (Red, Green, Blue).

Color Clustering

The K-means clustering algorithm groups similar colors together. It works by:

1. Starting with a specified number of cluster centers (e.g., 3 centers for 3 main colors)
2. Assigning each pixel to the nearest cluster center
3. Recalculating each cluster center based on the average of all pixels assigned to it
4. Repeating steps 2-3 until the cluster centers stabilize

Dominant Color Identification

After clustering, the system identifies the most prominent color clusters based on how many pixels belong to each cluster. These represent the dominant colors in the player's uniform.

Team Matching

The dominant colors are compared with reference colors for each team. The player is assigned to the team whose reference colors most closely match the dominant colors in the image.

Example: If a player image has dominant colors of red and white, and the reference colors for Team A are red and white while Team B's colors are blue and white, the system would assign the player to Team A because there's a better match for both dominant colors.

3 - Using the Program

3.1 Setting Up

To use the Football Analysis program, you need to set up your environment with the necessary software and libraries.

Requirements

- **Python:** Version 3.6 or higher
- **Libraries:** OpenCV, NumPy, scikit-learn, matplotlib
- **Input Data:** Football match videos or player images

Installation

You can install the required libraries using pip, Python's package manager:

```
pip install opencv-python numpy scikit-learn matplotlib
```

3.2 Running the Camera Movement Estimator

To use the Camera Movement Estimator, you need to:

1. Import the CameraMovementEstimator class
2. Load your video frames
3. Create an estimator instance with the first frame
4. Call the get_camera_movement() method with all frames
5. Use the resulting data for analysis or visualization

Here's a simplified example of how to use the Camera Movement Estimator:

```
# Import the necessary modules
from camera_movement_estimator import
CameraMovementEstimator
import cv2
```

```
# Load a video file
video = cv2.VideoCapture("football_match.mp4")

# Read all frames from the video
frames = []
while True:
    success, frame = video.read()
    if not success:
        break
    frames.append(frame)

# Initialize the estimator with the first frame
estimator = CameraMovementEstimator(frames[0])

# Get camera movement data
camera_movement = estimator.get_camera_movement(frames)

# Visualize the results
output_frames = estimator.draw_camera_movement(frames,
camera_movement)

# Display or save the output frames
for frame in output_frames:
    cv2.imshow("Camera Movement", frame)
    cv2.waitKey(30)  # Display each frame for 30ms
```

3.3 Running the Team Color Assignment

The Team Color Assignment is implemented in a Jupyter notebook, which provides an interactive environment for analyzing player images.

To use the Team Color Assignment:

1. Open the color_assignment.ipynb notebook in Jupyter
2. Load a player image
3. Define reference colors for each team

4. Run the color clustering code
5. View the team assignment result

Here's a simplified example of the code in the notebook:

```
# Import necessary libraries
import cv2
import numpy as np
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

# Load a player image
image_path = "player.jpg"
image = cv2.imread(image_path)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Reshape the image for clustering
pixels = image.reshape(-1, 3)

# Apply K-means clustering to find dominant colors
kmeans = KMeans(n_clusters=3)
kmeans.fit(pixels)
dominant_colors = kmeans.cluster_centers_

# Define reference team colors
team_a_color = np.array([255, 0, 0])  # Red
team_b_color = np.array([0, 0, 255])  # Blue

# Calculate distance to each team color
distance_to_a = np.linalg.norm(dominant_colors -
team_a_color, axis=1).min()
distance_to_b = np.linalg.norm(dominant_colors -
team_b_color, axis=1).min()

# Assign player to team with closest color match
if distance_to_a < distance_to_b:
    team = "Team A"
```

```
else:
    team = "Team B"

print(f"Player assigned to: {team}")
```

4 - Real-World Applications

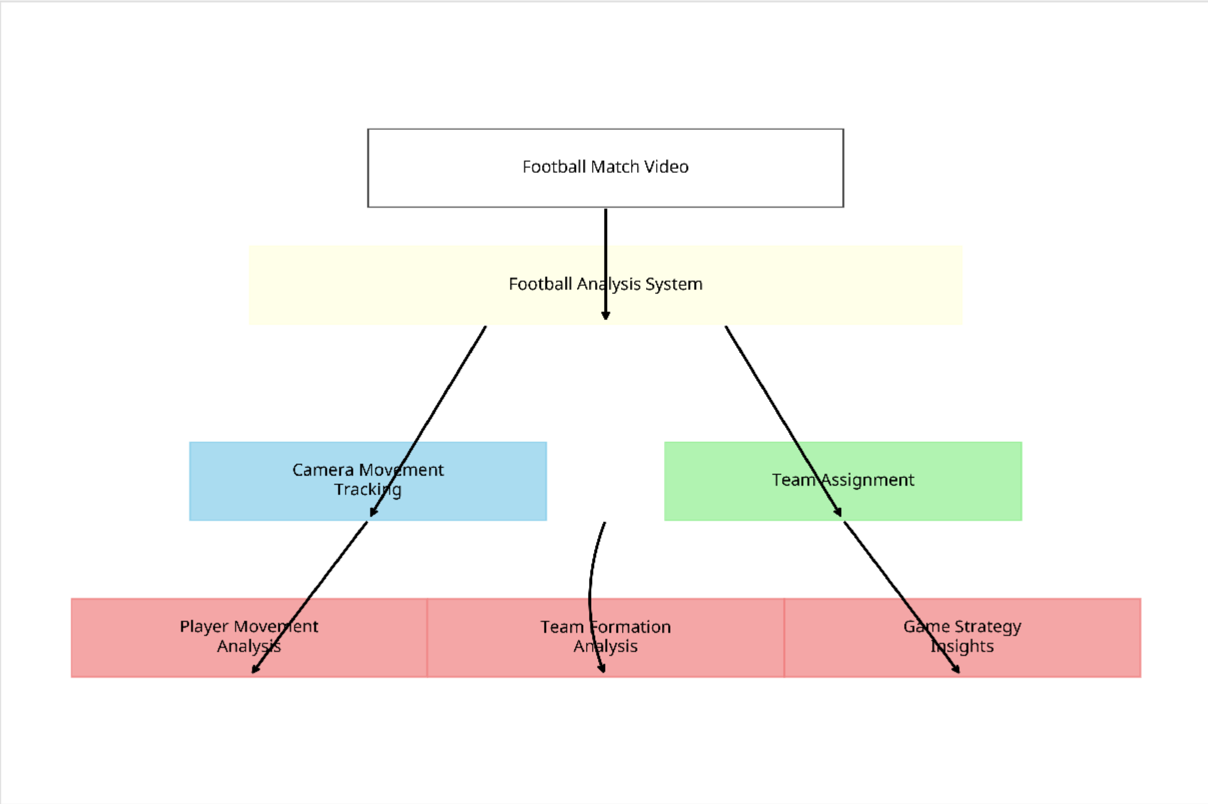

Figure 6: Real-World Applications of Football Analysis

4.1 Performance Analysis

The Football Analysis program can be used by coaches and analysts to evaluate player and team performance:

- **Player Tracking:** By accounting for camera movement, the program allows for accurate tracking of player positions and movements throughout a match.
- **Distance Covered:** Analysts can calculate how far each player runs during a match, providing insights into fitness levels and work rate.
- **Speed Analysis:** The program can be used to measure player sprint speeds and movement patterns.
- **Heat Maps:** By aggregating position data, analysts can create heat maps

showing where each player spends most of their time on the field.

Example: A coach notices that a midfielder's heat map shows they're spending too much time in their own half. Using the accurate position data provided by the program, the coach can adjust the team's tactics to ensure better positioning.

4.2 Tactical Analysis

The program provides valuable data for tactical analysis:

- **Formation Analysis:** By identifying which team each player belongs to and tracking their positions, the program can visualize team formations and how they change during a match.
- **Pressing Patterns:** Analysts can identify when and where teams apply pressure to opponents.
- **Defensive Organization:** The program can help evaluate how well teams maintain their defensive shape.
- **Attacking Movements:** Coaches can analyze patterns in attacking play to identify effective strategies.

Example: An analyst uses the program to track how a team's formation changes when transitioning from defense to attack. This reveals that certain players are out of position during these transitions, creating vulnerabilities that opponents can exploit.

4.3 Educational Use

The Football Analysis program is also valuable for educational purposes:

- **Teaching Tactics:** Coaches can use the program to demonstrate tactical concepts to players using real match footage.
- **Student Projects:** Sports science students can use the program for research projects on player movement and team tactics.
- **Learning Computer Vision:** Computer science students can study the program to learn how computer vision techniques are applied in sports analysis.

- **Developing Analytical Skills:** Students can develop data analysis skills by working with the outputs from the program.

> **Example:** A sports science professor assigns students a project to analyze how a professional team's formation changes in different game situations. Students use the Football Analysis program to track player positions and team structures, then present their findings on how the team adapts tactically.

Conclusion

The Football Analysis program demonstrates how computer vision and data analysis techniques can be applied to sports analysis, specifically football (soccer). By tracking camera movement and identifying player teams automatically, the program provides valuable tools for coaches, analysts, and students.

Key Benefits

- **Accuracy:** By accounting for camera movement, the program provides more accurate player position data than manual tracking.
- **Efficiency:** Automatic team assignment saves time compared to manually labeling each player.
- **Insights:** The combined data allows for deeper insights into player movements, team formations, and game strategies.
- **Accessibility:** The program makes advanced sports analysis techniques accessible to a wider audience, including students and amateur coaches.

Future Possibilities

The Football Analysis program could be extended in several ways:

- **Player Identification:** Adding face recognition to identify specific players
- **Ball Tracking:** Implementing algorithms to track the ball's position
- **Event Detection:** Automatically detecting key events like goals, passes, and tackles
- **Real-time Analysis:** Optimizing the code to work in real-time during live matches

By understanding how the Football Analysis program works, users can not only apply it to their own analysis needs but also appreciate the underlying computer vision concepts and potentially contribute to future developments in sports analysis technology.