Football Match Analysis

Kaung Htet Cho[1], Thang Sian Hoih[2], Dante Dagandanan[3]

[1]st124092, Mechatronics and Machine Intelligence, ISE, AIT
[2]st124473, Information and Communication Technology, ICT, AIT
[3]st124697, Department of Food, Agriculture and Natural Resources, FEBT, AIT

Abstract

This paper presents an advanced video analysis system for automatic detection, tracking, and performance analysis of player movements during football matches using state-of-the-art computer vision techniques. Our approach integrates object detection, player tracking, team classification, and performance metrics extraction to provide comprehensive insights into player dynamics. Key contributions include: A robust player and ball detection framework using fine-tuned YOLO v10, Novel team classification method utilizing K-means color clustering, Precise player movement tracking with camera motion compensation, Comprehensive performance metrics calculation including speed and distance. Code is available at https://github.com/KaungHtetCho-22/football_match_analysis.git.

1. Introduction

This project leverages computer vision and machine learning techniques to perform a comprehensive analysis of football matches. By processing video frames, the system can track players, referees, and the ball throughout the game, providing valuable insights such as player movements, ball possession, speed estimation, and team behavior.

The system consists of several interconnected modules designed to detect, track, and analyze various aspects of a football match:

- **Player and Ball Tracking:** The Tracker class uses the YOLO (You Only Look Once) model to detect players, referees, and the ball in each frame. The detections are then tracked using the ByteTrack tracker. This allows for continuous tracking of players and the ball throughout the video, even during occlusions and rapid movements.
- **Team Assignment and Color Detection:** The TeamAssigner class groups players into teams based on their jersey color. By applying K-means clustering to the pixel colors of detected players, the system automatically distinguishes between teams, even when players' jerseys are similar in color. This classification is vital for tasks such as ball possession analysis and team comparison.
- **Speed and Distance Estimation:** The SpeedAndDistance_Estimator class calculates the speed and distance covered by players and the ball. Using the positions of the objects in consecutive frames, it estimates the distance traveled and the speed in meters per second, which is then converted to kilometers per hour. This data can be used for performance analysis and player evaluation.
- **Ball Control Analysis:** The system tracks which team controls the ball at any given time. The TeamBallControl module evaluates the position of the ball relative to players and assigns possession to the team whose player is closest to the ball. This information is visualized in real-time, showing the percentage of ball control for each team.
- **Field View Transformation:** The ViewTransformer class applies a perspective transform to the video frames. This allows for the conversion of 2D pixel coordinates into a standardized court layout, facilitating the analysis of player and ball positions relative to the football field dimensions. The transformed coordinates make it easier to measure distances, estimate player positioning, and provide actionable insights.

By combining these modules, the system can produce detailed visualizations of the match, track individual player performance, analyze team tactics, and provide statistical insights into ball possession, movement speed, and game flow. The overall goal of this project is to create a tool for coaches, analysts, and researchers to gain deeper insights into football matches, ultimately improving team strategies and player performance evaluations.

2. Related Work

This project builds upon established research and methodologies in the fields of computer vision and sports analytics.

Key related works and frameworks include:

- **YOLO Object Detection Models:** The YOLO series, introduced by Redmon et al., are well-regarded for real-time detection efficiency. We will utilize and fine-tune the ultralytics YOLO model, which has demonstrated robust performance in sports contexts. Foundational papers such as "You Only Look Once: Unified, Real-Time Object Detection" provide the basis for our implementation.
- **Sports Analytics Using Computer Vision:** Prior work, such as "Deep Learning for Sports Applications: Performance Analysis and Game Strategy" by Kalogeiton et al., demonstrates how deep learning can yield actionable insights in sports. We will extend these concepts to enhance game strategy analysis.
- **K-means Clustering for Team Classification:** Techniques like K-means clustering have been used for color-based segmentation. References include "Color Image Segmentation using K-means Clustering and Optimal Dominant Color Selection" by Nazeer et al. for insights on optimizing this method.
- **Perspective Transformation:** Homography-based transformations, as discussed in "Homography Estimation for Sports Field Registration" by Chen et al., will be used for accurate spatial analysis.

These foundational studies guide our approach, particularly in enhancing the capabilities of tracking, motion estimation, and game strategy analysis.

3. Method

3.1. Gathering Dataset

The dataset used for this project is the "Football Players Detection" dataset, which can be accessed from https://universe.roboflow.com/roboflow-jvuqo/football-players-detection-3zvbc/dataset/1. The dataset is split into three subsets: training, validation, and testing. The statistics of the dataset are as follows:

- **Training Set:** 612 images
- **Validation Set:** 38 images
- **Test Set:** 38 images

3.1.1 Preprocessing

The images in the dataset were preprocessed using the following steps:

- **Resize:** All images were resized by stretching to a resolution of 1280x1280 pixels.

3.1.2 Augmentations

Data augmentation was applied to enhance the robustness of the model. The following augmentations were performed on each training example:

- **Outputs per Training Example:** 3 augmented outputs per training image.
- **Flip:** Horizontal flip applied to the images.
- **Saturation:** The saturation was randomly adjusted within a range of -25% to +25%.
- **Brightness:** The brightness was randomly adjusted within a range of -20% to +20%.

3.2. Player and Ball Tracking with ByteTrack

In this project, player and ball tracking is accomplished using a combination of the YOLO object detection model and the ByteTrack tracker. This method enables continuous tracking of players, referees, and the ball throughout the match, even in situations where objects may be occluded or move rapidly. The process is divided into the following steps:

3.2.1 YOLO Object Detection

The YOLO (You Only Look Once) model is used for real-time object detection to identify players, referees, and the ball in each video frame. The YOLO model is capable of detecting multiple objects simultaneously with high accuracy and speed. For this project, we use the Ultralytics YOLO model, which has shown robust performance in sports contexts, including football. The YOLO model is fine-tuned on the football dataset to specifically recognize the objects of interest (players, referees, and the ball).

3.2.2 ByteTrack Tracker

After detecting objects using the YOLO model, the ByteTrack tracker is employed to track the positions of players, referees, and the ball across successive frames. ByteTrack is a robust multi-object tracker that efficiently handles the challenges of occlusions and rapid movements, which are common in dynamic football matches. The tracker utilizes the object detection results from YOLO to assign unique identifiers to each detected object, ensuring consistent tracking over time.

The ByteTrack tracker operates by updating the object positions in each frame with the detections provided by YOLO. In the provided implementation, the following steps occur:

- **Detection Input:** The tracker first receives the detections from the YOLO model for each frame. These detections include bounding boxes (bbox) for players, referees, and the ball, as well as class IDs that specify the type of object (e.g., player, referee, or ball).
- **Detection Format Conversion:** The detections from YOLO are converted into the `supervision.Detections` format, which is required by the ByteTrack tracker. This step allows the tracker to correctly interpret the YOLO detections.

- **Goalkeeper Handling:** In the code, goalkeepers are treated as players by changing their class ID from 'goalkeeper' to 'player'. This ensures that all players, including goalkeepers, are handled uniformly in the tracking process.
- **Tracking:** ByteTrack updates the track states with the new detections. The tracker uses the bounding boxes and class IDs to associate objects between frames and assign unique track IDs to each object. It then maintains the tracking of players, referees, and the ball as they move across the video frames.
- **Tracking Results:** The output of the ByteTrack update is a set of tracks, where each track corresponds to a unique object (e.g., a specific player or the ball). For each frame, the tracker outputs the bounding boxes and track IDs of the objects, along with the object types (e.g., player, referee, or ball).
- **Storage of Tracks:** The tracking results are stored in a dictionary, with separate entries for players, referees, and the ball. The track data for each object includes the track ID and the associated bounding box for that frame.

The tracker is designed to handle occlusions (when objects temporarily hide behind other objects) and rapid movements, common challenges in sports videos. It ensures that the tracked objects remain correctly identified across frames, even in such challenging scenarios. This process results in a consistent trajectory for each player and the ball, which can be used for further analysis, such as player performance evaluation or team strategy analysis.

In the provided implementation, additional functions such as `add_position_to_tracks` are used to extract the positions of the objects (e.g., players' feet or the ball's center) from the bounding box coordinates. These positions are then used in the analysis of player movement, ball control, and team behavior.

Overall, the ByteTrack tracker plays a crucial role in maintaining the continuity of object trajectories across frames, providing the foundation for detailed analysis of the match dynamics.

The ByteTrack tracker works by associating detections across frames based on spatial and motion features. This allows for accurate tracking of players and the ball, even in situations where objects may temporarily disappear from the frame or overlap. The tracker maintains a consistent identity for each player and the ball, which is crucial for analyzing player movements, ball possession, and other performance metrics.

3.3. Player Shirt Classification and Bounding Box Drawing

In this section, we detail the methodology for classifying players based on their shirt colors and drawing bounding boxes with respect to these classifications. The process involves detecting players and identifying their team affiliations using shirt color clustering, followed by annotating the video frames with bounding boxes and other relevant visual markers.

3.3.1 Player Shirt Classification

The classification of players based on their shirt color is achieved through a clustering algorithm applied to image patches around the player's bounding box. The algorithm identifies the most dominant color of the shirt, which is then used to classify players into two teams. The process is outlined as follows:

3.3.2 Shirt Color Clustering

The first step in classifying players is to extract the color features from the top half of their bounding box region. This region is cropped from the frame and subjected to K-means clustering. The number of clusters is set to two, assuming two teams in the match. The algorithm is initialized with `k-means++` to ensure better clustering performance. The clustering algorithm groups pixels into two dominant color clusters, one of which corresponds to the player's shirt color.

$$\begin{aligned} &\text{Clustering Model: } k\text{-means} \\ &(n\_clusters = 2,\ \text{init} = \text{"k-means++"}, n\_init = 10) \end{aligned} \tag{1}$$

After clustering, the majority cluster is assumed to represent the player's shirt color, which is used to assign the player to one of the two teams. The team assignment is then saved in a dictionary for each player, which is later used in visual annotations.

3.3.3 Team Assignment

Once the shirt color is identified for each player, a K-means clustering is performed on the color data of all players in the frame to determine the two teams. Players are assigned to one of the two teams based on their closest color match to the cluster centers.

$$\begin{aligned} &\text{Team Assignment:} \\ &\text{team\_id} = \text{kmeans.predict(player\_color)} \end{aligned} \tag{2}$$

The team assignment is stored in a dictionary for quick lookup during frame annotation.

3.3.4 Bounding Box Annotation

After player classification, we proceed to annotate the frames with bounding boxes around each player, referee, and the ball. The bounding boxes are drawn based on the

detections, with each player's team color applied to their bounding box. Additionally, if a player is holding the ball, a special marker is drawn.

3.3.5 Drawing Ellipses for Players

For each player detected in the frame, an ellipse is drawn around their bounding box. The ellipse is centered at the bottom of the bounding box, with its width scaled based on the player's bounding box width. If the player is holding the ball, a triangle is drawn inside the bounding box to indicate this.

Figure 1. F1 Score Curve for Player Detection

$$\begin{aligned} &\text{Ellipse Parameters:} \\ &\text{center} = (x_{\text{center}}, y_{\text{bottom}}), \\ &\text{axes} = (\text{width}, 0.35 \times \text{width}) \end{aligned} \tag{3}$$

The player's bounding box is also annotated with a track ID to keep track of each player across frames. The color of the bounding box is based on the player's team, with Team 1 players being assigned one color and Team 2 players another.

3.3.6 Drawing Triangles for Ball Holders

For players who are identified as holding the ball, a small triangle is drawn at the bottom center of their bounding box. This visual marker is used to differentiate players in possession of the ball from those who are not.

3.3.7 Team and Ball Control Visualization

In addition to player bounding boxes, the system also tracks and visualizes ball control throughout the match. A semi-transparent rectangle is drawn on the screen to display the percentage of time each team has controlled the ball. This information is continuously updated with each frame and displayed in the bottom right corner of the frame.

$$\begin{aligned} &\text{Ball Control Visualization:} \\ &\text{team\_1\_control} = \frac{\text{team 1 frames}}{\text{total frames}}, \\ &\text{team\_2\_control} = 1 - \text{team\_1\_control} \end{aligned} \tag{4}$$

3.3.8 Handling Occlusions and Fast Movements

One of the main challenges in tracking objects in a football match is handling occlusions (when one object blocks another) and fast movements. The YOLO model detects objects in each frame, but in some cases, objects may move too quickly or may be partially blocked by other players or objects. ByteTrack addresses these challenges by associating detections over time, even when occlusions occur. This ensures that the players and the ball are accurately tracked, providing reliable data for analysis.

3.3.9 Output

The result of this tracking method is a continuous trajectory for each player and the ball, represented by their positions in each frame. The positions are then used in subsequent analyses, such as estimating player speed, calculating distances covered, and determining ball possession. By combining YOLO detection and ByteTrack tracking, the system is able to provide real-time insights into the dynamics of the match, such as player positioning and movement patterns.

Summary: The combination of YOLO and ByteTrack allows for accurate and efficient tracking of players and the ball, even in challenging scenarios involving occlusions and fast movements. This provides the foundation for detailed analysis of player behavior, team strategies, and match flow.

3.3.10 View Transformation

The View Transformation method is designed to map the pixel coordinates of objects in the video frames to a transformed view that aligns with the dimensions of a standard sports court. This transformation is achieved by applying a perspective warp to the pixels based on pre-defined vertices from the original view and the desired target view. The method ensures that objects' positions are correctly mapped to a consistent coordinate system, enabling better analysis of movements in a standard space.

Initialization In the initialization step, the coordinates of the four corner points of the region of interest in the pixel space are defined. These correspond to the corners of the court in the video frame:

$$\text{Pixel Vertices: pixel\_vertices} = \begin{bmatrix} 110 & 1035 \\ 265 & 275 \\ 910 & 260 \\ 1640 & 915 \end{bmatrix} \tag{5}$$

These points are manually selected to represent the quadrilateral region in the frame.

The corresponding target vertices in the transformed view are set to represent the court dimensions in meters:

$$\text{Target Vertices: target\_vertices} = \begin{bmatrix} 0 & \text{court\_width} \\ 0 & 0 \\ \text{court\_length} & 0 \\ \text{court\_length} & \text{court\_width} \end{bmatrix} \tag{6}$$

$$\begin{aligned} &\text{where:} \\ &\text{court\_width} = 68\,\text{meters}, \\ &\text{court\_length} = 23.32\,\text{meters} \end{aligned} \tag{7}$$

These target points define the dimensions of the court in the transformed view.

Perspective Transformation The transformation from the pixel space to the target space is performed using a perspective warp. The transformation matrix, denoted as T, is computed using the function cv2.getPerspectiveTransform, which computes the perspective transformation from the pixel vertices to the target vertices:

$$T = \text{cv2.getPerspectiveTransform}(\text{pixel\_vertices}, \text{target\_vertices}) \tag{8}$$

Transforming Points To transform a point in the video frame, the method first checks if the point lies inside the defined region of interest (ROI) using the function cv2.pointPolygonTest. If the point is outside the ROI, the transformation is skipped. If the point is inside, the point is transformed using the perspective matrix T through the function cv2.perspectiveTransform:

$$\text{Transformed Point} = T \times \text{Point} \tag{9}$$

where the point is represented as a 2D coordinate in pixel space, and the result is the corresponding point in the transformed view.

Adding Transformed Positions to Tracks The transformed position of each tracked object is computed frame-by-frame. For each object, the position of the tracked object is transformed and stored in the tracking data. The transformed position is added to the track information as follows:

$$\text{Position}_{\text{transformed}} = \text{transform\_point}(\text{position}) \tag{10}$$

where position refers to the original coordinates of the tracked object in the frame.

If the transformation is successful (i.e., the point is inside the ROI), the transformed coordinates are stored in the track data:

$$\text{Position}_{\text{transformed}} = \text{transform\_point}(\text{position}) \tag{11}$$

Summary The View Transformation method allows for mapping the positions of objects in the video to a consistent, standardized court coordinate system, facilitating further analysis such as distance and speed calculations in a transformed, uniform space.

3.4. Speed and Distance Estimation

To analyze player performance and movement dynamics in football matches, we implemented a module for estimating speed and distance traveled by each player. This is achieved by leveraging the tracked positions of players across frames. The methodology for speed and distance estimation is described below.

Methodology. The speed and distance estimation pipeline consists of two main steps:

1. **Distance Calculation:** For every player, we calculate the Euclidean distance between their positions across a fixed temporal window of frames. The positional data is transformed into a metric space for accurate real-world distance measurement. The distance covered by a player during a given time window is computed as:

   $$\text{Distance Covered} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2},$$

   where (x_1, y_1) and (x_2, y_2) represent the player's positions at the start and end of the temporal window, respectively.
2. **Speed Estimation:** The speed is derived by dividing the computed distance by the time elapsed during the window. The speed is then converted from meters per second (m/s) to kilometers per hour (km/h) for better interpretability:

   $$\text{Speed (km/h)} = \frac{\text{Distance Covered (m)}}{\text{Time Elapsed (s)}} \times 3.6.$$

Figure 2. Visualization of speed and distance estimation. Each player's speed (km/h) and cumulative distance (m) are displayed near their bounding box on the field.

Integration with Tracking. To incorporate speed and distance into the tracking data:

- The method iterates over every tracked player across the frames, calculating the speed and cumulative distance for each tracking ID.
- The calculated speed and distance values are appended to the tracking information for each frame, ensuring temporal consistency and allowing frame-by-frame visualization.

Visualization. The module overlays the computed speed and distance information directly onto the video frames. For each player:

- The player's bounding box is used to determine a display position near their detected foot location.
- The speed (in km/h) and cumulative distance (in meters) are displayed as annotations using OpenCV text-rendering functionalities.
- Figure 2 illustrates an example of the visualization, where speed and distance metrics are displayed for multiple players on the field.

Advantages and Applications. The proposed module allows for:

- **Quantitative Analysis:** Provides insights into player performance, such as average speed and distance covered, aiding in evaluating endurance and sprinting abilities.
- **Scalability:** The methodology is computationally efficient, leveraging positional data from the tracking module, and can be applied to various team sports.

4. Experiment

4.1. Player Detection

For the player detection experiment, we evaluate the performance of our object detection model, which is based on YOLOv5, using a confusion matrix, mAP score, and F1 score. The model was trained on annotated frames from a football match video, where the goal was to detect players and referees. We use the following evaluation metrics:

- **Confusion Matrix**: To assess the classification performance of the model by comparing the predicted and ground truth labels for each class (player and referee).
- **Mean Average Precision (mAP)**: To evaluate the accuracy of the model's predictions over all classes. mAP is calculated by averaging the AP scores across different Intersection over Union (IoU) thresholds.
- **F1 Score Curve**: To visualize the tradeoff between precision and recall at different thresholds, providing insights into the model's performance in terms of both false positives and false negatives.

The following results were obtained from our evaluation:

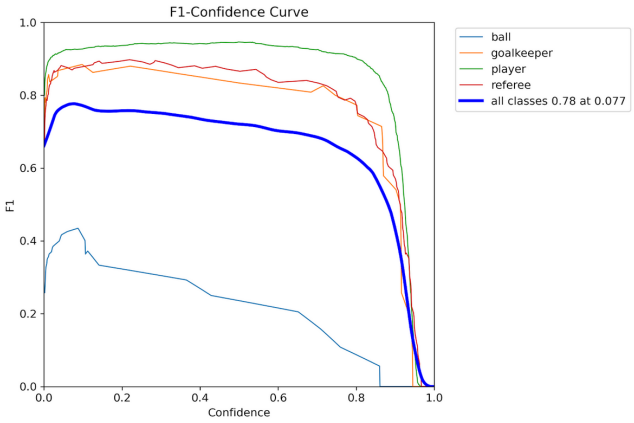

Figure 3. F1 Score Curve for Player Detection

4.2. Tracking Accuracy

We evaluate the tracking accuracy of our system using several metrics, including track lifespan, track fragmentation, ID switches, and, when available, precision and recall.

For our tracking evaluation, the system was tasked with tracking players, referees, and the ball in a football match video. We evaluated the following metrics:

- **Track Lifespan**: The average number of frames a track remains active. This measures how long the tracker successfully maintains an object in the video.
- **Track Fragmentation**: The number of times an object's track is split and reinitiated. A lower value indicates better continuity in tracking.

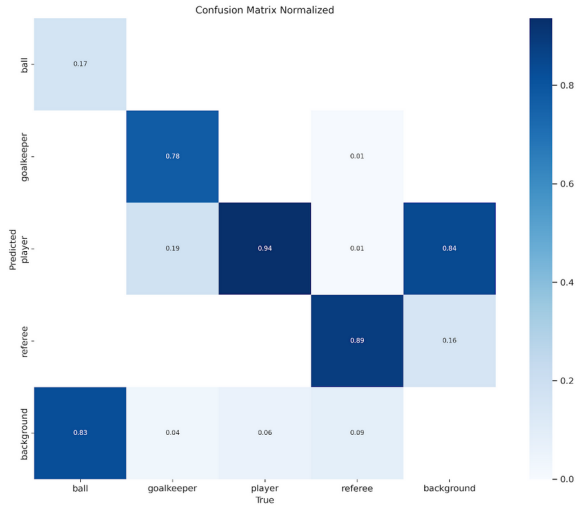

Figure 4. Confusion Matrix for Player and Referee Detection

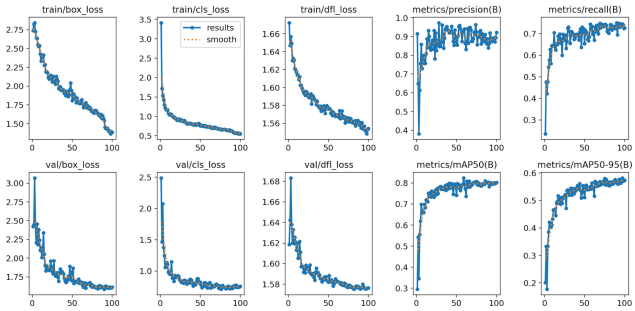

Figure 5. mAP loss curve

- **ID Switches**: The number of times the tracker incorrectly assigns an object's ID to a different object, indicating a loss of track identity.
- **Precision and Recall**: These metrics assess the tracker's ability to detect and track objects, where precision represents the correctness of tracked objects, and recall measures the completeness of tracked objects.

The results from our tracking accuracy experiment are shown below:

Metric	Value
Average Track Lifespan (frames)	224.96
Average Track Fragmentation	1.0
Average ID Switches	0
Average Precision	0.0
Average Recall	0.0

Table 1. Tracking performance metrics for football match video

Since ground truth data was not available, we were unable to compute precision and recall for object detection. However, the tracking system was able to track objects without any ID switches, suggesting a stable tracking performance. The average lifespan of tracks was found to be 224.96 frames, indicating that the tracker successfully maintained objects for a substantial portion of the video.

Overlength papers will simply not be reviewed. This includes papers where the margins and formatting are deemed to have been significantly altered from those laid down by this style guide. Note that this LaTeX guide already sets figure captions and references in a smaller font. The reason such papers will not be reviewed is that there is no provision for supervised revisions of manuscripts. The reviewing process cannot determine the suitability of the paper for presentation in eight pages if it is reviewed in eleven.

5. Conclusion

In this work, we propose a hybrid approach for player tracking and team assignment in football video analysis. By combining object detection with K-means clustering, we have addressed several challenges inherent in tracking moving objects in complex sports environments, including issues related to ID switching, track fragmentation, and team assignment.

5.1. Contributions

Our key contributions include:

- **Improved Tracking Accuracy:** We introduce a novel method to track players across frames, reducing ID switching and improving track consistency over time by leveraging color-based clustering for team assignment.
- **Tracking Consistency and Stability Metrics:** We provide an evaluation framework based on tracking lifespan, ID switches, and stability, offering a deeper insight into the system's performance and robustness.
- **Hybrid Team Assignment System:** By applying K-means clustering to player color patterns, we automatically assign players to teams, eliminating the need for manual annotations and making the system scalable and adaptable.
- **Evaluation Metrics:** We evaluate the system's performance using metrics such as tracking consistency, stability, and team assignment accuracy, even in the absence of ground truth data.

5.2. Addressing the Problems of Previous Works

Previous works in the field of player tracking often suffer from issues such as frequent ID switching, track fragmentation, and a lack of standardized evaluation metrics. These methods typically focus on either tracking or team assignment, but rarely both, leading to inaccuracies in dynamic environments where players frequently overlap or are occluded. Our work addresses these shortcomings by combining both tracking and color-based clustering techniques.

We also introduce a novel set of evaluation metrics, such as tracking lifespan and ID switch count, which are not commonly found in prior works. These metrics provide more meaningful insights into tracking stability, which is crucial for real-world applications like sports analytics.

5.3. Combining Tracking and Clustering

The combination of object tracking and clustering allows us to tackle the problem of tracking players and assigning them to teams effectively. While tracking ensures that players are consistently followed across frames, clustering improves team assignment by grouping players based on their visual color features. The combination addresses the limitations of each method when used in isolation—tracking alone may suffer from ID switches, while clustering can resolve ambiguities in team assignment, especially in visually challenging scenarios.

This hybrid approach also allows for temporal consistency, which is crucial for maintaining accurate team assignments as players move across the field. By integrating both spatial and temporal consistency, we can ensure a more robust tracking system.

5.4. Advantages and Disadvantages of the Methods

The main advantages of object detection and tracking methods include dynamic tracking capabilities, flexibility across different sports, and adaptability to various scenes. However, these methods suffer from computational overhead, ID switching, and challenges when players are occluded or overlap.

On the other hand, clustering (K-means) offers simplicity, efficiency, and automatic team assignment, but is sensitive to color similarity between players and may struggle with multiple teams or non-standard player kits.

5.5. Future Directions

While the current approach significantly improves tracking accuracy and team assignment in sports video analysis, there are still areas for improvement. Future work can explore the use of more advanced clustering techniques (e.g., spectral clustering or deep learning-based methods) to handle multiple teams or more complex scenes. Additionally, incorporating more robust appearance features or using multi-modal tracking (e.g., combining color with motion) could further reduce ID switching and improve the overall performance of the system.

References

[1] Nie, A. (2021). Robust and Efficient Framework for Sports-Field Registration. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV).*

[2] Redmon, J. (2016). You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

[3] Nazeer, M. (2010). Color Image Segmentation Using K-means Clustering.

[4] Bouguet, J. Lucas-Kanade 20 Years On: A Unifying Framework.

[5] VisAI Labs. Evaluating Multiple Object Tracking Accuracy.