Escola Tècnica Superior d'Enginyeria ETSE-UV

Grado en Ciencia de Datos


VNIVERSITAT DE VALÈNCIA

# Trabajo Fin de Grado

# Detección en un campo de fútbol mediante redes neuronales.

Autor: David Díaz Gregori

Tutor/es: Valero Laparra y Jorge Vila Tomas

VNIVERSITAT DE VALÈNCIA

Escola Tècnica Superior d'Enginyeria ETSE-UV

# Trabajo Fin de Grado

# Detección en un campo de fútbol mediante redes neuronales.

Autor: David Díaz Gregori

Tutor/es: Valero Laparra y Jorge Vila Tomas

**Declaración de autoría:**

Declaración de autoría: Yo,David Díaz Gregori, declaró la autoría del Trabajo Fin de Grado titulado “Detección en un campo de fútbol mediante redes neuronales” y que el citado trabajo no infringió las leyes en vigor sobre propiedad intelectual. El material no original que figura en este trabajo ha sido atribuido a sus legítimos autores.

Valencia, 17 de julio de 2025

Fdo: David Díaz Gregori

**Resumen**:

En este trabajo se ha desarrollado un sistema de detección y análisis táctico visual aplicado a partidos de fútbol, centrado en identificar los principales elementos del terreno de juego: balón, árbitros, jugadores de campo y porteros. Para ello, se ha utilizado un modelo preentrenado en formato YOLOv8, obtenido a través de Roboflow, capaz de realizar detecciones diferenciadas para cada clase de objeto.

El sistema se ha aplicado sobre un conjunto de catorce vídeos reales, permitiendo evaluar su comportamiento en diferentes condiciones de calidad y perspectiva. Debido a las limitaciones computacionales disponibles, se optó por un modelo ya entrenado, lo que ha permitido enfocar el trabajo en la optimización de sus resultados. En este sentido, una de las principales aportaciones del proyecto ha sido el desarrollo de una serie de funciones que corrigen las limitaciones del modelo base, mejorando de forma cuantificable la calidad de las detecciones. Estas funciones eliminan trayectorias erróneas, resuelven duplicidades de seguimiento, filtran falsos positivos y refinan la información útil.

Además, se ha implementado un sistema de clasificación automática de los jugadores por equipos, que, junto con la transformación de perspectiva, permite representar de forma precisa y visual su posición sobre un campo de fútbol en 2D. Esta representación final proporciona una herramienta eficaz para el análisis táctico de las jugadas y abre la puerta a futuras aplicaciones en el ámbito deportivo y analítico.

**Agradecimientos:**
Quiero expresar mi más profundo agradecimiento a todas las personas que han estado a mi lado y me han brindado su apoyo a lo largo de la realización de este Trabajo de Fin de Grado. En primer lugar, a mi familia, por su respaldo incondicional, su paciencia y por creer en mí en todo momento. Su apoyo ha sido una base fundamental durante este proceso.

A mis amigos, por acompañarme en los momentos de dificultad, por sus ánimos constantes y por estar siempre dispuestos a ayudar cuando lo he necesitado.

Finalmente, agradezco a mis tutores de TFG, Valero Laparra Pérez-Muelas y Jorge Vila Tomas. Gracias a su ayuda, he podido completar un proyecto sólido y bien estructurado.

# Capítulo 1

# Introducción

El análisis táctico en el fútbol ha dado un salto cualitativo en los últimos años gracias a los avances en visión por computador y aprendizaje automático. Estas tecnologías permiten extraer automáticamente información estructurada sobre la posición y el movimiento de los jugadores, lo que facilita nuevas formas de interpretar y estudiar el juego.

Este trabajo se enmarca en esa línea, desarrollando un sistema capaz de detectar y clasificar los elementos principales presentes en un partido de fútbol, balón, árbitros, jugadores de campo y portero, a partir de vídeos reales. A partir de estas detecciones, se proyecta su posición sobre un campo de fútbol en dos dimensiones, generando una visualización clara y funcional para el análisis táctico.

Uno de los aspectos clave del proyecto ha sido la mejora significativa de la calidad de las detecciones iniciales mediante el diseño e implementación de un conjunto de funciones específicas. Estas funciones corrigen errores habituales en este tipo de modelos, como trayectorias duplicadas, detecciones inconsistentes o elementos mal clasificados, aportando un valor añadido esencial para obtener resultados útiles y fiables.

## 1.1. Motivación

El uso de redes neuronales, es un campo en constante evolución con aplicaciones en múltiples disciplinas. Entre ellas, el análisis de imágenes y la automatización de tareas mediante modelos de aprendizaje profundo han demostrado ser herramientas clave para optimizar distintos procesos. Este ámbito me resulta especialmente interesante debido a la gran variedad de problemas que se pueden abordar con estas técnicas y su potencial para facilitar y mejorar el trabajo en diversos sectores.

Además, mi interés por este tema se vio reforzado durante mis prácticas preprofesionales, donde me enfrenté a un problema similar al que desarrollo en este trabajo. En aquel momento, mis conocimientos sobre redes neuronales y procesamiento de imágenes eran más limitados, lo que despertó en mí la curiosidad para profundizar en esta área y mejorar mis habilidades.

Otro factor clave en la elección de este proyecto es mi afición por el fútbol. Como seguidor del deporte, me resulta especialmente atractivo aplicar técnicas de inteligencia artificial en su análisis, dado el impacto que estas pueden tener en la mejora del rendimiento táctico y estratégico de los equipos.

Por ello, este trabajo representa una oportunidad para combinar mi interés en la inteligencia artificial con mi pasión por el fútbol, explorando nuevas formas de automatizar y visualizar información clave para el análisis del juego. *Figura 1.1*

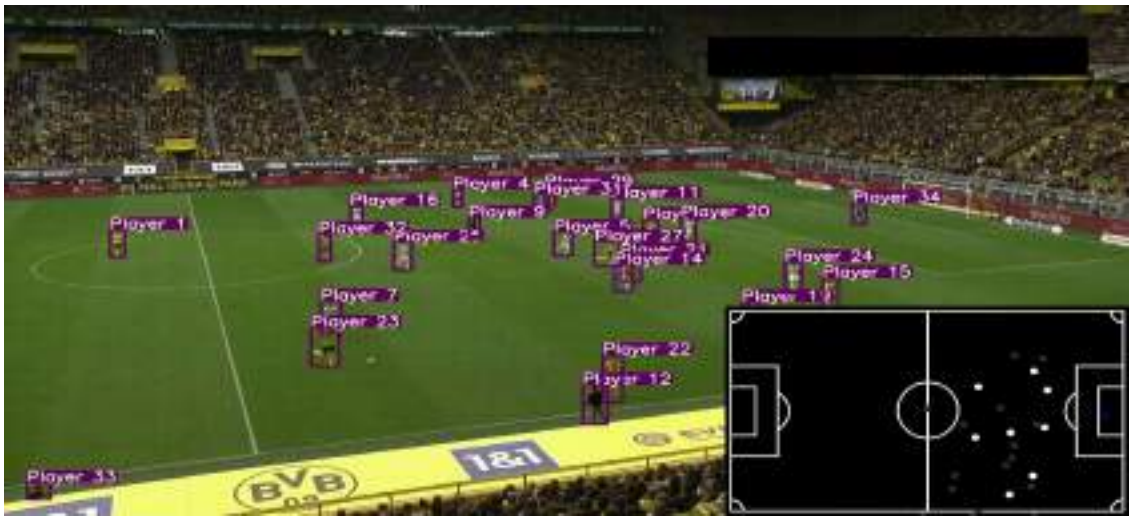



*Figura 1.1:Detección a tiempo real mediante modelo YOLOv8*

# 1.2. Objetivos

El objetivo general de este Trabajo de Fin de Grado es desarrollar un sistema automático que permita detectar y representar los elementos clave que intervienen en un partido de fútbol, tales como los jugadores de campo, los porteros, los árbitros y el balón, a través del uso de redes neuronales. La finalidad última de esta detección es trasladar esta información a una visualización en dos dimensiones del terreno de juego, permitiendo así un análisis táctico más claro, comprensible y automatizado que complemente las herramientas actuales de observación manual.

Para alcanzar este propósito general, se establecen una serie de objetivos específicos que permiten estructurar y guiar el desarrollo del sistema. En primer lugar, se plantea la selección de un modelo de detección previamente entrenado que se adapte adecuadamente al contexto futbolístico y que permita identificar con precisión los distintos elementos del terreno de juego. Dado que no se dispone de los recursos necesarios para entrenar un modelo desde cero, se opta por utilizar un modelo YOLOv8 preentrenado, adaptado específicamente a este tipo de escenas mediante Roboflow.

Una vez seleccionado el modelo, es necesario optimizar su rendimiento en el contexto real de los vídeos analizados. Esto implica desarrollar funciones complementarias que permitan mejorar la calidad de las detecciones, especialmente en situaciones donde el modelo puede fallar, como en casos de aglomeración de jugadores, detecciones duplicadas o trayectorias inconsistentes. Se plantea, por tanto, la creación de un conjunto de funciones que se encarguen de corregir errores comunes en la detección, mantener la coherencia en el seguimiento de los jugadores y filtrar elementos irrelevantes o mal clasificados, como jugadores detectados fuera del campo o árbitros mal etiquetados.

Otro objetivo importante es la clasificación automática de los jugadores por equipo, algo fundamental para cualquier análisis táctico. Dado que los modelos de detección no distinguen directamente entre equipos, se propone una función de clasificación basada en las características visuales de los jugadores, como los colores de las camisetas, que permita agrupar a los jugadores de manera precisa en función de su equipo correspondiente.

Finalmente, toda esta información procesada se debe proyectar de manera visual en un campo de fútbol en 2D. Esta representación final tiene como objetivo facilitar el análisis táctico al mostrar de forma clara y ordenada las posiciones relativas de los jugadores, los árbitros y el balón en cada momento del partido. Se trata de una herramienta que, a pesar de ser desarrollada en un entorno limitado en recursos, busca demostrar el potencial de las redes neuronales aplicadas al

análisis deportivo.

## 1.3. Organización de la memoria

Este trabajo se estructura en cuatro capítulos principales, además de los anexos y la bibliografía:

- Capítulo 1 – Introducción: se presenta el contexto general del trabajo, la motivación personal, los objetivos que se pretenden alcanzar y una descripción de la estructura del documento.

- Capítulo 2 – Materiales y métodos: se detallan los datos utilizados en el estudio, el diseño experimental del proyecto, las herramientas empleadas, la metodología aplicada para llevar a cabo la detección de los elementos y su representación en un campo de fútbol en dos dimensiones.

- Capítulo 3 – Resultados y discusión: se muestran los principales resultados obtenidos tras aplicar el sistema de detección y clasificación, incluyendo visualizaciones y análisis de casos. Se discuten los aciertos, las limitaciones observadas y posibles mejoras.

- Capítulo 4 – Conclusiones y proyección futura: se recogen las conclusiones generales del trabajo, valorando el cumplimiento de los objetivos propuestos, y se plantean posibles líneas de mejora y desarrollo futuro.

Finalmente, se incluyen los anexos, que complementan la memoria con información adicional como fragmentos de código o imágenes relevantes, y la bibliografía, donde se citan las fuentes consultadas para la realización del proyecto.

## 1.4. Estado del arte

En los últimos años, el uso de la inteligencia artificial en el ámbito deportivo ha crecido notablemente, especialmente en deportes como el fútbol. Gracias a los avances en detección de objetos y análisis de imágenes, es posible obtener información útil a partir de vídeos de partidos, como el posicionamiento de los jugadores o el seguimiento del balón. Este tipo de información resulta muy valiosa para entrenadores, analistas y cuerpos técnicos, ya que permite entender mejor lo que ocurre en el campo desde una perspectiva táctica.

*Figura 1.4.1: Análisis TRACAB [9]*

A nivel profesional, existen soluciones muy avanzadas como TRACAB *Figura 1.4.1*, Second Spectrum *Figura 1.4.2*,o StatsBomb *Figura1.4.3*, que utilizan múltiples cámaras, sensores y sistemas de localización para recopilar datos en tiempo real. Sin embargo, estas soluciones tienen un coste muy elevado y solo están al alcance de grandes clubes o competiciones profesionales.

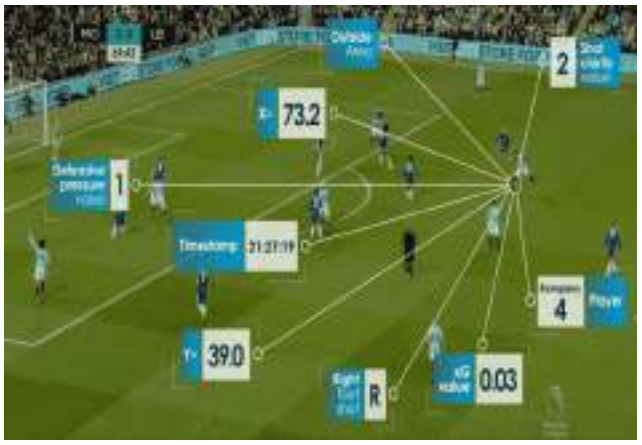



*Figura 1.4.2:Second Spectrum [7]*

*Figura 1.4.3:StatsBomb [8]*

En el campo de la inteligencia artificial y el software libre, los modelos como YOLO (You Only Look Once) han ganado popularidad por su capacidad de detectar objetos con rapidez y precisión [1]. Herramientas como Roboflow permiten entrenar estos modelos para detectar distintos elementos en imágenes o vídeos [6]. No obstante, la mayoría de estos modelos se limitan a detectar objetos en fotogramas individuales, sin ofrecer seguimiento, contexto táctico o representación visual avanzada.

Entre los algoritmos más empleados en visión por computador destacan YOLOv8 por su balance entre velocidad y precisión [4]. Alternativas como Faster R-CNN o DETR presentan mayor precisión en objetos superpuestos, pero con mayores requerimientos de hardware. Se eligió YOLOv8 por su disponibilidad en Roboflow, facilidad de integración y buen rendimiento en contextos deportivos.  Por ejemplo, en el trabajo de Osman et al. (2024) se implementa un sistema de detección de balones de fútbol mediante técnicas de transfer learning y mejora de imagen sobre YOLOv8 [13], obteniendo una precisión del 97.3% tras aplicar estrategias de mejora visual y ajuste fino del modelo. Este resultado evidencia su capacidad para ofrecer un rendimiento robusto incluso en entornos visualmente complejos o con ruido, condiciones similares a las presentes en los vídeos utilizados en este proyecto.

En este proyecto se parte de un modelo preentrenado obtenido desde Roboflow, pero se han desarrollado mejoras propias que permiten convertir esa detección básica en una herramienta útil para el análisis del juego. Se han incorporado funciones que eliminan detecciones incorrectas o poco fiables, como jugadores estáticos o duplicados, y se ha implementado un sistema de seguimiento que mantiene la identidad de cada jugador a lo largo del vídeo, asegurando la coherencia en la representación. Además, se ha desarrollado un sistema automático que permite clasificar a los jugadores en sus respectivos equipos basándose en el color de su indumentaria, lo que facilita su diferenciación en el análisis. Toda esta información procesada se proyecta sobre un campo de fútbol en dos dimensiones, generando una visualización táctica clara y precisa que permite estudiar mejor la distribución y el comportamiento de los jugadores en el terreno de juego.

Gracias a estas mejoras, el sistema propuesto permite obtener información precisa a partir de vídeos comunes, sin necesidad de equipos especiales. Esto lo hace especialmente útil para equipos modestos, escuelas de fútbol o personas interesadas en el análisis deportivo sin grandes recursos. El proyecto, por tanto, no solo tiene un componente técnico, sino también una aplicación práctica y accesible en el mundo del fútbol. Uno de los problemas puede ser el acceso a videos de calidad pero hay varias empresas que pueden proporcionarte estas imágenes como por ejemplo Wyscout [14], plataforma profesional utilizada por clubes y analistas, que proporciona vídeos etiquetados, resúmenes de partidos y cámaras tácticas desde ángulos elevados. O también Second Spectrum [7], compañía especializada en análisis táctico y seguimiento de jugadores.

# Capítulo 2

# Materiales y métodos

Este capítulo describe en detalle los recursos utilizados y el proceso seguido para el desarrollo del sistema de detección y análisis táctico de jugadores en el terreno de juego. En primer lugar, se presentan los datos empleados, incluyendo los vídeos de fútbol seleccionados y los modelos de detección utilizados. A continuación, se expone el diseño experimental, que estructura el flujo de trabajo desde la detección hasta la representación táctica. Finalmente, se describen tanto la metodología general como las herramientas técnicas aplicadas, junto con las funciones desarrolladas específicamente para mejorar la calidad de las detecciones y adaptarlas a los objetivos del análisis.

## 2.1. Datos del estudio

En el desarrollo de este trabajo se han utilizado dos tipos principales de datos, una serie de vídeos que sirven como un ejemplo visual de los modelos de detección aplicados, y los modelos preentrenados encargados de realizar dicha detección.

### 2.1.1. Vídeos del análisis

Se han empleado un total de nueve vídeos de partidos de fútbol, grabados desde una cámara lateral fija y elevada. Esta perspectiva proporciona una visión clara y completa del terreno de juego, asegurando una detección adecuada de los elementos presentes. Los vídeos están en formato MP4, con resoluciones comprendidas entre 720p y 1080p, y duraciones que varían entre 5 a 20 segundos *Figura 2.1.1.1*. Estos vídeos fueron seleccionados por su calidad visual y por presentar distintas situaciones de juego, lo que permite evaluar el rendimiento del sistema en contextos variados. Se ha intentado que en los videos haya diferentes lances del juego y así poder ver cómo se comporta nuestro modelo en estos y ver donde puede haber errores..

También se han utilizado tres videos en otra calidad de imágenes para enfrentar al modelo y ver su funcionamiento *Figura 2.1.1.2*.

Por último se han escogido dos videos de una perspectiva central, detrás de una portería, se hace con el objetivo de comprobar si esto afecta al modelo o si por otra parte la perspectiva diferente no tiene nada que ver. *Figura 2.1.1.3*.


*Figura 2.1.1.1: Imagen 1080p, cámara lateral*

*Figura 2.1.1.2:Imagen menor calidad, cámara lateral*

*Figura 2.1.1.3: Imagen 1080p, cámara frontal*

## 2.1.2. Modelos de detección utilizados

Para la detección de los elementos clave en el campo de juego (jugadores, porteros, árbitros y balón), se ha utilizado un modelo preentrenado en formato YOLO v8, descargado desde Roboflow, plataforma especializada en el entrenamiento y despliegue de modelos de inteligencia artificial. Este modelo fue entrenado previamente por terceros sobre un conjunto de datos etiquetado específicamente para tareas de detección en entornos futbolísticos.

El modelo fue elegido por su capacidad para distinguir entre diferentes tipos de objetos presentes en un partido. La elección de un modelo ya entrenado se justifica por la falta de recursos computacionales y de un conjunto de datos suficientemente amplio y segmentado que permitiera entrenar uno desde cero.

Este modelo fue entrenado por terceros utilizando un conjunto de datos que supera las 5.000 imágenes etiquetadas, con una anotación precisa por clase. Debido a que Roboflow no proporciona siempre detalles exactos del dataset, esta cifra es una estimación basada en la información proporcionada por el proyecto público con el que se ha trabajado.

Este modelo seleccionado pertenece a la familia YOLOv8s, versiones más pequeñas, que contiene aproximadamente 11.2 millones de parámetros. Por ello, se opta por un enfoque basado en transfer learning, reutilizando el conocimiento ya aprendido por el modelo para ajustarlo o aplicarlo directamente en un nuevo entorno sin necesidad de un reentrenamiento completo.

## 2.2. Diseño experimental

El diseño experimental del proyecto se basa en 5 etapas que tienen la finalidad de detectar y representar los elementos clave de un partido de fútbol y representarlos en una visualización táctica en 2D.

### 2.2.1. Selección del modelo de detección

Para llevar a cabo la detección de los elementos clave del terreno de juego se ha optado por utilizar un modelo preentrenado en formato YOLOv8, descargado desde la plataforma Roboflow. Este modelo fue previamente entrenado por terceros sobre un conjunto de datos específicamente etiquetado para escenarios futbolísticos, lo que permite detectar directamente las clases mencionadas sin necesidad de entrenamiento adicional. La elección de este modelo se basa en su precisión, rapidez en la inferencia y facilidad de integración en entornos ligeros como Google Colab.

YOLOv8 (You Only Look Once versión 8) es un modelo de detección de objetos desarrollado por Ultralytics que utiliza una arquitectura basada en redes convolucionales profundas. Se caracteriza por ofrecer detección en tiempo real con un buen equilibrio entre precisión y velocidad. El modelo utilizado en este trabajo es la versión YOLOv8n, que cuenta con aproximadamente 3,1 millones de parámetros y está diseñado para funcionar de forma eficiente incluso en dispositivos sin GPU dedicada. El tamaño de entrada de imágenes para la detección es de 640×640 píxeles.

Frente a otras alternativas disponibles, YOLOv8 presenta ventajas significativas en el contexto del proyecto. Modelos como Faster R-CNN, DETR o SSD ofrecen buena precisión, pero a costa de una velocidad de inferencia muy inferior. Por ejemplo, Faster R-CNN proporciona detecciones muy precisas, pero requiere hardware más potente y tiempo de procesamiento más alto, lo cual no es viable en entornos como Google Colab con recursos limitados. DETR, por su parte, utiliza una arquitectura basada en Transformers que aumenta considerablemente los requisitos computacionales, y no está optimizada para tareas en tiempo real. En cambio, YOLOv8 permite obtener resultados inmediatos y suficientemente precisos, lo que lo convierte en la opción más adecuada dada la disponibilidad de tiempo, recursos y el entorno de ejecución.

Además, para la detección de los puntos clave del campo de juego (líneas, bordes, semicírculos y áreas), se ha empleado un segundo modelo también descargado desde Roboflow y entrenado con arquitectura YOLOv8. Este modelo complementario permite localizar los vértices necesarios para aplicar la homografía y realizar la proyección de las posiciones de los objetos detectados sobre un campo virtual en dos dimensiones. Su uso ha sido esencial para transformar las coordenadas desde la imagen original al plano del terreno de juego, facilitando una visualización táctica más clara y útil para el análisis.

| **Modelo** | **Precisión** | **Velocidad(FPS)** | **Requisitos computacionales** | **Arquitectura** | **Adecuación al proyecto** |
|---|---|---|---|---|---|
| **YOLOv8n** | Alta | Muy alta(+60) | Bajos | CNN | Muy adecuada |
| **Faster R-CNN** | Muy alta | Baja(5-7) | Altos | CNN + RPN | Poco práctica |
| **SSD** | Media | Alta(30-40) | Media | CNN | Aceptable |
| **DETR** | Alta | Muy baja(1-3) | Muy alta | CNN + Transformer | No viable |

*Tabla 1:Comparaciones técnicas de los modelos*

Como se observa en la tabla comparativa *Tabla 1*, YOLOv8n ofrece el mejor equilibrio entre precisión, velocidad de inferencia y requisitos computacionales para este proyecto. Frente a Faster R-CNN y DETR, que si bien presentan una precisión muy alta, requieren entornos con alta capacidad de cómputo y no están pensados para aplicaciones en tiempo real, YOLOv8n permite trabajar de forma fluida incluso en entornos limitados como Google Colab gracias a sus solo 3,1 millones de parámetros y su capacidad para alcanzar más de 60 FPS. Además, al tratarse de un modelo con arquitectura libre de anclas, evita la necesidad de definir manualmente múltiples cajas ancla, regiones predefinidas de diferentes formas y tamaños que los modelos tradicionales utilizan para predecir objetos dentro de una imagen, simplificando su integración y mejorando la adaptabilidad a objetos con formas y posiciones variables, como ocurre en un terreno de juego. En comparación con SSD, que ofrece una buena velocidad pero menor precisión, YOLOv8n resulta más competitivo tanto en detección de objetos pequeños como en la identificación simultánea de múltiples clases. Por todo ello, se justifica su elección como la alternativa más eficiente, versátil y adecuada para cumplir los objetivos de detección en este trabajo.

### 2.2.2. Aplicación del modelo a los vídeos

Cada uno de los siete vídeos seleccionados es procesado de manera individual. Utilizando la librería OpenCV [5], los vídeos se dividen en fotogramas, y sobre cada frame se aplica el modelo de detección mediante la librería ultralytics. La salida del modelo consiste en un conjunto de cajas delimitadoras con las coordenadas de los objetos detectados y su clase correspondiente *Figura 2.2.2*.


*Figura 2.2.2: Fotogramas de como es dividido el video y como es la detección*

## 2.2.3. Mejora de la detección

El modelo de detección utilizado (YOLOv8) proporciona resultados aceptables en la mayoría de los vídeos, pero no está exento de errores, especialmente en contextos deportivos reales como un partido de fútbol, donde las condiciones de iluminación, la proximidad entre jugadores y el movimiento constante pueden dificultar una identificación precisa y estable de los elementos.
Para afrontar estas limitaciones, se han desarrollado un conjunto de funciones que actúan como capa de postprocesado y cuya finalidad es corregir errores comunes del modelo, filtrar resultados erróneos y consolidar la información útil.

### 2.2.3.1. Eliminación de trayectorias estáticas

Esta función *Figura 5.1* detecta y elimina aquellos jugadores cuya posición apenas varía a lo largo del tiempo, esto lo define un umbral, ya que suelen corresponder a detecciones erróneas. Este problema suele ocurrir más en los videos con poca calidad aunque también ha sido aparecido en algún video de buena calidad. Como se puede observar en la *Figura 2.2.3.1*, hay una detección la del jugador 28, que detecta algo inexistente que no se mueve en ningún frame.


*Figura 2.2.3.1: Detección inexistente*

### 2.2.3.2 Eliminación del árbitro y del portero

Ambas funciones *Figura 5.3* y *Figura 5.8*, eliminan de forma explícita las detecciones que corresponden al árbitro y al portero cuando se desea focalizar el análisis solo en los jugadores de campo.
Para identificar al árbitro, se evalúa su posición en el primer frame y se comparan las coordenadas con las primeras apariciones en detecciones. Si hay coincidencia, se eliminan esas trayectorias y sus respectivas imágenes.
El portero se elimina siguiendo una lógica similar. Esta separación permite trabajar más fácilmente con los jugadores de campo durante el proceso de clasificación por equipo y proyección.
Como podemos ver en las imágenes que hay a continuación *Figura 2.2.3.2*, en la primera imagen se detecta un jugador en medio del campo como árbitro y en la segunda vemos como el portero es detectado como jugador.


*Figura 2.2.3.2*

### 2.2.3.3 Eliminación de detecciones con pocos frames

Esta función *Figura 5.6,* descarta aquellas detecciones que solo aparecen en pocos frames, ya que se consideran inestables o ruidos. Se asegura así que las trayectorias analizadas sean consistentes en el tiempo.

### 2.2.3.4 Unión de trayectorias duplicadas

Estas funciones *Figura 5.2* y *Figura 5.7*, agrupan trayectorias que, aunque han sido tratadas como separadas, corresponden al mismo jugador. La primera identifica duplicados exactos, mientras que la segunda analiza similitudes en el espacio y tiempo para unir trayectorias que probablemente sean del mismo individuo. Esto mejora la continuidad del seguimiento y reduce el número de identificaciones incorrectas.

Como podemos ver en la *Figura 2.2.3.4,* se observa como el id 19 se pierde con el paso de frames en el video y se le acaba asignando un nuevo id el 24


*Figura 2.2.3.4*

### 2.2.3.5 Generación de trayectorias completas

En esta fase, el sistema transforma las trayectorias registradas de los jugadores en un formato más completo y estructurado, compatible con las etapas posteriores del proceso.
Específicamente, se reconstruyen las trayectorias incluyendo la información completa de la caja delimitadora, es decir, las coordenadas de las esquinas mínimas y máximas: `(x_min, y_min)` y `(x_max, y_max)`.

Esta información se había reducido previamente al calcular únicamente los puntos centrales `(x, y)` para simplificar ciertos cálculos, pero para muchas tareas posteriores, como la representación gráfica precisa o el análisis de superposición entre jugadores, es necesario recuperar las dimensiones originales de las detecciones.

Por ejemplo, la función de proyección táctica en el campo 2D requiere conocer el ancho y alto de cada detección para estimar correctamente su posición y escala. Del mismo modo, al comparar trayectorias o calcular áreas de influencia, contar con el tamaño exacto de la caja mejora significativamente la precisión del análisis.

### 2.2.3.6. Filtrado de jugadores fuera del campo

Una vez transformadas las coordenadas a una proyección 2D del campo, esta función filtra los jugadores cuyas posiciones proyectadas se encuentran fuera de los límites del terreno de juego *Figura 5.4*. Esto garantiza que solo se representen elementos válidos en la vista táctica.
Estas funciones forman un bloque esencial del sistema, permitiendo mejorar sensiblemente la precisión de las detecciones y asegurar que los resultados finales sean útiles para el análisis táctico, a pesar de las limitaciones del modelo de partida.
Esta función es útil por si se detectan a los jugadores que están por fuera calentando o si detecta al entrenador como jugador *Figura 2.2.3.6*.

*Figura 2.2.3.6*

## 2.2.4. Clasificación por equipo

A través de dos funciones de clasificación, se asigna a cada jugador a su equipo correspondiente utilizando información visual ,como el color de la camiseta, y agrupamiento espacial.

### 2.2.4.1 Clasificación de jugadores por equipo

Basándose en las primeras apariciones y en características visuales, principalmente color, esta función asigna a cada jugador a su equipo. Se emplea una lógica de agrupamiento no supervisado, que distingue a los jugadores de cada conjunto sin necesidad de etiquetas manuales *Figura 5.5*.

### 2.2.4.2 Asignación de equipo al portero

Una vez clasificados los jugadores, esta función asigna también el equipo correspondiente al portero *Figura 5.9*, permitiendo integrarlo en el sistema de representación sin conflictos en los IDs o colores del equipo. Para ello se utiliza un algoritmo de vecindad donde se busca el equipo que está más cerca al portero para así asignarlo a este *Figura 2.2.4.2*.

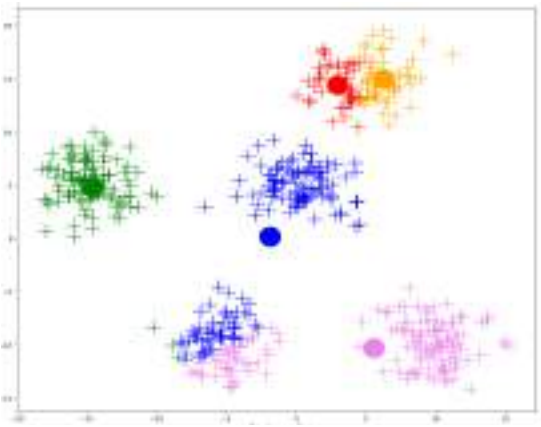


*Figura 2.2.4.2: Algoritmo de vecindad K-Means*

## 2.2.5 Proyección en campo 2D

Finalmente, las posiciones de los elementos detectados se representan en un campo de fútbol simulado en dos dimensiones. Para lograrlo se crea una vista del campo utilizando un modelo de detección de líneas de campo y un transformador de perspectiva que proyecta las posiciones reales detectadas al plano del campo.Con el campo dibujado y con las coordenadas de los objetos se hace una representación sin ningún problema.

Esta visualización permite analizar de forma más clara la distribución táctica y facilita el estudio del juego desde una perspectiva más abstracta y útil para el análisis técnico.

Como podemos observar se plasma en el campo los jugadores árbitros y balón, con su correcta clasificación por equipos, aunque primeramente como observamos en la imagen hay un jugador no detectado, esto nos hace ver que las funciones funcionan *Figura 2.2.5*.


*Figura 2.2.5: Proyección de un frame en el campo*

# 2.3. Metodología

La metodología utilizada en el proyecto está basada en el uso de modelos orientados a la detección ya entrenados, apoyada en técnicas de procesamiento y mejora de detecciones, clasificación y visualización táctica.

## 2.3.1. Entorno de desarrollo

El desarrollo se ha realizado íntegramente en Google Colab, un entorno que permite ejecutar código en GPU de forma gratuita, lo que ha facilitado la aplicación de modelos de detección avanzados sin requerir hardware local.

### 2.3.1.1 Librerias

Las librerías más importantes del proyecto son las siguientes:

- ultralytics: para la integración y ejecución de modelos YOLOv8.
- supervision: para manipular y visualizar resultados de detección.
- opencv: para lectura, procesamiento y visualización de vídeos e imágenes.
- numpy[10] y matplotlib[3]: para operaciones matemáticas y visualización. La librería numpy por problemas de librerias se tiene que eliminar la versión de colab y descargar una versión posterior
- Roboflow: para la descarga del modelo ya entrenado.

Otras librerías utilizadas son las siguientes:

- os: utilizada para manejar rutas de archivos, acceder al sistema de archivos
- userdata: para acceder a datos del usuario en Google Colab
- cv2_imshow: para visualizar imágenes ya que la función de OpenCV no funciona correctamente
- tqdm: utilizada para mostrar barras de progreso cuando se procesan vídeos frame a frame
- inference: para llamar a la función que carga el modelo de detección
- sports.: se utiliza para el uso de funciones en el ámbito deportivo, como la clasificación de equipos o encontrar los puntos clave del campo

### 2.3.2. Fases del sistema

Para lograr una detección precisa de los jugadores y una representación táctica útil del juego, el sistema desarrollado se estructura en distintas fases consecutivas. Cada una de estas etapas cumple una función específica dentro del flujo de trabajo, permitiendo transformar los datos de entrada, en este caso vídeos de partidos de fútbol, en una visualización clara y depurada sobre un campo en dos dimensiones.

#### 2.3.2.1. Carga y procesamiento de los vídeos

Los vídeos de entrada se cargan frame a frame utilizando OpenCV. Cada imagen se analiza con el modelo YOLOv8 para detectar los objetos: jugadores, porteros, árbitros y balón. El modelo devuelve coordenadas, clases y puntuaciones de confianza.

#### 2.3.2.2. Corrección y filtrado de detecciones

Una vez realizada la detección inicial, es necesario aplicar una serie de mecanismos de corrección y filtrado. Estos filtros están diseñados para abordar problemas comunes en la detección, como errores por movimiento limitado, identificadores duplicados o detecciones espurias.

En primer lugar, se eliminan aquellos jugadores cuya trayectoria no presenta desplazamientos significativos a lo largo del tiempo, ya que suelen ser falsos positivos o elementos ajenos al juego ,como entrenadores o personas estáicas,. También se descartan los objetos que aparecen en muy pocos fotogramas, lo que normalmente indica detecciones poco consistentes o errores de seguimiento.

Otro tipo de corrección importante es la unificación de trayectorias que pertenecen al mismo jugador pero han sido divididas en múltiples identificadores, bien por cruces con otros jugadores o por errores en el seguimiento. Además, se eliminan ciertos roles como el portero o el árbitro cuando no son relevantes para el análisis táctico que se desea realizar, centrándose así únicamente en los jugadores de campo.

Por último, se filtran aquellos jugadores que, tras la transformación de perspectiva, aparecen fuera de los límites del terreno de juego. Esta verificación garantiza que el análisis se realice únicamente con jugadores que realmente participan en la acción dentro del campo.

Todas estas operaciones están implementadas mediante funciones propias desarrolladas específicamente para este proyecto. Su implementación detallada puede consultarse en el repositorio de código disponible en GitHub [11]. Estas funciones contribuyen de forma clave a mejorar la precisión, coherencia y utilidad de las detecciones iniciales.

#### 2.3.2.3. Seguimiento temporal
Se implementa una lógica de primera aparición y asignación de IDs persistentes, lo que permite hacer un seguimiento individual de cada jugador a lo largo del tiempo, incluso en los casos donde el modelo falla temporalmente por la cercanía entre jugadores.

#### 2.3.2.4. Clasificación de jugadores por equipo
Mediante la función clasificacion_equipos(), se clasifican los jugadores según su indumentaria utilizando técnicas de agrupamiento por color. A continuación, se utiliza resolve_goalkeepers_team_id() para asignar también el equipo correspondiente al portero, esta función se basa en clusters asignando el portero al cluster de jugadores más cercano.

#### 2.3.2.5. Proyección en campo 2D
Para visualizar el resultado, se utiliza un modelo de detección de líneas de campo "FIELD_DETECTION_MODEL" que localiza puntos clave del terreno. Estos puntos se usan como referencia para definir una transformación de perspectiva mediante "ViewTransformer". Las coordenadas de los objetos detectados en el vídeo se proyectan en un campo en dos dimensiones, generando una vista táctica simplificada. Esta representación se realiza con la función "draw_points_on_pitch()", que pinta sobre el campo las posiciones del balón, árbitros y jugadores de cada equipo con colores diferenciados.

#### 2.3.2.6. Evaluación de la mejora
Se compara el número de detecciones válidas antes y después de aplicar las funciones de corrección, mostrando en pantalla la evolución del rendimiento y la ganancia obtenida mediante los ajustes realizados.

La metodología ha permitido construir un sistema robusto de detección, seguimiento y representación táctica, aprovechando recursos limitados y partiendo de modelos ya entrenados. La combinación de funciones específicas ha compensado las limitaciones del modelo base y ha proporcionado resultados válidos para el análisis del juego.

En la Tabla 3, podemos ver la diferencia que hay entre la detección antes de aplicar las funciones y cuando estas son utilizadas. Se puede ver que mejora tanto como la detección de jugadores como su clasificación.

## 2.4. Algoritmos utilizados y justificación
Este apartado presenta los principales algoritmos aplicados en el desarrollo del sistema, así como su justificación en función del problema tratado.

### 2.4.1. YOLOv8 (You Only Look Once v8)
YOLO es una arquitectura de detección de objetos en tiempo real basada en redes convolucionales profundas. Se eligió YOLOv8 por su rendimiento competitivo en tareas con múltiples clases, su velocidad de inferencia adecuada para vídeo, la facilidad de uso que ofrece a través de la librería Ultralytics y su buena integración con plataformas como Roboflow, que además proporciona soporte directo para exportar modelos entrenados. Alternativas como Faster R-CNN o DETR pueden ofrecer mayor precisión en ciertos contextos, pero requieren una cantidad considerable de recursos computacionales y no son tan adecuadas para entornos con hardware limitado, como es el caso de Google Colab, que ha sido el entorno principal de desarrollo de este proyecto *Figura 2.4.1*.


*Figura 2.4.1: Tipos de modelos de detección*

## 2.4.2. Filtrado de trayectorias (algoritmo propio)

Para mejorar la calidad de los datos obtenidos del modelo de detección, se han implementado funciones de filtrado que aplican criterios simples pero eficaces. Estas funciones eliminan trayectorias con desplazamientos mínimos, considerando que corresponden a detecciones estáticas que probablemente son erróneas; descartan detecciones que aparecen en un único frame, dado que se consideran inestables o ruido; y unifican trayectorias separadas pero que, por su similitud espacial y continuidad temporal, probablemente pertenecen al mismo jugador. Estos algoritmos, aunque basados en reglas heurísticas básicas, han demostrado ser eficaces para reducir significativamente los errores más comunes del modelo base y mejorar la consistencia de los resultados a lo largo del tiempo.

## 2.4.3. Agrupamiento por color (k-means)

Para clasificar a los jugadores por equipo se ha utilizado un método de agrupamiento basado en el color dominante de sus camisetas. A partir de las primeras apariciones de cada jugador se calcula el color medio de su indumentaria y se agrupan en dos conjuntos aplicando una variante del algoritmo k-means. Este enfoque no supervisado permite separar visualmente a los jugadores sin necesidad de etiquetas manuales, resultando especialmente útil en contextos donde no se dispone de anotaciones previas ni se requiere un aprendizaje supervisado *Figura 2.2.4.2*.

## 2.4.4. Homografía para transformación de perspectiva

Para representar las posiciones de los jugadores y otros elementos sobre un campo de fútbol en dos dimensiones, se ha utilizado una transformación por homografía *Figura 2.2.4*. Esta técnica permite proyectar los puntos desde el plano de la imagen original (en perspectiva) a un plano simulado del terreno de juego. La homografía se calcula a partir de los puntos de referencia extraídos por el modelo de detección de líneas de campo, permitiendo así una visualización táctica clara, precisa y coherente con la distribución real de los jugadores sobre el terreno.

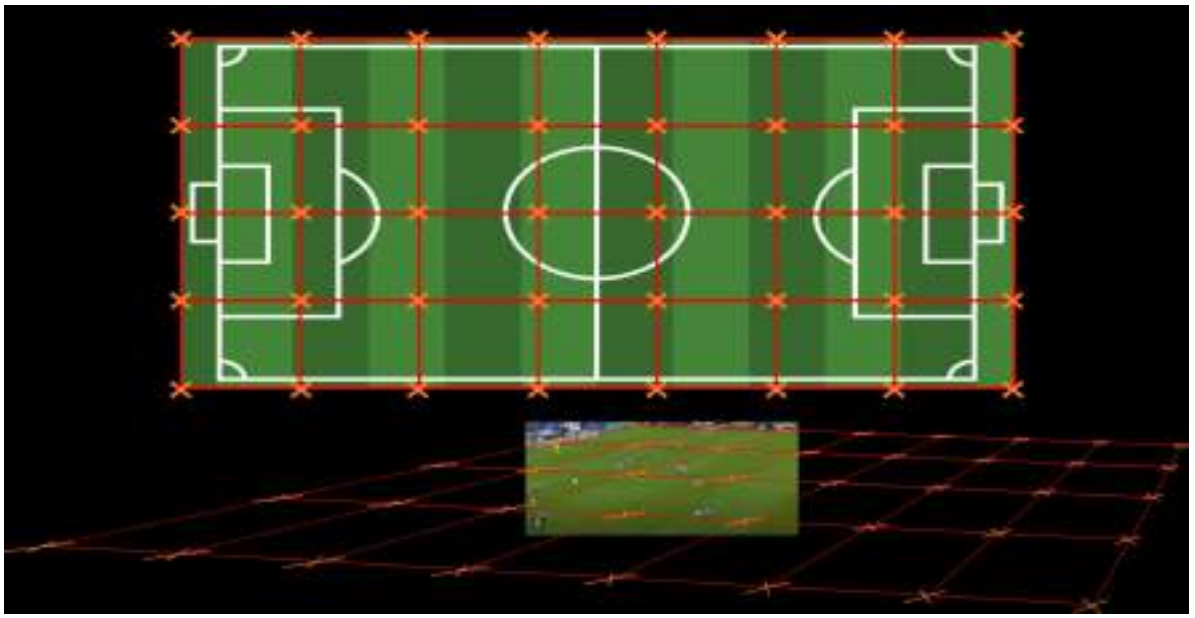


*Figura 2.4.4: Homografía para transformación*

# Capítulo 3

# Resultados y Discusión

En este capítulo se presentan los resultados obtenidos tras aplicar el sistema de detección y representación táctica desarrollado sobre un conjunto de 14 vídeos de fútbol. Se analizan tanto los resultados cuantitativos, mediante una comparación entre las detecciones antes y después de aplicar las funciones correctoras, como los resultados cualitativos, observando casos representativos donde las mejoras han sido significativas o donde el modelo ha mostrado limitaciones. La discusión se organiza en dos partes: primero se analiza el comportamiento del sistema vídeo por vídeo, y después se extraen conclusiones generales sobre la eficacia del modelo base, el impacto de las funciones implementadas y las condiciones que afectan al rendimiento del sistema.

## 3.1. Resultados

El sistema se ha evaluado sobre un total de 14 vídeos de fútbol, procesando cada uno mediante detección, filtrado, seguimiento, clasificación y proyección táctica. Estos videos se dividen en 9 videos los cuales tienen la mejor calidad y el ángulo donde se graba es el mejor. Por otra parte se tienen dos videos grabados de diferente ángulo con una calidad buena, se estima que estos videos no se van a poder proyectar bien ya que el modelo está entrenado para detectar jugadores en un ángulo diferente. Por último se han escogido tres videos con una menor calidad para ver si a menor calidad la detección empeora al detectar de peor manera los objetos tratados en el proyecto.

El objetivo es detectar correctamente a los jugadores de campo que deben estar presentes, 10 por equipo como máximo, excluyendo porteros y árbitros ya que estos no son tan importantes para un análisis táctico.

Primero vamos a comentar video a video sus resultados, dando toda la información disponible que tengamos.

### Video 1

Observamos que en el primer frame la detección no es correcta ya que de 20 jugadores se detectan 19 con una buena detección del arbitraje *Figura 3.1.1*.


*Figura 3.1.1: Primer frame Video 1*

En el frame 8 se detecta ya el jugador que falta de la detección, id 22, estos nos indica que el primer paso de encontrar a todos los jugadores ya lo tendríamos *Figura 3.1.2*.


*Figura 3.1.2: Octavo frame Video 1*

Durante el proceso se siguen asignando id que ya no nos interesan, con los siguientes id 24,26,28 *Figura 3.1.3*.

*Figura 3.1.3: Asignación id 24,26,28*

Seguidamente se aplican las funciones, las cuales nos indican que no hay portero y además eliminan los 3 ids sobrantes diciéndonos que estos jugadores ya tenían una asignación posterior. Vemos que en el video solo ha hecho falta el uso de la función de trayectorias parecidas.

Se detecta en un frame 20 jugadores indicandolo como máximo, este frame es el 8 donde ya hay 20 jugadores identificados. Mediante este valor calculamos el porcentaje de aciertos antes y después de las funciones, donde vemos que al principio hay 19 detecciones y al final 20. Esto deja claro que las funciones en este video han mejorado la detección para su posterior análisis táctico.

Finalmente se detecta que ningún jugador está fuera del campo y se plasma en el campo de 2D para su posterior análisis *Figura 3.1.4*.

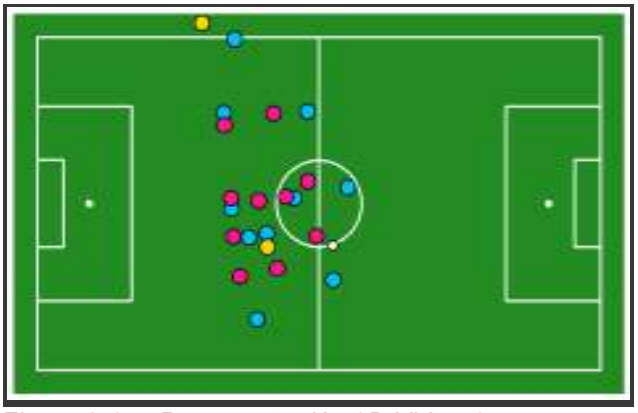


*Figura 3.1.4: Representación 2D Video 1*

Las funciones eliminan correctamente los IDs duplicados o efímeros (24, 26, 28) y validan que no hay portero ni árbitro mal clasificado. El resultado final es óptimo: se detectan 20 jugadores válidos, todos correctamente proyectados en el campo 2D. Es un ejemplo claro de cómo las funciones mejoran sin afectar detecciones válidas, permitiendo una reconstrucción táctica coherente.

## Vídeo 2

Observamos que en el primer frame la detección inicial identifica a 17 jugadores, faltando 3 jugadores para la correcta detección, con una correcta detección de los árbitros. Se observa a simple vista el problema de que el portero es detectado como jugador *Figura 3.2.1*.


*Figura 3.2.1: Primer frame Video 2*

A lo largo de las detecciones se observa que hay una detección masiva de los jugadores, esto es causado al haber una aglomeración de jugadores y no se le puede hacer un seguimiento correcto de los ids *Figura 3.2.2*.


*Figura 3.2.2: Malas detecciones Video 2*

Al haber una asignación masiva de ids las funciones causan problemas ya que son sensibles a tantos datos, además este problema se ve aumentado al trabajar con trayectorias muy cercanas al ser una falta lateral donde se agrupan muchos jugadores. Esto causa que se eliminen ids de jugadores que no deberían ser eliminados, como el 11,12,15,16.

El número máximo de jugadores detectados es 20 pero esto es producido porque el portero es contado como uno ya que en ningún momento se detecta a los 20 jugadores correctamente. El video nos da resultados negativos al perder detecciones mediante el proceso y encontrar menos jugadores que al principio, 17 antes y 16 después.

Como podemos observar la imagen cargada en 2D es errónea ya que nos ubica menos jugadores de los que debería ubicar y hay jugadores con dos trayectorias *Figura 3.2.3*.

*Figura 3.2.3: Representación 2D Video 2*

Las funciones, al aplicar criterios de fusión y eliminación, terminan descartando jugadores válidos (ids 11, 12, 15, 16). El resultado final es inferior al inicial (de 17 a 16 jugadores), y la representación 2D es inconsistente. Este vídeo evidencia la limitación del sistema ante escenas con poco espacio libre, donde los errores del modelo base se amplifican en las funciones de limpieza.

## Vídeo 3

En este vídeo, desde el primer frame se detectan sólo 17 jugadores, y se observa a simple vista que hay un jugador, el que está sacando el corner, que es asignado como árbitro *Figura 3.3.1*.

*Figura 3.3.1: Primer frame Video 3*

Podemos ver que a lo largo de la detección hay muchas detecciones erróneas sobre todo con los árbitros y jugadores/personas que están fuera del campo, esto no impide que se vayan detectando jugadores que están en la aglomeración de gente en el área y el jugador encargado de sacar el córner *Figura 3.3.2*.

*Figura 3.3.2: Malas detecciones Video 3*

Al aplicar a las funciones se eliminan los ids 36, 17, 19, 20, 23, por estar estáticos, eliminando así los jugadores detectados fuera y jugadores que se han detectado en pocos frames y no se ha movido, esto nos vendrá bien. Se elimina el id 18 por pertenecer a un árbitro y 27 por pertenecer al portero. Por último se eliminan los ids 35, 28, 29 por pertenecer a jugadores ya asignados.

Esto provoca que en la detección no se detecten los 20 jugadores que hay de máximo pero se detectan 19 más el portero con una buena clasificación, mejorando el resultado inicial de 17 jugadores detectados.

El resultado final se plasma de manera correcta en el campo de 2D donde observamos los 19 jugadores más el portero y los dos árbitros *Figura 3.3.3*.

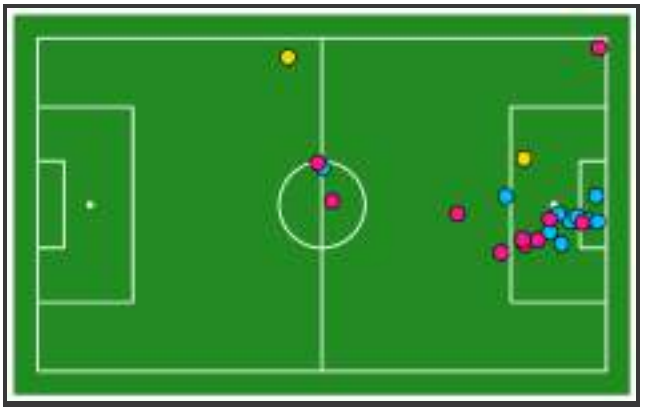


*Figura 3.3.3: Representación 2D Video 3*

Las funciones ayudan a eliminar elementos no válidos estáticos, árbitros, portero mal clasificado y mejoran el conteo hasta alcanzar 19 jugadores + portero. Aunque no se llega a los 20, la mejora respecto al punto de partida es clara. La proyección 2D final es correcta. Es un ejemplo de cómo el sistema puede recuperar precisión en condiciones visuales subóptimas, aunque con margen de mejora.

## Vídeo 4

Observamos que en el primer frame la detección inicial identifica a 19 jugadores, con una correcta detección de los árbitros. Se observa que falta un jugador defensivo que va de blanco que está en la misma banda donde está el balón *Figura 3.4.1*.

*Figura 3.4.1: Primer frame Video 4*

En los primeros frames se detecta al portero como jugador dos veces con id 23 y 24 *Figura 3.4.2*.

*Figura 3.4.2: Detecciones erróneas del portero*

Finalmente en mitad del proceso de detección se detecta al jugador que nos falta por encontrar, id 27, cumpliendo el objetivo de la detecció *Figura 3.4.3*.

*Figura 3.4.3: Detección id 27*

Después de esto hay dos asignaciones de id más que ya no nos interesan y que esperamos que las funciones los eliminen.

Se eliminan los siguientes id 23,24,31 al pertenecer al portero y no encontrar movilidad o directamente al detectar que son el portero. Por último se elimina el id 30 ya que pertenece a un jugador ya asignado.

Con todo esto se detectan los 20 jugadores máximos, con una buena clasificación de cada uno de los jugadores, árbitros y portero.

Todo esto es plasmado en el campo 2D donde podemos observar que todo está correcto, 20 jugadores, 3 árbitros y el portero *Figura 3.4.4*.

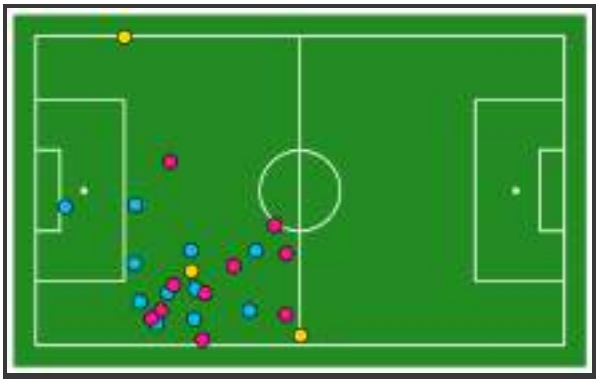


*Figura 3.4.4: Representación 2D Video 4*

Las funciones eliminan correctamente los IDs asociados al portero y a trayectorias duplicadas. El resultado es una detección limpia con 20 jugadores válidos, los tres árbitros y el portero correctamente clasificados. Es un ejemplo de detección inicial sólida que se perfecciona mediante las funciones de validación y limpieza.

## Vídeo 5

En este video observamos que de primeras ya se han detectado los 20 jugadores necesarios para la detección, pero vamos a aplicar las funciones para comprobar que no estropeara esta detección y que todos sean jugadores de verdad *Figura 3.5.1*.


*Figura 3.5.1: Primer frame Video 5*

Durante el proceso de detecciones vemos que algunos jugadores son detectados como árbitros en algún frame perdiendo su id y posteriormente asignando otro, en este caso el id 10 se pierde y se asigna el 24 *Figura 3.5.2*.



*Figura 3.5.2: Malas detecciones Video 5*

Cuando aplicamos las funciones se eliminan los ids 25,26 por poco movimiento de los id ya que son id que aparecen en pocas detecciones y no se mueven. También se elimina el id 24 que ya tenía un id asignado anteriormente.

Esto provoca que nos quedemos con la imagen inicial para la detección ya que es la correcta, en ella observamos los 20 jugadores 2 árbitros y el portero, todo esto se plasma en el campo de 2D *Figura 3.5.3*.

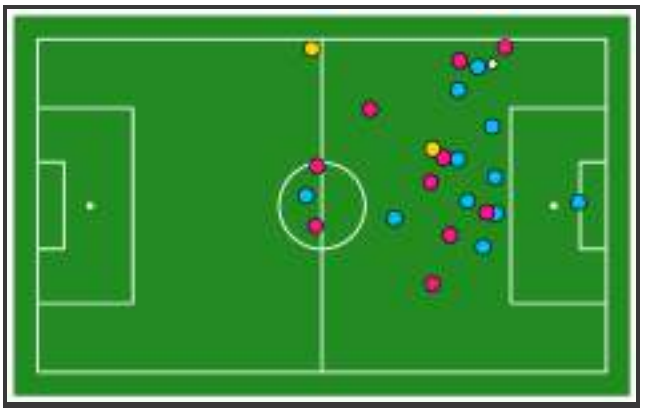


*Figura 3.5.3: Representación 2D Video 5*

Las funciones identifican y eliminan correctamente los IDs sin movimiento (25, 26) y los duplicados (24). La imagen inicial era buena, y se mantiene gracias a la intervención controlada de las funciones. Este vídeo demuestra que el sistema no sobre corrige en casos donde ya hay buena detección, lo cual es clave para su fiabilidad.

## Vídeo 6

Se detectan 19 jugadores de primeras con 3 árbitros, lo cual es negativo porque uno de los jugadores detectados es el portero. Además hay 20 jugadores sin contar el portero *Figura 3.6.1*.

*Figura 3.6.1: Primer frame Video 6*

En los siguientes frames se detectan los dos jugadores que faltan en la detección, aunque también se pierde un id ya que es detectado como árbitro pero no afecta en nada en el proceso *Figura 3.6.2*.

*Figura 3.6.2: Detecciones Video 6*

Las funciones no se aplican ya que no hay excesos de id, solo se aplica la función de portero pero como este no es detectado en ningún momento no se elimina el id que le pertenece a él como jugador.

Esto provoca que aunque se haya detectado a todos los jugadores y debería de ser correcta la detección final del proceso, no lo es. Ya que de 19 jugadores incluyendo al portero como uno de ellos pasamos a 21 contando al portero como un jugador.

Esto es plasmado en el campo 2D y vemos que la imagen parece correcta pero el portero está mal clasificado ya que no se ha detectado y no se aplica la función para etiquetar a un equipo y lo clasifica como un jugador más *Figura 3.6.3*.

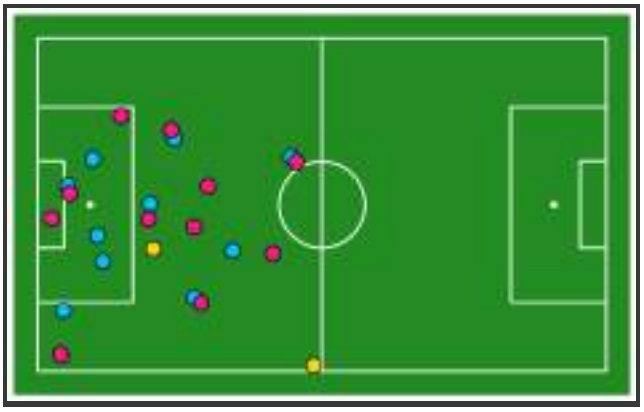


*Figura 3.6.3: Representación 2D Video 6*

Se alcanza un conteo de 21 jugadores, lo que rompe el límite lógico del sistema. El campo 2D parece correcto visualmente, pero está conceptualmente mal, pues el portero no está separado del análisis. Este vídeo resalta una limitación importante: cuando el modelo base no identifica una clase portero, el sistema no puede aplicar las funciones adecuadas.

## Vídeo 7

Este video es diferente ya que el frame a analizar tiene 19 jugadores, es decir menos cantidad que en todos los vistos anteriormente. Se observa que se detectan todos los jugadores de manera correcta, también los árbitros, así que vamos a aplicar las funciones para comprobar que no estropea esta detección y que en realidad todo lo detectado sean jugadores *Figura 3.7.1*.


*Figura 3.7.1: Primer frame Video 7*

Durante el proceso se van acercando los jugadores provocando así que se pierdan algunos ids asignando a estos jugadores nuevos ids. Que posteriormente nuestra función eliminará por ser ids que ya pertenecen a otros jugadores o por ser ids que aparecen tan poco que no se detecta movimiento *Figura 3.7.2*.




*Figura 3.7.2: Errores en la detección Video 3*

Cuando vamos a calcular el porcentaje de jugadores encontrados vemos que hay un problema y es que en algún momento el árbitro es detectado como jugador provocando que el número máximo sea 20 cuando en realidad son 19. Esto no afecta a la representación ya que los jugadores y árbitros son plasmados correctamente en el campo 2D *Figura 3.7.3*.

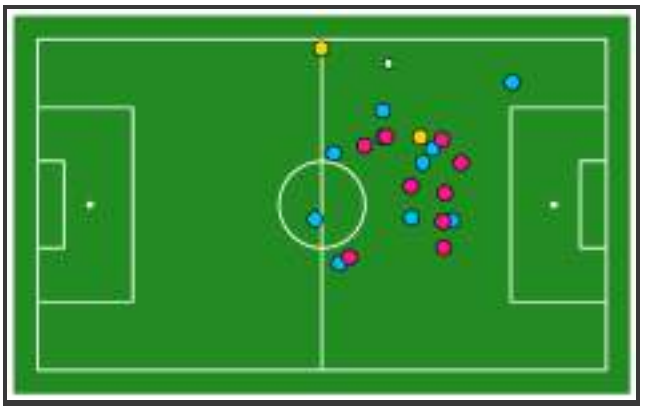


*Figura 3.7.3: Representación 2D Video 7*

Las funciones eliminan trayectorias duplicadas, corrigen IDs nuevos innecesarios y evitan que el conteo se dispare. El número máximo llega a 20 ,por error del árbitro clasificado como jugador, pero se ajusta correctamente al final. Es un ejemplo de cómo el sistema corrige bien situaciones de movimiento reducido y errores temporales.

## Vídeo 10

En este video observamos que de primeras ya se han detectado los 20 jugadores necesarios para la detección, pero vamos a aplicar las funciones para comprobar que no estropeara esta detección y que todos sean jugadores de verdad y no sea una mala detección *Figura 3.8.1*.

*Figura 3.8.1: Primer frame Video 10*

Durante el proceso se van acercando los jugadores provocando así que se pierdan algunos ids asignando ,a estos jugadores nuevos ids. Que posteriormente nuestra función eliminará por ser ids que ya pertenecen a otros jugadores o por ser ids que aparecen tan poco que no se detecta movimiento.

En un momento dado se detecta el portero como jugador pero este es eliminado posteriormente por la función de eliminar portero *Figura 3.8.2*.


*Figura 3.8.2: Frame portero como jugador*

Finalmente se detectan los 20 jugadores sin ninguna eliminación innecesaria ni ningún id sobrante en la representación del campo, con una detección de portero y del árbitro correcta y todo plasmado en el campo 2D. Cabe destacar que el balón no se ha detectado correctamente ya que se detecta un balón fuera del campo *Figura 3.8.3*.

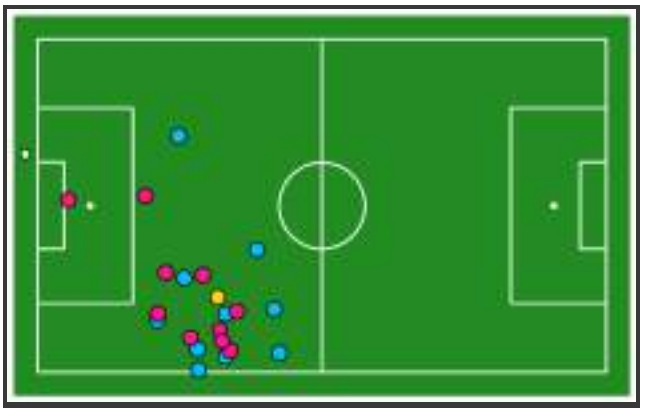


*Figura 3.8.3: Representación 2D Video 10*

El seguimiento mantiene la estabilidad de las trayectorias, y la proyección en el campo 2D es precisa. El único fallo menor es la detección incorrecta del balón fuera del campo. El vídeo demuestra que el sistema funciona bien incluso cuando el modelo comete un error parcial de clasificación.

## Vídeo 11

En este video observamos que de primeras se detectan 19 jugadores de primeras con 2 árbitros, aun nos falta un jugador por detectar ya que el número total de jugadores en este frame es 20 *Figura 3.9.1*.


*Figura 3.9.1: Primer frame Video 11*


*Figura 3.9.2: Detección jugador faltante Video 11*

Durante el proceso en el segundo frame se detecta el jugador que falta para completar la detección de todos los jugadores y no hay más detecciones de jugadores en todo el proceso. Las funciones no eliminan ningún id enseñando su buen rendimiento *Figura 3.9.2*.

Finalmente con todos los id detectados sólo queda plasmarlo todo en el campo de 2D donde podemos apreciar que todo es correcto, tanto la detección de jugadores y árbitros como la clasificación posterior de estos *Figura 3.9.3*.

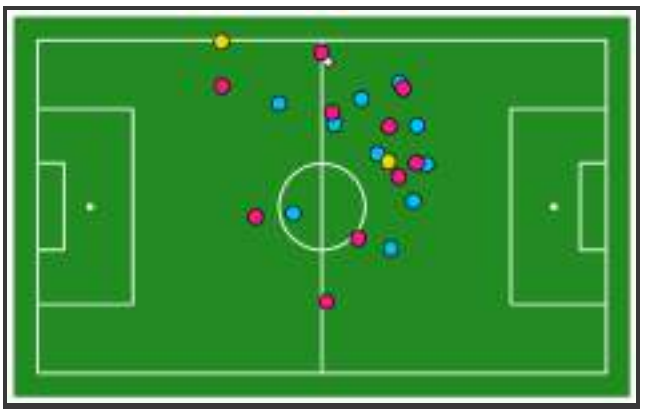


*Figura 3.9.3: Representación 2D Video 11*

Las funciones no eliminan IDs válidos y no generan falsos positivos. El campo 2D proyecta correctamente todos los elementos. Es el ejemplo más claro de resultado ideal del sistema bajo condiciones óptimas, validando la eficiencia general del pipeline.

## Video 15,16

Vamos a seguir con el análisis de los videos grabados en diferente ángulo, parte posterior de la portería.

Estos videos al estar en diferente ángulo tienen un número muy elevado de detecciones *Figura 3.10.2*, esto puede causar un problema en nuestro proceso de detección ya que es sensible a un número tan elevado de detecciones. Estas imágenes enseñan la primera detección de los videos *Figura 3.10.1*.

*Figura 3.10.1: Primer frame Video 15*

*Figura 3.10.2: Primer Frame Video 16*

Durante el proceso en ningún momento se encuentra el portero, esto es un problema ya que nunca nos lo va a etiquetar y su id nunca desaparecerá. Por lo que respecta a los árbitros en varios frames estos son detectados como jugadores. También sucede esto con los entrenadores y con la gente de alrededor del terreno de juego *Figura 3.10.3*.

*Figura 3.10.3: Frames erróneos de los Videos 15,16*

Además las funciones no funcionan correctamente por las trayectorias que se calculan y no se llega a eliminar los ids que no nos sirven.

Por último, la detección de puntos clave del campo falla en ambos casos. Esto impide que se aplique correctamente la homografía para transformar las coordenadas y proyectar los jugadores sobre un campo en 2D, dejando la visualización táctica inservible.

Las funciones desarrolladas intentan actuar, pero fallan en la mayoría de los casos debido a trayectorias erróneas, IDs superpuestos o trayectorias estáticas generadas por objetos que no forman parte del juego real. La limpieza es insuficiente, y el resultado final no es válido para el análisis táctico.

Estos vídeos exponen una de las principales limitaciones del sistema, la dependencia directa del ángulo de cámara y perspectiva para lograr resultados válidos. Se confirma que el modelo actual y sus funciones asociadas no están preparados para trabajar con grabaciones desde ángulos no frontales o no elevados, por lo que cualquier análisis táctico automatizado en estas condiciones es inviable sin un reentrenamiento o adaptación específica del modelo de detección

## Video 20

Para terminar, vamos a analizar los videos con diferente calidad de video para comprobar lo sensible que puede ser el modelo y las funciones a estos.

Como observamos en este video hay una gran cantidad de ids detectados de primeras, donde al portero se le detecta como jugador *Figura 3.11.1*.

*Figura 3.11.1: Primer frame video 20*

En el proceso de detección hay varios problemas ya que al tener menor calidad muchos jugadores dejan de identificarse y se identifican posteriormente, igualmente pasa con los árbitros que los detecta como jugadores o jugadores que detecta como árbitros *Figura 3.11.2*.

*Figura 3.11.2: Problemas con los árbitros video 20*

En cuanto a las funciones, estás se aplican y eliminan gran cantidad de id pero finalmente no puede eliminar tanta cantidad de id ya que hay muchos errores. Esto provoca que de 21 detecciones principales se pase a 23. La cual una es eliminada ya que corresponde a un jugador que está fuera del campo, quedándonos con 22 detecciones, cosa que es imposible *Figura 3.11.3*.

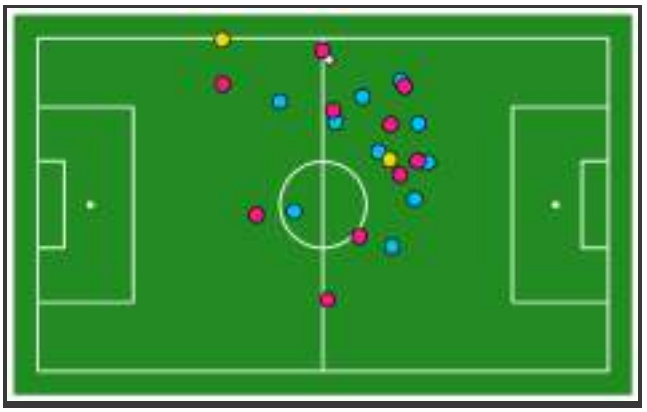


*Figura 3.12.3: Representación 2D Video 20*

Las funciones de eliminación de trayectorias estáticas y duplicadas logran reducir parcialmente el ruido, pero no son suficientes para alcanzar un resultado limpio. Se llega a un conteo final de 22 jugadores, una cifra superior al límite real de 20, lo cual invalida el análisis táctico.  El sistema muestra una fuerte sensibilidad ante resoluciones bajas, ya que los errores de clasificación y duplicación se agravan y no pueden corregirse completamente con las funciones actuales.

## Video 23

Aquí podemos observar que el modelo de detección detecta a todos los jugadores de primeras, en total 19, con los dos árbitros, aunque uno de los árbitros es etiquetado como árbitro y jugador causando problemas. Así que comprobaremos si de verdad la calidad puede afectar viendo si empeora esta situación *Figura 3.13.1*.

*Figura 3.13.1: Primer frame Video 23*

Durante la detección algun id se va perdiendo con el tiempo pero se le asigna otro id a ese jugador, en algún momento el árbitro es etiquetado como jugador pero se elimina este id y vuelve a ser árbitro, también se etiqueta a un punto del suelo como jugador por la mala calidad de imagen, incluso etiqueta a un jugador como árbitro y jugador a la vez, cosa que causa mucho problema *Figura 3.13.2*.

*Figura 3.13.2: Errores video 23*

Las funciones funcionan bien eliminando los id innecesarios, pero aún así no logra eliminar el último id, que podemos visualizar en la última imagen, ya que no logra a tener una trayectoria tan parecida como para eliminarlo. Las trayectorias se representan perfectamente con una buena clasificación, el único problema es que hay un id de más que es imposible de eliminar. Al final el modelo empeora porque hay un id que no logra eliminar, aunque el id de la imagen principal donde se hace una doble etiquetación del árbitro si que se consigue eliminar *Figura 3.13.3*.

*Figura 3.13.3 Representación 2D Video 23*

Aunque se trata de un vídeo con calidad limitada, el sistema consigue una detección y representación aceptables, aunque no totalmente limpias. Se confirma que las funciones tienen efecto, pero no eliminan todos los errores en condiciones adversas.

## Video 25

En este video vamos a trabajar con menor cantidad de jugadores, 10 jugadores más el portero. Como observamos la detección empieza mal ya que se detecta al árbitro y al portero como jugadores, cosa que no debería de ocurrir *Figura 3.14.1*.

*Figura 3.14.1: Primer frame Video 25*

Durante el proceso de detección el portero es identificado como jugador, y el árbitro va alternando su detección, cosa que nos va a causar un problema porque no se van a podemos eliminar los ids correspondientes a estos. Además se logra detectar al jugador faltante *Figura 3.14.2*.

*Figura 3.14.2: Errores video 25*

Las funciones eliminan logran eliminar 4 id que sobran pero sigue sin ser suficiente provocando que la representación en el campo de 2D no sea la correcta. Además passa de 11 jugadores a 13 jugadores, cosa que no debería ocurrir ya que el máximo de jugadores deberían ser 10 y no 14 como se indica en el frame con más detecciones *Figura 3.14.3*.

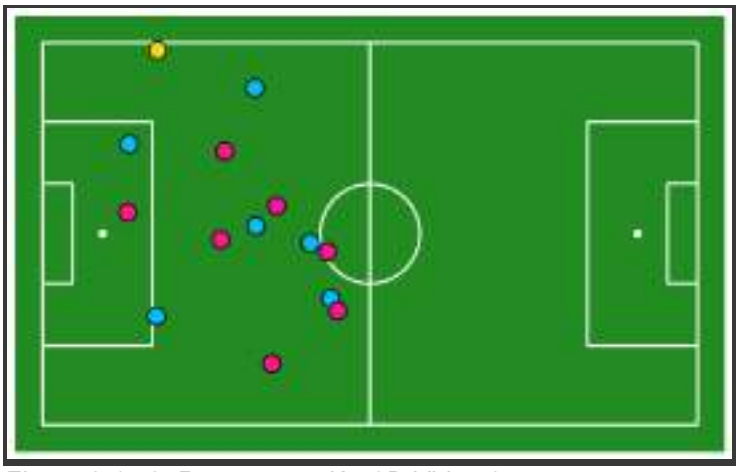


*Figura 3.14.3: Representación 2D Video 25*

Es un ejemplo donde, a pesar de trabajar con un número reducido de jugadores, la calidad visual y la mala clasificación impiden obtener un resultado fiable.
Este caso refleja que el sistema no solo es sensible a la cantidad de jugadores, sino también a la calidad de la imagen y al correcto etiquetado inicial. La suma de errores hace que el resultado final no sea válido para análisis táctico.

## 3.2. Discusión

Tras el análisis de los catorce vídeos procesados, se puede concluir que el sistema propuesto cumple con su objetivo principal en las condiciones adecuadas. La aplicación de funciones específicas para corregir, filtrar y mejorar las detecciones realizadas por el modelo base ha sido clave para transformar una detección inicial, en muchos casos incompleta o errónea, en una representación válida y útil para el análisis táctico. Estas funciones no solo corrigen errores comunes como trayectorias duplicadas o jugadores estáticos, sino que también permiten mantener la identidad de los jugadores a lo largo del tiempo, lo cual es esencial para generar información coherente y estable. En los vídeos donde la calidad es buena y el ángulo de grabación coincide con el entrenado por el modelo, como en los casos 1, 4, 5, 7, 10 y 11, se han obtenido resultados sólidos y consistentes, mostrando con claridad que el sistema puede alcanzar un nivel de precisión elevado cuando el entorno visual es favorable. Incluso en vídeos con detección parcial o errores iniciales, como el 3 o el 6, las funciones han ayudado a recuperar parte de la información o al menos estabilizar el conteo de jugadores en la visualización final.

Sin embargo, este proceso también ha puesto en evidencia varias limitaciones estructurales del modelo base de detección. Se ha comprobado que el modelo no es capaz de generalizar cuando se utilizan ángulos de cámara distintos a los previstos, como sucede en los vídeos 15 y 16, donde el enfoque desde detrás de la portería genera detecciones masivas incorrectas e impide realizar una proyección táctica válida. A esto se suma la pérdida de precisión cuando se trabaja con vídeos de baja calidad, donde aparecen falsos positivos, errores en la clasificación de roles como portero o árbitro, y trayectorias inestables que no siempre pueden ser corregidas. En casos como el vídeo 20 o el 25, la representación final se ve comprometida por estos factores.

Además, es importante señalar que el sistema depende completamente de que el modelo base detecte correctamente los roles presentes. Si un portero no es reconocido como tal desde el inicio, las funciones no pueden eliminarlo ni asignarlo a su equipo, lo cual afecta directamente la validez del análisis táctico. A pesar de ello, la respuesta del sistema ante los errores del modelo ha sido, en términos generales, satisfactoria. En múltiples casos se ha demostrado que, con un conjunto bien diseñado de funciones de mejora, es posible compensar parcialmente las limitaciones del modelo y obtener una representación más limpia, precisa y útil para el análisis posterior.

En definitiva, este trabajo demuestra que las funciones desarrolladas aportan un valor significativo al proceso de análisis automatizado, actuando como una capa de control y refinamiento que convierte una predicción bruta en información táctica útil. Aunque el sistema presenta limitaciones cuando se enfrenta a situaciones fuera del entorno entrenado, el enfoque implementado es válido y funcional. El sistema ha mostrado ser fiable en situaciones normales, y con mejoras futuras centradas en el entrenamiento de modelos más versátiles, una mejor detección de roles y el ajuste dinámico de parámetros, podría ampliarse su uso a contextos más variados y exigentes dentro del análisis deportivo automatizado.

| Vídeo | Detecciones iniciales | Detecciones finales | Diferencia | Observación |
|---|---|---|---|---|
| 1 | 19 | 20 | 1 | Obtiene más detecciones mejorando al original |
| 2 | 17 | 16 | –1 | Pérdida de IDs válidos por aglomeración |
| 3 | 17 | 19 | 2 | Obtiene más detecciones mejorando al original |
| 4 | 19 | 20 | 1 | Obtiene más detecciones mejorando al original |
| 5 | 20 | 20 | 0 | Detección inicial ya era completa, no estropea |
| 6 | 19 | 21 | 2 | Portero mal clasificado como jugador |
| 7 | 19 | 19 | 0 | Detección inicial ya era completa, no estropea |
| 10 | 20 | 20 | 0 | Detección inicial ya era completa, no estropea |
| 11 | 19 | 20 | 1 | Obtiene más detecciones mejorando al original |
| 15 | 24 | 23 | –1 | Mal funcionamiento por perspectiva diferente |
| 16 | 23 | 23 | 0 | Mal funcionamiento por perspectiva diferente |
| 20 | 21 | 22 | 1 | Mala calidad no detecta bien portero arbitro |
| 23 | 19 | 20 | 1 | Obtiene más detecciones mejorando al original |
| 25 | 11 | 13 | 2 | Mala calidad no detecta bien portero arbitro |

*Tabla 2*

De los 14 vídeos analizados *Tabla 2*, en 6 casos **(43%)** las funciones propuestas mejoraron claramente las detecciones, aumentando el número de jugadores identificados correctamente. En 3 vídeos **(21%)** no se aplicaron mejoras porque la detección inicial ya era completa. En los 5 vídeos restantes **(36%)**, no se observó mejora o incluso se introdujeron errores, debido principalmente a limitaciones del modelo base, baja calidad de imagen o ángulos no compatibles, estos errores son asumibles ya que estamos en búsqueda del límite del modelo. En promedio, las detecciones pasaron de **18,07** a **19,14** jugadores por vídeo, lo que representa una mejora media de **+1,0**7 jugadores. Estos resultados demuestran que el sistema mejora o mantiene la calidad de detección en un **64%** de los casos, aunque sigue siendo sensible a las condiciones del vídeo y la robustez del modelo

# Capítulo 4

# Conclusiones y proyección futura.

Este capítulo recoge las conclusiones generales del proyecto, evaluando el cumplimiento de los objetivos planteados y valorando el impacto de las soluciones implementadas sobre los resultados obtenidos. Además, se identifican las principales limitaciones encontradas durante el desarrollo, tanto técnicas como operativas, y se proponen líneas de trabajo futuro orientadas a mejorar el sistema. Entre ellas, se contemplan la integración de modelos más robustos, la mejora del entorno de ejecución, y la optimización de los procesos de clasificación y representación táctica en escenarios reales.

## 4.1. Conclusiones

A lo largo de este trabajo se ha desarrollado un sistema de detección y representación táctica automatizada en partidos de fútbol, utilizando un modelo de visión por computadora basado en YOLOv8 y complementado con funciones propias que mejoran la fiabilidad del proceso. El sistema ha sido evaluado sobre 14 vídeos que abarcan distintos contextos visuales, ángulos de grabación y calidades de imagen, lo que ha permitido valorar de forma amplia su funcionamiento. Se ha demostrado que el uso de funciones adicionales, como la eliminación de trayectorias estáticas, la fusión de IDs, la clasificación por equipo o el filtrado de detecciones erróneas, aporta un valor significativo. Estas funciones no solo corrigen los errores más frecuentes del modelo base, como la asignación incorrecta de roles o los falsos positivos por ruido visual, sino que también mantienen la coherencia en el seguimiento de jugadores a lo largo del tiempo. La representación final sobre un campo de fútbol en 2D permite obtener una visión táctica clara, mejorando la interpretabilidad respecto a una simple secuencia de fotogramas.

En contextos donde se dispone de vídeos con buena calidad y una cámara elevada en posición lateral ,condiciones similares a las que el modelo fue entrenado, el sistema ha mostrado un comportamiento muy sólido. En más de la mitad de los casos, las funciones han logrado mejorar cuantitativamente los resultados iniciales, detectando más jugadores válidos y corrigiendo errores previos sin introducir nuevas inconsistencias. Incluso en vídeos donde la detección inicial era ya buena, las funciones han sabido mantener la información válida sin sobrecorregir ni alterar la fiabilidad del sistema. Solo ha habido fallo cuando había una aglomeración de jugadores en una zona, que al estar el modelo entrenado con cajas delimitadoras provoca que algunos jugadores no se detecten correctamente al estar en la misma caja que otro.

No obstante, también se han puesto de manifiesto algunas limitaciones. Cuando el ángulo de grabación es diferente al previsto, como en las tomas desde detrás de la portería, o cuando la calidad de imagen es baja, el sistema pierde robustez. En estos casos, se generan múltiples errores en la identificación de jugadores, árbitros y porteros, las funciones no logran limpiar correctamente los datos, y la representación táctica resultante no es válida. Además, el sistema depende en gran medida de que el modelo base identifique correctamente los roles desde el inicio. Si un portero no es clasificado como tal, las funciones no pueden reetiquetarlo ni asignarlo a su equipo, lo que afecta directamente al análisis final.

En definitiva, el proyecto cumple con éxito su objetivo principal: automatizar la detección y visualización táctica de un partido de fútbol usando vídeos comunes, sin necesidad de equipamiento especializado. Las funciones desarrolladas han demostrado ser una capa de valor añadido que convierte una detección inicial básica en una herramienta más precisa y útil para el análisis táctico. Si bien existen limitaciones técnicas, especialmente en escenarios adversos, la base propuesta es sólida y funcional para su uso en contextos reales siempre que se respeten ciertas condiciones mínimas de calidad y grabación.

## 4.2. Trabajo futuro

De cara a futuras mejoras del sistema, hay varias líneas de trabajo que podrían explorarse para superar las limitaciones observadas. Una de las más relevantes sería entrenar un modelo de detección personalizado utilizando un conjunto de datos propio adaptado específicamente a las condiciones en las que se espera usar el sistema, incluyendo diferentes ángulos de cámara, condiciones lumínicas y contextos más variados. Esto permitiría aumentar la generalización y reducir la dependencia del ángulo y calidad del vídeo.

Otra línea de mejora sería integrar algoritmos de seguimiento más avanzados, como DeepSORT [2], o StrongSORT, que permitirían una mayor continuidad en la identificación de los jugadores y reducirían aún más los errores en la asignación de IDs. Asimismo, la clasificación de roles podría mejorarse mediante modelos auxiliares que analicen atributos contextuales o de movimiento, y no solo la información visual puntual de una detección.

Además, el uso de un modelo de segmentación en lugar de uno basado exclusivamente en cajas delimitadoras podría resolver uno de los problemas más frecuentes observados, la dificultad para separar correctamente a los jugadores en situaciones de aglomeración, como saques de esquina o faltas laterales. Al trabajar con máscaras precisas que segmentan el contorno real de los jugadores, se reducirían los errores de solapamiento que actualmente generan múltiples IDs para un mismo jugador o la detección de uno solo donde hay dos.

También sería interesante añadir una etapa de evaluación automática de la calidad de los datos de entrada, permitiendo al sistema adaptar sus parámetros según la resolución o el ángulo detectado. Esto abriría la posibilidad de aplicar el modelo en condiciones no ideales, aumentando su versatilidad.

Por último, uno de los principales factores limitantes durante el desarrollo de este trabajo ha sido el entorno de ejecución. El uso de Google Colab, aunque útil, impone restricciones en cuanto a tiempo de uso, recursos disponibles y almacenamiento en memoria, mientras que el equipo personal no contaba con la capacidad suficiente para procesar vídeos de forma intensiva. Para avanzar en las líneas de mejora propuestas y poder realizar un entrenamiento más personalizado, sería imprescindible contar con un entorno de desarrollo más potente y estable, que permita procesar datos en mayor volumen y entrenar modelos más complejos de forma eficiente.

En conjunto, el sistema desarrollado sienta una base prometedora para el análisis táctico automatizado en fútbol, y con mejoras centradas en la robustez del modelo, la gestión de diferentes condiciones de grabación y un entorno técnico más adecuado, podría convertirse en una herramienta aún más potente y versátil.

# Apéndice A

# Anexo

*Figura 5.1: pseudocódigo eliminar_trayectorias_estaticas*
Implementado en el archivo CodigoFinal [11], en la celda titulada Eliminar_trayectorias_estaticas.


*Figura 5.2: pseudocódigo marge_identical_trajectories*
Implementado en el archivo CodigoFinal [11], en la celda titulada marge_identical_trajectories.

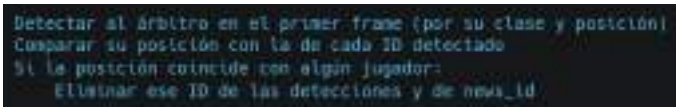



*Figura 5.3: pseudocódigo eliminar_arbitro*
Implementado en el archivo CodigoFinal [11], en la celda titulada Eliminar_arbitro.

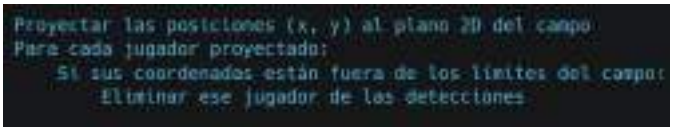



*Figura 5.4: pseudocódigo filtrar_jugadores_fuera_del_campo*
Implementado en el archivo CodigoFinal [11], en la celda titulada Filtrar_jugadores_fuera_del_campo.

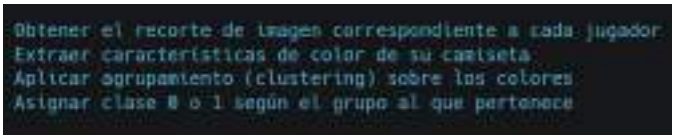



*Figura 5.5: pseudocódigo clasificacion_equipos*
Implementado en el archivo CodigoFinal [11], en la celda titulada Clasificacion_equipos.

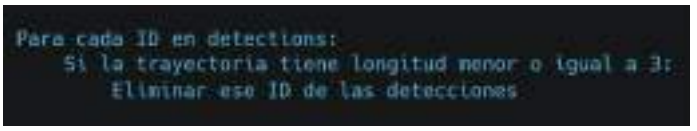
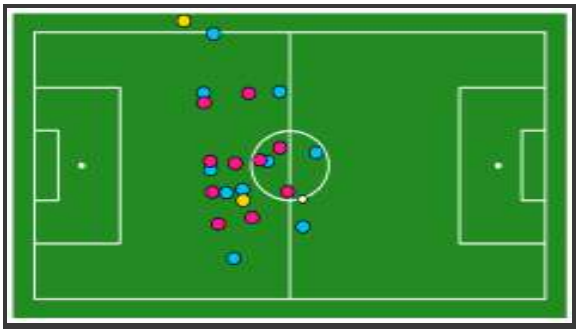
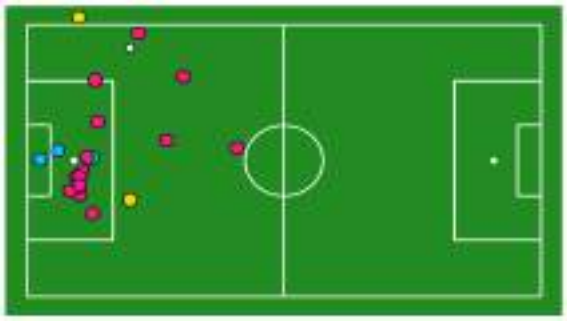
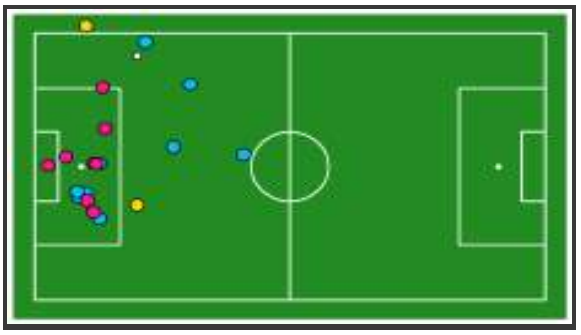
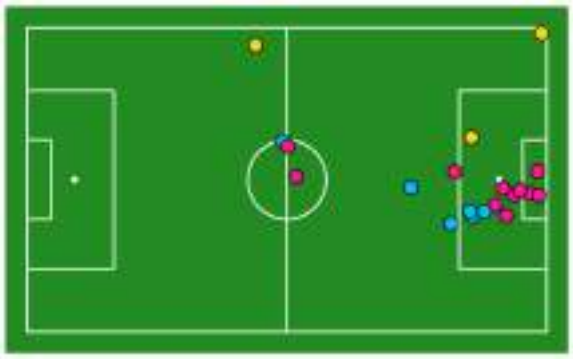
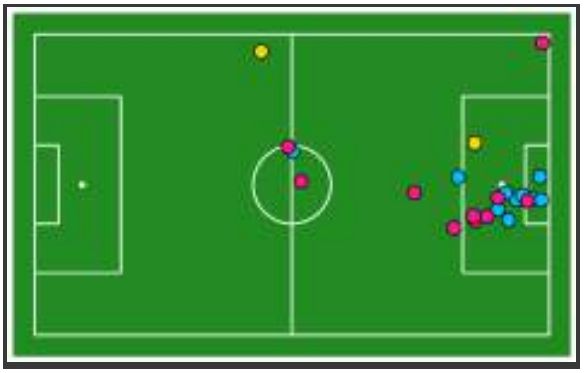
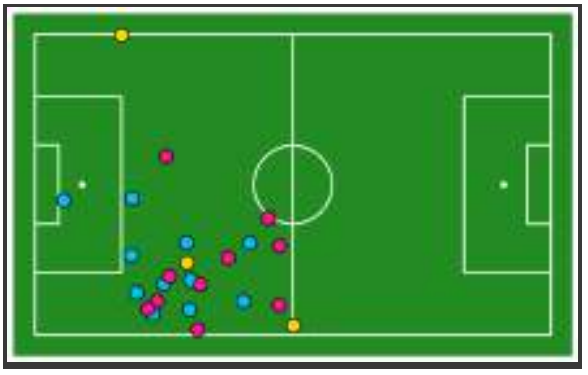
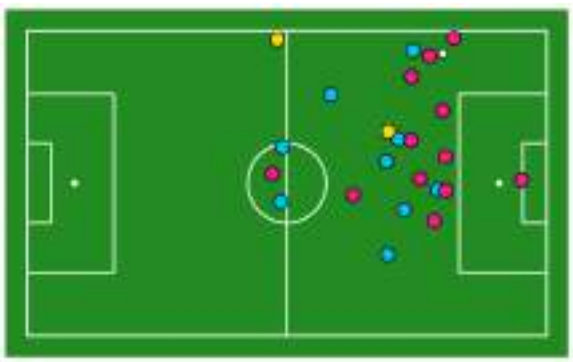
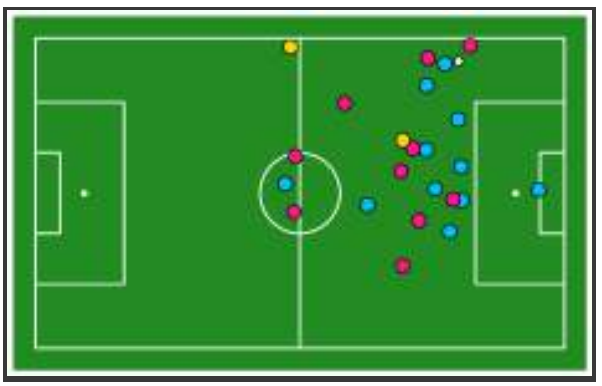


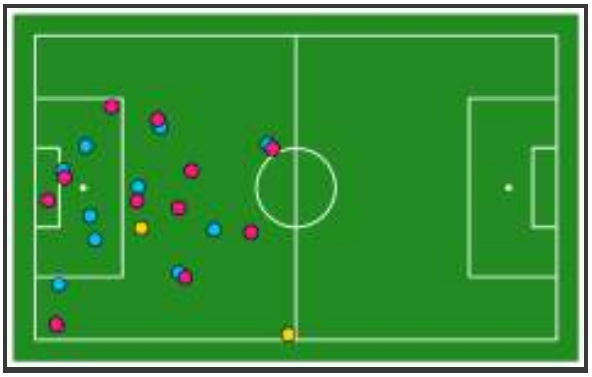
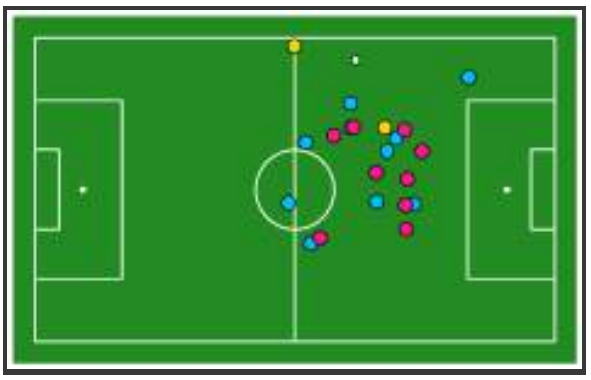
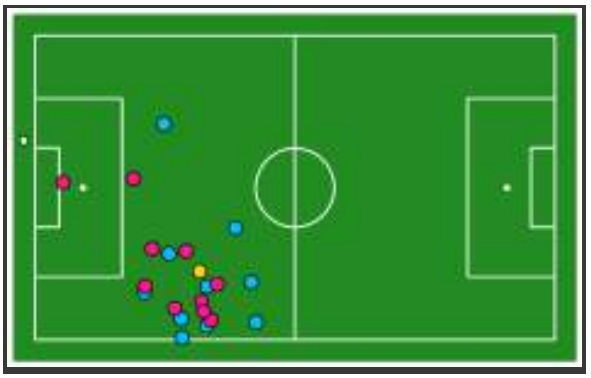
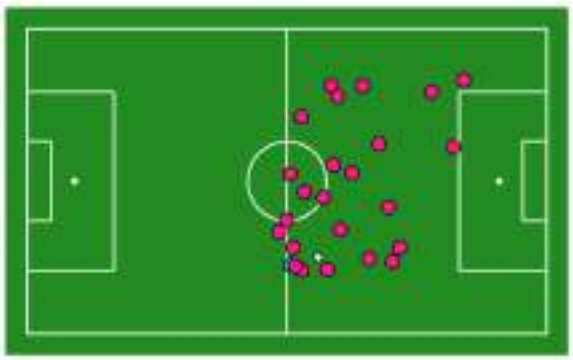

*Figura 5.6: pseudocódigo eliminar_ids_pocos_frames*
Implementado en el archivo CodigoFinal [11], en la celda titulada Eliminar_ids_pocos_frames.


*Figura 5.7: pseudocódigo merge_similar_trajectories*
Implementado en el archivo CodigoFinal [11], en la celda titulada Merge_similar_trajectories.


*Figura 5.8: pseudocódigo eliminar_portero*
Implementado en el archivo CodigoFinal [11], en la celda titulada Eliminar_portero.


*Figura 5.9: pseudocódigo resolve_goalkeepers_team_id*
Implementado en el archivo CodigoFinal [11], en la celda titulada Resolve_goalkeepers_team_id.

| Video | Primera detección(sin funciones) | Segunda detección(con funciones) |
|---|---|---|
| Video1 |  |  |
| Video2 |  |  |
| Video3 |  |  |
| Video4 |  |  |
| Video5 |  |  |

| | | |
|---|---|---|
| Video6 |  |  |
| Video7 |  |  |
| Video10 |  |  |
| Video11 |  |  |
| Video15 |  | No se calcula porque en la primera detección vemos que no se representa bien las ubicaciones de los jugadores en el campo. |

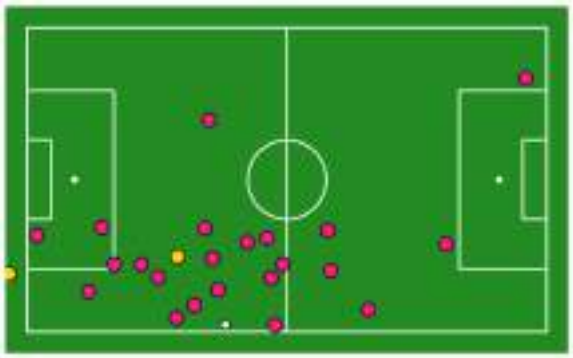
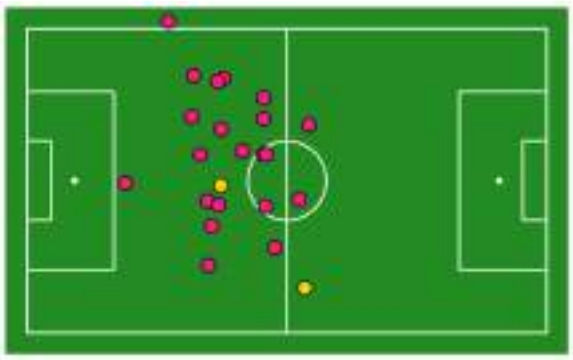
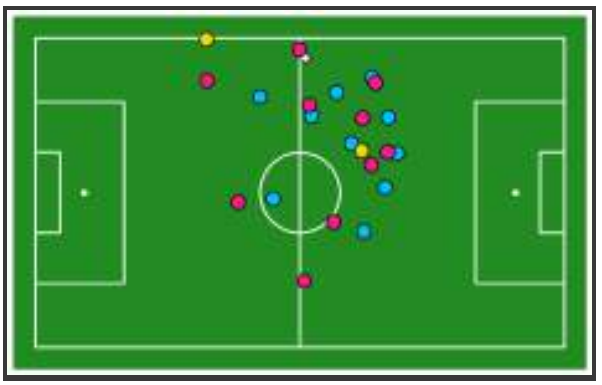
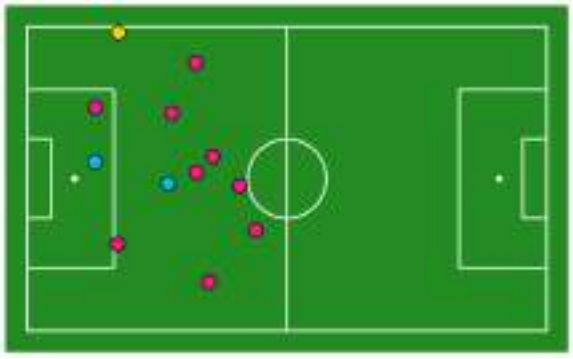
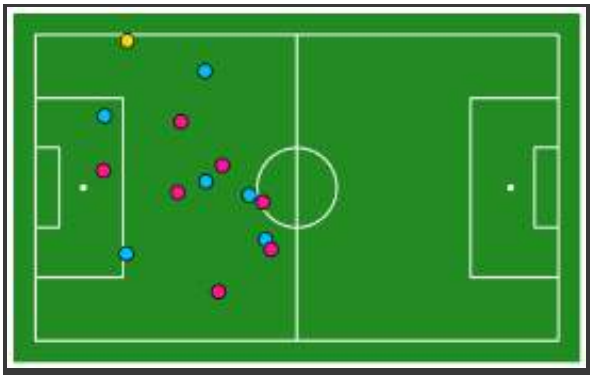
| | | |
|---|---|---|
| Video16 |  | No se calcula porque en la primera detección vemos que no se representa bien las ubicaciones de los jugadores en el campo. |
| Video20 |  |  |
| Video23 |  |  |
| Video25 |  |  |

*Tabla 3. Diferencia en la detección entre utilizar funciones o no.*

# Codigo Final

[11]CodigoFinal - Ruta con archivo zip donde se encuentra el codigo.

# Videos/Imagenes

[12]Videos - Ruta donde encontraremos los archivos Videos1.zip(video 1,2,3),Videos2.zip(video 4,5),Videos3.zip(video 6,7,10),Videos4.zip(video 11,15,16,20,23,25)
Imagenes

# Bibliografía


[1]Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). *YOLOv4: Optimal Speed and Accuracy of Object Detection*. arXiv preprint arXiv:2004.10934.

[2]Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016). *Simple Online and Realtime Tracking with a Deep Association Metric (DeepSORT)*. Proceedings of the IEEE International Conference on Image Processing (ICIP), 3464–3468.

[3]Hunter, J. D. (2007). *Matplotlib: A 2D Graphics Environment*. Computing in Science & Engineering, 9(3), 90–95.

[4]Jocher, G., et al. (2023). *Ultralytics YOLOv8*. GitHub repository: https://github.com/ultralytics/ultralytics

[5]OpenCV (2023). *Open Source Computer Vision Library*. Disponible en: https://opencv.org

[6]Roboflow (2023). *Custom Computer Vision Models and Datasets*. Disponible en: https://app.roboflow.com/david-c1ezj/football-players-detection-3zvbc-stcvd/models

[7]Second Spectrum (2023). *AI-Driven Sports Analytics Platform*. Disponible en: https://www.secondspectrum.com

[8]StatsBomb (2023). *Advanced Football Analytics*. Disponible en: https://statsbomb.com

[9]TRACAB (2023). *Real-time Sports Tracking and Data Systems*. Disponible en: https://tracab.com

[10]Van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). *The NumPy Array: A Structure for Efficient Numerical Computation*. Computing in Science & Engineering, 13(2), 22–30.

[13] Trabajo de Osman et al. (2024)

[14] Wyscout