

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Sistema de Análisis Automático de Vídeos de Pádel mediante Técnicas de Visión por Computador

*Automatic Video Analysis System for Padel Using Computer Vision
Techniques*

Sergio Pérez Lozano

La Laguna, 4 de Julio de 2025

Dña. María Elena Sánchez Nielsen, profesora Titular de Universidad adscrita al
Departamento de
Ingeniería informática y de Sistema de la Universidad de La Laguna, como tutora

C E R T I F I C A (N)

Que la presente memoria titulada:

“Sistema de Análisis Automático de Vídeos de Pádel mediante Técnicas de Visión por Computador”

ha sido realizada bajo su dirección por D. Sergio Pérez Lozano.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 4 de Julio de 2025.

Agradecimientos

Quisiera expresar mi más sincero agradecimiento a todas las personas que me han apoyado y acompañado a lo largo de la realización de este Trabajo de Fin de Grado.

En primer lugar, a mi tutora, Dña. María Elena Sánchez Nielsen, por su dedicación, orientación y valiosas sugerencias durante todas las fases del proyecto, que han sido fundamentales para su desarrollo y culminación.

A mis compañeros y amigos, por sus ánimos, consejos y por compartir conmigo tantas horas de trabajo y aprendizaje.

A mi familia, por su paciencia, comprensión y apoyo incondicional, no solo durante este proyecto, sino a lo largo de toda mi etapa universitaria.

Finalmente, a todas aquellas personas que, de una forma u otra, han contribuido a mi formación y crecimiento personal y profesional.

Sergio Pérez Lozano

Licencia



© Esta obra está bajo una licencia de **Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional (CC BY-NC-SA 4.0)**.

Resumen

Este Trabajo de Fin de Grado presenta el diseño e implementación de un sistema automático de análisis de vídeo para pádel utilizando técnicas de visión por computador y aprendizaje automático. El sistema detecta, sigue y visualiza los elementos clave de un partido de pádel, incluyendo jugadores, pelota, raquetas y pista, con el objetivo de extraer estadísticas tácticas y de rendimiento. La arquitectura sigue un enfoque modular, integrando métodos de última generación como YOLOv8 para la detección de jugadores y raquetas, un BallTrackerNet personalizado para el seguimiento de la pelota, y un mapeo de la pista basado en técnicas de visión por computador y homografía. Entre las funcionalidades adicionales se incluyen un minimapa en 2D para análisis táctico en tiempo real y estimación de la velocidad de los jugadores. El sistema ha sido probado con vídeos reales de partidos, logrando un rendimiento satisfactorio en detección y seguimiento a pesar de los desafíos como oclusiones, reflejos y movimientos a alta velocidad. Este trabajo contribuye al emergente campo del análisis del pádel al abordar la falta de conjuntos de datos y herramientas específicas, y abre nuevas vías para futuras investigaciones en análisis automático de deportes.

Palabras clave: Visión por computador, análisis de vídeo, detección y seguimiento, pádel, aprendizaje automático.

Abstract

This Final Degree Project presents the design and implementation of an automatic video analysis system for padel using computer vision and machine learning techniques. The system detects, tracks, and visualizes key elements of a padel match, including players, ball, rackets, and court, in order to extract tactical and performance-related statistics. The architecture follows a modular approach, integrating state-of-the-art methods such as YOLOv8 for player and racket detection, a customized BallTrackerNet for ball tracking, and homography-based court mapping. Additional features include a 2D minimap for real-time tactical analysis and speed estimation of players. The system has been tested on real match videos, achieving satisfactory detection and tracking performance despite challenges like occlusions, reflections, and high-speed movements. This work contributes to the emerging field of padel analytics by addressing the lack of dedicated datasets and tools, and opens avenues for further research in automatic sports analysis.

Keywords: Computer vision, video analysis, detection and tracking, padel, machine learning.

Índice general

1. Introducción	10
1.1. Motivación	10
1.2. Objetivos	11
1.2.1. Objetivo general	11
1.2.2. Objetivos específicos	11
2. Estado del arte	12
2.1. Visión por computador en deportes	12
2.1.1. Aplicaciones por deporte	12
2.1.2. Desafíos específicos del análisis deportivo	13
2.2. Detección de objetos en contextos deportivos	13
2.2.1. Evolución de las técnicas de detección	13
2.2.2. Detección específica de pelotas deportivas	14
2.3. Seguimiento multi-objeto (MOT)	14
2.3.1. Paradigmas de seguimiento	14
2.3.2. Métodos de asociación de datos	14
2.3.3. Algoritmos de seguimiento estado del arte	14
2.4. Análisis específico de deportes de raqueta	15
2.4.1. Características comunes	15
2.4.2. Sistemas existentes para tenis	15
2.5. Técnicas de aprendizaje profundo aplicadas	15
2.5.1. Arquitecturas de redes neuronales	15
2.6. Lagunas en el estado del arte	15
2.7. Oportunidades de investigación	16
3. Tecnologías utilizadas	17
3.1. Python y su ecosistema	17
3.2. YOLO (You Only Look Once)	17
3.2.1. Funcionamiento de YOLO	17
3.2.2. Ventajas de YOLO	18
3.3. TrackNet	19
3.4. OpenCV	19
3.5. Exploración de tecnologías alternativas	20
3.5.1. RetinaNet para detección de pelota	20
3.5.2. SAHI (Slicing Aided Hyper Inference)	20
3.5.3. Modelos preentrenados para raquetas	20

4. Implementación	22
4.1. Arquitectura general	22
4.1.1. Flujo de procesamiento	23
4.2. Componentes implementados	23
4.2.1. Keypoints de la cancha (<i>CourtLineDetector</i>)	23
4.2.2. Detección de la cancha (<i>CourtTracker</i>)	25
4.2.3. Detección de los jugadores (<i>PlayerTracker</i>)	27
4.2.4. Detección de la pelota (<i>PadelBallTracker</i>)	28
4.2.5. Detección de la raqueta - Desarrollo de modelo personalizado (<i>RacketTracker</i>)	30
4.3. Desarrollo e implementación del minimapa	31
4.3.1. Análisis de velocidad (<i>VelocityAnalyzer</i> y <i>BallVelocityAnalyzer</i>)	32
4.4. Optimizaciones implementadas	34
4.4.1. Sistema de caché	34
4.4.2. Gestión de recursos	35
5. Resultados y evaluación	36
5.1. Evaluación del rendimiento del sistema	36
5.1.1. Configuración experimental	36
5.1.2. Resultados de rendimiento	36
5.1.3. Análisis de resultados	37
5.1.4. Rendimiento global del sistema	38
5.2. Métricas de precisión	39
5.3. Observaciones y ajustes durante el desarrollo	39
5.3.1. Desafíos en la detección de pelota	39
5.3.2. Mejoras en la re-identificación de jugadores	40
6. Conclusiones y líneas futuras	41
6.1. Conclusiones	41
6.2. Trabajo futuro	42
6.2.1. Mejoras a corto plazo	42
6.2.2. Extensiones a largo plazo	42
7. Summary and Conclusions	43
7.1. Summary	43
7.2. Conclusions	43
8. Presupuesto	45
8.1. Resumen de costes	45
Bibliografía	45

Índice de figuras

4.1.	Diagrama general de la arquitectura del sistema y sus clases.	23
4.2.	Visualización de los puntos clave detectados en una imagen de la pista.	24
4.3.	Diagrama de clases del módulo <code>CourtLineDetector</code>	24
4.4.	Ejemplo de detección de la pista y estimación de homografía con el módulo <code>CourtTracker</code>	25
4.5.	Diagrama de clases del módulo <code>CourtTracker</code>	25
4.6.	Ejemplo de detección y seguimiento de jugadores en un partido de pádel (arriba) y diagrama del proceso implementado en <code>PlayerTracker</code> (abajo).	28
4.7.	Detección de la pelota de pádel en un fotograma real (arriba) y diagrama del proceso implementado en <code>BallTrackerNet</code> (abajo).	29
4.8.	Ejemplo de imagen etiquetada manualmente con <code>LabelImg</code> para la detección de raquetas (arriba) y diagrama del proceso implementado en <code>RacketTracker</code> (abajo).	30
4.9.	Minimap generado por el sistema (arriba) y diagrama de la clase <code>MinimapVisualizer</code> (abajo), mostrando las trayectorias y posiciones actuales de los jugadores y la pelota sobre la cancha.	32
4.10.	Panel de análisis de velocidad y distancia recorrida de los jugadores (arriba) y diagramas de las clases <code>VelocityAnalyzer</code> y <code>BallVelocityAnalyzer</code> (centro y abajo).	34

Índice de cuadros

5.1. Tiempos de ejecución por componente para vídeo de 7 segundos.	37
5.2. Tiempos de ejecución por componente para vídeo de 52 segundos.	37
5.3. Resultados de precisión en los dos vídeos analizados	39
8.1. Resumen del presupuesto estimado del proyecto.	45

Capítulo 1

Introducción

1.1. Motivación

Es ampliamente reconocido que la tecnología ha transformado de manera significativa el mundo del deporte, impactando no solo en la forma en que se practican las disciplinas deportivas, sino también en su gestión, análisis y regulación. Desde el uso de dispositivos de seguimiento en tiempo real hasta sistemas de arbitraje asistido por video, la tecnología se ha convertido en una herramienta fundamental para mejorar el rendimiento, aumentar la equidad y enriquecer la experiencia tanto de atletas como de espectadores.

Uno de los sectores más beneficiados por los avances tecnológicos ha sido el arbitraje. Herramientas como el VAR (Video Assistant Referee) en el fútbol, el ojo de halcón en el tenis, o los sensores de línea en el voleibol han reducido significativamente los errores arbitrales. Estas tecnologías permiten a los jueces tomar decisiones más precisas y justas, lo que fortalece la credibilidad de las competiciones y mejora la percepción del público.

La tecnología también ha revolucionado la manera en que los entrenadores y analistas diseñan estrategias de juego. El uso de software de análisis de datos, inteligencia artificial y simulaciones permite estudiar patrones de juego, debilidades del adversario y el rendimiento del propio equipo.

También el rendimiento de los deportistas ha alcanzado nuevos niveles gracias a la integración de tecnologías como sensores biométricos, wearables, plataformas de análisis biomecánico y cámaras de alta velocidad. Estas herramientas permiten monitorear parámetros fisiológicos en tiempo real, corregir posturas, prevenir lesiones y diseñar programas de entrenamiento personalizados. El resultado es un atleta más eficiente, saludable y competitivo.

El pádel es uno de los deportes de raqueta que más crecimiento ha experimentado en los últimos años, especialmente en España y América Latina. Este crecimiento ha generado una demanda creciente de herramientas tecnológicas avanzadas para el análisis deportivo. En este ámbito, la incorporación de tecnologías como la inteligencia artificial (IA) y la visión por computador representa una oportunidad transformadora. Estas herramientas pueden aportar mejoras significativas tanto en la práctica profesional como en el entrenamiento amateur. Una de las principales ventajas es la optimización del arbitraje. Mediante cámaras inteligentes y algoritmos de visión por computador, es posible detectar automáticamente si una pelota ha tocado la pared, salido del campo o si ha sido válida, reduciendo el margen de error humano. Esto no solo aporta justicia a las

competiciones, sino que también aumenta la transparencia de las decisiones arbitrales.

En el plano táctico y técnico, la inteligencia artificial permite analizar patrones de juego, posicionamientos y zonas de impacto con una precisión inalcanzable por métodos tradicionales. Entrenadores y jugadores pueden acceder a estadísticas sobre golpes, desplazamientos y jugadas efectivas, lo que permite diseñar estrategias personalizadas y detectar puntos débiles tanto propios como del oponente.

Asimismo, con el uso de sensores y cámaras conectadas a sistemas de IA, se pueden automatizar correcciones técnicas, evaluar la biomecánica del golpe y prevenir lesiones gracias a una mejor distribución del esfuerzo físico. Esto no solo mejora el rendimiento, sino que extiende la vida útil del deportista.

1.2. Objetivos

1.2.1. Objetivo general

En este contexto, el principal objetivo que se plantea es el desarrollo de un sistema automático de análisis de vídeos de pádel que permita la detección, seguimiento y visualización de elementos clave del juego mediante técnicas de visión por computador y aprendizaje automático que permita realizar análisis posteriores del rendimiento de los jugadores así como la elaboración de estadísticas del juego.

1.2.2. Objetivos específicos

Para conseguir este objetivo, es necesario abordar una serie de tareas más específicas como son:

- Implementar un sistema de detección automática de la pista de pádel mediante algoritmos de visión por ordenador.
- Realizar transformaciones geométricas que permitan referenciar la posición de los diferentes elementos en la pista.
- Desarrollar algoritmos de detección y seguimiento de jugadores utilizando técnicas de seguimiento multi-objeto específico para pádel.
- Implementar un sistema especializado de detección de pelota que considere las características específicas de las pelotas de pádel así como el cálculo de trayectorias.
- Implementar una estrategia para la detección de las raquetas de pádel.
- Implementar detección de raquetas de pádel y análisis de tipos de golpe.
- Desarrollar un sistema de visualización 2D mediante minimapa que permita ubicar en un mapa la posición de los diferentes jugadores y de la pelota durante el transcurso de un partido.
- Integrar todos los componentes en una aplicación funcional.

Capítulo 2

Estado del arte

2.1. Visión por computador en deportes

El análisis automático de videos deportivos ha sido un área de investigación activa durante las últimas décadas, evolucionando desde sistemas básicos de seguimiento hasta complejas plataformas de análisis táctico. Los primeros sistemas, desarrollados en los años 90, se centraban principalmente en la detección simple de objetos mediante técnicas de sustracción de fondo y filtros de color. La rápida evolución del campo puede dividirse en tres generaciones principales. La primera generación (1990-2005) incluye sistemas basados en técnicas clásicas de procesamiento de imágenes, utilizando principalmente filtros lineales, detección de bordes y métodos de segmentación básicos. Estos sistemas tenían limitaciones significativas en robustez y precisión [1]. La segunda generación (2005-2015) introduce técnicas de aprendizaje automático tradicional, como SVM (Support Vector Machines) y Random Forest, combinadas con descriptores de características manuales (HOG, SIFT, SURF) [2]. Esta generación hizo mejorar significativamente la precisión pero requería ingeniería de características intensiva. La tercera generación (2015-presente) representa la revolución del deep learning, particularmente con redes neuronales convolucionales (CNN), que ha transformado completamente el campo al permitir el aprendizaje automático de características y lograr precisiones antes inalcanzables [3].

2.1.1. Aplicaciones por deporte

En fútbol, que es el deporte más estudiado en visión por computador debido a su popularidad global y la disponibilidad de datos, sistemas como ChyronHego y Opta Sports han establecido estándares industriales para el análisis automático [4]. Las principales tareas incluyen seguimiento de jugadores y balón, detección de eventos como goles y faltas, análisis táctico de formaciones, y cálculo de métricas físicas como distancia recorrida y velocidad. En baloncesto, los sistemas de análisis se centran en seguimiento de jugadores en espacios reducidos, análisis de tiros y porcentajes de acierto, detección de estrategias ofensivas y defensivas, y medición de espacios y posicionamiento [5]. En tenis, más relevante para nuestro trabajo, las aplicaciones incluyen seguimiento de pelota de alta velocidad, detección de líneas de la cancha, análisis de golpes y técnica, y estadísticas de juego como aces y dobles faltas [6].

2.1.2. Desafíos específicos del análisis deportivo

El análisis automático de vídeos deportivos presenta desafíos únicos que no se encuentran en otras aplicaciones de visión por computador. El movimiento rápido de objetos, donde las pelotas deportivas pueden alcanzar velocidades extremas, requiere algoritmos especializados para su detección y seguimiento. El motion blur y la baja resolución temporal de las cámaras estándar complican esta tarea [7]. Las occlusiones frecuentes ocurren cuando los jugadores constantemente se ocultan entre sí, especialmente en deportes de equipo, lo que requiere técnicas sólidas de re-identificación y predicción de trayectorias [8]. El tamaño variable de objetos, desde pelotas pequeñas hasta jugadores completos, requiere arquitecturas multi-escala para su detección efectiva. Las condiciones ambientales variables incluyen iluminación cambiante entre interior y exterior, condiciones meteorológicas como lluvia y viento, calidad variable de vídeo en términos de resolución y frame rate, y ángulos de cámara diversos que complican la calibración y el análisis [9].

2.2. Detección de objetos en contextos deportivos

2.2.1. Evolución de las técnicas de detección

La complejidad de la tarea de la detección de objetos en imágenes de vídeo ha requerido el uso de una gran cantidad de estrategias para generar aproximaciones eficientes. Los métodos clásicos utilizaban técnicas de procesamiento de imágenes tradicionales como sustracción de fondo mediante modelado del fondo estático para detectar objetos en movimiento, donde algoritmos como Gaussian Mixture Models (GMM) y MOG2 fueron ampliamente utilizados [10]. La detección basada en color aprovechaba los colores distintivos de objetos deportivos como pelotas amarillas de tenis o camisetas de equipos. La detección de contornos utilizaba operadores como Canny y filtros de Sobel para detectar bordes de objetos. Los métodos de aprendizaje automático tradicional incluyen Viola-Jones adaptado para detectar objetos deportivos utilizando características Haar [11], HOG + SVM que combina Histograms of Oriented Gradients con Support Vector Machines para detección de personas [2], y Deformable Part Models (DPM) para modelado de objetos como colecciones de partes deformables [12].

Técnicas más modernas como el deep learning hace uso de redes neuronales profundas con arquitecturas de una etapa como YOLO (You Only Look Once), una familia de detectores que tratan la detección como un problema de regresión donde YOLOv8 y YOLOv11 son actualmente estado del arte en velocidad [13]. SSD (Single Shot MultiBox Detector) utiliza múltiples escalas de características para detectar objetos de diferentes tamaños [14]. RetinaNet introduce la función de pérdida Focal Loss para manejar el desbalance de clases [15]. Las arquitecturas de dos etapas incluyen la familia R-CNN desde R-CNN original hasta Faster R-CNN, estableciendo el paradigma de propuesta de regiones más clasificación [16, 17]. Mask R-CNN extiende Faster R-CNN para segmentación de instancias [18]. Cascade R-CNN mejora la precisión mediante múltiples etapas de refinamiento [19].

2.2.2. Detección específica de pelotas deportivas

La detección de pelotas presenta desafíos únicos debido a su tamaño pequeño, movimiento rápido y apariencia que puede cambiar según las condiciones de iluminación. Las características distintivas de pelotas incluyen forma circular o esférica, colores específicos como amarillo en tenis o blanco en pádel, patrones de textura únicos, y comportamiento de movimiento predecible [20]. Las técnicas especializadas incluyen filtros de forma circular utilizando transformada de Hough circular para detectar formas circulares en la imagen, filtrado por color en espacios de color específicos convirtiendo a espacios HSV o LAB para mejor separación de colores, redes neuronales especializadas entrenando modelos específicamente para pelotas deportivas con datasets aumentados, y seguimiento predictivo utilizando filtros de Kalman o filtros de partículas para predecir posiciones futuras basándose en física del movimiento [21].

2.3. Seguimiento multi-objeto (MOT)

El seguimiento de múltiples objetos es fundamental para el análisis deportivo, especialmente en deportes de equipo donde es necesario mantener la identidad de cada jugador a lo largo del tiempo. Diferentes aproximaciones se han desarrollado para abordar este problema complejo, entre las que se destacan las siguientes.

2.3.1. Paradigmas de seguimiento

El paradigma Tracking-by-Detection es dominante actualmente y separa el problema en dos etapas: detección para localizar objetos en cada frame, y asociación para conectar detecciones entre frames y formar trayectorias [22]. El paradigma Tracking-by-Regression incluye enfoques end-to-end que aprenden directamente las trayectorias sin etapa de detección explícita.

2.3.2. Métodos de asociación de datos

El Hungarian Algorithm es un algoritmo de asignación óptima utilizado para resolver el problema de asociación entre detecciones y tracks existentes [23]. Los filtros de Kalman permiten predicción del estado futuro de objetos basándose en modelos de movimiento, siendo especialmente útiles para objetos con movimiento suave y predecible [24]. Las métricas de similitud incluyen distancia euclidea que es simple pero efectiva para objetos con movimiento suave, Intersección sobre Unión (IoU) como medida de solapamiento entre cajas delimitadoras, y características visuales como histogramas de color, descriptores de textura, y embeddings de deep learning para re-identificación fiable [25].

2.3.3. Algoritmos de seguimiento estado del arte

SORT (Simple Online and Realtime Tracking) es un algoritmo básico pero efectivo que combina filtros de Kalman con Hungarian algorithm [26]. DeepSORT es una extensión de SORT que incorpora características visuales profundas para mejorar la re-identificación [25]. ByteTrack es un algoritmo reciente que utiliza detecciones de baja confianza para mejorar el recall en seguimiento [27]. StrongSORT mejora DeepSORT con técnicas de interpolación y re-identificación más estable [28].

2.4. Análisis específico de deportes de raqueta

2.4.1. Características comunes

Los deportes de raqueta comparten varias características que los hacen adecuados para análisis automático. El entorno controlado con pistas de dimensiones estándar y marcas claramente definidas facilita la calibración y el mapeo de coordenadas. Los objetos distintivos como raquetas, pelotas y vestimenta de jugadores son visualmente distintivos y facilitan su detección automática. Los patrones de movimiento siguen patrones biomecánicos predecibles que pueden ser modelados matemáticamente. Los eventos discretos permiten descomponer el juego en eventos discretos como saques, golpes y puntos [29].

2.4.2. Sistemas existentes para tenis

Hawk-Eye es el sistema comercial líder utilizado en torneos profesionales que utiliza múltiples cámaras de alta velocidad (hasta 340 fps), reconstrucción 3D de trayectorias de pelota, precisión de $\pm 3.6\text{mm}$ demostrada, pero con costo prohibitivo para aplicaciones amateur [30]. TennisBot es un sistema de código abierto para análisis amateur que incluye detección de jugadores y pelota usando YOLO, seguimiento básico con OpenCV, análisis estadístico simple, pero limitado a vistas fijas de cámara [31]. PlaySight es un sistema comercial para academias que ofrece análisis automático de técnica de golpeo, estadísticas en tiempo real, múltiples cámaras sincronizadas, e interfaz web para revisión posterior.

2.5. Técnicas de aprendizaje profundo aplicadas

2.5.1. Arquitecturas de redes neuronales

Las Redes Neuronales Convolucionales (CNN) forman la base de la mayoría de sistemas modernos de visión por computador deportiva [3]. Las arquitecturas comunes incluyen ResNet como redes residuales que permiten entrenar redes muy profundas [32], EfficientNet que optimiza el balance entre precisión y eficiencia computacional [33], y Vision Transformer (ViT) que aplica mecanismos de atención a visión por computador [34]. Las Redes Neuronales Recurrentes (RNN) se utilizan para modelar secuencias temporales en el análisis de movimiento, incluyendo LSTM (Long Short-Term Memory) para capturar dependencias a largo plazo [35], GRU (Gated Recurrent Units) como alternativa más simple a LSTM [36], y Transformer con mecanismos de atención para secuencias [37].

2.6. Lagunas en el estado del arte

A pesar de los avances significativos en visión por computador deportiva, existen varias lagunas específicas para el análisis de pádel. La falta de sistemas específicos para pádel es evidente ya que la mayoría de sistemas existentes están diseñados para tenis o fútbol, y no existen sistemas comerciales o académicos específicamente diseñados para las características únicas del pádel. Los datasets limitados representan otra laguna importante, ya que no existen datasets públicos anotados específicamente para pádel,

lo que limita el desarrollo y evaluación de algoritmos. El modelado de interacciones con paredes es insuficiente en los sistemas existentes que no modelan adecuadamente las interacciones de la pelota con las paredes, fundamental en pádel. Finalmente, el análisis táctico específico carece de métricas y análisis tácticos desarrollados específicamente para pádel, ya que los existentes para otros deportes no se aplican directamente debido a las características únicas del pádel.

2.7. Oportunidades de investigación

El análisis de las lagunas identificadas revela varias oportunidades de investigación importantes. El desarrollo de algoritmos específicos para pádel que consideren las interacciones con paredes y el juego en espacio confinado representa una oportunidad significativa. La creación de datasets anotados que permitan el desarrollo y evaluación de algoritmos específicos es fundamental para el avance del campo. El modelado físico de trayectorias que incorpore las características de las pelotas de pádel y las interacciones con diferentes superficies requiere investigación específica. Las métricas tácticas específicas que capturen las estrategias únicas del pádel necesitan ser desarrolladas. Los sistemas de tiempo real optimizados para las características computacionales de análisis de pádel representan otra área de oportunidad. Este trabajo de fin de grado se posiciona para abordar varias de estas oportunidades, contribuyendo significativamente al campo del análisis automático de deportes.

Capítulo 3

Tecnologías utilizadas

3.1. Python y su ecosistema

El desarrollo del sistema se ha realizado íntegramente en **Python**, aprovechando su amplia disponibilidad de librerías para visión por computador, aprendizaje profundo y procesamiento de vídeo. Este lenguaje ha permitido una rápida experimentación y desarrollo gracias a su sencillez y a herramientas como `OpenCV`, `PyTorch`, `NumPy` y `Matplotlib`, entre otras.

3.2. YOLO (You Only Look Once)

La detección de objetos es una tarea fundamental en visión por computador que consiste en localizar y clasificar instancias de objetos dentro de una imagen. Antes del desarrollo de arquitecturas modernas como YOLO, los métodos tradicionales para detección de objetos se basaban en el uso de clasificadores sobre ventanas deslizantes (*sliding window*). Este enfoque consistía en recorrer la imagen con una ventana de tamaño fijo (o varios tamaños) y aplicar en cada posición un clasificador binario, como un SVM o un clasificador de cascada, para determinar si esa región contenía el objeto de interés. Posteriormente se repetía el proceso a diferentes escalas para detectar objetos de distintos tamaños. Este método era computacionalmente costoso y lento, ya que requería evaluar miles o millones de ventanas por imagen y carecía de una integración eficiente entre localización y clasificación.

Para superar estas limitaciones, surgieron métodos basados en redes neuronales profundas que unifican ambas tareas. Uno de los más influyentes y eficientes es YOLO (*You Only Look Once*), una arquitectura de red neuronal convolucional diseñada específicamente para detección de objetos en tiempo real.

A diferencia de los métodos tradicionales, YOLO trata la detección como un único problema de regresión que predice directamente las coordenadas de las cajas delimitadoras y las probabilidades de las clases para toda la imagen en una sola pasada por la red. Esto permite que sea extremadamente rápido y adecuado para aplicaciones en tiempo real.

3.2.1. Funcionamiento de YOLO

El funcionamiento de YOLO se puede resumir en los siguientes pasos principales:

1. La imagen de entrada se divide en una cuadrícula de $S \times S$ celdas.
2. Cada celda es responsable de predecir un número fijo de cajas delimitadoras (bounding boxes) y las probabilidades de que un objeto de cada clase esté presente en esa celda.
3. Por cada caja, se predicen cuatro valores que representan las coordenadas normalizadas del centro de la caja (x, y) y su ancho y alto (w, h), junto con una puntuación de confianza que indica la certeza de que la caja contiene un objeto y qué tan bien se ajusta a él.
4. Finalmente, las predicciones de todas las celdas se combinan, y se aplica un proceso de supresión de no máximos (*Non-Maximum Suppression, NMS*) para eliminar las cajas redundantes y mantener sólo las más probables.

Este enfoque permite a YOLO lograr una alta velocidad de procesamiento, siendo capaz de procesar decenas de imágenes por segundo, lo que lo hace ideal para aplicaciones que requieren detección en tiempo real.

3.2.2. Ventajas de YOLO

Las principales ventajas de YOLO frente a los métodos anteriores y otras arquitecturas de detección son:

- **Velocidad:** dado que la detección se realiza en una única pasada por la red, es significativamente más rápido que los métodos basados en regiones como R-CNN.
- **Precisión:** a pesar de su rapidez, logra una buena precisión tanto en la localización como en la clasificación de objetos.
- **Facilidad de entrenamiento:** su arquitectura compacta y modular permite entrenar el modelo con datos personalizados de manera eficiente.
- **Predicciones globales:** considera toda la imagen en el momento de la predicción, lo que le permite evitar predicciones incoherentes y mejorar la detección de contextos complejos.
- **Disponibilidad de pesos preentrenados:** existen modelos ya entrenados sobre conjuntos de datos extensos que incluyen numerosas clases de objetos, como la persona, lo que permite utilizar estos pesos directamente para tareas comunes sin necesidad de entrenar desde cero. En este trabajo se han utilizado precisamente los pesos preentrenados de la red para detectar a los jugadores como instancias de la clase *person*.

Actualmente, existen múltiples versiones de YOLO, desde su primera versión hasta las más recientes, como la v8 utilizada en este proyecto. Cada nueva versión introduce mejoras en velocidad, precisión, capacidad de generalización y facilidad de integración.

3.3. TrackNet

TrackNet es una arquitectura específicamente diseñada para el seguimiento de pelotas en deportes de raqueta. Sus características principales son:

- Predicción de trayectorias de la pelota
- Manejo de occlusiones temporales
- Adaptación a diferentes deportes de raqueta

TrackNet funciona como un modelo de detección y seguimiento que toma secuencias de imágenes consecutivas como entrada y estima la posición de la pelota en cada fotograma. Gracias al uso de memoria temporal, el modelo es capaz de mantener la coherencia en la trayectoria estimada incluso en casos de ocultación parcial o total. Esta capacidad le permite no solo detectar la posición actual de la pelota, sino también predecir hacia dónde se moverá en los próximos instantes, lo que resulta especialmente útil en escenarios dinámicos y de alta velocidad.

Además del modelo basado en redes neuronales, TrackNet emplea técnicas clásicas de visión por computador como apoyo en el preprocesamiento y postprocesamiento. Por ejemplo, en los pasos previos se utiliza la transformada de Hough para la detección de objetos circulares en las máscaras generadas, lo que ayuda a localizar candidatas a pelota con mayor precisión. También se aplica una máscara binaria para destacar las regiones de interés y facilitar la identificación de la pelota en entornos con mucho ruido visual. Asimismo, el sistema incorpora estrategias para controlar el crecimiento aparente de la pelota (debido a cambios de perspectiva o desenfoque) y mantener una estimación consistente de su tamaño durante la trayectoria.

En este proyecto, TrackNet se ha configurado para recibir como entrada secuencias de tres fotogramas consecutivos, de modo que el modelo pueda explotar la información temporal entre ellos para estimar la posición actual y prever el movimiento futuro de la pelota. Esta configuración permite mejorar la fiabilidad del sistema frente a occlusiones y movimientos rápidos, adaptándose a las condiciones particulares del pádel.

3.4. OpenCV

OpenCV (Open Source Computer Vision Library) proporciona las herramientas fundamentales para:

- Procesamiento de imágenes y vídeo
- Transformaciones geométricas (homografías)
- Operaciones de dibujo y visualización

Además, OpenCV fue de vital importancia para la correcta detección de la homografía, ya que permitió aplicar técnicas clásicas de visión por computador como segmentación, cálculo de contornos y generación de máscaras para aislar la pista del fondo. Estas operaciones fueron fundamentales para identificar de forma fiable los bordes y las intersecciones que definen la geometría de la cancha, especialmente en condiciones de iluminación y perspectivas variadas. La combinación de estas técnicas con la detección automática de puntos clave permitió estimar la homografía con mayor consistencia y precisión.

3.5. Exploración de tecnologías alternativas

Durante el desarrollo del proyecto, se evaluaron diferentes enfoques y tecnologías para optimizar la detección de objetos específicos del pádel.

3.5.1. RetinaNet para detección de pelota

Durante el desarrollo del sistema se llevó a cabo una experimentación con RetinaNet, una arquitectura de detección de objetos basada en redes neuronales convolucionales que utiliza la *Focal Loss* para mitigar el desequilibrio entre clases dominantes y minoritarias. Esta red ha demostrado ser eficaz en escenarios con objetos pequeños y poco representados. Sin embargo, al aplicar RetinaNet al caso específico de detección de pelotas de pádel, los resultados fueron insatisfactorios. Las principales limitaciones observadas fueron la baja precisión en la detección de objetos de tamaño reducido como la pelota, especialmente en condiciones de movimiento rápido y fondo complejo. También debido a inconsistencia temporal, ya que el modelo no incorporaba información de cuadros anteriores, lo que generaba interrupciones en la trayectoria detectada y requería de mayor carga computacional, lo cual dificultaba su integración en un sistema de procesamiento en tiempo casi real. Por estas razones, RetinaNet fue descartado en favor de soluciones más adecuadas como TrackNet, que sí está optimizado para escenarios dinámicos y objetos de trayectoria continua como la pelota en deportes de raqueta.

3.5.2. SAHI (Slicing Aided Hyper Inference)

Se experimentó con SAHI aplicado a YOLO para mejorar la detección de objetos pequeños, como la pelota y las raquetas de pádel. Esta técnica consiste en dividir las imágenes en secciones más pequeñas para que el detector procese cada fragmento por separado, aumentando así la capacidad para localizar objetos diminutos con mayor precisión.

Inicialmente, se probó su aplicación para la detección de la pelota. Sin embargo, los resultados no justificaron su uso: el tiempo de procesamiento se incrementó drásticamente y los resultados fueron incluso peores que los obtenidos con el método original, sin mejoras apreciables en la detección. Por estos motivos, se descartó su uso para la pelota.

En el caso de las raquetas, dado que se trata de objetos igualmente pequeños y con una gran variabilidad en las imágenes, sí se decidió mantener la técnica SAHI para intentar mejorar la tasa de detección. No obstante, los resultados siguieron siendo pobres y el impacto negativo en los tiempos de ejecución fue considerable, convirtiéndose en el principal cuello de botella del sistema. A pesar de ello, se optó por mantenerla para las raquetas a modo experimental, dado que sin SAHI las detecciones eran prácticamente inexistentes.

3.5.3. Modelos preentrenados para raquetas

Se evaluaron modelos preentrenados para detección de raquetas de tenis disponibles públicamente. Sin embargo, estas soluciones no fueron efectivas para raquetas de pádel debido a las diferencias morfológicas significativas:

- Las raquetas de pádel no tienen cuerdas, sino una superficie sólida perforada

- Forma y proporciones diferentes a las raquetas de tenis
- Contexto de uso y posiciones de agarre distintas

Capítulo 4

Implementación

4.1. Arquitectura general

El sistema desarrollado sigue una arquitectura modular compuesta por los siguientes componentes principales:

1. **Módulo de Detección de Pista:** Identifica los límites de la cancha así como los keypoints más relevantes de la pista.
2. **Módulo de Seguimiento de Jugadores:** Detecta y sigue exclusivamente a los cuatro jugadores del partido.
3. **Módulo de Detección de Pelota:** Sistema especializado para pádel gracias al modelo de TrackNet.
4. **Módulo de Detección de Raquetas:** Identifica las raquetas en uso de cada jugador, realizado gracias a un gran etiquetado manual.
5. **Módulo de Visualización:** Genera el minimapa y overlay de información más destacable en tiempo real.
6. **Módulo de Estadísticas:** Calcula métricas físicas y tácticas, como la velocidad instantánea y acumulada de los jugadores, la distancia recorrida o la velocidad de la pelota.
7. **Módulo de Integración:** Coordina todos los componentes de una manera más sencilla y centralizada.

La Figura 4.1 muestra un diagrama general de la estructura del sistema y su organización en clases. Cada uno de los módulos mencionados está implementado mediante una clase principal que encapsula su funcionalidad y expone métodos específicos para inicializar, ejecutar y obtener resultados. Estas clases interactúan entre sí a través del módulo de integración, que actúa como controlador central, coordinando la ejecución de los distintos componentes y facilitando el flujo de datos entre ellos. Este diseño modular favorece la reutilización, la extensibilidad y el mantenimiento del código.

Para más detalles, el código fuente completo del proyecto está disponible en el siguiente repositorio de GitHub:

https://github.com/SergioPerezLoza/PadelDetections_TFG

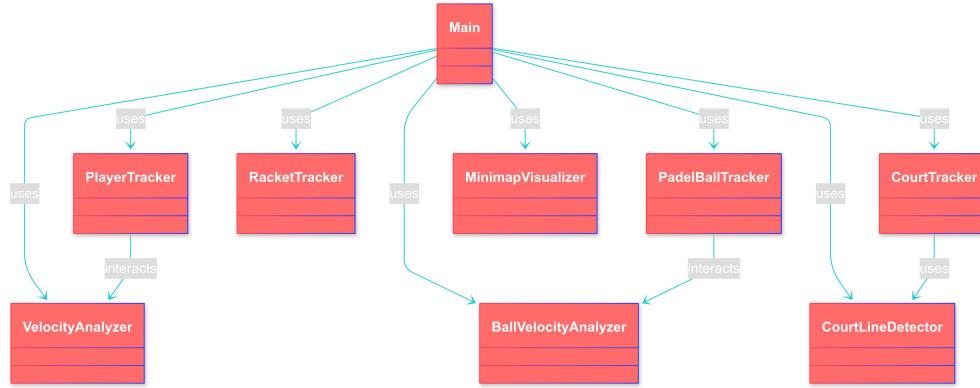


Figura 4.1: Diagrama general de la arquitectura del sistema y sus clases.

4.1.1. Flujo de procesamiento

El funcionamiento del sistema sigue un flujo secuencial en el que intervienen los distintos módulos descritos anteriormente, orquestados por el módulo de integración. El procesamiento de un vídeo completo sigue la siguiente secuencia:

Algorithm 1 Procesamiento de Vídeo de Pádel

- 1: Cargar vídeo de entrada
 - 2: Detectar pista y calcular matriz de homografía
 - 3: Inicializar estadísticas y variables
 - 4: **for** cada frame **do**
 - 5: Detectar jugadores
 - 6: Detectar pelota
 - 7: Detectar raquetas
 - 8: Filtrar detecciones válidas
 - 9: Actualizar trayectorias de jugadores y pelota
 - 10: Calcular velocidades de jugadores y pelota
 - 11: Actualizar estadísticas (distancia, velocidad media, etc.)
 - 12: Generar visualización (bounding boxes, trayectorias, estadísticas)
 - 13: Añadir minimapa táctico
 - 14: **end for**
 - 15: Guardar vídeo procesado
-

Este flujo garantiza que todas las detecciones y métricas se calculan de forma sincronizada y coherente en cada fotograma, produciendo un análisis completo del partido junto con su visualización correspondiente.

4.2. Componentes implementados

4.2.1. Keypoints de la cancha (CourtLineDetector)

Para la detección de los puntos clave de la pista de pádel se ha implementado una clase encargada de predecir y visualizar automáticamente las coordenadas más representativas de la cancha a partir de imágenes o fotogramas de vídeo. Esta funcionalidad se basa en una red neuronal convolucional del tipo *ResNet-50*, adaptada para devolver

diez puntos clave en forma de pares de coordenadas (x, y) . Los valores predichos por la red se reescalan adecuadamente para que se ajusten a la resolución original de cada imagen de entrada. A partir de estas coordenadas, es posible superponer sobre las imágenes los puntos detectados, tanto en fotogramas individuales como a lo largo de todo un vídeo, facilitando así su inspección visual y validación.

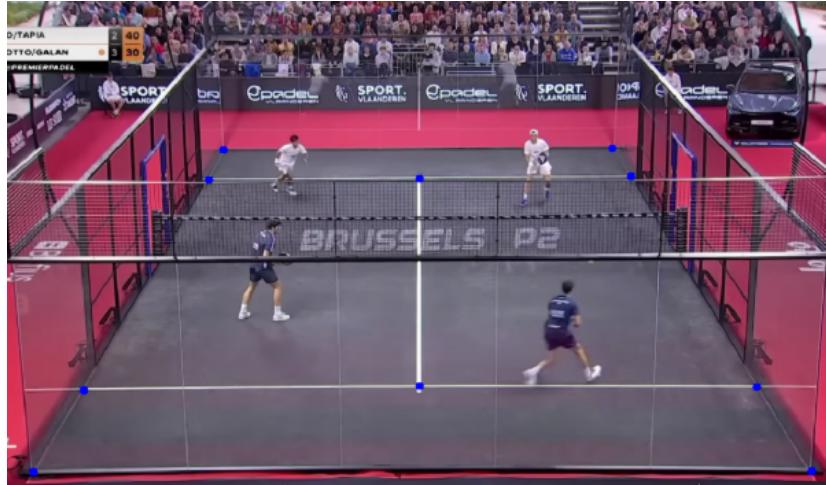


Figura 4.2: Visualización de los puntos clave detectados en una imagen de la pista.

La Figura 4.3 muestra un esquema de la estructura interna de la clase `CourtLineDetector` y su relación con otros componentes del sistema.

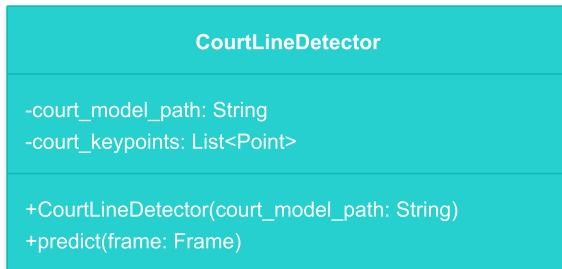


Figura 4.3: Diagrama de clases del módulo `CourtLineDetector`.

Además de su utilidad para la visualización y validación de la detección de la pista, los keypoints resultan fundamentales para la correcta identificación y selección de los jugadores en cada fotograma. Estos puntos permiten establecer una correspondencia geométrica entre la imagen original y las dimensiones reales de la pista, proyectando las posiciones de los jugadores y la pelota sobre un mapa bidimensional de forma coherente. En particular, el módulo de seguimiento de jugadores (`PlayerTracker`) utiliza los keypoints para calcular las distancias de todas las detecciones de personas con respecto a la pista y seleccionar automáticamente a los cuatro individuos más próximos a ella, descartando otros objetos o personas del público que puedan aparecer en la escena. Esta integración garantiza que el sistema identifique de forma consistente a los jugadores relevantes durante todo el partido, incluso en situaciones complejas con occlusiones o con múltiples personas visibles en el vídeo.

4.2.2. Detección de la cancha (CourtTracker)

En el desarrollo del sistema de análisis automático de vídeos de pádel, uno de los elementos clave ha sido la detección automática de la pista mediante el componente denominado **CourtTracker**. Este módulo tiene como objetivo principal localizar con precisión la posición y orientación de la cancha dentro del vídeo, permitiendo establecer una correspondencia geométrica entre las coordenadas del espacio de imagen y el espacio físico real. Esta capacidad es fundamental para que el sistema pueda situar correctamente a los jugadores y la pelota dentro de un contexto táctico, así como para generar visualizaciones como la del minimapa. La detección precisa de la pista es el punto de partida necesario para aplicar transformaciones como la homografía, que permite llevar cualquier detección del vídeo a un plano normalizado desde una vista aérea.

Un ejemplo del resultado de esta detección automática se muestra en la Figura 4.4, donde se aprecia la delimitación de la pista y la transformación geométrica estimada a partir de los puntos clave detectados.

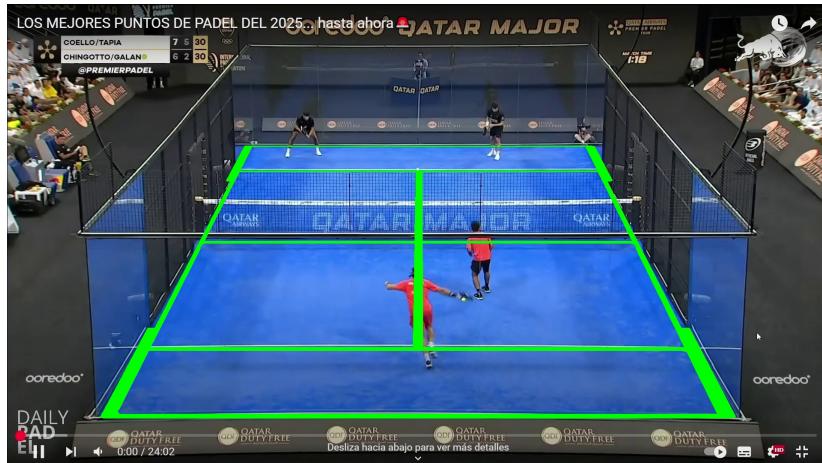


Figura 4.4: Ejemplo de detección de la pista y estimación de homografía con el módulo *CourtTracker*.

La Figura 4.5 muestra además un esquema de la estructura interna de la clase **CourtTracker** y cómo se organizan sus principales métodos y procesos, facilitando la comprensión de su diseño y funcionamiento.

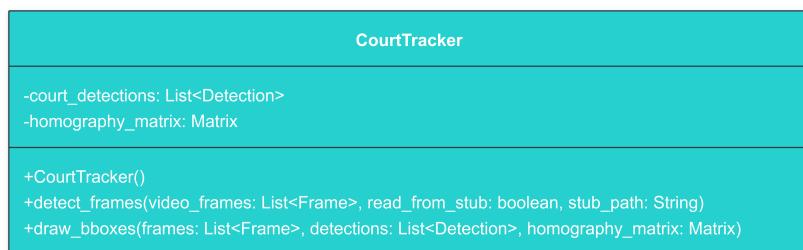


Figura 4.5: Diagrama de clases del módulo *CourtTracker*.

El proceso de detección se ha implementado mediante una combinación de técnicas clásicas de visión por computador. En primer lugar, se aplica segmentación de imagen utilizando K-means para aislar el área de juego respecto al fondo y facilitar la detección

de contornos. Posteriormente, se realiza una extracción de bordes y se detectan los contornos más relevantes en la imagen, de entre los cuales se seleccionan los que cumplen ciertos criterios geométricos para considerar que pertenecen a la pista. A partir de estos contornos se identifican puntos clave mediante la localización de intersecciones y extremos de las líneas detectadas. Estos puntos permiten estimar una homografía que transforma el espacio de píxeles en coordenadas reales de la cancha, como se ilustra en la Figura 4.4. La visualización superpuesta y el minimapa se basan en esta transformación, lo que permite un análisis posicional coherente a lo largo del vídeo. Esta lógica se encuentra encapsulada en el módulo **CourtTracker**, que puede funcionar de forma totalmente automática y sin supervisión, lo cual es esencial para el procesamiento de múltiples vídeos de forma escalable.

Durante el desarrollo de este componente se han encontrado diversas dificultades. Una de las más importantes ha sido la sensibilidad del sistema a las condiciones de iluminación y a los reflejos que generan los cristales de las pistas de pádel, los cuales a menudo provocan contornos o formas que confunden al detector. Además, la gran variabilidad en las perspectivas de grabación entre diferentes vídeos ha supuesto un reto para lograr una generalización fiable de los parámetros utilizados en la detección. En muchos casos, los contornos detectados no pertenecen a la pista real sino a elementos del fondo, sombras o líneas auxiliares, lo que ha requerido el desarrollo de filtros y heurísticas específicas para seleccionar únicamente los contornos válidos. Otro obstáculo ha sido la inexistencia de *datasets* públicos específicos para la detección de pistas de pádel, lo cual ha impedido entrenar modelos supervisados y ha obligado a apoyarse completamente en métodos no supervisados y heurísticos. Finalmente, también ha sido necesario considerar la estabilidad temporal de las detecciones, ya que errores puntuales pueden afectar la coherencia del análisis completo; para ello se ha aplicado una estrategia basada en la agregación de información a lo largo de múltiples fotogramas.

Cálculo de la homografía

La matriz de homografía es un elemento clave en la detección de la pista, ya que permite transformar las coordenadas de píxel en coordenadas reales de la cancha:

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

Esta matriz define una transformación proyectiva que relaciona dos planos: el plano de la imagen (en píxeles) y el plano de la pista (en coordenadas reales o normalizadas). Para calcularla, es necesario identificar al menos cuatro puntos no colineales en la imagen (*keypoints*) y sus correspondientes coordenadas en el plano real de la pista.

En este proyecto, los puntos clave de la cancha se determinan automáticamente mediante un análisis geométrico de los contornos detectados. A partir de los contornos de la máscara media de la pista, se localizan los puntos extremos superior, inferior, izquierdo y derecho de la pista y se extienden las líneas correspondientes para encontrar las intersecciones en las esquinas de la cancha. Estas cuatro intersecciones se ordenan para definir un cuadrilátero que representa la pista en la imagen.

Con las coordenadas de las esquinas en la imagen y las coordenadas de referencia de la cancha en el plano real, se calcula la homografía mediante la función `cv2.getPerspectiveTransform()`, que resuelve directamente el sistema lineal para obtener los coeficientes h_{ij} .

4.2.3. Detección de los jugadores (PlayerTracker)

Durante el desarrollo del sistema, se utilizó el modelo YOLOv8 combinado con el algoritmo de seguimiento ByteTrack para detectar y seguir a los jugadores en los partidos de pádel. Este enfoque permitió obtener detecciones precisas por fotograma y asignar identificadores numéricos (*track IDs*) a cada jugador. Sin embargo, pronto se evidenció un problema recurrente: estos identificadores son inestables, ya que el algoritmo puede reasignarlos si un jugador desaparece momentáneamente de la escena, cambia de postura, se cruza con otro jugador o incluso cambia de lado de la pista. Esto provocaba incoherencias en el análisis de trayectorias y dificultaba distinguir a los jugadores a lo largo del tiempo.

Para solucionar este problema, se desarrolló un sistema de **identificadores virtuales estables**, cuya lógica se implementó en la clase PlayerTracker. Este sistema se basa en una estrategia de reasignación dinámica e inteligente de los jugadores detectados a identificadores fijos (del 1 al 4), manteniéndolos constantes durante todo el partido. Inicialmente, se asignan en función de la posición de los jugadores en la pista utilizando la homografía previamente calculada. De este modo, se puede determinar en qué lado de la cancha se encuentra cada jugador y realizar la asignación virtual de forma coherente.

En la Figura 4.6 se ilustra un ejemplo de detección y seguimiento de jugadores en un partido de pádel, así como un esquema del proceso interno implementado en la clase PlayerTracker.

Para mantener la estabilidad, se implementó un historial por jugador, en el que se registran las posiciones pasadas, velocidades estimadas y marcos de tiempo de las últimas detecciones. Cuando un jugador desaparece temporalmente (por ejemplo, por oclusión), su posición se predice mediante una interpolación lineal basada en sus últimas ubicaciones válidas. Si el jugador vuelve a aparecer, se compara la detección nueva con la posición esperada según su trayectoria. Se calcula una distancia entre bounding boxes y se prioriza la asociación que respeta la coherencia espacial y temporal. Esto evita errores comunes como asignar un nuevo track ID a un jugador ya existente, o intercambiar jugadores cuando cambian de lado tras un punto.

Durante la implementación surgieron varias dificultades técnicas. Por ejemplo, fue necesario ajustar cuidadosamente los umbrales de distancia para decidir cuándo asociar una nueva detección con un jugador virtual o cuándo tratarla como un nuevo jugador. Además, se presentaron desafíos cuando los cuatro jugadores se concentraban en una zona similar, como cerca de la red, lo que provocaba confusión en el sistema de reasignación. Para mitigar esto, se introdujeron restricciones adicionales basadas en la dirección de movimiento y la historia de interacciones recientes.

Gracias a este sistema, fue posible realizar un seguimiento estable y coherente de cada jugador, permitiendo calcular trayectorias individuales, estadísticas por jugador y generar visualizaciones precisas con líneas de movimiento en la pista. Esta solución resultó ser esencial para el análisis automático de partidos, ya que proporcionó una capa de coherencia semántica sobre las detecciones brutas obtenidas por los modelos de visión por computador.

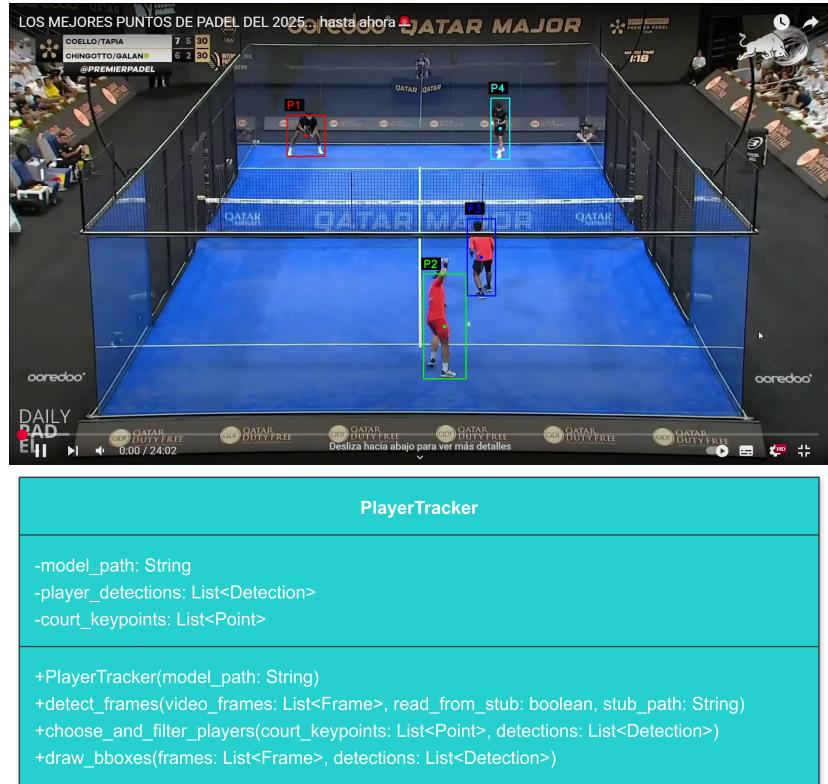


Figura 4.6: Ejemplo de detección y seguimiento de jugadores en un partido de pádel (arriba) y diagrama del proceso implementado en **PlayerTracker** (abajo).

4.2.4. Detección de la pelota (PadelBallTracker)

El código implementa un sistema de seguimiento de pelota en partidos de pádel utilizando técnicas de visión por computador y redes neuronales convolucionales. En concreto, se ha desarrollado una red convolucional personalizada denominada *BallTrackerNet*, diseñada con una arquitectura tipo encoder-decoder que toma como entrada tres imágenes consecutivas (el frame actual y los dos anteriores concatenados a nivel de canal), lo que permite capturar el contexto temporal necesario para localizar correctamente objetos de tamaño reducido como la pelota. La salida del modelo es un mapa de características en el que se identifica la probabilidad de presencia de la pelota en cada píxel, el cual es posteriormente procesado mediante técnicas de segmentación y detección de círculos con la transformada de Hough, permitiendo así extraer las coordenadas (x, y) más probables. Estas coordenadas, generadas inicialmente en una resolución fija asociada al modelo, se reescalan cuidadosamente al tamaño original del vídeo, garantizando así una detección precisa y coherente en el espacio de trabajo real.

En la Figura 4.7 se muestra un ejemplo visual de la detección de la pelota sobre un fotograma real del vídeo (parte superior) y un diagrama esquemático del proceso implementado en la red **BallTrackerNet** (parte inferior), ilustrando tanto la arquitectura como los pasos posteriores de segmentación y filtrado.

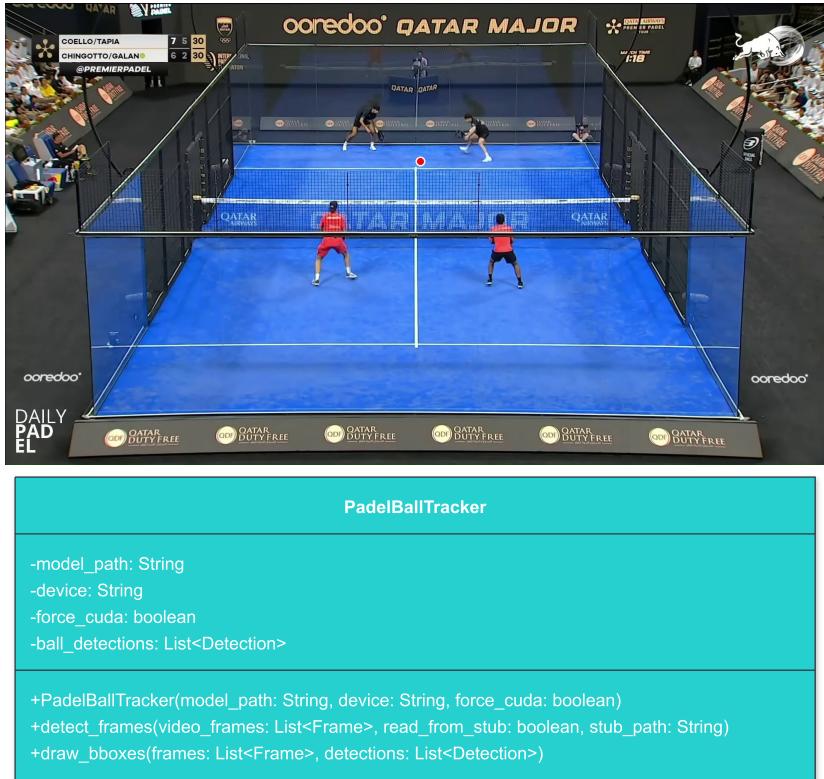


Figura 4.7: Detección de la pelota de pádel en un fotograma real (arriba) y diagrama del proceso implementado en BallTrackerNet (abajo).

Una vez obtenida la secuencia de posiciones, el sistema realiza un tratamiento adicional para filtrar *outliers*, definidos como aquellas detecciones que presentan saltos bruscos no coherentes con la trayectoria anterior. Para ello se calcula la distancia euclídea entre puntos consecutivos, y si esta supera un umbral prefijado, la predicción se descarta. Además, en caso de frames donde no se haya podido localizar la pelota, se lleva a cabo una interpolación lineal sobre los subsegmentos de la trayectoria, asegurando así la continuidad y estabilidad visual de la detección. Finalmente, las posiciones resultantes se convierten en una estructura de detecciones compatible con otros módulos del sistema y se visualizan sobre los fotogramas originales, incluyendo trazas de movimiento de longitud configurable y círculos resaltando la posición actual de la pelota, con elementos gráficos como colores, grosor variable y bordes para facilitar la interpretación visual del seguimiento.

Durante el desarrollo de este sistema se han presentado diversas dificultades. Una de las más destacables ha sido la necesidad de diseñar correctamente la arquitectura convolucional de forma que permita una detección sólida incluso en condiciones de movimiento rápido o baja visibilidad de la pelota. También ha resultado clave gestionar adecuadamente la escalabilidad espacial de las predicciones del modelo a resoluciones arbitrarias de vídeo, tarea en la que errores sutiles de redimensionado pueden provocar imprecisiones notables. Otro reto ha sido garantizar la robustez frente a falsos positivos, lo cual motivó la incorporación de mecanismos de filtrado por distancia y la interpolación selectiva de tramos incompletos. Asimismo, se ha tenido que gestionar la compatibilidad entre CPU y GPU, permitiendo acelerar la inferencia cuando CUDA está disponible pero sin que ello cause errores en entornos sin soporte. Por último, la visualización de la trayectoria y las detecciones ha exigido un diseño cuidadoso para

que el resultado sea informativo, visualmente claro y estéticamente coherente con el resto del sistema de análisis, como también se aprecia en la Figura 4.7.

4.2.5. Detección de la raqueta - Desarrollo de modelo personalizado (RacketTracker)

Para la detección de raquetas de pádel se optó por un enfoque personalizado, dado que no existen modelos preentrenados que identifiquen este tipo de objeto con precisión en contextos deportivos. Se decidió entrenar un modelo YOLOv8 desde cero, utilizando más de 1500 imágenes etiquetadas manualmente con la herramienta LabelImg. Esta es una aplicación de código abierto para la anotación de imágenes, ampliamente utilizada en tareas de visión por computador [38]. Permite definir manualmente cuadros delimitadores (*bounding boxes*) sobre objetos de interés y exportar las anotaciones en formatos compatibles con diferentes modelos de detección, como YOLO o Pascal VOC.

Este proceso de anotación se realizó delimitando cuidadosamente los cuadros delimitadores de las raquetas en una gran variedad de condiciones, ángulos y escenas, como se muestra en la Figura 4.8 (parte superior), donde se aprecia un ejemplo de imagen etiquetada manualmente.

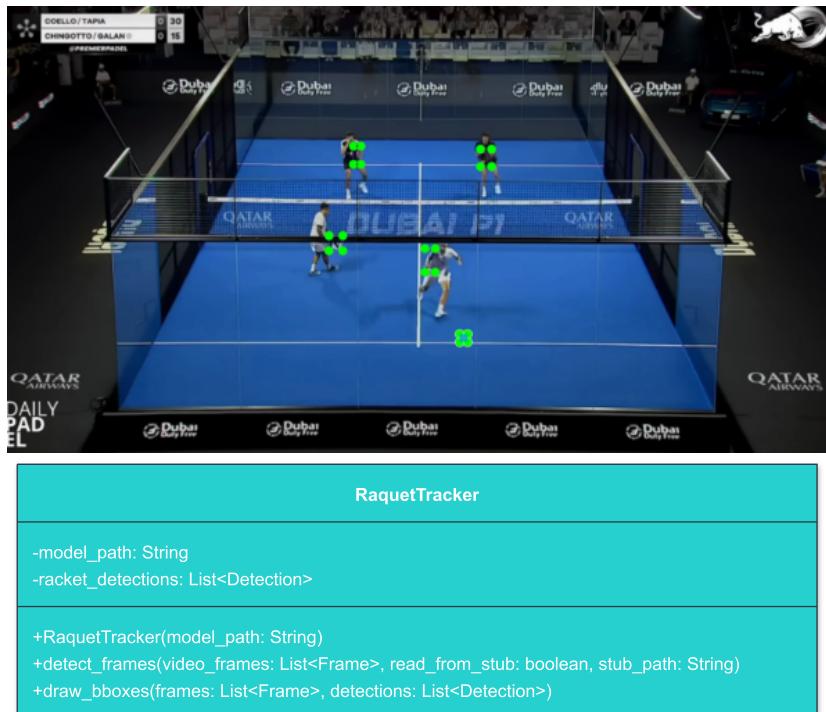


Figura 4.8: Ejemplo de imagen etiquetada manualmente con LabelImg para la detección de raquetas (arriba) y diagrama del proceso implementado en **RacketTracker** (abajo).

A pesar del esfuerzo dedicado a la creación del dataset, la detección no fue tan precisa ni robusta como se esperaba. Uno de los principales desafíos fue que las raquetas suelen ser objetos pequeños, parcialmente ocultos por los jugadores o el fondo, y en muchas ocasiones se confunden visualmente con los brazos o el entorno. Esto hizo que incluso con un conjunto relativamente amplio de datos, el modelo presentara una tasa de detección baja en ciertos contextos, especialmente en escenas con iluminación difícil o cuando la raqueta se encuentra en movimiento rápido.

Para intentar mejorar el rendimiento del modelo, se utilizó la técnica de **detección por regiones o 'slicing'**, empleando la librería SAHI (*Slicing Aided Hyper Inference*). Esta técnica divide cada imagen en múltiples fragmentos más pequeños y realiza la detección sobre cada uno de ellos por separado, lo que permite mejorar la precisión en objetos pequeños. En la Figura 4.8 (parte inferior) se muestra un diagrama del proceso completo implementado en **RacketTracker**, que incluye tanto el uso de YOLOv8 como la estrategia de slicing con SAHI. Se configuró el sistema para dividir cada fotograma en fragmentos de 80×80 píxeles con una superposición del 50%

La implementación se llevó a cabo mediante la clase **RacketTracker**, que encapsula toda la lógica de detección de raquetas. Esta clase carga el modelo YOLOv8 con SAHI, realiza la detección en todos los fotogramas del vídeo, y permite visualizar los resultados dibujando los cuadros delimitadores sobre las imágenes. Además, se incorporó una opción para guardar y reutilizar las detecciones en disco, optimizando así el tiempo de ejecución en pruebas repetidas.

Sin embargo, a pesar de estas mejoras, los resultados no alcanzaron la estabilidad deseada. En muchos fotogramas, la raqueta no era detectada, especialmente cuando se encontraba parcialmente oculta o fuera del centro del campo. Esto sugiere que, aunque 1500 imágenes etiquetadas representan un esfuerzo significativo, posiblemente no sean suficientes para cubrir la variabilidad visual necesaria. Una solución a este problema podría haber sido ampliar considerablemente el conjunto de datos, incluyendo más variaciones en fondo, posturas, tipos de raquetas y condiciones de iluminación. También se contempló la posibilidad de aplicar técnicas de aumento de datos (*data augmentation*) o utilizar modelos con mayor capacidad, aunque esto implicaría un mayor coste computacional.

4.3. Desarrollo e implementación del minimapa

En esta parte del proyecto se implementó un sistema de visualización denominado **MinimapVisualizer**, encargado de representar en un minimapa las posiciones y trayectorias de los jugadores y de la pelota durante un partido de pádel. El objetivo principal de este componente es transformar las coordenadas de detección, obtenidas en píxeles de la imagen del vídeo, a coordenadas normalizadas del mundo que permitan situar correctamente a los jugadores y la pelota en una representación esquemática de la cancha. Para ello, el sistema utiliza una matriz de homografía calculada previamente, que transforma puntos de la imagen al sistema de coordenadas de la cancha, y en caso de que dicha homografía no sea válida, aplica un método de respaldo para estimar las posiciones de manera aproximada.

En cada fotograma, la clase actualiza las posiciones de los jugadores tomando el punto medio de la base del *bounding box* (para situarlos en los pies sobre la pista), así como la posición de la pelota a partir de su centro. Estos puntos se almacenan en una cola de historial para poder dibujar las trayectorias suavizadas y con efecto de desvanecimiento. Posteriormente, se genera una imagen del minimapa donde se dibujan las líneas de la cancha, las trayectorias de la pelota y los jugadores, y las posiciones actuales de cada elemento, como se puede observar en la Figura 4.9 (parte superior).

Durante el desarrollo de este componente surgieron varias dificultades, principalmente relacionadas con la correcta transformación de las coordenadas. La homografía no siempre resultaba fiable, especialmente en los primeros fotogramas o cuando la detección de la pista fallaba, lo que provocaba que las posiciones quedaran fuera de

rango o mal situadas en el minimapa. Para mitigar este problema, se implementaron métodos de validación y normalización de las coordenadas, así como un sistema de respaldo cuando la homografía resultaba inválida. Otro reto importante fue encontrar una normalización adecuada de las coordenadas del mundo, ya que las escalas y los rangos variaban considerablemente dependiendo del vídeo y de cómo se había estimado la homografía. Para resolverlo, se calcularon estadísticas sobre los valores mínimos y máximos para ajustar la escala progresivamente a medida que se procesaban más fotogramas. También fue necesario invertir y ajustar los ejes para que la representación en el minimapa tuviera el mismo sentido que la realidad y no apareciera volteada o especular. Estas medidas permitieron obtener un resultado coherente y visualmente correcto.

El diagrama del proceso interno y la estructura de la clase `MinimapVisualizer` se muestran en la parte inferior de la Figura 4.9, donde se ilustran los principales métodos y flujo de datos que permiten la generación y actualización del minimapa en tiempo real.

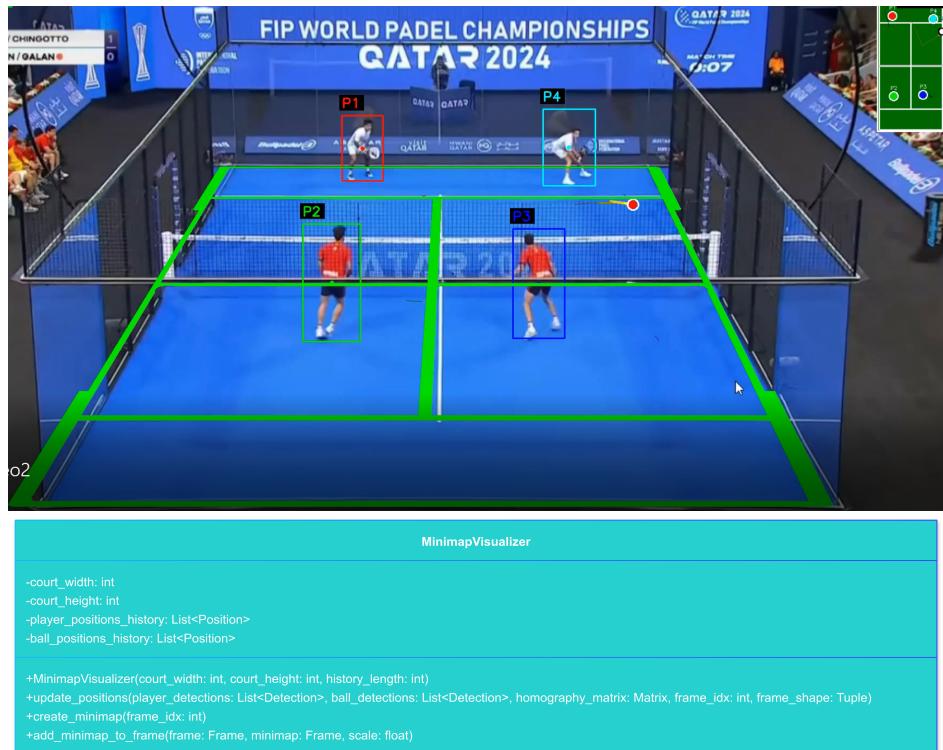


Figura 4.9: Minimap generado por el sistema (arriba) y diagrama de la clase `MinimapVisualizer` (abajo), mostrando las trayectorias y posiciones actuales de los jugadores y la pelota sobre la cancha.

4.3.1. Análisis de velocidad (`VelocityAnalyzer` y `BallVelocityAnalyzer`)

El análisis de la velocidad de los jugadores es una métrica fundamental para evaluar el rendimiento y la dinámica de un partido de pádel. Para ello, se ha desarrollado una clase `VelocityAnalyzer` que calcula la velocidad instantánea y acumulada de cada jugador a partir de las detecciones de posición en cada fotograma del vídeo. La velocidad se calcula midiendo la distancia entre posiciones consecutivas de un jugador,

expresada inicialmente en píxeles por frame, y posteriormente convertida a unidades reales (kilómetros por hora y metros) utilizando una calibración basada en la relación píxeles por metro y la tasa de frames por segundo del vídeo.

Para asegurar la precisión del análisis y evitar ruidos o movimientos erráticos en la detección, se implementa un umbral mínimo de movimiento y un filtro de suavizado que utiliza una ventana móvil con filtrado de valores atípicos. Además, se establecen límites realistas de velocidad máxima basados en estudios y referencias sobre la dinámica del pádel, descartando valores que superen los 25 km/h, que son improbables para este deporte.

El sistema registra y actualiza en cada fotograma la velocidad actual, la velocidad máxima alcanzada, y la distancia total recorrida por cada jugador, acumulando solo cuando el movimiento supera un umbral para garantizar que los datos reflejen desplazamientos reales y no pequeñas variaciones estáticas. Esto permite extraer estadísticas detalladas y relevantes para el análisis del rendimiento individual y colectivo.

Para facilitar la visualización, se diseñan paneles gráficos que muestran en tiempo real la velocidad instantánea y la distancia recorrida por cada jugador, con una representación clara y compacta en el vídeo. Esta información visual, junto con las estadísticas finales, proporciona una visión integral del esfuerzo físico y el dinamismo en el partido, tal como se muestra en la parte superior de la Figura 4.10.

Además, los diagramas de las clases *VelocityAnalyzer* y *BallVelocityAnalyzer*, presentados en la Figura 4.10 (centro y abajo), detallan la arquitectura y el flujo de datos interno que permiten realizar estos cálculos y actualizaciones en tiempo real, destacando también los colores asignados a cada jugador para facilitar su identificación y seguimiento durante el juego.

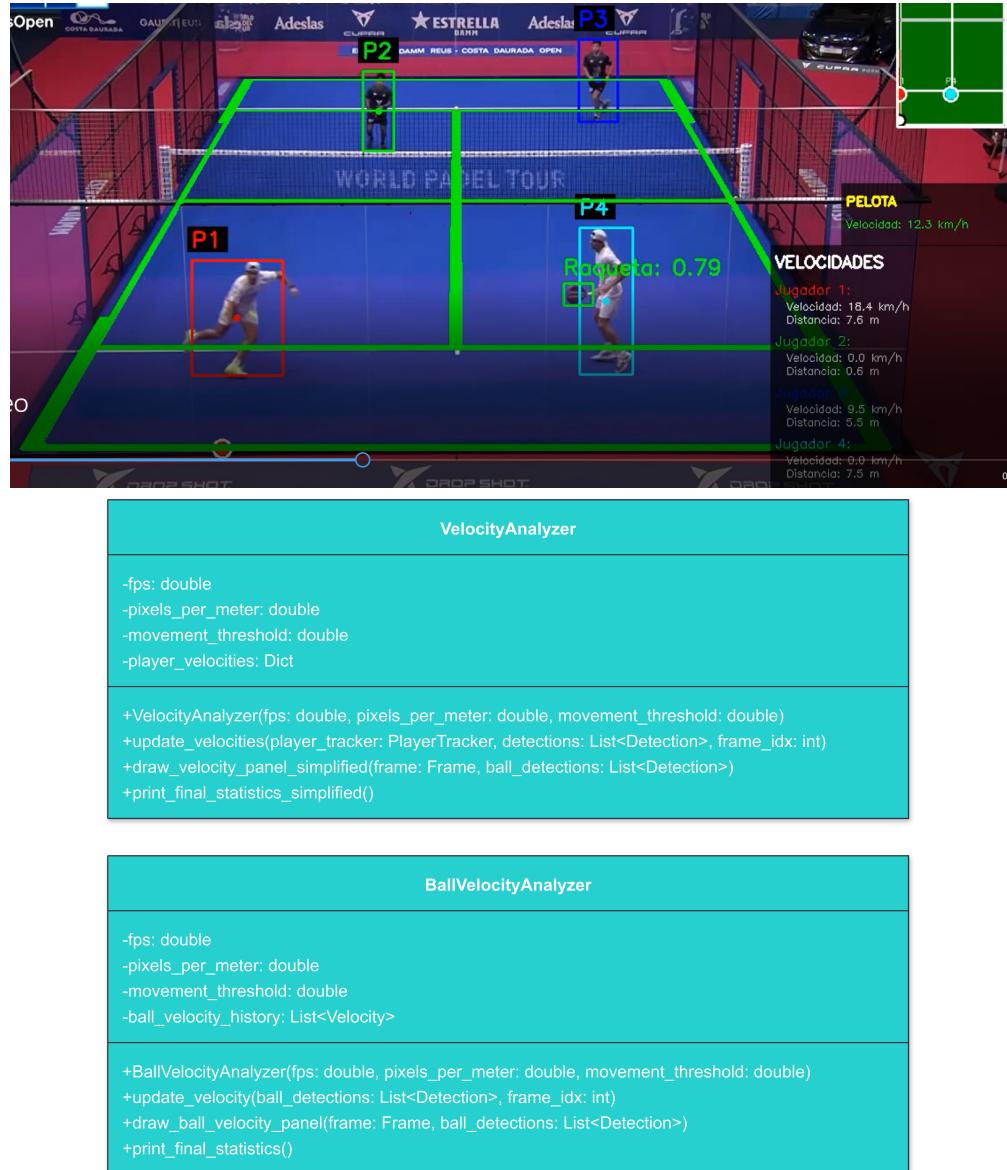


Figura 4.10: Panel de análisis de velocidad y distancia recorrida de los jugadores (arriba) y diagramas de las clases `VelocityAnalyzer` y `BallVelocityAnalyzer` (centro y abajo).

4.4. Optimizaciones implementadas

4.4.1. Sistema de caché

Para mejorar la eficiencia durante el desarrollo y testing, se implementó un sistema de caché que permite guardar y cargar detecciones previas:

- `player_detections.pkl`: Cache de detecciones de jugadores
- `padel_ball_detections.pkl`: Cache de detecciones de pelota
- `racket_detections.pkl`: Cache de detecciones de raquetas
- `court_detections.pkl`: Cache de detecciones de pista

4.4.2. Gestión de recursos

El sistema gestiona de forma automática los recursos de hardware, seleccionando preferentemente la GPU cuando está disponible y recurriendo a la CPU como alternativa en caso necesario, lo que asegura su compatibilidad y funcionamiento en distintos entornos de ejecución.

Capítulo 5

Resultados y evaluación

5.1. Evaluación del rendimiento del sistema

Para evaluar la eficiencia del sistema, se realizaron pruebas de rendimiento en dos configuraciones diferentes: procesamiento exclusivo con CPU y procesamiento acelerado con GPU. Los experimentos se llevaron a cabo utilizando vídeos de prueba de diferente duración para analizar el comportamiento del sistema bajo distintas cargas de trabajo.

5.1.1. Configuración experimental

Configuración CPU

Las pruebas con CPU se ejecutaron en un sistema con 56 núcleos Intel Xeon Gold 6238R a 2.2 GHz y 190 GB de RAM, procesando los vídeos de forma secuencial sin aceleración por hardware.

Configuración GPU

Para las pruebas con GPU se utilizó el supercomputador Anaga del Instituto Tecnológico y de Energías Renovables (ITER), específicamente un nodo equipado con 4 GPU NVIDIA A100 de 40 GB de VRAM cada una, con un total de 256 GB de RAM del sistema.

5.1.2. Resultados de rendimiento

Vídeo de prueba corto (7 segundos, 1080p)

Los resultados para el vídeo de prueba inicial se muestran en la Tabla 5.1. En procesamiento con CPU, puede observarse que los módulos más eficientes fueron la lectura y escritura de vídeo, el dibujo de las cajas delimitadoras y el procesamiento adicional de los fotogramas, todos con tiempos por debajo de los 5 segundos. La detección de jugadores y la detección de la pista presentaron tiempos moderados, del orden de 6.14 y 10.68 segundos, respectivamente.

Componente	Tiempo CPU (s)	Tiempo GPU (s)	Mejora (%)
Lectura de vídeo	0.78	0.83	-6.4
Detección de cancha	10.68	12.73	-19.2
Detección de jugadores	6.14	5.49	+10.6
Detección de pelota	112.43	5.26	+95.3
Detección de raqueta	7,491.58	1,983.46	+73.5
Dibujo de <i>bounding boxes</i>	3.62	1.44	+60.2
Procesamiento de fotogramas	1.17	1.43	-22.2
Guardado de vídeo	4.40	5.24	-19.1

Cuadro 5.1: Tiempos de ejecución por componente para vídeo de 7 segundos.

Vídeo de prueba largo (52 segundos, 1080p)

Para evaluar el comportamiento del sistema con cargas más realistas, se procesó un vídeo de mayor duración. Los resultados se presentan en la Tabla 5.2.

Componente	Tiempo CPU (s)	Tiempo GPU (s)	Mejora (%)
Lectura de vídeo	4.63	4.86	-5.0
Detección de cancha	9.75	11.17	-14.6
Detección de jugadores	50.60	73.23	-44.7
Detección de pelota	812.21	38.93	+95.2
Detección de raqueta	61,928.00	13,649.16	+78.0
Dibujo de <i>bounding boxes</i>	24.74	23.15	+6.4
Procesamiento de fotogramas	7.91	10.07	-27.3
Guardado de vídeo	37.32	41.20	-10.4

Cuadro 5.2: Tiempos de ejecución por componente para vídeo de 52 segundos.

5.1.3. Análisis de resultados

Detección de raquetas: el cuello de botella principal

El componente más costoso computacionalmente en ambas configuraciones fue la detección de raquetas, representando más del 90 % del tiempo total de procesamiento. Este módulo emplea la técnica *Slicing Aided Hyper Inference* (SAHI), que divide cada fotograma en múltiples fragmentos para mejorar la detección de objetos pequeños. Aunque esta estrategia logra mejores resultados de detección, implica una gran penalización en términos de rendimiento.

La aceleración por GPU redujo significativamente este tiempo (73.5 % en el vídeo corto y 78.0 % en el vídeo largo), pero sigue siendo el principal limitante del sistema. Para el vídeo de 52 segundos, el tiempo se redujo de aproximadamente 17.2 horas en CPU a 3.8 horas en GPU.

Detección de pelota: mayor beneficio de la aceleración

La detección de pelota mostró la mayor mejora relativa con GPU, con reducciones superiores al 95 % en ambos casos. Este componente se beneficia especialmente

de la parallelización masiva que ofrecen las GPU para el procesamiento de las redes neuronales.

Componentes con rendimiento similar o inferior en GPU

Sorprendentemente, algunos componentes mostraron rendimientos similares o incluso ligeramente inferiores en GPU, como la lectura/escritura de vídeo, detección de cancha y procesamiento de fotogramas. Esto se debe a que estas operaciones están limitadas por E/S o tienen una carga computacional relativamente baja que no justifica la transferencia de datos a GPU.

5.1.4. Rendimiento global del sistema

Velocidad de procesamiento

En términos de velocidad global de procesamiento, el sistema alcanzó, incluyendo todos los módulos:

- **CPU:** aproximadamente **0.5 fps**
- **GPU:** aproximadamente **1.8 fps**

Esta mejora con GPU es significativa en comparación con CPU, principalmente debido a la aceleración de las tareas de detección de pelota y raqueta, que son las más costosas. Sin embargo, el rendimiento sigue estando muy por debajo de los requerimientos para aplicaciones en tiempo real (30 fps o superiores).

Al analizar el impacto individual de la detección de la raqueta, se observa que este módulo es, con diferencia, el mayor cuello de botella. En el vídeo corto, el 98% del tiempo en CPU y más del 98% en GPU corresponden únicamente a esta tarea. En el vídeo largo, ocurre lo mismo: aproximadamente el 99% del tiempo en CPU y el 98% en GPU son atribuibles a la detección de la raqueta.

Si eliminamos el tiempo de la detección de la raqueta, la velocidad de procesamiento mejoraría considerablemente:

- En el vídeo corto, el tiempo total bajaría de 7630.8s a 139.22s en CPU, y de 2016.88s a 33.42s en GPU.
- En el vídeo largo, el tiempo total bajaría de 62875.2s a 947.2s en CPU, y de 13851.77s a 202.61s en GPU.

Esto equivaldría a velocidades aproximadas **sin contar la detección de raqueta** de:

- **Vídeo corto:**
 - CPU: 27 fps
 - GPU: 112 fps
- **Vídeo largo:**
 - CPU: 2.7 fps
 - GPU: 12.8 fps

Por tanto, el módulo de detección de raqueta es el principal responsable del bajo rendimiento, y optimizar o reemplazar este componente tendría un impacto muy positivo en la velocidad global del sistema, acercándolo incluso a condiciones casi en tiempo real en el caso del GPU para videos más cortos.

5.2. Métricas de precisión

Además de los tiempos de ejecución, se evaluó la precisión del sistema en la detección de los diferentes elementos utilizando dos videos de prueba. En el primer video, de 894 fotogramas, la pelota no fue detectada correctamente en 116 de ellos, lo que representa una precisión aproximada del 87 %. En cuanto a los jugadores, únicamente se produjeron 2 fallos de detección, alcanzando una precisión del 99.78 %.

En el segundo video, de mayor duración (1101 fotogramas), la pelota presentó 153 fallos de detección, lo que corresponde a una precisión cercana al 86 %. Para los jugadores, se contabilizaron 5 fallos sobre los 1101 fotogramas, manteniendo una precisión del 99.55 %. Aunque el número de fallos fue bajo, conviene destacar que este segundo video fue seleccionado intencionadamente por presentar un escenario más desafiante: en ese tramo del partido los jugadores se cruzaban con frecuencia y aparecían personas del público en el fondo, lo que incrementaba las posibilidades de confundir las detecciones y ponía a prueba la estabilidad del sistema en condiciones complejas.

En conjunto, los resultados muestran que la detección de la pelota mantiene una precisión media en torno al 86–87 %, mientras que la detección de jugadores es consistente, con una precisión superior al 99 % incluso en situaciones adversas.

Cuadro 5.3: Resultados de precisión en los dos videos analizados

Elemento	Vídeo	Fotogramas totales	Fallos	Precisión
Pelota	1	894	116	87.02 %
Jugadores	1	894	2	99.78 %
Pelota	2	1101	153	86.10 %
Jugadores	2	1101	5	99.55 %

5.3. Observaciones y ajustes durante el desarrollo

Durante el proceso de implementación y pruebas, se detectaron varias situaciones que requerían ajustes específicos o mejoras adicionales. A continuación se detallan algunos de los casos más relevantes:

5.3.1. Desafíos en la detección de pelota

Reflejos en cristales

Uno de los desafíos más significativos identificados fue la detección de la pelota cuando esta interactúa con los cristales de la pista. Los reflejos generados en las superficies transparentes crean múltiples candidatos visuales que confunden al modelo de detección, resultando en:

- Falsos positivos debido a reflejos de la pelota
- Pérdida temporal del seguimiento durante rebotes en cristal
- Dificultad para distinguir entre la pelota real y su reflejo

Este problema representa una limitación inherente del sistema que requiere desarrollo futuro con técnicas más avanzadas de filtrado temporal y validación física de trayectorias.

Limitaciones en escenarios de alta velocidad

La detección de la pelota presenta desafíos adicionales en momentos de alta velocidad, especialmente durante remates o golpes potentes, donde el motion blur y la velocidad de desplazamiento superan las capacidades del modelo actual.

5.3.2. Mejoras en la re-identificación de jugadores

La re-identificación de jugadores en el seguimiento durante el partido ha alcanzado un nivel de precisión muy alto gracias a varias mejoras implementadas en el sistema. Para evitar errores comunes como la pérdida de identidad, se aplicaron técnicas avanzadas que garantizan la continuidad y consistencia de las trayectorias de cada jugador, incluso en situaciones complejas como cruces o occlusiones temporales.

Entre las principales mejoras destacan:

- **Cálculo y seguimiento de trayectorias históricas:** Se mantiene un historial detallado de las posiciones de cada jugador para predecir sus movimientos futuros y validar nuevas detecciones.
- **Distancia mínima de coincidencia:** La asignación de detecciones se realiza evaluando la proximidad espacial con las posiciones previamente estimadas, evitando cambios bruscos de identidad.
- **Modelado de velocidad y movimiento:** Se analiza la velocidad y el desplazamiento entre frames para descartar movimientos poco realistas que puedan confundir la identidad.
- **Mapeo estable de IDs virtuales:** Se utiliza un sistema de IDs fijos que vincula las detecciones temporales a identificadores persistentes, manteniendo la coherencia durante todo el video.
- **Interpolación inteligente de posiciones:** Cuando un jugador no es detectado en un frame, se estima su posición basándose en movimientos previos para mantener el seguimiento continuo.

Capítulo 6

Conclusiones y líneas futuras

6.1. Conclusiones

Este Trabajo de Fin de Grado ha permitido desarrollar con éxito un sistema completo de análisis automático de vídeos de pádel, integrando diferentes técnicas de visión por computador y aprendizaje automático en un único sistema modular. La arquitectura diseñada facilita la incorporación de nuevos componentes, la reutilización de código y su mantenimiento.

El sistema desarrollado incluye funcionalidades como un minimapa táctico para la visualización de las posiciones y trayectorias de los jugadores y la pelota, la detección automática de la pista, y un módulo especializado para el seguimiento de la pelota. Asimismo, se implementaron mecanismos para reforzar el sistema frente a errores, occlusiones y condiciones cambiantes, lo que lo hace aplicable en contextos reales para entrenadores y analistas deportivos.

Durante el desarrollo se identificaron algunas limitaciones importantes, como la fuerte dependencia de las condiciones de iluminación y calidad del vídeo y la pérdida de identidad de los jugadores durante cruces y occlusiones prolongadas, así como la pérdida de la pelota cuando se refleja en el cristal o se mueve a gran velocidad. También destaca la alta dependencia de etiquetado manual para entrenar los modelos personalizados, y la ausencia de datasets públicos específicos para pádel, que obliga a crear los propios.

Uno de los principales desafíos observados es la elevada carga computacional asociada al procesamiento de vídeo, especialmente en el módulo de detección de raquetas, cuyo tiempo de ejecución resulta claramente desproporcionado debido a la necesidad de dividir las imágenes en fragmentos para detectar objetos pequeños. Esto evidencia que, si bien el sistema es funcional, para alcanzar un procesamiento cercano al tiempo real será necesario explorar otras tecnologías o técnicas, como la optimización de los modelos o el uso de arquitecturas más ligeras.

En definitiva, este proyecto contribuye al emergente campo del análisis automático de pádel, demostrando que es posible diseñar herramientas útiles y prácticas para el análisis táctico y técnico del juego. También pone de manifiesto la necesidad de seguir investigando en la mejora de la eficiencia computacional, la generación de datasets abiertos y la solidez de los modelos ante situaciones complejas en partidos reales.

6.2. Trabajo futuro

6.2.1. Mejoras a corto plazo

A corto plazo, se plantea la implementación de métricas automáticas de evaluación para cuantificar de manera objetiva el rendimiento de cada uno de los módulos del sistema. También se prevé la mejora del modelo de detección de raquetas, optimizándolo para reducir el tiempo de procesamiento sin comprometer la precisión, así como la optimización general del rendimiento para acercarse a un procesamiento en tiempo real. Otra mejora inmediata consistiría en el desarrollo de una interfaz gráfica de usuario que facilite el uso del sistema incluso para personas no técnicas. Además, se considera importante abordar el problema de los reflejos en los cristales de la pista mediante técnicas avanzadas de filtrado, y fortalecer los algoritmos de re-identificación para jugadores en situaciones de oclusión prolongada. Por último, sería muy beneficioso crear y poner a disposición de la comunidad investigadora un dataset público de pádel, que permita estandarizar las evaluaciones y fomentar nuevas investigaciones en este ámbito.

6.2.2. Extensiones a largo plazo

A más largo plazo, se propone extender el sistema con funcionalidades de análisis automático de estadísticas de juego, tales como la distribución de golpes o las zonas de mayor actividad. Asimismo, sería interesante incorporar técnicas de reconocimiento de patrones tácticos, que permitan identificar estrategias recurrentes de los jugadores. Sería interesante la integración del sistema con plataformas de transmisión en vivo, de modo que los datos y visualizaciones se puedan superponer en tiempo real durante las retransmisiones. Finalmente, una línea prometedora sería el desarrollo de una aplicación móvil complementaria, que permitiera a jugadores y entrenadores analizar partidos desde cualquier lugar. A largo plazo, incluso se podrían explorar técnicas de realidad aumentada para visualizar información directamente en la pista, aumentando el valor práctico y instructivo del sistema.

Capítulo 7

Summary and Conclusions

7.1. Summary

This project has focused on the design and implementation of an automatic video analysis system for padel, leveraging computer vision and machine learning techniques. The main objective was to detect, track, and visualize key elements of a padel match — players, ball, rackets, and court — in order to extract meaningful tactical and performance-related statistics.

The system has been developed using Python, taking advantage of libraries such as OpenCV and PyTorch. The architecture follows a modular design, consisting of several components: a court detector and tracker, player and racket trackers using YOLOv8 and SAHI, and a custom BallTrackerNet specialized in ball tracking. These modules were integrated into a pipeline capable of processing real match videos and generating outputs such as bounding boxes, trajectories, a 2D minimap, and speed and distance metrics.

Throughout the development, particular challenges such as occlusions, reflections on glass walls, fast movements, and the lack of public datasets specific to padel were addressed with custom solutions, heuristics, and strong estimations.

The system was evaluated on actual padel match footage, achieving satisfactory detection accuracy and acceptable performance, with the detection of rackets being the most computationally demanding due to the slicing technique used.

7.2. Conclusions

The main conclusions drawn from this project are the following:

- It is feasible to build a functional and solid automatic video analysis system for padel by adapting state-of-the-art techniques to the specific constraints of this sport.
- The modular architecture facilitates future extensions and improvements, allowing each component to evolve independently.
- The 2D minimap and the visualization of trajectories, speeds, and distances proved useful for tactical analysis, providing an intuitive understanding of players' movements.

- OpenCV played a critical role, not only in image processing and visualization, but also in computing homographies and extracting court contours reliably.
- While the system performs well overall, racket detection remains a bottleneck in terms of speed, and player re-identification under occlusions still poses challenges.

In summary, this project contributes to the emerging field of padel analytics by demonstrating the viability of automatic, data-driven tools for performance analysis. Future work should aim at optimizing the most time-consuming components, improving robustness under challenging conditions, and building annotated public datasets to foster further research.

Capítulo 8

Presupuesto

Este capítulo presenta una estimación detallada de los recursos económicos necesarios para la realización del Trabajo de Fin de Grado. El presupuesto incluye los costes asociados al tiempo de trabajo del estudiante, así como al uso de hardware, software y otros materiales requeridos durante el desarrollo del sistema. La finalidad de esta estimación es proporcionar una visión realista de los recursos implicados y justificar la viabilidad económica del proyecto.

8.1. Resumen de costes

Concepto	Descripción	Coste estimado (€)
Salario programador junior	300 horas a 18 €/hora (mercado)	5.400 €
Hardware	Uso de equipo personal (amortización)	500 €
Software	Licencias y servicios en la nube	0 € (software libre)
Uso de GPU	$4 \text{ GPUs} \times 3 \text{ h} \times 0,5 \text{ €/h}$	6 €
Uso de CPU	$56 \text{ CPUs} \times 20 \text{ h} \times 0,3 \text{ €/h}$	336 €
Total		6.242 €

Cuadro 8.1: Resumen del presupuesto estimado del proyecto.

Bibliografía

- [1] D. Yow, B.-L. Yeo, M. Yeung, and B. Liu, “Analysis and presentation of soccer video,” *Proceedings of IEEE International Conference on Image Processing*, vol. 3, pp. 697–700, 1995.
- [2] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR’05)*, vol. 1, pp. 886–893, IEEE, 2005.
- [3] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [4] M. Beetz, B. Kirchlechner, and M. Lames, “Aspogamo: Automated sports game analysis models,” in *International Journal of Computer Science in Sport*, vol. 5, pp. 29–44, 2006.
- [5] J. Perš, M. Bon, S. Kovačić, M. Šibila, and B. Dežman, “A trajectory-based analysis of coordinated team activity in a basketball game,” *Computer Vision and Image Understanding*, vol. 113, no. 5, pp. 612–621, 2009.
- [6] G. Pingali, Y. Jean, and I. Carlbom, “Real-time tennis tracking for televised tennis,” in *Proceedings 2000 International Conference on Image Processing*, vol. 2, pp. 49–52, IEEE, 2000.
- [7] J. Ren, J. Orwell, G. A. Jones, and M. Xu, “Learning to track and identify players from broadcast sports videos,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 12, pp. 1559–1567, 2006.
- [8] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, “Mot16: A benchmark for multi-object tracking,” *arXiv preprint arXiv:1603.00831*, 2016.
- [9] M. Fani, H. S. Yazdi, and D. A. Clausi, “Soccer video structure analysis by parallel feature fusion network and hidden-to-observable transferring markov model,” *Multimedia Tools and Applications*, vol. 76, no. 21, pp. 22343–22369, 2017.
- [10] Z. vZivković, “Improved adaptive gaussian mixture model for background subtraction,” *Proceedings of the 17th International Conference on Pattern Recognition*, vol. 2, pp. 28–31, 2004.
- [11] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition*, vol. 1, pp. I–I, IEEE, 2001.

- [12] P. Felzenszwalb, D. McAllester, and D. Ramanan, “A discriminatively trained, multiscale, deformable part model,” in *2008 IEEE conference on computer vision and pattern recognition*, pp. 1–8, IEEE, 2008.
- [13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.
- [14] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in *European conference on computer vision*, pp. 21–37, Springer, 2016.
- [15] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.
- [16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.
- [17] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, vol. 28, 2015.
- [18] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.
- [19] Z. Cai and N. Vasconcelos, “Cascade r-cnn: Delving into high quality object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6154–6162, 2018.
- [20] M. Archana and M. Geetha, “A survey on object detection and tracking methods,” *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 3, no. 2, pp. 691–696, 2015.
- [21] G. Gómez, P. López, and D. Link, “Automatic tennis ball detection and tracking,” in *2014 IEEE International Conference on Image Processing (ICIP)*, pp. 2808–2812, IEEE, 2014.
- [22] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, “Multiple object tracking: A literature review,” *Artificial Intelligence*, vol. 293, p. 103448, 2021.
- [23] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.
- [24] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.
- [25] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in *2017 IEEE international conference on image processing (ICIP)*, pp. 3645–3649, IEEE, 2017.

- [26] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in *2016 IEEE international conference on image processing (ICIP)*, pp. 3464–3468, IEEE, 2016.
- [27] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, “Bytetrack: Multi-object tracking by associating every detection box,” *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.
- [28] Y. Du, Y. Song, B. Yang, and Y. Zhao, “Strongsort: Make deepsort great again,” *arXiv preprint arXiv:2202.13514*, 2022.
- [29] J. Kautz, M. Sattler, R. Sarlette, R. Klein, and H.-P. Seidel, “Automated camera calibration and 3d egomotion estimation for augmented reality applications,” in *Proceedings Computer Animation 2000*, pp. 199–206, IEEE, 2000.
- [30] H. Collins and R. Evans, “The development of hawk-eye tennis calling system,” *International Sports Engineering Association*, 2005.
- [31] S. V. Mora and W. J. Knottenbelt, “Automatic analysis of tennis videos,” *arXiv preprint arXiv:1501.01221*, 2015.
- [32] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [33] M. Tan and Q. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in *International conference on machine learning*, pp. 6105–6114, PMLR, 2019.
- [34] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.
- [35] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [36] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.
- [37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [38] Tzutalin, “LabelImg: LabelImg is a graphical image annotation tool and label object bounding boxes in images.” <https://github.com/heartexlabs/labelImg>, 2015. Accedido el 2 de julio de 2025.