



TRABALLO FIN DE GRAO  
GRAO EN ENXEÑARÍA INFORMÁTICA  
MENCIÓN EN TECNOLOGÍAS DA INFORMACIÓN

## **Sistema para a obtención e visualización de estatísticas a partir de vídeos de pádel**

**Estudiante:** Óscar Lamas Ríos  
**Dirección:** Óscar Fresnedo Arias  
**Dirección:** Francisco Laport López

A Coruña, xuño de 2021.



*Por ser o apoio mais grande todos estes anos, Raquel Astray Calvo.*



## **Agradecementos**

Para comezar, agradecerles aos meus directores do proxecto, Óscar Fresnedo Arias e Francisco Laport López, por permitirme realizar con eles este proxecto, ademais de todo o esforzo e paciencia que tiveron conmigo.

Tamén lles estou moi agradecido polo apoio brindado a miña familia, compañeiros e profesores da facultade. Entre todos, fixeron do meu paso por esta institución unha experiencia satisfactoria e inolvidable.



## **Resumo**

A incorporación das novas tecnoloxías durante os últimos anos no ámbito deportivo permitiu que, cada día máis, sexa posible extraer grandes cantidades de información, datos e estatísticas de valor para a súa análise. Desta forma, a intelixencia artificial xoga un papel moi importante, permitindo procesar ditos datos e crear ferramentas para apoiar aos adestradores na súa estratexia, mellorar a práctica e o rendemento dos xogadores, xerar publicidade máis personalizada, ou analizar as debilidades dun rival, entre moitas outras aplicacións. Considerando este contexto e, sendo o pádel un deporte en continuo crecemento, buscaremos crear unha ferramenta software que proporcione información de interese a partir da análise de contido multimedia deste deporte. É de especial interese que o propio xogador poida fazer un estudo do seu partido, pois poderá estudar e corrixir os erros cometidos, os cales durante o encontro puideron pasan desapercibidos. É tamén importante a posibilidade de analizar o estilo de xogo dunha parella contrincante, pois achegará información dos seus puntos débiles que poderemos aproveitar durante o partido. Polo tanto, o noso proxecto dará como resultado unha aplicación web na que se poida consultar dita información dunha forma rápida e sinxela.

## **Abstract**

The incorporation of new technologies in recent years in the field of sports has promoted the extraction of large amounts of information, data and statistics of value for its subsequent analysis. In this regard, artificial intelligence plays a very important role, allowing to process of such data and create tools to support coaches in their strategy, improve player practice and performance, generate personalized advertising, or analyze the weaknesses of a rival, among many other applications. Considering this context and the continuous growth of padel, we will develop a software tool that provides information of interest from the analysis of multimedia content of this sport. It is of particular interest that the player himself can make a study of his game, as he will be able to study and correct the mistakes, which during the match may have gone unnoticed. It is also important to analyze the play of a rival since it will provide information on their weaknesses that we can take advantage of during the game. Therefore, our project will result in a web application where this information can be quickly and easily consulted.

---

**Palabras clave:**

- Intelixencia Artificial
- Visión artificial
- OpenCV
- Detección
- Máquinas Vector Soporte
- Histograma de Gradientes Orientados
- YOLOv3
- Estatísticas
- Django

**Keywords:**

- Artificial Intelligence
- Artificial Vision
- OpenCV
- Detección
- Support Vector Machines
- Histogram Of Oriented Gradients
- YOLOv3
- Statistics
- Django



# Índice Xeral

---

<b>1</b>	<b>Introdución</b>	<b>1</b>
1.1	Ámbito do problema . . . . .	1
1.2	Obxectivos . . . . .	3
1.3	Estrutura da memoria . . . . .	4
<b>2</b>	<b>Estado da arte</b>	<b>5</b>
2.1	Fútbol . . . . .	5
2.2	Tenis . . . . .	8
2.3	Pádel . . . . .	10
<b>3</b>	<b>Fundamentos tecnolóxicos</b>	<b>13</b>
3.1	Entorno de traballo . . . . .	13
3.1.1	Python . . . . .	13
3.1.2	Visual Studio Code . . . . .	14
3.1.3	OpenCV . . . . .	14
3.1.4	YOLO . . . . .	15
3.1.5	PostgreSQL . . . . .	15
3.1.6	Django . . . . .	16
3.1.7	Wondershare Filmora . . . . .	16
3.1.8	Git . . . . .	16
3.1.9	Overleaf . . . . .	17
3.2	Multimedia utilizada . . . . .	18
<b>4</b>	<b>Metodoloxía e planificación</b>	<b>19</b>
4.1	Descripción da metodoloxía . . . . .	19
4.2	Aplicación ao proxecto . . . . .	21
4.3	Planificación inicial e seguemento . . . . .	21
4.4	Costes económicos . . . . .	25

<b>5 Vista xeral da ferramenta</b>	<b>27</b>
5.1 Arquitectura do sistema . . . . .	27
5.2 Análise de requisitos . . . . .	29
5.2.1 Requisitos funcionais . . . . .	29
5.2.2 Requisitos non funcionais . . . . .	30
5.3 Diagramas de fluxo . . . . .	31
<b>6 Detección de xogadores</b>	<b>33</b>
6.1 Detección baseada en SVM e HOG . . . . .	33
6.1.1 Aplicación no proxecto . . . . .	36
6.1.2 Correción da detección . . . . .	41
6.1.3 Análise dos resultados de detección . . . . .	44
6.2 YOLO . . . . .	48
6.2.1 Residual blocks . . . . .	49
6.2.2 Bounding box regression . . . . .	49
6.2.3 Intersection Over Union (IOU) . . . . .	50
6.2.4 Combinación das 3 técnicas . . . . .	51
6.2.5 Aplicación de YOLO no proxecto . . . . .	51
6.2.6 Análise dos resultados da detección con YOLO . . . . .	52
6.3 Detección por parella de xogadores . . . . .	55
<b>7 Procesado dos datos e Gráficos</b>	<b>61</b>
7.1 Procesado de datos . . . . .	61
7.2 Transformación xeométrica . . . . .	63
7.3 Gráficos . . . . .	66
7.3.1 Gráficos individuais . . . . .	66
7.3.2 Gráficos colectivos . . . . .	68
<b>8 Aplicación web</b>	<b>71</b>
8.1 Deseño . . . . .	71
8.2 Bases de datos . . . . .	72
8.3 Exemplos de uso . . . . .	73
<b>9 Conclusíons e liñas futuras</b>	<b>79</b>
9.1 Conclusíons . . . . .	79
9.2 Liñas futuras . . . . .	80

## **ÍNDICE XERAL**

---

<b>A Material adicional</b>	<b>83</b>
A.1 Algoritmo de detección . . . . .	83
A.2 K-Means . . . . .	83
A.3 Transformación de perspectiva . . . . .	83
A.4 Gráficos . . . . .	83
<b>Relación de Acrónimos</b>	<b>93</b>
<b>Bibliografía</b>	<b>95</b>



# Índice de Figuras

---

2.1	Fases de preparación dun gol . . . . .	7
2.2	Táboa cos rangos de velocidade establecidos. . . . .	7
2.3	Gráfico de comparación de velocidad entre o xogador co dorsal 7 e o xogador co dorsal 3. . . . .	8
2.4	Coach Advisor . . . . .	9
2.5	Partido ficticio entre Roger Federer e Serena Williams . . . . .	9
2.6	Olocip . . . . .	10
3.1	Plano continuo que nos é de interese. . . . .	18
4.1	Planificación do proxecto xunto cun diagrama Gantt . . . . .	24
5.1	Overview do Proxecto . . . . .	27
5.2	Diagramas de fluxo do noso proxecto. . . . .	32
6.1	Clasificación de elementos usando SVM. . . . .	34
6.2	Deteccións con distintos valores para o parámetro <i>winStride</i> . . . . .	38
6.3	Exemplo de pirámide de imaxes. . . . .	38
6.4	Deteccións con distintos valores para o parámetro <i>scale</i> . . . . .	39
6.5	Resultado de aplicar Mean Shift Grouping sobre varias deteccións superpostas.	40
6.6	Resultado de aplicar NMS sobre varias deteccións superpostas. . . . .	40
6.7	Primeira aproximación de detección sen melloras. . . . .	41
6.8	Resultado tras aplicar NMS . . . . .	42
6.9	Liñas de código coas que aplicamos a máscara e procedemos ao reescalado da parte superior e inferior da pista. . . . .	43
6.10	Aplicación da máscara para a zona superior da pista. . . . .	43
6.11	Aplicación da máscara para a zona inferior da pista. . . . .	44
6.12	Detección coa máscara aplicada. . . . .	45

6.13	Exemplo das cuadrículas nas que se divide a imaxe.	49
6.14	Exemplo de cadro delimitador	50
6.15	Exemplo do funcionamento da técnica IOU.	51
6.16	Exemplo de combinación das 3 técnicas comentadas.	52
6.17	Detección de persoas utilizando YOLO	53
6.18	Conxunto de datos inicial xunto cos primeiros centroides	56
6.19	Primeira asignación dos puntos aos centroides	57
6.20	Cálculo dos novos centroides e visión dos puntos	57
6.21	Reasignación de puntos e cálculo dos novos centroides	58
6.22	Modelo final	58
6.23	Xogador correspondente á parella 1	59
6.24	Xogador correspondente á parella 2	59
7.1	Operación para realizar a transformación de perspectiva dunha coordenada.	63
7.2	Representación da transformación de perspectiva dunha ROI	64
7.3	Puntos que recollen o plano que imos transformar.	65
7.4	Imaxe tras a transformación de perspectiva.	65
7.5	Gráfico circular que representa as zonas do campo.	67
7.6	Mapa de calor para o xogador da dereita do partido do <i>Estrella Damm Santander Open 2021</i>	68
7.7	Diagrama de barras comparando as zonas da pista de dous xogadores.	69
7.8	Diagrama de barras das descoordinacións.	70
8.1	Funcionamento de Django.	72
8.2	Diagrama relacional	72
8.3	Páxina de inicio da nosa aplicación web.	74
8.4	Iniciar sesión na aplicación web.	75
8.5	Páxina de inicio da nosa aplicación web.	75
8.6	Sección da galería dos vídeos de un usuario.	76
8.7	Sección de contacto.	77
A.1	Script de probas da función <i>detectMultiScale</i>	84
A.2	Cores dominantes dunha imaxe con K-Means (I)	85
A.3	Cores dominantes dunha imaxe con K-Means (II)	85
A.4	Utilidades de K-Means	86
A.5	Liñas de código coas que aplicamos a transformación de perspectiva.	87
A.6	Liñas de código coas que construímos o gráfico de Zonas do Campo.	88
A.7	Liñas de código coas que debuxamos a pista.	88

## ÍNDICE DE FIGURAS

---

A.8	Liñas de código coas que construímos a matriz.	89
A.9	Liñas de código coas que construímos o mapa de calor.	90
A.10	Liñas de código coas que construímos o gráfico de comparación das zonas da pista.	91



# Índice de Táboas

---

4.1	Desviacións no proxecto. . . . .	25
6.1	Datos obtidos para a zona inferior da pista, Cupra Vigo Open 2021 . . . . .	46
6.2	Datos obtidos para a zona superior da pista, Cupra Vigo Open 2021 . . . . .	46
6.3	Datos da zona inferior da pista, Estrella Damm Santander Open 2021 . . . . .	47
6.4	Datos da zona superior da pista, Estrella Damm Santander Open 2021 . . . . .	47
6.5	Datos para a parella 1, Cupra Vigo Open 2021 . . . . .	53
6.6	Datos para a parella 2, Cupra Vigo Open 2021 . . . . .	54
6.7	Datos para a parella 1, Estrella Damm Santander Open 2021 . . . . .	54
6.8	Datos para a parella 2, Estrella Damm Santander Open 2021 . . . . .	55



# Capítulo 1

## Introducción

---

### 1.1 Ámbito do problema

A Intelixencia Artificial (**IA**) está revolucionando unha gran cantidade de sectores. Un deles é o sector deportivo que, gracias a **IA**, está acadando un nivel completamente novo a través de múltiples aplicacións enfocadas a mellorar, entre outras cousas, o rendemento de deportistas e técnicos.

Os sistemas de **IA** chegaron para cambiar a forma de facer as cousas. Nun sector no que, cada día máis, búscase a maior profesionalidade e especialización en todas as súas áreas, este tipo de sistemas proporcionan unha capacidade de mellora en canto a eficiencia e rendemento.

Cabe destacar tamén a importancia da **visión artificial** nas novas tecnoloxías. Podemos falar dela como unha técnica ou disciplina científica que inclúe os métodos necesarios para adquirir, procesar ou analizar e comprender as imaxes do mundo real, co fin de producir información numérica ou simbólica que poida ser tratada por un ordenador.

Tal e como os humanos usamos os nosos ollos e cerebros para comprender o mundo que nos rodea, a visión artificial trata de producir o mesmo efecto para que os ordenadores poidan percibir e comprender unha imaxe ou secuencia de imaxes e actuar segundo conveña nunha determinada situación. Esta comprensión conséguese a través de campos como son a xeometría e a estatística, entre outras disciplinas. Hai un gran número de tecnoloxías apoiadas na visión artificial, entre as cales se atopan o recoñecemento de obxectos, detección de sucesos, restauración de imaxes, etc.

Son moitas as áreas da industria deportiva nas que a **IA** e a visión artificial teñen cada vez un papel máis importante. A súa combinación pode resultar en sistemas moi útiles, onde podemos destacar as seguintes:

- Tecnoloxía de desenvolvemento para uniformes e equipación
- *Marketing*

- Periodismo automatizado
- Rendimento dos deportistas profesionais
- Observadores de fichaxes, métodos de traballo, etc.
- Servizos médicos
- Arbitraxes deportivas
- Análise de partidos e extracción de estatísticas relevantes.

Dentro destas áreas hai unha infinidade de posibilidades, podemos destacar os seguintes exemplos de cómo se pode mellorar o rendemento e a eficiencia no ámbito deportivo a través da implantación de sistemas de IA:

- **Mellora do rendemento:** a saúde e o deporte sempre van da man. A IA pode proporcionarnos dispositivos que recollen un gran número de datos e nos axudan a seguir o camiño correcto para ter unha boa saúde. Actualmente un gran número de deportistas levan un rigoroso control da súa composición corporal, así como do pulso ou frecuencia cardíaca durante a práctica de calquera deporte.
- **Adestramentos con IA:** podemos controlar nos adestramentos unha gran variedade de aspectos do xogo. Por exemplo, no caso do pádel, podemos medir unha gran cantidade de parámetros a través dunha combinación de cámaras e sensores, como son a velocidade coa que nos movemos, xiro e colocación dun saque, as zonas da pista nas que se xoga, os golpes realizados, os erros cometidos, etc.
- **Análise e selección de xogadores:** aínda que está menos relacionada co pádel, é importante mencionar este aspecto, pois é unha das grandes aplicacóns da IA na actualidade. En deportes como o fútbol, no que son moitos os xogadores que hai, ademais de que se invierte unha gran cantidade de diñeiro, a IA pode axudar a seleccionar os mellores talentos dunha forma moito máis obxectiva do que o faría unha persoa, baseándose en datos e predicións de potencial.
- **Toma de decisións arbitrais:** calquera competición, sen importar a disciplina, conta cunhas normas que establecen uns parámetros que se deben cumplir para que o resultado das accións sexa válido. O ollo humano non é infalible, e a marxe de erro pode chegar a ser moi alta cando afectan factores como o cansancio, ángulos complexos, etc. Incluso hai aspectos que o ollo humano non consigue captar, como diferencias moi sutís, pero que poden decidir quen gaña unha competición, ou se algún participante debe ser descualificado. Neste sentido, os sistemas de cámara superlenta, circuítos de vídeo

cerrado, ou o VAR, serven para impartir xustiza. Aínda que a marxe cero de erro non existe, con estes dispositivos é posible minimizala.

Os beneficios de utilizar a IA no deporte, xa sexa mediante os métodos comentados ou non, son enormes. Por un lado, é interesante **aplicar certa obxectividade** á análise do xogo sen ter que depender dun experto humano, que pode que obteña resultados distintos en función de factores como a experiencia, o contexto ou os seus coñecementos. Apoiándonos nas máquinas e na súa capacidade para almacenar e avaliar cantidades inxentes de información, podemos procesar unha **gran cantidad de datos**. Ademais, melloramos notablemente a **extracción de coñecemento** adquirida da explotación automática destes datos. Podemos descubrir algunas das razóns polas que un equipo gaña ou perde, datos para que os equipos técnicos poidan traballar e reforzar os aspectos positivos, así como corrixir o que está funcionando mal.

## 1.2 Obxectivos

O obxectivo principal deste proxecto é a elaboración dunha aplicación web que nos permitirá analizar partidos de pádel a partir de vídeos xa gravados, achegándonos información útil a través de diferentes gráficos, estadísticas e mapas de calor. Para isto, será necesario abordar os seguintes obxectivos concretos:

- Comprender as distintas funcionalidades que nos ofrece a librería OpenCV para o procesado de vídeos e para a detección de persoas baseadas en técnicas de aprendizaxe máquina.
- Aplicar as utilidades de OpenCV para o procesado dos *frames* dos vídeos de entrada, incluíndo diferentes técnicas básicas de procesado de imaxes (reescalado, descomposición multinivel, selección de áreas de interese, ...) xunto coa aplicación de máscaras sobre os *frames* que compoñen o vídeo co obxectivo de mellorar a detección de xogadores.
- Analizar e traballar cos principios básicos de recoñecemento e seguimento de xogadores en vídeos.
- Estudar como trasladar e representar graficamente a información de interese como pode ser as posicións dos xogadores nun mapa de dúas dimensións.
- Implementación dunha aplicación funcional que integre as diferentes etapas do proxecto, e que proporcione información analítica de interese para os usuarios de forma amigable e interactiva.

### **1.3 Estrutura da memoria**

Nesta sección faremos unha explicación moi breve de todo o contido da memoria resultante do noso proxecto. Temos a seguinte división por capítulos:

- **Introdución:** tratamos de plasmar o contexto no que se atopa a intelixencia artificial no mundo dos deportes. En relación con isto, explicamos en que vai consistir o noso proxecto, ademais dos distintos obxectivos que abordaremos. Tamén se inclúe esta sección, onde explicaremos brevemente a estrutura e o contido da memoria.
- **Estado da arte:** buscamos e analizamos produtos software no sector deportivo que utilicen a IA total ou parcialmente.
- **Fundamentos tecnolóxicos:** neste capítulo enumeramos as distintas tecnoloxías empregadas ao longo do noso proxecto, así como as versións utilizadas. Tamén explicaremos en qué consiste o contido multimedia que empregamos.
- **Metodoloxía e planificación:** capítulo dedicado á explicación en detalle da metodoloxía de traballo que se seguiu, así como a planificación inicial e o seguimento que se levou cabo, ademais dos custos económicos derivados do noso proxecto.
- **Vista xeral:** facemos unha breve descripción das diferentes partes que integran o noso proxecto.
- **Detección:** capítulo no que explicamos os algoritmos que se probaron para realizar unha detección dos xogadores de pádel, así como unha análise dos seus resultados.
- **Procesado de datos e gráficos:** capítulo no que se mencionan as estruturas de datos utilizadas no noso proxecto, así como o procesado e as transformacións xeométricas que experimentaron ditos datos.
- **Aplicación web:** capítulo dedicado á páxina web que se elaborou co obxectivo de exponer os servizos e utilizades elaboradas a posibles usuarios.
- **Conclusíons e liñas futuras:** finalmente presentamos as conclusíons extraídas a partir do traballo realizado.

## Capítulo 2

# Estado da arte

---

Como xa dixemos na introdución desta memoria, a IA está presente nun gran número de sectores, adquirindo cada vez máis importancia. O contexto deste proxecto é o sector deportivo, polo que neste capítulo realiza unha revisión dos produtos que xa existen no mercado, así como do uso que se lles da e da súa relevancia no deporte. Poñeremos especial atención no fútbol por ser o deporte rei actualmente, contando cun apoio enorme de organizacións e de aficionados que lle permite unha rápida incorporación das novas tecnoloxías. Ademais, analizaremos as aplicacións realizadas para o tenis por ser un deporte cunha estreita relación co pádel.

### 2.1 Fútbol

O fútbol conta cunha gran cantidade de apoio arredor do mundo, xa sexa por parte das organizacións como de aficionados. Como consecuencia, é o deporte para o que actualmente está mais explotada a IA. A continuación, describense algúns proxectos de interese:

- Das distintas ligas de fútbol que hai repartidas por todo o mundo, moitos consideran a **Premier League**, ou Liga Inglesa de Fútbol, como a de maior prestixio.

Moitos equipos desta liga plantexáronse a seguinte pregunta, *¿pode un algoritmo facer que un equipo xogue mellor ao fútbol e obteña melhores resultados?* A resposta é sí. Moitos dos grandes clubs saxóns puxérонse mans a obra a través dun acordo con **DeepMind**, unha compañía de IA inglesa. Na súa esencia, a proposta de DeepMind é adestrar sistemas e facer que estes acaben xogando miles de partidos de fútbol simulados e virtuais, nos que as condicións cambien unha e outra vez, para determinar cales son as mellores opcións en cada caso. A idea non é substituír os adestradores, senón desenvolver unha tecnoloxía de asistencia.

O obxectivo sería o de axudar ao adestrador a tomar decisións mellor informadas, ata o

punto de saber qué laterais poden ser más adecuados para xogar segundo qué contrincantes, ou qué esquemas e formacións son as mellores.

Algúns exemplos de moitas das conclusións que se poden sacar con este mecanismo son:

- Certos esquemas ou formacións son mellores contra uns contrincantes determinados.
- Tipo de xogadores cos que podemos ter más probabilidade de gañar contra uns contrincantes determinados.
- Axudar aos porteiros a “adivinar” a zona pola que o dianteiro rival vai tirar os penaltis.
- Tipos de pases que realiza o centrocampista rival.

Podemos ver unha ampliación desta información na referencia [1].

- **Humanox**, unha empresa de Cádiz, elaborou un produto innovador como son unhas novas espinilleras que, ademais de aliviar golpes, son capaces de recoller ata 50.000 datos do xogador que as leva, ofrecendo máis de 40 métricas distintas, desde a súa temperatura corporal ou a frecuencia cardíaca ata con qué perna e forza tocan o balón ou a súa velocidade no campo.

Actualmente, tanto o **Club Atlético Osasuna** como o **Cádiz Club de Fútbol** en todas as súas categorías están facendo uso de este producto. Ademais, conta xa con numerosos acordos con unha multitude de equipos.

Pódese ampliar a información sobre este proxecto na referencia [2].

- **AlProclips** é unha ferramenta deseñada polo **Grupo Mediapro** en colaboración con **IBM**. A funcionalidade que ten é, a partir de partidos completos, xerar resumos automáticos con todos os momentos chave do encontro, podendo fixar previamente unha serie de criterios personalizados.

Podemos ver unha ampliación desta información na referencia [3].

- **VisUsal**, un grupo de visualización da informática e analítica visual da universidade de Salamanca, desenvolveron un software de análise visual interactivo para equipos de fútbol. O obxectivo era crear unha ferramenta auxiliar para que se comprenda mellor o comportamento colectivo do equipo, ademais de tomar mellores decisións e deseñar estratexias. Cabe mencionar que analizan os partidos en tempo real, a partir de calquera retransmisión en vídeo destes. É a aplicación que máis relacionada está coa que se vai desenvolver no noso proxecto.

Manéxase unha gran cantidade de datos, complexos e dinámicos, utilizando distintos métodos de procesamento para presentalos dunha forma visual e interactiva. O usuario que manexa a aplicación pode cambiar o formato na que aparece representada a información, buscando resolver preguntas que só cunha representación pode resultar difícil.

Podemos ver unha ampliación desta información na referencia [4].

Algunha información destacable que nos ofrece esta ferramenta é:

- Fases de preparación dun gol, onde vemos recopiladas tanto as imaxes da retransmisión como unha representación visual destas nun mapa 2D que facilita a análise da posición dos xogadores. Na figura 2.1 podemos ver un exemplo desta presentación visual da información.

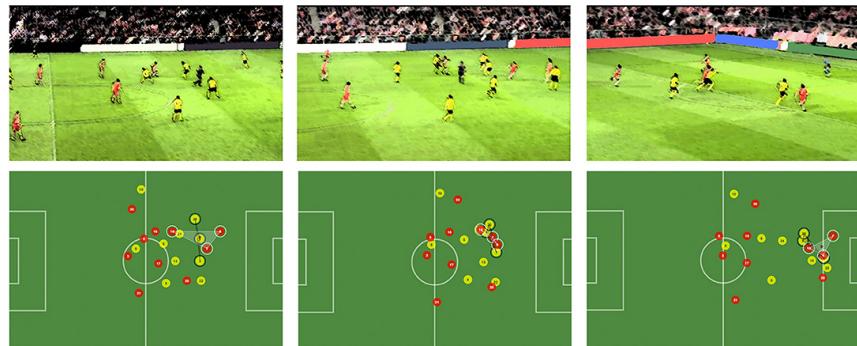


Figura 2.1: Fases de preparación dun gol.

- Comparación da velocidade entre dous xogadores, onde podemos establecer os nosos propios rangos de velocidade (ver a figura 2.2). Por exemplo, podemos comparar a velocidade dun atacante ca dun defensor nunha xogada de perigo, como se pode apreciar na figura 2.3

Pace	Speed Interval
Standing	Less than 0.2 m/s
Walking	Between 0.2 and 2.1 m/s.
Jogging	Between 2.1 and 3.8 m/s.
Running	Between 3.8 and 6.1 m/s.
Sprinting	More than 6.1 m/s.

Figura 2.2: Táboa cos rangos de velocidade establecidos.

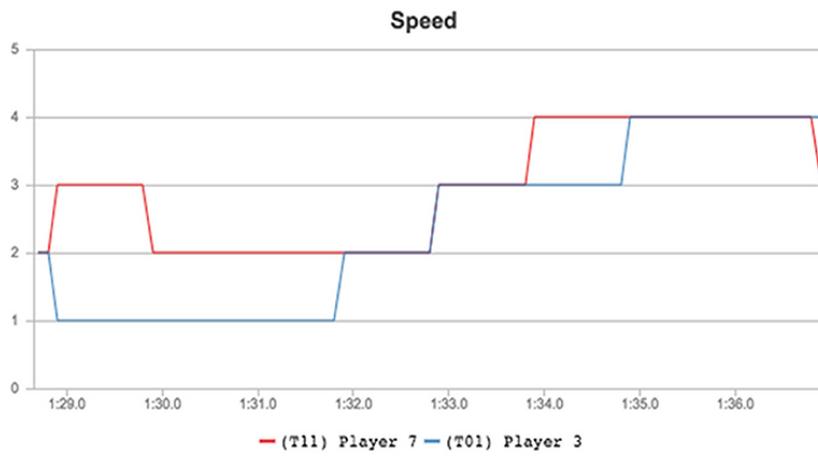


Figura 2.3: Gráfico de comparación de velocidad entre o xogador co dorsal 7 e o xogador co dorsal 3.

## 2.2 Tenis

- Como pudemos ver antes, IBM é unha das industrias pioneiras nas plataformas de IA. En referencia ao tenis, contamos co produto **Coach Advisor**, que se pode ver na figura 2.4. **Coach Advisor** é unha solución aloxada na nube, desenvolvida en colaboración coa **Asociación de Tenis de Estados Unidos** (USTA) e que aproveita a IA para brindar información aos adestradores profesionais. Parte da base dunha análise de vídeo, ademais de pequenos sensores no xogador e datos achegados previamente.

A base deste sistema é o que eles denominan **sistema de enerxía**, unha medida de carga fisiolóxica e intensidade mecánica destinada a avaliar o esforzo do xogador ao longo do tempo. Realiza un seguimento do traballo xeral realizado nun partido, tendo en conta a altura, o peso, a velocidade media e a distancia percorrida polo xogador. Ademais, controla a aceleración e desaceleración ponderada acumulativa en todo momento, o que permite medir a demanda física no corpo do xogador.

Dentro de **Coach Advisor**, as métricas do **sistema de enerxía** preséntanse xunto con estatísticas tradicionais e vídeos de partidos para que os adestradores os revisen. Os algoritmos adestrados para detectar o son dos golpes de pelota, os toques de atención dos árbitros, ruído do público e o son das zapatillas de deporte na cancha determinan os puntos de inicio e final do clip, o que lle permite aos adestradores saltar rapidamente aos momentos críticos do partido.

Podemos ver unha ampliación desta información na referencia [5].

- Outra aplicación interesante no mundo do tenis ven da man dos enxeñeiros da **Universidade de Stanford** con **Vid2Player**. Foron capaces de elaborar un sistema de



Figura 2.4: Coach Advisor

IA que, a partir dunha gran cantidade de gravacións de vídeo dos partidos, analiza os movementos de ambos os xogadores e consegue replicalos.

O seu sistema estuda os movementos e calcula a posición dos xogadores en cada momento, a súa colocación para recibir a bola e onde pode caer esta.

Como xa dixemos, o software pode replicar á perfección os movementos dos xogadores analizados, polo que é posible realizar simulacións de partidos ficticios. O programa é capaz de coller un partido histórico e cambialo por completo. Poderíamos ver a final de Wimbledon do ano 2008 coa vitoria de Roger Federer, en lugar de que Rafa Nadal se levara o trofeo, ou enfrentar dous xogadores que nunca competiron entre eles (figura 2.5)

Pódese ampliar a información sobre esta aplicación na referencia [6].



Figura 2.5: Partido ficticio entre Roger Federer e Serena Williams

- Por último, temos unha solución proporcionada por **Olocip**, compañía dedicada a unha gran variedade de produtos IA. Aportan unha solución capaz de modelar a información que se pode extraer dun partido de tenis e, cunha análise profunda do xogo, transformar dita información en novos coñecementos.

Con este software búscase proporcionar información complementaria ao equipo de traballo na análise do rendemento dos xogadores e dos seus rivais, así como da súa progresión. Ademais, emprégase numerosos principios de visión artificial para a representación do partido.

As características principales deste software son:

- Sistema de visión por computador capaz de sincronizar automaticamente movementos e eventos dos xogadores.
- Recoñecemento de xogadores e puntos anatómicos chave.
- Detección de pelota e golpeos.
- Rexistro de variables personalizadas. Podemos especificar qué tramo do partido nos interesa, así como os eventos ou tipos de golpes.

Podemos ver unha ampliación desta información na referencia [7].

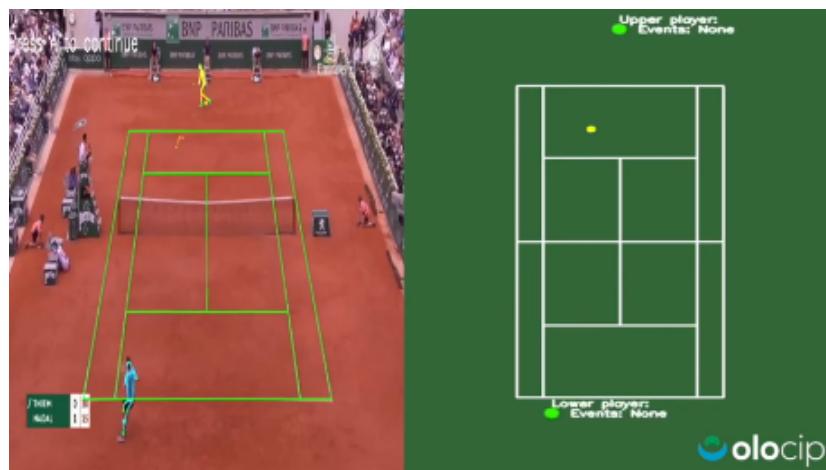


Figura 2.6: Olocip

## 2.3 Pádel

Durante as últimas décadas, e especialmente nos últimos anos, o pádel foi avanzando rapidamente na súa profesionalización. A preparación dos xogadores, o circuito profesional

e incluso o salto de calidade que pouco a pouco van experimentando as retransmisións, están nunha evolución continua e notable. Non obstante, no mercado actual non atopamos ferramentas que axuden á análise do xogo ou no manexo e creación de estatísticas. Unhas ferramentas que, como xa vimos noutros deportes, son de gran valor para os adestradores e os xogadores. Debido a isto, atopamos interesante o noso proxecto pois, máis alá dalgunha solución elaborada por aficionados deste deporte, non atopamos nada oficial en uso. Buscamos, polo tanto, aproveitar ese oco de mercado.



## Capítulo 3

# Fundamentos tecnolóxicos

---

Neste capítulo da memoria, explicaremos as tecnoloxías que foron utilizadas ao longo do proxecto. Falaremos tanto do software empregado para o desenvolvemento da ferramenta, como do material multimedia que foi utilizado de apoio e para comprobar o seu funcionamento.

### 3.1 Entorno de traballo

#### 3.1.1 Python

Un dos aspectos más importantes á hora de comezar o desenvolvemento dun produto software é seleccionar a linguaxe de programación. Neste caso, a decisión foi usar Python [8]. Trátase dunha linguaxe de programación de código aberto, cuxa filosofía ten como obxectivo manter sempre a lexibilidade do noso código. Python ademais de ser unha linguaxe de programación multiparadigma, tamén é multiplataforma. Estas dúas características fan que soporte parcialmente a programación orientada a obxectos e, áinda que en menor medida, tamén a programación funcional. Ao ser unha linguaxe interpretada, podemos escribir o noso código ata nun editor de texto se así o queremos. Porén, existen multitud de [IDE](#) que nos facilitan o traballo en moitos aspectos. Veremos en detalle isto no seguinte punto deste capítulo. Utilizamos esta linguaxe na totalidade do noso proxecto. Aínda que o código é compatible con versións anteriores, a que se utilizou neste caso foi a **versión 3.9.1**. Dentro das vantaxes más significativas de Python, podemos destacar as seguintes, pois tiveron especial relevancia á hora de elixir dito linguaxe como o pilar do noso proxecto:

- É unha linguaxe extremadamente produtiva, pois ten unha sintaxe simple e fácil de aprender.
- Licenza de código aberto, dispoñible a utilizar para os usuarios de forma gratuíta.

- Linguaxe portable e multiplataforma.
- Ten unha ampla comunidade activa de desenvolvemento.
- Gran cantidade de bibliotecas dispoñibles para estender as funcionalidades.
- Presenta unha fácil integración con outras linguaxes de programación.
- Permite a implementación integral dunha aplicación completa, desde a lóxica do *back-end* ata a parte da *front-end* para a interacción co usuario.

### 3.1.2 Visual Studio Code

Un **IDE** non é máis que unha aplicación informática que proporciona servizos integrais para facilitar ao programador o desenvolvemento dun software. No noso caso, empregamos **Visual Studio Code** [9] como **IDE** para Python, na súa **versión 1.56.2**, principalmente polas seguintes características:

- Soporte para múltiples linguaxes de programación, apoiándose na gran cantidade de extensións que axudan a minimizar os erros e a traballar máis rápido para unha linguaxe en concreto.
- Soporte de terminal. Moitas veces o usuario necesita comenzar a execución desde a raíz do directorio para certas accións particulares, polo que o terminal incorporado proporciona apoio ao usuario e así non ten que andar manexando unha pantalla máis.
- *Debugging* completo, con puntos de interrupción, pilas de chamadas e unha consola interactiva, entre outras cousas.
- Soporte multiplataforma, pois podemos utilizalo tanto en Windows, Linux ou Mac.
- Capacidad de abrir simultaneamente varios proxectos que conteñan múltiples arquivos ou carpetas.

### 3.1.3 OpenCV

Neste proxecto tamén se fixo uso de **OpenCV** [10], na súa versión **4.2.0**. Trátase dunha biblioteca libre de visión artificial desenvolvida por Intel, compatible con Python na súa totalidade. Entre as súas principais vantages, destaca o feito de que nos proporciona unha gran cantidade de funcións e capacidades relacionadas co procesado de vídeos, deseño de elementos gráficos, recoñecemento de obxectos, detección de movemento, reconstruccións 3D a partir de imaxes, etc. Neste caso concreto, faremos uso destas funcións nas seguintes áreas:

- Para todo o referente ao manexo dos *frames* que componen os vídeos que utilizaremos, incluíndo diferentes técnicas de procesado de imaxes (reescalado, descomposición multinivel, selección de áreas de interese, ...), xunto coa aplicación de máscaras sobre os *frames* co obxectivo de mellorar as deteccións de xogadores que faremos posteriormente.
- Para analizar os diferentes *frames* do vídeo e detectar os xogadores dun partido de pádel mediante algoritmos de aprendizaxe máquina.

### 3.1.4 YOLO

You Only Look Once [11] é un método de detección de obxectos de código aberto. Fai uso dunha única rede neuronal convolucional para detectar obxectos en imaxes. Para o seu funcionamento, a rede neuronal divide a imaxe en rexións, predicindo cadros de identificación e probabilidades para cada rexión. O algoritmo aprende representacións xeneralizables dos obxectos, obtendo un baixo erro de detección para entradas novas, diferentes ao conxunto de datos de adestramento.

No noso caso, utilizaremos dito algoritmo na súa **versión 3** (*YOLOv3*), xa pre-adestrado.

### 3.1.5 PostgreSQL

Desde o momento que decidimos traballar cunha aplicación web, necesitamos utilizar unha base de datos para gardar a distinta información que imos manexar. No noso caso, eliximos o xestor de bases de datos relacional **PostgreSQL** [12], na súa **versión 13**.

Unha das características más interesantes que ofrece este xestor é a posibilidade de utilízalo tanto cunha aplicación de escritorio como en servidor web. Ademais, presenta unha gran escalabilidade, pois podemos configurar PostgreSQL en cada equipo en función do hardware, axustándose ao número de CPUs e a cantidade de memoria dispoñible de forma óptima. Con isto logramos unha maior cantidade de peticións simultáneas á base de datos de forma correcta e con menor latencia. Outras características a destacar son:

- Estabilidade e confiabilidade. Con máis de 20 anos de desenvolvemento activo e en constante mellora, rara vez acontecerá unha caída da base de datos. Isto é grazas a súa capacidade de establecer un entorno de alta dispoñibilidade e ao chamado *Hot-Standby*, que permite que os clientes poidan realizar consultas de lectura mentres os servidores están en modo recuperación ou espera. Así pódense levar a cabo tarefas de mantemento ou recuperación sen bloquear completamente o sistema.
- Potencia e robustez. Cumpre coas características **ACID**, é dicir, achega atomicidade, consistencia, illamento e durabilidade. Esto permite que as transaccións non interfieren unhas con outras, garantizando a integridade da información na base de datos.

- Extensibilidade. Temos dispoñibles unha gran variedade de extensiós, tanto por parte dos desarrolladores de PostgreSQL como de terceiros. En particular, atopamos un gran número delas relacionadas con Python, polo que é moi interesante para o noso proxecto.

Por último, cabe destacar tamén a ferramenta gráfica *pgAdmin*, coa que podemos administrar as nosas bases de datos de forma fácil e intuitiva. Permiteños executar sentencias SQL, crear copias de seguridade ou incluso realizar tarefas de mantemento. Concretamente, no noso caso utilizamos a versión **pgAdmin 4** [13].

### 3.1.6 Django

Django é un *framework* web que nos da a capacidade de realizar aplicácións web de calquera complexidade [14]. Está escrito en Python e ten unha comunidade moi ampla, en continuo crecemento. No noso proxecto utilizaremos a **versión 3.1.5** para elaborar a aplicación web a través da cal expoñeremos o noso sistema para que poida ser usado.

Django tamén nos achega un grado de seguridade alto, pois implementa por defecto algunas medidas de seguridade para evitar SQL Injection, CSRF ou Clickjacking por JavaScript. Ademais, é un *framework* que presenta un grao de escalabilidade e versatilidade moi alto. Ademais de polas características que acabamos de comentar, o alumno está familiarizado con este framework, por iso foi o elixido.

Cabe mencionar tamén o uso de **Bootstrap** [15] na súa **versión 4** para o deseño da nosa páxina web. Trátase dun *kit* de ferramentas de código aberto para desenvolvimentos web con *HTML*, *CSS* e *JavaScript*. Inclúe unha gran cantidade de compoñentes, como son menús, cadros, botóns, formularios, entre moitos outros, a través dos cales permiteños crear interfaces de usuario limpas e totalmente adaptables a todo tipo de dispositivos e pantallas, sexa cal sexa o seu tamaño.

### 3.1.7 Wondershare Filmora

Para a edición do contido multimedia a utilizar, empregouse o software **Wondershare Filmora** [16], na súa **versión 10.0.0.91**. Calquera editor de vídeo sinxelo sería válido para este propósito, pero elixiuse o mencionado posto que o alumno xa estaba familiarizado con él.

### 3.1.8 Git

Git é basicamente un xestor de código fonte, tamén chamado xestor de versións [17]. Trátase dunha ferramenta que se encarga de rexistrar todos os cambios polos que van pasando os diferentes arquivos que forman parte do noso proxecto, xa sexan arquivos de código ou

doutro tipo. Faino de modo que poidamos consultar ditos cambios, reverter eses cambios, ver quen os fixo e cando, entre moitas outras cousas.

Ademais permite colaborar con outras persoas, traballando en paralelo en diferentes características do proxecto, resolver conflitos, ter copias de seguridade dos arquivos minimizando todo tipo de riscos, entre outras moitas características.

Aínda que esteemos a traballar en solitario neste proxecto, resulta fundamental protexer o código contra posibles problemas que xurdan durante o desenvolvemento ou para aforrar tempo no caso de que sexa necesario volver a unha versión previa do mesmo. Por esa razón, o uso dun sistema de control de versión é sempre unha boa práctica.

Podemos destacar as seguintes vantaxes de Git que o fan especialmente axeitado para este proxecto:

- Software de código aberto, e de uso gratuíto.
- Súper rápido e lixeiro, optimizado para facer operacións de control de maneira moi rápida.
- Podemos crear diferentes ramas para partes concretas do proxecto, e combinalas de forma rápida, sinxela e pouco propensa a problemas.
- A integridade da información está asegurada grazas ao seu modelo de almacenamento, polo cal mantemos en todo momento unha coherencia xeral dos nosos datos. Noutros sistemas tradicionais este aspecto era un problema grave.
- Permite un fluxo de traballo moi flexible.
- O concepto de área de preparación ou *staging* permite versionar os cambios como mellor nos conveña, non é todo ou nada.

Por outro lado, atopámonos cun software cunha curva de aprendizaxe moi pronunciada, no que temos unha gran cantidade de operacións e posibilidades que o converten nunha ferramenta bastante complexa para xente sen experiencia. Os comandos e algúns conceptos usados poden chegar a ser confusos se un non está afeito.

### 3.1.9 Overleaf

Para elaborar a memoria na que explicaremos todo o proceso do desenvolvemento do noso proxecto debemos seleccionar o editor de texto máis axeitado.

Overleaf [18] é un editor colaborativo baseado na nube que se utiliza para escribir, editar e publicar documentos científicos. Utiliza código *LaTeX*, o cal é compilado de forma automática a medida que imos escribindo os nosos documentos, mostrando os resultados de forma

simultánea. Ao ser unha plataforma web non necesitamos ningún tipo de instalación nin actualización. Ademais, funciona en todos os entornos e en calquera dispositivo.

É especialmente interesante utilizar *LaTeX* neste tipo de documentos, pois permítenos separar o contido do seu formato. Así, temos a oportunidade de concentrarnos no ‘qué’, e o sistema encargarase de cómo plasmar as nosas ideas visualmente no documento. Realiza de forma automática tarefas como numerar capítulos e figuras, engadir e organizar a bibliografía adecuada ou manter os índices e referencias.

## 3.2 Multimedia utilizada

O contido multimedia externo que utilizaremos no noso proxecto componse dunha serie de vídeos de distintos partidos de pádel correspondentes ao circuíto profesional de pádel de referencia mundial, o **World Padel Tour**. Istoos vídeos normalmente comparten unha serie de características respecto á colocación da cámara e ao tipo de plano para a gravación dos puntos.

Para facilitar o procesamento dos vídeos en fases posteriores, primeiro farase un preprocesado para seleccionar aquelas partes do vídeo onde se vexan os diferentes puntos do partido e os movementos dos xogadores, sen cambios de plano, repeticións a cámara lenta, descansos, entre outras cousas. Deste xeito, os nosos vídeos compoñeranse na súa totalidade dunha sucesión de planos como o que se mostra na figura 3.1. No noso caso, empregamos o software **Filmora** para levar a cabo esta fase de preparación dos vídeos en cru.



Figura 3.1: Plano continuo que nos é de interese.

## Capítulo 4

# Metodoloxía e planificación

---

É moi importante elixir unha metodoloxía adecuada para o desenvolvemento dun produto, pois define un marco de traballo práctico e funcional a través do cal minimizaremos os posibles riscos para acabar presentando un producto con éxito. Na elaboración deste proxecto, seguiuse unha metodoloxía de traballo **iterativa incremental**.

### 4.1 Descripción da metodoloxía

Para comprender mellor en qué consiste o modelo iterativo-incremental, debemos entender primeiro o modelo en cascada. Dito modelo segue un enfoque clásico no que se describe un método de desenvolvemento lineal e secuencial. Consta de varias fases, as cales están definidas por diferentes tarefas e obxectivos, de tal forma que o inicio de cada etapa debe esperar a finalización da anterior. A totalidade das fases describen o ciclo de vida do software ata a súa entrega. Aínda que nalgún proxecto poden ser máis, xeralmente temos 5 fases:

- **Análise:** planificación, análise e especificación dos requisitos. Fase que comeza cun estudo de viabilidade no que se avalían aspectos como os custos, rendibilidade, factibilidade, etc. Desta forma, obtemos como resultado unha folla de condicións ou requisitos xunto cunha estimación financeira do proxecto. Tamén nesta fase se leva a cabo unha definición detallada dos requisitos.
- **Deseño:** especificación do sistema que se vai desenvolver. Establécese a arquitectura software do proxecto, así como un plan de deseño detallado da mesma, centrándose en compoñentes concretos como interfaces, entornos de traballo, bibliotecas, etc.
- **Implementación:** inclúe a programación software, xunto coa detección de posibles erros e as probas unitarias correspondentes.
- **Verificación:** tamén coñecida como fase de proba, inclúe a integración do software

no entorno seleccionado. Compróbase se o software cumpre coas probas de aceptación establecidas.

- **Mantemento:** unha vez pasada a fase de proba, autorízase a aplicación produtiva do software. Inclúe a entrega do produto, xunto co seu mantemento e mellora.

O principal inconveniente deste modelo é que require ter unha especificación dos requisitos totalmente detallada e completa desde un primeiro momento, para poder iniciar a etapa do deseño. Se ao longo do proceso de desenvolvemento se cambian os requisitos, hai que refacer parte do traballo de deseño e implementación para poder fazer fronte aos novos cambios.

Como alternativa a este modelo lineal surxe a **metodoloxía iterativa incremental** que facilita que os requisitos non estean totalmente especificados para comenzar co desenvolvemento do software, de xeito que podemos adaptarnos de forma máis flexible a posibles cambios ou incidencias ao longo do ciclo de vida do producto.

Esta metodoloxía caracterízase porque, en cada nova entrega (ou incremento) que facemos do producto durante o seu ciclo de vida, imos engadindo novas funcionalidades (incremental), coa peculiaridade de que cada incremento tamén inclúe melloras sobre as funcionalidades que xa existían (iterativo).

Por tanto, os resultados parciais que imos obtendo pódense entender como incrementos ou miniproxectos nos que, en cada iteración, se repite un proceso de traballo similar para proporcionar un resultado completo. En cada unha destas iteracións séguese habitualmente o modelo típico de cascada (análise, deseño, implementación e probas). Deste xeito, ao final de cada iteración imos ter unha versión do producto sobre a que se van engadindo novas características nas seguintes iteracións. En cada iteración evoluciónase o producto a partir dos resultados completados nas iteracións anteriores, engadindo novos obxectivos e requisitos ou mellorando os que xa foron completados.

## Vantaxes

- A complexidade global do proxecto vese reducida en pequenas partes menos complexas.
- Con cada iteración, o risco que se asume é pequeno, pois se hai algunha incidencia ou cambio, pódese solucionar facilmente nas seguintes iteracións.
- O coñecemento adquirido sobre o producto é crecente e progresivo, sen necesidade de ter unha visión detallada de cada parte ao principio do desenvolvemento.
- O risco de que o proxecto falle na súa totalidade é mínimo, pois os errores dun incremento poden corrixirse no incremento posterior.

- Xestión das expectativas do cliente xa que, en cada nova iteración, este pode ver unha versión funcional do produto e seguir en detalle os avances que se van logrando ao longo do desenvolvemento.

#### Desvantaxes

- Pode resultar complicado que cada unha das nosas iteracións aporten realmente un valor ao cliente, debemos entregar uns resultados que sexan capaces de ser utilizados. Por iso, as iteracións son especialmente custosas.
- A disponibilidade do cliente debe ser alta durante todo o proxecto, pois participa nel de maneira continuada. Debe detallar os requisitos que se van desenvolver en cada unha das iteracións, ademais de revisar o resultado acadado ao final destas.

## 4.2 Aplicación ao proxecto

Como se comentou anteriormente, elixiuse a metodoloxía iterativa incremental para o desenvolvemento deste proxecto. Unhas das principais razóns que motivou esta elección foi a posibilidade que ofrece de adquirir un coñecemento máis progresivo e crecente das distintas tecnoloxías que se van usar. Este factor resulta fundamental xa que a IA era un campo totalmente descoñecido para o autor do traballo, así como a libraría de OpenCV, de vital importancia neste proxecto. Ademais, resultaba moi complicado ter unha visión detallada de cada parte do proxecto antes de comezar co seu desenvolvemento. Do mesmo xeito, os requirimentos da aplicación non estaban totalmente definidos dende un principio, senón que podía ir sufrindo certas variacións ou refinamentos a medida que avanzaba o proxecto. Por todo isto, dita metodoloxía pareceunos a mellor elección.

Con esta metodoloxía de traballo, teremos **varios incrementos** ou ciclos de vida individuais, para os que seguiremos as súas correspondentes etapas. Como se tratan de incrementos nos que se traballa con distintas tecnoloxías e aspectos do proxecto, é difícil dar unha explicación en común para todos eles, pero os aspectos principais sempre foron os mesmos. Os incrementos foron planificados desde un primeiro momento, especificando detalladamente os obxectivos a cumplir en cada un deles, cunha posterior reunión para analizar o resultado de incremento correspondente.

Segundo as directrices establecidas pola metodoloxía escollida, realizouse unha planificación inicial para o proxecto que se detalla na seguinte sección.

## 4.3 Planificación inicial e seguemento

Para comenzar con esta sección, debemos diferenciar entre dous tipos distintos de recursos humanos. Temos dúas persoas que actúan como directores do proxecto, e un membro único

no equipo de desenvolvemento.

Como en calquera outro proxecto, todo é resultado dunha primeira reunión inicial informativa, neste caso entre o alumno e os directores do traballo, que foi realizada o día **26/01/2021**. Nesta reunión debateuse sobre os diferentes contidos que se poderían tratar no traballo fin de grao (TFG) e acotouse o alcance do mesmo.

Ademais, fíxose tamén unha planificación inicial, establecendo os distintos incrementos ou iteracións que ía ter o proxecto. Programáronse reunións entre os directores e o alumno ao final de cada iteración, co obxectivo de avaliar o traballo realizado e definir os seguintes aspectos a abordar.

Segundo a metodoloxía incremental elixida, definíronse unha serie de fases ou etapas, de tal forma que o proxecto quedou organizado da seguinte maneira:

- **Iteración 1: Fase de contacto.** É necesario adentrarse un pouco no contexto deste traballo e facer unha toma de contacto coas tecnoloxías a utilizar, estamos ante unha etapa de mero estudio e aprendizaxe. O obxectivo é que o alumno se familiarice coas distintas librerías e software que se utilizarán no proxecto, como OpenCV, entre outras. Realizaranse distintos programas coas súas correspondentes probas. A duración estimada desta iteración é de **5 días**.
- **Iteración 2: Primeira aproximación.** Obtención da primeira versión do producto, onde o obxectivo era ser capaces de detectar aos xogadores de pádel e obter as súas posicións na pista ao longo da secuencia de vídeo. A duración estimada desta iteración é de **20 días**.
- **Iteración 3: Procesado de datos.** Etapa dedicada soamente ao procesado e almacenamento de datos obtidos na anterior iteración. Debemos definir a estrutura de datos más axeitada para o noso caso e realizar as transformacións xeométricas necesarias para que ditos datos sexan representativos e se poidan explotar na seguinte iteración. A duración estimada desta iteración é de **20 días**.
- **Iteración 4: Presentación visual.** Unha vez temos os datos preparados, debemos explotalos e sacarlle todo o valor que poidamos. O obxectivo desta iteración é conseguir representar visualmente a información que queremos, utilizando distintos gráficos e estatísticas relevantes. A duración estimada desta iteración é de **5 días**.
- **Iteración 5: Aplicación web.** O obxectivo desta fase é elaborar unha aplicación web, a través da cal se buscará ofrecer as funcionalidades feitas nas iteracións anteriores aos usuarios. Estes poderán subir os seus vídeos e ver as distintas estatísticas que lle presentamos. No noso caso utilizaremos o *framework* Django. A duración estimada desta iteración é de **20 días**.

- **Iteración 6: Exploración de alternativas e melloras.** Ao longo desta fase, buscaranse alternativas e outras estratexias de detección de xogadores co obxectivo de mellorar a precisión das posicións que obtemos. Ademais, tamén se seguirá traballando na aplicación web, engadindo novas funcionalidades como pode ser a xestión de usuarios. A duración estimada desta iteración é de **20 días**.
- **Iteración 7: Documentación do proxecto.** Elaboración dunha memoria para documentar o traballo realizado no proxecto. A duración estimada desta iteración é de **20 días**.

A duración das diferentes iteracións foi planificada con certa folgura para dispor de certa marxe á hora de evitar posibles desviacións durante a súa realización. Isto permitiu que a maior parte das iteracións remataran na data prevista.

Unicamente a elaboración memoria xunto coa extracción das conclusións finais comezou 10 días máis tarde do esperado. Tanto a iteración 5, dedicada á aplicación web, como a iteración 6, dedicada a exploración de alternativas e melloras do noso proxecto, consumiron cada unha 5 días máis do previsto, de aí o atraso comentado.

Podemos ver a planificación do proxecto final, xunto cun diagrama de *Gantt* na figura 4.1

Resumindo a información que acabamos de comentar, temos que, por parte do **alumno**, dedicóuselle ao proxecto un total de **360 horas**, repartidas ao longo duns **120 días**, dedicándolle un tempo aproximado de 3 horas cada día.

Por outro lado, debemos ter en conta as horas dedicadas por parte dos **directores**. Realizáronse un gran número de reunións no transcurso do proxecto, que acabarían facendo un total de **14 horas**, repartidas da seguinte forma:

- A reunión informativa inicial, na que se planificou o proxecto, cunha duración de 2 horas.
- Unha reunión para cada momento onde damos por finalizada unha iteración e pasamos á seguinte, é dicir, 6 reunións máis, tamén de 2 horas cada unha.

#### *4.3. Planificación inicial e seguimiento*

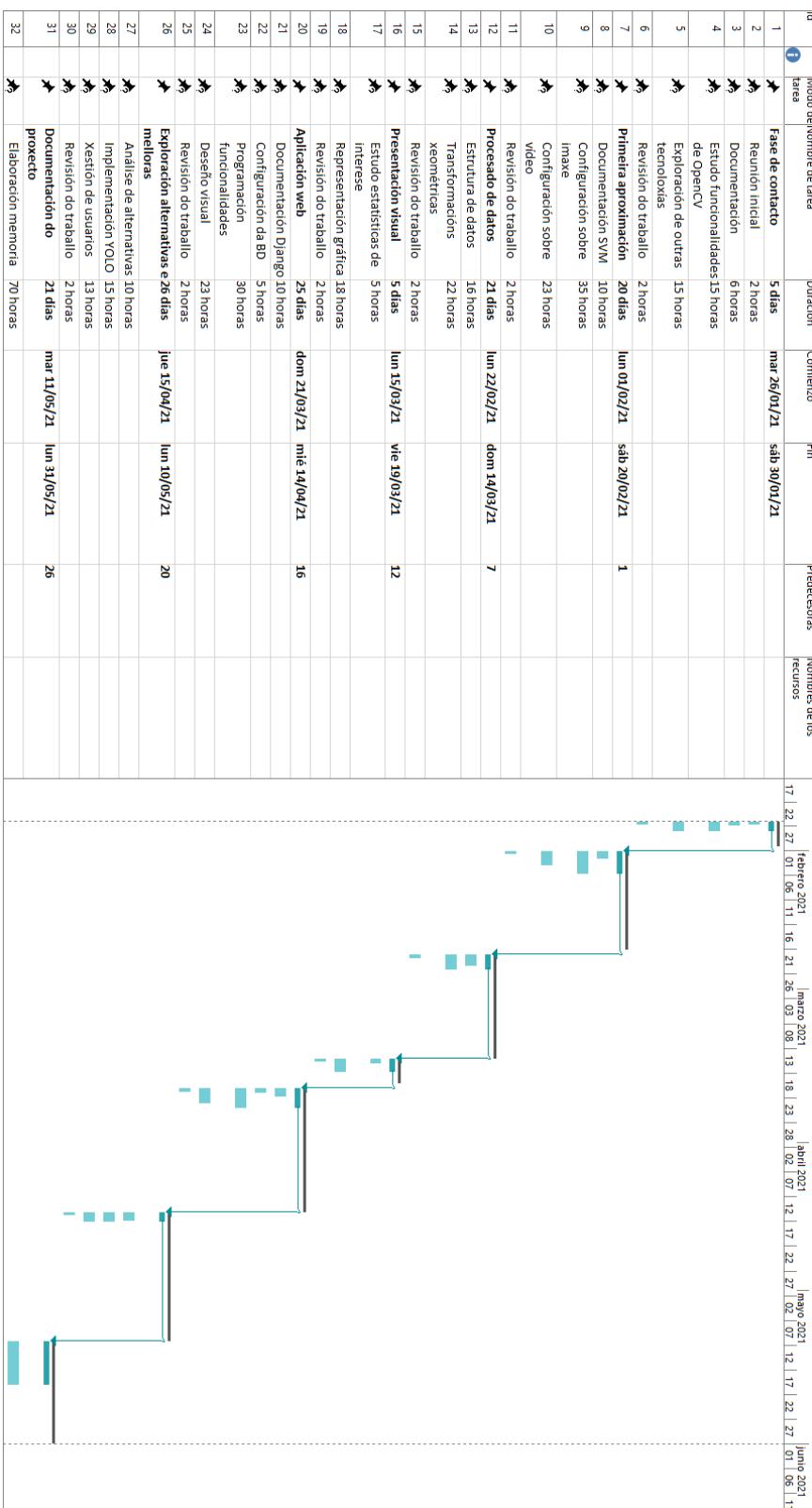


Figura 4.1: Planificación do proxecto xunto cun diagrama Gantt

## 4.4 Costes económicos

Para calcular os custos económicos resultantes da elaboración do proxecto, debemos ter en conta tres tipos distintos de recursos. Coa información do apartado anterior, procedemos a calcular o custo da totalidade do proxecto.

- **Recursos hardware:** 0 €, pois utilizáronse os ordenadores persoais tanto do alumno como dos directores.
- **Recursos software:** 0 €. Unicamente se utilizaron tecnoloxías de código aberto e de uso gratuito.
- **Recursos humanos:** Para o caso do custo asociado aos recursos humanos, consideráronse os salarios propostos polo [BOE](#) no día 6 de Marzo do ano 2018, no articulo "XVII Convenio colectivo estatal de empresas de consultaría, e estudos de mercado e da opinión pública" do Ministerio de Emprego e Seguridade Social, referencia [19]. Con isto, imos diferenciar entre os dous tipos de recursos humanos que xa mencionamos.
  - **Directores do proxecto:** considerouse un salario de 18,50 € por hora para á figura de director de proxecto, polo que o custo total asociado ao traballo dos dous directores é de **518 €**.
  - **Membro do equipo de desenvolvemento:** considerouse un salario de 16,20 € por hora para un desarrollador de software, dando como resultado un custo de **5.832 €**

Polo tanto, o custo total do proxecto sería de **6.350 €**. Ademais, podemos ver un resumo das desviacións en canto a datas e custos do noso proxecto na táboa [4.1](#).

	Estimado	Real
Data de inicio	26/01/2021	26/01/2021
Data de fin	21/05/2021	31/05/2021
Horas de traballo	350 horas	360 horas
Custo total	6.000 €	6.350 €

Táboa 4.1: Desviacións no proxecto.



## Capítulo 5

# Vista xeral da ferramenta

---

Neste capítulo realizaremos unha descripción xeral do sistema desenvolvido neste proxecto, analizando brevemente as diferentes compoñentes que o integran así como os requisitos que este debe cumplir. Nos seguintes capítulos faremos unha descripción máis detallada de cada una das partes do sistema.

### 5.1 Arquitectura do sistema

O obxectivo desta sección é ofrecer unha visión xeral da ferramenta que se busca desenvolver con este proxecto. Na figura 5.1, pode apreciarse un gráfico que resume os principais bloques ou módulos que conforman o sistema. Resulta interesante destacar que a arquitectura do sistema está organizada de forma modular, de tal xeito que as diferentes partes están desacopladas e poderían ser reemplazadas por outras coa mesma funcionalidade cun mínimo impacto sobre o resto do sistema.

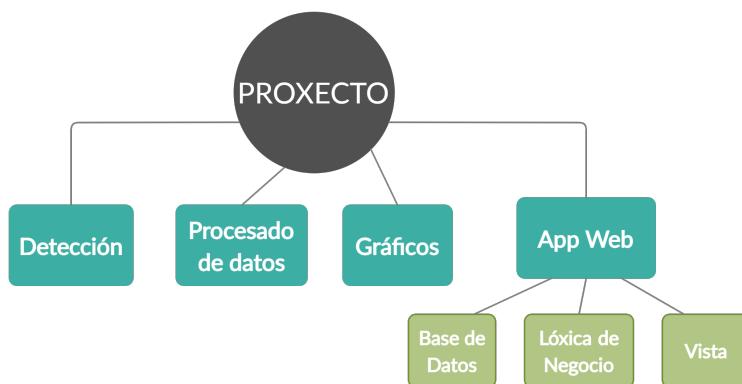


Figura 5.1: Overview do Proxecto

Como se pode ver, as principais compoñentes do proxecto serían as seguintes:

- **Detección:** elemento esencial na ferramenta que se encarga de realizar a detección dos xogadores no vídeo correspondente a un partido de pádel. Isto é a base de todo o proxecto, xa que debemos conseguir unha boa detección para levar a cabo un *tracking* dos xogadores. Probáronse distintas alternativas co obxectivo de conseguir que os datos proporcionados por este módulo foran precisos e nos permitiran obter as posicións dos xogadores na pista ao longo de cada un dos puntos dun partido.
- **Procesado dos datos:** componente encargada de procesar, almacenar e analizar os datos extraídos na detección, como poden ser as coordenadas dos xogadores ou a súa situación táctica nun determinado momento, entre outros datos. Para iso, debemos empregar estruturas de datos que se adapten tanto ao tipo de información extraída como ao noso sistema, así como realizar as transformacións e procesados necesarios para facilitar a súa representación gráfica.
- **Xeración de gráficos:** módulo responsable de xerar representacións visuais útiles a partir dos datos almacenados. Esta componente está deseñada para proporcionar gráficos que poden resultar de interese así como estatísticas referentes ao xogo e a colocación dos xogadores na pista. Esta información será logo presentada na vista da aplicación.
- **Aplicación Web:** parte do sistema encargada de presentar as distintas funcionalidades do noso software para que o usuario poida acceder a elas dunha forma cómoda e amigable. Deste xeito, a aplicación pode entenderse como unha plataforma na que os usuarios poden subir e analizar os seus propios vídeos, así como visualizar os resultados que máis lles interesen. Podemos facer a seguinte división en canto aos componentes da nosa aplicación:
  - **Base de datos:** utilizada para gardar a información non volátil relacionada co servizo que se está a ofrecer. Gardamos tanto a información sobre os usuarios que temos rexistrados como os datos dos seus vídeos tras ser procesados.
  - **Lóxica de negocio:** Conxunto de funcións e interfaces que definen o comportamento do sistema en función das interaccións e/ou dos datos proporcionados por un usuario. O resultado deste procesamento será logo representado na vista correspondente.
  - **Vista:** composta por toda a interface visual da aplicación. Representa o conxunto de pantallas e elementos gráficos cos que pode interactuar o usuario da aplicación.

## 5.2 Análise de requisitos

O termo análise aplicado a sistemas significa descompoñer dito sistema nas súas compoñentes para estudar cada unha delas, tanto como unha entidade illada como nunha interacción co resto de compoñentes [20][21][22]. Para que sexa útil, a análise debe seguir un proceso de síntese que consistirá en unir os compoñentes do sistema para determinar como funcionan en conxunto.

Podemos definir a análise de requisitos como o proceso de estudo das necesidades dos usuarios para chegar a unha definición dos requisitos do sistema, de hardware ou de software, así como o proceso de estudio e refinamento de ditos requisitos.

Un requisito é pois unha condición ou capacidade que necesita o usuario para resolver un problema ou conseguir un obxectivo determinado. Por extensión, o termo requisito aplica-se tamén as condicións que debe cumplir ou ter un sistema ou unha das súas compoñentes para satisfacer un contrato, unha norma ou unha especificación. Os requisitos dun sistema divídense en dous tipos, temos por un lado os **requisitos funcionais** e por outro os **non funcionais**.

### 5.2.1 Requisitos funcionais

Os requisitos funcionais dun sistema son declaracíons dos servizos ou funcións que proverá dito sistema. Exprésanse en termos de cal debe ser o comportamento do sistema nas distintas situacíons que poidan darse, así como da información que se vai manexar. Deben proporcionar unha descripción o suficientemente detallada para permitir o desenvolvemento e implementación do sistema. Unha xestión deficiente dos requisitos funcionais é citada como unha das causas más frecuentes polas que un proxecto acaba no fracaso. Por iso, é importante dedicarlle o tempo e esforzo necesario a especificación destes requisitos.

Tendo en conta isto, definíronse os seguintes requisitos funcionais para a nosa aplicación:

- **Procesado de vídeos:** o sistema debe recibir como entrada un determinado vídeo nun formato axeitado (mp4, .avi, ...) e ser capaz de procesar os seus *frames* para facilitar ou mellorar a detección dos xogadores. Neste caso, os vídeos de entrada corresponderán a partidos de pádel gravados cun plano fixo que, habitualmente, se usa neste deporte. Trátase dun plano xeral picado gravado desde detrás dun lado da pista que permite, dese xeito, cubrir toda a zona da pista. No entanto, este plano conleva certa distorsión xeométrica dos obxectos que aparecen nel que é necesario ter en conta na fase de detección e de procesado de datos.
- **Detección dos obxectos:** a aplicación debe levar a cabo unha detección dos xogadores que están sobre a pista de pádel de forma rápida e precisa.

- **Procesado de datos:** a aplicación debe levar a cabo todas as transformacións necesarias para que as coordenadas que obtemos dos xogadores poidan ser representadas correctamente en mapas e gráficos de dúas dimensíons. Ademais, deben analizarse os datos obtidos para extraer coñecemento útil para os xogadores como, por exemplo, a súa posición na pista (na rede, media pista ou en defensa), a coordinación co seu compaño/a ou detectar situacións tácticas desfavorables.
- **Representación dos datos:** a aplicación debe representar os datos que obtemos dunha maneira visual e intuitiva para o usuario a través de gráficas e estatísticas de interese.
- **Interacción co usuario:** o sistema debe dispoñer dunha interface web para que os usuarios poidan interactuar de forma sinxela con todas as funcionalidades que ofrece a aplicación.

### 5.2.2 Requisitos non funcionais

Son aqueles requisitos que non se refiren directamente ás funcións específicas que entrega o sistema, senón ás propiedades ou características deste, como poden ser a fiabilidade, a compatibilidade con outro software ou hardware, a facilidade de uso ou o tempo de resposta. Desta forma, os requisitos non funcionais que definimos para a nosa aplicación son os seguintes:

- **Usabilidade:** A aplicación ou produto final debe ser sinxelo de utilizar, cunha navegación intuitiva entre as distintas funcionalidades, apostando sempre pola simplicidade, sen elementos visuais innecesarios e cunha boa consistencia en canto ao estilo visual.
- **Dispositivos compatibles:** A aplicación debe ser accesible na maior cantidad de dispositivos posibles, sen que sexa necesario realizar instalacións ou configuracións de software adicional. Por iso, eliximos implementar unha aplicación web *responsive* accesible tanto desde dispositivos móbiles como en ordenadores.
- **Fiabilidade:** A aplicación debe proporcionar resultados correctos, entregando información fiable aos usuarios, sexa cal sexa o contido de entrada.
- **Axilidade:** Unha vez proporcionado o contido ou información de entrada, o noso sistema debe ser áxil en canto ao tempo que consume en producir a información de saída, buscando un tempo de resposta o menor posible.
- **Mantemento:** O software debe facilitar posibles adaptacións ou modificacións por se nun futuro se quere introducir algún tipo de modificación, sexa unha mellora sobre unha funcionalidade existente ou unha funcionalidade nova. Isto conséguese gracias a un deseño modular de toda a aplicación en bloques desacoplados.

### 5.3 Diagramas de fluxo

Un diagrama de fluxo é un diagrama de actividades mediante o cal se representa gráficamente algún proceso ou algoritmo. Neste caso, considerouse interesante elaborar dous diagramas de fluxo diferentes para facilitar unha visión xeral do sistema e ilustrar o funcionamento xeral da nosa aplicación.

Por un lado, temos o diagrama de fluxo da figura 5.2a, que representa a secuencia de accións e interaccións da parte correspondente á aplicación web. Para poder facer uso dos distintos servizos que esta ofrece, debemos dispor dunha conta de usuario. Unha vez temos a conta creada e iniciamos sesión, temos o seguinte abanico de opcións:

- A través de correo, contactar co responsable da páxina web en caso de que existira algún tipo de problema, ou mesmo para enviar calquera tipo de suxerencia ou mellora que o usuario considere interesante.
- Ver diferentes gráficos e estatísticas dos vídeos de pádel que o usuario subiu con anterioridade. O usuario pode seleccionar de forma intuitiva a información que quere visualizar en cada momento para un determinado partido.
- Eliminar un determinado vídeo da lista de vídeos do usuario.
- Seleccionar un novo vídeo que o usuario desexa ter na súa lista. Se o vídeo cumple coas condicións necesarias de formato, ademais de non estar previamente na lista do usuario, pásase a procesar dito vídeo para despois visualizar as distintas representacións visuais e estatísticas que se elaboraron.

Ademais, é de especial interese explicar en liñas xerais en que consiste o procesado dos vídeos. Para iso, elaborouse un diagrama de fluxo que pode verse na figura 5.2 e no que se mostran as principais accións que realiza o noso algoritmo. Podemos resumir o proceso da seguinte forma: procésanse de forma individual os *frames* que componen o vídeo, aplicándolle as máscaras para quedarnos coa información de interese e, a continuación, aplícase o algoritmo de detección para obter as posicións dos xogadores nun determinado *frame*. Se nun *frame* se detectan xogadores na pista de pádel, faise o procesado de datos correspondente para determinar e gardar as súas posicións. Unha vez terminado o procesado do vídeo, xéranse as representacións visuais e estatísticas a partir dos datos gardados de xeito estruturado durante o procesado.

Nos seguintes capítulos, explicaremos en detalle os principais bloques do sistema: detección de xogadores, procesado de datos e xeración dos gráficos/estatísticas, e a parte da aplicación web.

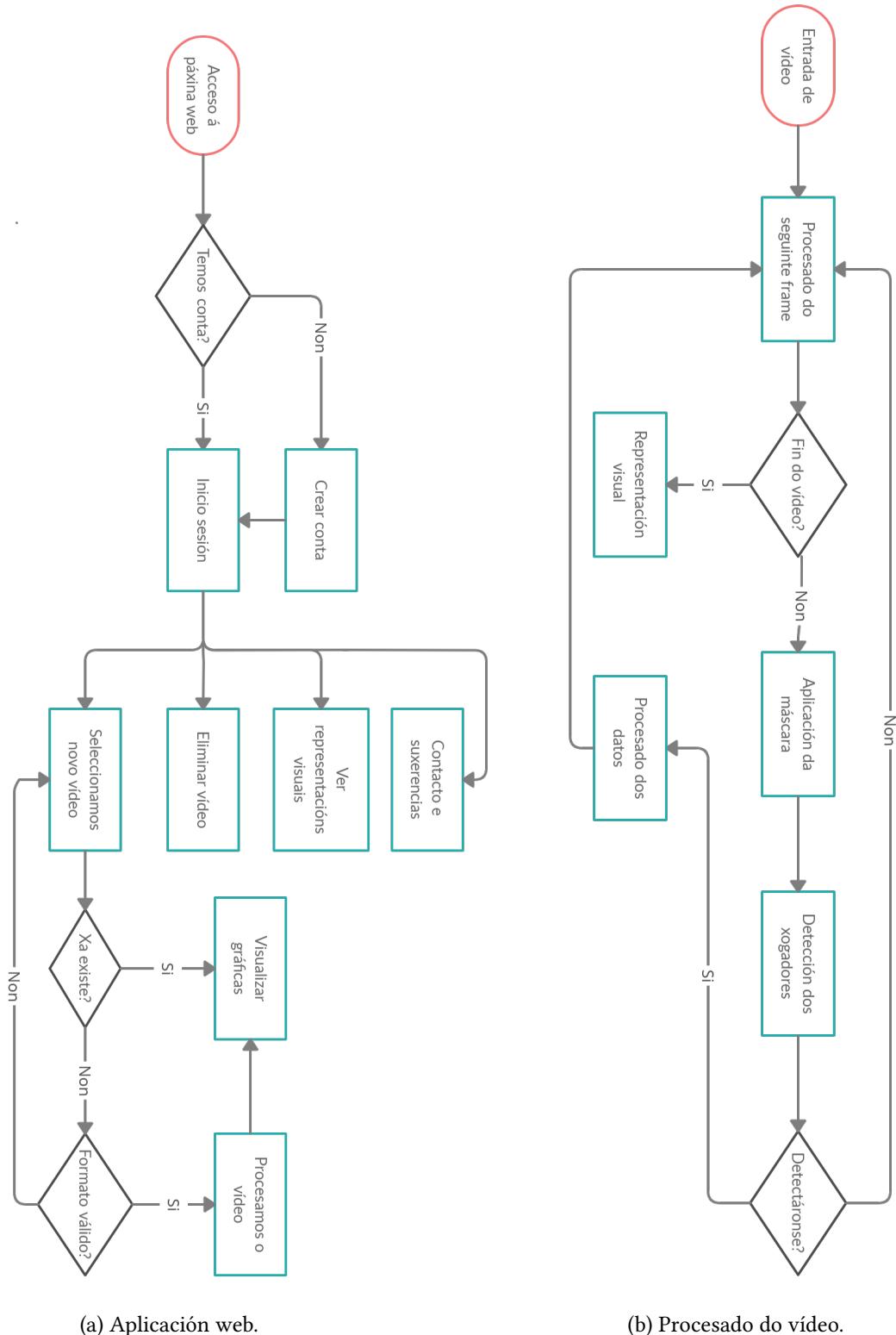


Figura 5.2: Diagramas de fluxo do noso proxecto.

## Capítulo 6

# Detección de xogadores

---

Como se comentou no capítulo anterior, o módulo encargado da detección dos xogadores no vídeo é chave tendo en conta que é fundamental obter uns datos precisos sobre as posiciones dos xogadores na pista para obter gráficos e estatísticas útiles e que representen correctamente o que aconteceu no partido. Ao longo do proxecto, probáronse varias alternativas para á detección dos xogadores nos *frames* dos vídeos. Neste capítulo, imos explicar as dúas aproximacións más relevantes por producir os mellores resultados de acordo as métricas empregadas.

O primeiro método combina [Histogram of Oriented Gradients \(HOG\)](#) para a extracción de características e un modelo lineal de [Support Vector Machines \(SVM\)](#). OpenCV conta cun modelo preadestrado de [SVM](#) baseado en [HOG](#) para detección de persoas en imaxes. O segundo método consiste no uso de [You Only Look Once \(YOLO\)](#), que está baseado nunha rede de neuronas artificiais.

### 6.1 Detección baseada en SVM e HOG

[SVM](#) é un algoritmo de aprendizaxe automático para a resolución de problemas de clasificación. É dicir, é unha técnica que nos permite determinar se un novo elemento de entrada pertence a unha clase ou a outra. Para iso, cada elemento de entrada que se quere clasificar debe ser representado por un vector de  $p$  características, as cales definen a súa pertenza a unha clase ou a outra.

Na súa esencia, [SVM](#) constrúe un hiperplano nun espazo  $p$ -dimensional que separa os elementos pertencentes a cada clase, de tal forma que a separación dos puntos do hiperplano aos elementos máis próximos de cada clase (vectores de soporte) é máxima. Desta forma, cando temos un novo elemento para clasificar, co seu correspondente vector de  $p$  características, este elemento pode ser representado como un punto no espazo  $p$ -dimensional e ver a que lado do hiperplano se sitúa, o que determina directamente a clase á que pertence.

Para determinar o hiperplano ou hiperplanos óptimos –no caso de haber máis de dúas clases–, é necesario un proceso de adestramento. Para iso, dispónse dun conxunto de adestramento etiquetado, de forma que os elementos deste conxunto están representados polos seus vectores de características e sábase a que clase pertenecen. Matematicamente, o proceso de adestramento consiste en resolver un problema de optimización para determinar os parámetros que definen o hiperplano, concretamente  $\omega$  e  $b$ , xa que para todos os puntos  $x$  do hiperplano cúmprese a seguinte ecuación:

$$\omega x^t + b = 0, \quad (6.1)$$

onde  $\omega$  indica a dirección do hiperplano e  $b$  o desprazamento respecto ao eixo de coordenadas. Na figura 6.1 podemos ver un exemplo para o caso dun problema de clasificación binaria (só hai dúas clases) nun espazo bidimensional, é dicir, con  $p = 2$ .

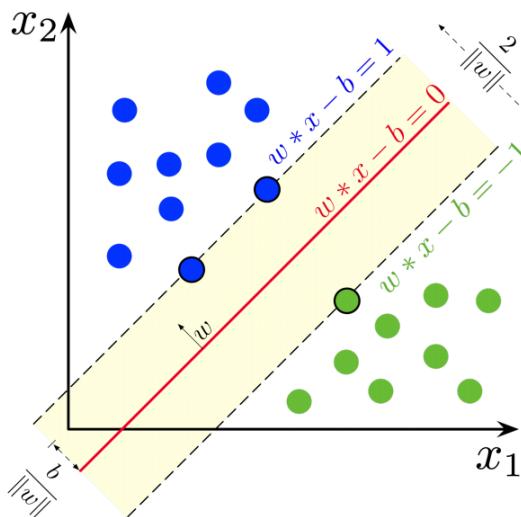


Figura 6.1: Clasificación de elementos usando SVM.

Hai dous factores decisivos para conseguir un funcionamento correcto no proceso de clasificación con SVM:

- Representar cada un dos elementos a clasificar mediante un vector de características adecuado que determine a súa pertenza a unha clase ou a outra coa menor ambigüidade posible.
- Adestrar correctamente o algoritmo, é dicir, construír o hiperplano, para minimizar os errores en fase de clasificación.

No noso caso, para a detección de persoas, debemos determinar para cada zona dunha imaxe

se esta se corresponde cunha persoa ou non. [SVM](#) é unha boa alternativa pois estamos ante un problema de clasificación binaria, pero é necesario dispoñer dun método que permita obter vectores de características representativas para os elementos dunha clase (persoa) e para os elementos doutra clase (non persoa). Isto pódese facer mediante un descriptor coñecido como é [Histogram of Oriented Gradients \(HOG\)](#), que foi proposto e adaptado para a clasificación de persoas, ademais de ser moi utilizado xunto con [SVM](#). A maneira na que traballa [HOG](#) é a seguinte:

- Partimos de imaxes RGB que se dividen en celas de 8x8. Elíxese dito tamaño pois está demostrado que, con este tamaño, [SVM](#) e [HOG](#) obteñen mellores resultados.
- Para cada una das tres compoñentes de cor da imaxe, calcúlanse os gradientes horizontais e verticais usando os kernels (matrices de convolución)  $[-1, 0, 1]$  e  $[-1, 0, 1]^T$ , respectivamente. Isto sinala, para cada píxel das celas, a dirección na que a variación da intensidade da cor é máxima e a magnitud desta variación.
- A continuación, para cada cela, calcúlase o histograma de gradientes considerando 9 *bins*. Estes *bins* correspóndense aos ángulos 0, 20, 40, ..., 160. A decisión do número de *bins* e os ángulos considerados corresponden coa proposta orixinal probada para o caso de detección de persoas. Neste caso, temos un histograma de gradientes orientados sen signo, xa que só se consideran ángulos entre  $0^\circ$  e  $180^\circ$ .
- Desta forma, para cada píxel da cela, mírase o ángulo correspondente ao gradiente e determinase a súa contribución para cada *bin*. Se o ángulo coincide exactamente cun ángulo representado por un *bin*, súmase toda a magnitud a ese *bin*. Se o ángulo coincide no medio de dous *bins*, súmase a parte proporcional da magnitud a cada *bin* en función da distancia aos ángulos representativos. Por exemplo, se a magnitud é 24 e o ángulo 35, sumaríase 6 ao *bin* correspondente a 20 e 18 ao *bin* correspondente a 40.
- O paso anterior produce un histograma con 9 valores para cada cela. O seguinte paso consiste en normalizar ditos valores para eliminar os efectos de iluminación e contraste, xa que o peso destes valores debería ser independente de se unha imaxe é máis escura ou máis clara. En lugar de facer unha normalización celta a cela, é preferible facela sobre un bloque de 4 celas contiguas.
- Desta forma, para cada bloque de 16x16 píxeles temos un vector de 36 gradientes orientados e normalizados. Utilízase unha xanela deslizante (16x16) que se vai desprazando horizontal e verticalmente. En cada paso desprázase 8 píxeles a dereita e cando chega ao final dunha fila, volve ao principio e desplázase 8 píxeles cara a abaixo, repetindo os pasos anteriores ata obter un vector de 36 elementos asociado a cada xanela (bloque). Ao final, concaténanse todos os vectores para formar un único vector de gradientes.

O uso de **HOG** para a detección de persoas foi amplamente investigado pola comunidade científica, chegando a un consenso sobre cales son os parámetros máis adecuados: subimaxes de 64x128, celas de 8x8, bloques de 16x16, 9 *bins*, gradientes sen signo, e desprazamento da xanela en 8 píxeles. OpenCV ofrécenos a posibilidade de construír un descriptor **HOG** por defecto que utilizará exactamente estes parámetros e que nos permitirá obter os vectores de características para cada área de 64x128 dentro dos *frames* do vídeo de entrada.

Polo tanto, **HOG** pode utilizarse para obter os vectores de características para alimentar o algoritmo de **SVM**, tanto na fase de adestramiento como na de clasificación. No noso caso, utilizaremos un clasificador **SVM** preadestrado específicamente para a detección de persoas a partir de vectores de características proporcionadas por **HOG**. Dese xeito, na fase de predición ou clasificación, podemos calcular para cada *frame* do vídeo os seus descritores **HOG** e detectar que zonas do *frame* se corresponden cos xogadores e cales non, usando este modelo preadestrado de **SVM**.

### 6.1.1 Aplicación no proxecto

Como xa adiantamos, no noso proxecto empregamos o detector incluido na librería de **OpenCV** que combina un modelo **lineal** de **SVM** con **HOG**. Para conseguir isto, apoiámosenos na función ***detectMultiScale*** de OpenCV, polo que imos explicar os seus parámetros e a información que nos vai aportar. Tendo en conta que debemos buscar un compromiso entre velocidade e precisión na detección, é importante analizar os efectos que se poden producir ao variar os parámetros da función. Os aspectos más importantes a ter en conta son os seguintes:

- Aumentar ou diminuir o número de deteccións de falsos positivos. Entendemos como falso positivo a detección dunha persoa na imaxe cando en realidade non existe dita persoa.
- Aumentar ou diminuir o número de falsos negativos, é dicir, non detectar unha persoa na imaxe cando si existe dita persoa.
- Aumentar ou diminuir a velocidade do proceso de detección.

Para analizar estes factores describiremos os **parámetros** más relevantes da función, incluíndo algúns exemplos:

- ***img***: este é o principal parámetro, xa que é a imaxe/*frame* sobre a que queremos detectar os xogadores. É o único argumento obligatorio para a función ***detectMultiScale***. A imaxe que pasamos pode ser tanto en cor como nunha escala de grises.

- ***winStride***: Aínda que é un parámetro opcional, ten unha grande importancia e debe configurarse correctamente, posto que ten unha repercusión moi alta non só na precisión do noso detector, senón tamén na velocidade de procesado.

O parámetro *winStride* é unha tupla de 2 elementos que define o “tamaño de paso” nos eixes x e y da xanela deslizante que se utiliza para recorrer a imaxe. En cada paso da xanela deslizante, extráense as características HOG da zona que ocupa, que son pasadas ao algoritmo [SVM](#) lineal para así clasificala. O proceso de extracción de características é bastante custoso, polo que é preferible avaliar a menor cantidade de xanelas posible se queremos buscar dar prioridade a velocidade na detección.

Canto máis pequeno é o *winStride*, máis xanelas deben avaliarse, o que pode traducirse nunha gran carga computacional. Por exemplo, o valor por defecto para este parámetro é de (8,8), o que supón un tempo medio de procesado dun *frame* do vídeo de 0.098 segundos. Mientras que se diminuímos o *winStride* a (4,4), aumenta o tempo de detección a 0.27 segundos.

De maneira similar, un valor máis grande para o *winStride* implica un menor número de xanelas a avaliar, polo que obteremos un tempo de execución máis pequeno. Mais, un valor demasiado grande provocará que nos perdamos deteccións do noso interese, é dicir, aumentará o número de falsos negativos. Na figura 6.2a, podemos ver un exemplo do que sucede cando aplicamos uns valores de (4,4) e (8,8), obtendo o mesmo resultado para ambas tuplas. No entanto, se subimos o valor da tupla a (16,16), aínda que obtemos un tempo de detección de tan só 0.07 segundos, perdemos certas deteccións, como podemos apreciar na figura 6.2b.

- ***padding***: Este parámetro é tamén unha tupla opcional que indica o número de píxeles nas direccións x e y cos que se rechea a [ROI](#) da xanela deslizante antes da extracción das características HOG. Como suxeriron Dalal e Triggs no seu articulo de 2005, “Histogram of Oriented Gradients for Human Detection” [23], incorporar unha pequena marxe ao redor da rexión de interese da imaxe antes da extracción e clasificación de características con [HOG](#) pode aumentar a precisión do noso detector. Os valores típicos son (8, 8), (16, 16), (24, 24) e (32, 32).
- ***scale***: É tamén un parámetro opcional. Para explicar este parámetro debemos introducir o concepto de pirámide de imaxes. Unha pirámide de imaxes é unha representación en múltiples escalas dunha imaxe, tal e como se pode apreciar na figura 6.3.

Este parámetro controla como cambia de tamaño a imaxe de entrada en cada nivel da pirámide. Esta circunstancia inflúe ademais no número de niveis que terá a propia pirámide. Unha escala máis pequena aumentará a cantidade de niveis nos que se descompón a imaxe, o que leva consigo un aumento do tempo necesario para a detección.

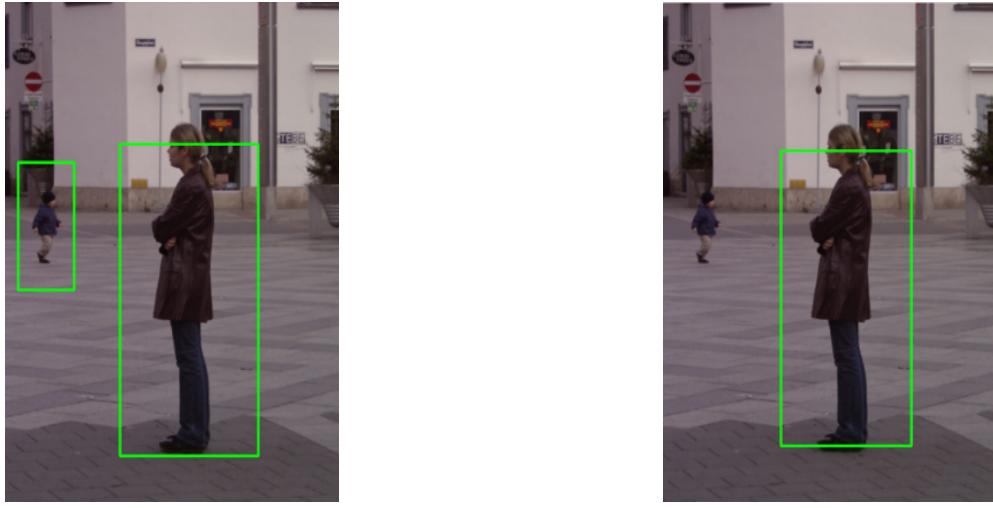


Figura 6.2: Deteccións con distintos valores para o parámetro `winStride`.

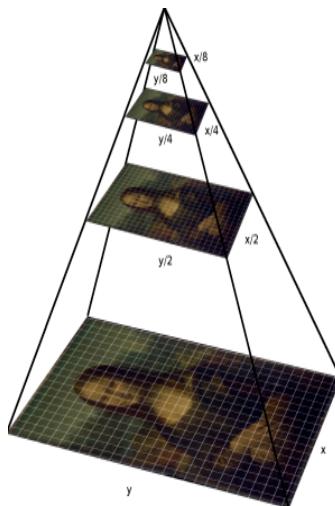
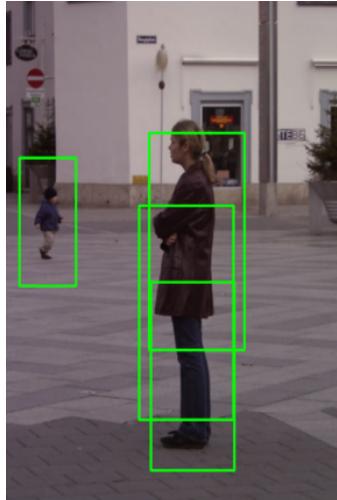


Figura 6.3: Exemplo de pirámide de imaxes.

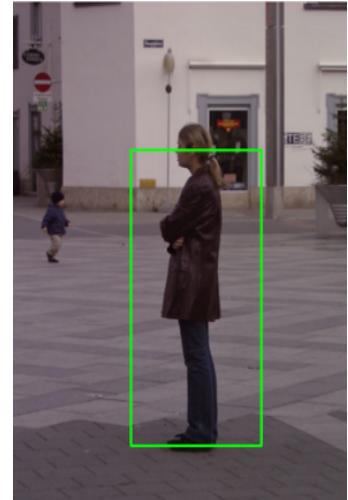
Por exemplo, cun valor para `scale` de 1.05, temos un tempo de 0.098 segundos, mentres que cun valor de 1.01 obtemos un tempo de 0.352 segundos. Ademais, como se aprecia na figura 6.4a, pode aparecer un problema de superposición dos cadros delimitadores da detección, áínda que este problema se pode solucionar de forma sinxela co seguinte parámetro que comentaremos.

Por outro lado, unha escala máis grande diminuirá a cantidade de capas da pirámide, así como o tempo de procesado necesario para a detección. Para un valor de 1.5, realizase a detección de persoas en só 0.02 segundos, co cal poderíamos procesar case 50 imaxes

por segundo. No entanto, isto conséguese a costa de pasar por alto un gran número de deteccións, como se pode apreciar na figura 6.4b.



(a) Exemplo de *Scale* con valor 1.01



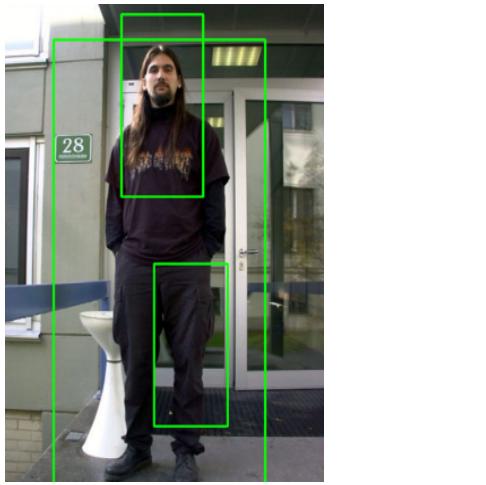
(b) Exemplo de *Scale* con valor 1.5

Figura 6.4: Deteccións con distintos valores para o parámetro *scale*.

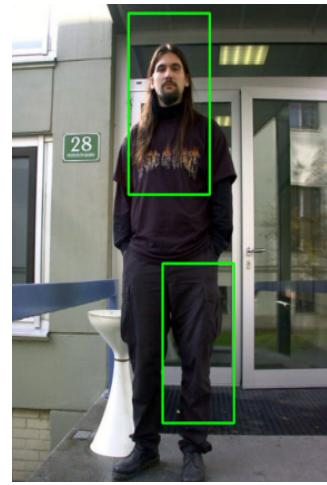
Os valores más comúns para este parámetro están no rango de [1.01, 1.5]. Debemos buscar o valor máis grande posible, sen sacrificar significativamente a precisión da detección.

- ***useMeanShiftGrouping***: É tamén un parámetro opcional. Este parámetro é un valor booleano que indica se se debe realizar “Mean Shift Grouping” ou agrupación de desprazamento para converter posibles cadros delimitadores superpostos nun único. Por defecto, ven cun valor predeterminado de *falso* e, nese caso, non se utiliza. Habilitar esta opción de corrección non deu bos resultados no noso proxecto, xa que xeraba múltiples cadros delimitadores para una mesma persoa ao realizar una detección, como podemos ver na figura 6.5. Para resolver este problema, probamos con outra alternativa e, no seu lugar, utilizamos **Non-Maximum Suppression (NMS)**. Ademais de ser más rápido, dá mellores resultados con deteccións finais moito más precisas, como podemos ver na figura 6.6.

Para concluír esta sección, imos explicar cal foi o procedemento para obter a configuración que se empregou neste proxecto para a detección con **SVM + HOG**. Elaborouse un *script* inicial co que se realizaron numerosas probas de detección. Este *script* recibe como entrada a imaxe sobre a que se quere facer a detección e o resto de parámetros anteriormente mencionados. Tras procesar a imaxe, o *script* danos como resultado unha imaxe na que aparecen representados os cadros delimitadores resultado da detección. Na figura A.1, podemos ver



(a) Exemplo de imaxe con varias deteccións superpostas



(b) Resultado de aplicar Mean Shift Grouping

Figura 6.5: Resultado de aplicar Mean Shift Grouping sobre varias deteccións superpostas.

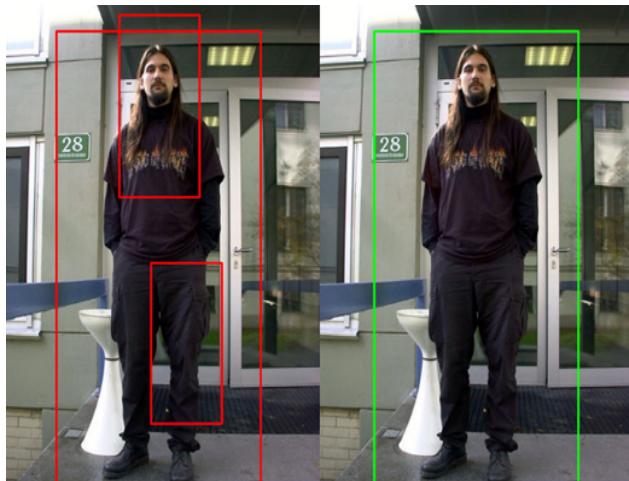


Figura 6.6: Resultado de aplicar NMS sobre varias deteccións superpostas.

dito *script*.

Tras varias probas, concluimos que a mellor configuración para o noso caso obtense cos seguintes valores:

- ***winStride***: con valor (4,4).
- ***padding***: con valor (8,8).
- ***scale***: con valor de 1.05.
- ***useMeanShiftGrouping***: non se usará este método, senón que se implementará [NMS](#).

### 6.1.2 Correción da detección

Con todo o explicado na sección anterior, non obtivemos un resultado completamente satisfactorio, pero conseguimos un punto de partida para comezar a traballar. Na figura 6.7, podemos ver un exemplo dos resultados que se obtiveron con esta aproximación e que ilustra algúns dos principais problemas que se detectaron. Entre eles, podemos destacar:

- Por un lado, temos **deteccións superpostas**, como podemos ver nos xogadores da parte inferior do campo.
- **Deteccións non interesantes** que, aínda que son correctas, non se corresponden co obxectivo de detectar aos xogadores como, por exemplo, o fotógrafo da parte inferior esquerda.
- **Deteccións erróneas**, como a da marxe da dereita ou o cubo de lixo da marxe esquerda.
- **Pouca precisión** acoutando aos xogadores. Vemos áreas de detección notablemente más grandes que os xogadores en sí, como ocorre no caso do xogador que está na parte superior esquerda, por exemplo.

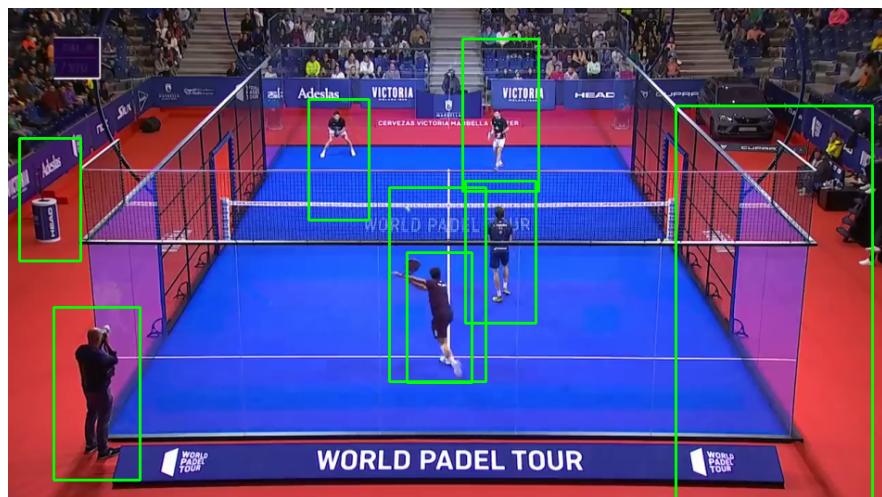


Figura 6.7: Primeira aproximación de detección sen melloras.

Para tratar de resolver istos problemas, aplicáronse unha serie de estratexas co obxectivo de mellor a precisión das deteccións e minimizar o número de falsos positivos:

- A primeira mellora que imos introducir será a aplicación da técnica **Non-Maximum Suppression (NMS)** para corrixir o problema das deteccións superpostas. Trátase dunha técnica utilizada en moitos algoritmos de intelixencia artificial. O seu obxectivo é seleccionar unha entidade, neste caso un cadro delimitador, entre outras moitas superpostas.

Os criterios de selección pódense elixir para adaptalos aos resultados que queremos. No noso caso, utilizamos a implementación que nos ofrece a librería *imutils*, onde debemos proporcionarlle como parámetro de entrada o *overlapThresh* ou rango de superposición, ademais do conxunto de cadros delimitadores. Na figura 6.8, podemos ver o resultado obtido tras aplicar dita técnica.

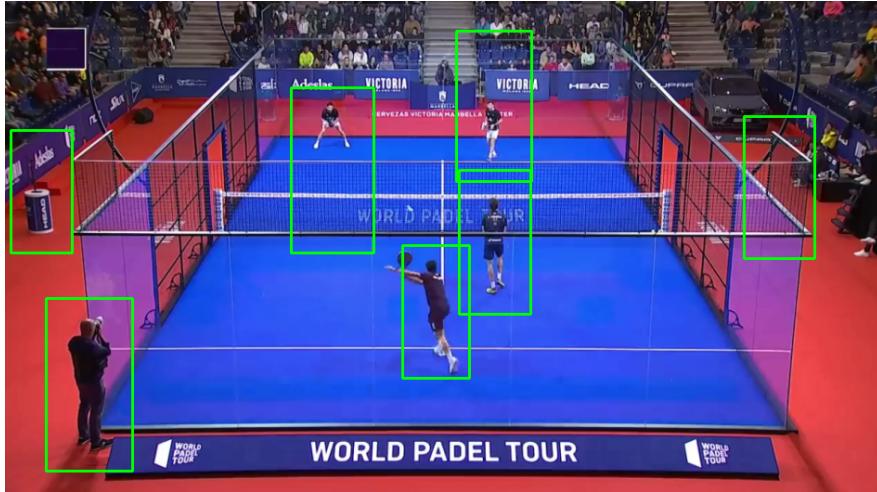


Figura 6.8: Resultado tras aplicar [NMS](#)

Como se pode ver, o funcionamento da [NMS](#) é correcto, pois eliminamos todos os cadros delimitadores superpostos. Mais, aínda debemos corrixir as deteccións erróneas ou falsos positivos que se producen no exterior da pista, ademais de intentar axustar mellor os recadros de detección, especialmente na parte superior da pista.

- Para afrontar o problema dos falsos positivos e mellorar os resultados tras aplicar [NMS](#), decidimos dividir o campo en dúas metades, así como eliminar todo o contido que non forma parte da pista, evitando así que persoas do público poidan ser detectadas como xogadores. Para iso, construímos dúas máscaras para quedarnos coa parte superior e coa parte inferior da pista, por separado. Ademais, realizamos un reescalado da imaxe, ampliándoa para ter os xogadores más facilmente recoñecibles. Resumindo, levamos a cabo o seguinte procedemento:
  1. Reescalado da imaxe. É importante facer un *resize* distinto segundo a zona da pista coa que estamos traballando. Para a parte superior, faremos unha maior ampliación para que os xogadores se vexan dun maior tamaño e a detección mellore.
  2. Construír un polígono co que delimitaremos cada zona de interese. Para iso, debemos crear unha matriz bidimensional das mesmas dimensións que as imaxes, na que indicaremos os vértices do noso polígono que se corresponden coas coor-

denadas dos puntos que limitan cada zona da pista.

3. Aplicar a máscara para seleccionar unicamente o contido que queda no interior do polígon. Aplicarase unha máscara para cada parte da pista (arriba e abaixo). Nas figuras 6.10 e 6.11, podemos ver as dúas máscaras aplicadas.

O código correspondente a istos pasos é mostrado na figura 6.9.

```
1 # reescalado da imaxe
2 frame = imutils.resize(frame, width = width)
3
4 # creamos a mascara
5 height,width,depth = frame.shape
6 polylines_frame = np.zeros((height,width), np.uint8)
7
8 # aplicamos a mascara
9 cv2.fillPoly(polylines_frame, pts =[pts], color=(255,255,255))
10 masked_data = cv2.bitwise_and(frame, frame, mask=polylines_frame)
```

Figura 6.9: Liñas de código coas que aplicamos a máscara e procedemos ao reescalado da parte superior e inferior da pista.

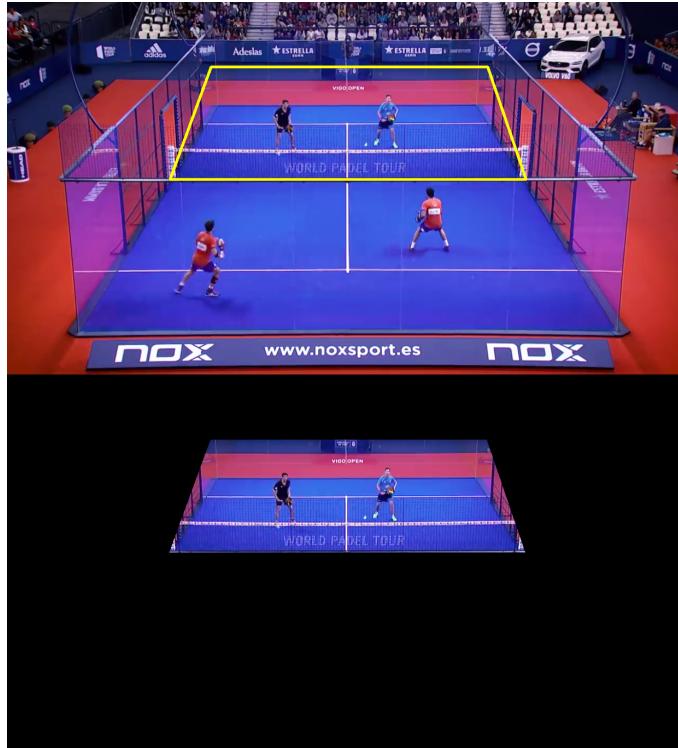


Figura 6.10: Aplicación da máscara para a zona superior da pista.

Con este procedemento melloramos sensiblemente as deteccións, sen apenas incrementar o custo computacional do módulo de detección. Nas figuras 6.12a e 6.12b, podemos



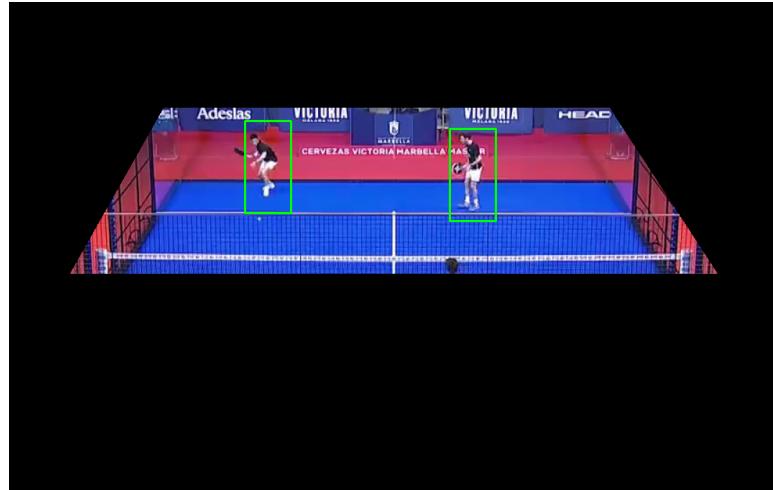
Figura 6.11: Aplicación da máscara para a zona inferior da pista.

ver o resultado de aplicar esta solución no proceso de detección dos xogadores.

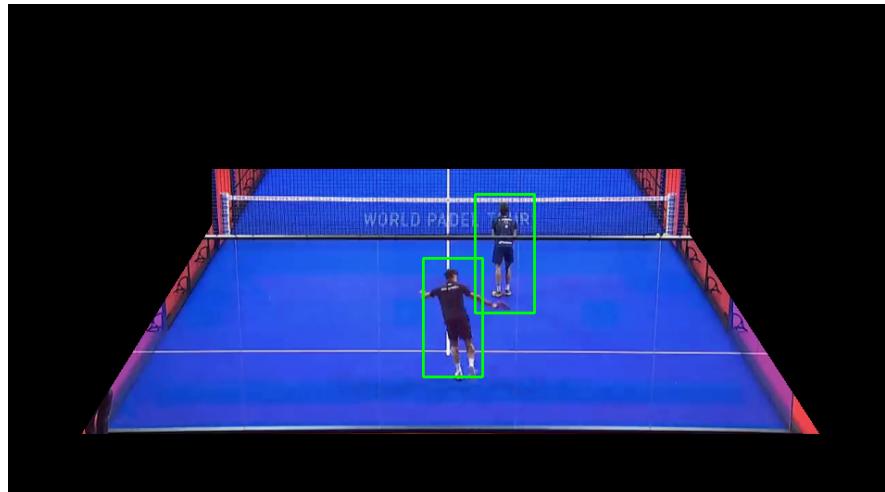
### 6.1.3 Análise dos resultados de detección

Para comprobar a utilidade da solución proposta para a detección de xogadores de pádel, resulta importante analizar a calidad das nosas deteccións e estudar o resultado de todo o proceso descrito. Para iso, introducimos unha serie de variables e métricas no noso código coas que buscamos avaliar os seguintes aspectos:

- Porcentaxe de *frames* nos que detectamos a un xogador respecto do total de *frames* que conforman o vídeo. Polo tanto, esta métrica é un bo indicador da precisión na detección xa que mide a proporción de deteccións positivas respecto do número total de deteccións.
- Porcentaxe de tempo efectivo no que detectamos ao xogador. Para calcular esta métrica, definimos unha resolución temporal dun segundo. Isto quere dicir que asumimos que un xogador é detectado de acordo a esta métrica se o módulo de detección nos devolve unha coincidencia positiva nalgún dos frames que componen cada bloque temporal dun segundo. Esta resolución temporal foi elixida tendo en conta que a posición dun xogador non varía significativamente dentro dun mesmo segundo.



(a) Zona superior da pista.



(b) Zona inferior da pista.

Figura 6.12: Detección coa máscara aplicada.

- Número de veces nas que se “perde” ao xogador. Consideramos que se “perde” a un xogador cando pasan máis de 2 segundos sen ser detectado.
- Número de veces nas que se producen máis de 2 deteccións simultáneas. Pode ser que a [NMS](#) non estea funcionando ben ou que a máscara que aplicamos necesite algún reaxuste, pois está detectando a xogadores da outra parte da pista ou a algunha persoa que non está participando no partido.

Tendo en conta estas métricas, faremos unha análise do funcionamento do método baseado en [SVM+HOG](#), considerando dous partidos de pádel distintos. Por un lado, utilizaremos un vídeo dun partido do *Cupra Vigo Open 2021*, ao cal nos referiremos baixo o termo de **Vi-**

go. Co obxectivo de obter unha información máis completa, analizaremos por separado cómo funciona a detección segundo a zona da pista coa que estamos a traballar. Pódense ver os resultados obtidos para as métricas consideradas nas táboas 6.1 e 6.2.

	Xogador Esquerda	Xogador Dereita
Nome do vídeo	Vigo	Vigo
Duración	20 min 20 s	20 min 20 s
Zona da pista	Inferior	Inferior
Frames por segundo (fps)	30	30
Frames totais	36.600	36.600
Frames detectados	33.892	33.086
Porcentaxe frames detectados	92,6 %	90,4 %
Porcentaxe tempo detectado	94 %	92,1 %
Número de perdas	3	4
+2 deteccións	24	24

Táboa 6.1: Datos obtidos para a zona inferior da pista, Cupra Vigo Open 2021

	Xogador Esquerda	Xogador Dereita
Nome do vídeo	Vigo	Vigo
Duración	20 min 20 s	20 min 20 s
Zona da pista	Superior	Superior
Frames por segundo (fps)	30	30
Frames totais	36.600	36.600
Frames detectados	15.811	15.299
Porcentaxe frames detectados	43,2 %	41,8 %
Porcentaxe tempo detectado	49,5 %	45,3 %
Número de perdas	22	27
+2 deteccións	18	18

Táboa 6.2: Datos obtidos para a zona superior da pista, Cupra Vigo Open 2021

Por outro lado, analizouse tamén un partido do *Estrella Damm Santander Open 2021*, ao cal nos referiremos baixo o termo de **Santander**. Os resultados obtidos ao analizar dito vídeo son mostrados nas táboas 6.3 e 6.4.

A partir dos resultados que se obtiveron, podemos extraer as seguintes conclusóns:

- O traballo da **NMS** é excelente, pois estamos eliminando as deteccións superpostas praticamente na súa totalidade. Ademais, as máscaras están facendo un bo traballo á hora

	Xogador Esquerda	Xogador Dereita
Nome do vídeo	Santander	Santander
Duración	21 min 40 s	21 min 40 s
Zona da pista	Inferior	Inferior
<i>Frames</i> por segundo (fps)	30	30
<i>Frames</i> totais	39.000	39.000
<i>Frames</i> detectados	36.075	37.011
Porcentaxe <i>frames</i> detectados	92,5 %	94,9 %
Porcentaxe tempo detectado	95,4 %	97,2 %
Número de perdas	2	1
+2 deteccións	9	9

Táboa 6.3: Datos da zona inferior da pista, Estrella Damm Santander Open 2021

	Xogador Esquerda	Xogador Dereita
Nome do vídeo	Santander	Santander
Duración	21 min 40 s	21 min 40 s
Zona da pista	Superior	Superior
<i>Frames</i> por segundo (fps)	30	30
<i>Frames</i> totais	39.000	39.000
<i>Frames</i> detectados	13.962	16.302
Porcentaxe <i>frames</i> detectados	36,8 %	41,8 %
Porcentaxe tempo detectado	41,9 %	44,6 %
Número de perdas	29	26
+2 deteccións	12	12

Táboa 6.4: Datos da zona superior da pista, Estrella Damm Santander Open 2021

de descartar á xente que pode haber nas gradas ou arredor da pista.

- Os resultados para a parte superior da pista son mellorables, xa que estamos detectando aos xogadores desa zona da pista nunha porcentaxe por debaixo do 50%. Debemos buscar alternativas e mellorar este aspecto.
- A pesar de que as porcentaxes de detección para os xogadores na parte de arriba é moi baixo, o número de veces que se “perden” é relativamente pequeno comparado coa duración total do vídeo. Isto implica que, a maioría de veces que non se detecta a un xogador nun determinado instante de tempo, ese xogador volve detectarse pasado un período de tempo relativamente curto (menos de 2 sg). Esta análise permítenos

concluír que sería posible interpolar con certa precisión as posicións que non temos dos xogadores por non ser detectados neses instantes de tempo.

## 6.2 YOLO

Tras a análise dos resultados que se obtiveron coa primeira aproximación baseada en SVM e HOG, chegamos a conclusión de que era necesario mellorar o proceso de detección. Despois de facer probas con distintas tecnoloxías decidimos, finalmente, utilizar You Only Look Once (YOLO) como segunda aproximación para a detección dos xogadores de pádel [24][25][26].

YOLO é un algoritmo que utiliza redes neuronais para proporcionar detección de obxectos en tempo real. Este algoritmo é bastante popular debido a súa velocidade e precisión. Para unha determinada imaxe, é capaz de recoñecer unha gran variedade de obxectos, proporcionando as probabilidades que teñen as deteccións de pertencer a unha determinada clase. Utilízase nun gran número de aplicacións como, por exemplo, para detectar sinais de tráfico, obxectos cotiáns, persoas, parquímetros, entre moitas outras cousas.

O algoritmo YOLO está baseado no uso de Redes Neuronais Convolucionais (CNN) para detectar os obxectos. Este tipo de redes son deseñadas para extraer aquellas características que nos interesan das distintas zonas da imaxe e que logo nos axudarán a clasificar que tipo de obxecto se atopa en cada una dasas áreas nas capas finais da rede. A partir das características que nos proporcionan as capas convolucionais da rede, pódese establecer a probabilidade de pertencer a cada unha das clases (tipos de obxectos) considerados. Como suxire o seu nome, realiza a predición ou detección sobre a imaxe nunha única execución do algoritmo. Isto supón unha importante vantaxe respecto outros clasificadores que necesitan realizar máis dunha “pasada” sobre a imaxe de entrada.

Actualmente, hai unha gran variedade de versións deste algoritmo. No noso caso, empregouse a versión YOLOv3, que pode ser integrada de forma sinxela para ser utilizada con OpenCV. Podemos resumir as súas capacidades e vantaxes nos seguintes puntos:

- **Velocidade:** como xa comentamos, con esta aproximación podemos detectar obxectos en tempo real, polo que a velocidade de detección vese mellorada notablemente respecto a outras alternativas similares.
- **Alta precisión:** estamos ante unha técnica predictiva que proporciona resultados precisos cunha marxe de erro mínima.
- **Capacidades de aprendizaxe:** o algoritmo ten excelente capacidades de aprendizaxe que lle permiten aprender as representacións dos obxectos, para aplicar ditos coñecementos na fase de detección.

O algoritmo **YOLO** necesita empregar tres técnicas ou métodos para realizar as deteccións, os cales son explicados nas seguintes seccións.

### 6.2.1 Residual blocks

É o primeiro paso do procesado que fai este algoritmo sobre unha imaxe de entrada. Consiste en dividir a mesma en varias cuadrículas, todas elas do mesmo tamaño. A figura 6.13 mostra un exemplo disto.

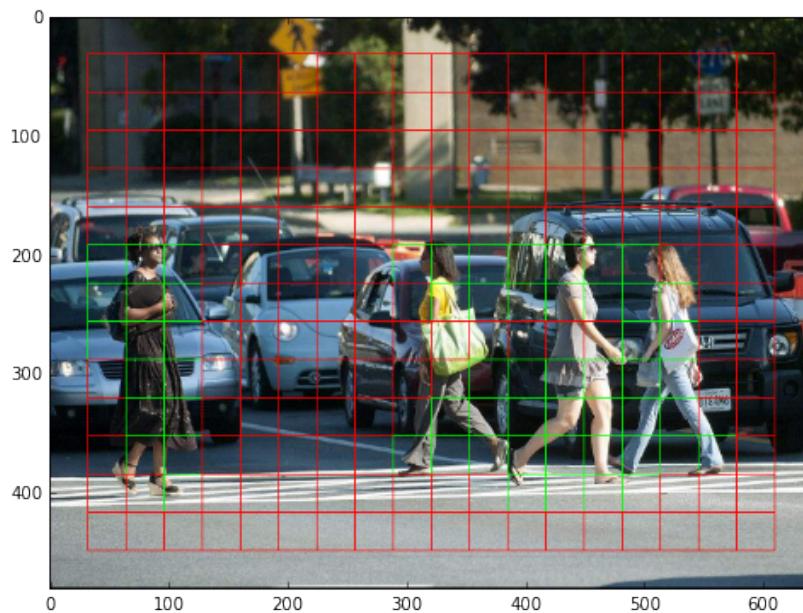


Figura 6.13: Exemplo das cuadrículas nas que se divide a imaxe.

Cada unha destas celas na que foi dividida a imaxe será a encargada de predicir os obxectos que aparecen nela. Por exemplo, se aparece o que se considera o centro dun obxecto dentro dunha determinada cela da cuadrícula, dita cela é a responsable de detectar o obxecto.

### 6.2.2 Bounding box regression

Podemos definir un cadro delimitador como o contorno que resalta o obxecto que estamos detectando nunha imaxe. Cada cadro delimitador consta dos seguintes atributos:

- Ancho ou  $b_w$ .
- Altura ou  $b_h$ .
- A clase á que pertence o obxecto como, por exemplo, persoa, automóbil, semáforo, bicicleta, especie de animal, etc. Está representada pola letra  $c$ .

- Probabilidade de pertencer á clase  $c$  ou  $p_c$ .
- Centro do cadro delimitador, representado polas coordenadas  $(b_x, b_y)$ .

Na imaxe da figura 6.14, móstrase un exemplo dos atributos que acabamos de comentar para o cadro delimitador marcado na imaxe cun contorno amarelo.

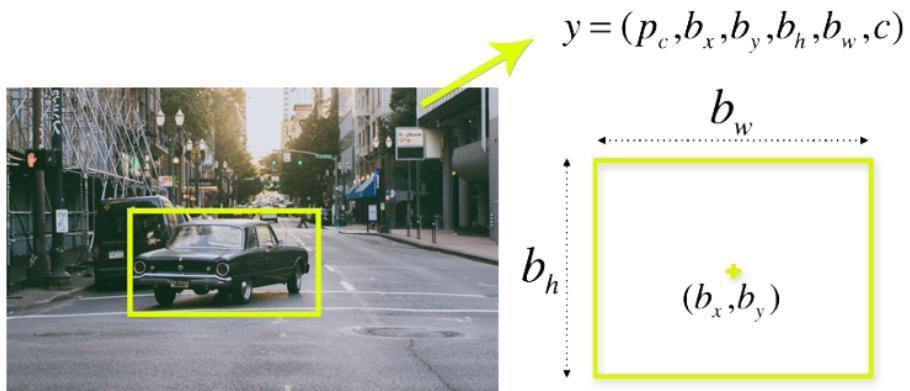


Figura 6.14: Exemplo de cadro delimitador

YOLO utiliza un modelo de regresión baseado nun só cadro delimitador para predecir a altura, o ancho, o centro e a clase á que pertence o obxecto que estamos detectando nunha determinada cela. Na imaxe de arriba, estamos representando coa variable  $y$  a probabilidade de que un obxecto que pertence a clase  $c$  se atope nese cadro delimitador.

### 6.2.3 Intersection Over Union (IOU)

A **Intersection Over Union (IOU)** é un fenómeno na detección de obxectos que describe cómo se superpoñen as diferentes caixas ou cadros descritos no apartado anterior. YOLO utiliza **IOU** para proporcionar unha única caixa de saída que rodea aos obxectos perfectamente.

Cada cela da cuadrícula é a responsable de predecir os cadros delimitadores e as súas puntuacións de confianza, é dicir, a probabilidade de pertencer a unha clase. O valor do **IOU** será igual a 1 se a caixa predicida pola cela responsable é igual ao cadro delimitador real. Deste xeito, esta técnica permite eliminar os cadros delimitadores que non son iguais ao cadro real. Polo tanto, estamos reemplazando en realidade o método da primeira aproximación de NMS por esta utilidade do algoritmo **YOLO**.

A seguinte imaxe (figura 6.15) proporciona un exemplo simple de cómo funciona esta técnica. Hai dous cadros delimitadores, onde o cadro azul é o predicho polo algoritmo, mentres que o cadro verde sería o cadro real. YOLO asegúrase de que o cadro delimitador final sexa o real.

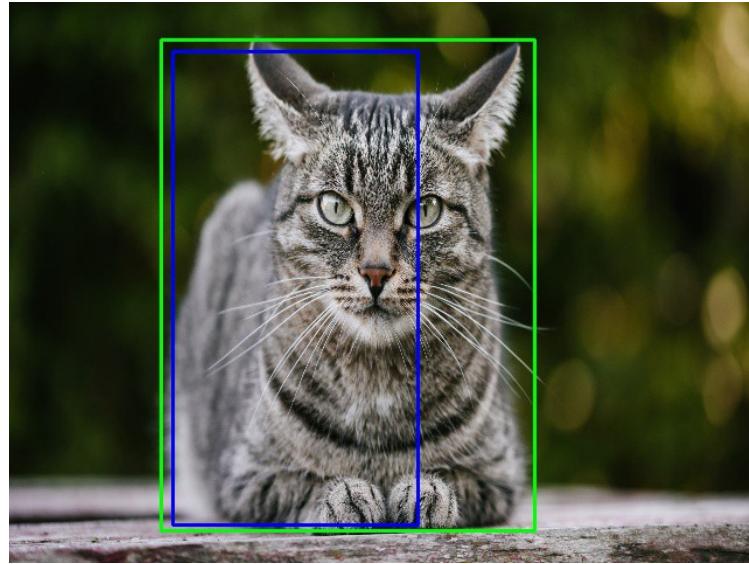


Figura 6.15: Exemplo do funcionamento da técnica IOU.

#### 6.2.4 Combinación das 3 técnicas

Nesta sección, vamos explicar brevemente como se combinan os tres conceptos anteriores para o recoñecemento de obxectos con [YOLO](#).

Primeiro, a imaxe divídese nunha cuadrícula formada por celas do mesmo tamaño. A continuación, cada cela predí os cadros delimitadores e proporciona as súas puntuacións de confianza. Ademais, tamén se asocian as clases ás que corresponde cada obxecto detectado. Por exemplo, na imaxe da figura 6.16, podemos apreciar polo menos tres clases de obxectos: un automóbil, un can e unha bicicleta. Todas as predicións realizanse simultaneamente utilizando unha única [CNN](#).

Por último, a [IOU](#) asegura que os cadros delimitadores predichos sexan iguais aos cadros delimitadores reais dos obxectos. Este fenómeno elimina os cadros innecesarios que non cumplen coas dimensións dos obxectos, como pode ser a altura ou o ancho. A detección final consistirá nos cadros delimitadores únicos que se axustan perfectamente a cada obxecto na imaxe.

Na imaxe da dereita da figura 6.16, pódese apreciar como o automóbil está rodeado por un cadre delimitador rosa, a bicicleta por un de cor amarelo, mentres que o can por un de cor azul, todos eles representando á perfección o tamaño real destes obxectos.

#### 6.2.5 Aplicación de [YOLO](#) no proxecto

Unha vez que entendemos como funciona o algoritmo [YOLO](#), podemos comentar o uso que lle damos no noso proxecto. Dado que o sistema foi implementado dunha forma modular,

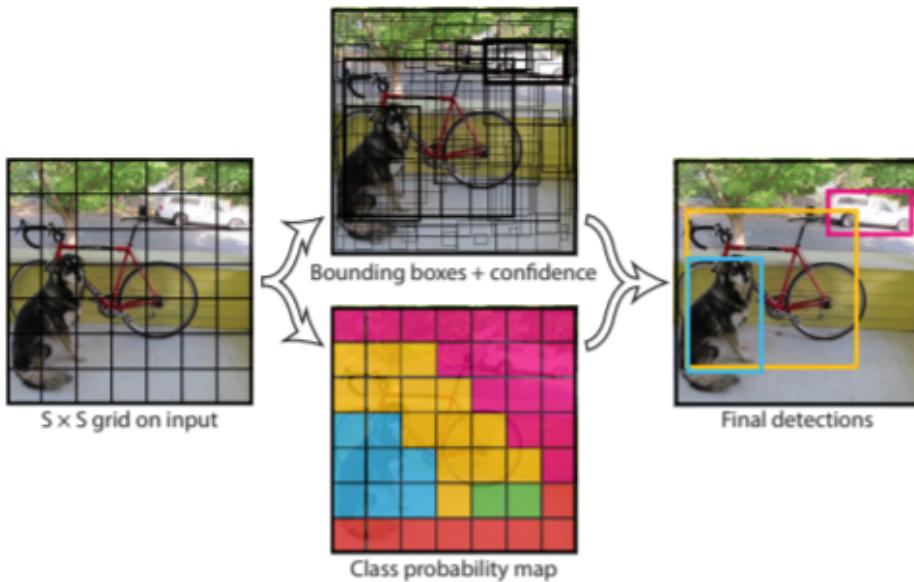


Figura 6.16: Exemplo de combinación das 3 técnicas comentadas.

podemos substituír o módulo encargado de facer as deteccións con **SVM** por un novo módulo que empregue **YOLO** para este propósito. Dado que **YOLO** se trata dun algoritmo de clasificación de propósito xeral e nos só estamos interesados en detectar os xogadores (persoas) na pista, simplemente temos que filtrar as saídas que nos proporciona **YOLO** en cada *frame* para quedarnos coas deteccións que pertencen á clase persoa cunha probabilidade suficientemente significativa. Na figura 6.17 podemos observar o resultado de aplicar **YOLO** para a detección de persoas nun partido de pádel. Cabe mencionar que se mantén o uso das máscaras explicadas na aproximación anterior para así centrar a atención sobre unha única zona da pista e lograr unha mellor detección.

### 6.2.6 Análise dos resultados da detección con **YOLO**

Para esta segunda aproximación, imos seguir o mesmo procedemento que no caso anterior co obxectivo de avaliar a calidade da detección. Faremos unha análise dos dous mesmos partidos que usamos co anterior algoritmo, para así poder comparar os resultados obtidos con cada unha das aproximacións.

Por tanto, primeiro temos o partido do *Cupra Vigo Open 2021*, ao cal nos referiremos baixo o termo de **Vigo**. Os resultados obtidos para este partido poden observarse nas táboas 6.5 e 6.6. O segundo vídeo corresponde ao partido do *Estrella Damm Santander Open 2021*, ao cal nos referiremos baixo o termo de **Santander**. Os resultados para este vídeo poden consultarse nas táboas 6.7 e 6.8.

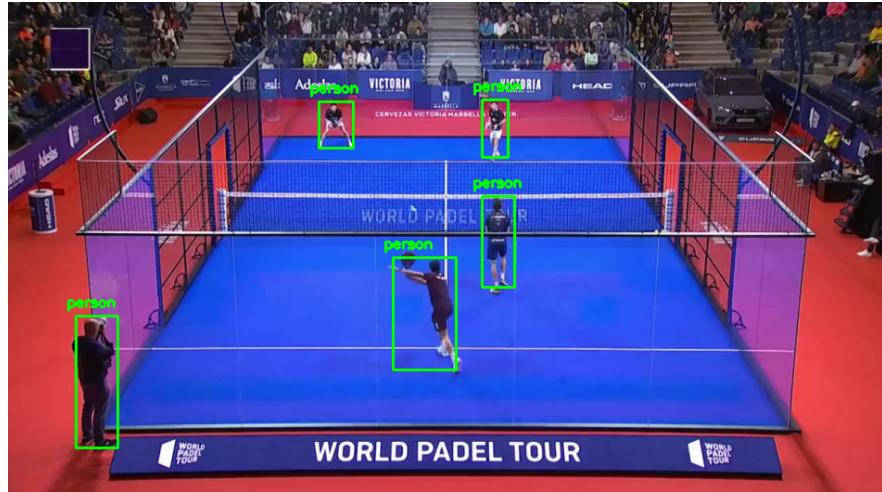


Figura 6.17: Detección de persoas utilizando YOLO

	Xogador Esquerda	Xogador Dereita
Nome do vídeo	Vigo	Vigo
Duración	20 min 20 s	20 min 20 s
Parella	1	1
Frames por segundo (fps)	30	30
Frames totais	36.600	36.600
Frames detectado	34.441	34.843
Porcentaxe frames detectados	94,1 %	95,2 %
Porcentaxe tempo detectado	99,2 %	98,6 %
Número de perdas	0	0
+2 deteccións	0	0

Táboa 6.5: Datos para a parella 1, Cupra Vigo Open 2021

O bo funcionamento do algoritmo YOLO queda reflexado na comparación entre as táboas 6.5–6.8 e as elaboradas a partir dos resultados obtidos coa aproximación baseada en SVM + HOG (ver táboas 6.1–6.4). Tendo en conta esta comparación, podemos extraer as seguintes conclusóns:

- Co anterior algoritmo, tiñamos unhas porcentaxes de detección relativamente baixas na zona superior da pista. Pola contra, ao empregar YOLO na etapa de detección, vemos unha gran mellora nas porcentaxes de detección, estando moi preto do 100% tanto na zona superior como na zona inferior da pista.
- Eliminamos por completo aquelas situacóns nas que había máis de dúas detección si-

	Xogador Esquerda	Xogador Dereita
Nome do vídeo	Vigo	Vigo
Duración	20 min 20 s	20 min 20 s
Parella	2	2
Frames por segundo (fps)	30	30
Frames totais	36.600	36.600
Frames detectado	35.612	35.465
Porcentaxe frames detectados	97,3 %	96,9 %
Porcentaxe tempo detectado	99,4 %	99,8 %
Número de perdas	0	0
+2 deteccións	0	0

Táboa 6.6: Datos para a parella 2, Cupra Vigo Open 2021

	Xogador Esquerda	Xogador Dereita
Nome do vídeo	Santander	Santander
Duración	21 min 40 s	21 min 40 s
Parella	1	1
Frames por segundo (fps)	30	30
Frames totais	39.000	39.000
Frames detectado	38.376	38.142
Porcentaxe frames detectados	98,4 %	97,8 %
Porcentaxe tempo detectado	100 %	99,6 %
Número de perdas	0	0
+2 deteccións	0	0

Táboa 6.7: Datos para a parella 1, Estrella Damm Santander Open 2021

multáneas nalgunha das zonas da pista ao aparecer demasiados cadros delimitadores. No caso de utilizar YOLO, vemos que, independentemente do vídeo e da zona da pista, non existen *frames* nos cales se obteñan máis de dous cadros delimitadores simultaneamente.

- En ningún partido perdemos a un xogador durante máis de 2 segundos.
- Tal e como podemos observar na figura 6.17, a detección dos xogadores é significativamente más precisa, conseguimos ter os xogadores mellor acoutados cos cadros delimitadores que con SVM + HOG (figura 6.8).

Considerando a importancia que ten detectar correctamente os xogadores na pista de pá-

	Xogador Esquerda	Xogador Dereita
Nome do vídeo	Santander	Santander
Duración	21 min 40 s	21 min 40 s
Parella	2	2
Frames por segundo (fps)	30	30
Frames totais	39.000	39.000
Frames detectado	38.532	38.181
Porcentaxe frames detectados	98,8 %	97,9 %
Porcentaxe tempo detectado	100 %	100 %
Número de perdas	0	0
+2 deteccións	0	0

Táboa 6.8: Datos para a parella 2, Estrella Damm Santander Open 2021

del, a decisión sobre que algoritmo empregar no módulo de detección do sistema está clara: **YOLO**.

### 6.3 Detección por parella de xogadores

Para poder asignarlle correctamente a cada xogador as súas posicións na pista durante un partido, é necesario detectar os cambios de campo que fan as parellas ao longo do partido. Tamén é importante destacar que, o xogador que na parte inferior xoga do lado dereito, ao cambiarse á parte superior da pista será o que xoga do lado esquierdo. Do mesmo xeito, o xogador que na parte inferior xoga do lado esquierdo, ao cambiarse á parte superior da pista será o que xoga do lado dereito. O mesmo ocorre cando o cambio se produce dende a zona superior á parte de abaxo da pista.

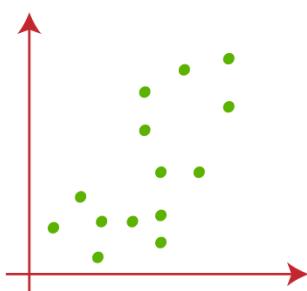
A forma máis natural de identificar que xogadores pertencen a mesma parella ou de diferenciar as dúas parellas que temos nun partido de pádel é pola cor da súa equipación que, por norma xeral, soen ser iguais para os dous xogadores que conforman unha parella de xogo. Desta forma, o seguinte paso despois de detectar aos xogadores nun *frame* será decidir a cal das dúas parellas pertencen cada detección.

Para levar a cabo esta tarefa, é necesario introducir o algoritmo de *K-Means* [27][28][29]. **K-Means** é un método de agrupamento ou *clustering*, é dicir, unha técnica para encontrar e clasificar  $K$  grupos de datos (clusters), de maneira que os elementos que comparten características semellantes estarán xuntos no mesmo grupo, separados dos grupos cos que non comparten características. Os algoritmos de *clustering* son considerados de aprendizaxe non supervisado. Este tipo de algoritmos buscan patróns nos datos sen ter unha predición es-

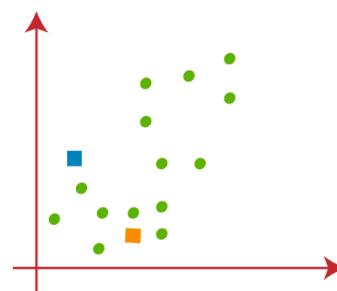
pecífica como obxectivo, é dicir, non existe unha clase concreta que debamos especificar. Describimos o funcionamento xeral de *K-Means* da seguinte forma:

- Inicialmente, debemos especificar o número de *clusters* ou grupos nos que imos segmentar os nosos datos. Supoñamos que dito número é  $k$ .
- Unha vez temos o número de *clusters*, o algoritmo establece de forma aleatoria  $k$  centroides, é dicir, os puntos que representan as posicíons centrais de cada grupo. Con isto, cada dato do conxunto a analizar é asignado ao centroide máis próximo, co que comparte más características.
- Calcúlanse de novo os centroides. Esta vez, calculando o punto medio dos datos que temos xa asignados a cada grupo ou *cluster*.
- Como os centroides cambiaron, débese facer unha nova asignación dos datos, xa que algún deles serán agora más semellantes a outro centroide.
- Séguese o proceso iterativo do cálculo de novos centroides coa correspondente asignación dos datos, ata o punto no que ditos centroides non varían. Deste xeito, temos como resultado unha clasificación final que maximiza a distancia entre os centroides e minimiza as diferenzas entre os datos que pertenecen ao mesmo *cluster*.

Para comprender o proceso explicado anteriormente, apoiarémonos nun exemplo visual. Supoñamos que temos o conxunto de datos representado na figura 6.18a, establecendo o número de *clusters* a dous e tendo uns centroides iniciais aleatorios como os representados na figura 6.18b mediante cadrados.



(a) Conxunto de datos



(b) Centroides aleatorios iniciais

Figura 6.18: Conxunto de datos inicial xunto cos primeiros centroides

Agora, asínñase cada punto ao seu centroide máis próximo, co que se supón que comparte más características. Esta primeira clasificación pode verse na figura 6.19.

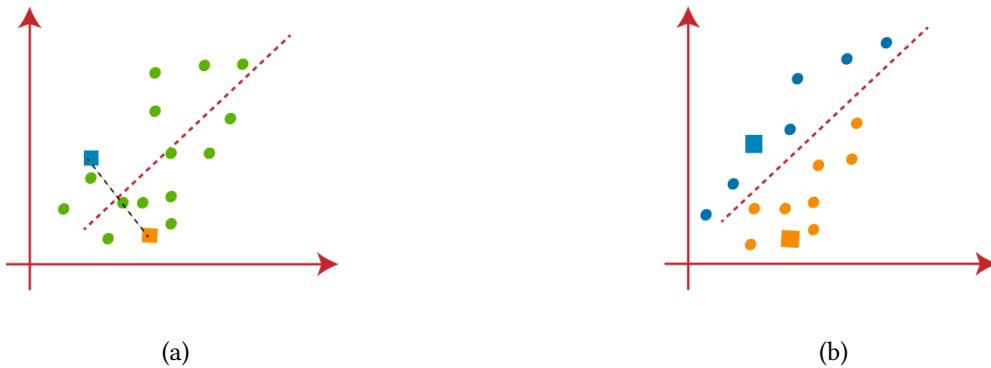


Figura 6.19: Primeira asignación dos puntos aos centroides

A continuación, calcúlanse os novos centroides determinando cal é o punto medio de cada *cluster*, formado polos anteriores centroides e os puntos asociados (figura 6.20a). Como vemos na figura, 6.20b temos puntos que deben asociarse a un *cluster* novo, concretamente, dous puntos pertencentes á cor azul e un punto pertencente á cor vermella.

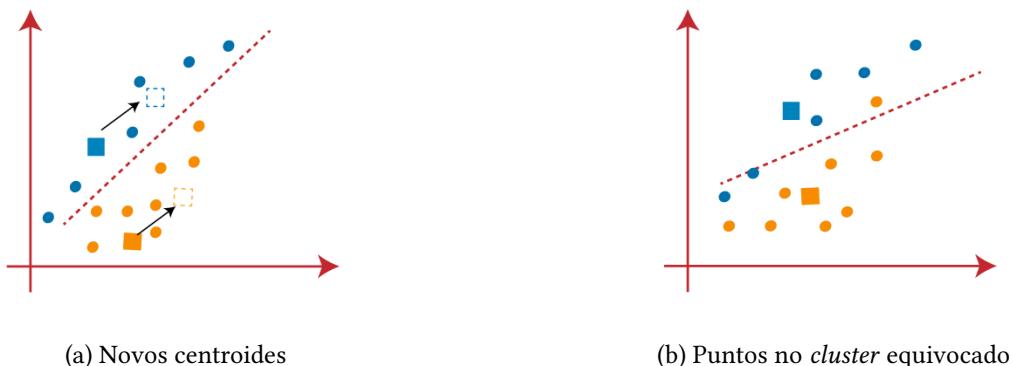


Figura 6.20: Cálculo dos novos centroides e visión dos puntos

Podemos ver a nova asignación na figura 6.21a e, novamente, o cálculo dos próximos centroides na figura 6.21b.

Despois de varias iteracións, obtemos o modelo final cos dous *clusters* xa establecidos (figura 6.22).

A continuación, vamos explicar como podemos aplicar este algoritmo para a detección das parellas nos vídeos de pádel. No noso proxecto, o conxunto de datos estará formado polas intensidades dos píxeles dunha imaxe RGB. Dada unha imaxe de tamaño  $M \times N$ , temos  $M \times N$  píxeles, cada un deles con tres componentes RGB: vermello, verde e azul, respectivamente. Os píxeles que pertenecen a un mesmo grupo serán similares en canto á súa cor, mentres que os píxeles que pertenecen a distintos grupos serán dunha cor significativamente diferente.

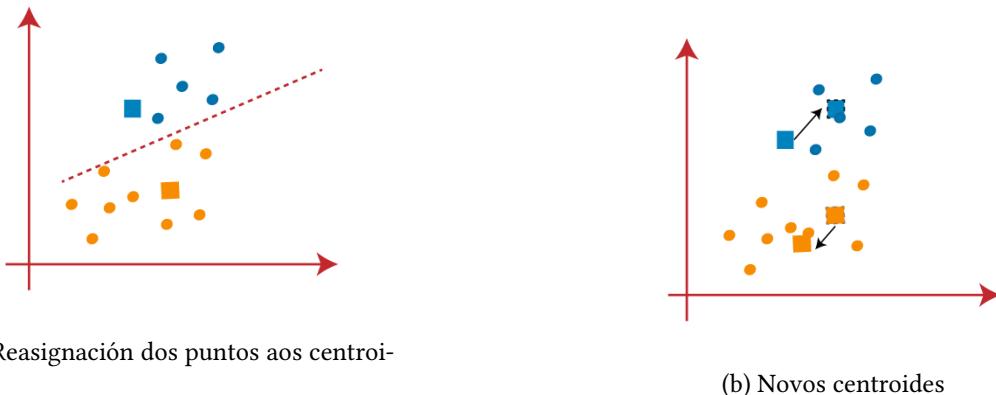


Figura 6.21: Reasignación de puntos e cálculo dos novos centroides



Figura 6.22: Modelo final

Como temos as coordenadas dos cadros delimitadores dos xogadores, información dada polo algoritmo de detección, únicamente temos que recortar esa sección da imaxe e procesala co algoritmo de *K-Means*.

Os mellores resultados veñen dados mediante o uso de 4 *clusters*. O que nos interesa é obter a cor da equipación da parella, pero a cor da camiseta pode ser diferente á cor do pantalón. Ademais, tamén se verá reflexada a cor da pista. Xa temos un total de 3 *clusters* para as cores mencionadas. Reservamos un *cluster* a maiores para cores que poidan aparecer como son as das liñas do campo, calzado dos xogadores, calcetíns, entre outras cousas.

Nas figuras 6.23 e 6.24, analízanse as cores dominantes dun xogador da parella 1 e dun xogador da parella 2 con *K-Means*, respectivamente. Facendo unha análise das cores dominantes que nos da como resultado o algoritmo, asignámosselle cada detección a unha parella ou a outra. Como se pode apreciar nas imaxes, a cor que aparece na porcentaxe máis alta soe ser a correspondente á cor da pista, polo cal a podemos descartar. Os *clusters* restantes danno as cores da camiseta e do pantalón que forman a equipación do xogador, polo que a combinación

das dúas seguintes cores dominantes danos a información da parella á que pertence o xogador detectado.



(a) Recorte a analizar



(b) Cores dominantes

Figura 6.23: Xogador correspondente á parella 1



(a) Recorte a analizar



(b) Cores dominantes

Figura 6.24: Xogador correspondente á parella 2

Nas figuras [A.2](#), [A.4](#) e [A.3](#), preséntase o código no que nos apoiamos para a análise das cores dominantes a través de *clustering* con *K-Means*.



## Capítulo 7

# Procesado dos datos e Gráficos

---

Unha vez feita a detección dos xogadores, debemos realizar o procesado destes datos e as transformacións xeométricas necesarias para extraer información de interese. Neste capítulo explicaremos os procesos que se seguiron para acadar esta tarefa, ademais mostraremos as representacións visuais xeradas ao representar ditos datos e as súas estatísticas.

### 7.1 Procesado de datos

Co proceso de detección explicado no capítulo anterior, obtemos a posición dos xogadores, representada por un conxunto de catro valores ( $x, y, w, h$ ). Explicaremos a qué corresponde cada variable con respecto aos cadros delimitadores que nos devuelven os algoritmos de detección descritos.

- **x** representa o valor da coordenada no eixe horizontal. En relación ao cadro delimitador, fai referencia á coordenada horizontal da esquina superior esquerda.
- **y** representa o valor da coordenada no eixe vertical. En relación ao cadro delimitador, fai referencia á coordenada vertical da esquina superior esquerda.
- **w** representa o ancho do xogador detectado. En relación ao cadro delimitador, fai referencia o seu ancho.
- **h** representa a altura do xogador detectado. En relación ao cadro delimitador, fai referencia a súa altura.

Ao analizar un vídeo estamos procesando unha gran cantidade de *frames*, non unha soa imaxe. Polo tanto, debemos almacenar os datos que imos procesando para cada *frame*. Python ofrécenos unha gran cantidade de opcións á hora de elixir a nosa estrutura de datos, coa posibilidade incluso de utilizar librerías externas. No noso caso decidimos gardar a información

en diccionarios. Un dicionario é unha estrutura de datos que almacena pares chave-valor, sendo as chaves únicas dentro dun mesmo dicionario e de calquera tipo inmutable. Como valor podemos almacenar tamén calquera tipo de datos, sexan enteiros, cadeas, listas, etc. De todas as propiedades e características que os definen, podemos destacar as seguintes tres como a principal razón de utilizar dita estrutura:

- Mellora a lexibilidade do noso código. Escribir as claves xunto cos seus valores agraga unha capa de documentación ao código. Se o código está más simplificado, temos unha depuración más sinxela, polo que a análise realizaranse moito más rápido.
- Velocidade de acceso a datos. Buscar unha clave en concreto nun dicionario é bastante eficiente en comparación coa procura dun elemento nunha lista grande e lineal, pois realizañase nun tempo constante. Para buscar un elemento nunha lista enorme, estaremos revisando todos os elementos desa lista ata atopar o que queremos. Se a cada elemento lle asignamos un par clave-valor, solo necesitamos acceder a dita clave.
- Gran variedade de métodos que temos disponíveis para realizar distintas operacións básicas. Como *items()*, *keys()*, *zip()*, *dict()*, entre moitos outros.

No noso proxecto, únicamente imos almacenar un par de valores  $(x, y)$  que representen o punto medio da base do rectángulo debuxado arredor dos xogadores. Para iso, non temos máis que calculalo da seguinte forma:  $(x + w/2, y + h)$ . Ademais, dado que no pádel os movementos realizados son de distancias cortas e moitos dos golpes realizañanse nas mesmas posicións, o noso sistema soamente almacenará unha coordenada  $(x, y)$  por segundo para cada un dos xogadores, xa que neste período de tempo as súas posicións non cambian dunha forma significativa. O procedemento que seguimos para almacenar os datos do vídeo é o seguinte:

- Creamos un dicionario para cada xogador. Inicializamos os dicionarios con tantas chaves como segundos teña o vídeo a analizar, deixando o seu valor baleiro nun principio.
- Segundo imos procesando o vídeo, para cada detección calculamos o par de coordenadas  $(x, y)$ . Tamén debemos calcular o segundo do vídeo ao que pertence o *frame* que estamos a procesar, para así almacenar as coordenadas na chave correcta do dicionario.
- Unha vez que temos a totalidade do vídeo procesado, as chaves do dicionario terán todas as posicións nas que estivo o xogador nese segundo. Como o noso obxectivo é quedarnos únicamente cunha coordenada por segundo, reducimos as coordenadas que temos para cada chave a un solo par  $(x, y)$ . Para isto, calculamos o punto medio de todos os pares.

## 7.2 Transformación xeométrica

Como xa comentamos no capítulo dos fundamentos tecnolóxicos, os vídeos que imos analizar presentan un plano constante, como pode observarse na figura 3.1. Polo tanto, áinda que as coordenadas que obtemos son precisas e representativas, terán o valor correspondente a dito plano. O noso obxectivo é facer una representación destes datos nun mapa 2D da pista de pádel, polo que debemos levar a cabo unha transformación de perspectiva. Como o seu nome indica, esta transformación está asociada cun cambio do punto de vista. Este tipo de transformación non conserva o paralelismo, a lonxitude nin o ángulo. Pero sí se conserva a colinealidade e a incidencia. Isto significa que as liñas rectas permanecen sempre rectas, sexa cal sexa o punto de vista resultante da transformación. Podemos expresar a transformación de perspectiva como na figura 7.1 [30].

$$\begin{bmatrix} t_i x' \\ t_i y' \\ t_i \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

↓  
*Scaling Factor*
↓  
*Transformation Matrix (M)*

Figura 7.1: Operación para realizar a transformación de perspectiva dunha coordenada.

Por un lado,  $(x', y')$  son os puntos transformados, mentres que  $(x, y)$  son os puntos orixinais. Como podemos ver na imaxe, necesitamos dunha matriz para que, multiplicada polos puntos orixinais, nos de como resultado o punto transformado. Se analizamos as variables que componen a matriz de transformación, temos que  $\begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix}$  define a transformación que se vai levar a cabo, sexa unha rotación, un escalado, unha transformación de perspectiva, entre outras. O vector de translación vén dado por  $\begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$  e vector de proxección por  $\begin{pmatrix} c_1 & c_2 \end{pmatrix}$ . Aplicando dita ecuación, buscaremos facer unha transformación de perspectiva, a cal vese representada na figura 7.2[30].

Para calcular dita matriz, utilizaremos o método `getPerspectiveTransform` de OpenCV. A este método, debemos proporcionarlle como entrada 2 conjuntos de 4 puntos, dos cales, 3 deles non poden ser colineais, é dicir, que se atopen na mesma liña.

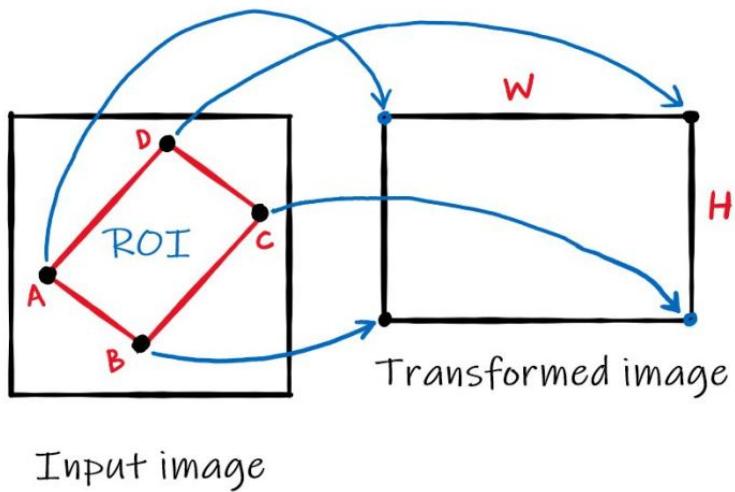


Figura 7.2: Representación da transformación de perspectiva dunha ROI

O primeiro conxunto debe conter os puntos orixinais que describen o plano que queremos transformar. No noso caso, será o plano que conteña a pista de pádel, podemos aprecialo na figura 7.3. O segundo conxunto debemos configuralo dándolle o tamaño que queremos que teña a pista no mapa 2D. Por exemplo, podemos construír unha pista de pádel de  $300 \times 600$  píxeles. Esta sería unha medida adecuada, xa que respetaría as dimensións reais da pista, de  $10 \times 20$  metros. Polo tanto, ambos conxuntos definiranse así no noso proxecto:

```
conjunto1 = np.float32([[297,150],[655,150],[845,465],[100,465]])
```

```
conjunto2 = np.float32([[0,0],[300,0],[300,600],[0,600]])
```

Unha vez que temos a matriz calculada, podemos utilizarla para realizar as transformacións dos puntos da imaxe que nos resulten de interese, como, por exemplo, a posición dos xogadores. Para facernos unha idea visual do resultado, realizamos a transformación da imaxe na figura 7.3. Para iso, utilizaremos a función *warpPerspective* tamén de OpenCV. Debemos proporcionarlle como entrada a matriz calculada no paso anterior, xunto coa imaxe orixinal e o tamaño da imaxe que queremos obter. Podemos ver a imaxe resultante na figura 7.4.

Vemos que o resultado da transformación é o que esperábamos, pois transforma o noso plano orixinal nunha imaxe de  $300 \times 600$  píxeles conservando a posición dos xogadores e as liñas. Con isto, procederemos a transformar os puntos que temos para cada segundo de vídeo da seguinte forma:

- Multiplicamos o noso punto  $(x, y)$  que representa a posición do xogador no plano orixinal (figura 7.3) pola matriz que calculamos co método *getPerspectiveTransform*, in-

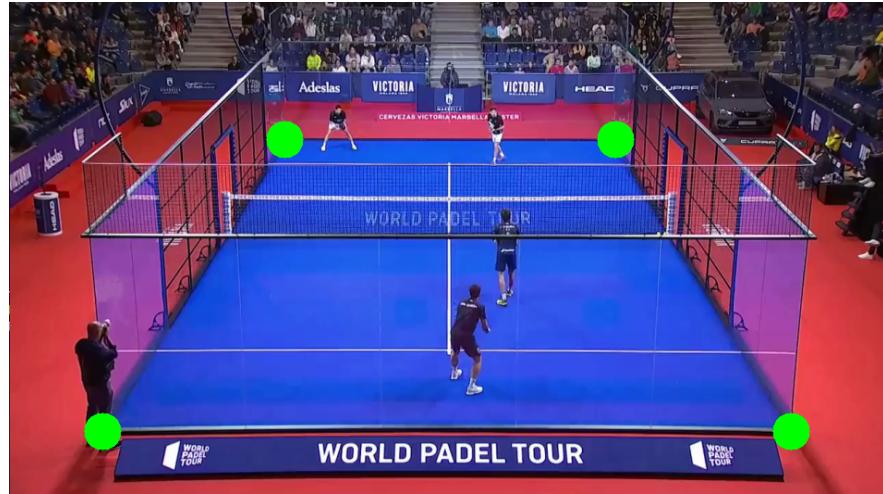


Figura 7.3: Puntos que recollen o plano que imos transformar.



Figura 7.4: Imaxe tras a transformación de perspectiva.

cluíndo unha terceira coordenada  $z$ , con valor 1. Debemos facer isto, pois a matriz de transformación ten un tamaño de  $3 \times 3$ .

$$p3 = \begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix} = Matriz * \begin{bmatrix} x & y & 1 \end{bmatrix} \quad (7.1)$$

- Dividimos o punto resultante da anterior operación pola terceira coordenada do propio punto, para así quedarnos co punto transformado  $(x', y')$ .

$$p2 = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x''/z'' & y''/z'' \end{bmatrix} \quad (7.2)$$

Podemos ver o código do que nos axudamos para realizar a transformación xeométrica na

figura A.5.

## 7.3 Gráficos

Unha vez que temos as posicións de cada xogador para cada segundo do partido transformados ao plano 2D que desexamos, podemos elaborar distintas representacións visuais dos datos e das súas estatísticas. Nas seguintes seccións faremos una descripción de cada unha delas.

### 7.3.1 Gráficos individuais

Neste tipo de gráficos elaboramos estatísticas individuais referentes a cada xogador do partido.

#### Zonas da pista

No pádel atopamos dúas posicións principais. Por un lado, a parella pode estar cerca da rede e atoparse en posición de ataque con intención de gañar o punto. Pola contra, a outra parella atoparase nunha posición de defensa no fondo da pista intentando que os rivais non lle fagan o punto. Entre estas dúas posicións principais, existe una zona de transición que nos permite pasar dunha zona a outra, é dicir, do fondo da pista á rede ou viceversa. Dita zona non é adecuada para xogar os puntos, xa que estaremos nunha posición intermedia onde non poderemos nin defender nin atacar adecuadamente e o rival poderá aproveitarse desta situación para gañarnos os puntos. Polo tanto, denominaremos esta parte da pista como “zona prohibida”.

Tendo isto en conta, unha información relevante para a análise dun partido de pádel é a zona da pista onde pasamos más tempo durante o partido. Dita información pode indicarnos se fomos más agresivos no noso xogo (pasamos más tempo cerca da rede) ou, se pola contra, fomos más defensivos (pasamos más tempo no fondo da pista). Polo tanto, o noso sistema xera un gráfico circular onde representa a porcentaxe de tempo que pasou un xogador nas diferentes zonas da pista ao longo do partido. Con este propósito, consideramos tres zonas, “Pegado á rede”, zona comprendida entre a posición da rede e a metade do campo, “Zona prohibida”, zona comprendida entre a metade do campo e un cuarto do tamaño da pista cara ó fondo, e “Fondo da pista”, zona comprendida entre a o límite da “Zona prohibida” e o final da pista.

Tomaremos como exemplo un dos partidos analizados no capítulo 6, correspondente ao xogador da dereita que comeza xogando na parte inferior da pista nun partido do *Estrella Damm Santander Open 2021*. Na figura 7.5 móstrase o gráfico xerado coas porcentaxes e a figura A.6 mostra o código co que xeramos dito gráfico.

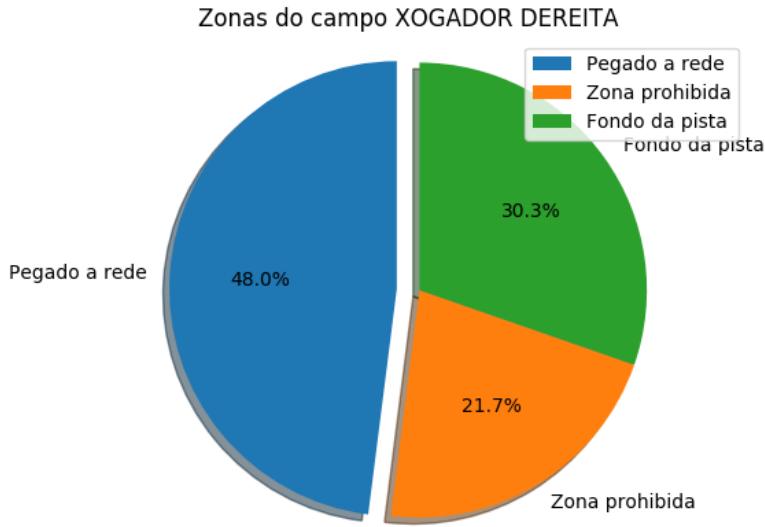


Figura 7.5: Gráfico circular que representa as zonas do campo.

### Mapa de calor

Un mapa de calor é unha técnica de visualización de datos que mide a magnitud dun fenómeno en cores. No noso caso, representaremos as posicións da pista polas que se moveu un xogador en concreto, utilizando a unha gama de cores cálida para as posicións da pista nas que máis tempo estivo, e unha gama de cores fría para as posicións que menos.

O proceso para a elaboración do mapa de calor foi o seguinte:

- En primeiro lugar, debuxamos unha pista ficticia en dúas dimensións sobre a que representaremos o mapa de calor que imos construír. Dita pista terá as mesas dimensións que o plano 2D empregado para facer a transformación xeométrica dos datos (ver sección 7.2), é dicir,  $300 \times 600$  píxeles. Podemos ver un exemplo do código na figura A.7.
- Dividimos o campo en  $N \times N$  posicións e calculamos as probabilidade que hai de que o xogador este nun momento concreto en cada unha dasas posicións a partir das coordenadas achegadas polo proceso de detección. Con estes datos, construímos unha matriz de  $N \times N$ . Para calcular ditas probabilidade, únicamente temos que dividir o número de veces as cales o xogador está nunha posición concreta entre as posicións totais que temos almacenadas. Unha vez que temos a probabilidade, debemos asociala á posición correspondente na matriz. Na figura A.8 podemos ver o código de como se realiza no proxecto.

- Unha vez que temos o campo e a matriz, seleccionamos a gama de cores que queremos utilizar, normalizamos as distintas posicións que construímos na pista e elaboramos o mapa de calor. As figuras 7.6 e A.9 mostran o mapa de calor xerado e o código empregado para a súa creación, respectivamente.

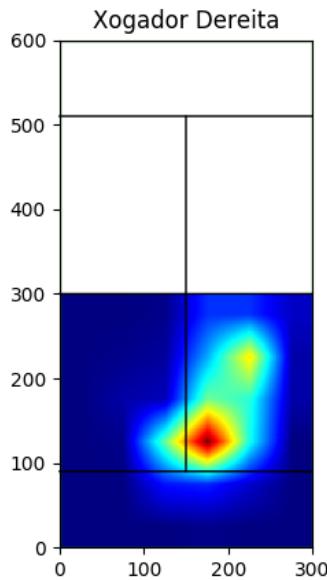


Figura 7.6: Mapa de calor para o xogador da dereita do partido do *Estrella Damm Santander Open 2021*

### 7.3.2 Gráficos colectivos

O sistema proporciona tamén gráficos colectivos. Estes son o resultado de analizar e comparar as posicións dos dous xogadores dunha mesma parella. Os distintos tipos de gráficos colectivos detállanse nas seguintes seccións.

#### Zonas da pista

O éxito dunha parella e o bo resultado dun partido de pádel dependen, en gran medida, da estratexia plantexada previamente a comezar o xogo. Nestes plantexamentos teóricos, é común deseñar un xogo máis agresivo se a pista é rápida e os rivais non teñen bo ataque, mentres que será un deseño máis defensivo se os rivais son bons rematadores e a pista é máis lenta do habitual. Polo tanto, resulta útil para os xogadores e adestradores analizar o tempo que cada xogador pasou nas diferentes zonas da pista, podendo así comprobar se a estratexia previamente plantexada foi seguida por ambos xogadores e se esta tivo o éxito esperado. Desta

maneira, decidimos comparar a frecuencia coa que ambos xogadores dunha mesma parella estaban nas diferentes zonas da pista. Para iso, coa mesma división da pista que para o gráfico individual, elaboramos un gráfico de barras conxunto. Un gráfico ou diagrama de barras está formado por barras rectangulares con lonxitudes proporcionais aos valores que estas representan. É unha alternativa moi utilizada nos casos nos que se quere comparar dous ou máis valores. Podemos ver un exemplo disto na figura 7.7 onde se representan as porcentaxes de tempo de cada xogador da parella 1 para o torneo *Estrella Damm Santander Open 2021*. podemos ver o código co que xeramos dito gráfico.

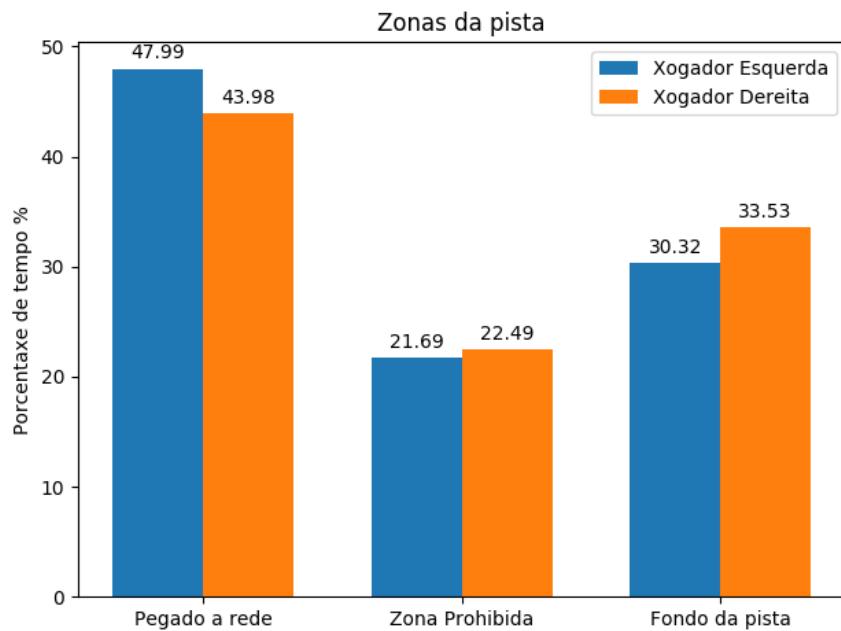


Figura 7.7: Diagrama de barras comparando as zonas da pista de dous xogadores.

### Descoordinación

No deporte do pádel, ao igual que noutrós deportes de equipo, é moi importante ter unha boa coordinación cos compañeiros/as. Unha descoordinación na parella pode levar a pérdida do punto e provocar que a estratexia de xogo falle. Polo tanto, resulta útil para os xogadores e adestradores, analizar dita descoordinación na parella, podendo así estudala e corrixila nos seguintes partidos. Utilizando un diagrama de barras, imos representar os momentos nos que os xogadores dunha mesma parella están descoordinados. Determinaremos que unha descoordinación ocorre cando sucede o seguinte:

- **Descoordinación horizontal:** momento no que ambos xogadores están do mesmo lado da pista, sexa esquierdo ou derecho.

- **Descoordinación vertical:** momento no que os xogadores dunha mesma parella están en diferentes zonas da pista, é dicir, un deles atópase na zona de ataque “pegado á rede”, mentres que o seu compaño/a atópase en zona defensiva no “fondo da pista”.

Este diagrama axudaralle aos xogadores a contabilizar o número de veces que están descoordinados, tanto nos movementos verticais de ataque/defensa ou nos movementos horizontais de cobertura da pista. Podemos ver un exemplo na figura 7.8 dos xogadores da parella 1 para o torneo *Estrella Damm Santander Open 2021*.

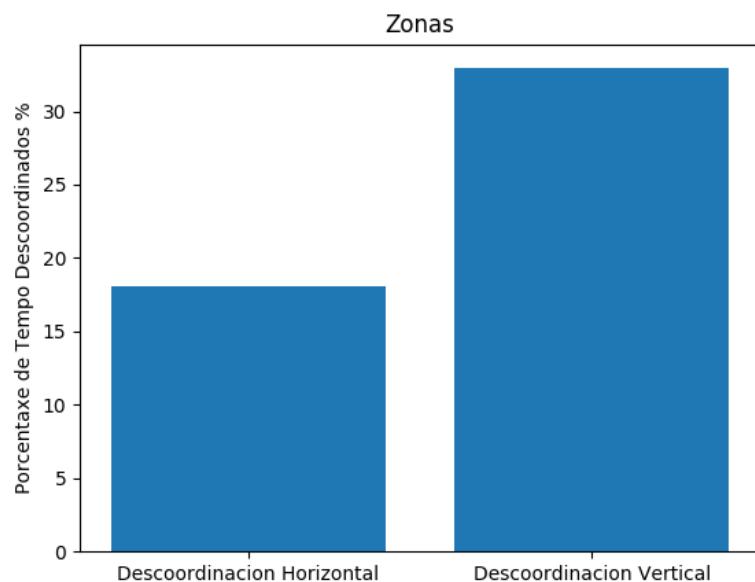


Figura 7.8: Diagrama de barras das descoordinacións.

## Capítulo 8

# Aplicación web

---

Como xa mencionamos anteriormente, desenvolveuse una aplicación web para que os usuarios do sistema poidan fazer uso de todas as funcionalidades que este ofrece dunha forma rápida e sinxela. Neste capítulo, explicaremos o proceso que se seguiu na elaboración de dita aplicación, así como a base de datos que esta manexa. Ademais, tamén mostraremos o aspecto visual da aplicación xunto cuns exemplos de uso.

### 8.1 Deseño

Un *framework* é un conxunto de componentes, reutilizables, escalables e de fácil manteemento, que nos proporciona diferentes ferramentas para o desenvolvemento dunha aplicación web. No noso proxecto, utilizamos **Django** (sección 3.1.6). O patrón de deseño que utilizamos con Django é unha variante do coñecido *Model View Controller (MVC)* [31], denominado *Model Template View (MTV)*.

Este patrón de deseño presenta as seguintes capas ou componentes:

- *Model* ou *Modelo (M)*, é a capa de acceso á base de datos. Contén toda a información sobre os datos, pois sabe cómo acceder a eles, cómo validalos, cál é o seu comportamento, e as relacións existentes entre todos eles.
- *Template* ou *Plantilla (T)*, é a capa de presentación. Contén as decisións relacionadas á forma na que son representados ou mostrados os distintos elementos dunha páxina web.
- *View* ou *Vista (V)*, é a capa da lóxiga de negocios. Contén a lóxica utilizada para acceder ao modelo e entregar a información na plantilla apropiada. Podemos ver esta componente como unha ponte entre os modelos e as plantillas do noso proxecto.

Na figura 8.1 podemos ver cómo funciona este patrón de deseño. Ao realizar *click* nun enlace ou escribir unha dirección *URL* no navegador (1), o primeiro ao que se accede é ao

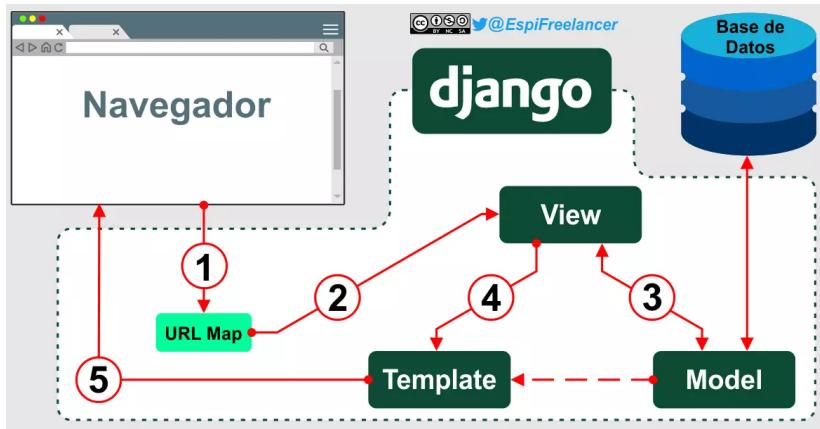


Figura 8.1: Funcionamento de Django.

mapa de *URLs*, dito mapa ten cada dirección asociada cunha *view* (2). Se para a presentación de dita *View* é necesario algún dato almacenado, este será solicitado ao *model* (3), o cal, a súa vez, realiza unha consulta á base de datos. Cando os datos son recuperados, estes son enviados á *template* (4), a cal contén a lóxica de presentación dos mesmos. Unha vez realizado este paso, a páxina é enviada ao navegador que fixo a correspondente solicitud (5).

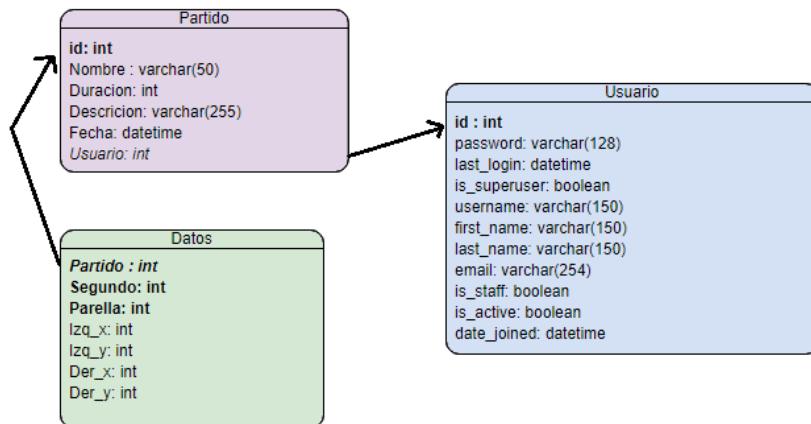


Figura 8.2: Diagrama relacional

## 8.2 Bases de datos

Co obxectivo de almacenar os datos relativos a análise dos vídeos e dos usuarios do noso sistema, na aplicación utilizamos o xestor de bases de datos relacionais **PostgreSQL** xunto con **PGAdmin4** para traballar coas diferentes entidades. En concreto, definíronse 3 entidades

distintas: **Usuario**, **Partido** e **Datos**. Cada unha destas entidades así como os seus atributos poden observarse na figura 8.2. Nela, as chaves primarias aparecen resaltadas en grosa, mentres que as chaves foráneas aparecen en cursiva.

A entidade **Usuario** é a que almacena os datos dos diferentes usuarios rexistrados no sistema. Entre os seus distintos atributos, destacamos o *id*, número identificativo único para cada usuario; o *email*, que será o correo electrónico co que iniciará sesión no sistema; e o *password*, contrasinal cifrada con algoritmo seguro que controla o acceso a páxina web. O resto de atributos conteñen información persoal do usuario como o seu nome, apelidos, data na que se rexistrou na aplicación, entre outros. Cabe mencionar que esta entidade é creada directamente por Django.

Por outro lado, temos a entidade **Partido**, a cal encárgase de almacenar a información dos partidos que os usuarios queren analizar. Nela, temos un identificador único para cada partido que realiza a función de chave primaria, *id*. O atributo *usuario* é unha chave foránea que fai referencia ao identificador dun usuario en concreto, o cal subiu vídeo do partido. O resto de atributos almacenan información propia do vídeo, como, por exemplo, o seu nome, a súa duración, a data na que foi gravado ou unha pequena descripción do mesmo.

Por último, a entidade **Datos** almacena a información resultante de procesar os vídeos co noso sistema. Ten unha chave primaria composta, formada polo conxunto dos tres atributos *partido*, *segundo* e *parella*. Para cada segundo dun vídeo, almacenamos as coordenadas de ambos xogadores dunha parella concreta. O atributo *partido*, é tamén chave foránea que fai referencia ao identificador dun **Partido** en concreto.

### 8.3 Exemplos de uso

Nesta sección mostraremos o aspecto visual da páxina web e a forma na que ofrecemos os nosos servizos a posibles usuarios interesados. Cabe mencionar que, ademais de Django, utilizouse a librería *Bootstrap* para o deseño visual da nosa aplicación e garantir que este se adaptase correctamente as diferentes resolucións dos dispositivos (sección 3.1.6).

Para ofrecer un deseño coherente en toda a aplicación, elaborouse unha plantilla común compartida por todas as seccións da páxina web, mantendo así elementos como as cabeceiras, menús e o pé de páxina.

Apoiándonos no diagrama fluxo presentado na figura 5.2a, imos plasmar mediante capturas de pantalla da aplicación o fluxo normal dun usuario ao empregar a nosa páxina.

O primeiro co que se atopa un usuario no noso sistema é a páxina de inicio da nosa aplicación. A cal é mostrada na figura 8.3. Nela, atopará información básica sobre o sistema así como unha explicación do seu funcionamento. Cabe mencionar que esta é a única sección que podemos visitar sen iniciar sesión coa nosa conta de usuario, para o resto de seccións, o

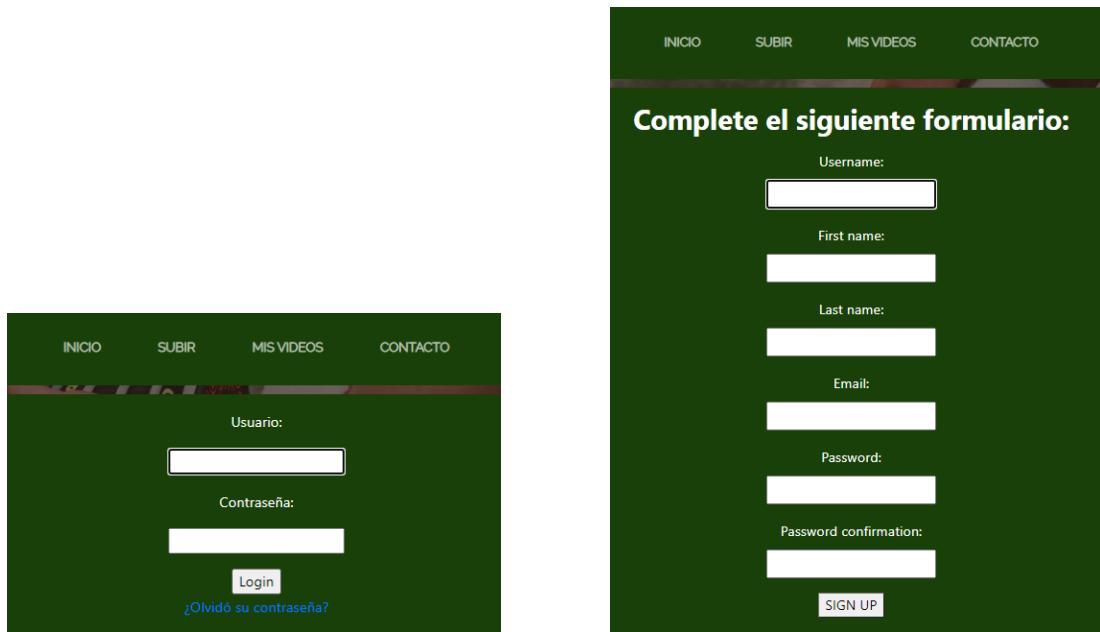


Figura 8.3: Páxina de inicio da nosa aplicación web.

inicio de sesión será requerido.

Para poder realizar dito inicio de sesión ou, no seu defecto, o rexistro como novo usuario, a aplicación emprega as pantallas mostradas nas figuras 8.4a e 8.4b, onde haberá que completar os distintos campos dos formularios para poder levar a cabo a correspondente acción. Unha vez iniciouse sesión, o usuario poderá xestionar os seus partidos, é dicir, subir novos ou eliminar partidos xa subidos. Estas opcións vense representadas na figura 8.5. Para subir un novo partido, o usuario debe seleccionar o arquivo desexado, indicar datos como son a data e a duración e, de forma opcional, engadir unha pequena descripción. Vemos tamén a lista dos partidos que xa temos procesados na nosa conta de usuario, os cales poderemos eliminar. Cabe mencionar que no caso de subir un arquivo que xa temos na nosa lista, ou un arquivo dun formato non aceptado, a aplicación mostraríanos unha mensaxe de error.

Por outro lado, o usuario poderá visualizar as representacións visuais (gráficos) dos seus partidos xa subidos e analizados polo sistema. Nesta sección, pódese escoller tanto unha vista en lista, mostrada na figura 8.6b, como unha vista en *grid*, mostrada na figura 8.6a, a cal pode resultar más cómoda no caso de ter unha gran cantidade de vídeos. A través desta sección, seleccionando o partido no cal temos interese, a parella que queremos analizar e se desexamos estatísticas colectivas ou individuais, podemos acceder aos distintos gráficos explicados no capítulo 7.



**(a) Cunha conta existente.**



**(b) Crear unha conta.**

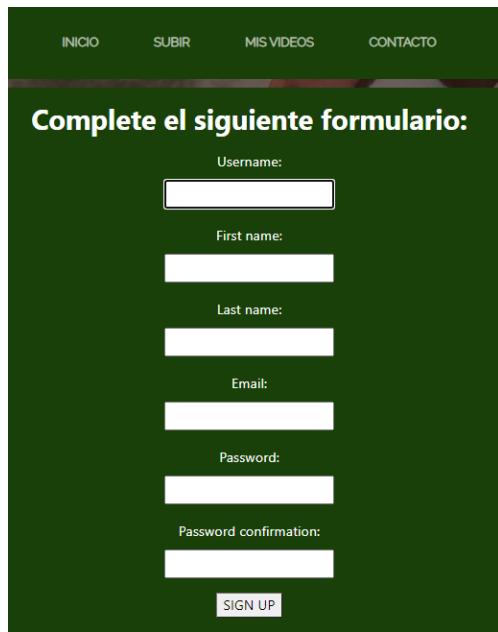
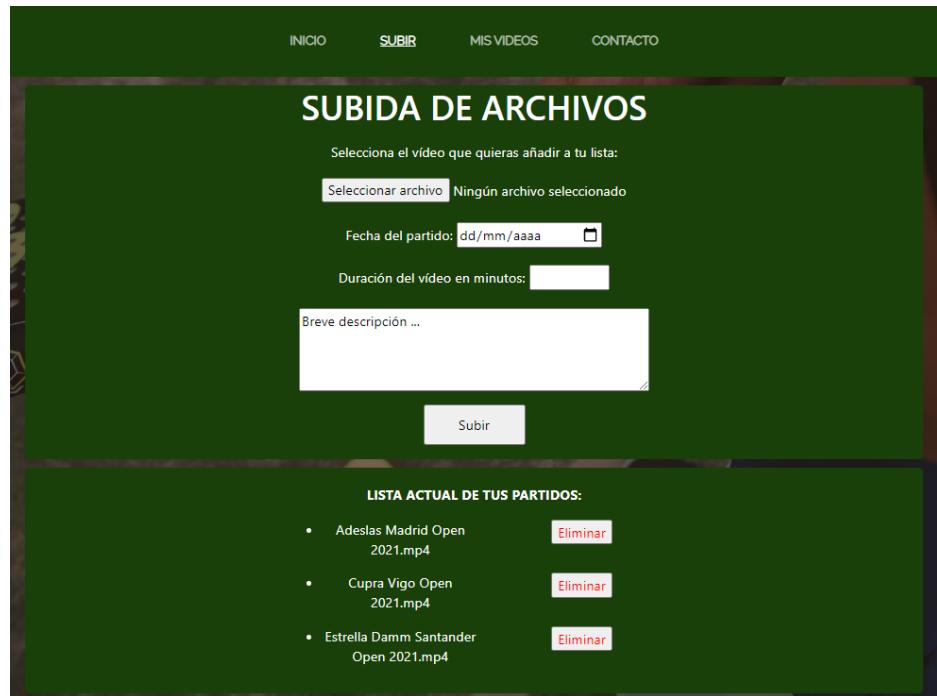


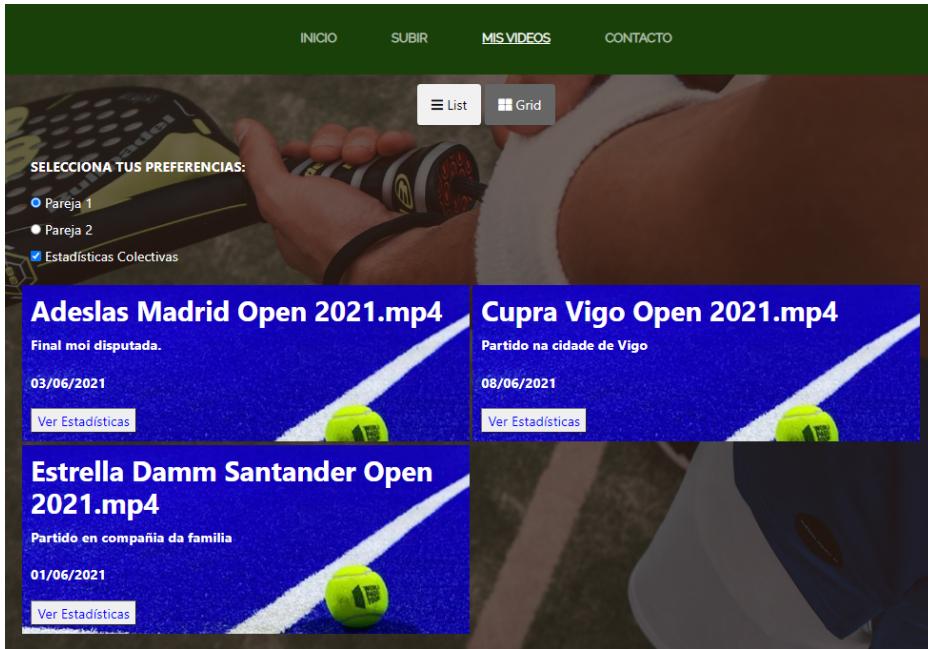
Figura 8.4: Iniciar sesión na aplicación web.



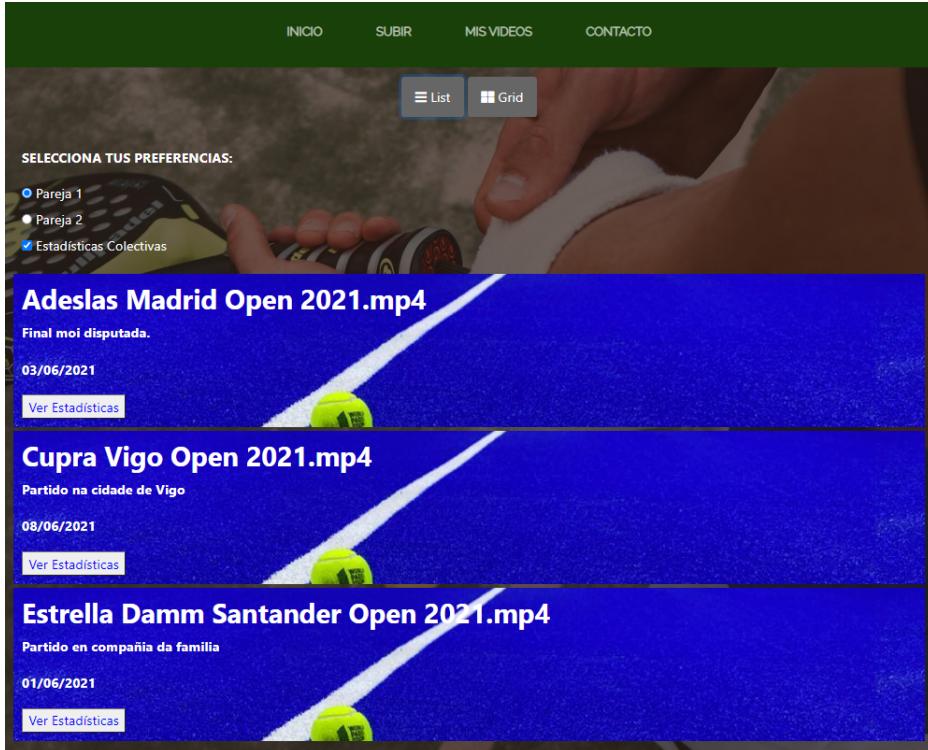
The screenshot shows the main interface of the web application. At the top, there is a navigation bar with links: INICIO, SUBIR (highlighted in red), MIS VIDEOS, and CONTACTO. Below the navigation bar, the title "SUBIDA DE ARCHIVOS" is displayed in large bold letters. The page contains several input fields: "Selecciona el video que quieras añadir a tu lista:" with a "Seleccionar archivo" button and a message "Ningún archivo seleccionado"; "Fecha del partido:" with a date input field "dd/mm/aaaa" and a calendar icon; "Duración del vídeo en minutos:" with an input field; and a "Breve descripción ..." text area. At the bottom of the page, there is a "Subir" button. Below this, a section titled "LISTA ACTUAL DE TUS PARTIDOS:" displays a list of uploaded videos:

- Adeslas Madrid Open 2021.mp4 Eliminar
- Cupra Vigo Open 2021.mp4 Eliminar
- Estrella Damm Santander Open 2021.mp4 Eliminar

Figura 8.5: Páxina de inicio da nosa aplicación web.



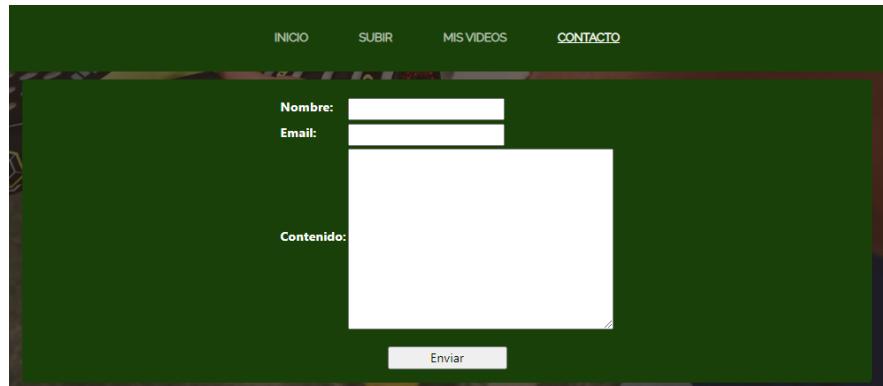
(a) Mostrando os vídeos nun grid.



(b) Mostrando os vídeos nunha lista.

Figura 8.6: Sección da galería dos vídeos de un usuario.

Por último, o sistema ofrece una sección de contacto para que os usuarios poidan comunicarse co responsable da páxina web, xa sexa por un problema no seu funcionamiento como por unha suxerencia. Esta sección pode verse na figura 8.7.



The screenshot shows a contact form on a dark-themed website. At the top, there is a navigation bar with links: INICIO, SUBIR, MIS VIDEOS, and CONTACTO. The CONTACTO link is underlined, indicating it is the active page. Below the navigation, there are three input fields: 'Nombre:' with a placeholder, 'Email:' with a placeholder, and a large text area labeled 'Contenido:' for message input. At the bottom right of the form is a 'Enviar' button.

Figura 8.7: Sección de contacto.



## Capítulo 9

# Conclusíons e liñas futuras

---

Neste capítulo analizamos o estado final do proxecto, así como as tarefas realizadas e coñecementos adquiridos ao longo deste. Ademais, tamén plantexamos as liñas futuras polas que podería continuar o desenvolvemento do noso sistema.

### 9.1 Conclusíons

O obxectivo principal do proxecto era o desenvolvemento dunha ferramenta software que, a partir de vídeos de partidos de pádel, proporcionara información de interese a xogadores e adestradores mediante representacións visuais amigables. Tras analizar o sistema desenvolto e os seus resultados, podemos concluir que se acadaron os obxectivos marcados ao comezo do proxecto. O resultado obtido foi o desexado, pois conseguiuse elaborar un sistema capaz de analizar partidos de pádel, extraer información estatística do xogo e dos xogadores e, finalmente, representar dita información dunha forma organizada é accesible nunha páxina web.

Ademais, debemos destacar os coñecementos adquiridos sobre tecnoloxías antes desconocidas polo alumno. Na mención “*Tecnoloxías da Información*” cursada no Grao de Enxeñaría Informática, campos como o da visión artificial ou manexo e edición de imaxes a través de librerías como OpenCV, non eran obxecto de estudio. Polo tanto, valoro moi positivamente o aprendido ao longo deste proxecto, xa que axudoume a ampliar os meus coñecementos.

Por outro lado, tiven oportunidade de reforzar e afianzar todos os aspectos relacionados co desenvolvemento de aplicacións web. Ademais, tamén considero moi importante haber implementado un proxecto de comezo a fin, pois reforzáronse e ampliáronse os coñecementos de análise, deseño e desenvolvemento de proxectos que se estudaron ao longo do grao.

## 9.2 Liñas futuras

O noso proxecto pode considerarse unha primeira aproximación do sistema proposto, polo que, áinda que nos atopamos ante un produto completo, existen opcións de mellora e ampliación. Debido, en gran parte, á modularidade do software, estas liñas futuras de mellora poden implantarse con facilidade. Para comenzar, a aplicación web pode ser sometida a calquera tipo de modificación, xa sexa visual, de deseño, ou das funcionalidades que esta ofrece aos usuarios, mostrando novos gráficos ou extraendo distintas informacións estatísticas dos partidos.

Por outro lado, podemos variar os algoritmos a utilizar na componente de detección dos xogadores, intentando mellorar a precisión na detección e reducir os falsos positivos. Ademais, nun futuro, pode ser interesante detectar e facer seguimento da pelota e das palas dos xogadores, conseguindo establecer os comezos e finais dos puntos, así como a calidad e tipos de golpes que executa cada xogador. Isto, permitiría a ferramenta facer unha análise moito máis completa do partido, así como obter estatísticas personalizadas dos xogadores dos puntos gañados, perdidos ou fallados.

# **Apéndices**



# Apéndice A

## Material adicional

---

### A.1 Algoritmo de detección

Na figura [A.1](#) podemos ver un *script* que nos serviu de axuda á hora de configurar os distintos parámetros da función *detectMultiScale*.

### A.2 K-Means

O código empregado para aplicar K-Means no noso proxecto pode atoparse nas figuras [A.2](#), [A.3](#) e [A.4](#).

### A.3 Transformación de perspectiva

O código empregado no noso proxecto para realizar a transformación de perspectiva pódemolo ver na figura [A.5](#).

### A.4 Gráficos

Por último, mostramos o código para algúns dos nosos gráficos. Na figura [A.6](#) podemos ver o correspondente ao gráfico circular que reflexa as porcentaxes de tempo que pasou un xogador nas distintas zonas da pista. Na figura [A.10](#) vemos como construímos o diagrama de barras comparando as zonas da pista entre dous xogadores dunha mesma parella mentres que nas figuras [A.7](#), [A.8](#) e [A.9](#) temos o código co que construímos o mapa de calor.

```

1 # importacion de paquetes necesarios
2 from __future__ import print_function
3 import argparse
4 import datetime
5 import imutils
6 import cv2
7
8 # construccion do argparse
9 ap = argparse.ArgumentParser()
10 ap.add_argument("-i", "--image", required=True,
11     help="path imaxe de entrada")
12 ap.add_argument("-w", "--win-stride", type=str, default="(8, 8)",
13     help="window stride")
14 ap.add_argument("-p", "--padding", type=str, default="(16, 16)",
15     help="object padding")
16 ap.add_argument("-s", "--scale", type=float, default=1.05,
17     help="image pyramid scale")
18 ap.add_argument("-m", "--mean-shift", type=int, default=-1,
19     help="usamos mean shift grouping o no")
20 args = vars(ap.parse_args())
21
22 # evaluamos os argumentos
23 winStride = eval(args["win_stride"])
24 padding = eval(args["padding"])
25 meanShift = True if args["mean_shift"] > 0 else False
26
27 # inicializamos o detector
28 hog = cv2.HOGDescriptor()
29 hog.setSVMClassifier(cv2.HOGDescriptor_getDefaultPeopleDetector())
30
31 # cargamos a imaxe
32 image = cv2.imread(args["image"])
33 image = imutils.resize(image, width=min(400, image.shape[1]))
34
35 # detección de persoas na imaxe e medición do tempo do procesado
36 start = datetime.datetime.now()
37 (rects, weights) = hog.detectMultiScale(image, winStride=winStride,
38     padding=padding, scale=args["scale"],
39     useMeanshiftGrouping=meanShift)
40 print("[INFO] tempo de procesado: {}".format(
41     (datetime.datetime.now() - start).total_seconds()))
42
43 # dibuxamos os rectangulos correspondentes as deteccions
44 for (x, y, w, h) in rects:
45     cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
46
47 # enseñamos a imaxe por pantalla
48 cv2.imshow("Detections", image)
49 cv2.waitKey(0)

```

Figura A.1: Script de probas da función ***detectMultiScale***

```

1 # importamos os paquetes necesarios
2 from sklearn.cluster import KMeans
3 import matplotlib.pyplot as plt
4 import argparse
5 import utils
6 import cv2
7
8 # construimos os argumentos
9 ap = argparse.ArgumentParser()
10 ap.add_argument("-i", "--image", required = True, help = "Path to
11                 the image")
12 ap.add_argument("-c", "--clusters", required = True, type = int,
13                 help = "# of clusters")
14 args = vars(ap.parse_args())
15
16 # cargamos a imaxe e convertimola de BGR a RGB
17 # desta forma podemos mostrala con matplotlib
18 image = cv2.imread(args["image"])
19 image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
20
21 # mostramos a imaxe
22 plt.figure()
23 plt.axis("off")
24 plt.imshow(image)
25
26 # reshape da imaxe para convertila nunha lista de pixeles
27 image = image.reshape((image.shape[0] * image.shape[1], 3))

```

Figura A.2: Cores dominantes dunha imaxe con K-Means (I)

```

1 # cluster da intensidade dos pixeles
2 clt = KMeans(n_clusters = args["clusters"])
3 clt.fit(image)
4
5 # construimos un histograma dos cluster e creamos a figura
6 # que representa o numero de pixeles asignados a cada cor
7 hist = utils.centroid_histogram(clt)
8 bar = utils.plot_colors(hist, clt.cluster_centers_)
9
10 # mostramos a barra das cores
11 plt.figure()
12 plt.axis("off")
13 plt.imshow(bar)
14 plt.show()

```

Figura A.3: Cores dominantes dunha imaxe con K-Means (II)

```

1 # importamos os paquetes necesarios
2 import numpy as np
3 import cv2
4
5 def centroid_histogram(clt):
6     # collemos o numero de clusters e creamos un histograma
7     # en función do numero de pixeles asignados a cada cluster
8     numLabels = np.arange(0, len(np.unique(clt.labels_)) + 1)
9     (hist, _) = np.histogram(clt.labels_, bins = numLabels)
10
11    # normalizamos o histograma
12    hist = hist.astype("float")
13    hist /= hist.sum()
14
15    # devolvemos o histograma
16    return hist
17
18 def plot_colors(hist, centroids):
19
20     # inicializamos a barra de cores representando a frecuencia de
21     # cada cor
22     bar = np.zeros((50, 300, 3), dtype = "uint8")
23     startX = 0
24
25     # bucle para recorrer o porcentaje e a cor de cada cluster
26     for (percent, color) in zip(hist, centroids):
27         # indicamos o porcentaje de cada cor
28         endX = startX + (percent * 300)
29         cv2.rectangle(bar, (int(startX), 0), (int(endX), 50),
30             color.astype("uint8").tolist(), -1)
31         startX = endX
32
33     # devolvemos a barra de cores
34     return bar

```

Figura A.4: Utilidades de K-Means

```

1 #obtemos a matriz de transformacion de perspectiva
2 def obtenerMatriz(r,c):
3     pts1 = np.float32([[297,150],[655,150],[845,465],[100,465]])
4     pts2 = np.float32([[0,0],[r,0],[r,c],[0,c]])
5     return cv2.getPerspectiveTransform(pts1, pts2)
6
7 #funcion pa visualizar a imaxe transformada
8 def transformarPerspectiva(r,c):
9     src = cv2.imread('captura_campo_950.png')
10    pts1 = np.float32([[297,150],[655,150],[845,465],[100,465]])
11    pts2 = np.float32([[0,0],[r,0],[r,c],[0,c]])
12    M = cv2.getPerspectiveTransform(pts1, pts2)
13    dst = cv2.warpPerspective(src, M, (r,c))
14    cv2.imshow('captura_campo_950.png', src)
15    cv2.imshow('Transform', dst)
16    cv2.waitKey()
17
18 #ca matriz de transformacion e o punto orixinal, obtemos o punto
19 #resultante
20 def calcularPunto(matriz,pt):
21     #append de z=1
22     pt = np.append(pt,1)
23     #multiplicamos matriz por vector (x,y,z)
24     pt = np.matmul(matriz,pt)
25     #temos x'y'z', debemos eliminar z'
26     pt = pt / pt[2]
27     pt = np.delete(pt,2)
28     return pt

```

Figura A.5: Liñas de código coas que aplicamos a transformación de perspectiva.

```

1 def graficoPastel(valores, jugador):
2
3     zonas_campo = 'Pegado a rede', 'Zona prohibida', 'Fondo da pista'
4     #Declaramos o tamano de cada 'rebanada' e en sumatoria todos
5     #deben dar o 100%
6
7     size = sumaLista(valores)
8     for e in valores:
9         e = (e/size)*100
10
11    #En este punto senalamos que posicion debe 'resaltarse' e o seu
12    #valor, se se coloca 0, omitese
13    maxpos = valores.index(max(valores))
14    if maxpos==0:
15        explode = (0.1, 0, 0)
16    elif maxpos==1:
17        explode = (0, 0.1, 0)
18    else:
19        explode = (0, 0, 0.1)
20
21    fig1, ax1 = plt.subplots()
22    #Creamos o grafico, anadindo os valores
23    ax1.pie(valores, explode=explode, labels=zonas_campo,
24             autopct='%.1f%%',
25             shadow=True, startangle=90)
26    #senalamos a forma, en este caso 'equal' e para dar forma circular
27    ax1.axis('equal')
28    plt.title("Zonas do campo XOGADOR "+ jugador)
29    plt.legend()
30    plt.savefig('grafica_pastel_'+jugador+'.png')
31    plt.show()

```

Figura A.6: Liñas de código coas que construímos o gráfico de Zonas do Campo.

```

1 def draw_pitch(ax,b):
2     Pitch = Rectangle([0,0], width = 300, height = 600, fill = b,
3     color = 'green')
4     red = ConnectionPatch([0,300], [300,300], "data", "data", color
5     = 'black')
6     rightline = ConnectionPatch([0,90], [300,90], "data", "data",
7     color = 'black')
8     leftline = ConnectionPatch([0,510], [300,510], "data", "data",
9     color = 'black')
10    centerline = ConnectionPatch([150,90], [150,510], "data",
11    "data", color = 'black')
12
13    element = [Pitch, rightline, leftline, centerline, red]
14    for i in element:
15        ax.add_patch(i)

```

Figura A.7: Liñas de código coas que debuxamos a pista.

```
1 def heatMatrix_aux(n, r, c, dic):
2
3     matriz = np.zeros((n, n))
4
5     for array in dic:
6         if len(dic[array])!=0:
7
8             x = dic[array][0]
9             y = dic[array][1]-(c/2)
10            fila = int(y/(r/n))
11            columna = int(x/(r/n))
12
13            if fila==n:
14                fila = fila-1
15            if columna==n:
16                columna = columna-1
17            matriz[fila,columna]+=1
18
19    num_elementos = np.sum(matriz)
20
21    for i in range(len(matriz)):
22        for j in range(len(matriz[i])):
23            matriz[i][j] = matriz[i][j]/num_elementos
24
25    return matriz
```

Figura A.8: Liñas de código coas que construímos a matriz.

```
1 def heatMatrix(n,r,c,dic):
2
3     fig, ax = plt.subplots()
4     draw_pitch(ax, False)
5
6     ax.set_aspect('equal')
7     ax.set_title("Xogador Dereita")
8     ax.set_xlim(0, r)
9     ax.set_ylim(0, c)
10
11    matriz = heatMatrix_aux(n,r,c,dic)
12    print(matriz)
13    matriz = cv2.resize(matriz, (300,300))
14    plt.matshow(matriz)
15    heatmapshow = None
16    heatmapshow = cv2.normalize(matriz, heatmapshow, alpha=0,
17        beta=255, norm_type=cv2.NORM_MINMAX, dtype=cv2.CV_8U)
18    heatmapshow = cv2.applyColorMap(heatmapshow, cv2.COLORMAP_JET)
19    cv2.imshow("Heatmap", heatmapshow)
20
21    plt.imshow(cv2.cvtColor(heatmapshow, cv2.COLOR_BGR2RGB))
22    plt.savefig('heatmap.png')
23    plt.show()
24    cv2.waitKey(0)
```

Figura A.9: Liñas de código coas que construímos o mapa de calor.

## APÉNDICE A. MATERIAL ADICIONAL

---

```
1 def graficoBarras(valores1,valores2,segundos):
2
3     asistencia = [ 'Pegado a rede', 'Zona Prohibida', 'Fondo da pista' ]
4
5     a = (valores1[0]*100)/float(segundos)
6     b = (valores1[1]*100)/float(segundos)
7     c = (valores1[2]*100)/float(segundos)
8
9     d = (valores2[0]*100)/float(segundos)
10    e = (valores2[1]*100)/float(segundos)
11    f = (valores2[2]*100)/float(segundos)
12
13    jugador_izq = [round(a,2),round(b,2),round(c,2)]
14    jugador_der = [round(d,2),round(e,2),round(f,2)]
15
16    print(jugador_izq)
17    print(jugador_der)
18
19    #Obtenemos la posicion de cada etiqueta en el eje de X
20    x = np.arange(len(asistencia))
21    #tamano de cada barra
22    width = 0.35
23
24    fig, ax = plt.subplots()
25
26    #Generamos las barras para el conjunto de hombres
27    rects1 = ax.bar(x - width/2, jugador_izq, width, label='Xogador
28        Esquerda')
29    #Generamos las barras para el conjunto de mujeres
30    rects2 = ax.bar(x + width/2, jugador_der, width, label='Xogador
31        Dereita')
32
33    #Anadimos las etiquetas de identificacion de valores en el grafico
34    ax.set_ylabel('Porcentaxe de tempo %')
35    ax.set_title('Zonas da pista')
36    ax.set_xticks(x)
37    ax.set_xticklabels(asistencia)
38    #Anadimos un legend() esto permite mmostrar con colores a que
39    # pertence cada valor.
40    ax.legend()
41
42    #Anadimos las etiquetas para cada barra
43    autolabel(rects1,ax)
44    autolabel(rects2,ax)
45    fig.tight_layout()
46    plt.savefig('zonas_comparacion.png')
47    #Mostramos la grafica con el metodo show()
48    plt.show()
```

Figura A.10: Liñas de código coas que construímos o gráfico de comparación das zonas da pista.



# Relación de Acrónimos

---

**ACID** Atomicity, Consistency, Isolation and Durability. [15](#)

**BOE** Boletín Oficial do Estado. [25](#)

**CNN** Redes Neuronais Convolucionais. [48, 51](#)

**CSRF** Cross Site Request Forgery. [16](#)

**HOG** Histogram of Oriented Gradients. [33, 35–37, 39, 45, 48, 53, 54](#)

**IA** Intelixencia Artificial. [1–5, 8–10, 21](#)

**IDE** Entorno de Desarrollo Integrado. [13, 14](#)

**IOU** Intersection Over Union. [50, 51](#)

**MTV** Model Template View. [71](#)

**MVC** Model View Controller. [71](#)

**NMS** Non-Maximum Suppression. [v, 39–42, 45, 46, 50](#)

**ROI** Region Of Interest. [vi, 37, 64](#)

**SVM** Support Vector Machines. [33–37, 39, 45, 48, 52–54](#)

**VAR** Video Assistant Referee. [3](#)

**YOLO** You Only Look Once. [ii, vi, 15, 33, 48–55](#)



# Bibliografía

---

- [1] K. Tuyls, S. Omidshafiei, and P. Muller, “Game plan: What ai can do for football, and what football can do for ai,” 5 2021. [En liña]. Dispoñible en: <https://www.jair.org/index.php/jair/article/view/12505/26683>
- [2] I. Laguna, “La inteligencia artificial juega al fútbol,” 4 2021. [En liña]. Dispoñible en: <https://www.efe.com/efe/espana/deportes/la-inteligencia-artificial-juega-al-futbol/10006-4503587>
- [3] M. Informática, “Aiproclips, la inteligencia artificial aplicada al fútbol,” 9 2020. [En liña]. Dispoñible en: <https://mrinformatica.es/aiproclips-la-inteligencia-artificial-aplicada-al-futbol/>
- [4] A. B. Santos, R. Theron, A. Losada, J. Sampaio, and C. L. Peñas, “Data-driven visual performance analysis in soccer: An exploratory prototype,” *Frontiers in Psychology*, 12 2018. [En liña]. Dispoñible en: <https://www.frontiersin.org/articles/10.3389/fpsyg.2018.02416/full>
- [5] J. P. Oliveros, “Ibm crea el mejor entrenador de tenis: Inteligencia artificial,” 8 2019. [En liña]. Dispoñible en: <https://criptotendencia.com/2019/08/29/ibm-crea-el-mejor-entrenador-de-tenis-inteligencia-artificial/>
- [6] H. Zhang, C. Sciuotto, M. Agrawala, and K. Fatahalian, “Vid2player: Controllable video sprites that behave and appear like professional tennis players,” 12 2020. [En liña]. Dispoñible en: <https://arxiv.org/pdf/2008.04524.pdf>
- [7] “Inteligencia artificial aplicada al análisis de rendimiento,” data de consulta: 2021-05-24. [En liña]. Dispoñible en: <https://olocip.com/tenis/>
- [8] “Python, open-source programming language.” [En liña]. Dispoñible en: <https://www.python.org/>
- [9] “Visual studio code.” [En liña]. Dispoñible en: <https://code.visualstudio.com/>

- [10] “Opencv, enter the world of computer vision.” [En liña]. Dispoñible en: <https://opencv.org/>
- [11] “Yolo object detection with opencv and python.” [En liña]. Dispoñible en: <https://www.visiongeek.io/2018/07/yolo-object-detection-opencv-python.html>
- [12] “Postgresql: The world’s most advanced open source relational database.” [En liña]. Dispoñible en: <https://www.postgresql.org/>
- [13] “pgadmin, postgresql tools.” [En liña]. Dispoñible en: <https://www.pgadmin.org/>
- [14] “Django, the web framework for perfectionists with deadlines.” [En liña]. Dispoñible en: <https://www.djangoproject.com/>
- [15] “Bootstrap, quickly design and customize responsive sites.” [En liña]. Dispoñible en: <https://getbootstrap.com/>
- [16] “Wondershare filmora, para los creativos.” [En liña]. Dispoñible en: <https://filmora.wondershare.es/>
- [17] “Git, everything is local.” [En liña]. Dispoñible en: <https://git-scm.com/>
- [18] “El editor de latex fácil de usar, online y colaborativo.” [En liña]. Dispoñible en: <https://es.overleaf.com/>
- [19] D. X. de Empleo, “Xvii convenio colectivo estatal de empresas de consultoría, e estudios de mercados e da opinión pública,” *Boletín Oficial do Estado*, 3 2018. [En liña]. Dispoñible en: [https://boe.es/eli/es/res/2018/02/22/\(3\)](https://boe.es/eli/es/res/2018/02/22/(3))
- [20] “Requerimientos funcionales y no funcionales, ejemplos y consejos.” [En liña]. Dispoñible en: <https://medium.com/@requeridosblog/requerimientos-funcionales-y-no-funcionales-ejemplos-y-tips-aa31cb59b22a>
- [21] “Técnicas para identificar requisitos funcionales y no funcionales.” [En liña]. Dispoñible en: <https://sites.google.com/site/metodologiareq/capitulo-ii/tecnicas-para-identificar-requisitos-funcionales-y-no-funcionales>
- [22] “¿qué es un requerimiento funcional?” [En liña]. Dispoñible en: <http://www.pmoinformatica.com/2018/05/que-es-requerimiento-funcional.html>
- [23] “Histograms of oriented gradients for human detection, cvpr de 2005.” [En liña]. Dispoñible en: <https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>

## BIBLIOGRAFÍA

---

- [24] “Introduction to yolo algorithm for object detection.” [En línea]. Disponible en: <https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/>
- [25] “A state of the art algorithm for real-time object detection system.” [En línea]. Disponible en: <https://towardsdatascience.com/yolo-you-only-look-once-3dbdbb608ec4>
- [26] “Overview of the yolo object detection algorithm.” [En línea]. Disponible en: <https://medium.com/@ODSC/overview-of-the-yolo-object-detection-algorithm-7b52a745d3e0>
- [27] “K-means clustering algorithm.” [En línea]. Disponible en: <https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning>
- [28] “Understanding k-means clustering in machine learning.” [En línea]. Disponible en: <https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1>
- [29] “Opencv and python k-means color clustering.” [En línea]. Disponible en: <https://www.pyimagesearch.com/2014/05/26/opencv-python-k-means-color-clustering/>
- [30] “What is perspective transformation?” [En línea]. Disponible en: <https://theailearner.com/tag/cv2-getperspectivetransform/>
- [31] “Modelo–vista–controlador.” [En línea]. Disponible en: <https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador>

