



UNIVERSITAT  
POLITÈCNICA  
DE VALÈNCIA



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Análisis e integración de librerías open source para el estudio de las capacidades computacionales de sistemas de biometría facial en videoporteros convencionales.

Trabajo Fin de Máster

Máster Universitario en Ingeniería Informática

AUTOR/A: Rodríguez Pérez, Raúl

Tutor/a: Pérez Blasco, Pascual

CURSO ACADÉMICO: 2023/2024

*A mi familia y pareja,  
que pese a las dificultades  
que han traído estos años,  
nunca han dudado de mí.  
Muchísimas gracias.*



# Resumen

---

El presente trabajo fin de máster tiene como principal objetivo analizar la viabilidad de incorporar funcionalidades relacionadas con la detección y el reconocimiento facial en un videoportero comercial. En los últimos años, los avances en el campo de la inteligencia artificial han impulsado significativamente las capacidades de la visión por computador. Los logros obtenidos con la creación de algoritmos y modelos basados en técnicas de aprendizaje profundo han revolucionado la forma en la que procesamos y analizamos imágenes, mejorando notablemente el rendimiento y abriendo nuevas posibilidades en áreas como el reconocimiento de objetos, la detección de patrones, etc. Al considerar la integración de las funcionalidades mencionadas en el videoportero comercial, es importante tener en cuenta que, debido a la naturaleza del dispositivo, existirán limitaciones computacionales inherentes a los sistemas embebidos. Por tanto, uno de los principales objetivos del proyecto se basa en conseguir resultados que velen por una positiva experiencia del usuario a expensas de las limitaciones mencionadas.

El trabajo constará de diversas fases claramente diferenciadas. En primer lugar, se llevará a cabo un estudio tanto de las últimas técnicas y modelos para la visión por computador, como de las principales herramientas de código abierto (bibliotecas, marcos de trabajo, etc.) para el procesamiento de dichos modelos. Una vez finalizada la etapa de análisis e investigación, se realizará una preselección de aquellas herramientas y modelos que, en primera instancia, cumplan con los requisitos establecidos por el sistema objetivo (sistema Linux embebido con arquitectura ARM). A continuación, se llevará a cabo la integración de las herramientas en el sistema, además del diseño de las pruebas para evaluar el rendimiento de los modelos en el videoportero. Una vez realizadas las pruebas, se analizarán los resultados y se determinará la viabilidad de incluir las funcionalidades de detección y reconocimiento facial en el videoportero comercial.

Finalmente, debemos destacar que en este trabajo también se abordará las preocupaciones y cuestiones éticas que surgen a raíz de la integración de técnicas como la detección y el reconocimiento facial en sistemas comerciales. Por un lado, el uso de estas tecnologías podría mejorar significativamente la seguridad de un edificio, reduciendo el riesgo de intrusiones a personas no autorizadas. Sin embargo, la falta de transparencia en muchas ocasiones sobre la gestión realizada con los datos personales de los usuarios suele provocar el rechazo por parte de estos. Por otro lado, también hay que tener en cuenta que los modelos de reconocimiento pre entrenados que se seleccionen pueden estar sesgados dependiendo de si los datos con los que han sido entrenados son o no lo suficientemente representativos. Esto puede producir decisiones erróneas por parte del sistema, provocando la discriminación de algunos usuarios según sus rasgos.

**Palabras clave:** detección facial, reconocimiento facial, visión por computador, sistema embebido, inteligencia artificial, procesamiento de imágenes

---

# Abstract

---

The main objective of this master's thesis is to analyze the potential of incorporating functionalities related to face detection and recognition in a commercial video door entry system. In recent years, advances in the field of artificial intelligence have significantly boosted the capabilities of computer vision. The achievements obtained with the creation of algorithms and models based on deep learning techniques have revolutionized the way in which we process and analyze images, significantly improving performance and opening new possibilities in areas such as object recognition, pattern detection, etc. When considering the integration of the mentioned functionalities in the commercial video door entry system, it is important to keep in mind that, due to the nature of the device, there will be computational limitations inherent to embedded systems. Therefore, one of the main objectives of the project is based on achieving results that ensure a positive user experience at the expense of the limitations.

The work will consist of several clearly distinguishable phases. First, a study of the latest techniques and models for computer vision, as well as of the main open-source tools (libraries, frameworks, etc.) for the processing of these models will be carried out. Once the analysis and research stage are completed, a pre-selection of those tools and models that, in the first instance, meet the requirements established by the target system (embedded Linux system with ARM architecture) will be made. Next, the integration of the tools into the system will be carried out, in addition to the design of the tests to evaluate the performance of the models in the video door entry system. Once the tests have been carried out, the results will be analyzed and the potential of including the facial detection and recognition functionalities in the commercial video door entry system will be determined.

Finally, we should note that this paper will also address the concerns and ethical issues that arise from the integration of techniques such as face detection and recognition into commercial systems. On the one hand, the use of these technologies could significantly improve the security of a building, reducing the risk of intrusion by unauthorized persons. However, the lack of transparency in many cases about the management of users' personal data often leads to their rejection. On the other hand, it must also be considered that the pre-trained recognition models that are selected may be biased depending on whether the data with which they have been trained are sufficiently representative. This can lead to erroneous decisions by the system, causing discrimination of some users based on their features.

**Keywords:** face detection, face recognition, computer vision, embedded system, artificial intelligence, image processing

---



# Índice general

---

<b>Resumen .....</b>	<b>4</b>
<b>Índice general .....</b>	<b>7</b>
<b>Índice de Figuras .....</b>	<b>9</b>
<b>Índice de Tablas .....</b>	<b>10</b>
<b>Glosario.....</b>	<b>10</b>
<b>Acrónimos .....</b>	<b>11</b>
<b>1. Introducción.....</b>	<b>13</b>
1.1 <i>Motivación .....</i>	<i>13</i>
1.2 <i>Objetivos.....</i>	<i>14</i>
1.3 <i>Estructura de la memoria.....</i>	<i>15</i>
<b>2. Estado del arte.....</b>	<b>17</b>
2.1 <i>Detección facial.....</i>	<i>19</i>
2.2 <i>Reconocimiento facial .....</i>	<i>24</i>
<b>3. Análisis del problema .....</b>	<b>30</b>
3.1 <i>Especificación de requisitos.....</i>	<i>30</i>
3.2 <i>Análisis del marco legal y ético .....</i>	<i>32</i>
3.2.1 <i>Protección de datos.....</i>	<i>32</i>
3.2.2 <i>Propiedad intelectual.....</i>	<i>33</i>
3.2.3 <i>Marco ético.....</i>	<i>33</i>
3.3 <i>Análisis de riesgos .....</i>	<i>33</i>
3.3.1 <i>Identificación de los eventos de riesgo .....</i>	<i>34</i>
3.3.2 <i>Evaluación del riesgo.....</i>	<i>35</i>
3.3.3 <i>Confección de respuestas .....</i>	<i>36</i>
<b>4. Planificación de la solución .....</b>	<b>39</b>
4.1 <i>Metodología .....</i>	<i>39</i>
4.2 <i>Estimación del plan de trabajo .....</i>	<i>40</i>
4.3 <i>Presupuesto .....</i>	<i>41</i>
4.3.1 <i>Recursos humanos.....</i>	<i>41</i>
4.3.2 <i>Recursos materiales.....</i>	<i>42</i>
4.3.3 <i>Presupuesto final .....</i>	<i>43</i>
<b>5. Desarrollo de la solución .....</b>	<b>44</b>
5.1 <i>Detección facial.....</i>	<i>44</i>
5.1.1 <i>Fase de análisis .....</i>	<i>44</i>
5.1.2 <i>Integración de las bibliotecas seleccionadas .....</i>	<i>46</i>

5.1.3	Subsistema de visión por computador .....	47
5.1.4	Preselección de modelos .....	48
5.1.5	Pruebas iniciales .....	50
5.1.6	Tareas de optimización.....	54
5.1.7	Pruebas finales y conclusiones.....	61
5.2	<i>Reconocimiento facial</i> .....	66
5.2.1	Primero pasos .....	66
5.2.2	Proceso de verificación de identidad .....	67
5.2.3	Diseño de la prueba de concepto .....	69
5.2.4	Registro del usuario.....	70
5.2.5	Suplantación de identidad .....	72
5.2.6	Tareas de reconocimiento facial .....	76
<b>6.</b>	<b>Conclusiones .....</b>	<b>80</b>
6.1	<i>Trabajo futuro</i> .....	81
	<b>Bibliografía .....</b>	<b>83</b>



# Índice de Figuras

---

<b>Figura 1.1:</b> Logo de la empresa Fermax.....	13
<b>Figura 2.1 :</b> Explicación del experimento de Hubel & Wiesel.....	17
<b>Figura 2.2:</b> Diferencias entre la visión por computador y la visión humana.....	18
<b>Figura 2.3:</b> Resultado de ejemplo tras aplicar la detección facial .....	20
<b>Figura 2.4:</b> Ejemplo de las características empleadas en el algoritmo Haar-Cascade.....	21
<b>Figura 2.5:</b> Nuevos modelos de detección facial y de objetos en los últimos años.....	21
<b>Figura 2.6:</b> Fórmula de la Intersección sobre la unión .....	22
<b>Figura 2.7:</b> Fórmula de la precisión .....	23
<b>Figura 2.8:</b> Fórmula del recall .....	23
<b>Figura 2.9:</b> Conjunto de validación de WIDER Face.....	23
<b>Figura 2.10:</b> Entrenamiento e inferencia de los modelos de aprendizaje profundo .....	24
<b>Figura 2.11:</b> Enfoques sobre el reconocimiento facial a lo largo de los años .....	25
<b>Figura 2.12:</b> Fases de entrenamiento e inferencia del reconocimiento facial.....	27
<b>Figura 2.13:</b> Evolución temporal de los benchmarks sobre reconocimiento facial.....	29
<b>Figura 3.1:</b> Matriz de riesgos .....	36
<b>Figura 4.1:</b> Diagrama de Gantt.....	41
<b>Figura 5.1:</b> Ejemplo de la compilación cruzada.....	46
<b>Figura 5.2:</b> Posible flujo inicial del subsistema.....	47
<b>Figura 5.3:</b> Flujo modificado del subsistema - conversión YUV a RGB .....	48
<b>Figura 5.4:</b> Resultados prueba básica.....	51
<b>Figura 5.5:</b> Gráfica de los tiempos de ejecución .....	52
<b>Figura 5.6:</b> Gráfica del tiempo para el pintado de las caras detectadas.....	53
<b>Figura 5.7:</b> Resultados de la prueba de eficacia.....	54
<b>Figura 5.8:</b> Código de la conversión manual.....	56
<b>Figura 5.9:</b> Nuevo flujo con el shader.....	57
<b>Figura 5.10:</b> Nuevo flujo para la paralelización.....	58
<b>Figura 5.11:</b> Resultado de la paralelización con el modelo S3FD .....	59
<b>Figura 5.12:</b> Consulta al GitHub de opencv_zoo .....	60
<b>Figura 5.13:</b> Flujo de las pruebas finales .....	62
<b>Figura 5.14:</b> Resultados prueba final de velocidad – Detección facial .....	63
<b>Figura 5.15:</b> Resultados prueba final de precisión – Detección facial .....	64
<b>Figura 5.16:</b> Tabla resultado de las pruebas de eficacia.....	65
<b>Figura 5.17:</b> Sistema de verificación de identidad .....	67
<b>Figura 5.18:</b> Flujo prueba de concepto.....	69
<b>Figura 5.19:</b> Proceso de reconocimiento.....	70
<b>Figura 5.20:</b> Grafica lectura de biometrías.....	72
<b>Figura 5.21:</b> Análisis del modelo de profundidad – MiDaS dpt_levit_224 .....	74
<b>Figura 5.22:</b> Prueba de velocidad para el peor caso.....	77
<b>Figura 5.23:</b> Resultado de la prueba de eficacia – Registro de usuario en el momento .....	78
<b>Figura 5.24:</b> Resultado de la prueba de eficacia – Registro de usuario previo .....	78

# Índice de Tablas

---

<b>Tabla 3.1:</b> Requisitos funcionales del sistema.....	31
<b>Tabla 3.2:</b> Requisitos no funcionales del sistema.....	32
<b>Tabla 3.3:</b> Descripción de los eventos de riesgo .....	35
<b>Tabla 3.4:</b> Caracterización de las amenazas .....	36
<b>Tabla 3.5:</b> Confección de respuestas ante las posibles amenazas.....	37
<b>Tabla 4.1:</b> Estimación inicial del plan de trabajo .....	40
<b>Tabla 4.2:</b> Presupuesto de los recursos humanos .....	42
<b>Tabla 4.3:</b> Presupuesto de los recursos hardware.....	42
<b>Tabla 4.4:</b> Presupuesto de los recursos software .....	42
<b>Tabla 4.5:</b> Presupuesto de final .....	43
<b>Tabla 5.1:</b> Comparación de bibliotecas y marcos de trabajo.....	45
<b>Tabla 5.2:</b> Modelos de detección facial analizados .....	49
<b>Tabla 5.3:</b> Modelos de detección facial analizados .....	52
<b>Tabla 5.4:</b> Modelos de detección facial analizados .....	53
<b>Tabla 5.5:</b> Nuevos requisitos funcionales del sistema.....	68
<b>Tabla 5.6:</b> Nuevos requisitos NO funcionales del sistema .....	68
<b>Tabla 5.7:</b> Resultados de la prueba de velocidad para el reconocimiento facial .....	76

## Glosario

---

**Detección facial** Proceso informático para identificar y localizar rostros humanos presentes en un fotograma.

**Reconocimiento facial** Proceso informático para identificar personas en un fotograma, a partir de los rasgos faciales de sus rostros.

**Videoportero** Sistema de seguridad que permite la comunicación audiovisual de personas que desean ingresar a una comunidad, véase una casa, un edificio, etc.

**Visión por computador** Campo que estudia el procesamiento de las imágenes captadas del mundo real con el objetivo de generar información de valor.

**Big data** Término que hace referencia a un gran conjunto de datos que puede analizarse con el motivo de revelar patrones, tendencias y asociaciones (especialmente en relación con el comportamiento y las interacciones humanas) para generar información de valor.

**Redes neuronales** Método de inteligencia artificial basado en el cerebro humano que crea un sistema adaptable para que los ordenadores puedan aprender de sus errores y mejorar continuamente.

**C++** Lenguaje de programación de alto nivel y de propósito general que deriva del lenguaje C, añadiendo funcionalidades adicionales como la programación orientada a objetos.

**Linux** Sistema operativo de código abierto que ofrece una gran configuración personalizable según las necesidades de los usuarios, además de poseer una gran comunidad activa de desarrollo.

**Sistema embebido** Sistema de computación basado en un microprocesador o un microcontrolador, el cual, está diseñado para realizar una o varias tareas específicas (generalmente en tiempo real).

**Neurofisiólogos** Rama de la biología y la psicología que se ocupa de estudiar los procesos del sistema nervioso.

**Corteza visual primaria** Área del cerebro que procesa la información visual.

**Aprendizaje profundo** Subconjunto de métodos de aprendizaje automático que tratan de reconocer patrones complejos y realizar asociaciones de manera similar a la de un humano.

**Red neuronal** Programa que emplea procesos que imitan el comportamiento de las neuronas biológicas para identificar, evaluar y llegar a conclusiones.

**Adboost** Técnica de aprendizaje automático que se basa en crear predictores sencillos en secuencia, para que cada uno mejore y ajuste lo que el anterior no pudo.

**Biometría facial** Tecnología que permiten identificar a una persona por características físicas únicas en su cara.

## Acrónimos

---

**ARM** Advanced RISC Machine

**GHz** Gigahertz ó Gigahercio

**MB** Megabyte

**RAM** Random Access Memory

**IA** Inteligencia Artificial

**CNN** Convolutional Neural Network

**AP** Average Precision

**DNI** Documento Nacional de Identidad

**PM** Project Manager



# Introducción

---

El presente documento pretende mostrar el Trabajo de Fin de Máster, cuyo principal objetivo reside en el análisis y la integración de funcionalidades relacionadas con la **detección** y el **reconocimiento facial** en un **videoportero** comercial. Este proyecto se ha llevado a cabo en colaboración con la empresa valenciana **Fermax**, y con motivo de la realización de mis prácticas de empresa. Fermax proporciona sistemas y soluciones para la conectividad y el control de acceso al hogar. Con 75 años de experiencia en el sector y con presencia en más de 70 países, hoy en día son líderes en la fabricación y comercialización de sistemas de porteros y videoporteros.



*Figura 1.1: Logo de la empresa Fermax*

Cabe destacar que dicha empresa ha sido la encargada de proporcionar todo el material necesario e indispensable para la realización de este proyecto. Por un lado, me ha suministrado el equipamiento necesario para llevar a cabo todas las tareas, principalmente un ordenador y un videoportero que presenta las siguientes especificaciones técnicas: Quad-core **ARM** Cortex-A7 @ 1.5 **GHz** | 32 bits | 512 **MB** de **RAM**. Y, por otro lado, me ha proporcionado un espacio de trabajo junto con los actuales empleados del departamento de I+D de la empresa. Esto supuso mi integración en un entorno real de trabajo, favoreciendo así mi desarrollo como profesional en el sector.

## 1.1 Motivación

---

Antes de iniciar las prácticas, se organizó una reunión con el tutor de la empresa colaboradora con motivo de elegir la temática principal del proyecto. En dicha reunión, se expusieron las diversas temáticas de los trabajos que había disponibles para elegir. Estos tenían como objetivo en común añadir funcionalidades innovadoras que aportasen valor a los sistemas de videoporteros actuales y futuros de la empresa. Finalmente, me decanté por elegir el proyecto que suponía realizar una labor de investigación sobre la implementación de funcionalidades de detección y reconocimiento facial en los videoporteros. Dicha elección estuvo sujeta a diversos factores. Por

un lado, tenía especial interés en estudiar y aprender sobre el sector de la **visión por computador** (también conocido como *Computer Vision* o, por sus siglas, **CV**). Tras el auge de las inteligencias artificiales y la expansión de los análisis de grandes volúmenes de datos (*big data*), este campo había sufrido un crecimiento exponencial en los últimos años. Y dado que, durante mis estudios, no había tenido opción de trabajar de primera mano con temáticas relacionadas al campo como, por ejemplo, utilizar modelos generados por **redes neuronales**, pensé que esta sería una muy buena oportunidad para ello. Por otro lado, la puesta en marcha de este proyecto suponía un desafío intelectual, ya que si bien es cierto que disponía de experiencia y conocimientos previos sobre el sistema y el lenguaje con los que iba a trabajar (implementación de código escrito en **C++** sobre un sistema **Linux** embebido), aplicar dichos conocimientos sobre un área desconocida para mí como lo era la visión por computador, iba a suponer un reto. Además, cabe destacar que el proyecto en sí presenta una clara dificultad añadida al tratar de implementar funcionalidades relacionadas con la detección y el reconocimiento facial sobre un **sistema embebido**. Esta dificultad viene dada por las restricciones computacionales de dichos sistemas, ya que, como se pudo demostrar en la sección anterior, el videoportero a utilizar dispone de recursos hardware limitados. Por último, cabe mencionar que una de las principales motivaciones fue la idea de poder contribuir a integrar nuevas funcionalidades para productos comerciales actuales o futuros. Como todo estudiante ansioso por comenzar su etapa en el mundo laboral, tener la posibilidad de iniciar en un proyecto real que pueda impactar positivamente en la experiencia del usuario con el producto, me llena de ilusión y motivación para comenzar con el trabajo. Con estas motivaciones en mente y, tras la elección de la temática del trabajo, el siguiente punto del proyecto sería establecer los principales objetivos de este.

## 1.2 Objetivos

---

En el contexto de un proyecto en colaboración con una empresa o un tercero, es fundamental situar el trabajo en su contexto inicial. En este caso, la idea del trabajo no surge de una necesidad imperiosa de la empresa por conseguir nuevos clientes o por diseñar el mejor videoportero del mercado. Sino que, más bien, se establece como una propuesta a largo plazo en la que, gracias a las conclusiones recopiladas una vez realizado el proyecto, se pueda analizar la viabilidad de implementar funcionalidades de visión por computador en los videoporteros actuales y futuros. Incluir dichas funcionalidades en sus productos supondría un gran beneficio para la empresa en diversos aspectos. Por ejemplo, se vería reforzada la seguridad de las viviendas tras incluir funcionalidades como el reconocimiento facial. Por otro lado, podrían aplicar una reducción de costes al implementar funcionalidades software que reemplacen componentes hardware (p.ej. sensores de proximidad). O, incluso, podrían verse beneficiados por el aumento del potencial de marketing del producto, ya que, como mencionamos previamente, el campo de la visión por computador es muy popular en la actualidad.

Si bien es cierto que las posibles ventajas son muchas y muy variadas, los desafíos que se han de afrontar para poder disfrutar de las mismas serán igual de relevantes. Por este motivo, es muy importante tener una clara definición de qué se pretende conseguir con este trabajo. Para ello, se definen los siguientes objetivos principales:

1. Llevar a cabo el análisis y el listado de las principales bibliotecas y/o marcos de trabajo utilizados en el campo de la visión por computador.

2. Seleccionar e integrar las principales bibliotecas y/o marcos de trabajo que puedan ser instalados y compilados en nuestro sistema objetivo (sistema Linux con arquitectura ARM).
3. Analizar, seleccionar e integrar modelos pre entrenados de detección facial que tengan compatibilidad con sistemas embebidos y/o puedan ejecutarse en tiempo real.
4. Diseñar y analizar los resultados de las pruebas para evaluar la eficacia y la velocidad de los modelos de detección facial integrados en el videoportero
5. Llevar a cabo la selección el mejor modelo de detección facial según los resultados de las pruebas.
6. Analizar, seleccionar e integrar los modelos de reconocimiento facial que se puedan integrar junto con la biblioteca y el modelo de detección seleccionado previamente.
7. Diseñar y analizar los resultados de las pruebas de eficacia y velocidad aplicadas a los modelos de reconocimiento facial seleccionados

## 1.3 Estructura de la memoria

---

Una vez claros los principales objetivos del proyecto, en los próximos capítulos se detallarán todas las tareas que se han realizado para cumplir con cada uno de ellos. Por lo tanto, podemos describir brevemente el contenido de cada uno de los próximos capítulos de la siguiente manera:

- **Capítulo 2: Estado del arte**, en este capítulo se pretende poner un poco más en contexto al lector sobre los principales conceptos que se van a tratar a lo largo del documento, sobre todo aquellos que tengan relación con el campo de la visión por computador.
- **Capítulo 3: Análisis del problema**, en esta sección se define de manera más extensa el problema a solventar, llevando a cabo el análisis desde diferentes perspectivas como la especificación de requisitos, análisis de seguridad, análisis de riesgos, etc.
- **Capítulo 4: Planificación de la solución propuesta**, en este capítulo se definen los bloques que constituyen la propuesta de trabajo, además de especificar la metodología y las herramientas que se emplearán para su puesta en marcha.
- **Capítulo 5: Desarrollo de la solución propuesta**, en este apartado se mostrará todo el trabajo realizado para la implantación de la solución, resaltando los resultados obtenidos, y documentando los aspectos relacionados con la gestión del cambio.
- **Capítulo 6: Conclusiones y trabajo futuro**, en este último capítulo se exponen las conclusiones del trabajo. Aquí se analiza si la solución propuesta ha cumplido con todos los objetivos planteados, además de hablar sobre los diversos problemas afrontados, las soluciones aplicadas y los futuros objetivos del proyecto.

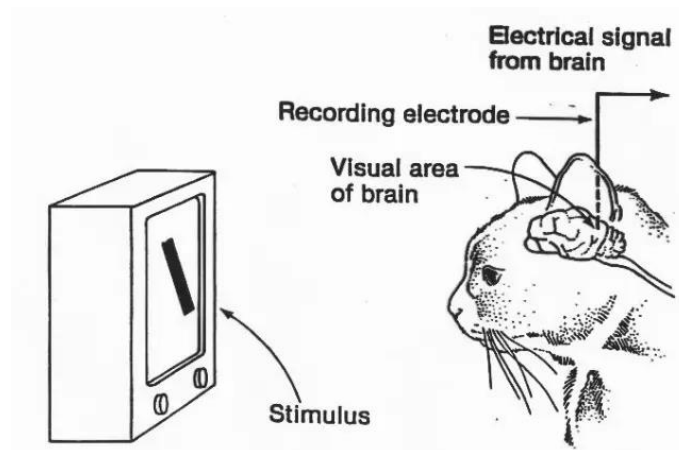
- **Bibliografía**, listado de fuentes originales como libros, artículos o revistas que se han utilizado para la redacción y/o elaboración del trabajo.
- **Anexos**, aspectos complementarios que no son necesarios para entender el contenido del documento de forma completa.



# Estado del arte

---

Como se ha comentado en la introducción, el principal campo a tratar en este documento será la visión por computador. A diferencia de lo que puede parecer en primera instancia, ya que el campo ha ganado terreno significativamente en los últimos años debido al crecimiento exponencial en la aplicación de técnicas de **IA**, este tiene una trayectoria de investigación con muchos años a sus espaldas (Demush, 2019). De hecho, muchos científicos e ingenieros coinciden en que el primer artículo en relación con la visión por computador fue publicado por dos **neurofisiólogos** a finales de los años 50. Los autores de dicha publicación (Hubel & Wiesel, 1959) pretendían mostrar cuál era la actividad neuronal de la zona de la **corteza visual primaria** del cerebro de un gato, mientras le mostraban diversas imágenes. Tras meses de experimentación, los investigadores establecieron que el cerebro respondía inicialmente a los elementos como bordes y líneas definidos. Esta conclusión sugiere que el procesamiento visual que realiza el cerebro comienza siempre con estructuras simples antes de abordar detalles más complejos. Siendo esta conclusión esencialmente el principio detrás de diversas técnicas como el **aprendizaje profundo** (también conocido como *deep learning*).



*Figura 2.1 : Explicación del experimento de Hubel & Wiesel*

A partir de la publicación de este artículo, podríamos mencionar una lista extensa de acontecimientos que impulsaron al crecimiento y la investigación del campo de la visión por computador. Por ejemplo, podemos mencionar la invención del primer **escáner digital** por Russel Kirsch en 1959, lo que permitió transformar las imágenes en matrices de números para que los ordenadores pudieran reconocerlas. Por otro lado, podríamos destacar también el artículo de uno de los padres de internet (Roberts, 1963) en donde se exploraron las posibilidades de extraer

información geométrica tridimensional, a partir de perspectivas bidimensionales. O incluso podríamos hablar de lo que muchos consideran como el nacimiento de la visión por computador como campo, el lanzamiento del proyecto Summer Vision Project (Papert, 1966). En dicho proyecto se tenía como principal objetivo desarrollar un sistema visual que pudiera segmentar el fondo de una imagen real y extraer objetos de ella.

Como hemos constatado, este campo se origina en un pasado mucho más remoto y engloba una mayor diversidad de conceptos de lo que generalmente se cree. Pero, suponiendo que necesitamos una definición que englobe todos los matices de este campo, ¿cuál sería la más adecuada? La empresa tecnológica multinacional **IBM** (International Business Machines Corporation) definió en 2023 el campo de la visión por computador como un campo de la IA que utiliza el **aprendizaje automático** (también conocido como *machine learning*) y las **redes neuronales** para enseñar a ordenadores y sistemas a extraer información significativa o valiosa de imágenes digitales, vídeos y otros tipos de entradas visuales (IBM, 2021). Aunque dicha definición sea correcta desde el punto de vista teórico, podemos emplear otra frase que se menciona en el mismo documento para una mejor comprensión: “*If AI enables computers to think, computer vision enables them to see, observe and understand*”. Esta frase nos sugiere el propósito que se persigue realmente con la investigación en este campo, siendo este poder replicar y mejorar (por medio de ordenadores y cámaras) las funciones básicas que realizamos cotidianamente con nuestra vista, conocimiento y contexto evolutivo. Siendo aún más específicos, la visión por computador pretende, por medio de datos, algoritmos y cámaras, realizar funciones como detectar objetos, discernir la distancia a la que se encuentran dichos objetos, reconocer a personas a través de los rasgos faciales y, en definitiva, llevar a cabo una extensa lista de funciones que nosotros, como humanos, podemos realizar a través de nuestras retinas, nervios ópticos y córtex visual.

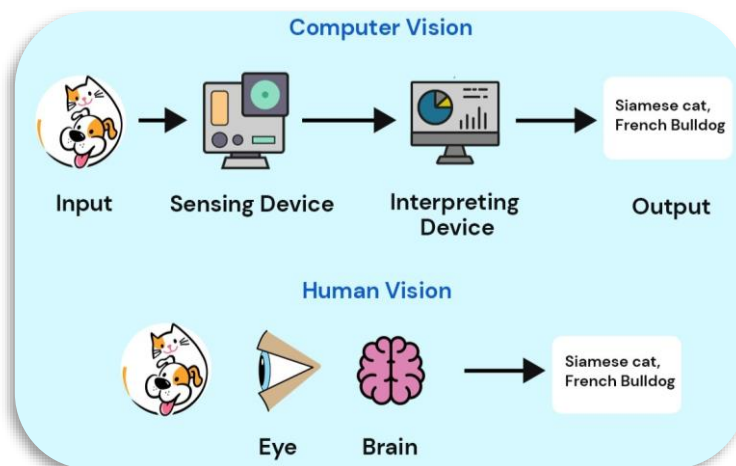


Figura 2.2: Diferencias entre la visión por computador y la visión humana

Hoy en día las aplicaciones reales del campo de la visión por computador en negocios como el ocio, el transporte o la sanidad, son muy extensas. Uno de los factores clave, además de los obvios, como la mejora constante del hardware de los ordenadores que nos proporciona mejores y mayores prestaciones, radica en el hecho de que actualmente tenemos acceso a una gran avalancha

de información gracias al uso de herramientas visuales tales como teléfonos móviles, sistemas de seguridad, cámaras de tráfico, etc. Las organizaciones y empresas compiten por acceder a estas grandes cantidades de datos, ya que, con un correcto procesamiento de la información, se puede generar mucho valor. Del mismo modo, algunas de las funciones más destacadas que pretende resolver esta disciplina son:

1. **Detección y clasificación de objetos:** área que estudia cómo localizar (distinguir en qué posición se encuentra) y clasificar (discernir si es un perro, una persona, una manzana, etc.) objetos en una imagen o video. Muy utilizada en áreas como la conducción autónoma o la videovigilancia.
2. **Seguimiento de objetos:** aplicación que permite realizar el seguimiento de objetos en imágenes o videos. Suele aplicarse junto con la aplicación anterior para sectores como la videovigilancia, la robótica o la realidad aumentada.
3. **Reconocimiento de expresiones faciales:** esta aplicación permite discernir qué expresión facial posee una persona en una imagen o video, siendo algunas de las opciones: feliz, neutra, disgustada, triste, etc. Esta técnica es empleada principalmente para estudios sociológicos o en animaciones 3D de personajes.
4. **Estimación de la pose:** proceso con el cual se predicen y determinan las ubicaciones de las articulaciones y otros puntos clave del cuerpo humano a partir de una imagen o video. Muy utilizada en el sector de la salud, estudiando los movimientos corporales de los pacientes para aplicaciones médicas y terapéuticas.
5. **Detección de textos:** aplicación que permite encontrar áreas de texto en imágenes, pudiendo ser caracteres impresos, números, logotipos, etc. Esta técnica es utilizada en aplicaciones prácticas como la traducción automática de textos empleando la cámara de los móviles.

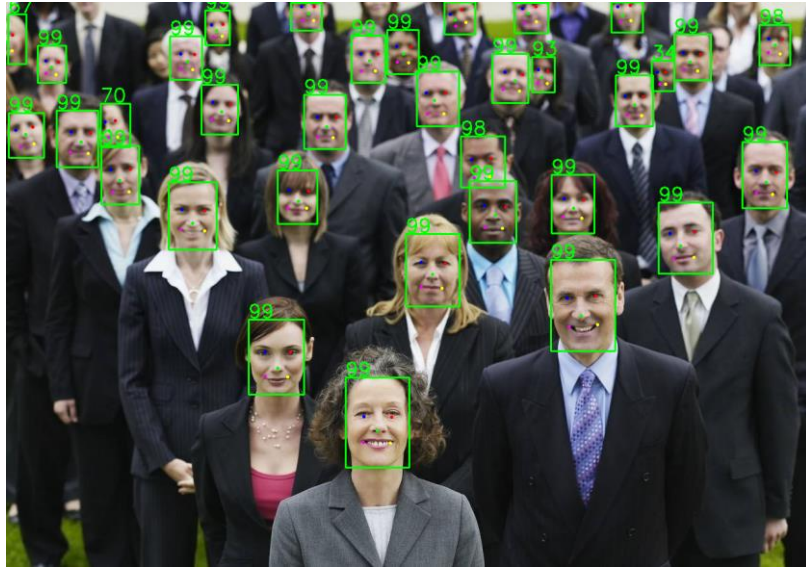
Estas no son todas las aplicaciones reales que tiene este campo, de hecho, se han omitido deliberadamente dos de las más importantes, la **detección** y el **reconocimiento facial**. A continuación, se profundizará en cada una de ellas, ya que una correcta comprensión de estas es fundamental para alcanzar los objetivos principales de este proyecto.

## 2.1 Detección facial

---

La detección facial es una de las funciones más prácticas y fundamentales de la visión por computador. Esta tarea consiste en identificar rostros humanos en imágenes devolviendo las ubicaciones de estos mediante cuadros delimitadores como el de la figura 2.3. Si hablamos de las aplicaciones prácticas de emplear esta técnica debemos destacar que, aunque existan áreas en las que se emplea únicamente esta tarea (p.ej. el análisis del tráfico de personas que circulan por una determinada calle), la detección facial es principalmente utilizada como el primer paso para llevar a cabo otras tareas relacionadas con el rostro humano como, por ejemplo, el reconocimiento facial, el seguimiento facial, el reconocimiento de expresiones faciales, la detección de puntos de

referencia faciales, etc. Teniendo en cuenta que la tecnología evoluciona cada vez más rápido, junto a ella también han evolucionado los algoritmos y enfoques empleados para detectar y analizar características faciales. Esto sugiere que, si somos capaces de desarrollar detectores faciales más rápidos y precisos, podremos esperar una mejora significativa en todas aquellas tareas dependientes de la detección facial (Yuantao, Shiqi, Hanyang, Yan-Ran, & Jianguo, 2022).



*Figura 2.3: Resultado de ejemplo tras aplicar la detección facial*

La manera en la que se lleva a cabo la detección facial ha experimentado un notable avance en las últimas décadas. Sin embargo, si tuviéramos que señalar un evento en concreto que actuó como punto de inflexión, marcando un cambio significativo en este periodo, sin duda sería la publicación del artículo *Rapid Object Detection using a Boosted Cascade of Simple Features* (Viola & Jones, 2001). Este trabajo incluyó un enfoque innovador que sentó las bases para realizar la detección facial de manera rápida y eficiente. Los conceptos clave para poder conseguir tal hazaña fueron la inclusión de una nueva forma de representación de imágenes llamada **imagen integral**, proporcionando así una mayor velocidad en el cálculo de las características empleadas por el detector. Por otro lado, tenemos el uso de un algoritmo de aprendizaje basado en **AdaBoost** que permitía seleccionar características visuales críticas para producir clasificadores más eficaces. Y, por último, un nuevo método para combinar clasificadores cada vez más complejos formando una “**cascada**”, lo que facilitaba el descarte de regiones de fondo de la imagen y así se dedicaba más tiempo de cálculo a las regiones prometedoras similares a objetos.

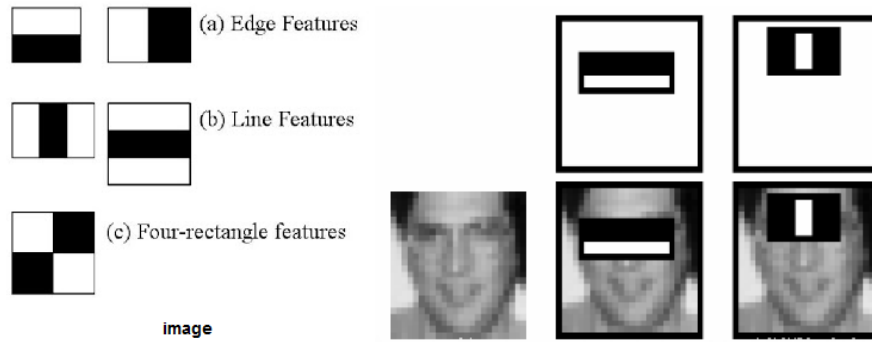


Figura 2.4: Ejemplo de las características empleadas en el algoritmo Haar-Cascade

Durante varios años los métodos basados en características como, el clasificador AdaBoost en cascada, fueron los métodos dominantes para la detección facial. Sin embargo, con la aparición de las técnicas basadas en aprendizaje profundo, esta situación cambió significativamente. Estas nuevas técnicas emplean redes neuronales (p. ej. CNN) para entrenar modelos utilizando enormes bases de datos de imágenes de rostros, consiguiendo así aprender automáticamente sobre las características necesarias para representarlos. Por consiguiente, estos métodos mostraron resultados mucho más precisos y robustos, ya que, a diferencia de los métodos basados en características, los métodos basados en aprendizaje profundo logran identificar rostros incluso en situaciones complejas como con cambios de poses, variaciones en la iluminación, oclusión parcial de las caras, etc.

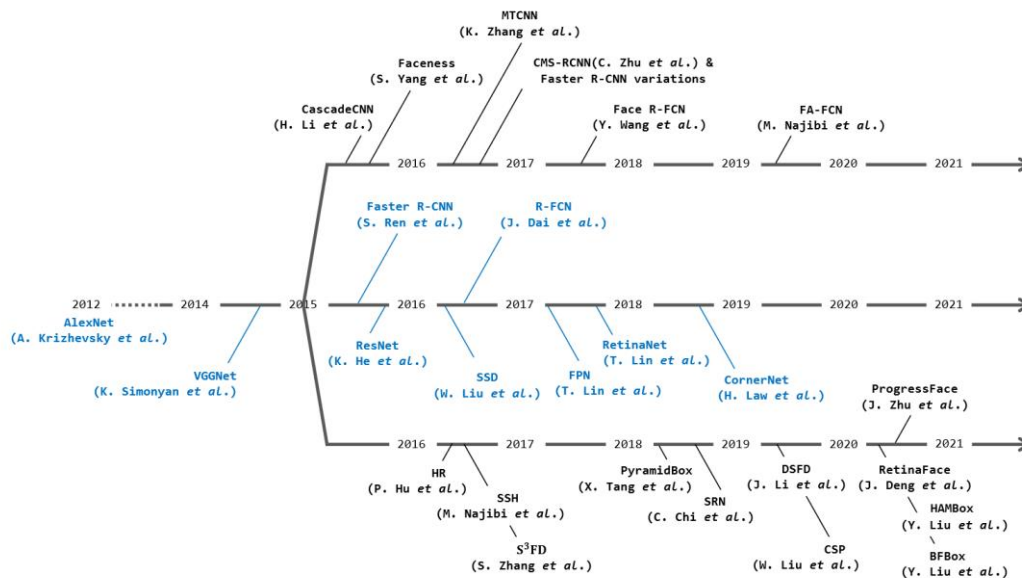


Figura 2.5: Nuevos modelos de detección facial y de objetos en los últimos años

Otras áreas mucho más desafiantes que la detección de rostros como la detección genérica de objetos, se han visto gratamente beneficiadas tras aplicar nuevas metodologías basadas en aprendizaje profundo. De hecho, tal y como se aprecia en la figura 2.5, en los últimos años se han diseñado numerosos modelos para la detección de objetos (línea central de la figura) y, junto a estos, se puede apreciar que la tendencia actual de los nuevos modelos de detección facial (línea superior e inferior de la figura) es adoptar o inspirar el enfoque de estos nuevos modelos de detección de objetos. Aunque en este trabajo no vamos a profundizar mucho más sobre estos conceptos, siendo más específicos, los actuales detectores de rostros basados en técnicas de aprendizaje profundo se basan en tres categorías: Marcos de trabajo de múltiples etapas, de dos etapas y de una etapa (Feng, Yu, Peng, Li, & Zhang, 2022).

Los enfoques que se crearon para resolver la tarea de detección facial a partir del uso de métodos basados en aprendizaje profundo generaron la necesidad de establecer nuevos métodos rigurosos para evaluar su rendimiento. De esta manera, se podrían comparar y determinar cuál enfoque resuelve mejor la tarea dependiendo de sus resultados. Para evaluar los diferentes algoritmos de detección facial se emplean grandes bases de datos de imágenes a color que plasman situaciones cotidianas de la vida real. Existen un gran número de bases de datos públicas como Fddb (Jain & Learned-Miller, 2021), AFW (Xiangxin & Ramanan, 2012), PASCAL Face (Yan, Zhang, Lei, & Li, 2014), etc. Aunque sin duda, desde su aparición en 2016, la base de datos **WIDER Face** (Yang, Luo, Loy, & Tang, 2015) ha sido el banco de datos más popular y utilizado para evaluar modelos de detección facial. Esta base de datos posee la asombrosa cantidad de **32,202 imágenes**, en las cuales han sido etiquetadas **393,703 caras** con diversos grados de variabilidad en cuanto a la pose, la escala o la oclusión. De hecho, algo que hace resaltar a WIDER Face sobre el resto de las bases de datos, es que presta especial atención a la detección de rostros pequeños, proporcionando casi el 50 % de las caras etiquetadas con un tamaño de entre 10 a 50 píxeles. Para evaluar el rendimiento de los algoritmos, los usuarios deben entrenar sus modelos empleando datos externos o los conjuntos proporcionados por WIDER Face. A continuación, deben probar sus modelos con el subconjunto para pruebas y enviar los resultados de la detección para su evaluación. WIDER Face adopta las mismas métricas de evaluación que la base de datos PASCAL VOC (Everingham, y otros, 2015), de entre las que principalmente destacamos las siguientes medidas:

- **Intersección sobre unión (IoU):** También conocido como **índice de Jaccard**, mide la superposición entre la predicción (rostros detectados con el modelo a evaluar) y la verdad fundamental (*ground truth* o, en nuestro caso, rostros etiquetados manualmente) en términos de área.

$$IoU = \frac{area(P) \cap area(GT)}{area(P) \cup area(GT)}$$

Figura 2.6: Fórmula de la Intersección sobre la unión

- **Precisión:** Mide la proporción de predicciones positivas correctas en relación con todas las predicciones positivas realizadas por el modelo (verdaderos positivos + falsos positivos).

$$Precision_n = \frac{TP_n}{TP_n + FP_n}$$

Figura 2.7: Fórmula de la precisión

- **Recall:** También conocido como tasa de verdaderos positivos, mide la capacidad del modelo para detectar correctamente los ejemplos positivos, utilizando la proporción de verdaderos positivos en relación con los verdaderos positivos y los falsos negativos.

$$Recall_n = \frac{TP_n}{TP_n + FN_n}$$

Figura 2.8: Fórmula del recall

- **Curva de Precisión-Recall (PR):** Esta curva representa cómo varía la precisión (eje y) con respecto al *recall* (eje x) a diferentes umbrales de confianza. Normalmente se calcula el área bajo esta curva para representar el rendimiento global de un detector de caras. Al valor del área bajo la curva se denomina comúnmente como la precisión promedio (**AP**), y sirve para resumir la curva en un solo valor. WIDER Face utiliza principalmente este valor para evaluar el rendimiento de los modelos para cada uno de sus subconjuntos de prueba (*Easy*, *Medium* y *Hard*).

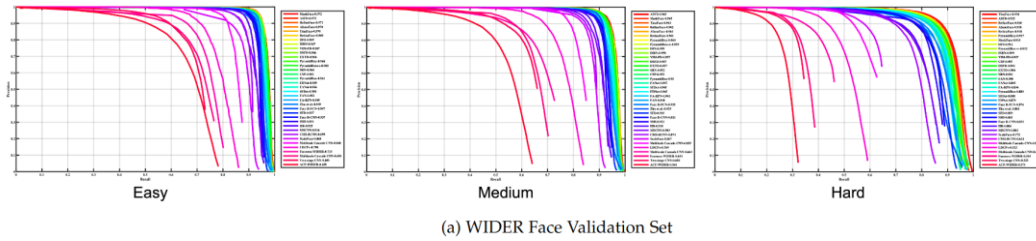


Figura 2.9: Conjunto de validación de WIDER Face

Antes de pasar al siguiente apartado, es conveniente explicar una serie de conceptos relacionados con la detección facial que se mencionarán en futuros capítulos de este trabajo. Para la comprensión del alcance de este proyecto, es fundamental conocer las diferencias entre los **algoritmos de entrenamiento** para la detección facial y los **algoritmos de inferencia**. Los algoritmos empleados para el entrenamiento son los responsables de enseñar a un modelo de inteligencia artificial a aprender sobre los datos. En el transcurso del entrenamiento, el modelo debe ir ajustando progresivamente sus parámetros (como pesos o sesgos) para descubrir patrones o reglas que relacionen los datos de entrada con los resultados deseados, en nuestro caso, detectar las caras de una imagen. Como resultado del entrenamiento, se suelen obtener dos ficheros: el **fichero de configuración** y el de **pesos** (aunque en algunos casos ambos ficheros se juntan en uno solo). El fichero de configuración almacena todos los datos relativos a la arquitectura de la



red neuronal utilizada para la detección, ya sea información sobre las capas, conexiones, etc. Por otro lado, el fichero de pesos contiene los pesos (conexiones entre las neuronas) y sesgos (términos adicionales) aprendidos por la red neuronal durante el entrenamiento. Una vez finalizado el entrenamiento y obtenidos los ficheros de salida, entran en juego los algoritmos de inferencia. Estos son los encargados de **aplicar el modelo generado** durante el entrenamiento a **nuevos datos** (imágenes) con el propósito de detectar caras. Durante la inferencia, se aplican los pesos calculados en el entrenamiento a las entradas de datos para obtener las predicciones finales. Generalmente, en el área de detección facial, estas predicciones se basan en las coordenadas finales de las caras detectadas junto con un valor de confianza. Las coordenadas suelen tener la siguiente forma:  $\{x, y, width, height\}$ , en donde  $\{x,y\}$  hace referencia al punto superior izquierdo del cuadro delimitador de la cara,  $width$  el ancho de la cara, y  $height$  el alto de la cara. Por otro lado, el valor de confianza hace referencia al porcentaje de seguridad con el que la red neuronal afirma que en las coordenadas devueltas hay una cara (p. ej. un valor de confianza del 100 % indica que en las coordenadas devueltas se ha encontrado con total seguridad una cara).

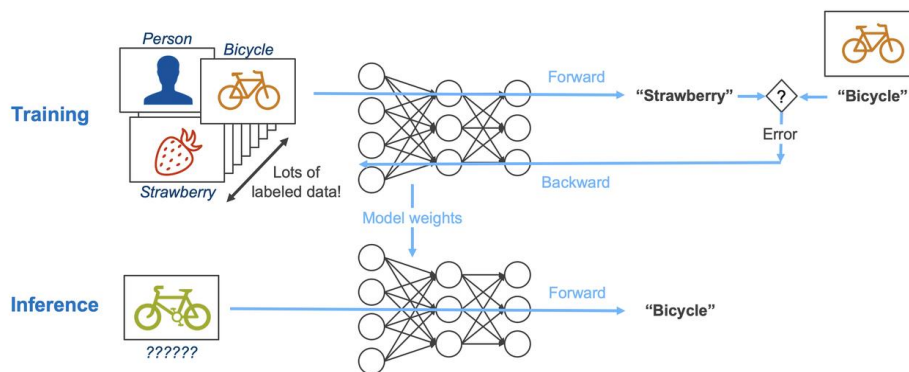


Figura 2.10: Entrenamiento e inferencia de los modelos de aprendizaje profundo

Una vez explicadas las diferencias, cabe destacar que dada la naturaleza del trabajo y con motivo de cumplir con los objetivos de este, se han empleado únicamente algoritmos de inferencia con modelos de detección facial pre entrenados. Por lo que, no se ha llevado a cabo ningún trabajo que tenga relación con el proceso de entrenamiento y confección de un modelo propio que cumpla con los requisitos exigidos por el trabajo.

## 2.2 Reconocimiento facial

El reconocimiento facial es una de las áreas de investigación más importantes en el campo de la inteligencia artificial y la visión por computador. Esta tarea consiste en identificar y verificar individuos basándose en sus propios rasgos y características faciales (p. ej. distancia entre los ojos, forma de la nariz y mandíbula, etc.). En las últimas tres décadas, el uso de esta aplicación se ha popularizado debido a una serie de factores: en primer lugar, tenemos la gran variedad de aplicaciones comerciales que tiene el uso de esta técnica, como, por ejemplo, su uso para controlar el acceso seguro a viviendas, oficinas o áreas restringidas (reemplazando métodos tradicionales como tarjetas o pines), como método de autenticación biométrica para móviles y muchos más usos en sectores como las finanzas o la milicia. Por otro lado, aunque esta tarea ha aportado



avances significativos en varias áreas tecnológicas, su uso también ha planteado una gran variedad de problemas éticos. La mayoría de estos problemas están relacionados con una falta de transparencia y consentimiento a las personas afectadas por el uso de estas aplicaciones, como, por ejemplo, con cámaras de videovigilancia. Esta situación genera inquietud en muchas personas, ya que implica una vulneración de su privacidad y de sus datos personales al capturar y analizar sus rostros sin consentimiento.

Otro factor importante ha sido la disponibilidad de nuevas tecnologías capaces de aumentar el rendimiento y, por consiguiente, mejorar los resultados al aplicar el reconocimiento facial. Al igual que con la detección facial, el uso de nuevas técnicas basadas en aprendizaje profundo ha marcado un antes y un después en este campo. Por último, cabe destacar que parte del reconocimiento y aceptación de la población hacia el uso de estas técnicas ha sido influenciada por las constantes referencias en medios de entretenimiento como series o películas. Esto se debe a que en la gran mayoría de obras audiovisuales (tales como *Los Vengadores*, *Terminator 2*, *Coco*, ...) se presenta el uso de estas técnicas como un gran avance tecnológico, el cual, otorga un alto grado de seguridad y abre la puerta a la integración de una gran variedad de funcionalidades creadas a partir de estas.

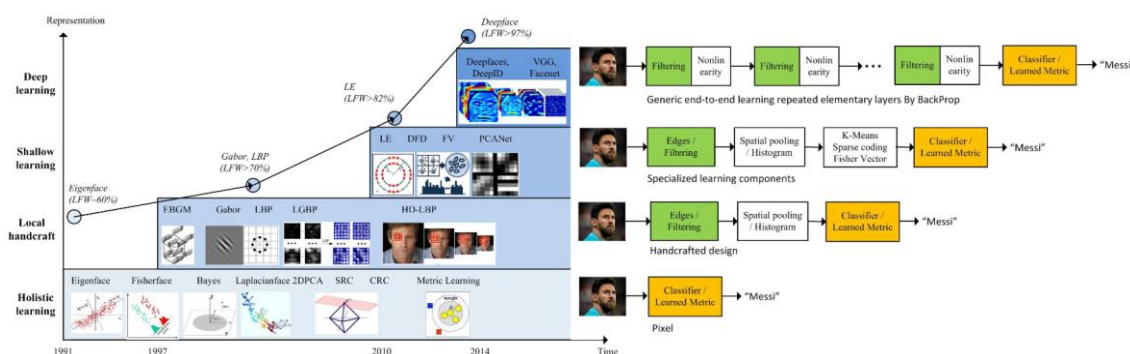


Figura 2.11: Enfoques sobre el reconocimiento facial a lo largo de los años

Tal como se aprecia en la figura 2.11, desde 2014 se han producido grandes avances en el área del reconocimiento facial al incluir el uso de **redes neuronales convolucionales profundas**. Sin embargo, ya desde la década de los 90 el reconocimiento facial iba evolucionando y adoptando nuevos y diversos enfoques (Wang & Deng, 2021):

- **Aprendizaje Holístico (Holistic Learning):** A principios de 1990, los enfoques holísticos dominaban el campo. Dichos métodos consideraban todo el rostro como una única entidad, denominada **patrón holístico**. Uno de los métodos más notables fue el enfoque de EigenFace (Turk & Pentland, 1991), el cual, utilizando técnicas de álgebra lineal, extraía características de baja dimensión a partir de imágenes faciales. El principal problema de este enfoque eran las dificultades para reconocer caras que estuvieran en entornos no controlados (cambios en condiciones ambientales, postura, expresión, ...).
- **Descriptores Locales Manuales (Handcrafted Local Descriptors):** En la época de los 2000 se popularizaron los enfoques basados en descriptores locales y se introdujeron los

métodos de aprendizaje de características locales. Estos métodos se centran en extraer características específicas de regiones localizadas de la cara, pero, al igual que el enfoque holístico, estos tenían problemas para detectar caras con variaciones de pose, iluminación, etc. Algunos de los métodos más conocidos de este enfoque son **LBP** (*Local Binary Patterns*) (Ahonen, Hadid, & Pietikainen, 2006) o **HOG** (*Histogram of Oriented Gradients*) (Dalal & Triggs, 2005).

- **Aprendizaje Superficial** (*Shallow Learning*): A principios de 2010, los descriptores locales basados en métodos de aprendizaje superficial fueron introducidos al campo. Estos métodos, a diferencia de los actuales métodos de aprendizaje profundo, emplean arquitecturas mucho más simples y con muchas menos capas. Para este enfoque se utilizaban algoritmos tradicionales de aprendizaje automático como **SVM** (*Support Vector Machines*) o **Random Forests**. Y, al igual que en los casos anteriores, este nuevo enfoque proporcionaba mejores resultados, pero seguía teniendo limitaciones con las variaciones de escenarios del mundo real.
- **Aprendizaje Profundo** (*Deep Learning*): La llegada de las técnicas basadas en aprendizaje profundo revolucionó el campo del reconocimiento facial, pero, el punto de inflexión más grande no llegaría hasta 2014 en donde el modelo **DeepFace** (Taigman, Yang, Ranzato, & Wolf, 2014) logró una precisión nunca antes vista en el famoso *benchmark* **LFW** (*Labeled Faces in the wild*) (Huang, Ramesh, Berg, & Learned-Miller, 2007), acercándose por primera vez al rendimiento humano (**DeepFace**: 97,35 % vs. **Humano**: 97,53 %) tras haber entrenado el modelo empleando 9 capas y cerca de 4 millones de imágenes faciales. Inspirados por estos resultados, los investigadores han optado por centrarse en emplear técnicas basadas en aprendizaje profundo en casi todos los aspectos del reconocimiento facial: diseño de algoritmos, datos de entrenamiento, escenarios de aplicación, etc. Además, con la llegada de nuevos modelos como **FaceNet** (Schroff, Kalenichenko, & Philbin, 2015) y otras arquitecturas como **ResNet** (He, Zhang, Ren, & Sun, 2015) o **MobileNet** (Howard, y otros, 2017), se ha logrado mejorar gratamente los resultados obtenidos anteriormente con imágenes que incluyen variaciones en pose, iluminación, etc.

Una vez tenemos en cuenta cuales son los enfoques actuales para el campo del reconocimiento facial, vamos a enfocarnos en describir las etapas necesarias para obtener los resultados de la identificación de un rostro. Al igual que hemos comentado en el apartado anterior con la detección facial, no entraremos en detalle sobre los procesos de entrenamiento de un modelo de reconocimiento facial basado en técnicas de aprendizaje profundo. Aunque, en la figura 2.12, se resume visualmente las etapas necesarias para dicho entrenamiento.

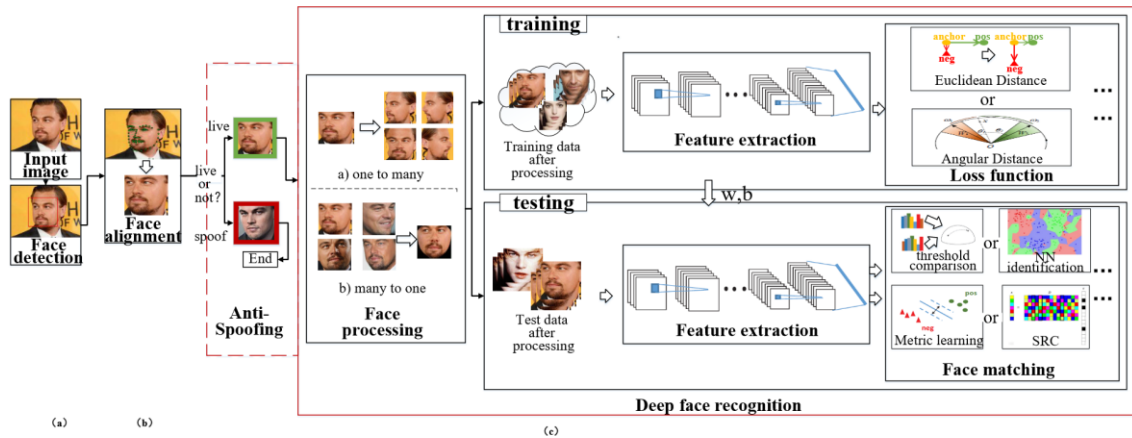


Figura 2.12: Fases de entrenamiento e inferencia del reconocimiento facial

Como podemos observar en la figura 2.12, la obtención del resultado tras aplicar el reconocimiento facial mediante el modelo entrenado, consta de tres etapas: en primer lugar tenemos el preprocesamiento de las imágenes, tras esto, comienza la etapa de extracción de las características faciales de cada una de las caras detectadas en la imagen, y, por último, tendríamos el proceso para analizar las correspondencias de las características faciales obtenidas de la imagen analizada, con las demás imágenes previamente etiquetadas (imágenes preprocesadas que suelen estar almacenadas en una base de datos). Siendo más específicos, podemos describir qué acciones se llevan a cabo en cada una de las etapas de la siguiente manera (Wang, y otros, 2022):

1. **Detección facial** (*fase preprocesing*): Como comentamos en el apartado anterior, la detección facial es comúnmente conocida por ser la primera fase de otras tareas relacionadas con el rostro humano y, en el caso del reconocimiento facial, esta no es una excepción. En esta primera etapa del preprocesamiento de la imagen se realiza la detección de caras junto con los puntos de referencia de estas (ojos, nariz y boca). Como resultado final, obtenemos las coordenadas de cada cara (con el formato anteriormente comentado:  $\{x,y,weight,height\}$ ) junto con las coordenadas de cada uno de los puntos de referencia.
2. **Suplantación de identidad** (*fase preprocesing*): En las aplicaciones comerciales del reconocimiento facial es muy importante comprobar si las caras introducidas son reales o no, ya que podrían tratarse de imágenes impresas, vídeos o máscaras 3D utilizadas para burlar el sistema y suplantar la identidad de otra persona. Por eso, una vez realizada la detección facial comienza la etapa de prevención ante una posible suplantación de identidad (también conocido como *anti-spoofing*). Hoy en día existen numerosos enfoques empleados para abordar este desafío (IQA (Solomon & Cios, 2023), STDN (Liu, Stehouwer, & Liu, 2020), operadores convolucionales, detección de parpadeo, etc.), pero, a pesar de los avances de estos algoritmos, los métodos de ataque también se actualizan, recordándonos que seguimos con la necesidad de continuar mejorando la seguridad y robustez de los sistemas de reconocimiento facial.
3. **Alineación de caras** (*fase preprocesing*): Las caras detectadas en la primera fase suelen variar debido a varios factores como pueden ser la pose o la perspectiva, reduciendo

significativamente la eficacia del reconocimiento. Por este motivo, la alineación de las caras es una técnica eficaz para paliar este problema. Un conjunto de imágenes faciales pertenecientes a una misma identidad tendrá una diferencia menor una vez se haya aplicado la alineación facial, haciendo que el clasificador sea más discriminativo. Comúnmente se utilizan **transformaciones 2D** (ya que son más rápidas que sus contrapartes: transformación 3D y Redes de Transformación Espacial) para calibrar los puntos de referencia faciales con las plantillas frontales predefinidas. A este tipo de transformación se le conoce como **transformación afín**. Cabe destacar que recientemente algunos métodos de reconocimiento facial optan por omitir esta fase, al realizar un mejor entrenamiento con una mayor cantidad de caras con variedad de poses, perspectivas, etc.

4. **Proceso de inferencia** (*feature extraction*): Una vez se ha terminado la etapa del preprocesamiento de la imagen, se procede a aplicar el modelo entrenado a la imagen. Al igual que los modelos de detección facial, los modelos de reconocimiento facial siguen generalmente un formato de dos ficheros: uno de configuración donde se describe la arquitectura de la red y otro que almacena los valores de los pesos aprendidos durante el entrenamiento. El resultado de esta etapa devuelve la **biometría facial** (comúnmente conocida como *face embeddings*) correspondiente a cada cara, siendo esta una representación matemática única para permitir la comparación con el resto de las caras.
5. **Comparación de características faciales** (*face matching*): En la última parte del proceso de reconocimiento facial se aplica la comparación de las biometrías faciales. Esta, a su vez, se puede dividir en dos partes: la primera etapa se basa en la **verificación**, en la que se determina si la cara procesada ‘A’ pertenece a la misma identidad que la cara ‘B’, la cual ha sido previamente preprocesada y almacenada en una base de datos (**problema 1:1**). Por otro lado, tenemos la etapa de la **identificación**, en la que se analiza si la cara procesada ‘A’ posee la misma identidad que al menos una cara del conjunto de caras preprocesadas ‘S’ (**problema 1:N**).

El último punto por tratar en este apartado sobre el reconocimiento facial son las bases de datos o *benchmarks* para comprobar el rendimiento de los modelos. En las últimas tres décadas se han elaborado una enorme cantidad de bases de datos de rostros (figura 2.13) con diversas características como diferencias de escala, diversidad de ambientes, variedad de fuentes para la obtención de imágenes, etc. Y es que, a medida que el rendimiento de algunas de las bases de datos se iba saturando (p. ej. la base de datos LFW (Huang, Ramesh, Berg, & Learned-Miller, 2007), uno de los *benchmarks* más conocidos y utilizados para evaluar el rendimiento antes de la aparición de las técnicas de aprendizaje profundo) se han ido desarrollando bases de datos cada vez más complejas (p. ej. MS-Celeb-1M (Guo, Zhang, Hu, & He, 2016), VGGface2 (Cao, Shen, Xie, Parkhi, & Zisserman, 2017), and Megaface (Kemelmacher-Shlizerman, Seitz, Miller, & Brossard, 2015)), pudiendo incluso afirmar que el desarrollo de dichas bases de datos ha dirigido en gran medida el rumbo de la investigación sobre el campo del reconocimiento facial.

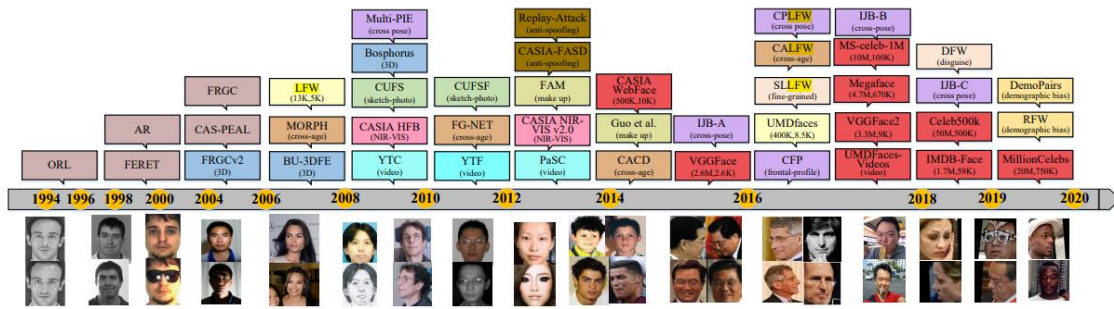


Figura 2.13: Evolución temporal de los benchmarks sobre reconocimiento facial

# Análisis del problema

El objetivo principal de esta sección se basa en presentar un detallado análisis del problema empleando técnicas o métodos adecuados según la naturaleza del proyecto. Por este motivo, se llevará a cabo la especificación de los requisitos del proyecto, donde se definirá lo que nuestra solución será capaz de hacer. Tras esto, dada la implicación del proyecto en dilemas éticos y legales, se realizará un análisis del marco legal y ético. Y, por último, se llevará a cabo un análisis de riesgos en donde se expondrá, principalmente, el impacto y las medidas de los posibles riesgos producidos por la futura integración de este trabajo en un proyecto de mayor envergadura.

## 3.1 Especificación de requisitos

A la hora de listar los diversos requisitos del sistema se elaborarán dos tablas dependiendo de si se tratan de requisitos funcionales o no funcionales (tabla 3.1 y 3.2). Del mismo modo, en ambas tablas se definirá la descripción del requisito junto con su propio identificador:

ID	Descripción del requisito funcional
RF-1	Integración de las principales bibliotecas o marcos de trabajo en el sistema embebido
RF-2	Integración de los modelos de detección facial seleccionados en el sistema embebido
RF-3	Proporcionar una interfaz de código para utilizar el modelo de detección facial deseado
RF-4	Permitir la configuración de parámetros específicos según el modelo de detección empleado
RF-5	Permitir ejecutarse independiente del hardware específico que se emplee para la obtención de los fotogramas
RF-6	Permitir obtener los fotogramas para detectar caras a través de imágenes o videos
RF-7	Permitir aplicar los algoritmos de detección empleando fotogramas con distintas resoluciones
RF-8	Evaluar si los modelos de detección facial cumplen con el objetivo de velocidad establecido
RF-9	Evaluar si los modelos de detección son capaces de detectar caras con accesorios faciales puestos
RF-10	Evaluar si los modelos de detección son capaces de detectar caras en imágenes donde dichas caras se encuentran parcialmente ocluidas

RF-11	Evaluar si los modelos de detección son capaces de detectar caras en imágenes bajo distintas condiciones lumínicas
RF-12	Evaluar si los modelos de detección son capaces de detectar caras con variaciones de pose
RF-13	Evaluar si los modelos de detección son capaces de detectar más de cinco caras en una imagen
RF-14	Integración de los modelos de reconocimiento facial compatibles con el modelo de detección y la biblioteca utilizada
RF-15	Proporcionar una interfaz de código para utilizar el modelo de reconocimiento facial deseado
RF-16	Permitir la configuración de parámetros específicos según el modelo de reconocimiento empleado
RF-17	Evaluar si los modelos de reconocimiento facial cumplen con el objetivo de velocidad establecido
RF-18	Evaluar si los modelos de reconocimiento son capaces de reconocer caras con accesorios faciales puestos
RF-19	Evaluar si los modelos de reconocimiento son capaces de reconocer caras con distintas expresiones faciales
RF-20	Evaluar si los modelos de reconocimiento son capaces de reconocer caras con variaciones en la pose
RF-21	Evaluar si los modelos de reconocimiento son capaces de reconocer caras con diferentes características personales

*Tabla 3.1: Requisitos funcionales del sistema*

ID	Descripción del requisito NO funcional
RNF-1	Las bibliotecas o marcos de trabajo seleccionadas deben ser compatibles para su compilación e integración en un sistema Linux embebido
RNF-2	Todo el código que se utilice para la elaboración del trabajo (bibliotecas, marcos de trabajo, modelos, etc.) debe de ser software de código abierto y con licencia apta para la comercialización
RNF-3	Los modelos de detección seleccionados deben ser capaces de ejecutarse en sistemas de tiempo real, como el sistema embebido objetivo
RNF-4	Tanto el código que se implemente para la solución, como el código de terceros integrado, debe estar escrito en el lenguaje de programación de C++
RNF-5	La documentación del código implementado se realizará utilizando el formato de comentarios de <b>Doxygen</b>
RNF-6	El modelo de detección seleccionado debe ser capaz de ejecutarse con un tiempo cercano a 40 milisegundos
RNF-7	El modelo de detección seleccionado debe ser capaz de detectar caras, aunque estas lleven accesorios faciales como gorros, gafas de vista, gafas de sol y mascarillas
RNF-8	El modelo de detección seleccionado debe ser capaz de detectar caras en entornos de luminosidad variable como durante la noche o con luz natural de lado, de frente y de espaldas
RNF-9	El modelo de detección seleccionado debe ser capaz de detectar caras con variabilidad de pose, por ejemplo, caras frontales o laterales

RNF-10	La detección de la cara en las pruebas pertinentes se hará situando la cara a un mínimo de 25 cm y un máximo de 1 m con respecto a la cámara del videoportero
RNF-11	Las pruebas de luminosidad para simular un entorno nocturno se realizarán dentro de un cuarto sin ventanas ni acceso de luz por ningún lado
RNF-12	El modelo de reconocimiento seleccionado debe ser capaz de reconocer caras en un tiempo de ejecución menor que 2 segundos
RNF-13	El modelo de reconocimiento seleccionado debe ser capaz de reconocer caras, aunque estas lleven gorros, gafas de vista o mascarillas
RNF-14	El modelo de reconocimiento seleccionado debe ser capaz de reconocer caras que presente distintas expresiones como una cara neutra, sonriendo o con los ojos cerrados
RNF-15	El modelo de reconocimiento seleccionado debe ser capaz de reconocer caras con variabilidad de pose como caras de frente o laterales a ambos lados
RNF-16	El modelo de reconocimiento seleccionado debe ser capaz de reconocer caras con distintas características físicas como densidad de cabello, barba o color de piel

*Tabla 3.2: Requisitos no funcionales del sistema*

## 3.2 Análisis del marco legal y ético

Este apartado tiene como principal propósito evaluar todos los aspectos éticos y legales relacionados con la implementación de un sistema de detección y reconocimiento facial en un videoportero. Este análisis es fundamental para garantizar a los usuarios de nuestro sistema que se cumpla rigurosamente con la normativa vigente y que, bajo ningún concepto, se vulneren sus derechos.

### 3.2.1 Protección de datos

Si bien es cierto que para cumplir con el propósito de la detección facial no es necesario el almacenamiento de datos faciales, debemos de plantearnos que, para el correcto funcionamiento del reconocimiento facial en el sistema, se deberá almacenar la característica asociada a cada cara de los inquilinos de una vivienda. La gestión de estos datos nos sugiere analizar y tener en cuenta qué normativa y/o medidas se deben aplicar para garantizar la seguridad de los datos personales de los usuarios. En este caso particular, se deberá aplicar la normativa española que regula el tratamiento de los datos personales, la Ley Orgánica de Protección de Datos (**LOPD**). Esta ley establece una serie de medidas de obligatorio cumplimiento para las organizaciones que manejen todo tipo de datos personales de sus usuarios. Algunos aspectos para tener en cuenta son:

- **Medidas de seguridad:** Se deben aplicar medidas de seguridad adecuadas para la protección de los datos. Por ejemplo, aplicar cifrado de datos previo al almacenamiento, controlar el acceso a los datos para que su modificación o lectura solo lo pueda realizar el personal autorizado, etc.
- **Derechos de los usuarios:** Los usuarios deben tener derecho en todo momento para poder acceder, rectificar, cancelar u oponerse al tratamiento de sus datos personales.



- **Sanciones:** Las organizaciones que no cumplan con la LOPD pueden ser sancionadas económicamente con multas de hasta los 600.000 euros, dependiendo del tipo de infracción.

### 3.2.2 Propiedad intelectual

Antes de desarrollar cualquier sistema, siempre es conveniente analizar el software de terceros que vamos a utilizar para asegurarnos de no infringir patentes o derechos de autor existentes. En el caso de este proyecto y, haciendo referencia al requisito no funcional RNF-2, todas las bibliotecas o marcos de trabajo de terceros que se utilicen deben de ser de código abierto. Además, es conveniente revisar los términos de licencia de dicho software, ya que en muchos casos el código puede ser de uso libre, pero la licencia puede no permitir el uso de dicho software con fines comerciales. En el caso de este proyecto, se procurará encontrar software de código libre que tengan como único requisito colocar un *disclaimer* al principio del código. Generalmente, estos se utilizan para especificar aspectos como la licencia del código, los autores, las modificaciones realizadas con respecto al código original, etc.

### 3.2.3 Marco ético

Como comentamos en el capítulo anterior, si queremos hacer un correcto análisis del problema que se plantea en este proyecto, debemos analizar el marco ético que envuelve al mismo. Al incluir funcionalidades relacionadas con la visión por computador en un videoportero, se plantean diversas situaciones que deben ser analizadas para conocer la mejor manera de lidiar con ellas. Por ejemplo, al recopilar datos sensibles de los usuarios como datos biométricos, la empresa deberá obtener el consentimiento explícito por parte de los usuarios para la manipulación de dichos datos. Además, es conveniente que el sistema sea transparente e informe a los usuarios si están siendo analizados, o, si se está realizando alguna clase de extracción de información. Por otro lado, al hablar en específico sobre la detección y el reconocimiento facial, se debe garantizar la equidad en cuanto al funcionamiento para diferentes grupos demográficos (raza, género, edad, ...). Hoy en día, se pueden encontrar diversos artículos (Birhane, 2022) que evidencian la existencia de sistemas de detección y reconocimiento facial discriminatorios para las personas de piel oscura. De hecho, el artículo *Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification* (Buolamwini & Gebru, 2018) concienció sobre la importancia de abordar los sesgos en sistemas de IA y la necesidad de emplear *datasets* con datos más inclusivos y representativos a la hora de entrenar los modelos.

## 3.3 Análisis de riesgos

---

Llevar a cabo una correcta gestión de los riesgos que amenazan un proyecto es fundamental para garantizar el éxito de este. Con este motivo en mente, se propone abordar un análisis de los riesgos que amenazan al proyecto, pudiendo así evaluar y confeccionar medidas para poder cumplir con los objetivos y procedimientos establecidos. El procedimiento para llevar a cabo el análisis se dividirá en tres partes: la primera es la identificación de los eventos de riesgo, en donde se elaborará un listado con los posibles eventos que pueden llegar a generar un impacto en los

objetivos del proyecto. Una vez hecho el listado, se llevará a cabo la segunda parte, la evaluación del riesgo. En esta etapa se estima la probabilidad de ocurrencia del evento y el impacto generado por el mismo, haciendo uso de una matriz de riesgos. Finalmente, el último apartado se trata de la respuesta, en donde gracias a la información recaudada en las otras etapas, se eligen los riesgos más críticos y se proponen medidas para neutralizarlos siguiendo los principios de la metodología PRINCE2.

### 3.3.1 Identificación de los eventos de riesgo

El primer paso para identificar los eventos de riesgo de nuestro proyecto es seleccionar cuales son las tareas y los recursos críticos de este. En este caso, vamos a seleccionar como las tareas críticas a cada uno de los objetivos principales listados en la introducción de este trabajo. Por otro lado, en cuanto a los recursos críticos del proyecto podemos seleccionar a: **R1**-Equipo I+D de la empresa, **R2**-Tutor de la empresa, **R3**-Material de trabajo. Véase que cada recurso crítico viene precedido por un identificador, esto se debe a que haremos referencia a cada uno de estos en próximos apartados. Con los tareas y recursos críticos en mente, podemos llevar a cabo la descripción de los eventos de riesgo:

ID	Causa del riesgo	Evento del riesgo	Efecto del riesgo
1	Falta de experiencia en la implementación y compilación cruzada de bibliotecas de visión por computador	No se logra integrar correctamente una biblioteca o marco de trabajo	Retraso en el desarrollo del proyecto
2	Incompatibilidad de las bibliotecas con la arquitectura ARM	La biblioteca seleccionada no se compila correctamente en el sistema objetivo	Pérdida de tiempo y recursos
3	Modelos pre entrenados inadecuados	Los modelos de detección facial no funcionan en tiempo real	Rendimiento deficiente en el videoportero
4	Problemas de iluminación en el videoportero	Variaciones en la precisión debido a condiciones de luz	Resultados inconsistentes
5	Falta de documentación o soporte para los modelos seleccionados	Dificultad para resolver problemas o ajustar los modelos	Retrasos en la implementación
6	Similitud de rostro con la persona registrada (Falsos positivos)	El modelo identifica erróneamente a una persona no registrada	Acceso no autorizado a la vivienda
7	Usuarios de diversas etnias tratan de acceder a su vivienda	El modelo tiene dificultades para reconocer rostros de diferentes etnias	Sesgo y discriminación
8	Cambios en la apariencia del usuario de la vivienda	Cambios en el peinado, maquillaje o uso de accesorios	Dificultad para reconocer al mismo usuario
9	Falta de colaboración del equipo de I+D	Retrasos o errores en la integración debido a falta de cooperación	Impacto en el desarrollo del proyecto

10	Almacenamiento inseguro de datos biométricos	Filtración de datos biométricos	Demandas por partes de los usuarios del sistema
11	Falta de recursos hardware para pruebas	No se pueden realizar pruebas exhaustivas	Incertidumbre sobre el rendimiento real
12	Dependencia excesiva del tutor o equipo de I+D	Retrasos si el tutor o el equipo no están disponibles	Pérdida de tiempo y oportunidades

*Tabla 3.3: Descripción de los eventos de riesgo*

### 3.3.2 Evaluación del riesgo

Para realizar un correcto análisis de los riesgos del proyecto, primero vamos a **caracterizar las amenazas** definiendo: la tarea o recurso crítico a la que afecta, su posible efecto en el proyecto y también cómo de probable es que ocurra dicha amenaza. Esta caracterización se hace con el motivo de elaborar, a posteriori, una **matriz de riesgos**. Dicha matriz es una herramienta empleada en la gestión de proyectos para evaluar el nivel de riesgo relacionado con una actividad a través de una codificación de color. Así mismo, esta permite implementar estrategias efectivas para contrarrestar los riesgos y garantizar el éxito del proyecto.

ID	Evento del riesgo	Tarea o recurso afectado	Objetivo (Tiempo, coste y Alcance)	Frecuencia	Efecto
1	No se logra integrar correctamente una biblioteca o marco de trabajo	Objetivo nº2	Tiempo y alcance	Probable	Crítico
2	La biblioteca seleccionada no se compila correctamente en el sistema objetivo	Objetivo nº2	Tiempo	Probable	Marginal
3	Los modelos de detección facial no funcionan en tiempo real	Objetivo nº3	Tiempo y alcance	Ocasional	Crítico
4	Variaciones en la precisión debido a condiciones de luz	Objetivo nº4 y Objetivo nº7	Tiempo, alcance y coste	Frecuente	Crítico
5	Dificultad para resolver problemas o ajustar los modelos	Objetivo nº3	Tiempo	Rara vez	Despreciable
6	El modelo identifica erróneamente a una persona no registrada	Objetivo nº7	Tiempo y coste	Improbable	Marginal
7	El modelo tiene dificultades para reconocer rostros de diferentes etnias	Objetivo nº7	Coste	Improbable	Catastrófico

8	Cambios en el peinado, maquillaje o uso de accesorios	Objetivo nº7	Tiempo y alcance	Rara vez	Marginal
9	Retrasos o errores en la integración debido a falta de cooperación	Objetivo nº2, Objetivo nº6 y R1	Tiempo	Ocasional	Marginal
10	Filtración de datos biométricos	Objetivo nº6 y Objetivo nº7	Coste	Improbable	Catastrófico
11	No se pueden realizar pruebas exhaustivas	Objetivo nº4, Objetivo nº7 y R3	Tiempo y coste	Rara vez	Despreciable
12	Retrasos si el tutor o el equipo no están disponibles	R1 y R2	Tiempo	Rara vez	Marginal

*Tabla 3.4: Caracterización de las amenazas*

Evaluación del impacto de los riesgos					
Efecto	Frecuencia				
	Frecuente	Probable	Ocasional	Rara vez	Improbable
Catastrófico					7, 10
Crítico	4	1	3		
Marginal		2	9	8, 12	6
Despreciable				5, 11	

*Figura 3.1: Matriz de riesgos*

### 3.3.3 Confección de respuestas

Una vez analizado el posible impacto y frecuencia de los diversos riesgos, el último paso se basa en confeccionar las contramedidas para paliar los efectos negativos de las amenazas. En la gestión de riesgos existen dos enfoques de medidas de respuesta: enfoque proactivo y reactivo. El enfoque proactivo busca predecir eventos adversos y tomar medidas para evitar que estos ocurran. Por otro lado, el enfoque reactivo busca actuar cuando surge un problema, comúnmente se le asocia con la extinción de incendios. Por otro lado, en la tabla 3.5, se han utilizado los 6 tipos de respuestas contra las amenazas que sugiere PRINCE2: evitar, reducir, estrategia alternativa, transferir, compartir y aceptar. PRINCE2 es una metodología de gestión de proyectos ampliamente utilizada, la cual, propone un conjunto de buenas prácticas para convertir a los proyectos con una gran carga de incertidumbre, en proyectos controlados.

<b>ID</b>	<b>Evento del riesgo</b>	<b>Medida de respuesta</b>	<b>Tipo (PRINCE2)</b>	<b>Descripción</b>
1	No se logra integrar correctamente una biblioteca o marco de trabajo	Reactiva	Aceptar	Se descarta dicha biblioteca o marco y se pasa a intentar integrar la siguiente
2	La biblioteca seleccionada no se compila correctamente en el sistema objetivo	Proactiva	Evitar	Revisar previamente la documentación de la biblioteca para analizar si es compatible
3	Los modelos de detección facial no funcionan en tiempo real	Proactiva	Evitar	Revisar previamente la documentación del modelo
4	Variaciones en la precisión debido a condiciones de luz	Proactiva	Reducir	Pruebas exhaustivas que permitan analizar y reducir el problema
5	Dificultad para resolver problemas o ajustar los modelos	Reactiva	Reducir	Estudiar la documentación del modelo en busca de soluciones
6	El modelo identifica erróneamente a una persona no registrada	Proactiva	Reducir	Pruebas exhaustivas que permitan analizar y reducir el problema
7	El modelo tiene dificultades para reconocer rostros de diferentes etnias	Proactiva	Evitar	Variedad de pruebas con personas de distintas etnias
8	Cambios en el peinado, maquillaje o uso de accesorios	Proactiva	Evitar	Variedad de pruebas con personas con cambios de aspecto
9	Retrasos o errores en la integración debido a falta de cooperación	Proactiva	Evitar	Notificar previamente las dudas al equipo de desarrollo
10	Filtración de datos biométricos	Proactiva	Evitar	Codificar los datos de manera segura en la BBDD
11	No se pueden realizar pruebas exhaustivas	Reactiva	Contingencia	Diseñar entornos de prueba alternativos, como el uso de otro videoportero
12	Retrasos si el tutor o el equipo no están disponibles	Proactiva	Evitar	Planificar las reuniones de dudas con antelación

*Tabla 3.5: Confección de respuestas ante las posibles amenazas*



# Planificación de la solución

---

El contenido de este capítulo se divide en tres partes: en primer lugar, se define la metodología que ha sido empleada para elaborar este proyecto. Tras esto, se realiza una comparación entre la estimación de horas, esfuerzo y material del plan de trabajo inicial, con los recursos reales invertidos en las fases de desarrollo. Y, por último, se define el presupuesto del proyecto, determinando la cantidad de recursos humanos y materiales utilizados para el correcto progreso del trabajo.

## 4.1 Metodología

---

Durante el transcurso de este proyecto, aunque no se siguió una metodología en específico, se adoptó un enfoque colaborativo y flexible similar al empleado en las metodologías ágiles. Esta semejanza se puede ver reflejada con la presencia de conceptos clave como pueden ser los roles de las personas implicadas. En primer lugar, podemos mencionar al tutor de mi empresa de prácticas como el *Product Owner*, ya que él es el encargado de representar los intereses del usuario final, definiendo y priorizando los requisitos del producto. Por otro lado, estaría el equipo de desarrollo, el cual, corresponde con el equipo de I+D de la empresa. Aunque en este caso el equipo no son los encargados de implementar el software como tal, sino que tienen como función principal apoyar y orientar al alumno en todo el proceso del desarrollo. Finalmente, cabe destacar al tutor de la universidad. En este caso, podemos ver a dicho tutor como un *stakeholder*, el cual, es responsable de que se cumplan tanto sus expectativas como los objetivos del proyecto.

Otro concepto clave que podemos apreciar en la metodología empleada para el proyecto, es el tipo de reuniones ágiles empleadas. Principalmente, la metodología de trabajo se basó en una dinámica de reuniones semanales al final de la semana con el tutor de la empresa. En dichas reuniones se comentaban los avances que se habían logrado en el proyecto, así como los contratiempos, posibles problemas o dudas acerca de las tareas en desarrollo. Como resultado de estas sesiones, el tutor conseguía tener una retroalimentación continua del proyecto permitiendo, si la situación lo requería, realizar ajustes en los objetivos respondiendo así de manera ágil a los cambios. Para complementar estas sesiones, también se debe destacar las reuniones diarias que se llevaban a cabo con los trabajadores del equipo de I+D de la empresa. Dichas reuniones, más que para reenfocar los objetivos del proyecto, tenían como principal objetivo resolver las posibles dudas que iban surgiendo, sobre todo, dudas relacionadas sobre como implementar las funcionalidades investigadas en el sistema actual del videoportero. Por último, cabe destacar la comunicación con el tutor de la universidad. Esta comunicación, aunque fuese limitada, sirvió para arrojar un punto de vista distinto al desarrollo del proyecto, consiguiendo así asesoramiento sobre cómo se deberían realizar determinadas tareas.

Por último, otro concepto fundamental dentro de las metodologías ágiles radica en los procesos iterativos y las entregas incrementales. Debido a la naturaleza del proyecto, no se estipuló desde un inicio una estructura clara de las iteraciones que eran necesarias para obtener el mínimo producto viable (MVP). Aunque cabe destacar que, dentro de las distintas etapas del proyecto, una vez se concluía el apartado de análisis e investigación y se procedía a implementar y realizar las pruebas, el tutor de la empresa establecía la duración del *sprint*, así como las diferentes entregas que se deberían llevar a cabo durante el transcurso de este.

## 4.2 Estimación del plan de trabajo

El objetivo principal de esta sección es mostrar la estimación inicial del plan de trabajo junto con el desarrollo efectivo del mismo. Al inicio del proyecto, se llevó a cabo un plan de trabajo inicial en donde se estimaron las posibles fases de desarrollo junto con sus costes humanos y materiales. El plan inicial en cuestión es el siguiente:

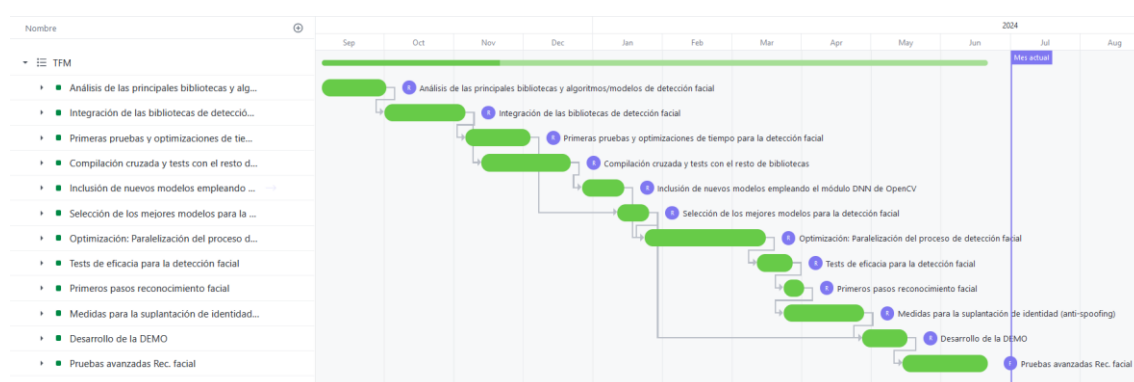
Fase	Material necesario	Horas/personas
Investigación sobre los modelos de detección facial	Ordenador personal	25 horas / 1 persona
Selección de los modelos de detección facial más adecuados para nuestro sistema	Ordenador personal	50 horas / 1 persona
Implementación de los modelos de detección facial	Ordenador personal y videoportero	200 horas / 1 persona
Pruebas de rendimiento, precisión y eficacia	Ordenador personal y videoportero	50 horas / 1 persona
Investigación sobre los modelos de reconocimiento facial	Ordenador personal	25 horas / 1 persona
Selección de los modelos de reconocimiento facial más adecuados para nuestro sistema	Ordenador personal	50 horas / 1 persona
Implementación de los modelos de reconocimiento facial	Ordenador personal y videoportero	265 horas / 1 persona
Pruebas de rendimiento, precisión y eficacia	Ordenador personal y videoportero	75 horas / 1 persona
Confección de una demo que incorpora las funcionalidades implementadas	Ordenador personal y videoportero	150 horas / 1 persona
Análisis de trabajo futuro	Ordenador personal	10 horas / 1 persona

Tabla 4.1: Estimación inicial del plan de trabajo



Como es evidente, debido a la naturaleza incierta del trabajo junto con la falta de experiencia para realizar estimaciones, el plan inicial de trabajo presenta diversas limitaciones. Principalmente, se destaca la poca especificación de las fases de desarrollo del proyecto, listando únicamente fases generales que engloban diversas subtarefas.

Del mismo modo, una vez iniciado el proyecto y, durante todo el transcurso de este, se ha llevado a cabo la confección de un diagrama de Gantt con el que poder distinguir las fases de desarrollo que finalmente se han empleado para el proceso del proyecto. En dicho diagrama, no solo podemos apreciar las distintas fases de desarrollo, sino que podemos analizar sus dependencias junto con la duración de estas. Para confeccionar este diagrama se ha utilizado la web **ClickUp**, la cual oferta de manera gratuita diversas herramientas para el desarrollo de diagramas. El diagrama de Gantt final se puede apreciar en la siguiente imagen o visitando este [enlace](#) para una mejor visualización:



*Figura 4.1: Diagrama de Gantt*

## 4.3 Presupuesto

El objetivo de esta sección es analizar cuáles han sido los costes totales para llevar a cabo este trabajo. Dichos costes se dividen en dos partes, en primer lugar, tenemos los costes referentes a los recursos humanos y, por otro lado, tenemos los costes relacionados con el material empleado en el trabajo. Dicho material se corresponde no solo con herramientas físicas como el ordenador, sino que también incluye todo el software que se haya utilizado.

### 4.3.1 Recursos humanos

Además de los recursos materiales empleados para el correcto desarrollo del trabajo, debemos de tener en cuenta el coste de los recursos humanos si queremos llevar a cabo una estimación lo más precisa posible del presupuesto del proyecto. Por ello, para conocer los costes relativos a los puestos de trabajo de las personas implicadas en el proyecto, he consultado páginas web como **Indeed**, **Payscale** o **Glassdoor**. Estas webs ofrecen datos salariales considerados fiables, ya que la información facilitada procede de los propios empleados, ofertas de empleo o directamente de

las empresas. En la tabla 4.2 podemos apreciar los costes finales calculados tras el análisis de los datos de las tres webs:

Trabajo asignado	Precio por hora [€]	Horas de trabajo	Coste total [€]
Ingeniero de software junior	22	875	19.250
2x Ingeniero de firmware senior	40	200	8.000
CTO hardware	75	60	4.500

*Tabla 4.2: Presupuesto de los recursos humanos*

### 4.3.2 Recursos materiales

Como hemos comentado en apartados anteriores, por recursos materiales se comprenden todos aquellos recursos tangibles (fuentes de alimentación, ordenador personal, etc.) e intangibles (software utilizado) que se han empleado para lograr el éxito del proyecto. Del mismo modo, cabe destacar que como parte del requisito no funcional RFN-2, todo el material software que se ha utilizado para este trabajo ha sido de código abierto. Por lo tanto, tal y como podemos observar en la tabla 4.4, el cumplimiento con éxito del requisito funcional ha repercutido de manera muy positiva en el presupuesto final del proyecto, eliminando los costes relativos a los recursos software utilizados.

Material	Coste total [€]
Ordenador portátil	950
Ordenador de sobremesa	900
Periféricos	350
Videoportero	1000
Fuente de alimentación	25
Soporte del videoportero	20
Panel para pruebas	35
Cables duox	5
Cable UART	3

*Tabla 4.3: Presupuesto de los recursos hardware*

Material	Coste total [€]
Bibliotecas (OpenCV, Dlib, ...)	0 €
Modelos pre entrenados	0 €
Visual Studio Code	0 €
Github	0 €
Docker	0 €
Microsoft Teams	0 €
Inkscape	0 €

*Tabla 4.4: Presupuesto de los recursos software*

### 4.3.3 Presupuesto final

Finalmente, tras haber calculado por separado los costes de cada uno de los recursos implicados en el desarrollo del proyecto, el último paso consiste en llevar a cabo la suma para obtener la estimación del presupuesto final del proyecto:

<b>Presupuestos</b>	<b>Coste total [€]</b>
Recursos humanos	31.750
Recursos hardware	3.288
Recursos software	0
<b><i>Presupuesto final</i></b>	<b><i>35.038 €</i></b>

*Tabla 4.5: Presupuesto de final*

## Desarrollo de la solución

---

En este capítulo del proyecto se describirá todo el trabajo realizado, abarcando desde las tareas de investigación y análisis, hasta la implementación de la solución y las pruebas.

### 5.1 Detección facial

---

El primer bloque de este proyecto trata de exponer todas aquellas tareas relacionadas con la detección facial. Tal y como se ha visto en capítulos anteriores, la detección facial es un proceso fundamental para este trabajo. Esto se debe principalmente a que, dependiendo de los resultados que obtengamos cuando realicemos las pruebas de rendimiento, se podrá analizar y sacar las conclusiones oportunas sobre la viabilidad de añadir reconocimiento facial al sistema.

#### 5.1.1 Fase de análisis

En el capítulo del estado de arte, se ha expuesto un extenso análisis sobre los principales conceptos relacionados con la detección facial. Dicho análisis forma parte de una de las primeras tareas que se llevaron a cabo en este proyecto, aunque, cabe destacar que, la parte principal del trabajo de investigación inicial se divide en dos partes: la investigación sobre las principales bibliotecas y marcos de trabajo para la visión por computador y, por otro lado, el análisis de los algoritmos de detección facial más utilizados en la actualidad.

Las bibliotecas de visión por computador se definen como colecciones de herramientas software (funciones, algoritmos, etc.) diseñados para realizar tareas concretas sobre el procesamiento y análisis de datos visuales (imágenes o vídeos). Por otro lado, los marcos de trabajo para la visión por computador proveen al usuario de un ecosistema software más amplio para desarrollar soluciones completas de visión artificial. Por ejemplo, los marcos de trabajo para la visión por computador suelen contener un conjunto más completo de herramientas para implementar todo tipo de funcionalidades como el diseño y entrenamiento de modelos, el procesamiento de datos, el proceso de inferencia, entre muchas otras. Podemos ver a las bibliotecas como herramientas individuales y, a los marcos, como cajas de herramientas que organizan y agilizan el proceso de desarrollo.

Nombre	Tipo	Lenguaje	Tiempo real	ARM	Comunidad	Propósito
OpenCV	Biblioteca	C++, Python	Sí	Sí	Buena y activa	General dentro del campo de la visión por computador

Tensorflow	Marco de trabajo	Python	Sí	Sí	Buena y activa	Tareas de aprendizaje profundo
PyTorch	Marco de trabajo	Python	Sí	Sí	Grande y activa	Tareas de aprendizaje profundo
Dlib	Biblioteca	C++, Python	Sí	Sí	Moderada pero activa	Variedad de tareas de visión por computador
Caffe	Marco de trabajo	C++, Python	Sí	No	Moderada pero activa	Tareas de aprendizaje profundo
Open VINO	Marco de trabajo	C++, Python	Sí	Sí	Moderada	Inferencia optimizada para hardware Intel
Keras	Marco de trabajo	Python	Sí	Sí	Grande y activa	Tareas de aprendizaje profundo
YOLO	Biblioteca	C, C++, Python	Sí	Sí	Grande y activa	Detección de objetos en tiempo real
DeepFace	Biblioteca	Python	Sí	No	Moderada	Reconocimiento facial
Libface detection	Biblioteca	C++	Sí	Sí	Pequeña	Detección facial

*Tabla 5.1: Comparación de bibliotecas y marcos de trabajo*

Tras un proceso de investigación, se propuso la tarea de confeccionar una tabla comparativa en donde se listasen las principales bibliotecas y marcos de trabajo para la visión por computador junto con sus características clave. El resultado se puede observar en la tabla 5.1. Con dicha tabla, se realizó una primera fase de preselección de las bibliotecas o marcos que, a priori, pudieran funcionar en nuestro sistema objetivo. Esta decisión estuvo sujeta principalmente a los requisitos funcionales y no funcionales RNF-1 y RNF-2, en los que se delimita que la biblioteca seleccionada debe: utilizar el lenguaje de programación C++, ser compatible con la compilación cruzada para un sistema ARM embebido y, por último, ejecutar en tiempo real las diversas funciones de visión por computador. Con esto en mente, las bibliotecas elegidas provisionalmente han sido: **OpenCV, Dlib, y Libfacedetection**.

Como parte final de la primera fase de investigación, existía la tarea de investigar sobre los algoritmos de detección facial más utilizados en la actualidad. Sin embargo, a diferencia de la investigación de las bibliotecas o marcos, en esta ocasión no se pretendía llevar a cabo una preselección de los posibles mejores algoritmos para nuestro sistema, sino que el objetivo principal de esta tarea residía en obtener una base sólida de conocimientos sobre los algoritmos más utilizados en la detección facial. En capítulos anteriores, se empleó la figura 2.5 para exponer la reciente tendencia de los algoritmos de detección facial para adoptar enfoques similares a los utilizados en los nuevos modelos de detección de objetos. Además, con ayuda de dicha figura, podemos observar la gran cantidad de algoritmos de detección facial que han sido publicados entre los años 2012 a 2021. Y es que, a raíz de la publicación de Viola Jones (Viola & Jones, 2001) y, sobre todo, tras la aparición de las técnicas basadas en el aprendizaje profundo, se puede

apreciar que la confección de nuevos algoritmos para la detección facial ha experimentado un crecimiento significativo desde entonces. Como resultado, las evaluaciones de los rendimientos de estos nuevos modelos han experimentado el mismo crecimiento, pasando de una precisión media (*AP\_HARD*) en el conjunto de validación de WIDER Face de **0.400 en 2015** (*Multiscale Cascade CNN*), a una precisión de **0.934 en 2020** haciendo uso del modelo *TinaFace*.

### 5.1.2 Integración de las bibliotecas seleccionadas

Una vez adquiridos los conocimientos principales acerca del campo de la visión por computador, la siguiente tarea del trabajo consistía en realizar la compilación cruzada de la lista de bibliotecas preseleccionadas. Para entender realmente en qué consiste la compilación cruzada, primero es necesario tener claro el concepto de compilación. Los lenguajes de programación compilados (C, C++, C#, COBOL, ...) convierten el código fuente escrito en lenguaje de alto nivel en código máquina u objeto a través del compilador. El proceso de compilación consta de diversas etapas: Análisis léxico y sintáctico, generación de código intermedio, análisis semántico, optimización y, finalmente, la generación del código objeto convirtiendo el código fuente en instrucciones que el ordenador pueda entender y ejecutar. Así pues, con los conceptos de compilación y compilador claros, podemos definir la compilación cruzada como un tipo de compilación cuya finalidad es crear código ejecutable para plataformas distintas de aquella en la que se está ejecutando el compilador.

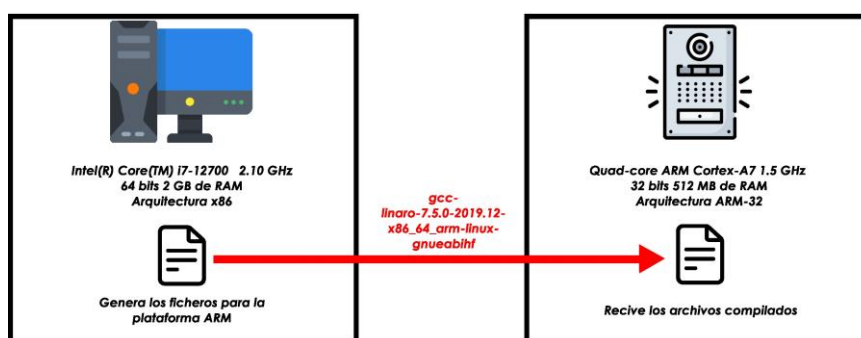


Figura 5.1: Ejemplo de la compilación cruzada

La compilación de una biblioteca de visión por computador es un trabajo duro en sí mismo y aún más teniendo un sistema con recursos hardware tan limitados como vemos en la figura 5.1. Con estas especificaciones, afrontamos que la compilación completa de la biblioteca podría tardar desde varios minutos a horas, dependiendo de las dimensiones de esta. Este es el motivo principal del uso de la compilación cruzada, utilizar ordenadores de mayores prestaciones para realizar el trabajo de compilación en mucho menos tiempo. En nuestro caso, la compilación cruzada se ha realizado con el ordenador de sobremesa de la empresa, el cual, tiene las especificaciones que observamos en la figura 5.1. Finalmente, las bibliotecas que se han conseguido integrar con éxito en nuestro sistema ARM tras el proceso de compilación cruzada han sido: OpenCV, Dlib y libfacedetection, destacando que, para cada una de ellas, se ha seguido las instrucciones encontradas en sus respectivas documentaciones.

### 5.1.3 Subsistema de visión por computador

Una vez finalizada la compilación cruzada de las bibliotecas, el próximo objetivo constaba en diseñar e implementar un subsistema dentro de la arquitectura actual del videoportero. Este subsistema permitiría ejecutar las diversas funcionalidades de visión por computador disponibles tras haber compilado con éxito las bibliotecas. Teniendo claro el tipo de dispositivo que se iba a emplear, se suponía que el flujo de nuestro subsistema debería ser algo muy similar al expuesto en la figura 5.2.

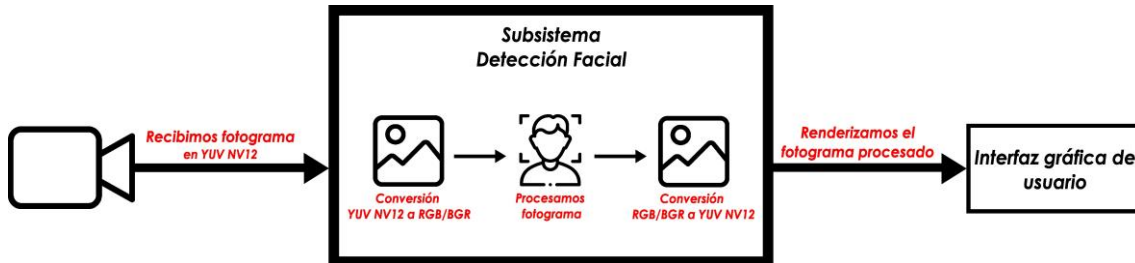


*Figura 5.2: Posible flujo inicial del subsistema*

En dicho flujo, la primera acción la realiza la cámara de nuestro videoportero, la cual, nos debe proporcionar los diversos fotogramas que vaya capturando. Estos fotogramas presentan una particularidad, ya que emplean una codificación de color **YUV NV12** en vez de una codificación de color algo más común como puede ser RGB. La codificación YUV NV12 se emplea para optimizar el espacio de almacenamiento, es decir, en NV12 los píxeles de luminancia (componente Y) se almacenan de forma contigua, y los píxeles de crominancia (UV) se almacenan de forma entrelazada (esta forma varía según el tipo de YUV que se emplee). Esta forma de codificación supone que, para representar, por ejemplo, un fotograma de 16 píxeles, en NV12 se requieren 24 bytes (16 Y + 8 UV) frente a los 48 bytes que necesitaríamos utilizando RGB (16 píxeles \* 3 componentes). Una vez adquirido el fotograma, el siguiente paso será procesarlo ejecutando el proceso de visión por computador seleccionado. Finalizado dicho proceso, este devolverá un resultado con el que podemos llevar a cabo la fase de post procesamiento. Si, por ejemplo, seleccionamos la funcionalidad de detección facial, la fase de procesamiento nos devolverá los datos necesarios para poder dibujar en el fotograma un cuadro delimitador por cada cara detectada.

El flujo que se presentó en la figura 5.2 podría haber sido el flujo definitivo para nuestro subsistema. Sin embargo, una vez dio comienzo la implementación de dicho diseño, surgieron complicaciones. El problema principal surgía a raíz de la particular codificación de color que empleaba la cámara de nuestro videoportero, puesto que, las diversas funcionalidades de visión por computador normalmente utilizaban codificaciones de color RGB o BGR para su correcto funcionamiento. Como resultado, nos veríamos forzados a realizar una conversión de color previa a la fase de procesamiento, aumentando así el tiempo de ejecución de nuestro subsistema. Sin embargo, este no sería el único contratiempo, ya que, por la configuración del sistema del videoportero, era necesario que, para pintar el fotograma procesado empleando la GUI, este debía estar en el formato de codificación de color YUV NV12. Este requisito surgía debido a que la GUI, para pintar por la pantalla del videoportero, se encargaba de realizar la conversión de YUV NV12 a RGB a través de un *shader* que se ejecutaba en la GPU. Por lo tanto, se procedió a

seleccionar provisionalmente el flujo expuesto en la figura 5.3, donde realizamos las conversiones comentadas en la etapa previa y posterior al procesamiento del fotograma. Tras la implementación de esta versión provisional del subsistema, se estaba en posición de realizar las primeras pruebas de rendimiento con las diversas bibliotecas integradas. Pero antes, se debía realizar una preselección de los modelos de detección facial que se emplearían para las pruebas.



*Figura 5.3: Flujo modificado del subsistema - conversión YUV a RGB*

#### 5.1.4 Preselección de modelos

Cada una de las bibliotecas de visión por computador que fueron seleccionadas contaban con características únicas que las distinguían del resto. Antes de exponer como se llevó a cabo el proceso de preselección de los modelos de detección facial, se va a contextualizar y explicar, brevemente, los principales atributos de las tres bibliotecas seleccionadas:

- **OpenCV:** Es una de las bibliotecas de código abierto más utilizadas en el campo de la visión por computador. Ofrece una infinidad de herramientas y funcionalidades para el procesamiento de imágenes o vídeos como reconocimiento facial, detección de objetos, calibración de cámaras, etc. Una de sus principales ventajas es su compatibilidad con múltiples plataformas, además de su optimización para ejecutar funcionalidades de visión por computador a tiempo real con distintas especificaciones de hardware. Por otro lado, ofrece una interfaz en diversos lenguajes de programación como C++, Python o Java. Finalmente, debemos destacar que esta biblioteca presenta una gran comunidad activa que, junto con una abundante documentación, reduce significativamente la curva de aprendizaje.
- **Dlib:** aunque menos conocida que OpenCV, Dlib es una biblioteca de procesamiento de imágenes y aprendizaje automático, utilizada tanto en la industria como en el mundo académico en una amplia gama de dominios, como pueden ser los dispositivos embebidos, la robótica, los teléfonos móviles, etc. Aunque no presenta una gran comunidad, su documentación es bastante completa, ofreciendo una gran gama de ejemplos para poder utilizar sin problema todas y cada una de las utilidades de la biblioteca. Además, presenta un código portable y con licencia libre, que permite utilizar la biblioteca en cualquier aplicación de manera rápida e intuitiva.
- **Libfacedetection:** es una biblioteca de código abierto para la detección facial en imágenes basadas en redes neuronales convolucionales (CNN). La creación de esta biblioteca fue realizada principalmente por el profesor asociado del departamento de Informática e Ingeniería de la Universidad Meridional de Ciencia y Tecnología de Shenzhen (China), Shiqi Yu. Dicho profesor convirtió el modelo de detección facial en



variables estáticas (matrices) almacenadas en archivos fuente del lenguaje de programación C. Esta característica junto con la posibilidad de compilar el código para diversas plataformas (solo es necesario un compilador de C++), además de permitir el uso de instrucciones SIMD para acelerar el proceso de detección, ha conseguido llamar la atención del sector por sus increíbles resultados en el campo de la detección facial.

Una vez expuestas las principales características es necesario incidir en aquella que tuvo una mayor repercusión para la selección de los modelos de detección facial. Dicha característica se trata del propósito general de las bibliotecas, ya que, por ejemplo, libfacedetection tiene como objetivo principal ofrecer la funcionalidad de detección facial a través de su propio modelo pre entrenado. Como consecuencia, esta biblioteca no permite realizar pruebas con otros modelos de detección que no sean el suyo. Por otro lado, en el caso de dlib, es importante señalar que, aunque esta disponga de la posibilidad de utilizar su propio modelo o uno ajeno, la integración e implementación del código necesario para emplear un modelo ajeno no es trivial, además de que carece de ejemplos para entender correctamente su funcionamiento. Afortunadamente, la biblioteca OpenCV presenta no tan solo una recomendación de uso de un modelo particular pre entrando por ellos, sino que también dispone de un módulo de aprendizaje profundo (**módulo DNN**), el cual permite realizar el proceso de inferencia sobre diversos modelos pre entrenados utilizando diversos marcos de aprendizaje profundo como pueden ser: Caffe, Tensorflow, ONNX, Torch o Darknet.

Modelo	Biblioteca	Marco de trabajo	Extensión	Resolución de entrada	Seleccionado
Haar	OpenCV	-	.xml	Dinámico	<b>Sí</b>
HOG	DLIB	-	Integrado en la biblioteca	Dinámico	<b>Sí</b>
YuNet	OpenCV	-	.onnx	Dinámico	<b>Sí</b>
S3FD	libfacedetection	-	Integrado en la biblioteca	Dinámico	<b>Sí</b>
YoloV5-face	OpenCV DNN	Darknet	.pt   .onnx	Fijo (640x640)	No
YoloV7-face	OpenCV DNN	Darknet	.pt   .onnx	Fijo (640x640)	No
YoloV8-face	OpenCV DNN	Darknet	.pt   .onnx	Fijo (640x640)	<b>Sí</b>
BlazeFace	OpenCV DNN	MediaPipe	.tflite	Fijo (128x128)	No
MTCNN	OpenCV DNN	Caffe	.prototxt y .caffemodel	Dinámico	No
RetinaFace	OpenCV DNN	Tensorflow	.params y .json	Dinámico	No
DSFD	OpenCV DNN	Pytorch	.pth	Dinámico	No
SSD	OpenCV	-	.prototxt y .caffemodel	Fijo (300x300)	<b>Sí</b>

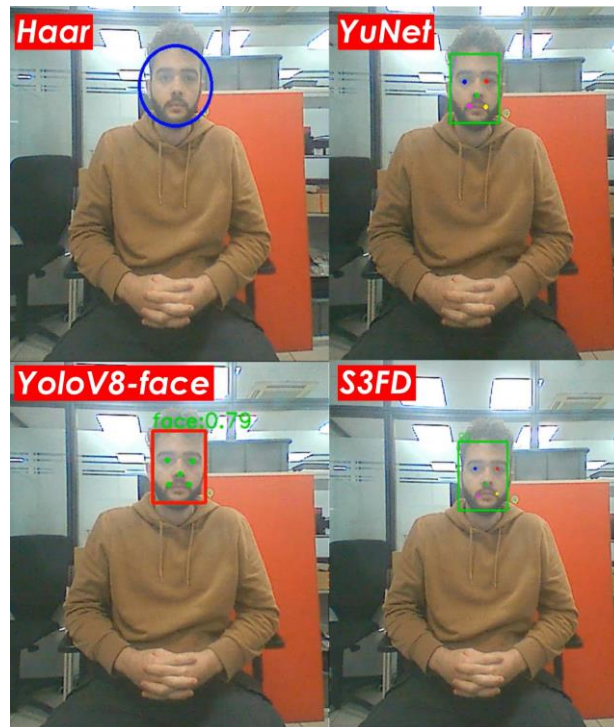
Tabla 5.2: Modelos de detección facial analizados

Teniendo toda esta información en cuenta, se confeccionó una lista de los posibles modelos de detección facial que serían empleados en las pruebas iniciales. Dicha preselección de modelos se expone en la tabla 5.2 junto con diversos atributos de estos. En la preselección de los modelos se consideraron diversos aspectos como la elección de modelos pre entrenados basados en los algoritmos más avanzados de detección facial. Y, aunque se consiguió una lista completa y variada de modelos, no todos los preseleccionados se pudieron llegar a implementar con éxito en el subsistema. Los motivos por los que no se pudieron llegar a probar determinados modelos fueron muy variados, desde problemas con la compatibilidad del formato del modelo y el módulo DNN de OpenCV, hasta problemas por el tamaño del modelo, o simplemente la falta de experiencia al utilizar el módulo de DNN. Y es que cabe resaltar que la tarea más compleja no era llegar a importar el modelo, sino conseguir averiguar qué resultado se obtenía tras realizar el proceso de inferencia. Resolver esta incógnita era clave para el éxito del proceso, ya que la implementación del código del post procesamiento del fotograma era dependiente del tipo de modelo y del formato del resultado tras la inferencia. A pesar de todas las dificultades y, tal y como se expone en la columna de “seleccionados” de la tabla 5.2, se logró implementar con éxito **seis modelos** que serían empleados en las primeras pruebas de rendimiento de la detección facial.

### 5.1.5 Pruebas iniciales

En el contexto del diseño de las pruebas iniciales para la detección facial, debemos resaltar que se crearon con el objetivo de ofrecer datos sobre el coste computacional del procesamiento de un fotograma y del pintado de los rostros detectados. Adicionalmente, también se tuvo en consideración evaluar la eficacia de los modelos bajo distintas situaciones como, por ejemplo, fotogramas en donde el usuario no muestre la cara de manera frontal a la cámara. Para evaluar dichas situaciones, se elaboraron diferentes videos grabados con la cámara del videoportero:

- **Prueba básica:** Este video está compuesto por 25 fotogramas, en los cuales se muestra a una persona frente al videoportero con una expresión neutra y sin realizar ningún tipo de movimiento con la cara. Dicha persona se encuentra a unos 50 cm de la cámara del videoportero. Cabe destacar, que este video será el seleccionado para evaluar el coste computacional del procesamiento de un fotograma y el pintado de los rostros detectados.
- **Prueba de movimiento:** En este video que consta de 175 fotogramas se muestra una persona situada a unos 80 cm del videoportero. Durante el transcurso de este, dicha persona realiza una serie de movimientos laterales con la cabeza hacia ambos lados y, para finalizar, se tapa la boca con la mano. El objetivo principal de este video se basa en evaluar si los modelos son capaces de detectar la cara ante el movimiento del rostro e, incluso, cuando la cara está siendo ocluida parcialmente.



*Figura 5.4: Resultados prueba básica*

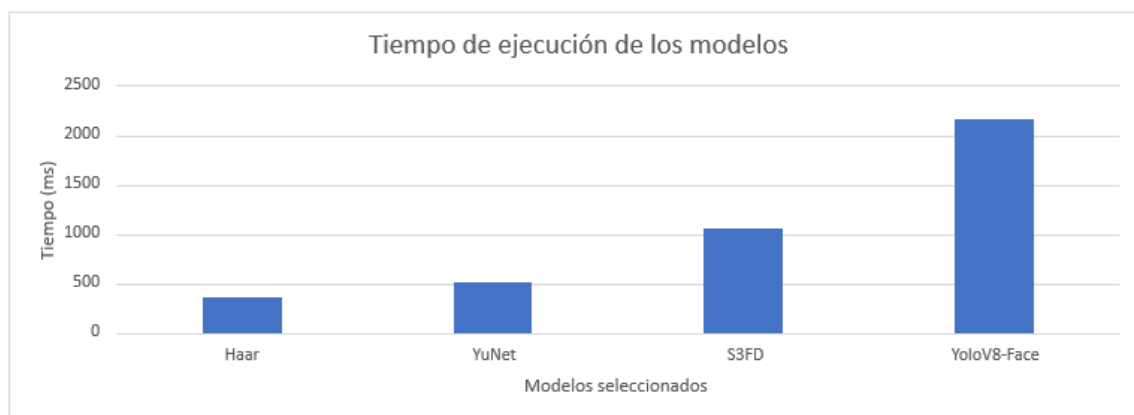
- **Prueba de pareja:** Este video pretende evaluar si los modelos son capaces de detectar correctamente y durante 50 fotogramas, dos rostros situados a unos 50 cm del videoportero.
- **Prueba de distancia:** Por último, con este video se pretende evaluar la distancia a la que el modelo es capaz de detectar la cara de una persona. Dicha persona parte de una distancia de 180 cm y, durante 175 fotogramas, se va desplazando lentamente hasta llegar a unos 50 cm con respecto a la cámara del videoportero.

Un factor clave a destacar antes de pasar con la descripción de las pruebas reside en la resolución empleada para los fotogramas de los videos. Los fotogramas que nos devuelve la cámara de nuestro videoportero se encuentran en una resolución de 1280x720. Sin embargo, tal y como se puede observar en la columna “Resolución de entrada” en la tabla 5.2, hay modelos que permiten un tipo de resolución dinámica o fija. Esto quiere decir que, para poder realizar el proceso de inferencia, hay algunos modelos que necesitan que el fotograma que utilizemos tenga una resolución fija (p.ej. 320x320), mientras que otros no necesitan una resolución en concreto y podemos emplear la que más nos convenga. Este concepto de “resolución dinámica” realmente no permite utilizar la resolución que nosotros queramos, pero este concepto se explicará más a fondo en el apartado de optimización. Asimismo, para que las pruebas fuesen lo más objetivas posibles, se tomó la decisión de emplear la misma resolución de fotogramas para todos los modelos. Por lo tanto, las características de la primera prueba sobre el rendimiento de los modelos de detección facial son:

<b>Prueba</b>	Primera prueba para analizar la velocidad de la detección facial
<b>Modelos para evaluar</b>	Haar, YuNet, S3FD, YoloV8-face
<b>Descripción de la prueba</b>	Esta prueba consiste en aplicar los diferentes modelos para cada uno de los <b>25 fotogramas</b> del video “prueba básica”. Dichos fotogramas deben estar en una resolución de <b>640x640</b> , debido a que es la única resolución posible para el modelo de YoloV8-face. Se tomarán nota sobre los datos relacionados con el coste computacional de procesar el fotograma y del pintado de las caras detectadas.
<b>Resultado de la prueba</b>	Todos los fotogramas del video serán procesados y pintados según haya detectado o no una cara. Por tanto, como resultado de la prueba se obtendrá el mismo vídeo, pero con las caras detectadas pintadas. Según el tipo de modelo, las caras detectadas se pintan por medio de circunferencias o de cuadrados
<b>Consideraciones</b>	Se respeta la relación de aspecto del fotograma original de la cámara (1280x720), por lo que se reduce a la mitad el tamaño del fotograma y se rellena de negro el espacio restante para obtener la resolución deseada.

*Tabla 5.3: Modelos de detección facial analizados*

Para realizar el análisis de los resultados de la prueba se pretende elaborar dos gráficas. En ambas gráficas se evaluará el coste de tiempo en milisegundos (eje y) frente al rendimiento de cada algoritmo (eje X). La única diferencia entre las mismas reside en que una contendrá los datos relacionados con el coste medio del procesamiento y la otra el coste medio del pintado (dibujar el cuadro delimitador en el fotograma original) de las caras detectadas. Podemos observar los resultados obtenidos tras la ejecución de la prueba en las figuras 5.4 y 5.5.



*Figura 5.5: Gráfica de los tiempos de ejecución*

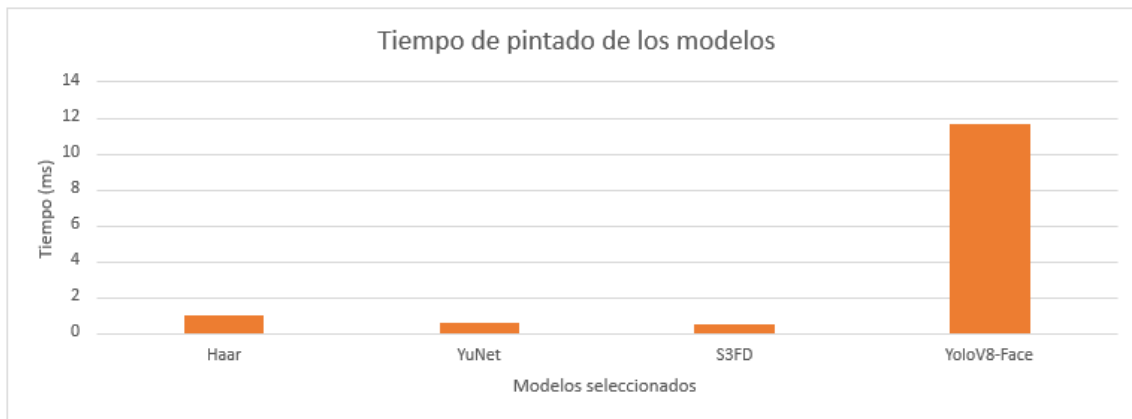


Figura 5.6: Gráfica del tiempo para el pintado de las caras detectadas

Observando los datos, podemos llegar a una serie de conclusiones. En primer lugar, podemos apreciar que el modelo YoloV8-face tarda en promedio 5 veces más en el procesamiento del fotograma que el modelo con menor promedio (Haar, < 500 ms). Por otro lado, viendo los datos de la gráfica donde se evalúa el coste de pintar las caras detectadas, podemos apreciar que el tiempo que se tarda en pintar las caras es despreciable (< 2 ms), salvo en el caso de utilizar el modelo de YoloV8-face, el cual presenta un tiempo mucho mayor que los demás modelos. Una vez sacadas las conclusiones de esta primera prueba, se procede a describir la siguiente que tiene como objetivo evaluar la eficacia de los modelos de detección facial:

Prueba	Primera prueba para analizar la eficacia de la detección facial
Modelos para evaluar	Haar, YuNet, S3FD, YoloV8-face
Descripción de la prueba	<p>Esta prueba consiste en aplicar, para cada uno de los videos elaborados previamente, los diferentes modelos a evaluar. Al igual que en la prueba anterior, los fotogramas deben estar en una resolución de <b>640x640</b>. Según el comportamiento de los modelos con cada uno de los videos, podemos designar su eficacia de la siguiente forma:</p> <ul style="list-style-type: none"> <li>- <b>Perfecto:</b> Cumple completamente con los objetivos del video</li> <li>- <b>Bien:</b> Cumple los objetivos del video, pero con algunos errores</li> <li>- <b>Regular:</b> Apenas cumple con los objetivos del video</li> <li>- <b>Mal:</b> No cumple con los objetivos del video</li> </ul>

Tabla 5.4: Modelos de detección facial analizados

Los resultados de esta primera prueba de eficacia se muestran en la tabla 5.6. Una vez analizada dicha tabla y junto con los resultados de la prueba anterior podemos apreciar que, aunque el YoloV8-face sea el modelo que mayor tiempo consume, es también el único que ha conseguido cumplir de manera impoluta los objetivos de los cuatro vídeos. Por otro lado, destacamos que el modelo Haar el cual tenía los mejores resultados en cuanto a tiempos, es también el que peor eficacia ha demostrado. Si analizamos los videos resultantes tras aplicar la prueba, podemos

apreciar que el principal problema del modelo Haar radica en los falsos positivos, ya que detecta caras en zonas donde no las hay. Como conclusión de estas primeras pruebas y tras la consulta con el tutor de la empresa, los únicos modelos con los que se seguirán realizando pruebas son el modelo **YuNet** y el **S3FD**. Esta decisión se ha tomado en base a que ambos han presentan buenos resultados en ambas pruebas, mientras que el resto de los modelos fallaba como mínimo en una de las dos.

Algoritmo	Short	Long	Couple	Howfar
Haar	Regular	Mal	Bien	Mal
YuNet	Perfecto	Bien	Perfecto	Bien
S3FD	Perfecto	Perfecto	Perfecto	Bien
YoloV8-face	Perfecto	Perfecto	Perfecto	Perfecto

*Figura 5.7: Resultados de la prueba de eficacia*

### 5.1.6 Tareas de optimización

Los resultados obtenidos en las pruebas iniciales aún estaban lejos de cumplir con el objetivo de nuestro requisito no funcional **RNF-6**. Por lo tanto, se propuso analizar todos los posibles enfoques para reducir no solo el tiempo de procesamiento del fotograma al aplicar los modelos de detección facial, sino que también el tiempo de ejecución del subsistema en general. Las opciones consideradas fueron las siguientes:

- **Minimizar la resolución del fotograma**

Desde el inicio del proyecto, se consideró esta opción como una de las primeras alternativas para reducir el coste de tiempo en el proceso de inferencia. Esto se debe en gran parte a que, como es evidente, al utilizar fotogramas de menor resolución estamos disminuyendo la complejidad computacional al procesar muchos menos píxeles, lo que se traduce en una inferencia más rápida. Para apreciar con mayor claridad cuál es el porcentaje de mejora que se consigue al ir reduciendo la resolución del fotograma, podemos observar los resultados expuestos en las gráficas de la figura 5.1, en el apartado de las pruebas finales y conclusiones.

Asimismo, y, antes de dar paso a la siguiente opción de optimización, debemos aclarar el concepto de resolución dinámica que se comentó en apartados anteriores. Al comienzo del proyecto, sobre todo en la fase de investigación y preselección de los diferentes modelos, se encontraron artículos o documentación en los que se mencionaba este concepto (Han, y otros, 2021). En la mayoría de las ocasiones se resaltaba que, de cara al usuario, este podría emplear un fotograma con cualquier resolución ya que el modelo estaba preparado para ello. Del mismo modo, esta definición expuesta de esta manera resultaba un poco inverosímil, ya que en teoría los modelos de aprendizaje profundo se entrenan empleando un rango específico de resoluciones. Finalmente,

esta incógnita se logró resolver durante las fases posteriores del proyecto. En concreto, una vez que se comenzó a indagar en el código del algoritmo de inferencia contenido en la biblioteca OpenCV, se pudo observar y concluir que el modelo de detección YuNet necesitaba que la resolución del fotograma a procesar fuese múltiplo de ocho. De hecho, en la misma clase donde se define el algoritmo de inferencia, nos encontramos con una función de preprocesamiento en la cual se modifica la resolución del fotograma a procesar hasta que cumpla con la condición comentada.

Esto nos permite concluir que, en nuestro caso, el concepto de resolución dinámica no es más que una abstracción para que el usuario final no se tenga que preocupar de cuál es la resolución de su fotograma.

- **Implementar el cambio de espacio de color manualmente**

Al proponer esta opción, nos enfocamos en mejorar el tiempo de ejecución de nuestro subsistema. Esta mejora es fundamental ya que, en términos generales, cuanto menor sea el tiempo mayor será la satisfacción de nuestros usuarios. En esta ocasión, se pretendía analizar si, implementando manualmente el proceso de conversión de espacio de color (YUV NV12 a RGB/BGR), podríamos disminuir el tiempo de ejecución. Esta idea surgió a partir de análisis del tiempo que consumía realizar dicho proceso de conversión utilizando las funciones proporcionadas por OpenCV. Además, es importante destacar que no se intentó mejorar el proceso de conversión inverso (RGB/BGR a YUV NV12) aplicando esta opción de optimización, debido a que ya se realizaba de manera manual. Esto se debía a que ninguna de las bibliotecas integradas en el sistema proporcionaba una función para realizar dicha conversión. Tras realizar la toma de tiempos, pudimos observar que en promedio el proceso de conversión YUV NV12 → RGB tardaba unos **25 ms**.

Para tratar de reducir dicho tiempo, se llevó a cabo la implementación del método expuesto en la figura 5.7. En dicho método, se comienza obteniendo los planos Y (luminancia) y UV (crominancia) del fotograma a convertir. Tras esto, se procede a recorrer cada pixel de la imagen utilizando dos bucles anidados. Dentro los bucles, se lleva a cabo el cálculo de los índices para acceder a los valores de los planos Y | UV, se realiza la conversión de espacio de color ajustando los valores para que estén en un rango válido [0, 255] y, por último, los valores calculados se almacenan en un buffer de salida (*ptrBuffer*) en el orden BGR, ya que, OpenCV emplea esta codificación para sus técnicas de visión por computador. Una vez implementado, se procedió a realizar la toma de tiempos, pero, lastimosamente, el método manual tardaba en promedio **90 ms**. Por lo tanto, tras comprobar que la conversión manual tardaba mucho más tiempo que la conversión empleando los métodos de OpenCV, se **descartó esta opción** como una posible mejora del tiempo de ejecución del subsistema.

```

void FaceDetection::colorConversionToBGR(Shared_yuvframe_ptr frameIn) {
    // Separamos los planos Y, U y V
    unsigned char * yPlane = frameIn->get_y();
    unsigned char * uvPlane = frameIn->get_uv();

    // Puntero a los datos de nuestro Mat bgr
    uint8_t* ptrBuffer = bgr.data;

    for (int y = 0; y < height; ++y) {
        for (int x = 0; x < width; ++x) {

            // Calculamos el índice del plano Y
            int index = y * width + x;
            int uvIndex = (y / 2) * width + x - (x % 2);

            // Calculamos la posición de inicio de cada plano
            int Y = yPlane[index] - 16;
            int U = uvPlane[uvIndex] - 128;
            int V = uvPlane[uvIndex + 1] - 128;

            // YUV to RGB conversion
            int R = 1.164 * Y + 1.596 * V;
            int G = 1.164 * Y - 0.392 * U - 0.813 * V;
            int B = 1.164 * Y + 2.017 * U;

            // Ajustamos los valores a un rango válido [0, 255].
            R = std::min(std::max(R, 0), 255);
            G = std::min(std::max(G, 0), 255);
            B = std::min(std::max(B, 0), 255);

            // Añadimos los resultados al buffer de salida
            int bgrIndex = index * 3;
            ptrBuffer[bgrIndex] = static_cast<unsigned char>(B);
            ptrBuffer[bgrIndex + 1] = static_cast<unsigned char>(G);
            ptrBuffer[bgrIndex + 2] = static_cast<unsigned char>(R);
        }
    }
}

```

Figura 5.8: Código de la conversión manual

- Crear un *shader* para el cambio de espacio de color en la GPU

Otra de las posibles opciones que se barajaron para reducir el tiempo de ejecución de los procesos de conversión de espacio de color, tenía como principal objetivo desarrollar un *shader* para realizar dicho proceso a través de la GPU. Anteriormente, este proceso de conversión se estaba realizando a través de los métodos incluidos en la biblioteca de OpenCV, por lo que, la CPU era la encargada de llevar a cabo la conversión. De este modo, la idea de esta opción de optimización se fundamentaba en el hecho de que las GPU están diseñadas y optimizadas específicamente para realizar tareas de procesamiento de imágenes, lo que implica que debería de costar menos tiempo llevar a cabo la conversión en la GPU que en la CPU. Como consecuencia, se debían de realizar una serie de cambios al flujo que habíamos establecido para nuestro subsistema.



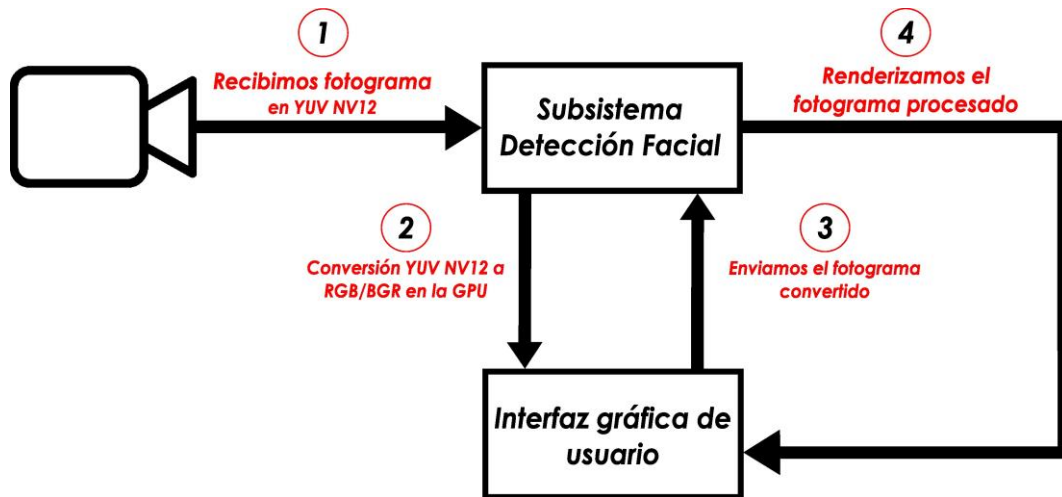


Figura 5.9: Nuevo flujo con el shader

Podemos observar la nueva propuesta para el flujo del subsistema en la figura 5.8. En esta ocasión, una vez se ha obtenido el fotograma de la cámara, se enviará a la interfaz gráfica de usuario (GUI) para poder aplicarle el *shader* de conversión de espacio de color de YUV NV12 a RGB/BGR. A partir de aquí, se realizaría el procesamiento del fotograma aplicando el modelo de detección tal cual se realizaba en versiones anteriores. Una vez terminado el procesamiento, se volvería a enviar el fotograma a la interfaz gráfica, pero esta vez con el objetivo de simplemente renderizar el resultado por la pantalla del videoportero. En esta ocasión, no se debe realizar ningún tipo de conversión ya que el fotograma posee el formato adecuado para su renderizado.

Durante la implementación del nuevo flujo para el subsistema, se presentaron numerosos desafíos que se tuvieron que abordar. Por ejemplo, uno de los primeros desafíos consistía en analizar y estudiar en detalle todos los aspectos relevantes al flujo de comunicación entre nuestro subsistema y la interfaz gráfica de usuario. Tener una visión clara de cómo estaba desarrollada dicha comunicación, nos proporcionaba una ventaja significativa a la hora de querer modificarla. Del mismo modo, el desarrollo del *shader* para realizar la conversión de espacio de color fue otra tarea que, inicialmente, demandaba tiempo y esfuerzo; sin embargo, gracias a la experiencia previa en la creación de este tipo de componentes, no representó un desafío excesivamente complejo.

El principal problema para llevar a cabo la implementación fue determinar cómo almacenar el resultado de aplicar el *shader* de conversión al fotograma, ya que debíamos encontrar la manera de extraer los resultados de la conversión en la GPU y almacenarlos de tal manera que la CPU pudiera emplear dichos datos a posteriori. Finalmente, indagando en la documentación de OpenGL (biblioteca que se empleaba como API para renderizar las imágenes) se logró encontrar la solución a nuestro problema. En nuestro caso debíamos utilizar el objeto “frameBuffer” o marco de trama, el cual, es un objeto de OpenGL que permite almacenar buffers de datos relacionados con la imagen que se va a renderizar. Este tipo de objeto nos permite almacenar en memoria el resultado de aplicar el *shader* a la imagen. Por lo que, empleando este tipo de objetos, se consiguió obtener los datos del fotograma convertido a RGB/BGR para su posterior procesamiento con el modelo de detección facial.

Tras la implementación del nuevo flujo se procedió a realizar nuevamente la toma de tiempos para comparar los resultados de la conversión en la GPU frente a la CPU. En esta ocasión, se observó como la conversión en la GPU tardaba en promedio un 94% menos que la conversión en la CPU (900 us a 15 ms). Por lo tanto, los datos obtenidos coinciden con nuestra hipótesis inicial, confirmando que el proceso de conversión de espacio de color es más rápido si se ejecuta en la GPU que en la CPU. Desafortunadamente, en etapas posteriores del desarrollo del trabajo se determinó que el flujo propuesto en la figura 5.3 no sería el definitivo. Esta decisión se tomó en base a los resultados obtenidos una vez se comenzó a desarrollar la prueba de concepto final del trabajo. Al analizar el coste de tiempo de cada una de las partes del subsistema, se descubrió que la función de OpenGL “glReadPixels()”, empleada para almacenar el contenido del fotograma convertido a RGB/BGR, causaba un importante incremento en el tiempo total del subsistema ya que de por si este método tardaba en promedio **112 ms**. Atendiendo a estos resultados insatisfactorios, se llegó a la conclusión de que, debido a la limitación de memoria disponible en el videoportero para manejar la carga de lectura y escritura con mayor velocidad, se **descartaría esta opción** para optimizar el tiempo del subsistema.

- **Paralelizar el proceso de inferencia**

Una de las opciones finales que se consideraron para reducir el tiempo de ejecución del proceso de inferencia fue paralelizar dicho proceso. En resumen, se pretendía dividir el fotograma a procesar en ‘n’ partes y asignar un hilo de ejecución a cada una. Dichos hilos serían los encargados de realizar el proceso de inferencia con su respectiva sección del fotograma. En última instancia, los resultados obtenidos de la ejecución de todos los hilos (posiciones de caras detectadas) se utilizarían para pintar las caras detectadas sobre el fotograma original. Esta opción presenta una clara problemática desde un inicio: dependiendo de cómo se realice la división del fotograma, será necesario gestionar los casos en los que una cara sea recortada, ya que esto haría imposible su detección. A pesar de esto y, dado que la paralelización del proceso de inferencia no resultaba un proceso complejo de realizar, se decidió, en conjunto con el tutor de la empresa, seguir adelante con esta opción y evaluar con los futuros resultados si la mejora obtenida era lo suficientemente satisfactoria como para lidiar con el problema de las caras cortadas.

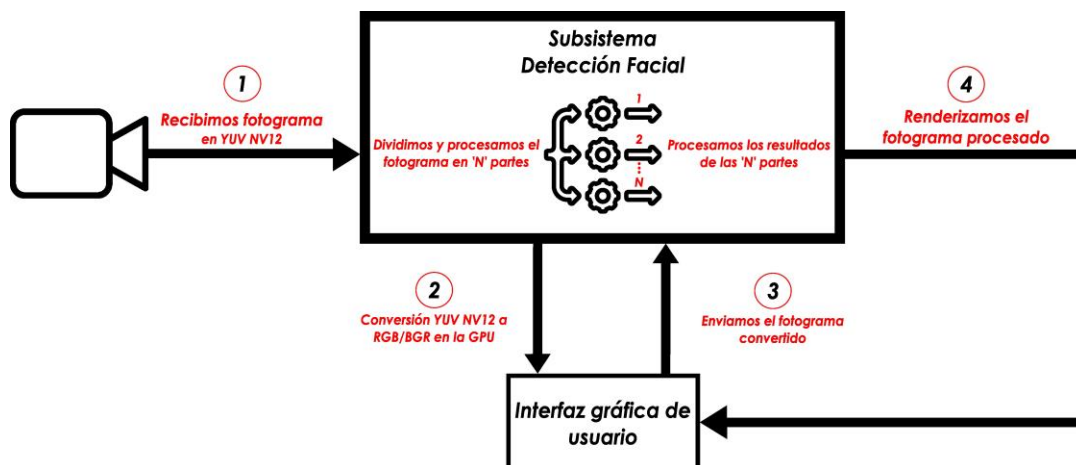
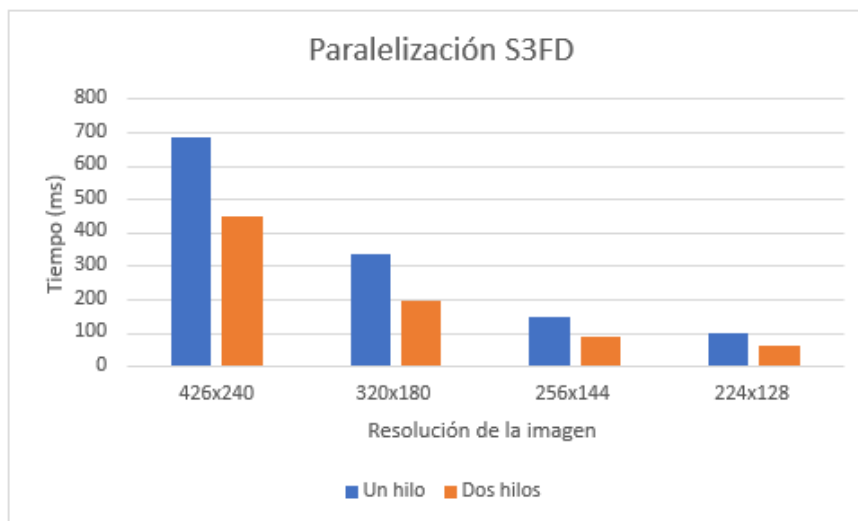


Figura 5.10: Nuevo flujo para la paralelización

El diseño de la estructura empleada en el proceso de paralelización se puede observar en la figura 5.9. Tras realizar la conversión del fotograma en la GPU, se decidirá en cuantos fragmentos se quiere dividir y se creará un hilo de ejecución por cada uno de dichos fragmentos. Cada hilo de ejecución contará con los atributos necesarios para llevar a cabo el proceso de inferencia con su respectivo fragmento de la imagen. El hilo principal esperará a que termine la ejecución de todos los hilos y obtendrá, como resultado, una estructura de datos que contendrá las coordenadas de las caras que han detectado los distintos hilos. Con dicho resultado, el último paso consiste simplemente en pintar las caras detectadas sobre el fotograma original.

Las primeras pruebas que se realizaron una vez se implementó el proceso de paralelización, se llevaron a cabo con el modelo S3FD. Inicialmente, este modelo presentaba diversos problemas a la hora de paralelizar el proceso de inferencia. En concreto, se obtuvo un error de segmentación cuyo origen no se lograba identificar. La solución a dicho problema se encontró finalmente en la sección de “issues” del GitHub oficial de la biblioteca libfacedetection. En esta sección, el profesor y autor de la biblioteca, ShiqiYu, explica que la primera vez que se realiza la llamada a la función ‘facedetect\_cnn’ (método para procesar la inferencia en libfacedetection), se realizan tareas de inicialización de algunos parámetros necesarios para la inferencia. Por lo tanto, si se quiere llamar a la función en paralelo, es necesario realizar una llamada a la función previa a comenzar la ejecución de la sección paralela. Una vez solucionados todos los inconvenientes, se optó por diseñar una prueba para evaluar si la paralelización del proceso de inferencia resulta en una mejora en el coste de tiempo. En la prueba mencionada, se dividió el fotograma original verticalmente, obteniendo dos secciones para su procesamiento. Los resultados obtenidos de dicha prueba se pueden observar en la siguiente gráfica:

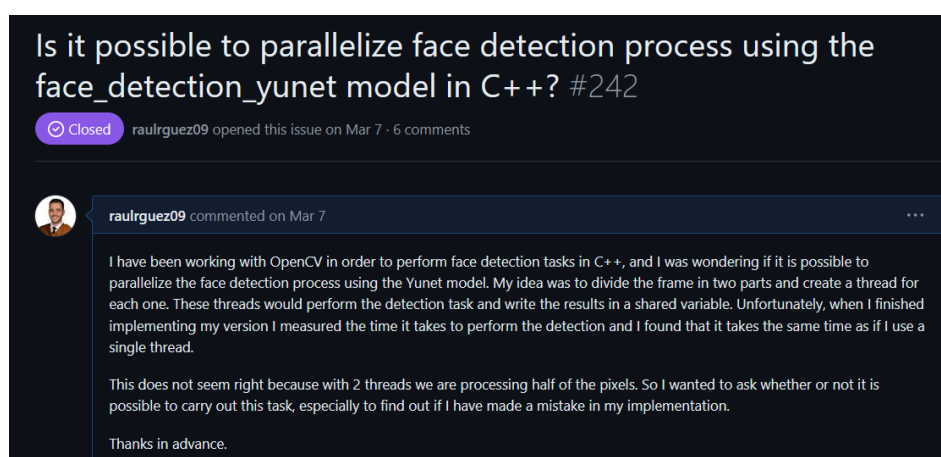


*Figura 5.11: Resultado de la paralelización con el modelo S3FD*

Analizando la gráfica 5.10 podemos apreciar que, paralelizando el proceso de inferencia con el modelo S3FD, conseguimos una mejora significativa en el coste del tiempo de dicho proceso. Del

mismo modo, antes de tomar una decisión final acerca de la paralelización, se debía comprobar que dicho proceso también supusiera una mejora significativa con el modelo YuNet.

En el contexto del modelo YuNet, se empleó el mismo diseño para la sección paralela que en el caso del modelo S3FD. Por lo tanto, tras finalizar la fase de implementación, se aplicó la misma prueba que se había utilizado previamente para evaluar el impacto de la paralelización. En esta ocasión, el problema principal no se basó en ningún error de segmentación, sino en los mismos resultados de la prueba. Dichos resultados indicaron que el tiempo de inferencia no mejoraba con la paralelización; de hecho, en muchos de los casos, suponía un ligero aumento del tiempo. Este análisis carecía de sentido desde una perspectiva teórica, por lo que, en busca de una explicación razonable, se dedicó tiempo a examinar minuciosamente el código de la biblioteca OpenCV para procesar la inferencia del modelo YuNet. Inicialmente, no se pudo llegar a ninguna conclusión detrás de los resultados insatisfactorios. Sin embargo, tras abrir una consulta en la sección “Issues” del repositorio de GitHub OpenCV\_ZOO, finalmente se encontró la respuesta a nuestra incógnita.



*Figura 5.12: Consulta al GitHub de opencv\_zoo*

Uno de los miembros de la organización de OpenCV se ocupó de responder a todas las dudas que se habían planteado en la consulta. En primer lugar, explicó que el motivo por el que los resultados no eran satisfactorios era debido a que el proceso de detección facial en OpenCV ya estaba siendo paralelizado. Tras esta respuesta, los resultados obtenidos cobraron sentido, ya que, como es evidente, paralelizar un proceso que ya está siendo paralelizado no garantiza mejoras significativas en el tiempo. No conforme con esto, el mismo miembro nos recomendó dos posibles enfoques que podíamos utilizar para mejorar el coste del tiempo en la inferencia. Por un lado, nos recomendó reducir la resolución del fotograma, enfoque que ya habíamos comprobado su eficacia en apartados anteriores. Y, por otro lado, nos recomendó ejecutar el proceso de inferencia por lotes de fotogramas. Esto quiere decir que, modificando el propio código de inferencia, se podrían procesar varios fotogramas en la misma ejecución. Con esta información, se tomó la decisión de abordar y probar si, aplicando el segundo enfoque, podríamos obtener una mejora significativa en el coste del tiempo. Es decir, el objetivo era verificar si el tiempo necesario para procesar dos fotogramas en una sola ejecución era significativamente menor que el tiempo requerido para procesar esos dos fotogramas en dos ejecuciones separadas.

El mayor desafío de esta tarea, además de tener que trabajar directamente con el código de inferencia sin emplear la API integrada de OpenCV, consistía en saber gestionar el resultado del proceso de inferencia. El resultado en cuestión incluiría los datos sobre las caras detectadas de los 'n' fotogramas que se quieran procesar. Por tanto, conocer dónde finalizan los datos de un fotograma y dónde comienzan los datos del siguiente, era esencial para obtener un resultado coherente en la detección facial. Una vez resueltos estos desafíos, se procedió a evaluar el impacto que suponía trabajar con lotes de fotogramas. Los resultados indicaron que, en comparación con el procesamiento individual de fotogramas, no había una mejora significativa en el tiempo al procesar por lotes de fotogramas. En consecuencia, se determinó que la única opción viable para reducir el coste de tiempo del proceso de inferencia con el modelo Yunet era reducir la resolución de la imagen.

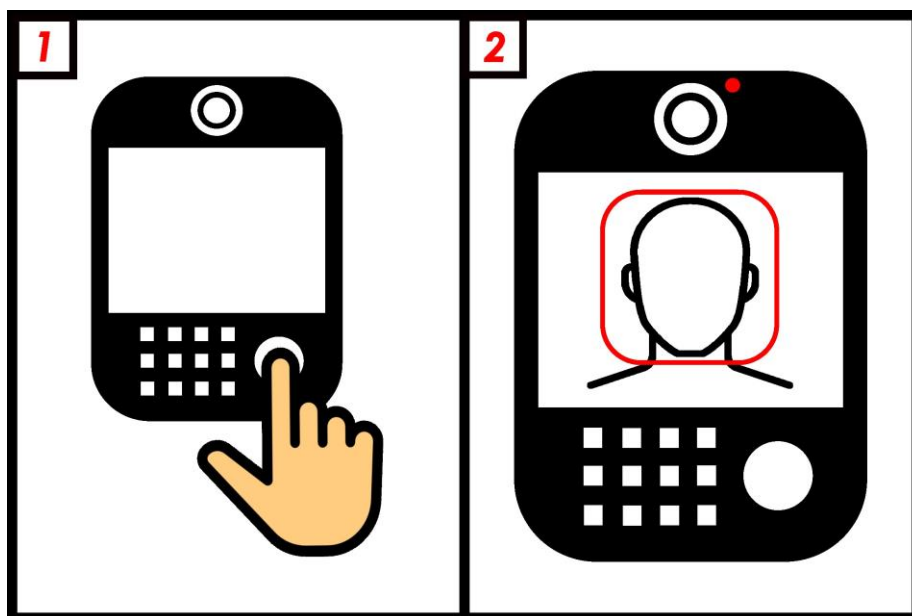
En conclusión, tras analizar los resultados que se obtuvieron con las pruebas realizadas en ambos modelos, se optó por **descartar la opción de paralelizar** el proceso de inferencia. Eso se debe a que después de evaluar los resultados obtenidos al aplicar la paralelización en el modelo S3FD y, considerando la complejidad de gestionar las caras cortadas, así como la imposibilidad de aplicar este proceso al modelo YuNet, se determinó que el esfuerzo necesario para seguir aplicando esta opción no justificaba la inversión.

- **Procesamiento selectivo de fotogramas**

Uno de los enfoques más utilizados para reducir la carga y el tiempo de procesamiento de las tareas de visión por computador se basa en realizar un procesamiento selectivo de los fotogramas. Esto quiere decir que, en lugar de procesar todos los fotogramas aplicando la funcionalidad de visión por computador deseada, se pueda procesar, por ejemplo, uno de cada cinco fotogramas. Este enfoque se basa en la observación de que la diferencia entre los fotogramas al transcurrir una cantidad mínima de tiempo es prácticamente nula. Por lo tanto, en el contexto de la detección facial, podemos procesar y representar uno de cada 'x' fotogramas sin afectar negativamente la experiencia del usuario. En contraste con los demás enfoques desarrollados en esta sección, para evaluar el impacto del procesamiento selectivo de fotogramas en el subsistema, es necesario esperar a la implementación de la estructura definitiva del subsistema. Por consiguiente, dado que dicha estructura se describirá en el próximo capítulo, será entonces cuando se expongan las pruebas y decisiones tomadas acerca de este enfoque.

### **5.1.7 Pruebas finales y conclusiones**

Tras analizar las conclusiones obtenidas en la fase de optimización, se procede a realizar las pruebas finales para evaluar y seleccionar el mejor modelo de detección facial de entre nuestros candidatos.



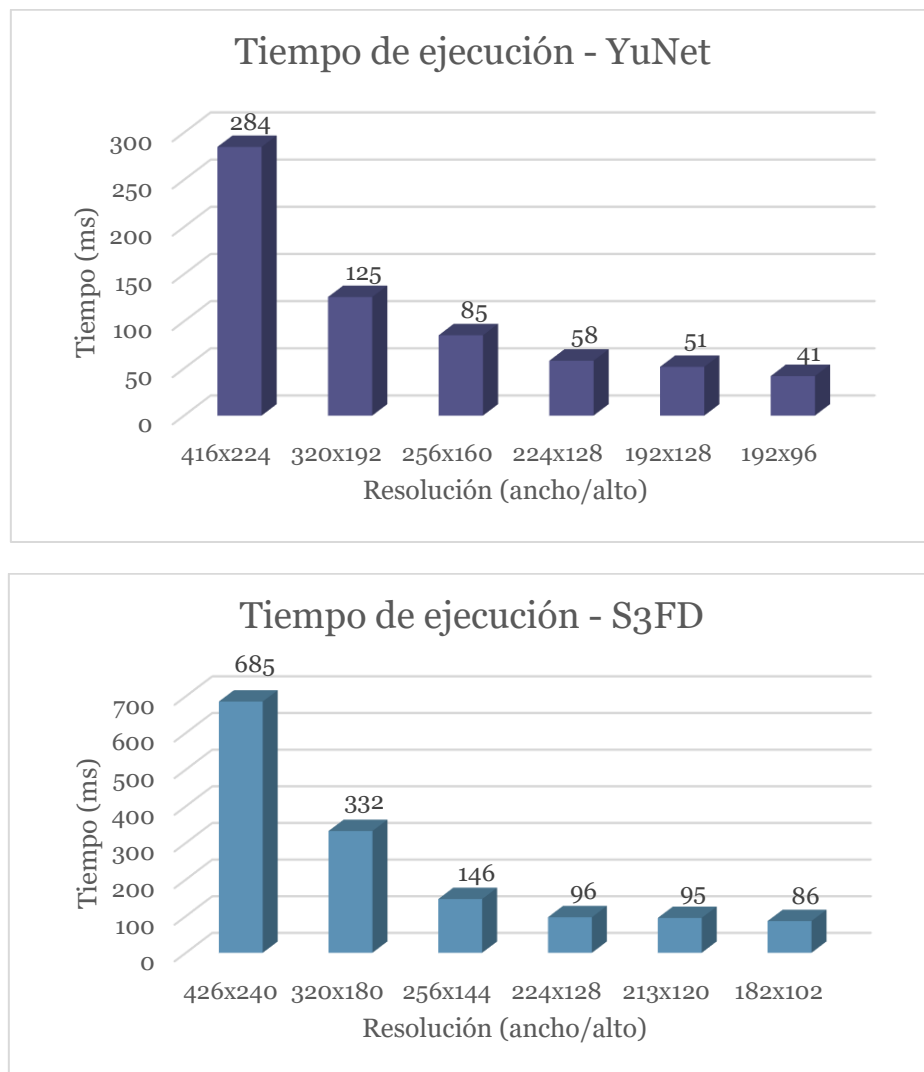
*Figura 5.13: Flujo de las pruebas finales*

En esta ocasión, se ha diseñado una pequeña demo para llevar a cabo las pruebas, en lugar de utilizar videos pregrabados como en las pruebas iniciales. En la figura 5.12, podemos ver una descripción de los pasos a seguir para la demo de las pruebas. En primer lugar, esta demo requiere que el usuario se ponga en frente del videoportero. Una vez de frente al dispositivo, el usuario pulsará cualquiera de los botones disponibles, provocando que la pantalla se encienda y muestre la vista de la detección facial. En esta vista, la cámara se activa y el usuario puede verse en la pantalla del videoportero a modo de espejo. Internamente, la cámara realiza la tarea de enviar uno de cada tres fotogramas capturados para aplicar el proceso de detección facial. En el caso de detectar caras, los datos asociados a la posición de dichas caras se envían a la GUI para su renderizado. Por lo tanto, a efectos del usuario, una vez comienza la vista de la detección facial, verá cómo se dibuja un cuadro delimitador por cada una de las diferentes caras que aparezcan por pantalla.

- **Prueba de velocidad y precisión**

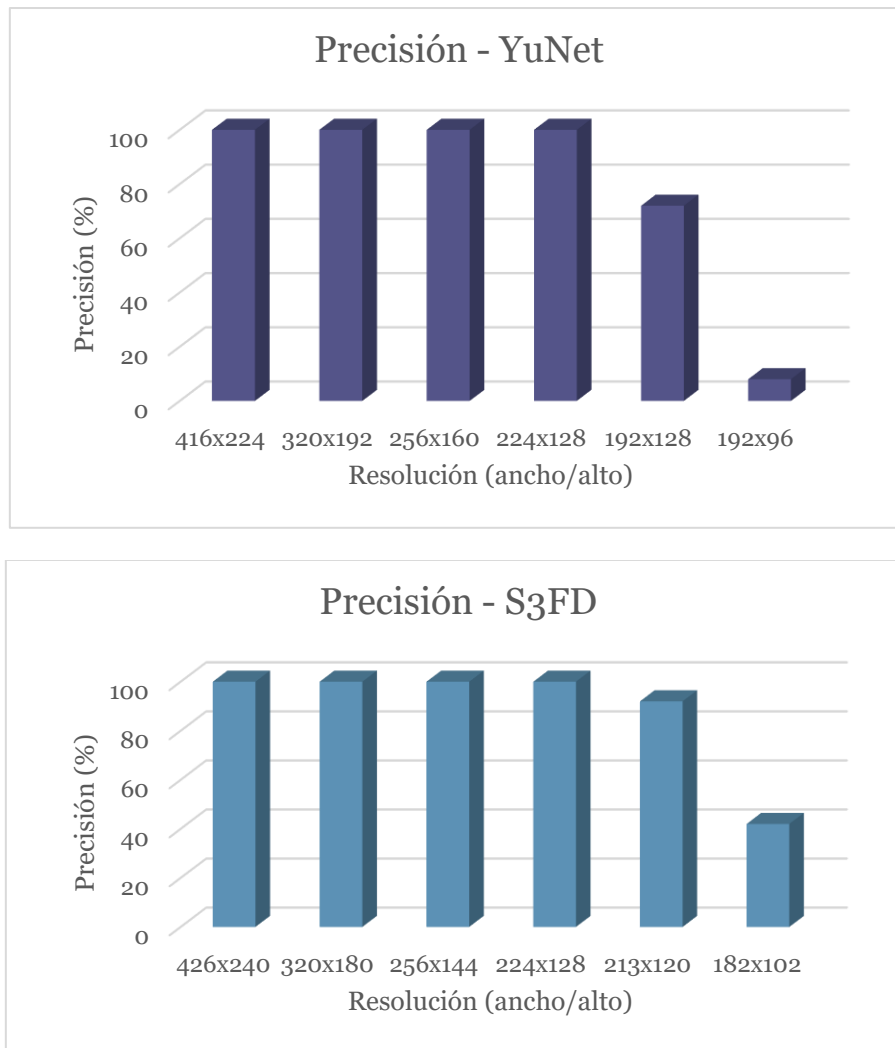
Al igual que vimos en las pruebas iniciales, uno de los factores a evaluar será el coste de tiempo del proceso de inferencia para cada uno de los modelos. Es decir, el tiempo que se tarda para procesar el fotograma y obtener las posiciones de las caras detectadas. Del mismo modo, hay que destacar que se añadió un nuevo factor a evaluar para estas pruebas finales, dicho factor se basa en la precisión. Este atributo medirá la relación entre el número de fotogramas en donde se detectan caras y el número total de fotogramas. El desarrollo de la prueba es simple, un único usuario se pondrá de frente al videoportero a una distancia de unos 50 cm. Una vez en la posición correcta, el usuario activará la vista de detección facial y dejará que se procese su imagen durante unos segundos. Al concluir este proceso, se obtendrán los datos con el tiempo de ejecución del proceso de inferencia, así como la información relevante sobre si se ha detectado o no la cara en

el fotograma. Los resultados obtenidos de la prueba de velocidad y precisión los podemos observar en las figuras 5.13 y 5.14.



*Figura 5.14: Resultados prueba final de velocidad – Detección facial*

En primer lugar, la figura 5.13 muestra las gráficas que analizan el coste de tiempo promedio, para cada uno de los modelos, con respecto a las distintas resoluciones aplicadas al fotograma. El valor promedio del tiempo se ha calculado a partir de 100 muestras. Es relevante señalar que las resoluciones entre ambos modelos difieren. Eso se debe a que, para evitar el reescalado que realiza el modelo YuNet con el fin de obtener una resolución apropiada (múltiplo de ocho), se optó por establecer directamente la resolución adecuada para dicho modelo. Si analizamos brevemente los resultados, apreciamos a simple vista que el modelo YuNet presenta mejores resultados en cuanto al tiempo. Esto se puede comprobar al observar que el modelo YuNet presenta tres resoluciones por debajo de los 50 ms, mientras que, en el caso del S3FD, todas las resoluciones sobrepasan los 80 ms.



*Figura 5.15: Resultados prueba final de precisión – Detección facial*

Tras analizar las gráficas sobre el coste de tiempo, pasamos ahora a analizar las gráficas de precisión expuestas en la figura 5.14. En dichas gráficas, se muestra la relación entre el porcentaje de fotogramas en los que se ha detectado la cara del usuario para cada una de las resoluciones aplicadas. Al analizar los resultados expuestos, concluimos que, para ambos modelos, utilizar una resolución inferior a 224 x 128 conlleva una pérdida de precisión.

- **Prueba de eficacia**

Como se ha mencionado en capítulos anteriores, la detección precisa de rostros en un fotograma puede verse afectada por diversas condiciones (oclusión parcial de la cara, condiciones lumínicas adversas, accesorios faciales, etc.). Por lo tanto, para evaluar la eficacia de los modelos, estos se han sometido a una prueba completa que evalúa su rendimiento para cada uno de estos escenarios. Por ejemplo, para evaluar el caso en el que el usuario lleve gafas de sol, la prueba aplicada consiste en: establecer la resolución del fotograma y la distancia a la que tiene que estar el usuario y, a continuación, empleando la demo comentada al inicio del capítulo, comprobar que se detecta y



pinta correctamente la cara del usuario mientras lleva gafas de sol. En la tabla de la figura 5.15, se muestran los resultados obtenidos en la prueba de eficacia a través del siguiente código de color: verde (El objetivo del escenario se cumple siempre), amarillo (El objetivo del escenario se cumple con algunas excepciones), y rojo (El objetivo del escenario no se cumple).

	Resolución de la imagen	Distancia	Caras de frente	Caras de lado	Accesorios faciales				Expresiones faciales	Condiciones lumínicas			Max num de caras	Oclusión parcial		
					Gafas	Gafas de sol	Mascarilla	Gorra		Amanecer	Mediodía	Noche		Mitad dcha de la cara	Mitad izda de la cara	Parte inferior de la cara
	Resolución final de la imagen para aplicar la detección	"Detecta y pinta la cara a una distancia determinada"	"Detecta y pinta las caras en posición frontal"	"Detecta y pinta las caras laterales"	"Detecta y pinta las caras que llevan puesto gafas"	"Detecta y pinta las caras que llevan puesto gafas de sol"	"Detecta y pinta las caras que llevan puesto una braga en el cuello"	"Detecta y pinta las caras que llevan puesto una gorra"	"Detecta y pinta las caras con expresiones faciales muy marcadas"	"Detecta y pinta las caras con la luz del amanecer"	"Detecta y pinta las caras con la luz del mediodía"	"Detecta y pinta las caras por la noche"	"Detecta y pinta más de 6/8 caras"	"Detecta y pinta caras tapando la mitad derecha de su cara"	"Detecta y pinta caras tapando la mitad izquierda de su cara"	"Detecta y pinta caras tapando la parte inferior de su cara"
YUNET	320x192	20 60 140	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✗ ✗ ✗	✓ ✓ ✓
	256x160	20 60 130	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✗ ✗ ✗	✓ ✓ ✓
	224x128	20 60 110	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✗ ✗ ✗	✓ ✓ ✓
	192x128	20 60 100	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✗ ✗ ✗	✓ ✓ ✓
	256x144	20 60 120	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✗ ✗ ✗	✓ ✓ ✓
	213x120	20 60 110	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✗ ✗ ✗	✓ ✓ ✓
S3FD	182x102	20 60 80	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✓ ✓ ✓	✗ ✗ ✗	✗ ✗ ✗	✓ ✓ ✓

Figura 5.16: Tabla resultado de las pruebas de eficacia

Analizando la tabla de izquierda a derecha, observamos que en las primeras columnas se definen las diferentes resoluciones empleadas para cada modelo, junto con la distancia establecida para posicionar al usuario con respecto al videoportero. A partir de aquí, por cada columna se definen diversas situaciones que pueden afectar al correcto funcionamiento de la detección facial. Estas situaciones se pueden agrupar de la siguiente manera:

- **Orientación de la cara:** Estas pruebas se realizaron cambiando la orientación de la cara del usuario con respecto a la cámara del videoportero. Analizando los resultados, la orientación de la cara no supone un inconveniente para la detección facial aplicando ambos modelos.
- **Expresiones faciales:** En esta ocasión se investigaron las situaciones en las que el usuario no tuviera una expresión facial neutra, sino que estuviera riendo, triste, con la boca abierta, etc. Los resultados obtenidos de esta situación fueron muy satisfactorios, concluyendo que la diversidad de expresiones faciales no supone un problema para la detección.
- **Accesorios faciales:** Para representar estas situaciones, el usuario de prueba debía llevar diversos accesorios faciales en el transcurso de la prueba. Analizando los resultados, podemos observar que el accesorio facial que perjudica en mayor medida a la detección es la braga/mascarilla. Este resultado permite recordar la época de la pandemia por el Covid-19. En aquel momento, debido a la falta de entrenamiento de los modelos de detección frente a personas con mascarillas, se desarrollaron modelos exclusivos para la detección de rostros ante estas situaciones (Sethi, Kathuria, & Kaushik, 2021).

- **Condiciones lumínicas:** Las condiciones de iluminación suelen afectar significativamente a diversas funcionalidades del campo de la visión por computador. Para estas pruebas, se establecieron diferentes situaciones en donde la luz (natural) diera al usuario desde distintos ángulos. Además, se simuló un entorno de completa oscuridad en donde el único punto de luz fuese la propia iluminación del videoportero. Como se observa en la tabla, estas situaciones no supusieron ningún obstáculo para la detección facial.
- **Oclusión parcial de la cara:** El propósito de esta situación se basa en evaluar si el modelo es capaz de detectar una cara que está siendo parcialmente ocluida por un obstáculo, en este caso, un folio. Como se puede apreciar, los resultados ante estas situaciones no han sido favorables. Del mismo modo, debemos destacar una situación que permanece sin resolver hoy en día. En el caso de ocluir la cara con otros elementos como, la propia mano o un trapo de cocina, los resultados que se obtuvieron fueron más satisfactorios. Esto nos llevó a la conclusión de que, dependiendo del tipo de material con el que se cubra parte de la cara, los modelos pueden tener un rendimiento mejor o peor.

Si analizamos las pruebas de eficacia en su conjunto, podemos concluir que ambos modelos tuvieron un desempeño muy similar en la mayoría de las situaciones. Como único caso a destacar, se podría afirmar que el modelo YuNet rinde un poco mejor en el caso de que el usuario llevara puesto una braga o mascarilla. Del mismo modo, evaluando los resultados obtenidos tanto en esta prueba, como en la de velocidad y precisión, se concluyó que el modelo de detección más adecuado para nuestro sistema era el **YuNet**. En consecuencia, **OpenCV** fue la biblioteca seleccionada para llevar a cabo el resto de las tareas del proyecto.

## 5.2 Reconocimiento facial

---

En este capítulo del proyecto se expondrán todas aquellas tareas relacionadas con el reconocimiento facial. Después de verificar que la detección facial es un proceso viable para integrar en nuestro limitado sistema, el objetivo principal de esta sección es evaluar la factibilidad de integrar el reconocimiento facial. Esta evaluación es crucial para los intereses de la empresa, ya que una vez se integre el reconocimiento facial en el videoportero, se podría desarrollar una prueba de concepto que incluya esta funcionalidad como parte del sistema de acceso a la vivienda. Los resultados extraídos de dicha prueba serán utilizados por la empresa para evaluar la viabilidad de integrar el reconocimiento facial en sus productos actuales o futuros.

### 5.2.1 Primero pasos

Como conclusión de las pruebas realizadas para la detección facial, OpenCV había sido elegida para ser la biblioteca principal con la que realizar el resto de las tareas sobre la visión por computador. En consecuencia, una vez se comenzó con las tareas de investigación sobre los modelos de reconocimiento facial, el primer paso fue comprobar si OpenCV utilizaba algún

modelo en concreto para esta tarea. Tras una breve búsqueda, se comprobó que, en efecto, OpenCV empleaba el modelo **SFace** para todas las tareas de reconocimiento facial. Este modelo ha presentado unos sólidos resultados haciendo uso de una novedosa función de pérdida denominada *sigmoid-constrained hypersphere loss*. Esta función, en conjunto con la estrategia de optimizar los objetivos intraclase e interclase, busca mitigar el problema del sobreajuste. El objetivo principal es permitir que el modelo aprenda a distinguir entre muestras de diferentes clases sin exagerar las diferencias dentro de una misma clase.

Cabe destacar que, en la propia documentación de OpenCV, se puede encontrar un tutorial de como implementar el reconocimiento facial haciendo uso de una API integrada. Con esta información, se propuso la idea de, antes de seguir investigando sobre más modelos de reconocimiento facial, completar dicho tutorial y llevar a cabo una prueba para **evaluar el rendimiento** y la eficacia del modelo **SFace**. Para llevar a cabo esta prueba, se emplearon imágenes de tipo DNI de personas seleccionadas al azar de internet, así como dos imágenes de un compañero de trabajo del equipo de I+D: una actual y otra de hace dos años. Los resultados obtenidos con el modelo fueron altamente prometedores, ya que no solo lograba reconocer las caras de manera precisa, sino que, además, el tiempo promedio del proceso de inferencia era de aproximadamente **650 ms**. Tras una reunión con el tutor de la empresa y en vista de los buenos resultados, se optó por dejar de lado la investigación de más modelos para el reconocimiento facial y utilizar el modelo SFace. Es importante destacar que esta decisión se tomó, en parte, por la considerable inversión de tiempo y esfuerzo que supuso la investigación e implementación de los modelos de detección facial. Así pues, con esta consideración en mente y, teniendo en cuenta que los resultados con el modelo SFace habían sido positivos, se decidió que dicho modelo formaría parte de la prueba de concepto que se empezaría a desarrollar a partir de este momento.

### 5.2.2 Proceso de verificación de identidad

Tras los sorprendentes resultados con respecto a la velocidad y eficacia por parte de las tareas de detección y reconocimiento facial, se propuso desarrollar una prueba de concepto donde se pusieran a prueba estas funcionalidades en una situación real. En concreto, la prueba de concepto tiene como principal objetivo desarrollar un sistema de verificación de identidad.

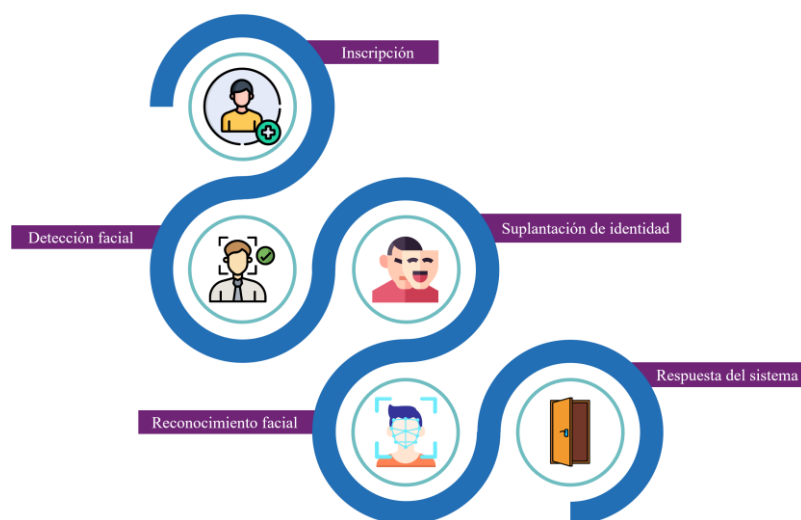


Figura 5.17: Sistema de verificación de identidad

En la figura 5.16 se muestran todas las etapas que conforman el sistema de verificación de identidad a implementar. Con respecto al trabajo previo, podemos destacar que se incluyen tres nuevas funcionalidades: Inscripción del usuario, prevención ante una posible suplantación de identidad y la respuesta del sistema. Por lo tanto, a lo largo de esta sección, se definirán todas las etapas del sistema, exponiendo sus principales características, así como los resultados obtenidos con las respectivas pruebas aplicadas. Además, debido a que la implementación de esta prueba de concepto no estaba prevista en un inicio del proyecto, se proceden a listar los respectivos requisitos relacionados a cada una de las nuevas funcionalidades:

<b>ID</b>	<b>Descripción del requisito funcional</b>
RF-22	Generar la biometría facial de un usuario a partir de una imagen de este
RF-23	Almacenar la biometría facial de los usuarios en el propio sistema
RF-24	Realizar la comparación de biometrías para identificar a los usuarios que tratan de acceder a la vivienda
RF-25	Abrir la puerta de la vivienda siempre y cuando se haya reconocido al usuario
RF-26	Denegar el acceso si no se ha encontrado un usuario con la misma biometría en la base de datos
RF-27	Discernir si se encuentra bajo un ataque de suplantación de identidad
RF-28	Denegar el acceso a la vivienda si se ha detectado un ataque de suplantación de identidad
RF-29	Continuar con el proceso de reconocimiento facial si no se ha detectado un ataque de suplantación de identidad

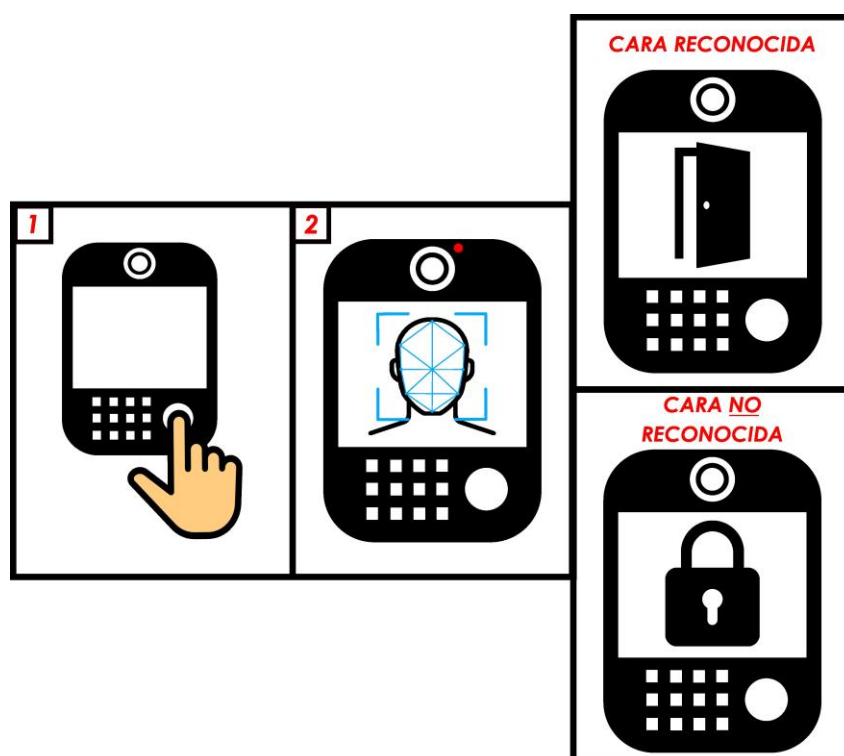
*Tabla 5.5: Nuevos requisitos funcionales del sistema*

<b>ID</b>	<b>Descripción del requisito NO funcional</b>
RNF-17	El registro de usuario debe poder realizarse usando la propia placa o utilizando una imagen
RNF-18	El registro de usuario se realizará con una imagen cuya resolución no sea inferior a 480x640
RNF-19	Cada biometría facial se exportará y almacenará en un fichero dentro del propio sistema del videoportero
RNF-20	Cada fichero solo tendrá exclusivamente los datos de la biometría facial de una única cara
RNF-21	Se debe identificar como ataques los intentos de acceder al sistema usando imágenes impresas o videos con un dispositivo móvil o Tablet
RNF-22	El tiempo de ejecución del proceso para determinar si se trata de un ataque no debe durar más de 500 ms

*Tabla 5.6: Nuevos requisitos NO funcionales del sistema*

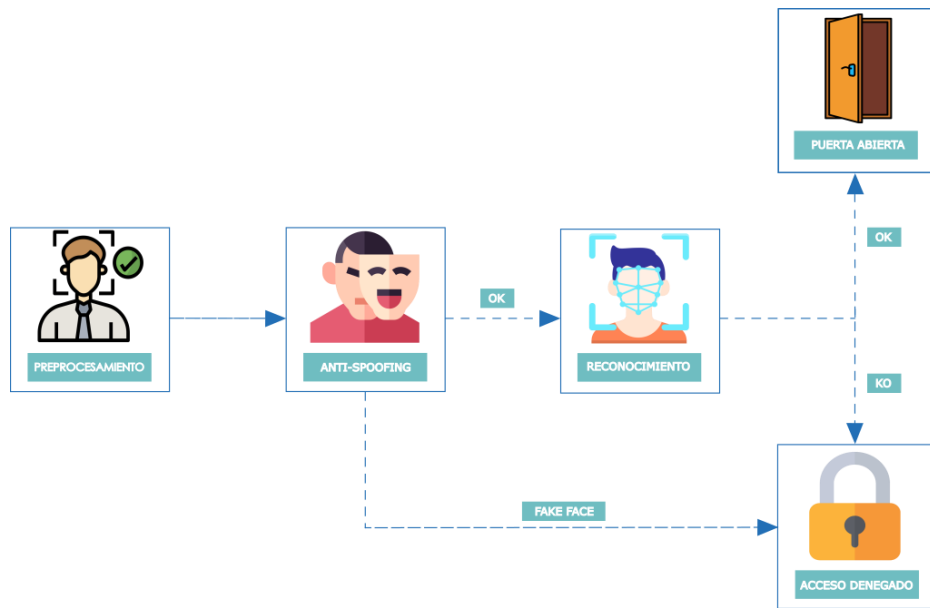
### 5.2.3 Diseño de la prueba de concepto

En relación con el diseño de la prueba de concepto, se sugirió basarse en el diseño utilizado para las pruebas de detección facial, con algunas modificaciones. La descripción de los pasos a seguir para la demo de la prueba de concepto se describe en la figura 5.17:



*Figura 5.18: Flujo prueba de concepto*

Tal como se mencionó previamente, los primeros pasos de la demo son idénticos a los realizados para la prueba de detección facial. En este caso, el usuario se coloca frente al videoportero y toca uno de los botones disponibles, activando así la primera vista. Con respecto al diseño de la vista, esta se compone de tres partes: el título de la vista, la sección donde se muestran los fotogramas capturados desde la cámara y un pequeño icono de la silueta de una cara. Los usuarios deben acercarse al videoportero de manera que su cara se corresponda con el icono de la silueta. Esta acción activa el proceso de reconocimiento facial, procesando la cara del usuario y deliberando si puede o no acceder a la vivienda. Del mismo modo, antes de llevar a cabo este proceso de reconocimiento y, contando con que el usuario no se encuentra registrado en el sistema, el paso previo al reconocimiento sería la inscripción del usuario. Para ello, una vez nos encontremos en la vista de reconocimiento, el usuario podrá pasar a la vista creada para la inscripción pulsando un botón concreto. En esta vista, el usuario deberá tener una expresión neutra en su cara acercándose lo más posible a la cámara del videoportero. Una vez completado este paso, simplemente tendrá que pulsar un botón y el videoportero empezará a procesar la imagen y almacenar su biometría facial. Una vez registrado en el sistema el usuario podrá volver a la vista del reconocimiento y probar si ya puede acceder a la vivienda.



*Figura 5.19: Proceso de reconocimiento*

En el momento que el usuario activa la vista de reconocimiento facial, internamente comienza el proceso de verificación de identidad. La figura 5.18 muestra el flujo de este proceso. Al activarse la vista, da comienzo la etapa de preprocesamiento en la que se realiza el proceso de detección facial para uno de cada tres fotogramas capturados por la cámara. Esta etapa no da comienzo a la siguiente fase del proceso de verificación de identidad hasta que no se detecte una cara que cumpla con ciertas restricciones. En definitiva, para permitir que la cara detectada pase a la siguiente fase del proceso, esta debe estar correctamente situada con respecto al icono de la silueta que hemos comentado en el párrafo anterior. La siguiente fase del proceso evalúa si existe un intento de suplantación de identidad con respecto a la cara que se quiere procesar. Es decir, en esta fase se comprueba si la cara detectada es una cara real o una cara falsa que pretende suplantar la identidad de un inquilino de la vivienda (puede ser una foto impresa, un video o una máscara 3D). Si se detecta que es una cara falsa, se prohíbe el acceso a la vivienda; en el caso contrario, se evalúa la cara detectada a través del proceso de inferencia con el modelo de reconocimiento facial. En esta etapa se adquiere la biometría facial del usuario que quiere acceder a la vivienda y se compara con las biometrías de los usuarios registrados. Si existe una biometría igual en la base de datos, se permite acceder a la vivienda; en caso contrario, se deniega el acceso.

Con los conceptos claros sobre la finalidad y el diseño del sistema de verificación de identidad que se pretende desarrollar, en las próximas secciones se describirán con mayor detalle todas y cada una de las fases que componen dicho sistema.

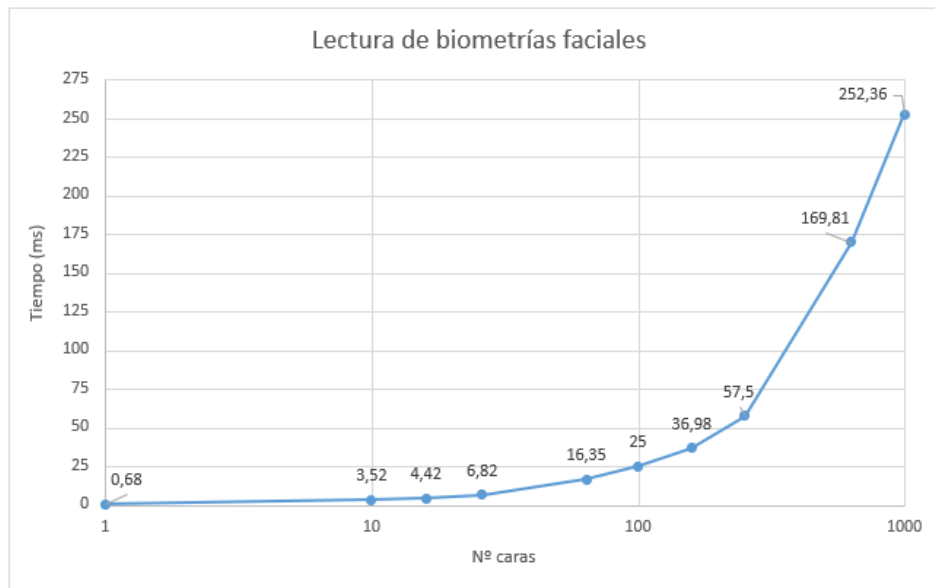
#### **5.2.4 Registro del usuario**

En esta etapa del sistema de verificación de identidad se realiza el proceso para la obtención de la biometría facial del usuario a raíz de una imagen de este. La biometría o característica de la cara se define como patrones faciales únicos para identificar a una persona. Estos patrones se extraen a través de características distintivas como la distancia entre los ojos, los contornos de la

boca, la forma de la nariz, etc. Para extraer de manera más precisa estos patrones, se suele recomendar que el usuario presente una expresión facial neutra en la imagen empleada para la captura de datos faciales. A la hora de implementar el proceso del registro del usuario empleando la biblioteca OpenCV, se definen los siguientes pasos:

- **Detección facial:** A partir de la imagen del usuario, se detecta y obtiene la posición de su cara. Los datos que se obtienen de la posición son: las coordenadas x e y correspondientes a la esquina superior izquierda del cuadro delimitador (el mismo cuadro que se usa para pintar las caras detectadas), junto con las dimensiones de ancho y alto de la cara.
- **Alineación de la cara:** este proceso se emplea para calibrar los puntos de referencia faciales con las plantillas frontales predefinidas. Es decir, si la cara detectada presenta algún tipo de desviación en la pose, se aplican transformaciones 2D para centrar los puntos de referencia permitiendo que el clasificador sea más discriminativo. En el caso que la cara detectada presente una postura correcta, en esta fase solo se recortará la cara detectada de la imagen original, creando una nueva imagen con una dimensión de 128x128.
- **Extracción de la biometría facial:** En esta etapa se realiza el proceso de inferencia, en el que se emplea la imagen recortada de la cara detectada y se obtiene como resultado un vector de 128 números decimales. Este vector representa la biometría facial de la cara detectada, y es lo que se utilizará para realizar la comparación y deducir si dos biometrías pertenecen o no a la misma cara.
- **Almacenamiento de la biometría en la BBDD:** Tras el proceso de extracción de la biometría facial del usuario, el último paso en el proceso del registro es almacenar dicha biometría en la base de datos pertinente. En el caso de la prueba de concepto, la biometría facial se almacena en un fichero de texto, el cual, una vez que el videoportero se enciende, el sistema lee y almacena en memoria.

Actualmente, el videoportero es el encargado de realizar el proceso de registro a través de la vista creada para ello. Por lo tanto, debido a las limitaciones hardware de nuestro sistema, se ha calculado que en promedio el tiempo total para extraer y almacenar la biometría facial de una imagen de 480x640, es de **5 a 6 segundos**. A priori, este tiempo no supone un problema en lo absoluto, ya que el registro es un proceso previo al reconocimiento facial, el cual, sí debe de ejecutarse con la mayor rapidez posible.



*Figura 5.20: Grafica lectura de biometrías*

Por otro lado, otro aspecto a destacar de este proceso es que cada uno de los ficheros donde se almacenan los datos biométricos ocupan solo **512 bytes**. Debido a esta consideración, los datos biométricos se almacenaron en la propia placa para la prueba de concepto, dado que no representaban ningún inconveniente en términos de espacio. De hecho, en la figura 5.19 podemos observar una gráfica donde se expone el coste de tiempo que tarda el sistema en leer y almacenar en memoria ‘n’ número de caras. Podemos observar que, para leer y almacenar 999 caras, el coste total es de **252 ms** de media, concluyendo que la lectura de estas biometrías no supone un coste significativo al sistema.

### 5.2.5 Suplantación de identidad

En el sistema para la verificación de identidad que se ha plantado, es crucial tener contramedidas para prevenir que personas externas traten de acceder a la vivienda suplantando la identidad de un residente. El conjunto de técnicas y tecnologías diseñadas para prevenir o detectar estos intentos de suplantación de identidad en sistemas biométricos se conoce por el término inglés *anti-spoofing*. Actualmente, los principales ataques que se realizan para la suplantación de identidad en dichos sistemas se dividen en dos grupos:

- **Ataques 2D:** Los atacantes utilizan fotografías impresas o videos de la persona a la que pretenden suplantar la identidad.
- **Ataques 3D:** En esta ocasión, los atacantes emplean máscaras 3D de diferentes materiales (resina, papel, ...) para tratar de reproducir la cara de la víctima.

Actualmente, los modelos de aprendizaje profundo han demostrado ser efectivos en el desempeño de esta tarea, ya que, al aprender patrones complejos y representaciones de diversas características, estos se vuelven bastante precisos en identificar señales de falsificación como



fotografías o videos. Cabe destacar que, en el caso de implementar sistemas de verificación de identidad en lugares críticos como oficinas o aeropuertos, el uso de estos modelos de aprendizaje profundo se suele complementar con todo tipo de sensores de profundidad, infrarrojos, etc. Esto se debe a que, a medida que se investigan más técnicas que sirvan para prevenir estas situaciones, los atacantes también desarrollan nuevos tipos de ataques. Por ello, el uso de sensores se emplea de cara a aumentar la seguridad del sistema, pero, como es evidente, aumentando el coste final del producto.

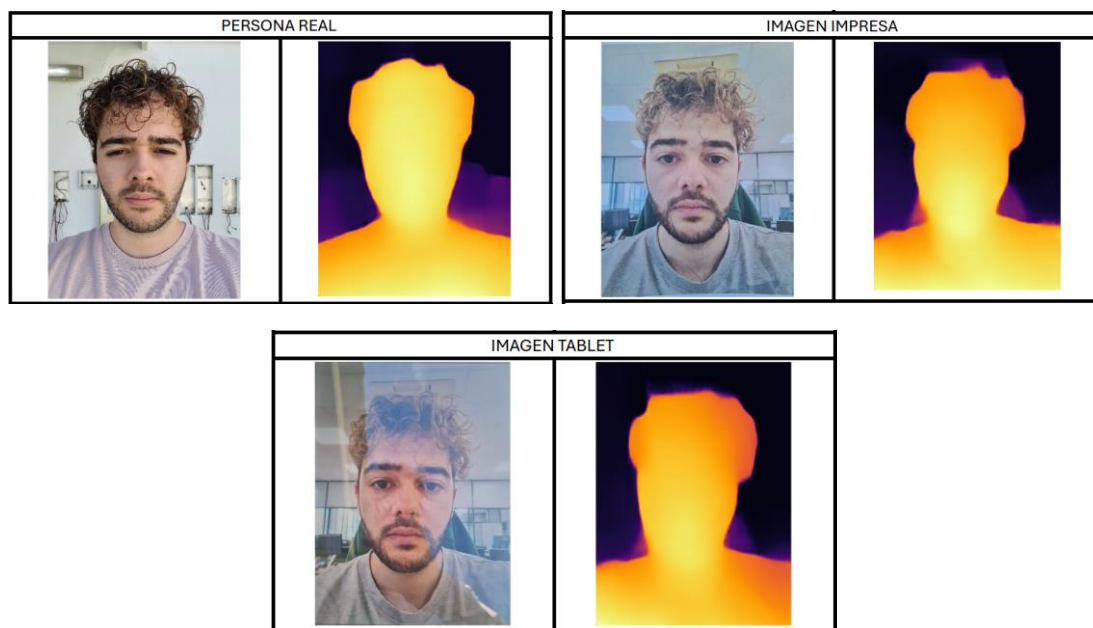
Tras un largo proceso de análisis, donde se investigó acerca de las técnicas para prevenir la suplantación de identidad que podríamos implementar en nuestro sistema, se afrontaron diversos inconvenientes. Principalmente, se comprobó que la mayoría de los modelos de aprendizaje profundo entrenados para cumplir con dicho objetivo, no eran de código abierto. Este era uno de los principales objetivos de nuestro proyecto (RNF-2), por lo que, al no encontrar un modelo pre entrenado de código abierto, se tendría que recurrir a analizar técnicas alternativas.

- **Técnicas alternativas para prevenir la suplantación de identidad**

La elección de técnicas alternativas para prevenir la suplantación de identidad es compleja debido a la necesidad de que dichas alternativas sean efectivas contra los tres tipos de ataques más comunes: fotografías impresas, videos y máscaras. Además, considerando las limitaciones hardware de nuestro videoportero, es crucial que el tiempo de procesamiento y respuesta para determinar si estamos ante un ataque sea lo más rápido posible. A raíz de esto, se inició un proceso de análisis para evaluar posibles alternativas, de entre las que destacamos las siguientes:

- **Detección de parpadeo:** El propósito principal de esta medida es verificar la fiabilidad de un rostro a través del parpadeo de los ojos. La ventaja principal de esta técnica reside en su rápida implementación, puesto que existen una gran variedad de ejemplos y tutoriales en la propia documentación de OpenCV. Pero, desafortunadamente, las desventajas presentes en la misma pesan más que las ventajas. Por un lado, esta medida solo sería efectiva frente a los ataques con fotografías impresas, puesto que en los otros ataques principales se puede suplantar el parpadeo de los ojos. Por otro lado, para analizar el parpadeo de los ojos en la prueba de concepto de nuestro sistema, debemos recopilar información de varios fotogramas consecutivos, ya que el usuario puede que se quede mirando fijamente a la pantalla del videoportero hasta que le reconozca la cara. Esto, como es evidente, implica diversas complejidades como el aumento del tiempo de procesamiento y respuesta de la técnica.
- **Modelos de estimación de profundidad:** Otra de las alternativas que se analizó fue el uso de modelos de aprendizaje profundo para estimar la profundidad de una imagen 2D. Esta idea se fundamenta a través del mapa de profundidad que se obtiene como resultado del proceso de inferencia. El objetivo consistía en determinar, utilizando la información contenida en dicho mapa, si el atacante está empleando una fotografía impresa o un dispositivo para reproducir videos. En la figura 5.20 se muestran los resultados de las pruebas que se realizaron (en el ordenador de sobremesa) para comprobar si esta medida pudiera ser una opción válida. Como se puede observar, el modelo de estimación de profundidad empleado ([MiDaS](#)) ha conseguido determinar correctamente la profundidad para todos los posibles ataques. Desafortunadamente, esta idea se acabó descartando debido a que, por regla general, los modelos que se emplean para esta tarea suelen ocupar bastante espacio (entre 100 MB a 1 GB). Por lo que, aunque la técnica funcione

correctamente, el espacio que ocupa el modelo supondría un problema dada nuestra limitación de memoria (500 MB).



*Figura 5.21: Análisis del modelo de profundidad – MiDaS dpt\_levit\_224*

- **Reconocimiento de voz:** Una de las alternativas propuestas por el tutor de la empresa era emplear el reconocimiento de voz como medida preventiva. En esta ocasión se emplearía la voz del usuario para determinar que no es un intento de suplantación. Esta alternativa era la más ambiciosa de todas, ya que estaríamos entrando en un nuevo campo ajeno al de la visión por computador. Del mismo modo, la idea fue descartada rápidamente. Esto fue debido a la complejidad inherente de esta medida, ya que, para su correcto uso, debíamos lidiar con los problemas de suplantación de audios, problemas con la nitidez, ruidos de fondo, etc. Además, actualmente existen muy pocas bibliotecas que permitan implementar, de manera rápida y eficiente, funcionalidades de lingüística computacional en C++ (se destaca la biblioteca en Python [Hugging Face](#) como la más utilizada para estas tareas).

La escasez de alternativas que permitieran cumplir con los objetivos del proyecto hizo que, por un momento, se replanteasen los requisitos de este. De hecho, al no encontrar otras opciones, se estuvo barajando la posibilidad de comprar un modelo pre entrenado a una empresa de terceros. Afortunadamente y, contra todo pronóstico, se consiguió encontrar un [modelo](#) de código abierto para cumplir con la tarea propuesta. Inicialmente, dicho modelo cumplía con la mayoría de los requisitos: su peso es extremadamente ligero ocupando solo 12 MB, la licencia del modelo es de código abierto y de libre comercialización y, además, el modelo presenta un formato compatible para su integración con el módulo DNN. Por lo que, una vez se encontró un buen candidato para resolver la tarea de la suplantación de identidad, se comenzó a preparar las pruebas pertinentes para evaluar si cumplía con el resto de los requisitos.

- **Pruebas y resultados**

Una vez se examinó la estructura del modelo, se procedió a diseñar las pruebas para evaluar su eficacia y tiempo de procesamiento. Una vez aplicado el proceso de inferencia, el modelo devuelve 2 números decimales correspondientes con la probabilidad de ser o no cara real. Para entender correctamente el significado de estos valores, es más sencillo representarlos por porcentajes. Así, por ejemplo, para una cara ‘x’ el resultado podría ser: 85% de probabilidades de ser una cara real y 15% de ser una cara falsa. Cabe destacar que, como se aprecia en el ejemplo, la suma de las probabilidades nunca supera el 100%, por lo tanto, estos valores siempre son una diferencia de 100.





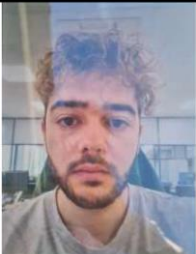

PERSONA REAL		IMAGEN IMPRESA	
Real face: 99,3948%	Real face: 99,5149%	Real face: 0,5386 %	Real face: 0,3053 %
Spoofed face: 0,6052%	Spoofed face: 0,4851%	Spoofed face: 99,4614 %	Spoofed face: 99,6947 %
			

IMAGEN TABLET	
Real face: 0,1901 %	Real face: 0,3964 %
Spoofed face: 99,8099 %	Spoofed face: 99,6036 %
	

**Figura 5.21:** Pruebas de eficacia con el modelo *anti-spoofing*

Para medir la eficacia del modelo, se replicaron diversas imágenes que simulaban distintos tipos de ataques para la suplantación del rostro. En la figura 5.21 podemos ver la distribución de dichas imágenes. Por un lado, nos encontramos con las imágenes que no representan ningún tipo de ataque, es decir, caras reales que intentan acceder al sistema. Mientras que, en el resto de los casos, se representan imágenes que simulan distintos tipos de ataques (imágenes impresas o imágenes desde un móvil/Tablet). Por otro lado, es relevante mencionar que las imágenes en las que aparece el rostro de una mujer se utilizaron para verificar los resultados cuando se aprecia el fondo de la escena. Finalmente, si nos fijamos en los resultados de la prueba, podemos concluir que el modelo consigue distinguir a la perfección cuando se trata de un ataque y cuando no. Además, el tiempo promedio del proceso de inferencia de los modelos es de aproximadamente **220 ms**, lo cual satisface con creces los requisitos establecidos.

### 5.2.6 Tareas de reconocimiento facial

La etapa del reconocimiento facial es el pilar que sustenta al sistema de verificación de identidad. El objetivo principal de esta etapa se basa en identificar la identidad de una persona a través de las características inherentes de su rostro. Para ello, se extrae la biometría facial del usuario que trata de acceder a la vivienda. Una vez extraída, el proceso habitual consiste en comparar la biométrica extraída con todas las demás biometrías almacenadas en la base de datos (recopiladas en la etapa de registro de usuario). A la hora de implementar la comparación entre las biometrías, OpenCV ofrece dos posibles técnicas:

- **FR\_COSINE**: Mide el coseno del ángulo entre los dos vectores
- **FR\_NORM\_L2**: Realiza el cálculo de la distancia euclídea entre los dos vectores

La elección de qué técnica utilizar es totalmente dependiente del usuario, pero, normalmente se suelen emplear ambas para aumentar la fiabilidad del resultado. El resultado del reconocimiento será un número decimal (llamémosle 'x') obtenido tras la comparación de las biometrías faciales (1:1). Para determinar si ambas biometrías pertenecen o no a la misma persona, el resultado debe cumplir con una condición específica, dependiendo de la técnica empleada:

- **Es la misma persona**  $\rightarrow x \geq 0.363$  (FR\_COSINE) y/o  $x \leq 1.128$  (FR\_NORM\_L2)
- **NO es la misma persona**  $\rightarrow x < 0.363$  (FR\_COSINE) y/o  $x > 1.128$  (FR\_NORM\_L2)

El equipo de OpenCV ha determinado el valor de los umbrales (0.363 y 1.128) en función del rendimiento del modelo de reconocimiento facial en un conjunto de datos de validación. Con toda esta información presente y, una vez implementado el proceso de reconocimiento facial en la demo, se procedió a evaluar la velocidad y eficacia del modelo SFace.

#### • Pruebas de velocidad

Podemos dividir las pruebas para evaluar la velocidad del modelo SFace en dos partes. En primer lugar, se tomaron 80 muestras del coste de tiempo del proceso de reconocimiento facial al completo (proceso de inferencia + comparación de las biometrías). Los resultados de estas pruebas se muestran en la siguiente tabla:

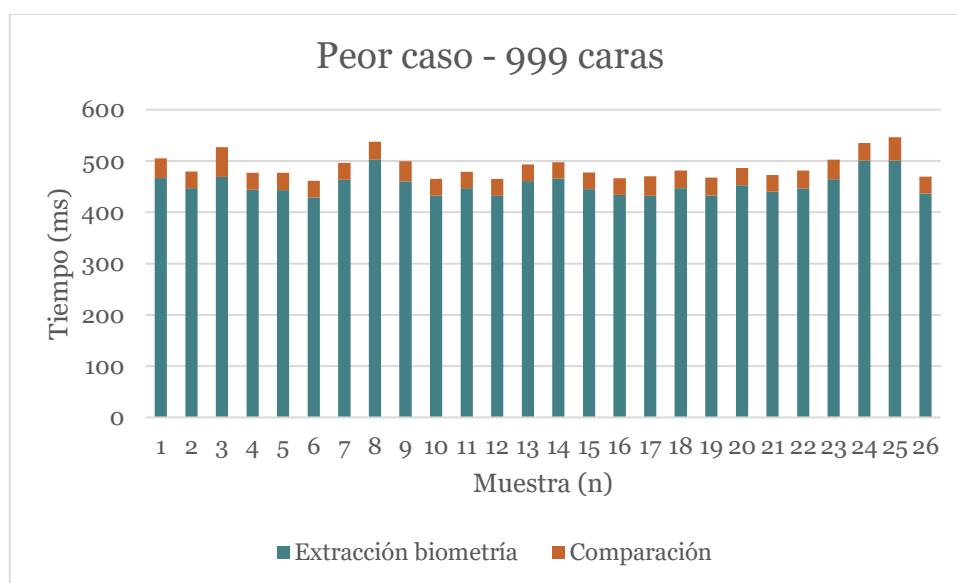
Valor mínimo	Valor máximo	Valor promedio
424,94 ms	642,25 ms	475,72 ms

*Tabla 5.7: Resultados de la prueba de velocidad para el reconocimiento facial*

Tal como se puede apreciar, la tabla presenta los valores mínimo, máximo y promedio de la muestra. Cabe destacar que el valor promedio analizado (477 ms) es considerablemente menor al valor promedio mencionado al comienzo de este capítulo, el cual se obtuvo tras realizar las primeras pruebas (650 ms). Esta diferencia surgió principalmente por la falta de optimización inicial del proceso de verificación de identidad. Es decir, estas pruebas finales se realizaron bajo la revisión de que el proceso de verificación de identidad estuviera lo más optimizado posible.

Por ejemplo, al inicio se realizaba el recorte de la cara detectada en dos ocasiones; una previa al proceso del modelo de suplantación de identidad y otra previa al modelo de reconocimiento. Como es evidente, realizar dos veces este proceso es redundante por lo que se modificó y, por ende, se consiguió mejorar los tiempos del proceso.

Por otro lado, la segunda prueba de velocidad que se diseñó buscaba evaluar el coste de tiempo del proceso de comparación de biometrías con respecto al tiempo total del reconocimiento, considerando el peor caso de todos. Es decir, se planteó la situación donde hubieran almacenadas 999 caras de un edificio y, la cara del usuario que pretende entrar a su vivienda estuviera almacenada la última. Dado el diseño de la fase de comparación, que implica un proceso lineal de recorrido del vector de biometrías, se espera que este sea el caso en el que, a priori, el costo de tiempo de la fase de comparación sea mayor.



*Figura 5.22: Prueba de velocidad para el peor caso*

Los resultados de la prueba se exponen en la gráfica de la figura 5.21. En dicha gráfica, podemos observar cómo el proceso de extracción de biometría supone el mayor coste de tiempo en la etapa del reconocimiento. Mientras que, el proceso de comparación en el peor caso solo representa un 8% del tiempo de ejecución total. De hecho, también se llega a la conclusión de que, en un escenario normal, el tiempo de comparación es prácticamente insignificante, ya que en el peor caso solo representa unos 35 ms en promedio.

- **Pruebas de eficacia**

Para las pruebas de eficacia del reconocimiento se realizó un diseño similar al empleado en las pruebas para la detección facial. En este caso, se evaluaría la eficacia del modelo ante diversas situaciones que pudieran perjudicar su resultado. Dada la enorme diversidad de situaciones que se pueden dar en un caso real del uso del videoportero, se procuró realizar una selección de aquellas que pudieran tener un mayor impacto negativo. Esta selección de situaciones se agrupó

en: diversidad de expresiones faciales, uso de accesorios faciales, variedad de poses de la cara y, finalmente, características personales como color de piel, densidad del cabello, etc. Además, en esta ocasión, la evaluación de dichas situaciones se pretendía aplicar bajo condiciones lumínicas adversas, como, por ejemplo, con el sol de cara al videoportero. Para llevar a cabo la prueba, la empresa proporcionó todo el material necesario, ofreciendo una terraza y un panel, con el que poder desplazar el videoportero, para cumplir con las condiciones lumínicas establecidas. Es importante mencionar que, la eficacia del reconocimiento facial depende en gran medida de la correcta extracción de la biometría en el momento del registro del usuario. Por ejemplo, la posible diferencia de luminosidad entre la biometría del registro y la biometría del intento de acceso a la vivienda, podría ser un factor que perjudique el resultado del reconocimiento. Por ello, para evaluar si estos casos son realmente un problema para el reconocimiento facial, se propuso realizar el mismo conjunto de pruebas en dos partes: uno realizando el registro del usuario en el momento y otro realizando el registro previamente en un entorno favorable.

Condiciones lumínicas	Expresiones faciales				Posición de la cara			Accesorios faciales				Características personales							
	Cara neutra	Sonriendo (boca cerrada)	Sonriendo (boca abierta)	Ojos cerrados	Cara recta	Cara inclinada 45° lado	Cara inclinada 45° boca	Gafas de vista	Gafas de sol	Bragas / buff	Gorra	Pelo corto	Pelo largo	Barba	Imberbe	Calvo	Blanco	Moreno	Negro
Luz de lado	100%	100%	90%	100%	100%	70%	80%	100%	-	-	-	100%	100%	100%	100%	100%	100%	-	-
Luz de cara	100%	100%	90%	100%	100%	70%	80%	100%	-	-	-	100%	100%	100%	100%	100%	100%	-	-
Luz de espaldas	60%	50%	40%	40%	60%	40%	40%	60%	-	-	-	60%	60%	60%	60%	50%	60%	-	-
Ocureidad	100%	100%	90%	100%	100%	80%	80%	100%	-	-	-	100%	100%	100%	100%	100%	100%	-	-

Figura 5.23: Resultado de la prueba de eficacia – Registro de usuario en el momento

Los resultados de las pruebas se exponen en las figuras 5.22 y 5.23. Estas tablas evalúan porcentualmente la eficacia que se ha logrado obtener con el modelo en una situación específica, con una condición lumínica concreta. Es decir, durante la creación de la tabla, se registró el número de ocasiones en las que el modelo identificó correctamente al usuario que trataba de acceder a la vivienda, de un total de 20 intentos. Por lo tanto, el porcentaje que aparece en cada casilla viene dado por la relación entre el número de identificaciones correctas y el número de intentos totales. Antes de pasar a hablar sobre los resultados cabe mencionar que algunas de las situaciones están pendientes de evaluar (aquellas en las que aparece un guion). Del mismo modo, no solo quedan por evaluar situaciones que se encuentran en la tabla, sino que también falta por aplicar las pruebas a personas de distintas etnias. Estas pruebas se han retrasado debido a los problemas logísticos que suponen, aunque se pretende llevarlas a cabo con la mayor brevedad posible.

Condiciones lumínicas	Expresiones faciales				Posición de la cara			Accesorios faciales				Características personales							
	Cara neutra	Sonriendo (boca cerrada)	Sonriendo (boca abierta)	Ojos cerrados	Cara recta	Cara inclinada 45° lado	Cara inclinada 45° boca	Gafas de vista	Gafas de sol	Bragas / buff	Gorra	Pelo corto	Pelo largo	Barba	Imberbe	Calvo	Blanco	Moreno	Negro
Luz de lado	100%	100%	90%	100%	100%	80%	90%	100%	-	-	-	100%	100%	100%	100%	100%	100%	-	-
Luz de cara	100%	100%	80%	90%	100%	80%	80%	100%	-	-	-	100%	100%	100%	100%	100%	100%	-	-
Luz de espaldas	60%	50%	30%	50%	60%	40%	30%	60%	-	-	-	60%	60%	60%	60%	50%	60%	-	-
Ocureidad	100%	90%	80%	100%	100%	90%	90%	100%	-	-	-	100%	100%	100%	100%	100%	100%	-	-

Figura 5.24: Resultado de la prueba de eficacia – Registro de usuario previo

Como conclusión, si observamos ambas tablas nos damos cuenta de que el rendimiento del modelo de reconocimiento facial en la mayoría de los casos realiza una actuación más que adecuada. Pero, en el caso de encontrar la luz de espaldas, comienza a disminuir la eficacia del modelo sin importar la forma en la que hacemos el registro de usuarios. Estos resultados nos han

llevado a pensar que el modelo tiende a rendir peor cuando la cara del usuario se encuentra a la sombra. Actualmente, no se ha encontrado una solución para este problema, pero se sigue investigando para encontrarla lo antes posible. Por ahora, la opción más factible que se ha planteado es realizar un ajuste de la intensidad lumínica de la imagen previo al reconocimiento facial. En dicho proceso, aplicaríamos una media de intensidad lumínica al fotograma, oscureciendo los puntos de luz más brillantes e iluminando los puntos más oscuros.



## Conclusiones

---

En este proyecto se ha abordado la tarea de analizar la viabilidad de añadir funcionalidades relacionadas con la detección y el reconocimiento facial en un videoportero comercial. Como se mencionó en capítulos anteriores, esta tarea presentaba una gran dificultad adicional debido a los recursos limitados del sistema embebido empleado. Estas limitaciones no solo afectaban al número de modelos pre entrenados para la detección y el reconocimiento facial a los que podíamos acceder, sino que, representaron todo un desafío a la hora de integrar las bibliotecas o marcos de trabajo para el campo de la visión por computador. Pese a todos los desafíos afrontados, podemos concluir que, no solo es viable la integración de estas nuevas funcionalidades, sino que los resultados tan satisfactorios en las pruebas sugieren que el proyecto es incluso viable para integrar a futuro en los diversos productos comerciales de la empresa.

El trabajo constante, junto con una buena organización y gestión de los cambios en los objetivos del proyecto, han supuesto el pilar fundamental para obtener los resultados tan satisfactorios que se han conseguido. No solo se han cumplido con todos los requisitos y objetivos impuestos en un inicio del proyecto, sino que, se han logrado cumplir los nuevos requisitos asociados al sistema de verificación de identidad. Esta prueba de concepto es una sólida evidencia del esfuerzo y la ilusión que se ha invertido en este proyecto. Sin lugar a duda, conseguir ejecutar todo el proceso de verificación de identidad en menos de un segundo ha superado con creces las expectativas, tanto de mis compañeros de trabajo del equipo I+D como de mi tutor de prácticas de la empresa o, incluso, las de nuestros *stakeholders* de la organización (jefes de proyecto '*PM*' y el departamento de ventas de la empresa).

Este hecho supone un impacto tremendamente positivo para la empresa, ya que ha obtenido nuevos servicios para ofrecer a sus usuarios con el único coste de las horas invertidas en el proceso del proyecto, es decir, sin sobrecostes de software de terceros. Del mismo modo, el éxito de este proyecto supone para mí una gran satisfacción al ver como he sido parte fundamental de los cimientos de un proyecto de tal envergadura. Cabe mencionar que en el transcurso de este proyecto he puesto en práctica no solo los conocimientos adquiridos durante el transcurso del Máster de Ingeniería Informática, sino que dicho proyecto me ha impulsado a adquirir nuevos conocimientos al utilizar nuevas herramientas, metodologías e investigar nuevas áreas de la informática como la visión por computador. Por si fuera poco, al llevar a cabo el trabajo en un entorno cooperativo con los miembros del equipo de I+D de la empresa, se han puesto a prueba mis habilidades comunicativas y de trabajo en equipo. De igual manera, es relevante señalar que sin el buen ambiente del equipo y la disposición de todos para aprender de los demás, este trabajo hubiera supuesto aún más sacrificio y esfuerzo de lo que realmente ha sido.



## 6.1 Trabajo futuro

---

Una vez expuestas las conclusiones del proyecto, se procede a detallar las acciones que deben llevarse a cabo a partir de este momento. Aunque el proyecto haya sido un éxito y se hayan cumplido con todos los requisitos de este, aún queda mucho trabajo para incluir y comercializar las funciones de detección y reconocimiento facial en los videoporteros de la empresa Fermax. De igual manera, dicho trabajo puede dividirse según las dependencias temporales de las tareas: corto o largo plazo. Por lo tanto, las tareas principales que deben llevarse a cabo a partir de este momento son:

- Realizar las pruebas restantes para evaluar la eficacia del modelo de reconocimiento facial (prueba con diferentes etnias, más diversidad de entornos lumínicos, etc.)
- Realizar pruebas empleando imágenes con mejor resolución para evaluar la eficacia del modelo de suplantación de identidad
- Selección del medio definitivo para realizar el registro de usuario (aplicación móvil, videoportero, etc.)
- Implementación e integración del proceso de registro de usuario en el medio seleccionado
- Análisis de la viabilidad de almacenar más de una biometría facial por usuario. Este caso se investigará para lidiar, por ejemplo, con los casos en los que el usuario lleve gafas de vista.
- Crear base de datos robusta y segura para almacenar las biometrías faciales.
- Solucionar la incidencia del problema de precisión del reconocimiento facial ante la situación de tener la luz natural incidiendo desde nuestra espalda
- Estudio e implementación de nuevas funcionalidades a raíz de la integración de la biblioteca de visión por computador



# Bibliografía

---

- Ahonen, T., Hadid, A., & Pietikainen, M. (2006). Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2037-2041. doi:10.1109/TPAMI.2006.244
- Birhane, A. (2022). The unseen Black faces of AI algorithms. *Nature*, 451-452. doi:10.1038/d41586-022-03050-7
- Buolamwini, J., & Gebru, T. (2018). Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, 77-91. Obtenido de <https://proceedings.mlr.press/v81/buolamwini18a/buolamwini18a.pdf>
- Cao, Q., Shen, L., Xie, W., Parkhi, O. M., & Zisserman, A. (2017). VGGFace2: A dataset for recognising faces across pose and age. doi:10.48550/ARXIV.1710.08092
- Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (págs. 886-893). San Diego, CA, USA: IEEE. doi:10.1109/CVPR.2005.177
- Demush, R. (26 de Febrero de 2019). A Brief History of Computer Vision (and Convolutional Neural Networks). *Hackernoon*. Obtenido de <https://hackernoon.com/a-brief-history-of-computer-vision-and-convolutional-neural-networks-8fe8aacc79f3>
- Everingham, M., Eslami, S. M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2015). The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, 98-136. doi:10.1007/s11263-014-0733-5
- Feng, Y., Yu, S., Peng, H., Li, Y.-R., & Zhang, J. (2022). Detect Faces Efficiently: A Survey and Evaluations. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 3-6. Obtenido de <https://ieeexplore.ieee.org/document/9580485/>
- Guo, Y., Zhang, L., Hu, Y., & He, X. G. (2016). MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition. *arXiv (Cornell University)*. doi:10.48550/ARXIV.1607.08221
- Han, Y., Huang, G., Song, S., Yang, L., Wang, H., & Wang, Y. (2021). Dynamic Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7436-7456. Obtenido de <https://arxiv.org/abs/2102.04906>
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *Computer Vision and Pattern Recognition Conference (CVPR)* (págs. 770-778). Las Vegas: IEEE. doi:10.48550/ARXIV.1512.03385
- Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., & Weyand, T. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv (Cornell University)*. doi:10.48550/ARXIV.1704.04861

- Huang, G. B., Ramesh, M., Berg, T., & Learned-Miller, E. (2007). Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. *University of Massachusetts Amherst*. Obtenido de <https://vis-www.cs.umass.edu/lfw/>
- Hubel, D. H., & Wiesel, T. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 574-591. doi:10.1113/jphysiol.1959.sp006308
- IBM. (7 de Mayo de 2021). *What is Computer Vision?* Obtenido de <https://www.ibm.com/topics/computer-vision>
- Jain, V., & Learned-Miller, E. (2021). Fddb: A Benchmark for Face Detection in Unconstrained Settings. *University of Massachusetts Amherst*. Obtenido de <https://people.cs.umass.edu/~elm/papers/fddb.pdf>
- Kemelmacher-Shlizerman, I., Seitz, S., Miller, D., & Brossard, E. (2015). The MegaFace Benchmark: 1 Million Faces for Recognition at Scale. *Computer Vision and Pattern Recognition Conference (CVPR)* (págs. 4873-4882). Las Vegas: IEEE. doi:10.48550/ARXIV.1512.00596
- Liu, Y., Stehouwer, J., & Liu, X. (2020). On Disentangling Spoof Trace for Generic Face Anti-Spoofing. doi:10.48550/ARXIV.2007.09273
- Papert, S. (1966). *The Summer Vision Project*. Massachusetts Institute of Technology. Obtenido de <https://dspace.mit.edu/handle/1721.1/6125>
- Roberts, L. (1963). *Machine Perception of Three-Dimensional Solids*. Massachusetts Institute of Technology. Obtenido de <https://dspace.mit.edu/handle/1721.1/11589>
- Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A Unified Embedding for Face Recognition and Clustering. doi:10.48550/ARXIV.1503.03832
- Sethi, S., Kathuria, M., & Kaushik, T. (2021). Face mask detection using deep learning: An approach to reduce risk of. *Journal of Biomedical Informatics*. doi:<https://doi.org/10.1016/j.jbi.2021.103848>
- Solomon, E., & Cios, K. J. (2023). FASS: Face Anti-Spoofing System Using Image Quality Features and Deep Learning. *Electronics*, 2199. doi:10.3390/electronics12102199
- Taigman, Y., Yang, M., Ranzato, M., & Wolf, L. (2014). DeepFace: Closing the Gap to Human-Level Performance in Face Verification. *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (págs. 1701-1708). Columbus, OH, USA: IEEE. doi:10.1109/CVPR.2014.220
- Turk, M., & Pentland, A. (1991). Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*, 71-86. doi:10.1162/jocn.1991.3.1.71
- Viola, P., & Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001)*. doi:10.1109/CVPR.2001.990517
- Wang, M., & Deng, W. (2021). Deep Face Recognition: A Survey. *Neurocomputing*, 215-244. doi:10.1016/j.neucom.2020.10.081

- Wang, X., Peng, J., ZHANG, S., Chen, B., Wang, Y., & Guo, Y. (2022). A Survey of Face Recognition. Obtenido de <http://arxiv.org/abs/2212.13038>
- Wu, W., Peng, H., & Yu, S. (2023). YuNet: A Tiny Millisecond-level Face Detector. *Machine Intelligence Research*, 656-665. doi:10.1007/s11633-023-1423-y
- Xiangxin, Z., & Ramanan, D. (2012). Face detection, pose estimation, and landmark localization in the wild. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, (págs. 2879-2886). Providence, RI. doi:10.1109/CVPR.2012.6248014
- Yan, J., Zhang, X., Lei, Z., & Li, S. Z. (2014). Face detection by structural models. *Image and Vision Computing*, 790-799. Obtenido de <https://linkinghub.elsevier.com/retrieve/pii/S0262885613001765>
- Yang, S., Luo, P., Loy, C. C., & Tang, X. (2015). WIDER FACE: A Face Detection Benchmark. *Chinese University of Hong Kong*, 1-12. doi:10.48550/ARXIV.1511.06523
- Yuantao, F., Shiqi, Y., Hanyang, P., Yan-Ran, L., & Jianguo, Z. (2022). Detect Faces Efficiently: A Survey and Evaluations. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 1-18. doi:10.1109/TBIOM.2021.3120412



## ANEXO

### OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. <b>Fin de la pobreza.</b>				X
ODS 2. <b>Hambre cero.</b>				X
ODS 3. <b>Salud y bienestar.</b>				X
ODS 4. <b>Educación de calidad.</b>		X		
ODS 5. <b>Igualdad de género.</b>				X
ODS 6. <b>Agua limpia y saneamiento.</b>				X
ODS 7. <b>Energía asequible y no contaminante.</b>			X	
ODS 8. <b>Trabajo decente y crecimiento económico.</b>				X
ODS 9. <b>Industria, innovación e infraestructuras.</b>	X			
ODS 10. <b>Reducción de las desigualdades.</b>	X			
ODS 11. <b>Ciudades y comunidades sostenibles.</b>				X
ODS 12. <b>Producción y consumo responsables.</b>				X
ODS 13. <b>Acción por el clima.</b>				X
ODS 14. <b>Vida submarina.</b>				X
ODS 15. <b>Vida de ecosistemas terrestres.</b>				X
ODS 16. <b>Paz, justicia e instituciones sólidas.</b>	X			
ODS 17. <b>Alianzas para lograr objetivos.</b>				X

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

Este TFM ha abordado el proyecto de investigación para analizar la viabilidad de incluir funcionalidades de detección y reconocimiento facial en un videoportero comercial con recursos hardware limitados. En la actualidad, el reconocimiento facial es una tecnología que sigue evolucionando y transformando nuestra interacción con la seguridad y la comodidad. Las recientes innovaciones en las áreas de inteligencia artificial (IA) han producido un aumento de las capacidades del reconocimiento facial, haciéndolo más rápido y preciso al emplear algoritmos y modelos basados en técnicas de aprendizaje profundo. En el caso de integrar dicha funcionalidad en un videoportero, no solo podría mejorar la seguridad al acceso del edificio, reduciendo el riesgo de intrusiones a personas no autorizadas, sino que podría eliminar la necesidad de tarjetas o códigos PIN para el acceso de la vivienda, mejorando la experiencia del usuario a la par que fortaleciendo la seguridad. Por lo tanto, la adopción de tecnologías como el reconocimiento facial promueven la consecución de objetivos de desarrollo sostenible como los números 9 (Industria, innovación e infraestructuras) y 11 (Ciudades y comunidades sostenibles), ya que, esta tecnología permite optimizar la gestión de la infraestructura del edificio, simplificando su acceso y mejorando su seguridad. Esto contribuye a crear comunidades más seguras y eficientes, las cuales pueden reducir la necesidad de emplear recursos físicos como tarjetas de acceso.

Del mismo modo, los sistemas de reconocimiento facial están cada vez más presentes en nuestra vida cotidiana, a menudo sin que seamos conscientes de ello. Este hecho ha generado cierta controversia, ya que han surgido ciertas preocupaciones sobre la privacidad y el uso indebido de los datos personales. En relación con el objetivo de desarrollo sostenible que aboga por el acceso a la justicia y a la protección de los derechos individuales (ODS 16 “Paz, Justicia e Instituciones Sólidas”), un videoportero que incluya funcionalidades relacionadas con el reconocimiento facial debe proporcionar información de manera transparente a los usuarios sobre qué datos se almacenarán, cómo se protegerán sus datos y quién tendrá acceso a dichos datos. Esta preocupación radica en el hecho de que ningún sistema de seguridad es infalible, y si una base de datos con biometrías faciales es pirateada, las consecuencias podrían ser muy graves (suplantación de identidad, extorsión, acoso, etc.). Por ello, es de vital importancia realizar una correcta gestión de los datos personales de los usuarios, además de transmitir confianza a raíz de las medidas de seguridad abordadas para proteger dichos datos. Esto nos lleva a destacar otro aspecto fundamental como es la educación de los usuarios (siguiendo el ODS 4 “Educación de calidad”). La conciencia



pública puede desempeñar un papel crucial en la mitigación de riesgos y, a su vez, en la promoción de un uso responsable de estas tecnologías. Así pues, como desarrolladores implicados es nuestro deber informar a los usuarios sobre aspectos como la información almacenada, cómo se recopila o para qué se emplea, ya que, con dicha información, los usuarios pueden tomar decisiones siendo totalmente autoconscientes y protegiendo su seguridad.

Otro factor para tener en cuenta con la integración de técnicas como el reconocimiento facial en videoporteros es que estos pueden ser inherentemente sesgados, especialmente si los modelos seleccionados han sido entrenados con datos no representativos. Esto podría llevar a decisiones erróneas, provocando la discriminación de algunos usuarios. Por ejemplo, si el conjunto de datos que se ha empleado para entrenar al modelo de reconocimiento facial contiene principalmente imágenes de personas de un grupo demográfico específico (p. ej. personas de piel clara), podría provocar dificultades a la hora de reconocer con precisión a personas de otros grupos (p. ej. personas de piel oscura). Las discriminaciones que se podrían producir al denegar el acceso a alguien legítimo supondrían no solo una pésima imagen para la empresa, sino una violación de los derechos fundamentales de los usuarios. Por lo tanto, con el propósito de cumplir con el objetivo de desarrollo sostenible que busca reducir las disparidades y garantizar la igualdad de oportunidades (ODS 10 “Reducción de las desigualdades”), se debe cerciorar de que los modelos empleados para el reconocimiento facial hayan sido entrenados con bases de datos inclusivas.

En conclusión, aunque la integración de funcionalidades como el reconocimiento facial en videoporteros ofrecen ventajas en aspectos como la gestión de la infraestructura o la seguridad, es crucial abordar otras preocupaciones para garantizar la privacidad y la equidad en su implementación.