Universidad de Valladolid

TRABAJO FIN DE MÁSTER

Máster en Ingeniería de Telecomunicación

Sistema biométrico multimodal con imágenes acústicas y LiDAR utilizando una red neuronal

Autor:

Andrés Martín Yeves

Tutores:

Alberto Izquierdo Fuente

Lara Del Val Puente

Valladolid, 25 de Septiembre de 2023

"Hagas lo que hagas, ámalo"
Giuseppe Tornatore 1988

Prólogo

El Trabajo Fin de Máster presentado a continuación se titula: "Sistema biométrico multimodal con imágenes acústicas y LiDAR utilizando una red neuronal".

Este trabajo representa un importante avance en la investigación que comencé en mi proyecto final de grado titulado "Parametrización y Clasificación de Imágenes Acústicas para Reconocimiento Biométrico". De hecho, este desarrollo se ha alimentado tanto del conocimiento adquirido durante mis prácticas como de las conclusiones y líneas futuras planteadas en mi TFG.

Si echo la vista hacia atrás, encuentro claramente un punto de inflexión en la elección del tema de mi TFG. Planteé varias posibilidades, pero fue hablando con mi tutor de TFG y ahora de mi TFM, Alberto Izquierdo con quién encontré en este proyecto el reto personal que buscaba. Es sin duda la primera persona a la que debo agradecer ya que si no fuese por la ilusión y confianza que me transmitió desde el primer momento, ninguno de estos trabajos existiría. No me quiero olvidar tampoco del resto del equipo del GPA. Gracias a Lara y a Juanjo por tratarme como uno más y prestarme siempre el apoyo necesario para sacar el proyecto adelante.

Quiero agradecer también a todas las personas que no he mencionado y se han prestado voluntarias de manera altruista para construir la base de datos de imágenes acústicas necesaria para el proyecto. Gracias a Pablo Carretero, Pablo Villacorta, Irene, Raquel y Noemi.

Por último, quería finalizar remarcando el hecho de que sacar adelante un proyecto no consiste únicamente en las personas que contribuyen directamente en él, sino también en todas las que, con su apoyo, te ayudan a avanzar y superar los obstáculos en el camino. Gracias a mi familia por entenderme, apoyarme y construir conmigo y no sobre mí las bases de mi desarrollo académico y, sobre todo, personal; a Victoria por preocuparse y hacer lo imposible para levantarme en mi momento más difícil; a mis amigos de mi pequeño pueblo de Soria, con quienes nunca existen preocupaciones; y por último, a mis amigos de Valladolid, en especial a Paula, Mario Laustalet y a mis amigos de Delicias, por hacer mucho más llevadero el camino desde el principio hasta aquí.

Andrés Martín Yeves

Valladolid, 18 de septiembre de 2023

Resumen

Este proyecto se inscribe dentro del campo de la biometría acústica formando parte de una investigación más amplia con escasa literatura existente.

A partir de imágenes acústicas 3D capturadas mediante un array fractal de sensores MEMS de alta resolución, junto con una cámara LiDAR diseñada para entornos interiores, este estudio presenta el hardware y la implementación de un software propio desarrollado en LabVIEW, destinados a la adquisición síncrona y procesamiento de imágenes resultantes de la fusión de ambos sistemas. Esta integración facilita la investigación de imágenes acústicas biométricas filtrando las imágenes espacialmente.

El propósito final es implementar un sistema biométrico de verificación. En este contexto, tras una reflexión acerca de múltiples alternativas, se programa en python una arquitectura particular de red neuronal convolucional siamesa.

Para cualquier aplicación de inteligencia artificial, resulta esencial disponer de un conjunto de datos amplio y de alta calidad. Con este propósito, se ha creado una base de datos compuesta por 21000 imágenes multifrecuenciales pertenecientes a 10 individuos. Finalmente haciendo uso de esta base de datos, se realizan proyecciones 2D destinadas a servir como entrada para la mencionada red neuronal.

Se concluye con la evaluación de los resultados, lo que ha conducido al descarte de la arquitectura inicialmente propuesta y la reflexión sobre posibles vías de mejora.

Palabras clave: Biometría acústica, array de micrófonos MEMS, LiDAR, Verificación, Redes siamesas, Redes neuronales convolucionales

Abstract

This project falls within the field of acoustic biometrics and is part of a larger research project with little existing literature.

Using 3D acoustic images captured with a fractal array of high-resolution MEMS sensors combined with a LiDAR camera designed for indoor environments, this study presents the hardware and the implementation of a proprietary software developed in LabVIEW for the synchronous acquisition and processing of images resulting from the fusion of both systems. This integration facilitates the research of biometric acoustic images by spatially filtering the images.

The final purpose is to implement a biometric verification system. In this context, after studying multiple alternatives, a particular Siamese convolutional neural network architecture has been programmed using Python.

For any artificial intelligence application, it is essential to have a large and high quality dataset. With this purpose, a database composed of 21000 multi-frequency images corresponding to 10 individuals has been created. Finally, using this database, 2D projections are made to be used as an input for the mentioned neural network.

Finally, the results were evaluated, leading to the discard of the initially proposed architecture and a discussion about possible ways to improve the results.

Keywords: Acoustic biometrics, MEMS microphone array, LiDAR, Verification, Siamese networks, Convolutional neural networks (CNN)

Índice general

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

Actualmente vivimos en una era en la que los sistemas biométricos se encuentran presentes en nuestro día a día. ¿Cuántas veces al día utiliza una persona promedio su huella dactilar o el reconocimiento facial para desbloquear su teléfono móvil? Nos encontramos rodeados de tecnología de identificación que, si bien resultó asombrosa hace no tantos años [1], en la actualidad a nadie despierta sorpresa.

Sin embargo, imagine por un momento la posibilidad de ser reconocido no por su apariencia visible, sino por el eco sutil de su presencia. Esta es la meta que se persigue en la investigación a la que pertenece este proyecto y aunque puede parecer difícil de creer, la existencia de ecos acústicos suficientemente representativos de la identidad de una persona se ha podido confirmar en estudios anteriores [2].

El sistema de identificación biométrica planteado, al estar basado en reflexiones acústicas, supone una gran ventaja en entornos de poca o nula visibilidad. Mientras una cámara de vídeo resultaría inútil, el sistema acústico respondería de igual manera puesto que el sonido no precisa de luz para su propagación. Esta característica dota al sistema de posibles aplicaciones interesantes como la vigilancia o identificación en entornos de trabajo de baja visibilidad.

En la publicación inmediatamente anterior perteneciente a esta investigación [3], se estudió la posibilidad de implementar un modelo paramétrico que permitiese asociar un conjunto de características a cada imagen acústica. No obstante, debido a la complejidad del análisis de las imágenes, se planteó a modo de linea futura filtrar las imágenes acústicas con información LiDAR para facilitar el estudio y probar nuevos algoritmos de identificación. Además, algunos estudios corroboran que el funcionamiento de los sistemas biométricos multimodales es muy superior a los monomodales, por lo que combinar ambas fuentes de información puede ser una idea prometedora [4], [5].

Los sistemas de identificación funcionan esencialmente como un sistema de reconocimiento de patrones. Es decir, consiguen extraer de un conjunto de datos una serie de parámetros o características que identifiquen de manera inequívoca al individuo en cuestión [6].

En estudios anteriores [3], ese conjunto de datos de entrada al algoritmo se obtenía a partir de un modelo paramétrico. No obstante, se plantea el problema desde un enfoque nuevo en la investigación, centrado no en extraer una serie de parámetros de las imágenes sino en utilizarlas directamente. En este contexto, la carga computacional aumenta y las redes neuronales convolucionales (CNN) son especialmente adecuadas para procesar imágenes e identificar patrones visuales [7], [8]. De hecho en el ámbito biométrico son muchas las fuentes de información para las que se han utilizado redes de este tipo. Desde los conocidos sistemas basados en huella dactilar [9] o reconocimiento facial [10] hasta otros más complejos utilizando por ejemplo imágenes de venas dactilares [11] o señales electroencefalográficas [12].

En comparación con los enfoques tradicionales de aprendizaje automático, los métodos

basados en aprendizaje profundo han mostrado mejores resultados en términos de precisión y velocidad de procesamiento en el reconocimiento de imágenes [10]. Aún así, no todo es el algoritmo de entrenamiento. Es indispensable contar con un set de imágenes amplio y de calidad para obtener buenos resultados por lo que se debe plantear un sistema de procesado ingenioso que permita obtener datos significativos y con la mínima información redundante. Además, antes de abordar el procesado, es preciso desarrollar un sistema de adquisición síncrono para ambos sistemas (acústico y LiDAR) y decidir algunos parámetros determinantes para las firmas acústicas como la frecuencia o ancho del pulso transmitido.

1.1. Objetivos

1.1.1. Objetivo general

A partir de imágenes acústicas adquiridas de manera inteligente mediante un array de sensores MEMS de alta resolución y utilizando un escáner LiDAR, el objetivo general es el diseño, implementación y evaluación de prestaciones de un sistema biométrico multimodal que permita validar correctamente la identidad de distintos individuos a través de un algoritmo de aprendizaje automático.

1.1.2. Objetivos específicos

Por otra parte, aunque necesarios para la consecución del objetivo general, otros objetivos específicos se han planteado:

- Diseñar e implementar en LabVIEW los drivers necesarios para la obtención de los datos ofrecidos por la cámara LiDAR de intel "L515".
- Desarrollar una etapa de procesado para fusionar de manera coherente la información acústica y LiDAR.
- Obtener una base de datos que permita alcanzar conclusiones válidas y servir de cimiento para futuros análisis.
- Estudiar la información acústica disponible para inferir qué características pueden ser potencialmente diferenciales entre los individuos.
- Analizar la utilidad de la información acústica multifrecuencial y estudiar a través de qué frecuencias se pueden obtener mejores prestaciones.
- Plantear y examinar un algoritmo de inteligencia artificial adecuado para resolver el problema de validación.
- Proporcionar un análisis estadístico final del sistema biométrico.
- Evaluar la conveniencia del uso de imágenes acústicas para identificación de individuos.

1.2. Estructura y fases del proyecto

En base a la consecución de los objetivos planteados y tomando en cuenta las etapas del proyecto, se presenta y justifica a continuación la estructura del presente informe.

En primera instancia, el marco teórico proporciona al lector una perspectiva general del proyecto propuesto. Para ello, se exponen y analizan diversos conceptos considerados esenciales para la comprensión del desarrollo y los resultados que se plantean a lo largo del estudio.

Los siguientes capítulos sin embargo, se encuentran organizados de forma secuencial, tal y como se aborda el desarrollo del proyecto. En línea con este enfoque, el proyecto se puede separar principalmente en tres fases perfectamente distinguibles:

1.2.1. Diseño e implementación del sistema de adquisición

Como se ha justificado previamente, el proyecto comienza desde la adquisición de las imágenes tanto acústicas como LiDAR. En el capítulo 3 de la memoria se presentan los sistemas de adquisición, incluyéndose el hardware y el software asociado. Además de desarrollar el software, en este capítulo se deciden factores como la disposición de los sensores en el array, posición física óptima, parámetros en la conformación y otras variables significativas como el tiempo de pulso transmitido, las frecuencias de transmisión o el margen de captura.

Por último, en la sección 4.1 perteneciente al capítulo 4 se explica el procesado seguido hasta la obtención de una única imagen resultado de la fusión de la imágen acústica y LiDAR.

1.2.2. Obtención de firmas acústicas

En general, se requiere de una cantidad considerable de muestras para aplicar un algoritmo de *deep learning*. Además la solidez del estudio acústico está fuertemente relacionada con la cantidad de individuos e imágenes capturadas de cada uno por lo que se ha considerado importante capturar un mayor número de individuos que en estudios previos.

Otro aspecto a considerar es el escenario de captura y la importancia de generar condiciones estables para minimizar la variabilidad en las imágenes debido a elementos externos al individuo.

Esta etapa se desarrolla en la sección 4.2 perteneciente al capítulo 4 donde se presenta el escenario de adquisición, estructura de datos y estrategia de captura de muestras.

1.2.3. Desarrollo y test de un algoritmo de verificación

Por último, una vez obtenidas todas las muestras disponibles para abordar el problema de verificación, se define e implementa el algoritmo de inteligencia artificial que se describe en el capítulo 5 donde se plantea el desafío desde la perspectiva de implementar un algoritmo adecuado a los datos disponibles.

Previo a la elección del algoritmo, en este capítulo se realiza un primer análisis acerca de la información acústica y las características que pueden conferir a las imágenes un carácter identificativo de cada individuo.

Seguidamente, se construyen los conjuntos de entrenamiento, validación y test diseñando un algortimo específico para la division y agrupación en parejas de los datos obtenidos.

Finalmente, para comprobar el funcionamiento del sistema, se han introducido imágenes nuevas pertenecientes al conjunto de test, (es decir, que no se habían utilizado previamente en la fase de entrenamiento. El objetivo final es proporcionar un análisis de la probabilidad de acierto y error del sistema a la hora de verificar los individuos teniendo presente la aplicación como identificador biométrico del sistema.

1.3. Estado del arte

El carácter novedoso del proyecto tiene su origen en el uso de imágenes acústicas para biometría dado que es algo que no se ha realizado con anterioridad. Es por ello que en la actualidad es aún muy escasa la investigación sobre sistemas de reconocimiento biométrico basados en técnicas pulso-eco por medio de un array acústico.

En particular, el uso de las técnicas pulso-eco es mucho más habitual en el ámbito de la detección. De hecho, las tecnologías RADAR y SONAR siempre han estado muy presentes en la localización de obstáculos en navegación o conducción.

Fuera del ámbito de identificación biométrica, los estudios de Moebus y Zoubir, abordan el problema de la parametrización de imágenes obtenidas a través de un array acústico modelando los ecos a través de GMM (Gaussian-Mixture-Model) [13]. Estos mismos autores emplean una banda de frecuencia ultrasónica para generar a través de un array planar imágenes acúsicas 3D usando técnicas de conformación (beamforming) [14]. De hecho, Moebus aplicó estos avances en su tesis donde consiguió clasificar distintos objetos a través de sus imágenes acústicas para poder discriminar humanos de otros objetos con una precisión cercana al 97 % [15].

Por otra parte, la utilización de imágenes LiDAR en el área de la biometría tiene un escaso desarrollo en la bibliografía. No obstante, algunos estudios emplean la forma de andar de diferentes individuos como rasgo biométrico. Por ejemplo, en un estudio realizado en 2022 [16], se presenta un sistema biométrico que utiliza el vídeo capturado con un sensor LiDAR colocado a la altura del tobillo de los individuos. Este sistema emplea una red neuronal para categorizar entre 26 posibles individuos, la persona registrada de acuerdo a su andar, logrando niveles de precisión cercanos al 99 %.

Por el contrario, sensores LiDAR se utilizan tanto en el sector de la robótica [17],[18] como en el de automoción [19] para la detección y seguimiento de objetos.

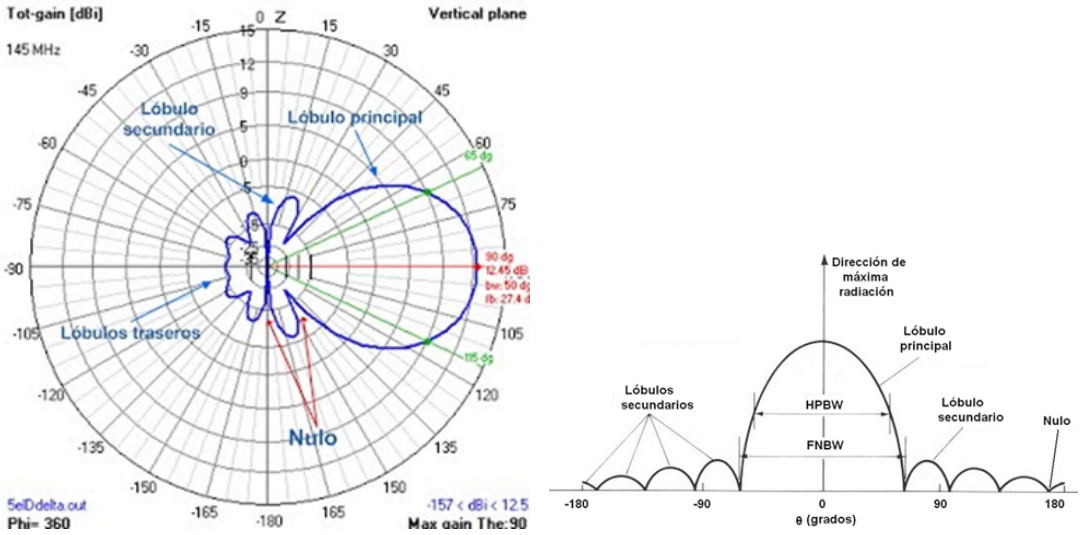
Aún así, como se ha comentado previamente, este proyecto pertenece a una investigación en curso a través de la cual si se pueden encontrar estudios previos relevantes. En 2015, en un estudio perteneciente a esta investigación se propuso un primer sistema de biometría acústica utilizando un array simple. En concreto, el sistema empleaba un array unidimensional, preprocesaba las imágenes utilizando técnicas GMM y por último las clasificaba a través de SVM (Support Vector Machine) [20].

Finalmente, el presente trabajo constituye la continuación lógica de la publicación más reciente en relación a esta investigación [3] siendo su génesis las lineas de mejora planteadas en 2022 en dicho proyecto.

Capítulo 2. Marco Teórico

2.1. Imágenes acústicas multifrecuencia

Utilizando un array de micrófonos 2D y enviando una señal sinusoidal pulsada a través de un transmisor, se obtiene una imagen acústica en 3 dimensiones. Si además se tiene en cuenta que el sistema puede funcionar en múltiples frecuencias, se puede considerar estas imágenes como 4D. La representación gráfica de las características de radiación de una antena es el diagrama de radiación (beampattern). A través de esta representación, pueden obtenerse, entre otros parámetros, la dirección de apuntamiento, la directividad de la antena o el ancho del lóbulo principal. El diagrama de radiación de la antena se puede representar en tres o en dos dimensiones dependiendo de la dimensionalidad del espacio de estudio. La figura adjunta consiste en una representación 2D para una coordenada angular (Figura 2.1).



(a) (b)

Figura 2.1: Representaciones 2D del diagrama de radiación de una antena

El principio pulso-eco (Figura 2.2) estima la distancia a la cual se encuentra un objeto para una dirección espacial determinada, asumiendo que el diagrama de radiación de la antena es lo suficientemente estrecho.

Figura 2.2: Técnica pulso-eco aplicada con un solo transceptor

Las imágenes acústicas se obtienen cuando se escanea un área del espacio. Es decir, se barren distintos ángulos que son abarcados en la región objetivo. Este escaneo se puede realizar de forma mecánica o electrónica. Las antenas de barrido mecánico suelen incluir una o más piezas articuladas como lentes o espejos que permiten al sistema apuntar físicamente a las distintas zonas del espacio para así capturar la región de interés [21].

Al disponer de un array de sensores, se puede realizar este escaneo de forma electrónica gracias a un proceso de conformación de haces (beamforming). De este modo, es posible construir una imagen acústica a través de la información recibida en cada uno de los sensores del array.

Un transmisor genera la señal pulsada y se forma una imagen a partir de las señales provenientes de los ecos que se reciben en cada uno de los distintos micrófonos del array [22]. Para determinar los ángulos y distancia a la que se encuentra el blanco, se siguen los principios básicos de los sistemas RADAR (Radio Detection and Ranging). Se calcula la distancia a partir de la velocidad del sonido en el medio y el tiempo empleado por la onda acústica en viajar hasta el blanco y volver. La dirección o posición angular se puede determinar gracias a la dirección de llegada de la onda reflejada [23].

El conformador (beamformer), como se puede ver en la figura inferior (Figura 2.3), combina linealmente las señales recibidas en cada micrófono x_n. Para ello, las multiplica por sus pesos complejos asociados w_n y se suman entre si obteniendo una señal de salida correspondiente a un ángulo de apuntamiento especifico que permite construir la imagen 2D evaluado el espacio de vigilancia en un conjunto significativo de ángulos [2]. Así, modificando el valor de los pesos (desfases), se consigue que el diagrama de radiación "apunte" a posiciones espaciales diferentes, ya que según los valores de w_n, las señales capturadas por los micrófonos se sumarán o no en fase.

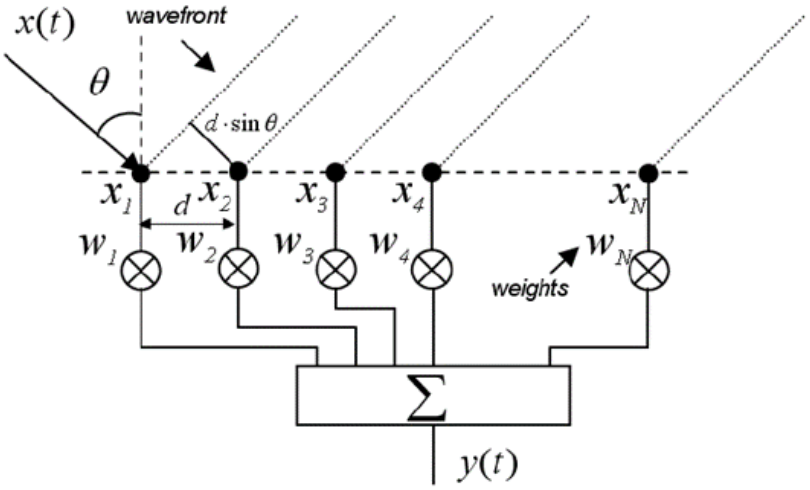


Figura 2.3: Estructura de un conformador [2]

Existen distintas técnicas de conformado y de hecho se puede reducir el nivel de los lóbulos secundarios en el diagrama de radiación modificando el módulo de los pesos complejos w_n. Esta práctica se conoce como enventanado espacial y puede ser muy útil dependiendo la aplicación. Pero no todo son ventajas ya que disminuir el efecto de los lóbulos secundarios implica un aumento del ancho del lóbulo principal y por ende una pérdida de resolución espacial. Enventanar permite también permite tener en cuenta los valores de ganancia individual de cada uno de los micrófonos para normalizar sus diferencias. Estas imágenes se obtienen en sistema de coordenadas esféricas por lo que las 3 primeras dimensiones representan acimut, elevación y rango, mientras que la cuarta representaría, para las imágenes multifrecuencia, cada una de las frecuencias utilizadas. El centro del array corresponde al centro de coordenadas 0, el rango r es la distancia desde ese centro hasta el punto y por último las magnitudes acimut θ y elevación φ, recogen los ángulos horizontal y vertical respectivamente (Figura 2.4).

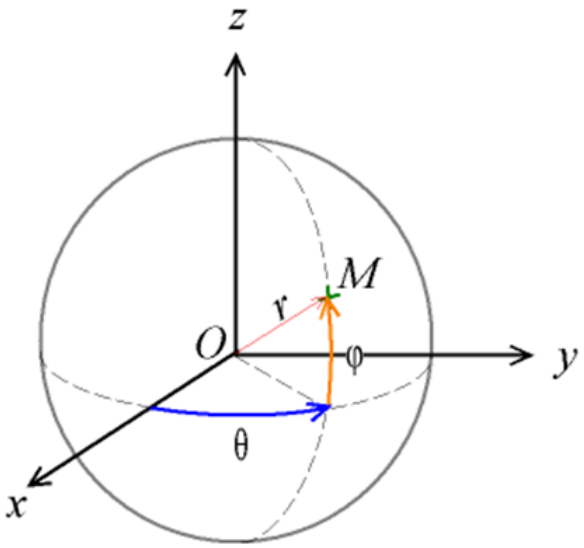


Figura 2.4: Sistema de coordenadas esféricas

Para visualizar las imágenes 3D, típicamente se utilizarán proyecciones de distinto tipo sobre los distintos planos o en su defecto cortes en la imagen. Además, en sus representaciones las imágenes se normalizan respecto a un valor máximo de amplitud. A continuación, se muestra un ejemplo de una imagen acústica de un único blanco y representada por su proyección sobre el plano formado por las coordenadas acimut y elevación (Figura 2.5). Los puntos con mayor energía se representan como puntos más claros y los que menos energía contienen se representan en un tono más oscuro.

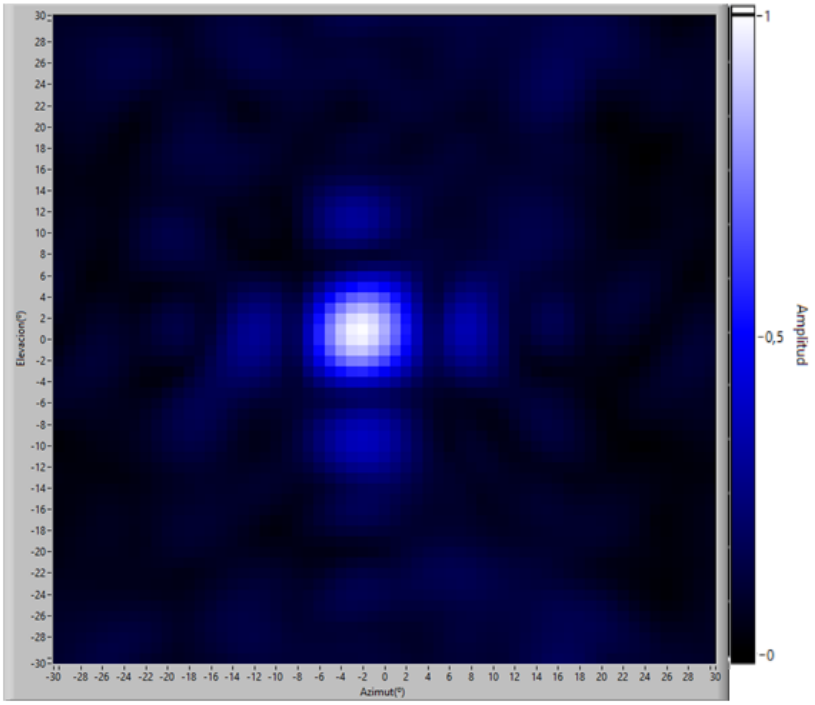


Figura 2.5: Proyección sobre el plano azimut-elevación de un blanco puntual

2.2. Biometría acústica y antropometría

El término "biometría" se refiere al hecho de identificar una persona en base a sus características fisiológicas o comportamiento. Un sistema biométrico adquiere datos biométricos de un individuo, extrae las características consideradas y posteriormente compara dichas características con un modelo existente creado previamente. Así, se decide o bien quién es la persona en cuestión o bien si es quien dice ser dependiendo si el sistema está orientado a identificación o verificación de individuos [24].

Existen varias cualidades que se buscan a la hora de escoger unas características biométricas de calidad. Principalmente: universalidad, singularidad, permanencia, desempeño, aceptabilidad, y fiabilidad [25]. Las cualidades más relevantes para este trabajo serán las 4 primeras:

- Universalidad se refiere al hecho de encontrar unos factores biométricos que estén presentes en todos los individuos.

- Singularidad es una medida de cómo de único es ese factor biométrico sobre la población total de los individuos.
- Permanencia se refiere a cómo de constante se mantiene la característica correspondiente con el paso del tiempo.
- Por último, desempeño se refiere a cómo de bien funciona el sistema. Factores cómo la exactitud, rapidez y robustez son tenidas en cuenta acorde con el requerimiento de recursos.

El término "biometría acústica" se refiere a las técnicas de biometría que se basan en ondas acústicas para su funcionamiento. En el caso en el que la frecuencia de las ondas se encuentre por encima del espectro auditivo (aproximadamente 20 kHz), se suele emplear el término "biometría ultrasónica".

Por otra parte, asumiendo que el factor principal que origina las diferencias entre las imágenes acústicas de unas personas y otras es la figura antropométrica del individuo, cabe preguntarse cómo de diferentes son unas figuras humanas de otras. Es por ello por lo que la antropometría tiene cabida en este estudio.

Antonio Carmona, recoge en su libro [26] los principales aspectos antropométricos de la población laboral española. En este libro, se realiza un análisis estadístico de diversas medidas tomadas a una serie de individuos. De los datos presentados en el mencionado escrito, se puede inferir que los percentiles en los que una persona se encuentra para las diferentes medidas antropométricas son notablemente distintos entre sí. Es evidente que un individuo que presente una alta estatura contará seguramente con una gran envergadura, pero difícilmente se encontrará en el mismo percentil con respecto al resto de la población en ambas medidas.

Esto es una conclusión interesante, ya que demuestra la utilidad de considerar diversas medidas de un individuo para su identificación. Esto no sería producente si las distintas medidas fuesen redundantes entre sí. Finalmente, la respuesta acústica asociada a las características antropométricas de cada individuo es lo que se denominará como "firma acústica" y será identificativo de cada persona.

2.3. Redes neuronales convolucionales

Los principios de funcionamiento de las redes neuronales convolucionales son un tema complejo. Por ello se ha considerado necesario explicar de manera extensa y con cierto nivel de detalle los conceptos más importantes para el caso de estudio.

En esencia, el aprendizaje profundo o *deep learning* se refiere a una categoría de técnicas de aprendizaje automático que emplean modelos formados por múltiples capas de procesamiento apiladas unas sobre otras. Por ello, las redes neuronales convolucionales representan un modelo típico en el aprendizaje profundo.

Enmarcado en el ámbito de aprendizaje profundo, la discusión comienza por las redes neuronales artificiales ANN (Artificial Neural Network). Las ANN se componen principalmente de un elevado número de nodos computacionales interconectados (denominados neuronas), que trabajan de forma entrelazada y distribuida para aprender

colectivamente sobre los datos de entrada y optimizarlos [27]. De esta forma, la estructura básica de una ANN puede modelarse como se observa en la imagen a continuación (Figura 2.6). Se aplica la entrada en forma de vector multidimensional a la capa de entrada que lo distribuirá a las capas ocultas. A continuación, estas toman decisiones a partir de la capa anterior y consideran cómo un cambio estocástico dentro de sí misma perjudica o mejora la salida final. Esto es lo que se conoce como proceso de aprendizaje [27].

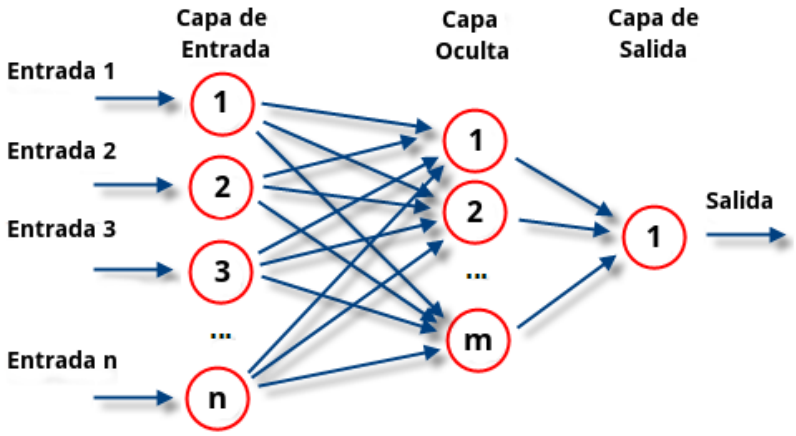


Figura 2.6: Ejemplo básico de una red neuronal artificial

Los dos paradigmas de aprendizaje clave en las tareas de procesamiento de imágenes son el aprendizaje supervisado y no supervisado. A diferencia del aprendizaje no supervisado, el aprendizaje supervisado consiste en aprender a partir de entradas previamente etiquetadas, que actúan como objetivos. De este modo, para cada ejemplo de entrenamiento, habrá un conjunto de valores de entrada y uno o varios valores de salida asociados. En consecuencia, el objetivo de esta forma de entrenamiento es reducir el error global de clasificación de los modelos mediante el cálculo correcto de los valores de salida [27].

Las redes neuronales convolucionales CNN (Convolutional Neural Networks), al igual que las ANN tradicionales, están compuestas por neuronas que se optimizan mediante el aprendizaje. Cada neurona recibe una entrada y realiza una operación. La única diferencia entre las CNN y las ANN tradicionales es que las redes neuronales convolucionales se utilizan principalmente en el campo de reconocimiento de patrones en imágenes. Una de las mayores limitaciones de las tradicionales ANN, es la complejidad computacional necesaria para calcular datos de imágenes. Por ejemplo, ante imágenes monocromas pequeñas de sólo 28x28, una sola neurona de la primera capa oculta contendrá 784 pesos (28x28x1). En un caso más común, ante una imagen en color de tamaño 64x64, el número de pesos asciende a 12288 (64x64x3) [27].

Como se ha señalado, las CNN se basan principalmente en que la entrada está compuesta por imágenes. Esto hace que la arquitectura se configure de manera que se adapte mejor a la necesidad de tratar con el tipo específico de datos. De esta manera, las neuronas en las capas de la CNN se componen de 3 dimensiones: la dimensionalidad espacial de entrada (ancho y alto) y la profundidad. A diferencia de las ANN estándar,

las neuronas dentro de cualquier capa sólo se conectarán a una pequeña región de la capa anterior denominada campo receptivo o *receptive field* (Figura 2.7). De esta manera, se reduce considerablemente el número de pesos [27].

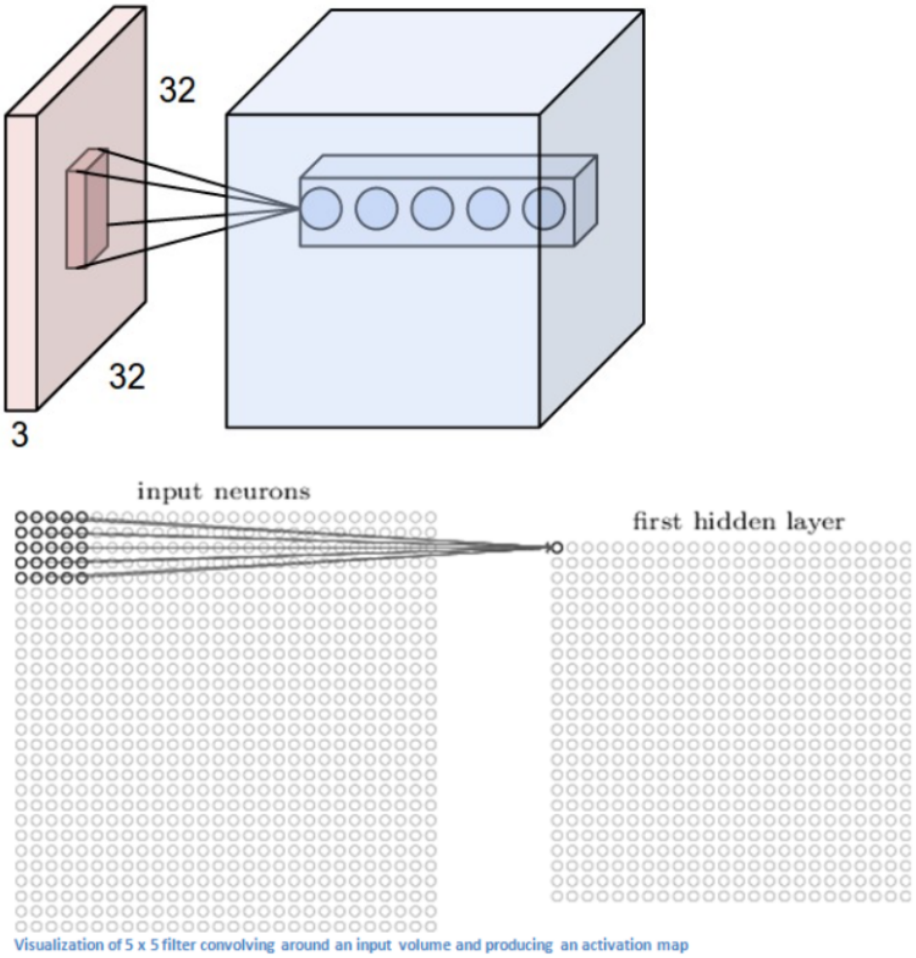


Figura 2.7: *Receptive field* de una neurona

Es importante explicar también los conceptos de época (*epoch*) y lote (*batch*) que se emplean más adelante. Un lote es un conjunto de ejemplos que se evalúan de manera conjunta durante el entrenamiento. El procesamiento por lotes mejora la eficiencia computacional y permite actualizar los pesos de la red más suavemente.

Por el contrario una época representa una iteración completa a través de la totalidad de datos de entrenamiento. Es decir, la primera época finaliza cuando la red ha sido entrenada con cada ejemplo una vez independientemente del tamaño del lote. A grandes rasgos, si el número de épocas es demasiado pequeño, el modelo puede no aprender las relaciones de los datos (*underfitting*) mientras que si es demasiado grande aumenta el tiempo de entrenamiento y puede ser redundante o incluso producirse un fenómeno de sobreajuste (*overfitting*).

2.3.1. Arquitectura general de una red neuronal convolucional

Las CNN constan fundamentalmente de cuatro tipos de capas: capas convolucionales con capas ReLu (de unidad lineal rectificada), capas de agrupamiento y capas totalmente conectadas (Figura 2.8). Aunque también se utilizan otras capas como las de normalización o "dropout".

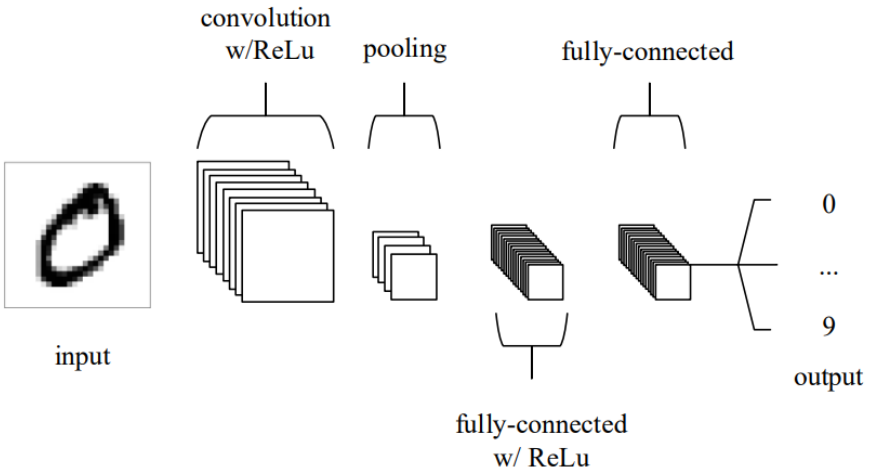


Figura 2.8: Arquitectura de una red neuronal convolucional simple de 5 capas

Capas convolucionales

Las capas convolucionales desempeñan un papel fundamental en el funcionamiento de las CNN. Los parámetros de las capas se centran en el uso de "kernels" aprendibles por la red. Cuando los datos llegan a una capa de este tipo, se realiza el producto escalar entre cada filtro y el campo receptivo de entrada, es decir, la región de tamaño del "kernel"que se desplaza sobre la matriz de entrada. Esta convolución produce un mapa de activación 2D (Figura 2.9). De la misma manera que ocurre en filtrado de imágenes, se desplaza el kernel a través de la imagen de entrada calculando todos los valores de la nueva imagen 2D [27].

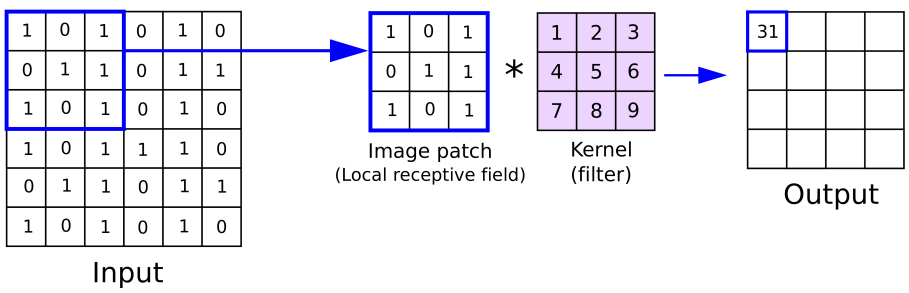


Figura 2.9: Resultado de aplicar una convolución aplicando un kernel 3x3

De este modo, el tamaño del campo receptivo será igual al tamaño del kernel y condicionará directamente el número de pesos en cada neurona. Si como en el ejemplo planteado anteriormente (imagen en color de dimensiones 64x64x3), se establece el tamaño del "kernel" en 6x6, se obtiene un total de 108 pesos en cada neurona de la capa. Esto es más de 100 veces menos del número de pesos (12288) que en una ANN [27].

Existen tres hiperparámetros que permiten optimizar los resultados de estas operaciones:

Número de kernels La convolución se puede realizar a través de más de un kernel (Figura 2.10). Esto aumenta el número total de neuronas de la red y la capacidad de reconocimiento de patrones del modelo. De este modo, el número de filtros determina el número de matrices por las que se van a convolucionar las imagenes de entrada a la capa.

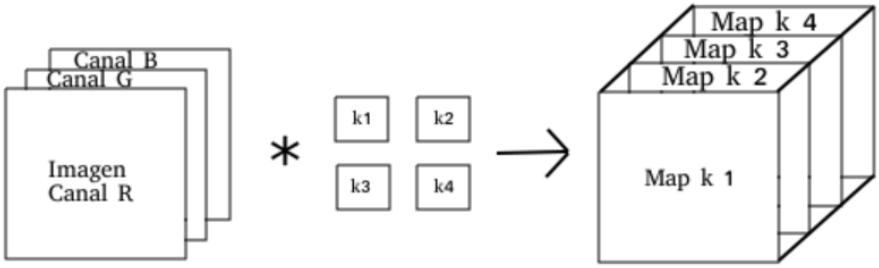


Figura 2.10: Capa convolucional de 4 kernels

Tamaño del kernel El tamaño del kernel, campo receptivo o *Receptive field* es otro parámetro a considerar. Los filtros suelen ser cuadrados en la mayoría de las ocasiones aunque se pueden definir rectangulares. Un tamaño menor implica menor número de neuronas en la red reduciendo la complejidad de tal forma que puede disminuir significativamente la capacidad de reconocimiento de patrones del modelo.

Paso (stride) Es posible definir también el paso en el que se fija el desplazamiento en torno a la dimension espacial de entrada para situar el campo receptivo. Por ejemplo, si el paso es igual a 1, el campo receptivo se encontraría muy superpuesto y produciría activaciones grandes. Por el contrario, fijando un número mayor, se reduce el solapamiento, y se obtiene una salida de dimensiones espaciales menores [27]. A continuación se muestran dos ejemplos que ilustran el comportamiento de la capa según el paso utilizado (Figura 2.11) donde los recuadros de colores naranja, morado y verde marcan 3 campos receptivos consecutivos y su correspondencia con los píxeles de salida.

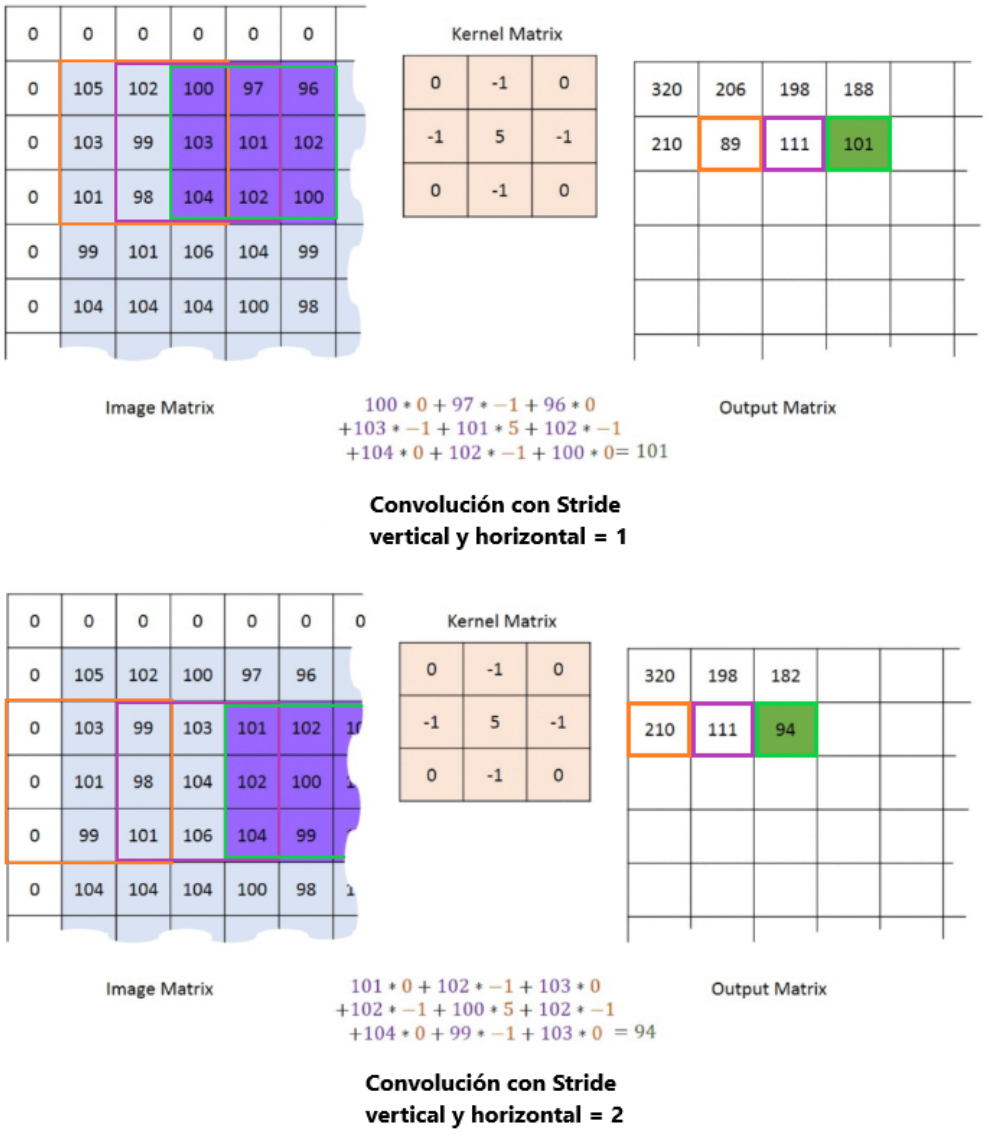


Figura 2.11: Operación de una capa convolucional con paso = 1 y paso = 2

zero-padding El relleno con ceros es un método eficaz para controlar las dimensiones de las imágenes de salida.

Normalmente el filtro comienza en una esquina de la imagen recorriendo todas las filas y columnas hasta que el filtro llega a la esquina opuesta de la imagen. Una visión alternativa para aplicar un filtro a una imagen es garantizar que cada píxel de la imagen tenga la oportunidad de estar en el centro del filtro ya que de la otra forma los píxeles del borde de la imagen de entrada sólo están expuestos al borde del filtro.

Sin embargo, esta visión generaría problemas ya que en la entrada, los píxeles en el borde no cuentan con la totalidad de píxeles vecinos alrededor necesarios para aplicar el kernel. Surge entonces la idea de agregar un borde de "x" píxeles alrededor del exterior de la imagen permitiendo a todos los píxeles interactuar con el filtro de la misma manera. Esto se conoce como "zero-padding" y se traduce en más oportunidades para

que el filtro detecte características y una salida de mapa de características de mayores dimensiones.

Por ejemplo, en el caso de aplicar un filtro de 3×3 a una imagen de entrada de 8×8 es factible agregar un borde de un píxel alrededor del exterior de la imagen. Tras aplicar "zero-padding", las nuevas dimensiones de la imagen de entrada son 10x10 por lo que tras aplicar el filtro 3x3, se obtiene un mapa de características de 8x8.

Dimensionalidad Por último, es importante entender que utilizando está técnica se alteran las dimensiones de salida de las capas convolucionales. Para calcular esto, suponiendo una imagen de entrada de tamaño (W_1 x H_1 x D_1), se pueden emplear las siguientes fórmulas calcular el tamaño del volumen a la salida (W_2 x H_2 x D_2) [28],[28]:

$$W_2 = \frac{(W_1 - F) + 2 \cdot P}{S + 1} \tag{2.1}$$

$$H_2 = \frac{(H_1 - F) + 2 \cdot P}{S + 1} \tag{2.2}$$

$$D_2 = K \tag{2.3}$$

Dónde:

- Número de filtros = K
- Tamaño del filtro = F
- Paso = S
- Cantidad de relleno a cero = P

Por último, si el resultado de esta ecuación no es un entero, el paso no se han ajustado correctamente, ya que las neuronas no podrán ajustarse perfectamente a la entrada dada [27]. Por ejemplo, si se utilizase una matriz de entrada de tamaño 6x6 como la de la figura 2.9 y se aplicase sobre ella una capa convolucional consistente en 4 "kernels" de dimensiones 3x3, un paso de 2 y un *padding* de 0, entonces:

$$W_2 = \frac{(6 - 3) + 2 \cdot 0}{2 + 1} = \frac{5}{3} \tag{2.4}$$

$$H_2 = \frac{(6 - 3) + 2 \cdot 0}{2 + 1} = \frac{5}{3} \tag{2.5}$$

$$D_2 = 4 \tag{2.6}$$

Como W_2 y H_2 no son números enteros, los parámetros están mal ajustados ya que la matriz de entrada no se puede ajustar a ellos y no se puede recorrer pasando las mismas veces por cada elemento. La solución en este caso sería utilizar un paso de 1 o de 3 ya que aplicar *padding* no solucionaría el problema.

Capas ReLu

Las capas ReLU (Rectified linear unit) tienen como objetivo aplicar la función de activación por elementos a la salida producida por la capa anterior. Típicamente, después de una capa convolucional se aplica una capa ReLU para introducir no linealidad en el modelo y extraer características importantes [27],[28].

Distintas funciones de activación se pueden emplear en cada capa ReLU [28]:

- $f(x) = max(0, x)$ (de las más utilizadas)
- $f(x) = tanh(x)$
- $f(x) = |tanh(x)|$ (permite entrenar con mayor rapidez y con un rendimiento similar)
- $f(x) = \frac{1}{1+e^{-x}}$ (función sigmoide)

Al tratarse de operaciones elemento a elemento, la aplicación de una capa RELU deja el tamaño del volumen sin cambios. Además, no existe ningún parámetro a establecer ni contienen pesos entrenables y como se comenta más tarde, la propagación hacia atrás es sencilla ya que no se realiza ninguna operación [28].

Capas de agrupamiento (pooling)

Las capas de pooling se utilizan para ir reduciendo el tamaño de los mapas de activaciones, disminuyendo así el número de parámetros y la complejidad computacional del modelo ya que de otra forma no sería posible ejecutarlos en GPUs. Esa capa opera sobre cada mapa de activación a la entrada y escala su dimensionalidad según la función utilizada. En esencia, la operación de agrupación implica la extracción de regiones (k x k) y aplicar una función f al mapa seleccionando un sólo valor para cada caso. Esto se hace para todos los volúmenes de entrada por lo que conservan su profundidad ($D_1 = D_2$) [27],[28].

Existen distintas opciones para la función f empleada en la agrupación siendo las más empleadas:

- $f = max$: Es la función que ha demostrado dar los mejores resultados, consiste en escoger el valor máximo de entre los obtenidos a través de la ventana de entrada y recibe el nombre de max-pooling.
- $f = mean$: Consiste en calcular la media entre los elementos de la ventana y se conoce con el nombre de average-pooling.
- Agrupación L2 (L2-pooling): La salida es la raíz cuadrada de la suma de los cuadrados de los elementos de la ventana de entrada

Aunque también se podría aplicar una operación del tipo "zero-padding" no es habitual por lo que principalmente dos parámetros determinan la operación de estas capas:

- Tamaño de la ventana de agrupamiento = F
- Paso = S

Debido a la naturaleza destructiva de las capas de pooling, normalmente se suele utilizar $F = 2$ y $S = F$ permitiendo que la capa se extienda por toda la entrada (Figura 2.12). De hecho, un tamaño del filtro superior a 3 normalmente disminuye el rendimiento del modelo [27].

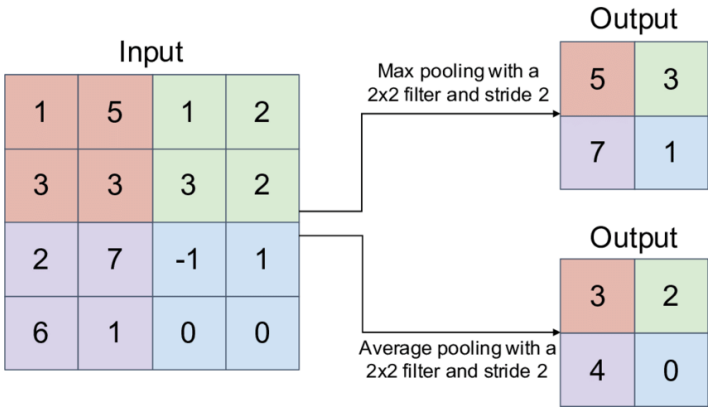


Figura 2.12: Max-pooling y Average-pooling: filtro 2x2 y $S = 2$

De nuevo y de forma similar a las capas convolucionales, en las capas de agrupamiento se alteran también las dimensiones de salida [29]. Suponiendo un volumen de entrada de tamaño (W_1 x H_1 x D_1), se obtiene un volumen a la salida de tamaño (W_2 x H_2 x D_2) tal que:

$$W_2 = \frac{W_1 - F}{S} + 1 \tag{2.7}$$

$$H_2 = \frac{H_1 - F}{S} + 1 \tag{2.8}$$

$$D_2 = D_1 \tag{2.9}$$

Una vez más, el resultado de esta ecuación debe ser un entero ya que de lo contrario los parámetros no se han ajustado correctamente.

Capa de Dropout

Dropout es una técnica de regularización que permite durante el entrenamiento desactivar una serie de neuronas seleccionadas al azar. De esta manera, su contribución a la activación de las neuronas siguientes se excluye temporalmente en la propagación hacia delante y no participan en la retropropagación. El resultado es una red menos sensible a pesos específicos, lo que permite una mejor generalización y por ende un menor sobreajuste a los datos de entrenamiento. Además, mejora la eficiencia computacional en redes con gran número de parámetros dado que la red se vuelve más pequeña durante cada una de las iteraciones [28],[30].

No obstante, hay que tener en cuenta que el uso de capas de "dropout" aumenta el número de iteraciones necesarias para converger [30].

Capas de normalización

La normalización es una técnica de preprocesamiento utilizada para estandarizar datos. En otras palabras, tener diferentes fuentes de datos dentro del mismo rango. No normalizar los datos antes de entrenar puede resultar en que las características con los rangos más altos podrían sesgar los modelos hacia ellas.

Pensándolo detenidamente, en las redes neuronales, la salida de cada capa sirve como entrada hacia la siguiente capa por lo que: si normalizar las entradas al modelo ayuda a mejorar el rendimiento del modelo, ¿normalizar las entradas a cada capa también ayuda a mejorar el rendimiento del modelo?

La respuesta es afirmativa la mayoría de veces. Sin embargo, normalizar las entradas de las capas intermedias es más complicado ya que cambian constantemente por lo que calcular continuamente las estadísticas de todo el conjunto de datos sería inviable desde el punto de vista computacional. Una estrategia común en normalización es restar la media y dividir entre la desviación típica el conjunto de datos. Esto permite obtener una distribución de datos insesgados (de media 0) y con desviación típica igual a 1.

La capa de normalización más común es "BatchNormalization" (BN) que se realiza a lo largo de minilotes *mini-batches*. Matemáticamente, la capa transforma cada entrada del minilote actual restando la media de entrada en ese minilote y dividiéndola por la desviación estándar [31]. No obstante, cada capa no siempre necesita entradas con media cero y varianza unitaria, sino que en algunos casos el modelo podría funcionar mejor con alguna otra media y varianza. Por lo tanto, la capa BN también introduce dos parámetros de media y varianza que se pueden aprender: γ y β .

La operación de normalización se realiza tal como se muestra en la tabla a continuación (Figura 2.13):

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Figura 2.13: BN aplicada sobre un mini lote B para obtener la salida y 2.13

donde parámetro 'ϵ' corresponde con un valor muy pequeño necesario para la estabilidad numérica del denominador.

Para finalizar con BN, entrenar una red usando normalización por lotes es beneficioso para la generalización de la red lo que permite eliminar o reducir la fuerza de las capas dropout. Además, la capa calcula las estadísticas de lote (media y varianza) en cada iteración de entrenamiento, por lo que requiere tamaños de lote más grandes durante el entrenamiento para que estos parámetros puedan aproximarse de manera efectiva 2.13. El uso de BN permite también utilizar tasas de aprendizaje más altas para los optimizadores dado que los pequeños cambios en parámetros de una capa no se propagan a otras capas.

Otras capas de normalización son posibles como las capas LRN (Local Response Normalization) o en castellano, de "normalización de respuesta local". Este tipo de capas se introdujeron por primera vez en la conocida arquitectura "AlexNet" y permiten mejorar de manera local el contraste, de forma que los valores máximos locales de los píxeles se utilicen como excitación para las capas siguientes.

Es una capa no entrenable que normaliza al cuadrado los valores de los píxeles en un mapa de características dentro de una ventana o vecindad local. Dependiendo si en la vecindad se utiliza información de distinta profundidad o no, esta normalización puede ser intra-canal o inter-canal.

La normalización sigue la siguiente fórmula [30]:

$$\mathrm{LRN}(x,y) = a_{x,y} \cdot \left(k + \alpha \sum_{j=\mathrm{máx}(0,i-n/2)}^{\mathrm{mín}(N_X-1,i+n/2)} (a_{j,x,y})^2 \right)^{-\beta} \tag{2.10}$$

donde la suma abarca "n" mapas de núcleos adyacentes en la misma posición espacial y "N" es el número total de núcleos de la capa. Las constantes k, n α y β son hiperparámetros cuyos valores se determinan mediante validación [30].

A modo de resumen, BN normaliza las activaciones de toda una capa utilizando estadísticas de lotes de datos, mientras que LRN normaliza localmente dentro de ventanas de activación entre píxeles vecinos ayudando a las neuronas a detectar características locales.

Capa Fully-connected o densas

Al igual que la disposición de las neuronas en las tradicionales ANN, las capas totalmente conectadas contienen neuronas que están directamente conectadas a todas las neuronas de la capa anterior [27]. La idea es realizar una agrupación de la información obtenida para la clasificación final

Aunque existen otros parámetros que se pueden configurar como los pesos y el sesgo, el parámetro configurable principal en este tipo de capas es el número de neuronas que la componen N. De este modo, la salida de esta capa será un único vector de dimensiones

1 x 1 x N y por ello no se emplean capas de convolución después del uso de una capa densa [28].

Por lo general, en las redes neuronales convolucionales, se suele utilizar una secuencia de capas completamente conectadas, estando la última de estas capas asociada con el número de clases presentes en el conjunto de datos. En la capa de salida se introducen los valores finales y se realizará la clasificación mediante una función probabilística específica [28].

Capa de pérdida

La capa de pérdida es la capa final en todo modelo y permite comparar los valores reales de las imágenes etiquetadas con las predicciones. Se pueden plantear distintas funciones de pérdida aunque en todas ellas el objetivo es obtener un número que permita valorar la calidad de la predicción [28].

Un ejemplo es la función de pérdida euclídea [28]:

$$L(y, y') = \frac{1}{2K} \sum_{i=1}^{K} \|y'_i - y_i\|_2^2 \tag{2.11}$$

donde K es el número total de neuronas en la capa de pérdida, y'_i representa las predicciones de las imágenes e y_i representa los valores reales de las imágenes.

2.3.2. Optimizador y cálculo de pesos

El último elemento que permite el entrenamiento propiamente de los pesos de las neuronas es el optimizador. El optimizador se encarga de modificar los parámetros de la red en función de los resultados de la capa de pérdida. Dos de los optimizadores más comunes son SGD (Stochastic Gradient Descent) y Adam (Adaptive Moment Estimation) [32].

El SGD u optimizador por descenso de gradiente se basa en el cálculo de la primera derivada de la función de pérdida. De esta manera, la actualización de los pesos se produce de manera iterativa según la siguiente ecuación:

$$w_{t+1} = w_t - \alpha \nabla J(y_i, y'_i) \tag{2.12}$$

donde w son los pesos, J es la función de pérdida calculada para cada muestra de entrenamiento y'_i y su correspondiente etiqueta y_i y α es la tasa de aprendizaje o *learning rate*, uno de los hiperparámetros más importantes que marca la longitud del paso en el optimizador. Una tasa más alta permite una convergencia más rápida aunque las fluctuaciones de w pueden ser muy grandes provocando que el modelo no converja adecuadamente [32].

No obstante, la actualización de pesos se suele hacer por *batches* ahorrando tiempo de cálculo y proporcionando una estimación más estable. De esta manera, la ecuación anterior (ecuación 2.12) queda modificada de la siguiente manera [32]:

$$w_{n+1} = w_n - \alpha \sum_{i=1}^{batchSize} \nabla J(y_i, y_i') \tag{2.13}$$

Otro optimizador ampliamente utilizado es el Adam. Este utiliza los momentos de primer y segundo orden por lo que aunque es más rápido y converge más fácilmente, es más exigente computacionalmente. El procedimiento de cálculo sería el siguiente [32]:

1. $m_t = (1 - \beta_1)\nabla J(y_i, y_i') + \beta_1 \cdot m_{t-1}$
2. $v_t = (1 - \beta_2)(\nabla J(y_i, y_i'))^2 + \beta_2 \cdot v_{t-1}$
3. $\hat{m}_t = \frac{m_t}{1-\beta_1^t}$
4. $\hat{v}_t = \frac{v_t}{1-\beta_2^t}$
5. $w_{t+1} = w_t - \frac{\alpha \cdot \hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}}$

donde m_t y v_t representan los momentos de primer y segundo orden, β_1 y β_2 controlan la influencia del factor de gradiente y $\hat{m}_t$ y $\hat{v}_t$ representan una corrección de sesgo para los momentos. Finalmente, como en situaciones anteriores, ϵ es un valor cercano a cero para garantizar estabilidad numérica evitando que se anule el denominador, α es la tasa de aprendizaje y w son los pesos antes (w_t) y después de aplicar el optimizador (w_{t+1}).

Dos procesos fundamentales en el entrenamiento de una red neuronal y que se han ido comentando en el resto de secciones son los de propagación hacia adelante (Forward propagation) y retropropagación (Backpropagation). El concepto de propagación hacia delante describe el proceso natural por el cual los datos de entrada van atravesando las distintas capas de la red desde la capa de entrada hasta la capa de salida.

Por otra parte, la retropropagación es algo más compleja pero esencial en el proceso de entrenamiento. Después de la propagación hacia adelante y el cálculo de la función de pérdida, el algoritmo calcula unos gradientes de la función respecto a los pesos de la red $\frac{\partial L}{\partial F}$ comenzando desde la capa de salida hacia atrás. Así, los gradientes indican cómo debe ajustarse cada peso y permiten minimizar la pérdida global. En esencia, la retropropagación no es más que una forma de propagar la pérdida total de vuelta a la red para conocer de qué parte de pérdida es responsable cada nodo y actualizar los pesos en consecuencia [33].

El proceso de retropropagación (Figura 2.14) se muestra a continuación para un nodo de la red:

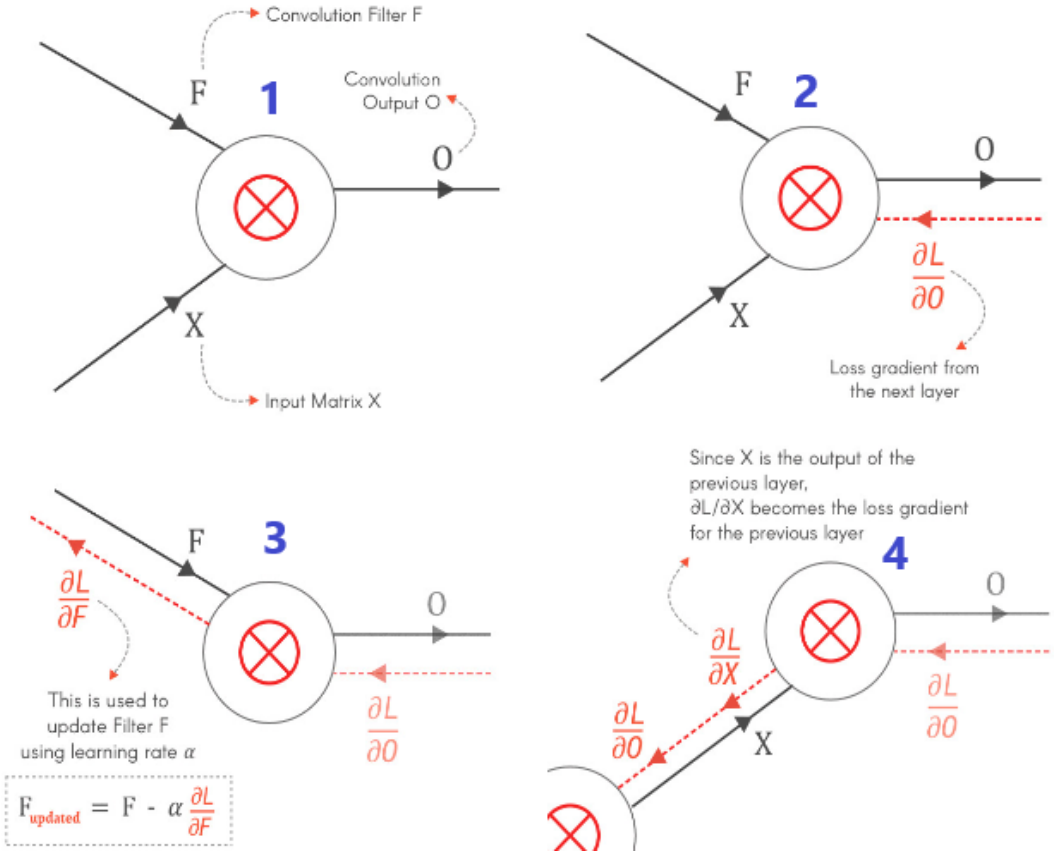


Figura 2.14: Proceso de retropropagación sobre un nodo en una red neuronal

Para el cálculo de $\frac{\partial L}{\partial F}$ se aplica la regla de la cadena: $\frac{\partial L}{\partial F} = \frac{\partial L}{\partial O} \cdot \frac{\partial O}{\partial F}$

2.4. Métricas y evaluación de algoritmos

En la fase de test de un algoritmo de inteligencia artificial, existen diversas métricas que permiten valorar su rendimiento. El elemento empleado por excelencia es la matriz de confusión (Figura 2.15). Una matriz de confusión resume el rendimiento del algoritmo a través de una matriz bidimensional. Cada columna se encuentra asociada a las predicciones sobre una clase C_j mientras que cada fila representa cada clase real C_i. De esta manera la celda N_{ij} representa el número de elementos de la clase C_i que han sido etiquetados por el clasificador como C_j [34].

De esta manera, cada celda N_{ij} de la tabla representa el número de veces que el modelo ha predicho correcta o incorrectamente cada clase y permite ver cómo son las equivocaciones. De hecho, se llama "matriz de confusión" porque hace que sea fácil detectar dónde el sistema está confundiendo dos clases.

		Predicted Values			
Actual Values		C_1	C_2	…	C_n
	C_1	N_{11}	N_{12}	…	N_{1n}
	C_2	N_{21}	N_{22}	…	N_{2n}
	⋮	⋮	⋮	⋮	⋮
	C_n	N_{n1}	N_{n2}	…	N_{nn}

Figura 2.15: Matriz de confusión para clasificación multiclase

Un caso especial de la matriz de confusión se da en los problemas de clasificación binaria, es decir donde sólo existen dos posibles clases: una designada como clase positiva y la otra como clase negativa (Figura 2.16). En este contexto, las cuatro celdas de la matriz se designan directamente como verdaderos positivos (TP), falsos positivos (FP), verdaderos negativos (TN) y falsos negativos (FN) [34]. Un problema de verificación de identidad puede considerarse uno de clasificación binaria puesto que únicamente dos clases son posibles: la persona es quién dice ser o no.

		Predicción	
		Positivos	**Negativos**
Observación	**Positivos**	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Figura 2.16: Matriz de confusión para clasificación binaria

Para evaluar la calidad de un algoritmo de aprendizaje automático, existen distintas métricas que permiten desentrañar su rendimiento según distintos aspectos. Estas métricas pueden ser de rango, umbral o probabilidad, aunque en la práctica las métricas de umbral suelen ser de las más utilizadas para medir el rendimiento de los clasificadores [35]. Por último, aunque la explicación es fácilmente extrapolable para un problema multiclase, las métricas y ecuaciones se plantean bajo el enfoque de un problema de verificación y por ende de clasificación binaria.

2.4.1. Exactitud (Accuracy)

Tanto en problemas de clasificación binaria como en problemas de clasificación multiclase, la exactitud es la métrica de evaluación más utilizada para discriminar y seleccionar la solución óptima. Además, es una métrica sencilla y fácil de calcular [35]. La exactitud representa la relación entre las predicciones correctas sobre el total de predicciones evaluadas y se puede calcular como:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.14}$$

A pesar de ser la métrica más usada, muchas veces es utilizada erróneamente puesto que requiere de unas observaciones balanceadas (es decir el mismo número de cada clase).

2.4.2. Tasa de error (Error rate)

La tasa de error es la métrica complementaria de la exactitud, por lo que la suma de ambos debe ser igual a 1. Mientras que la exactitud mide la relación de las predicciones correctas, la tasa de error mide la proporción de predicciones incorrectas sobre el número total de predicciones evaluadas [35]. El parámetro se puede obtener como:

$$ErrorRate = \frac{FP + FN}{TP + TN + FP + FN} \tag{2.15}$$

2.4.3. Precisión (precision)

Es importante no confundir la precisión con la exactitud. Mientras que la exactitud es una medida más genérica, la precisión se centra en las predicciones positivas. Es decir, la precisión mide las identificaciones positivas que se realizan correctamente sobre el total de predicciones positivas [35],[36]. Traducido al contexto de verificación biométrica, mide el número de equivocaciones sobre los casos en los que la persona era ella misma respecto a los que se ha considerado la misma persona sin serlo. Esta métrica por tanto es muy importante en el ámbito de estudio ya que los falsos positivos son críticos a la hora de verificar la identidad ya que miden las posibilidades de encontrarse con un impostor.

$$Precision = \frac{TP}{TP + FP} \tag{2.16}$$

2.4.4. Exhaustividad (recall)

La exhaustividad, al igual que la precisión se centra en las predicciones positivas. Pero en este caso resuelve la pregunta de cuántos individuos pertenecientes a una

clase fueron identificados correctamente respecto al total de individuos pertenecientes a esa clase [36]. En el ámbito de verificación denota el número de equivocaciones sobre los casos en los que la persona era ella misma respecto a los que se ha considerado una persona distinta sin serlo. Es evidente por tanto que la exhaustividad es una medida mucho menos relevante que la precisión para el caso de estudio. Resulta un inconveniente mucho mayor que alguien acceda a un teléfono móvil ajeno a través del sistema biométrico de otro (FP) a que simplemente se deba aplicar de nuevo el método biométrico porque simplemente no se ha detectado al propietario correctamente (FN).

En algunos documentos, se puede encontrar la palabra "sensitivity" o "True Positive Rate (TPR)" para definir el "recall" puesto que ambos términos son aceptados [37]. El parámetro se puede modelar como se expone a continuación:

$$Recall = \frac{TP}{TP + FN} \tag{2.17}$$

2.4.5. Especificidad (Specificity)

Por otro lado, la especificidad o tasa negativa verdadera (TNR) se centra en las predicciones negativas. Aplicado al caso de estudio, la pregunta que la especificidad responde sería la siguiente: ¿Cuántos individuos impostores son identificados correctamente?

Al igual que ocurre con la precisión, es importante que la especificidad sea un valor alto puesto que representa directamente la seguridad del sistema de identificación (al depender de nuevo de los falsos positivos). Si el valor de la especificidad fuese 1, querría decir que todos los impostores son detectados correctamente [35]. Este valor se puede calcular a través de la siguiente fórmula:

$$Specificity = \frac{TN}{TN + FP} \tag{2.18}$$

2.4.6. Valor F1 (F1 Score)

Sería deseable que el modelo presentase unos valores de precisión y exhaustividad elevados, pero por desgracia estos valores normalmente se encuentran en tensión, es decir, mejorar la precisión típicamente reduce la exhaustividad y viceversa. Ante esta problemática, surge el valor-F que representa la media armónica entre los valores de precisión y exhaustividad. Al igual que la exactitud, el valor-F, reporta buenos resultados como discriminador, aunque no se suele utilizar para problemas de clasificación multiclase [35]. El valor-F se puede calcular de la siguiente manera:

$$Valor - F = \frac{2 \cdot Precision \cdot recall}{Precision + recall} \tag{2.19}$$

2.4.7. Curva ROC y área bajo la curva (AUC)

En muchos casos, la salida del algoritmo en lugar de ser directamente una predicción, es una probabilidad de pertenecer a una clase. En esos casos, distintas predicciones y por ende distintas matrices de confusión se pueden lograr según la elección de un umbral que permita distinguir las predicciones positivas de las negativas.

La curva ROC (Receiver Operating Characteristic) es un gráfico que permite medir el rendimiento del modelo de clasificación utilizando distintos umbrales. Para ello, se trazan dos variables a lo largo de la curva para diferentes valores del umbral: la exhaustividad o tasa de verdaderos positivos (TPR) y a la tasa de falsos positivos (FPR) [38],[39].

La tasa de verdaderos positivos corresponde con la exhaustividad (como se ha comentado en la sección 2.4.4) y la FPR se puede calcular de la siguiente forma:

$$FPR = 1 - specificity = \frac{FP}{TP + FN} \tag{2.20}$$

A continuación se muestra un ejemplo de curva ROC

En el dibujo a continuación (Figura 2.17) es posible observar la curva ROC de diversos modelos representados en distintos colores. Un clasificador aleatorio (que asigne 0 o 1 al azar) sería representado en color rojo. Según se desplaza la curva hacia la esquina superior izquierda del gráfico, la calidad del modelo va aumentando hasta llegar al modelo perfecto que se encuentra representado en color morado. Esto se debe a que el modelo mejora en su tasa de verdaderos positivos, minimizando también la tasa de falsos positivos.

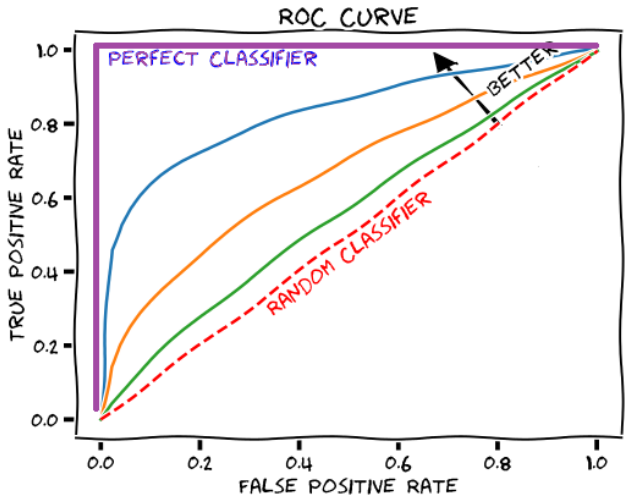


Figura 2.17: Ejemplo de curva ROC

Obteniendo el área bajo la curva ROC se obtiene el parámetro AUC. Este resume la curva ROC e indica la capacidad del clasificador para distinguir distintas clases. De esta manera, se considera que el modelo de aprendizaje automático discrimina bien entre

clases positivas y negativas a medida que aumenta el AUC [38]. Así pues, observando la gráfica anterior (Figura 2.17) el clasificador aleatorio (en rojo) tendría un AUC de 0.5 mientras que con el clasificador perfecto (en morado) se obtendría un AUC de 1.

2.4.8. Curva PR (Precision-Recall)

La abundancia de ejemplos negativos es un fenómeno común en muchos casos de aprendizaje automático.

Por ejemplo en modelos de detección de infección por COVID-19 los datos empleados cuentan con mayor número de casos negativos o en el ámbito web la mayoría de las páginas son irrelevantes para la mayoría de las consultas. En estas situaciones la precisión no es una medida de clasificación sensata ya que sobrevalora el clasificador siempre negativo [40].

Este desbalance desemboca en que nuestro interés principal se enfoque en la capacidad del modelo para predecir la clase minoritaria (clase 1). De este modo, apenas interesa el rendimiento del modelo en la clase mayoritaria (clase 0) por lo que no se ha de prestar demasiada atención a los verdaderos negativos. Una buena solución en este caso es ignorar por completo los verdaderos negativos y utilizar métricas como la precisión en lugar de la FPR [40].

Tal vez motivados por el atractivo de los gráficos ROC, muchos investigadores han empezado a elaborar gráficos Precision-Recall o PR representando la precisión en el eje "Y" frente a la exhaustividad en el eje "X" . Esto da como resultado una curva que tan solo tiene en cuenta la clase positiva [40].

A continuación, se muestra un ejemplo de la curva para una serie de modelos, cada uno con un poder predictivo distinto (Figura 2.18). El modelo representado en morado corresponde a un clasificador perfecto, mientras que las curvas más alejadas de esa esquina representan modelos peores.

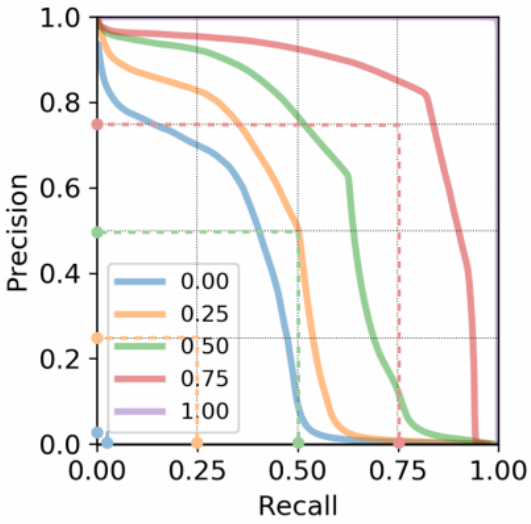


Figura 2.18: Ejemplo de curva precision-recall

2.5. Herramientas de desarrollo

Para desarrollar la algorítmica del proyecto ha sido imprescindible el uso de principalmente estas tres herramientas:

- Entorno de desarrollo LabVIEW: empleado tanto en adquisición como en procesamiento de datos (Sección 2.5.1).
- Lenguaje de programación python: lenguaje sobre el que se desarrolla el algoritmo de verificación (Sección 2.5.2).
- Servicio de nube "Google colab": empleado en la ejecución y pruebas del algoritmo (Sección 2.5.3).

2.5.1. Entorno de desarrollo LabVIEW

Jeff Kodosky, cofundador de NI (National Instruments) en 1976, es considerado como el padre de LabVIEW por científicos de todo el mundo [41]. Cuando en 1986 se lanzó la primera versión de LabVIEW, fue anunciada como una manera no programática de crear software para pruebas y medidas. Sin embargo, detrás de esto, existía un lenguaje de programación gráfica riguroso y bien definido conocido como "G" [42].

LabVIEW es un entorno de programación gráfico, es decir, se representa como un diagrama bidimensional formado por nodos y cables interconectados entre sí. El flujo de datos indica la semántica del lenguaje. Es decir, los nodos realizan cálculos basados en sus valores de entrada y sólo cuando se completan propagan sus valores de salida hacia los nodos posteriores. De esta manera, cada cable transporta un único dato que puede ser de distintos tipos (entero, flotante, cadena de caracteres, booleano...) [42].

Cada módulo de software en LabVIEW/G recibe el nombre de instrumento virtual o "VI". Al abrir un VI, aparecen dos ventanas (Figura 2.19): el panel frontal y el diagrama de bloques. El panel frontal corresponde a la interfaz de usuario mientras que el diagrama de bloques contiene todos los nodos y cables que transfieren datos intercomunicándose con otros objetos del diagrama de bloques [43].

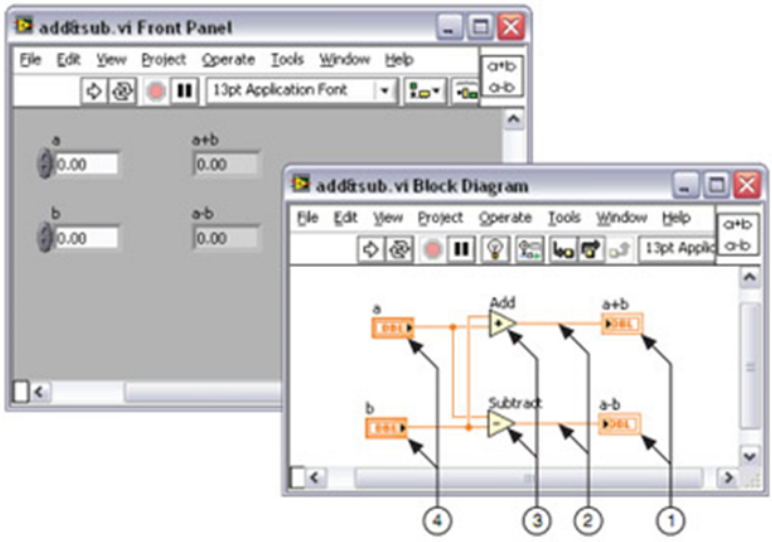


Figura 2.19: Panel frontal y diagrama de bloques [43]

Existen numerosos tipos de nodos. Los terminales de indicador y control (1 y 4 en la imagen superior) permiten introducir y visualizar datos a través del panel frontal. También puede haber funciones, constantes, estructuras o subVIs. Estos últimos corresponde a una llamada de subrutina y permiten ejecutar un VI dentro de otro [43].

A través de las distintas paletas de Controles y Funciones, se puede acceder a los distintos módulos previamente instalados en el PC [43]. Por ejemplo, en la categoría "Optimization VIs" se puede acceder a varias funciones de optimización. En la imagen inferior (Figura 2.20), se puede ver la paleta de controles con la categoría "Moderno" desplegada.

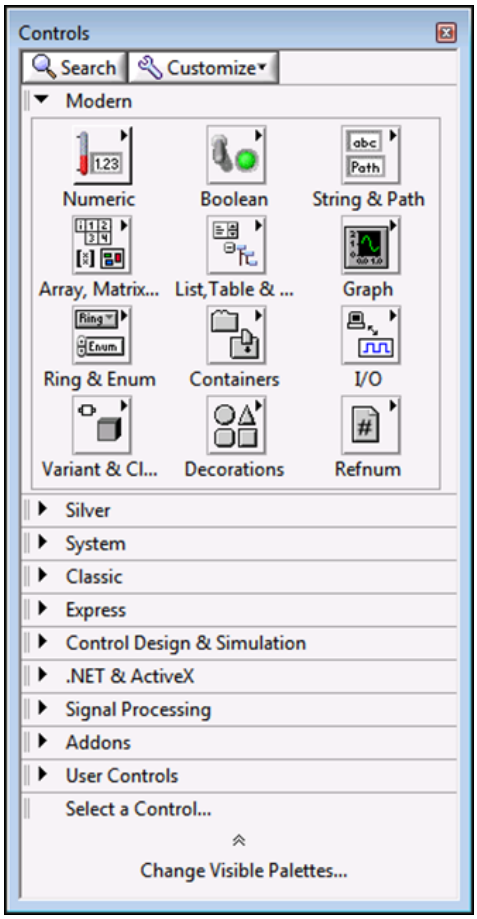


Figura 2.20: Paleta de controles

2.5.2. Lenguaje de programación Python

Python es un lenguaje de programación de uso general que fue concebido originalmente en la década de 1980 por Guido Van Rossum y ha ganado interés en los últimos años. La capacidad de python de ejecutarse fácilmente en (casi) cualquier sistema operativo, disposición de amplia gama de bibliotecas y carácter de software libre son alguna de las razones por las cuales este lenguaje se ha popularizado [44].

Ya sea en el campo de aprendizaje automático como más específicamente en aprendizaje profundo, python es la elección preferida debido a bibliotecas líderes como PyTorch [45], scikit-learn [46] o TensorFlow [47], que facilitan la construcción y el entrenamiento de modelos de aprendizaje automático y redes neuronales profundas.

Por último, aunque no estén orientadas directamente al ámbito de inteligencia artificial, otras librerias como "NumPy" (Numerical Pyton), "Pandas" deben ser mencionadas puesto que son fundamentales para el cálculo numérico o análisis y manipulación de datos estructurados.

2.5.3. Google Colab

Google Colaboratory o "Colab" para abreviar, es un producto de "Google Research". "Colab" permite escribir y ejecutar código python a través de un entorno de "jupyter notebook" (Figura 2.21). Este servicio es idóneo para ejecutar modelos de aprendizaje automático puesto que, a través de GPUs o TPUs disponibles, permite cubrir la demanda de carga computacional que estos modelos suelen exigir [48].

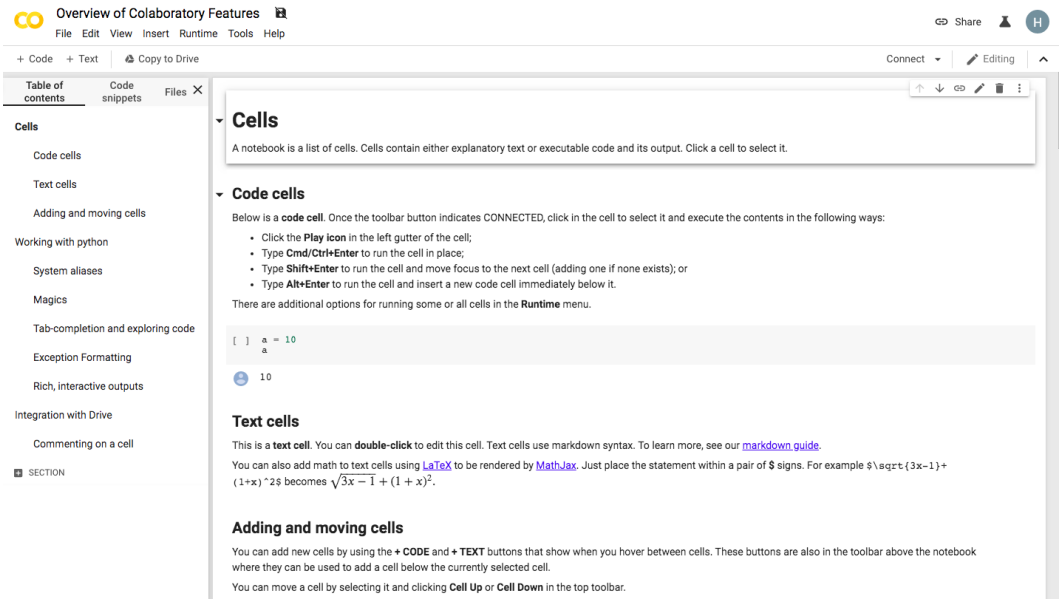


Figura 2.21: Entorno de ejecución en Google Colab

Una funcionalidad importante de "Google colab" es que permite montar el sistema de ficheros de Google Drive localmente en la máquina en la que se ejecuta el código. Esto es una enorme ventaja puesto que al trabajar con *machine learning* o aplicaciones de ciencia de datos, en la mayoría de las ocasiones se cuenta con enormes archivos por lo que si fuese necesario subirlos cada vez que se conecta con un entorno distinto de ejecución, sería una pérdida de tiempo crítica. De este modo, basta con subir estos archivos a "Drive" y montarlo en la máquina para acceder a ellos como si estuvieran en local [48].

El servicio "Colab" cuenta con las principales librerías de python preinstaladas (como numpy, pandas, scikit-learn o pytorch) [48]. No obstante, además de código python, el entorno permite intercalar celdas de marcado para crear documentación o explicaciones y ejecutar comandos de shell utilizando el símbolo "!" antes de la linea .

Como detalle final, cabe mencionar que el entorno de ejecución permite escoger entre distintas opciones como acelerador por hardware [48]:

- CPU (Central Processing Unit): Son unidades de procesamiento de propósito general
- GPU (Graphics Processing Unit): Diseñadas para procesar muchos datos simultáneamente lo que las convierte en una opción útil en aplicaciones de aprendizaje automático.

- TPU (Tensor Processing Unit): Diseñadas especialmente para acelerar operaciones de álgebra lineal y manipulación de tensores, que son fundamentales en aplicaciones de *machine learning* y de *deep learning*.

Capítulo 3. Adquisición de imágenes

Cómo se ha mencionado en la introducción, el diseño del sistema de adquisición es uno de los pilares fundamentales del proyecto.

3.1. Sistema de adquisición hardware

El sistema (Figura 3.1) esta formado fundamentalmente por 3 fuentes de adquisición: el sensor LiDAR (señalado con una flecha amarilla), una cámara óptica (señalada con una flecha roja) y el sistema acústico (a su vez compuesto por distintos dispositivos que se explican más adelante).

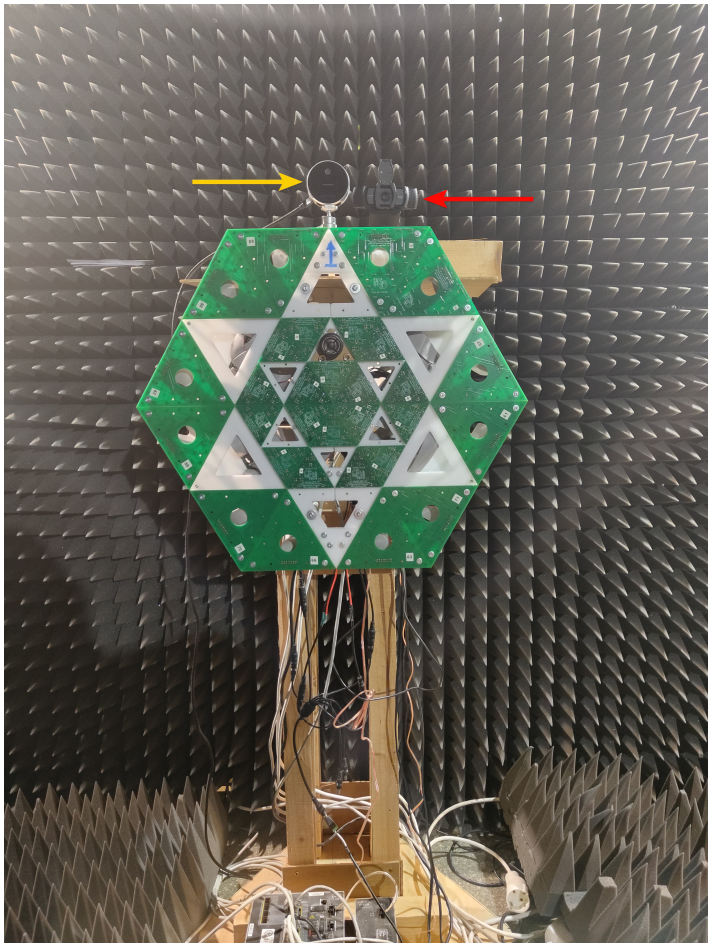


Figura 3.1: Sistema de adquisición

En este proyecto, no se utilizan imágenes ópticas por lo que la cámara óptica únicamente se utiliza como referencia o soporte en caso de algún problema en la etapa de

procesado o análisis.

3.1.1. Sensor LiDAR

En el proyecto se ha empleado la cámara LiDAR de Intel modelo L515 [49]. Esta cámara diseñada para aplicaciones interiores cuenta con un alcance desde 0.25 hasta 9 metros en condiciones ideales de reflectividad. El dispositivo (Figura 3.2) cuenta con un trípode que además de servir de soporte para la cámara, permite rotarla fácilmente.

Figura 3.2: cámara Intel modelo L515

La conectividad del sensor se realiza a través de una entrada USB-C 3.1 Gen 1 que se emplea para transmitir los datos capturados mediante múltiples *streamming*. Dispone de unidad de medición inercial o IMU (Inertial Measurement Unit), cámara RGB y LiDAR, aunque el estudio se ha centrado en esta última.

El láser emite una longitud de onda λ entre 844nm y 875nm en el rango de temperaturas de (0 °C - 50 °C) [49]. Es decir, superior al rango de longitudes de onda detectado por el ojo humano en el espectro electromagnético (380 nm a 760 nm).

La cámara LiDAR cuenta con un campo de visión o FOV (Field of View) de 70 ° en horizontal frente a 55 ° en vertical, por lo que cubre mayores distancias horizontales.

Por otro lado, la resolución angular depende de la resolución de *streaming* escogida: QVGA (320 x 240), VGA (640 x 480) y XGA (1024 x 768) [49]. El sensor proporciona por tanto distintos tamaños de imágenes 2D donde en cada píxel se incluye información

acerca de la distancia al blanco. Así pues, tomando por ejemplo una resolución QVGA y denotando el azimut como θ y la elevación como φ, es posible calcular la resolución angular proporcionada por el sensor en ambas coordenadas como:

$$\Delta\theta = \frac{70\,^\circ}{320px} \approx 0{,}219^\circ/px \tag{3.1}$$

$$\Delta\varphi = \frac{55\,^\circ}{240px} \approx 0{,}229^\circ/px \tag{3.2}$$

Fijada la resolución angular en azimut $\Delta\theta$ y elevación $\Delta\varphi$, el cálculo de la resolución horizontal Δx y vertical Δz resulta trivial (Ecuaciones 3.3 y 3.4). Eso sí, es preciso recordar que la resolución espacial en coordenadas cartesianas depende de la distancia a la que se encuentre el blanco d (Figura 3.3).

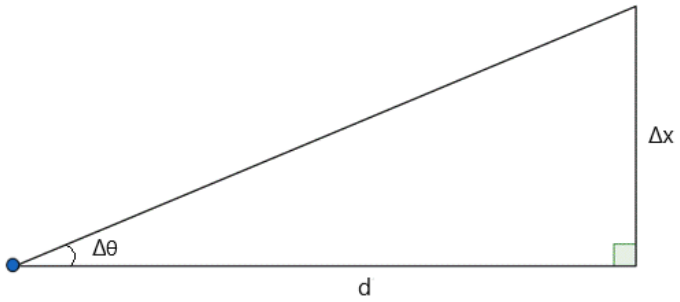


Figura 3.3: Diagrama para el cálculo de la resolución espacial

$$\tan(\Delta\theta) = \frac{\Delta x}{d} \Rightarrow \Delta x = d \cdot \tan(\Delta\theta) \tag{3.3}$$

$$\tan(\Delta\varphi) = \frac{\Delta z}{d} \Rightarrow \Delta z = d \cdot \tan(\Delta\varphi) \tag{3.4}$$

A modo de ejemplo, las resoluciones manejadas, para QVGA y un blanco situado a 2,2 metros (distancia típica empleada en las capturas), se tiene que $\Delta x \approx 0{,}841cm$ y $\Delta z \approx 0{,}879cm$.

Respecto a la resolución en el eje Δy, el fabricante especifica un error en la precisión medio menor a 1.4cm y una desviación estándar de 1.55cm para blancos situados hasta 9 metros empleando resolución VGA y suponiendo un 95 % de reflectividad [49].

De esta forma se asume que la resolución proporcionada por la cámara LiDAR será suficiente para el objetivo planteado.

Finalmente, aunque el dispositivo permite una velocidad de fotograma o *frame rate* de hasta 60fps para el streaming proporcionado por la camara RGB, este valor es de 30fps como máximo para el caso de la cámara de profundidad LiDAR [49].

3.1.2. Sistema acústico

Tweeter

Un altavoz especializado en altas frecuencias (*tweeter*) es el encargado de generar la señal acústica a la frecuencia deseada. Este altavoz transmite los pulsos acústicos según el ancho de pulso y la frecuencia configurada en el software.

El hecho de utilizar un único altavoz implica que no se practica ningún tipo de conformado en transmisión. Utilizar N altavoces permitiría utilizar un pulso con una intensidad acústica N veces mayor. De esta manera, el sistema sería capaz de concentrar mucha energía en un punto, pero esto obligaría a utilizar mucho más tiempo para escanear una zona del espacio ya que habría que ir apuntando uno a uno con el array de altavoces a cada punto del espacio. Por el contrario, el conformado aplicado en recepción permite muestrear una única vez a todos los sensores en paralelo por lo que no aumenta el tiempo de adquisición.

Para este proyecto, se ha utilizado un *tweeter* de 32 mm de diámetro ubicado lo más cerca posible del centro del array (Figura 3.4).

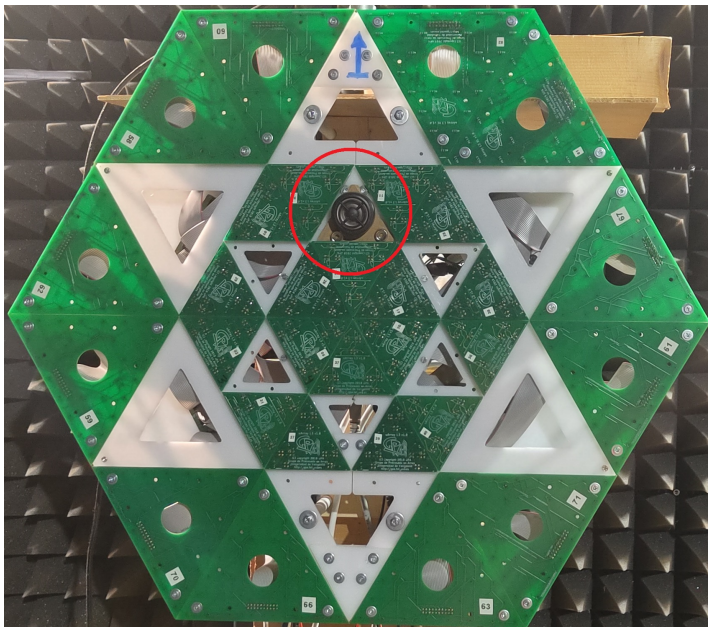


Figura 3.4: Ubicación del *Tweeter* en el sistema acústico

La motivación de colocar la fuente acústica lo más cerca del centro del array nace del concepto de la diferencia de caminos entre la onda acústica incidente y reflejada. Aunque se podría tener en cuenta, el software de adquisición estima la distancia en rango a la que se encuentra el blanco como $d/2$, siendo d el total de distancia recorrida por la onda. Es decir, $d = d_1 + d_2$ pero $d_1 \neq d_2$ (Figura 3.5).

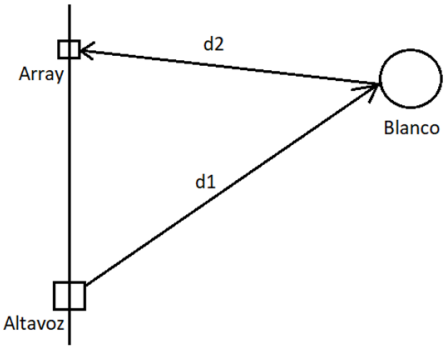


Figura 3.5: Diferencia de caminos entre las ondas incidente y reflejada

Sin embargo, la arquitectura del array imposibilita colocar el *tweeter* en el centro del array por lo que, un pequeño error producido por este aspecto se arrastra en la capturas realizadas.

MEMS Array

El sistema encargado de capturar los ecos acústicos reflejados consiste en un array hexagonal plano formado por 810 micrófonos MEMS que se encuentran espaciados en 6 sectores siguiendo una disposición de tipo fractal (Figura 3.6).

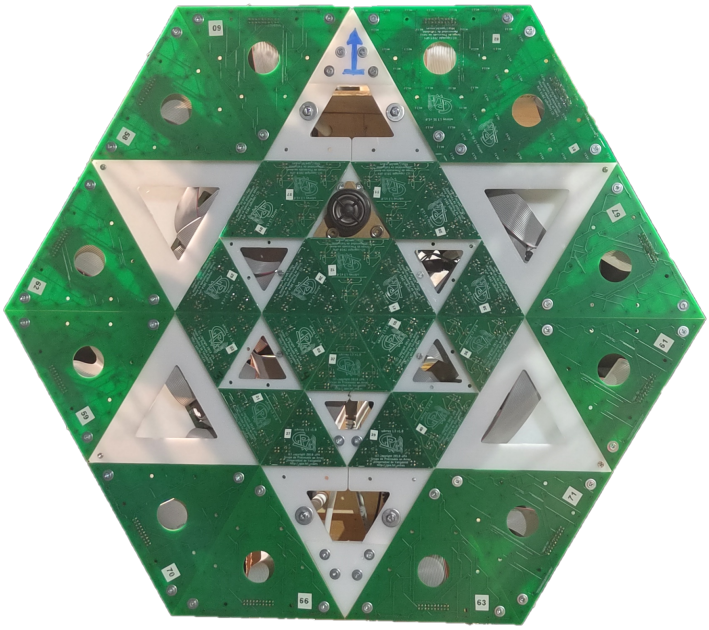


Figura 3.6: Diferencia de caminos entre las ondas incidente y reflejada

La tecnología geométrica fractal es en esencia un patrón infinito y se emplea en números ámbitos de la ciencia y pueden ser de carácter aleatorio o determinista [50]. Uno de los fractales deterministas más populares es el triángulo de Sierpinski (Figura 3.7).

Figura 3.7: Triángulo de Sierpinski

En el ámbito de las antenas, el comportamiento multibanda asociado a la forma fractal fue introducido por C.P. Baliarda donde se evaluó el desempeño de antenas monopolo de Sierpinski para resonar en distintas bandas frecuenciales. De hecho, además de geometrías como "Koch Island" o "Minkowski", uno de los mejores ejemplos de antenas de bucle fractal es la geometría que constituye el array: hexagonal con fractales triangulares [50].

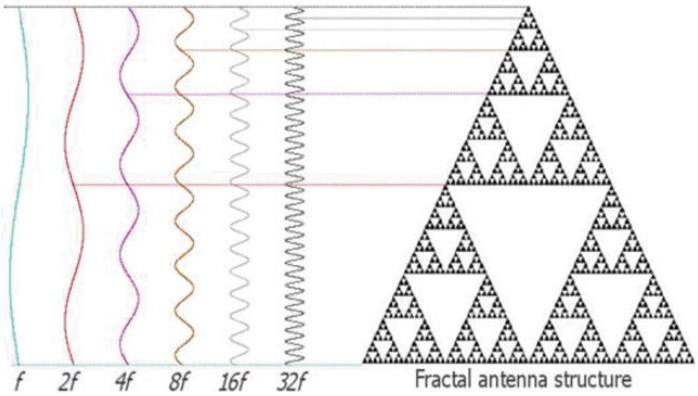


Figura 3.8: Respuesta frecuencial logarítmica del triángulo de Sierpinski [50]

Los micrófonos digitales MEMS (Micro-Electro-Mechanical System) están compuestos por dos componentes principales, el sensor acústico y el circuito controlador que incluirá un convertidor ADC. A parte de ser una tecnología de bajo coste, los sensores MEMS reducen significativamente el área ocupada por el sistema lo que hace posible construir arrays con un mayor número de sensores [51], [52].

El sensor normalmente es un elemento capacitivo formado por dos placas de silicio, una fija, llamada backplate y otra flexible, llamada diaphragm. Los cambios de presión en el aire generados por las ondas acústicas hacen moverse a la membrana flexible. Cuando esto ocurre, la capacitancia entre la placa móvil y la fija cambia y el circuito controlador mide los valores obtenidos [52]. Los micrófonos MEMS se encapsulan para proteger el circuito de agentes externos y permitir a las ondas de sonido llegar al sensor

acústico sólo a través de la apertura en la parte superior o inferior del sensor [52]. En concreto, para este proyecto se han utilizado los micrófonos MEMS MP34DT01 [53].

Para finalizar, dada la simetría radial del array presentado, se obtiene un diagrama de radiación simétrico. De este modo, el sistema proporciona la misma resolución en ambas coordenadas vertical y horizontal.

Sistema de adquisición y procesamiento acústico

La clave del sistema hardware de adquisición de imágenes acústicas es el sistema de NI sbRIO 9607 (Figura 3.9). Esta plataforma pertenece a la familia de dispositivos reconfigurables Entrada-Salida o RIO (Reconfigurable Input-Output) de NI [54].

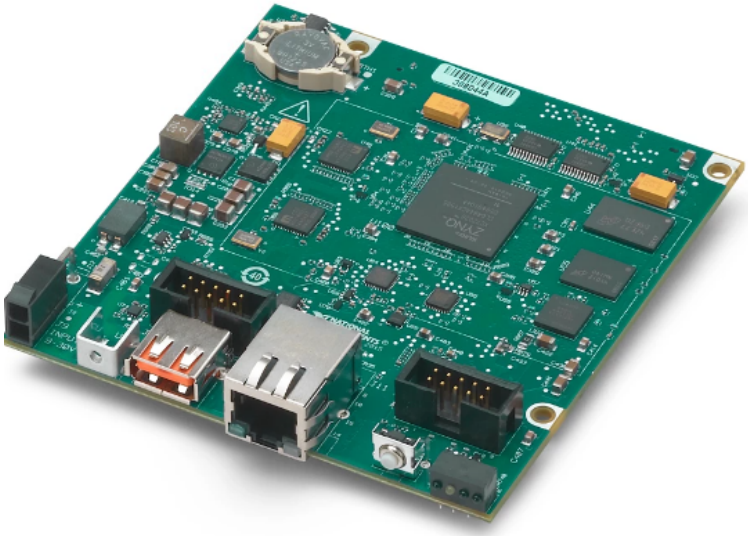


Figura 3.9: sbRIO 9607

El sbRIO-9607 es un controlador embebido que integra un procesador dual-core ARM Cortex-A9 de 667 MHz y una FPGA Xilinx Zynq-7000 reconfigurable de 96 líneas de entrada/salida digitales. Ambos dispositivos se comunican entre sí a través de un bus de datos en el interior del chip [54].

El procesador ARM es capaz de ejecutar a la vez todos los algoritmos software para generar las señales acústicas. Sin embargo, aunque sbRIO podría funcionar como un sistema aislado, al no contar con un sistema de visualización y necesitarse en este caso del uso de más de una plataforma sbRIO, se ha centralizado el sistema desde un PC (Figura 3.10).

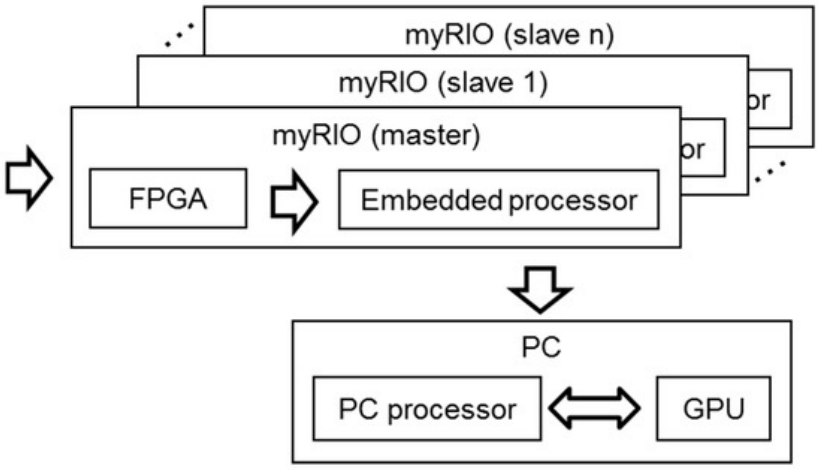
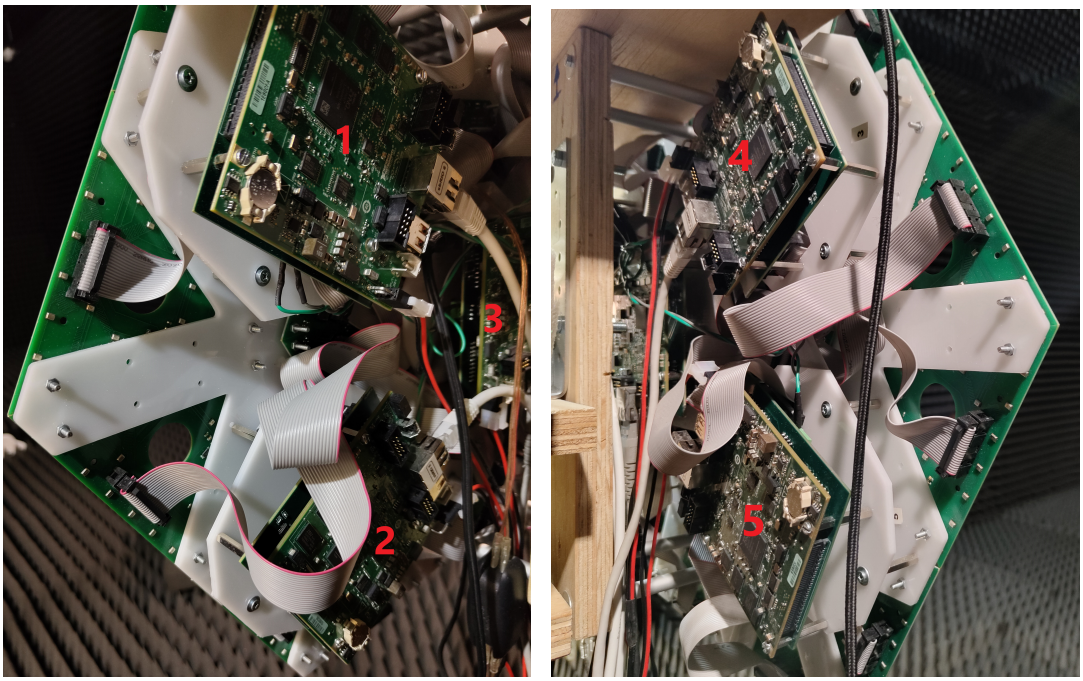


Figura 3.10: Esquema a alto nivel del sistema de procesado

Se reservan algunas líneas E/S para generar el reloj y sincronización mientras que el resto se utilizan para conectar los micrófonos MEMS. Sin embargo, como el número de entradas digitales con las que cuenta la FPGA es menor que el número de micrófonos, se permite en cada canal recoger la información de dos sensores (uno en el flanco de subida del reloj y otro en el flanco de bajada).

El procesador ARM cuenta con un puerto Giga Ethernet que permite realizar la comunicación E/S del chip a través de un conmutador al que se ha de conectar previamente. Así, el procesador que funciona en Linux tiempo real descarga en la FPGA el código que se va a ejecutar y la FPGA lee todos los canales en paralelo y de manera síncrona.

En este proyecto, para garantizar la captura síncrona de los 810 micrófonos del array, se han utilizado 5 tarjetas sBRIO conectadas a los distintos micrófonos del array (Figura 3.11).



(a) Parte izquierda (b) Parte derecha

Figura 3.11: 5 tarjetas sBRIO conectadas al array acústico

Las sBRIO se interconectan entre sí a través de un conmutador (Figura 3.12) que a su vez se conecta con el ordenador que se utiliza como interfaz de usuario y unidad de procesamiento.

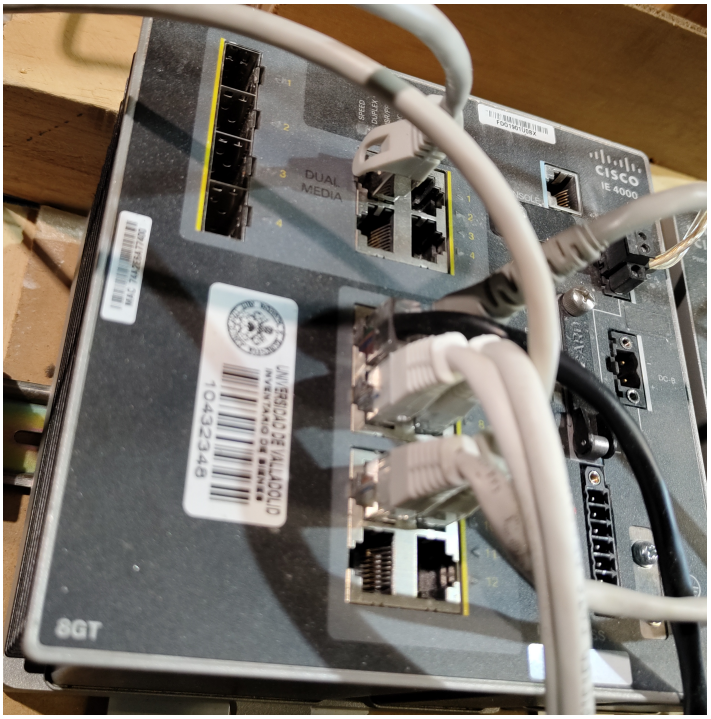


Figura 3.12: Conexiones en el router CISCO

Por otra parte, el *tweeter* necesita de una señal analógica para generar el pulso acústico. Para solucionar esto, la FPGA cuenta con un conversor DAC a través del cual se genera el pulso de transmisión. Una vez generado, la sbRIO se comunica con un amplificador externo de audio (Figura 3.13). El amplificador se conecta al tweeter ajustando la impedancia y generando una potencia de audio significativa para transmitir la señal.

Figura 3.13: Amplificador externo

3.2. Sistema de adquisición software

El software de adquisición ha sido enteramente desarrollado en LabVIEW. El sistema debe ser capaz de capturar de forma síncrona las señales acústicas y la imagen LiDAR y guardarlas en disco para su posterior procesado.

3.2.1. Adquisición de información acústica y LiDAR

Para la parte acústica, se han utilizado las librerías desarrolladas por el equipo del GPA (Grupo de Procesado en Array) de la Universidad de Valladolid. A pesar de que el grupo cuenta con un software propio de captura, visualización y procesado de imágenes acústicas, ha sido preciso diseñar una nueva aplicación de captura que integre de forma simlutánea el sensor LiDAR y el array acústico.

En lo que respecta a la adquisición de señales acústicas, se procede de acuerdo a la siguiente secuencia:

1. Se leen 2 ficheros previamente configurados que contienen el array activo y los parámetros del pulso transmitido.
2. Se configura el pulso en los DAC de la sbRIO.
3. Se transmite el pulso y se capturan las N señales provenientes de los N sensores.
4. Finalmente se genera un ultimo fichero que almacena las señales capturadas y los parámetros proporcionados.

Por otra parte, la adquisición LiDAR ha sido un problema más complejo de resolver. Intel proporciona un sistema de visionado "Intel RealSense Viewer" (Figura 3.14) para toda su gama de productos, lo que proporciona al usuario una herramienta sencilla para evaluar el funcionamiento del dispositivo. Aunque el manual [55] está diseñado para las series D400 y SR300, el funcionamiento es análogo para el modelo L515.

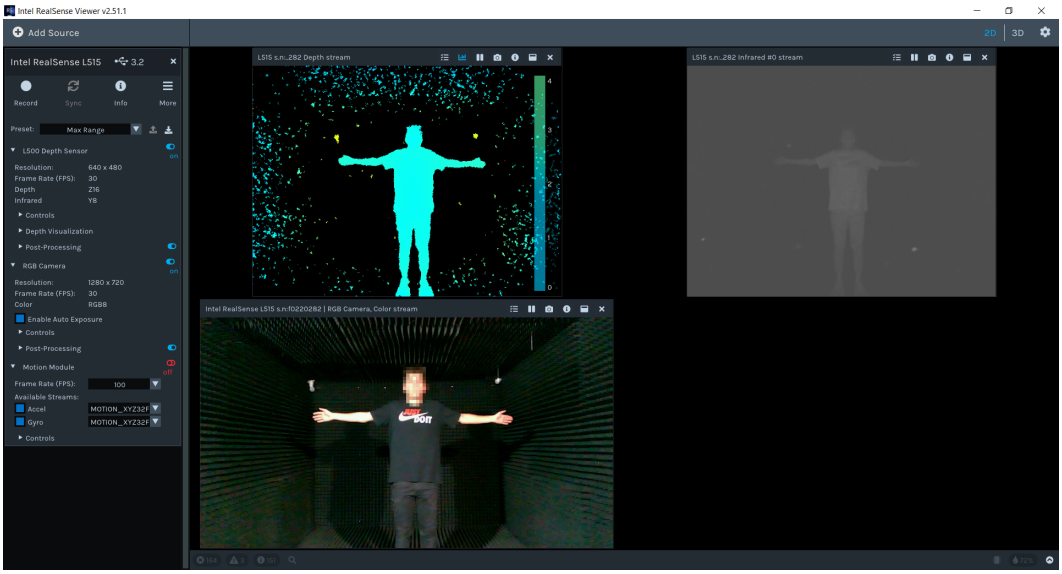


Figura 3.14: Intel RealSense Viewer Software

El software permite grabar archivos de vídeo en formato propietario para seleccionarlos más tarde como fuente de entrada y así reproducir de nuevo la grabación o aplicar cualquier postprocesado [55].

No obstante, esta funcionalidad no ha sido considerada para implementar la adquisición en el proyecto por dos razones principales:

1. No se obtiene un archivo de datos "en crudo" que sea facilmente manipulable.
2. No es posible controlar la captura síncrona de la imagen LiDAR y acústica.

Por otra parte, Intel cuenta con algunos wrappers software en distintos lenguajes de programación para algunos de sus modelos, sin embargo aunque si existen en LabVIEW para el modelo D400, no se encuentran desarrolladas para el modelo L515 por lo que ha sido necesaria su implementación para el proyecto.

Diseñadas las funciones de adquisición, a grandes rasgos, el mecanismo de captura LiDAR es el siguiente:

1. Inicializar el dispositivo LiDAR seleccionando el sensor (L500 Depth o cámara RGB) y el dispositivo (en caso de que hubiese más de uno conectado)
2. Tras configurar el *streamming* deseado sobre el dispositivo escogido, se selecciona y se inicia una cola para recibir los *frames* envíados por el LiDAR.
3. Se espera a recibir cada *frame* y una vez recibido, se consulta el ID de *streamming* para identificar a cuál pertenece y se recoge el identificador del fotograma.
4. A partir del identificador de fotograma, se obtiene un puntero a los datos de la imagen.
5. Se copia el contenido de memoria de la dirección apuntada por el puntero a un array 2D que se multiplica por la escala de profundidad para obtener los valores en las unidades correctas.

6. Se libera de la memoria el puntero al fotograma preparando la obtención del siguiente y el proceso continúa desde el paso 3.
7. Finalización del *streamming* y liberación de memoria .

Un factor muy importante a tener en cuenta es que la escala de profundidad es diferente en los productos de la serie D400 y la cámara L515 como se comenta en el apartado *issues* correspondiente al enlace git oficial de IntelRealSense [56]. Mientras que para modelos como el D435 es de 0.001, el factor para obtener la salida de la cámara L515 en metros es de 0.00025.

3.2.2. Adquisición síncrona

Como se ha comentado, la adquisición síncrona es el problema principal a resolver en el desarrollo del software de adquisición. Para que el sistema sea eficiente y pueda capturarse un número significativo de imágenes por segundo, la inicialización de los 2 subsistemas debe hacerse una única vez durante todo el proceso de captura (como ocurre típicamente en un sistema síncrono).

La solución planteada consiste en la ejecución de dos bucles paralelos y sincronizados. De este modo, es posible capturar una imagen acústica y asociada a ella, obtener la imagen del *streamming* más próxima. Por último, al estar conectado el LiDAR directamente al PC, no existen retardos provocados por el paso a través de nodos en la red que dificulten la sincronización.

De este modo, el sistema final desarrollado permite la captura síncrona de aproximadamente 2000 muestras en 12 minutos lo que supone una media de $2,\widehat{7}$ capturas por segundo.

3.3. Consideraciones de adquisición

3.3.1. Escenario físico

Para simplificar el escenario de procesado y tener en cuenta únicamente las reflexiones acústicas producidas por el individuo "capturado", el sistema de adquisición se encuentra dentro de una cámara anecoica (Figura 3.15). Esto permite absorber casi la totalidad de las reflexiones producidas por las ondas acústicas en las paredes, suelo y techo de la sala. Para reducir las reflexiones acústicas, la cámara anecoica se ha construido a partir de cuñas piramidales hechas de espumas cuyo coeficiente de absorción acústica (directamente proporcional a la absorción del material) es elevado.

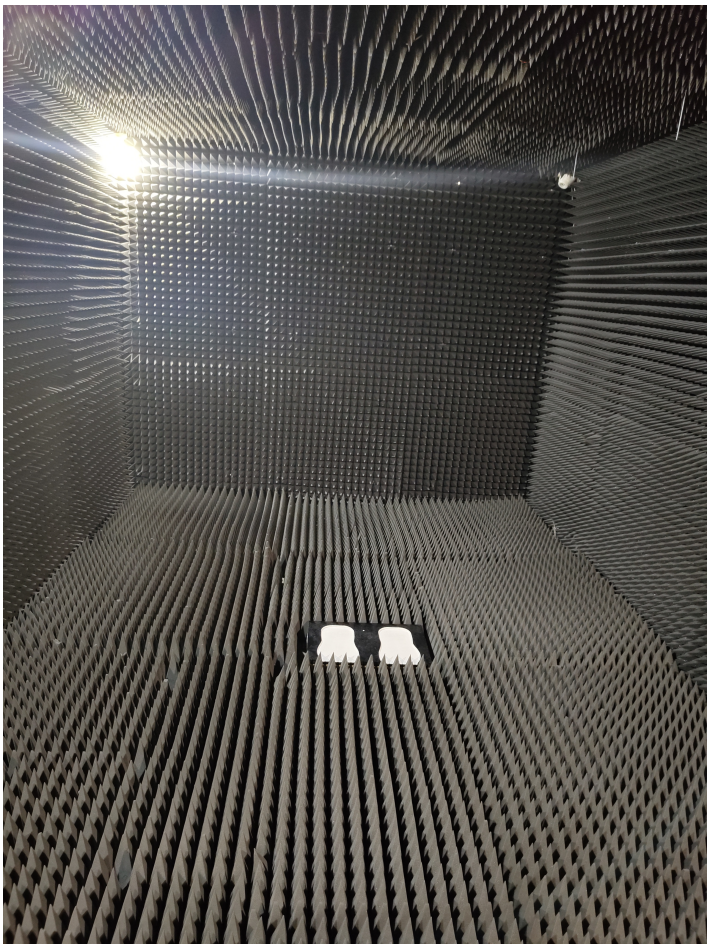


Figura 3.15: Espacio de adquisición

Además como se puede observar en la imagen anterior (Figura 3.15), se ha construido un pedestal con unas huellas de referencia para elevar el sujeto sobre el suelo de la cámara y garantizar una posición más estable en el conjunto de capturas realizadas.

Otro tema importante es la posición física del sistema de adquisición. En concreto, la altura del centro del array respecto al suelo si es notablemente significativa, ya que repercute directamente en el espacio capturado en la imagen. Se ha colocado el array a una distancia de aproximadamente 90 centímetros de altura respecto del pedestal permitiendo así centrar en elevación al sujeto promedio.

3.3.2. Posición del individuo

La captura de los rasgos antropométricos más significativos depende notablemente de la postura del individuo a la hora de ser capturado mediante el sistema acústico. Por ejemplo, la cantidad de información antropométrica obtenida si el individuo extiende los brazos no es comparable a la que se obtiene si los esconde detrás de la espalda. En estudios anteriores [2], se consideraron cuatro posibles posiciones (Figura 3.16) sobre las que se ha partido en esta ocasión.

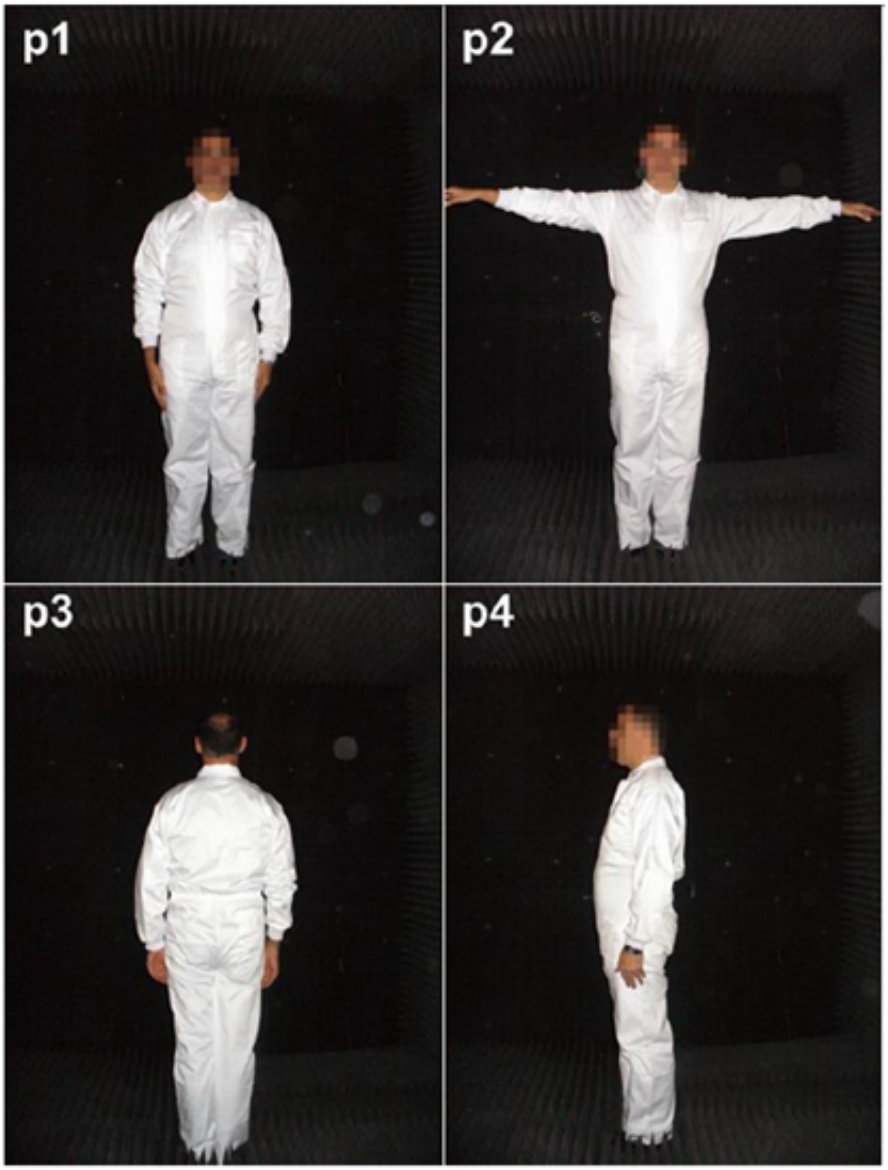


Figura 3.16: Posiciones para adquisición planteadas en anteriores estudios [2]

En lugar de capturar 4 posiciones diferentes, se ha utilizado una sola que proporcione suficiente información biométrica con el objetivo de reducir el tiempo de adquisición de las imágenes.

La posición empleada en las capturas (Figura 3.17) se ha decidido partiendo de la posición (p2) en base a unas pautas establecidas para la colocación de las extremidades del individuo. En esencia son:

1. Separar ligeramente más las piernas.
2. Colocar las palmas abiertas apuntando al sistema de adquisición incrementando así su detección al ser mayor la superficie sobre la que incide la señal acústica.

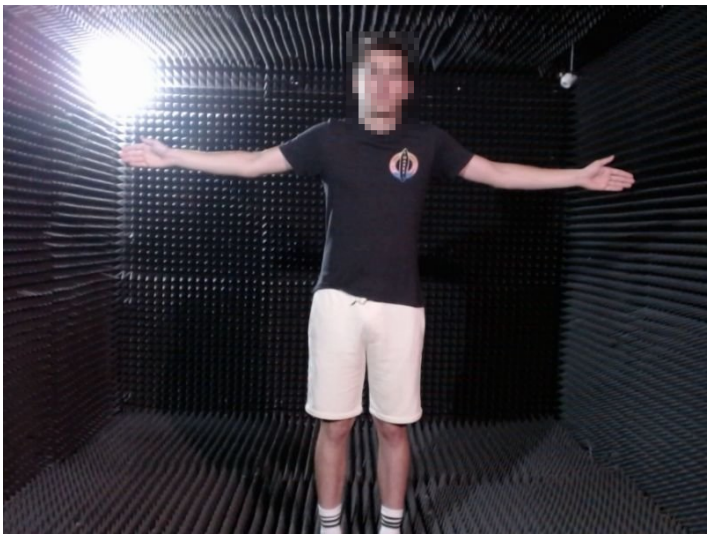


Figura 3.17: Posición del individuo escogida para las capturas

Finalmente la distancia del individuo al sistema de adquisición es de aproximadamente 2.2 metros. Para ello, se ha colocado el pedestal a esta distancia comprobando que es suficiente para capturar la totalidad de la persona con ambos sistemas.

3.3.3. Pulso acústico

Como se ha comentado previamente, el *tweeter* transmite un pulso acústico que ha de ser configurado previamente.

Para el presente trabajo, se ha empleado un pulso de naturaleza gaussiana cuyo ancho de pulso y elección de frecuencia se comenta en los apartados a continuación.

Ancho de pulso

La elección del valor del ancho de pulso supone una relación de compromiso. Es decir, cuanto mayor sea el ancho de pulso, mayor será la energía recibida pero peor será la resolución en rango y peor será el efecto del pulso directo por lo que no es una decisión trivial [2].

Típicamente, en los sistemas RADAR clásicos, existe un rango mínimo a partir del cual el sistema puede medir. Esto es así debido al pulso directo que reciben los micrófonos directamente de la señal acústica generada en el *tweeter*, es decir, sin llegar a reflejarse en ningún blanco. Esto ocurre en los valores de rangos cercanos al array y como es lógico, si el ancho del pulso es mayor, este efecto se extenderá más en rango y con mayor energía.

Por otra parte, el ancho de pulso condiciona la resolución en rango. Si la separación en rango entre dos objetos es comparable al ancho de pulso, entonces será imposible distinguir los dos blancos. En concreto, la resolución en rango se encuentra directamente repercutida por el ancho de banda de la señal transmitida. A grandes rasgos, empleando

un pulso rectangular, un pulso más estrecho implica un aumento del ancho de banda y por ende una mejor resolución en rango.

En contraposición, si el ancho de pulso es mayor, la onda acústica contiene más energía y el array será capaz de detectar blancos más débiles por lo que aumenta la sensibilidad del sistema.

En base a estas consideraciones, para este proyecto, tras probar con tiempos de pulso de 1ms, 2ms y 3ms se ha decidido utilizar 3ms ya que al contar con el apoyo del sistema LiDAR para discriminar al individuo en rango, se prioriza la sensibilidad del sistema.

Frecuencia de pulso

La elección de la frecuencia a utilizar de nuevo supone un compromiso de diseño. Para un espaciado entre sensores prefijado del array, la utilización de frecuencias demasiado altas produce *aliassing* en el dominio espacial generando *gratting lobes*. Esto es un efecto indeseado puesto que se genera "energía confundible" con el lóbulo principal. De este modo, el array tiene una respuesta espacial equivalente al máximo del lóbulo del array para un ángulo al que no está apuntando.

Por otra parte, el ancho del lóbulo principal es inversamente proporcional a la la frecuencia determinando así la directividad del array por lo que un ancho de lóbulo menor será capaz de distinguir con mayor precisión dos blancos angularmente próximos.

De esta manera, es importante tener en cuenta la zona de exploración angular en el sistema. Al limitar los ángulos de excrusión máxima es posible conseguir que los *gratting lobes* caigan fuera del espacio de exploración permitiendo frecuencias de trabajo mayores. En el caso de estudio se ha escogido un área de barrido tanto en azimut como en elevación de $-30°$ a $30°$ garantizando la inexistencia de *gratting lobes* para las frecuencias empleadas.

Finalmente, con la idea de maximizar la información acústica recogida en una sola captura y generar una base de datos multifrecuencial, se ha configurado un pulso resultado de la suma de 3 tonos de frecuencias en el pulso transmitido: 16, 18 y 20kHz. Por lo que el triple de información acústica es capturada en el mismo tiempo.

Capítulo 4. Procesado y obtención de muestras

4.1. Procesado y fusión de imágenes

La integración de los sistemas acústico y LiDAR conlleva el procesado de las imágenes obtenidas por cada sistema y su posterior fusionado. En base a esto, se estructura la presente sección en 3 bloques fundamentales: "Procesado acústico", "Procesado LiDAR" y por último "Fusión de imágenes acústica y LiDAR".

4.1.1. Procesado acústico

A partir de las señales acústicas capturadas en el array de microfonos MEMS, se obtiene una imagen multifrecuencial. Esta imagen consiste en esencia en una matriz de datos 4D (3D para cada frecuencia).

Beamforming

Obtenidas las señales en el tiempo, se requiere de un algoritmo de *beamforming* para combinar las señales capturadas de cada micrófono en un conjunto predefinido de ángulos de apuntamiento y de esta forma obtener una imagen acústica.

Para conformar las señales en banda estrecha, se puede emplear una técnica que extrae las componentes en fase y cuadratura de las señales y realiza el producto con el desfase asociado al ángulo de apuntamiento requerido. Al utilizar una frecuencia única, también es posible aplicar un algoritmo delay-sum que retarda las señales aplicando un retardo distinto por cada ángulo de apuntamiento.

Sin embargo, en este proyecto se utiliza un pulso compuesto por múltiples tonos en lugar de un pulso único. Ante este tipo de pulso multifrecuencial se ha utilizado un algoritmo de conformación basado en la transformada rápida de Fourier o FFT (Fast Fourier Transform).

El algoritmo "delay-sum" necesita mucha memoria para almacenar las señales retardadas mientras que el algoritmo basado en la FFT [57] sólo requiere multiplicar por los desfases siendo por ello óptimo cuando se envían múltiples tonos en la señal.

Por último, los algoritmos de conformado asumen el modelo de onda plana para la propagación de las señales, el aire como un medio homogéneo y que los micrófonos cuentan todos con un diagrama de radiación omnidireccional y misma respuesta frecuencial.

En el proceso de beamforming se escoge también la cuadrícula o "grid" espacial con el que se quiere "barrer" el escenario. Esto determina el tamaño del array 3D obtenido a

la salida y determina la resolución de la imagen. La elección del tamaño del "grid" es una situación de compromiso puesto que un "grid" con un espaciado pequeño obtiene una mejor resolución a coste de incrementar la carga computacional.

No obstante,un "grid" con un espaciado pequeño no mejora la resolución significativamente dado que el factor limitante pasa a ser el ancho del haz que depende directamente de la frecuencia empleada.

En el dominio temporal se han enventanado los datos mediante una ventana de Hamming para minimizar el manchado frecuencial en el cálculo de la FFT y mejorar así la precisión en rango.

Por último, el "grid" escogido cuenta con una resolución angular de $0{,}5^o$ y recorre en azimut y elevación los ángulos entre -30^o y 30^o. Respecto a la resolución en rango, se ha escogido una cuadrícula de 1 centímetro de precisión que cubre la sección entre 2 y 3 metros. El "grid" se ha escogido a partir de 2 metros evitando de esta forma el efecto del pulso directo comentado previamente en la sección 3.3.3. Este efecto se puede observar fácilmente en los primeros instantes de las señales en el domino del tiempo antes de realizar la conformación (Figura 4.1).

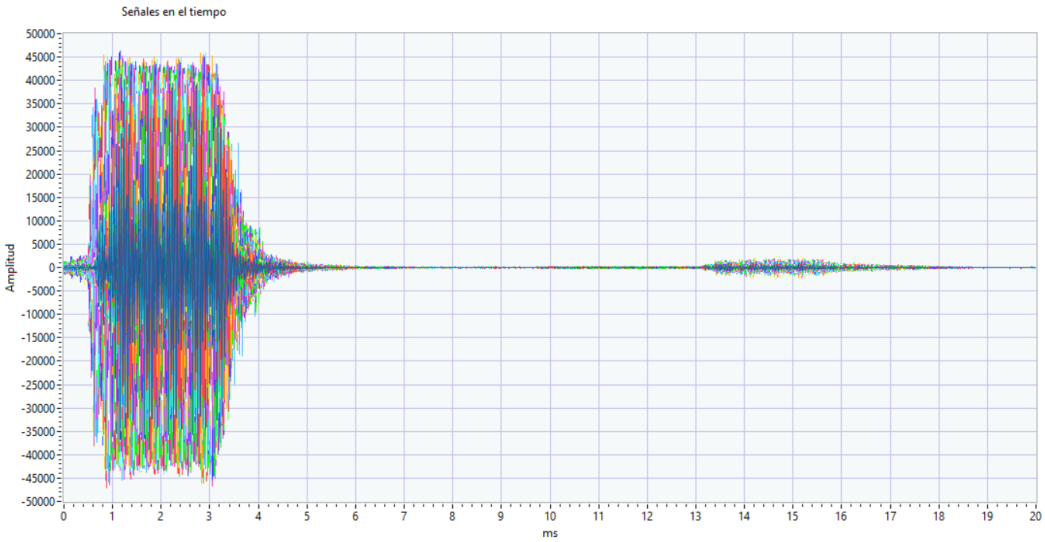


Figura 4.1: Señales en el dominio del tiempo capturadas por los sensores

Filtrado

Esta etapa se ha implementado antes de la etapa de conformación aplicando un filtro paso banda (BPF) con banda de paso entre 12 y 22 kHz.

De este modo, se eliminan principalmente algunas pequeñas interferencias situadas en otras frecuencias (p.e en 16, 18 y 20 kHz) además de la señal en banda base (Figuras 4.2 y 4.3).

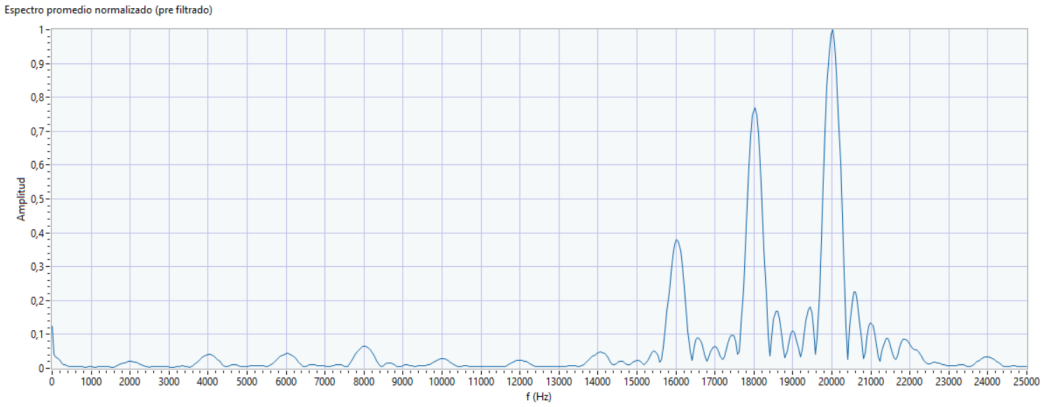


Figura 4.2: DFT promedio normalizada antes de la etapa de filtrado

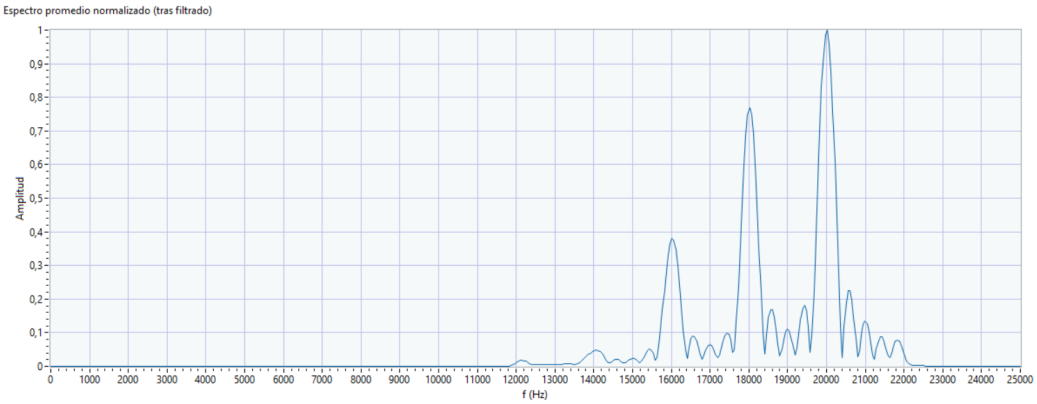


Figura 4.3: DFT promedio normalizada tras la etapa de filtrado

Como última anotación, se puede observar que la amplitud de los tonos recibidos aumenta conforme la frecuencia es mayor. Esto se debe esencialmente a la ganancia creciente de los MEMS para frecuencias altas y próximas a su frecuencia de resonancia.

4.1.2. Procesado LiDAR

Tras la fase de adquisición, en el sistema LiDAR, se obtiene una matriz bidimensional de números que representan la profundidad del píxel y según la escala utilizada. En concreto, se ha empleado una resolución QVGA puesto que la resolución limitante será la de las imágenes acústicas. De este modo, los datos de partida se pueden visualizar como una imagen 2D de tamaño 320x240 en la que los ejes vertical y horizontal representan el número de píxel y el color representa la distancia del punto reflejado (Figura 4.4).

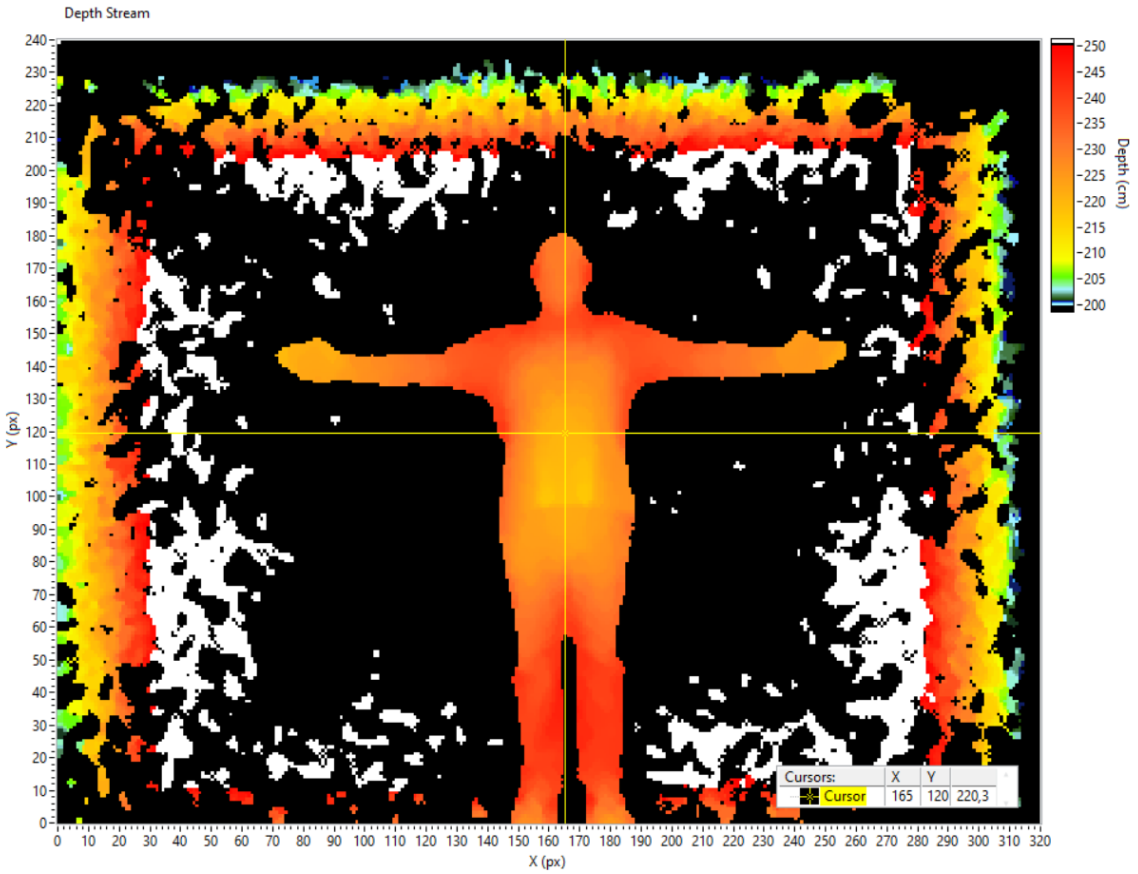


Figura 4.4: Imagen LiDAR 2D de partida

La primera etapa de procesado LiDAR consiste en transformar esta imagen de entrada en una lista de puntos en coordenadas polares (θ, φ, r) y cartesianas (x, y, z) fácilmente operables. Se proporciona a continuación un esquema orientativo (Figura 4.5) para tomar como referencia de los sistemas de coordenadas empleados (donde la posición del sensor LiDAR se toma como origen de coordenadas) y más concretamente el centro del transmisor IR [49].

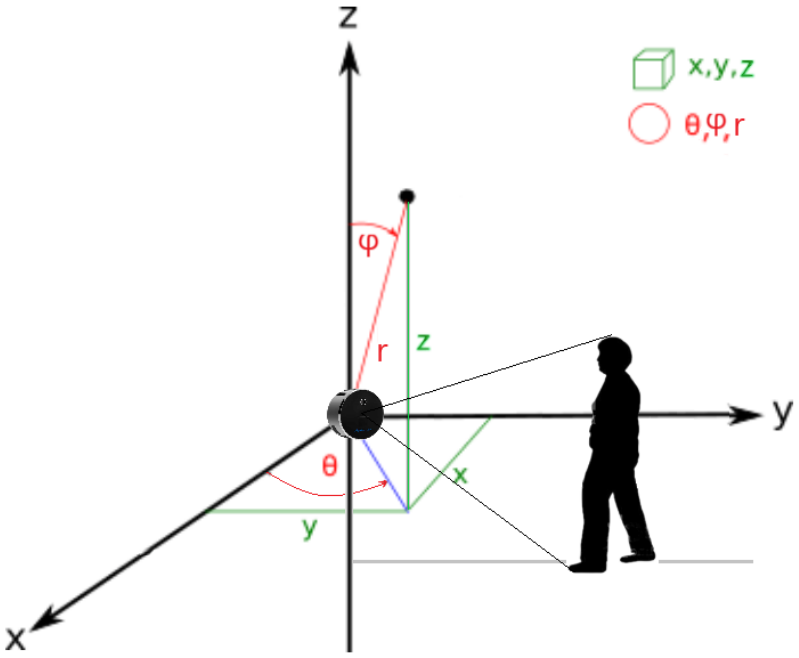


Figura 4.5: Esquema orientativo de los sistemas de coordenadas empleados

Para transformar la imagen a una lista de puntos, en cada píxel se encuentra la información de distancia y mientras que el índice vertical y horizontal del píxel determinan la elevación φ y azimut θ a la que se encuentra (conocido el FOV y la resolución angular, su cálculo resulta trivial). El resto de coordenadas se pueden calcular tal que:

$$z = \tan(\varphi) \cdot y \tag{4.1}$$

$$x = \tan(\theta) \cdot y \tag{4.2}$$

$$r = \sqrt{x^2 + y^2 + z^2} \tag{4.3}$$

De este modo, al tener los datos en un formato de lista, es posible realizar representaciones 3D para visualizar la captura 4.6.

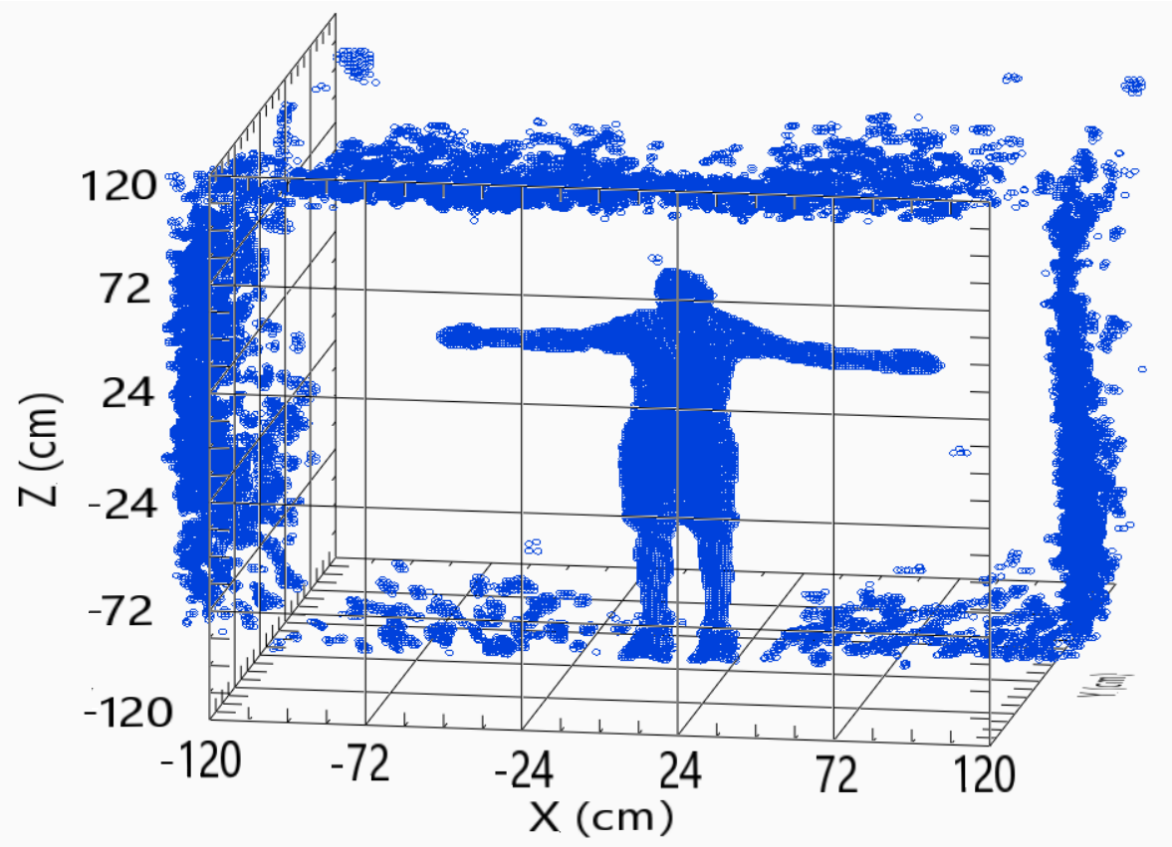


Figura 4.6: Nube de puntos LiDAR adquirida

Sistema de transformación de coordenadas

No hay que perder de vista que el sistema de coordenadas empleado en las capturas acústicas utiliza como centro de coordenadas el centro del array, mientras que, como se ha señalado previamente, el LiDAR toma como punto (0,0,0) el centro del transmisor IR [49]. Por ello, se hace necesario realizar un cambio de coordenadas previo a fusionar las imágenes acústicas y LiDAR.

Como es lógico, ambos sistemas no pueden coexistir en el mismo espacio por lo que la solución pasa por introducir esta nueva etapa de procesado. La transformación de coordenadas se realiza sobre la lista de puntos LiDAR dada su fácil operabilidad. Esta etapa de procesado consiste en transformar las coordenadas de todos los puntos acorde con la diferencia longitudinal entre las coordenadas de ambos centros. Es decir, el centro LiDAR aunque alineado con el centro del array en x, se encuentra 31 centímetros por encima (eje z) y 5 centímetros por detrás (eje y).

Esta operación se realiza en cartesianas. Para un sistema de coordenadas XY y un sistema X'Y', la operación de traslación sobre un punto A a su correspondiente A' en el nuevo centro, se tiene que $X'_A = X_A + X_0$ y $Y'_A = Y_A + Y_0$. Espacialmente, se muestra el concepto en un ejemplo donde tanto X_0 como Y_0 corresponden a valores negativos (Figura 4.7).

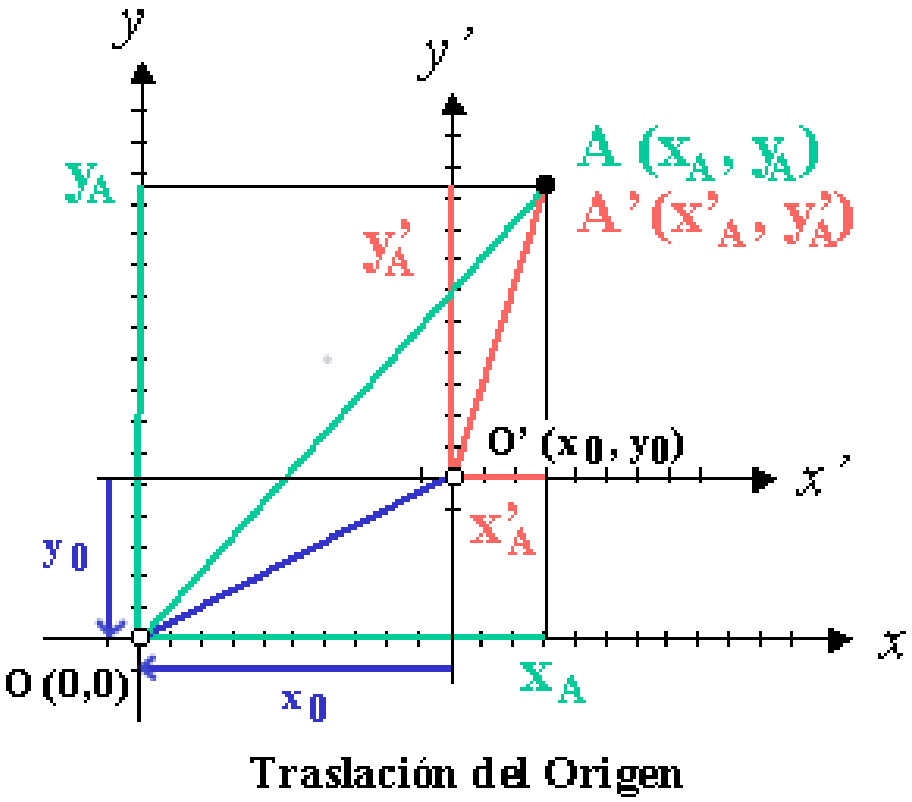


Figura 4.7: Traslación del centro en coordenadas cartesianas

Extrapolando el ejemplo anterior y siendo (X_0,Y_0 y X_0) las distancias calculadas desde el nuevo centro hasta el centro LiDAR, se tiene que $X_0 = -5$, $Y_0 = +31$ y $Z_0 = 0$. De este modo para todos la transformación de un punto A a otro A' en el nuevo sistema de coordenadas se realiza tal que $X'_A = X_A + X_0$, $Y'_A = Y_A + Y_0$ y $Z'_A = Z_A + Z_0$.

De este modo, si un punto capturado por el LiDAR se encontraba en $x = 5$, al realizar el cambio de coordenadas el punto pasa a estar en $x = 0$ dado que el LiDAR se encuentra 5 centímetros por detrás.

Mientras se realizan las traslaciones, se recalculan los puntos en coordenadas polares para mantener la información en ambos sistemas.

Segmentación LiDAR

Observando la nube de puntos LiDAR obtenida en la primera etapa de procesado (Figura 4.6), se puede ver como la imagen es sumamente ruidosa. El objetivo es obtener una imagen LiDAR donde se encuentre exclusivamente la persona capturada dado que los puntos pertenecientes al suelo, techo y paredes de la cámara no son relevantes.

En este contexto, toma especial relevancia una de las etapas de procesado más críticas en la adquisición de imágenes LiDAR 3D: la segmentación [58]. Esta técnica, consiste en clasificar puntos de una nube 3D en grupos según distintos criterios de homogeneidad como la posición de los puntos, forma u otros atributos. Esto permite al sistema identificar objetos o regiones de interés.

De hecho, la segmentación semántica es una técnica muy importante sistemas de conducción autónoma y consiste no sólo en identificar regiones sino también en asignar una etiqueta específica a cada región identificada, es decir categorizar el elemento entre distintas clases. Para ello, muchos sistemas emplean algoritmos de inteligencia artificial y se ayudan de la información obtenida a través de una cámara óptica [59].

No obstante, el problema de segmentación enfrentado en este proyecto es notablemente más sencillo que los que puedan darse en el ámbito de la conducción autónoma y de hecho al haber un único objeto que identificar en las imágenes, puede verse también como una etapa de filtrado espacial.

En segmentación, se han demostrado empíricamente las ventajas del método de "extracción del terreno" antes de la agrupación 3D de objetos [58]. Este método de segmentación tiene como objetivo separar los puntos que representan el suelo de los que no y esta basado en agrupar vóxeles adyacentes en función de las medias y varianza vertical [58].

Basado en este método, se ha diseñado un sistema de segmentación que emplea parámetros estadísticos de la imagen para así separar al individuo del entorno.

Primeramente, se realiza un filtrado espacial en el que se utilizan las medidas de la cámara anecoica eliminando en base a unos umbrales preestablecidos información perteneciente a las paredes y techo de la cámara (Figura 4.8).

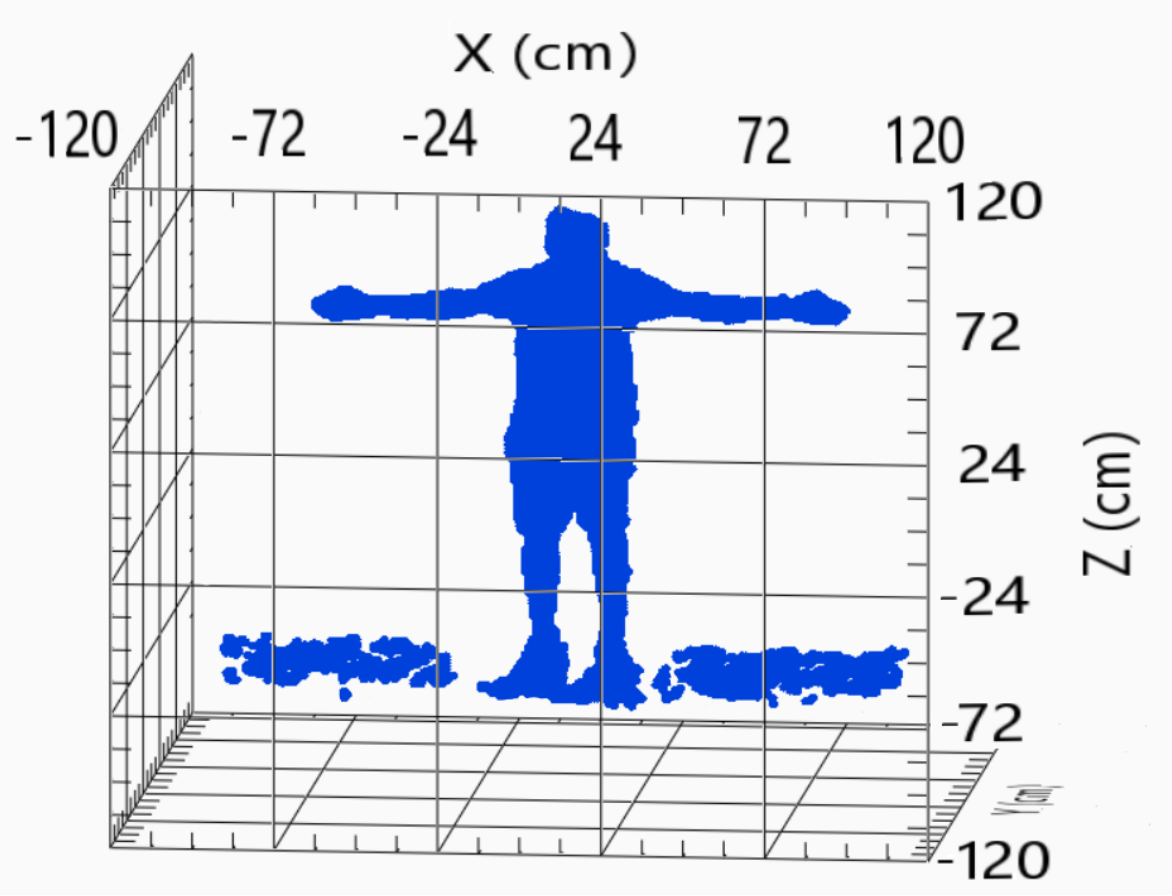


Figura 4.8: Segmentación. Primer filtrado espacial

En este punto, se identifican posiciones y referencias clave para la segmentación de la persona en base a la construcción de histogramas en las distintas dimensiones. Esta idea permite realizar una segmentación independiente del individuo escaneado. En concreto, se identifica mediante este método el suelo, el torso y la posición de los brazos determinando así los umbrales espaciales específicos para la persona capturada. Además, se calcula también el centroide geométrico de la persona. Este proceso permite una segmentación final donde el individuo se encuentra perfectamente separado del entorno (Figura 4.9).

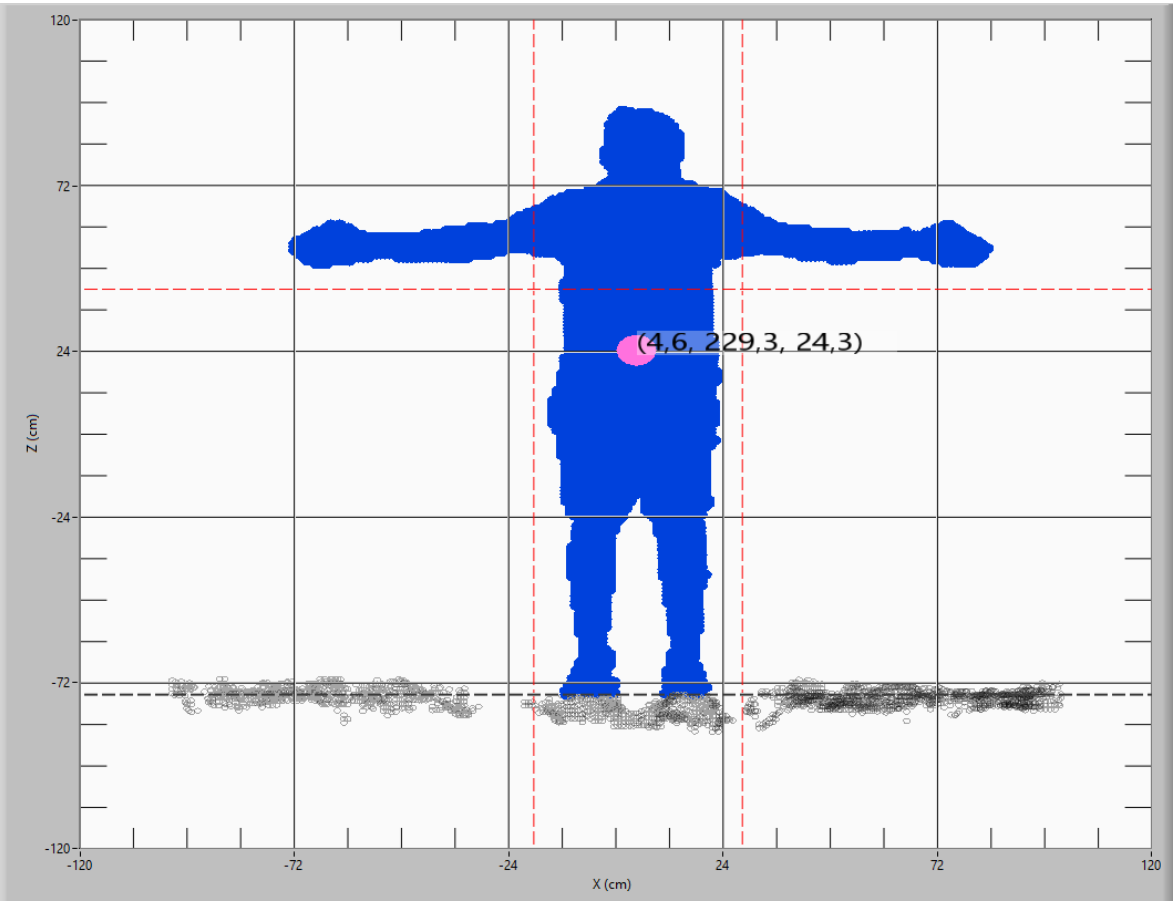


Figura 4.9: Segmentación. Referencias y filtrado inteligente

Para ilustrar el concepto planteado, se muestran a continuación dos de los histogramas que permiten determinar estas referencias (Figuras 4.10 y 4.11).

El primer histograma (Figura 4.10) se ha calculado en Z sobre los puntos resultantes tras una primera eliminación del suelo y parte de los pies del individuo (al ser calculado a partir de la altura mínima más cierto umbral). En el histograma, se observan dos barras que sobresalen especialmente frente a las demás. Esto corresponde a la altura de los brazos dado que por la posición escogida, es la región en Z donde existe un mayor número de puntos.

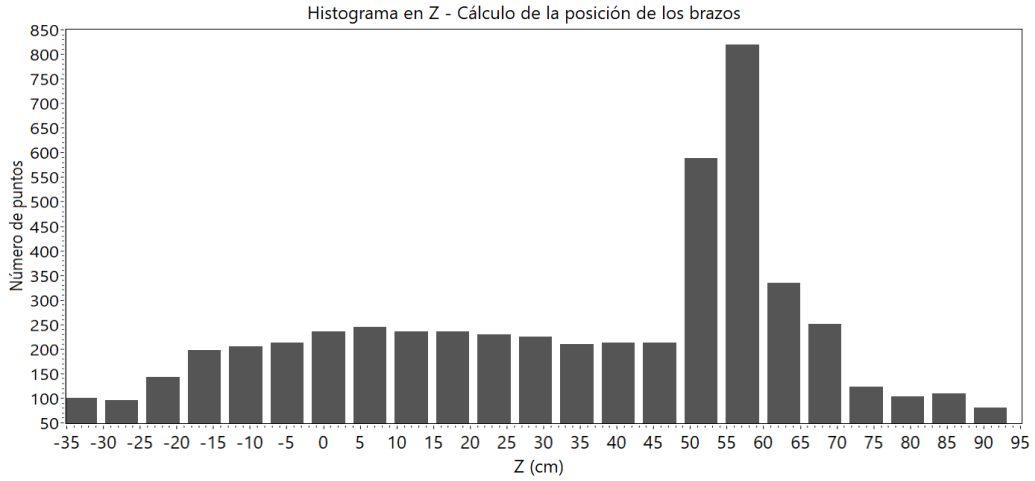


Figura 4.10: Histograma referencia en el cálculo de la altura de los brazos

Otro ejemplo de este proceso se puede observar en un segundo histograma calculado esta vez sobre el eje X (Figura 4.11). Este se ha calculado sobre todos los puntos y en él se aprecia claramente dónde se encuentra el torso ya que existe un mayor número de puntos en esas coordenadas (aproximadamente entre -20 y 25). Incluso visualmente se puede intuir el centro en X de la persona dado el mínimo local que corresponde al hueco entre las piernas (donde a pesar de la cabeza, el número de puntos es menor).

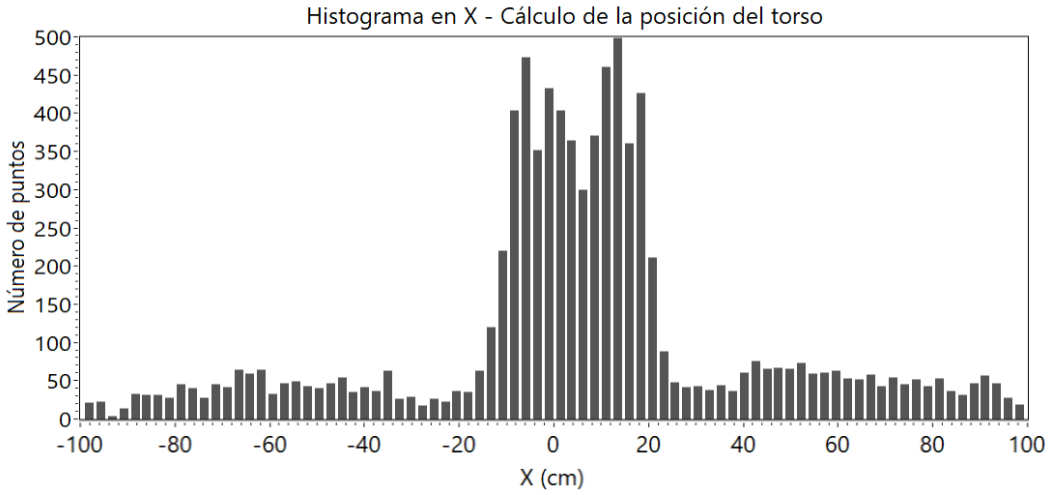


Figura 4.11: Histograma referencia en el cálculo de la posición del torso

De esta manera, al finalizar esta etapa de procesado, se obtiene una lista de puntos correspondientes únicamente al individuo y manteniendo su correspondencia en coordenadas polares.

4.1.3. Fusión de imágenes acústica y LiDAR

Debido a la naturaleza de propagación del pulso acústico y algoritmos de conformado, la fusión de imágenes se ha realizado en coordenadas polares, el entorno de operación natural del sistema acústico.

Submuestreo LiDAR

Como se ha comentado en capítulos anteriores, la resolución de las imágenes LiDAR es mayor que la de las imágenes acústicas por lo que aunque los sistemas de coordenadas sean los mismos y esten referenciados sobre el mismo origen, la fusión no es posible si la resolución no es la misma. Para ello, el *grid* 3D de distribución uniforme empleado en la conformación de las imágenes acústicas se ha aplicado sobre la nube de puntos LiDAR asociando cada uno de los puntos de la nube a un voxel de la cuadrícula y realizando de este modo un submuestreo sobre la imagen LiDAR original.

Para representar esta matriz 3D, se realizan proyecciones por acumulación o suma en las que los valores de los puntos 3D se suman mapeando los valores resultantes sobre cada uno de los planos de proyección (Figura 4.12).

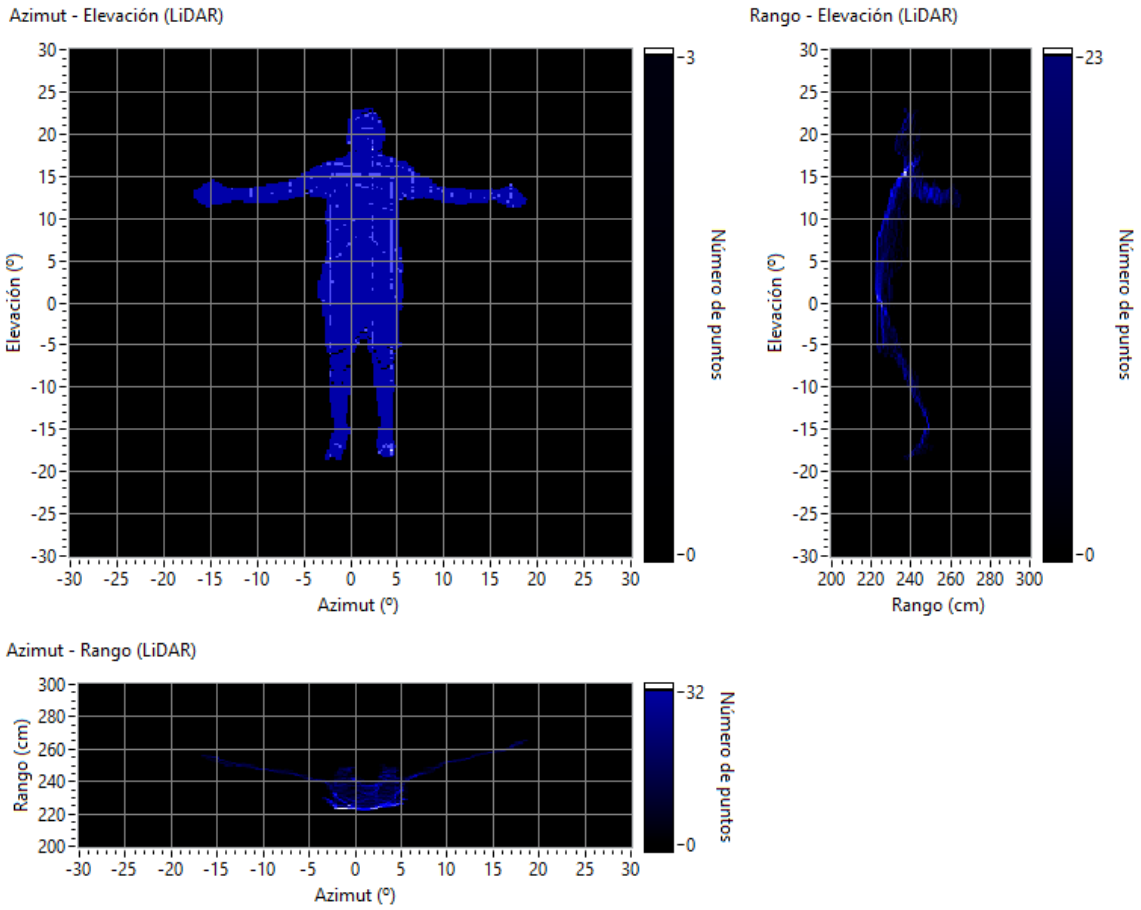


Figura 4.12: Imagen LiDAR. Proyecciones obtenidas por acumulación de puntos

En las representaciones que involucran el rango el cuerpo del individuo da la sensación de estar "arqueado", sin embargo, esto es un efecto de la representación en polares ya que a una misma distancia, los puntos con mayor azimut o elevación se encuentran más alejados en rango.

Fusión de información y proyecciones

Fusión 3D Finalmente, obtenidas ambas matrices 3D (acústica y LiDAR) calculadas sobre el mismo *grid*, la opción más sencilla consiste en aplicar la imagen LiDAR a modo de máscara sobre la imagen acústica eliminando así todos los vóxeles no presentes en la captura LiDAR y representando en cada uno la energía acústica asociada a ese punto.

Teniendo en cuenta que en el sistema acústico se emplean pulsos de 3 ms, el rango sobre el que se captura la energía producida por un eco se extiende notablemente en el espacio (Figura 4.13).

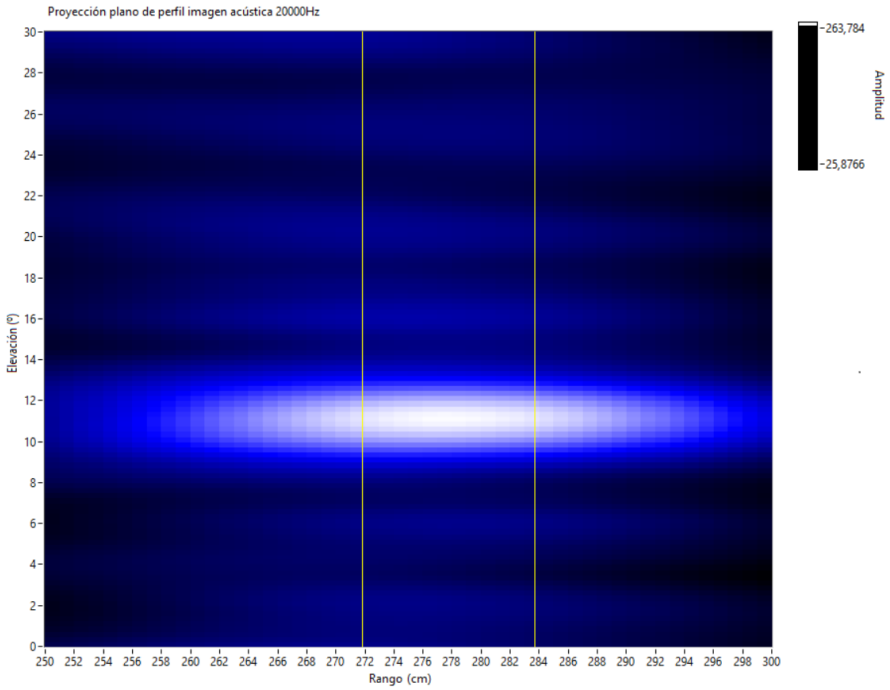


Figura 4.13: Proyección en plano de perfil de un blanco puntual con pulso de 3m

En base a este razonamiento, otra filosofía sería emplear para cada vóxel sus X vecinos posteriores y/o anteriores para escoger el valor de energía medio o máximo en esa ventana. Esta técnica proporciona robustez ante pequeños errores en rango y tiene en cuenta mayor información acústica para el cálculo final. Para el caso de estudio, ambos métodos han proporcionado resultados muy similares por lo que finalmente se ha optado por este segundo aunque utilizando una ventana pequeña (sólo se tienen en cuenta los 2 vóxeles adyacentes).

Proyecciones Finalmente, obtenida la imagen 3D, existen diversas opciones válidas para realizar el paso a una proyección 2D. Una posibilidad es realizar la suma acumulativa (Figura 4.14).

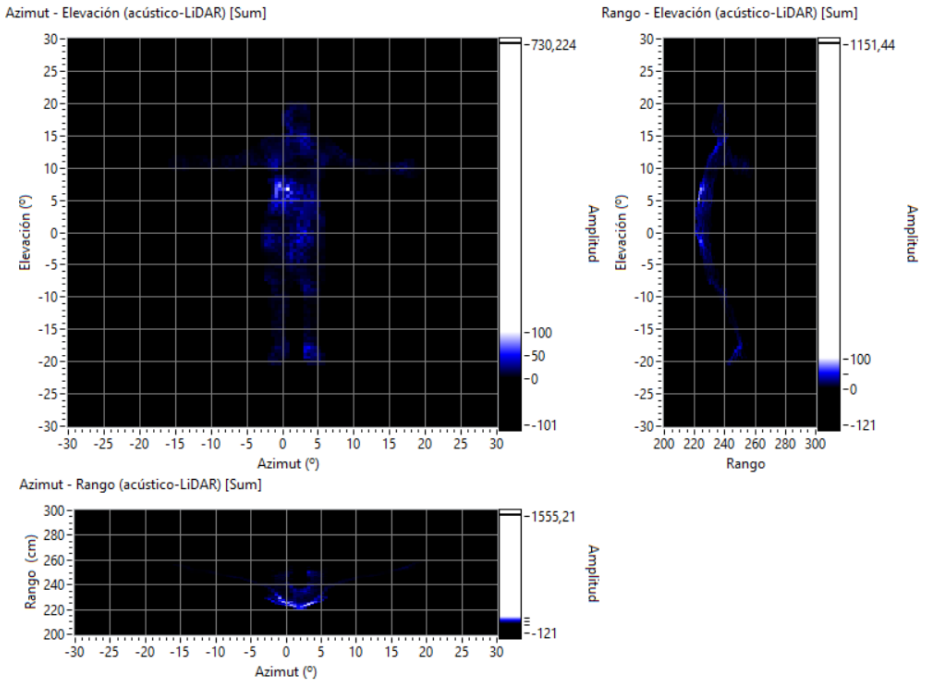


Figura 4.14: Proyección de tipo suma sobre una imagen acústico - LiDAR (20kHz)

No obstante, este tipo de proyección da más peso a las zonas con mayor número de puntos por lo que otra posibilidad es realizar la media entre todos los vóxeles proyectados eliminando así esa dependencia con el número de puntos (Figura 4.15).

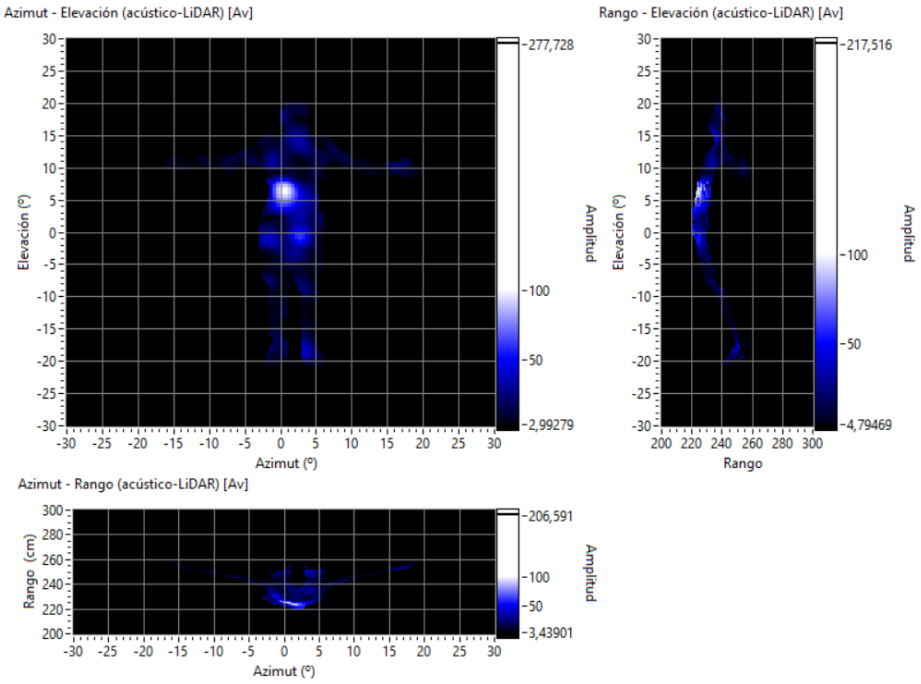


Figura 4.15: Proyección media sobre una imagen acústico - LiDAR (20kHz)

Finalmente, aunque se trata de un procesado no lineal, se ha optado por aplicar una proyección MIP (Maximum Intensity Proyection) (Figura 4.16). Esta proyección es una herramienta que se emplea habitualmente en la representación de imágenes médicas 3D como resonancias magnéticas [60] o tomografías computarizadas [61] y consiste en proyectar el píxel con valor máximo.

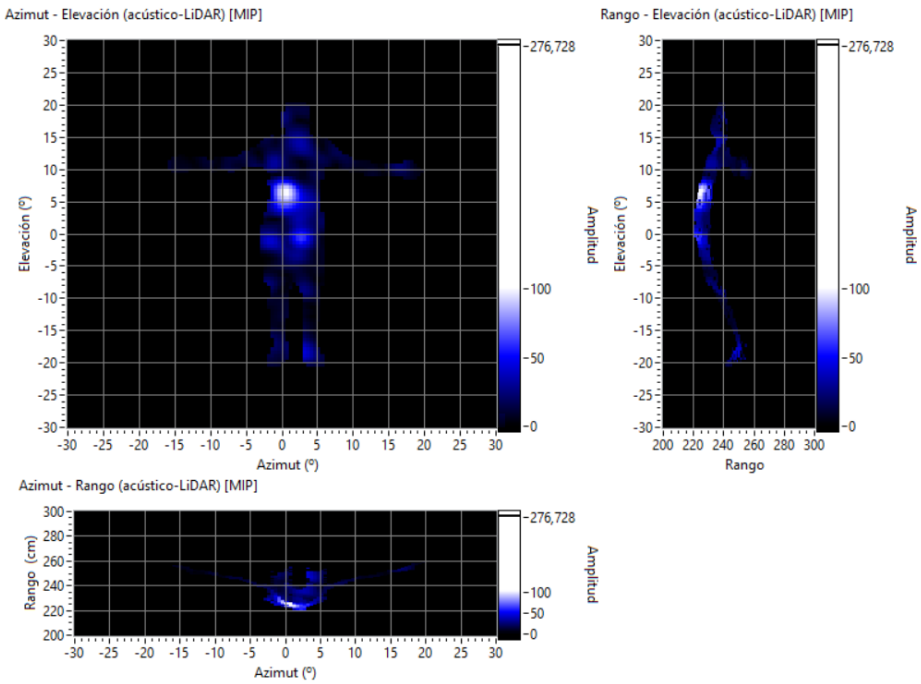


Figura 4.16: Proyección MIP sobre una imagen acústico - LiDAR (20kHz)

Dado que la respuesta del sistema ante un blanco puntual se observa en las imágenes como un máximo de energía, en anteriores estudios se plantea sintetizar las imágenes acústicas tomadas sobre los individuos en base a los máximos presentes en la imagen. Este razonamiento sugiere que la información biométrica más relevante se encuentra en las principales reflexiones acústicas.

Desde esta perspectiva, la mejor opción consiste en mantener la información acerca de los máximos empleando proyecciones MIP. De hecho, utilizando proyecciones MIP, se obtienen resultados más fieles a las imágenes acústicas.

Verificación empírica del sistema de calibración Previamente a la obtención de la base de datos, se ha realizado un experimento previo que ha permitido verificar la correcta calibración de ambos sistemas. Para ello, se ha posicionado un pequeño balón de plástico sobre el techo de la cámara anecoica que actúa ante el sistema acústico como un blanco puntual (Figura 4.17).

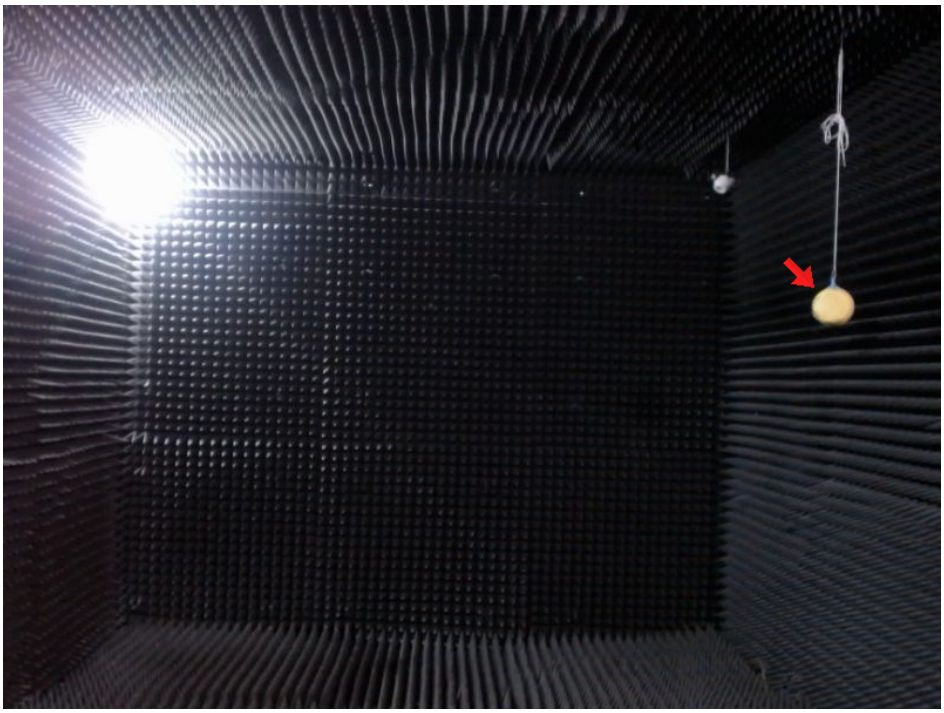
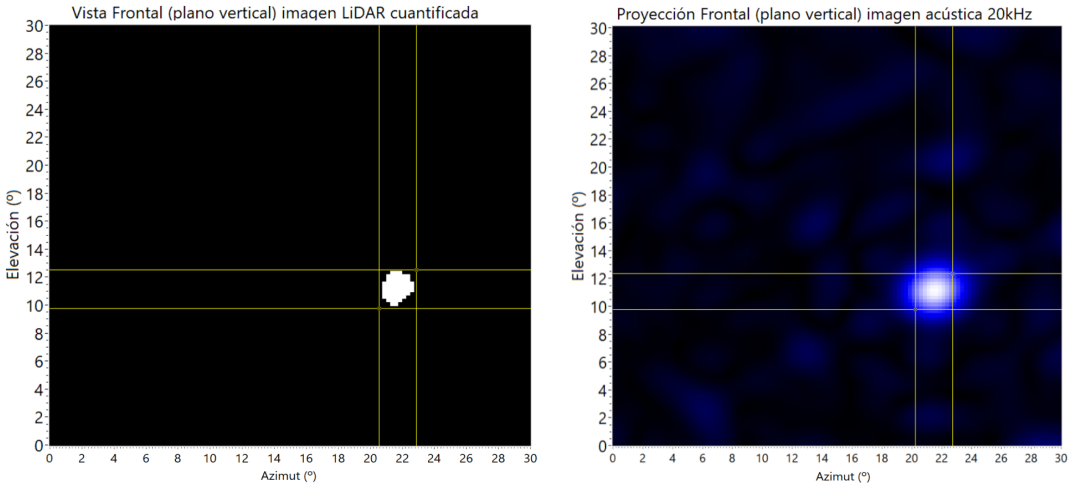


Figura 4.17: Escenario de calibración

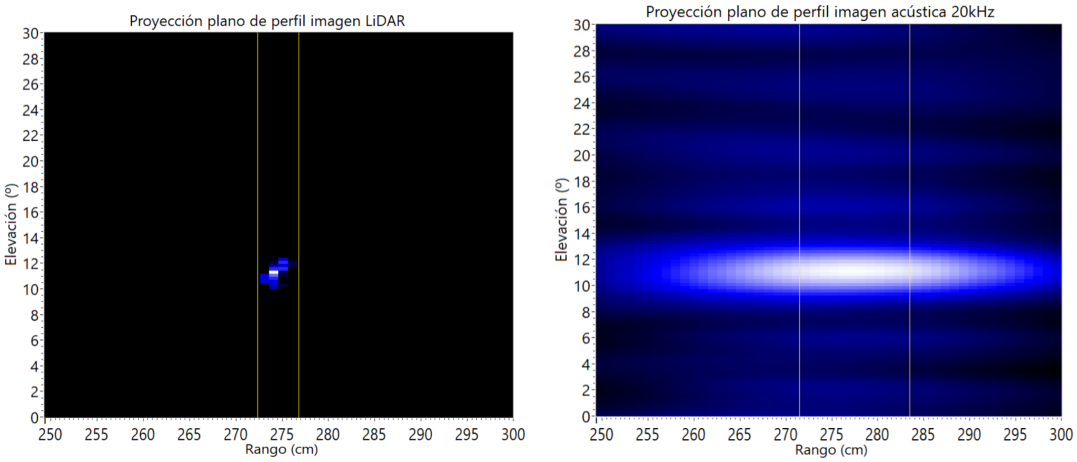
De esta manera, verificando las coordenadas del blanco en ambos sistemas (Figuras 4.18 y 4.19) se ha corroborado la correcta calibración del sistema global.



(a) Captura LiDAR (b) Captura acústica

Figura 4.18: Vista frontal del blanco puntual

Observando la proyección en el perfil rango - elevación (Figura 4.19), se puede ver también como los sistemas están correctamente calibrados observando que el primer punto donde se ve la máxima energía del pulso corresponde con la posición en rango del blanco. No obstante como se ha comentado previamente, el pulso acústico se extiende notablemente en rango dada su duración de 3ms.



(a) Captura LiDAR (b) Captura acústica

Figura 4.19: Vista de perfil del blanco puntual

4.2. Obtención de firmas

En este proyecto se ha considerado importante contar con un número significativo de muestras e individuos capturados con el fin de realizar un estudio más amplio que en proyectos anteriores y establecer una base de datos sobre la que futuros trabajos puedan asentar sus bases.

De este modo, mientras que en estudios anteriores se capturaron 3 individuos [3], en el presente proyecto se ha tomado muestras de 10 sujetos.

En total, se han capturado 2100 muestras de cada individuo. Como se expone en la sección 3.2.2, esto supone algo más de 12 minutos en los que el individuo debe mantener la postura empleada en las capturas. No obstante, para evitar mantener los brazos extendidos durante 12 minutos seguidos, los datos se han tomado por tandas permitiendo descansar al individuo capturado entre ellas.

En resumen, 2100 muestras de 10 individuos con diferentes características (Tabla 4.1) conforman la base de datos elaborada.

ID	**Género**	**Constitución**	**Altura**
00	Masculino	Muy fuerte	Alta
01	Masculino	Normal	Promedio
02	Femenino	Normal	Promedio
03	Femenino	Normal	Pequeña
04	Masculino	Normal	Alta
05	Masculino	Normal	Promedio
06	Masculino	Muy fuerte	Alta
07	Femenino	Delgada	Pequeña
08	Femenino	Normal	Pequeña
09	Femenino	Normal	Promedio

Tabla 4.1: Características morfológicas de los individuos capturados

Capítulo 5. Desarrollo del identificador

5.1. Análisis preliminar

Uno de los objetivos de este proyecto consiste en estudiar las características acústicas que puedan ser diferenciales entre los individuos. Contar con una base de datos como la construida posibilita llevar a cabo este estudio por lo que antes de desarrollar el algoritmo de identificación, se ha realizado un estudio estadístico de las imágenes capturadas que facilite una comprensión más completa de su naturaleza.

El análisis se ha centrado en los máximos absolutos de energía de cada imagen y más en concreto, en su variación entre individuos y frecuencias. Para ello, primeramente se ha obtenido para cada individuo un histograma que representa las amplitudes máximas en cada una de sus 2100 imágenes capturadas (Figura 5.1). Todos los histogramas se han representado siguiendo la regla de "Freedman-Diaconis" [62] para seleccionar la anchura de los intervalos empleados.

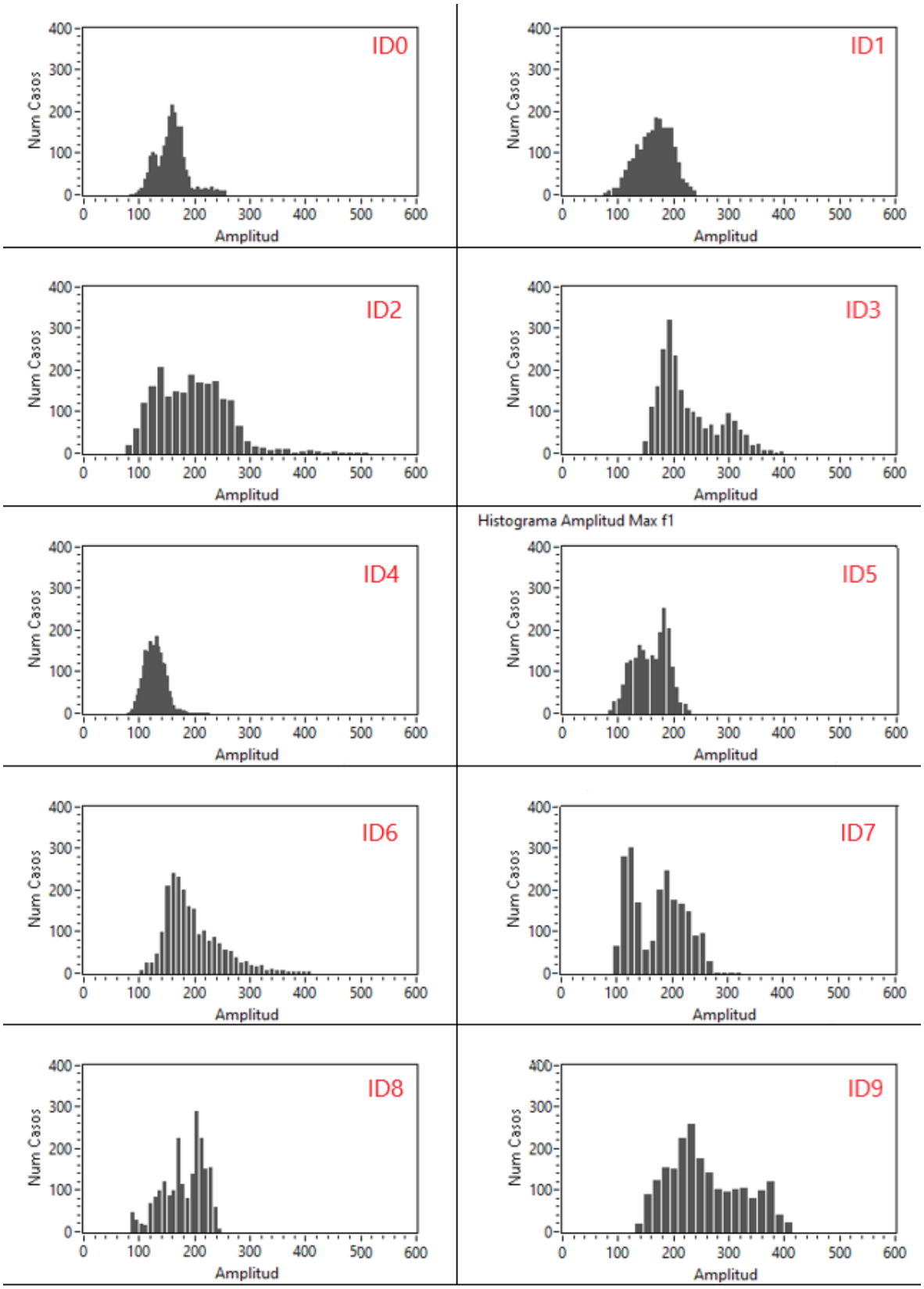


Figura 5.1: Histograma de amplitudes máximas para cada individuo (a $f_1 = 16kHz$)

En general, es interesante observar cómo cada individuo presenta un histograma dife-

rente respecto al de los demás. Este resultado permite corroborar la hipótesis de que la información de energía absoluta es importante puesto que es una característica más que permite diferenciar a los individuos. No obstante, se presume que la variabilidad en estos valores puede tener una relación directa con la estabilidad de la postura del individuo al ser capturado. De hecho, es inevitable que cuando una persona está siendo capturada realice micro movimientos que la hagan cambiar ligeramente la postura u orientación de ciertas partes del cuerpo. Dependiendo de la postura y la orientación, algunas zonas de reflexión pueden quedar ortogonales a la línea recta que los une con el array variando la energía reflejada [15].

Se han representado también estos histogramas para las frecuencias $f_2 = 18kHz$ y $f_3 = 20kHz$ observando las diferencias a la hora de emplear una frecuencia distinta. Los resultados han sido similares obteniendo histogramas con formas comparables para las distintas frecuencias sobre el mismo individuo. No obstante dada la ganancia superior de los micrófonos ante frecuencias más elevadas, las amplitudes son mayores en estos casos. Un ejemplo de este comportamiento se muestra para los individuos 2 y 4 (Figuras 5.2 y 5.3)

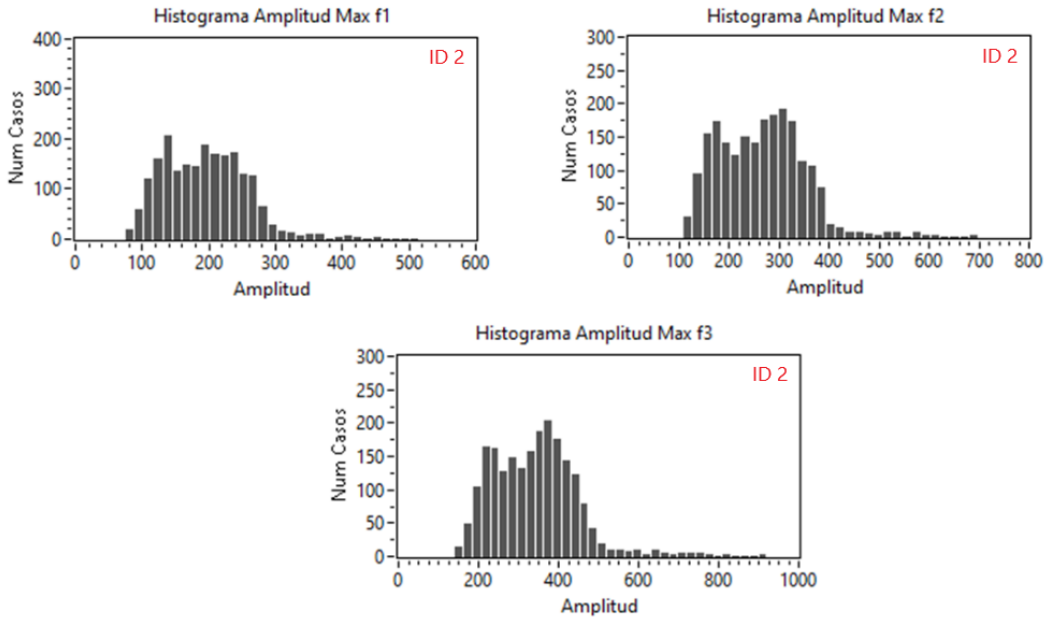


Figura 5.2: Histograma de amplitudes máximas para el individuo 2 (a f_1,f_2 y f_3)

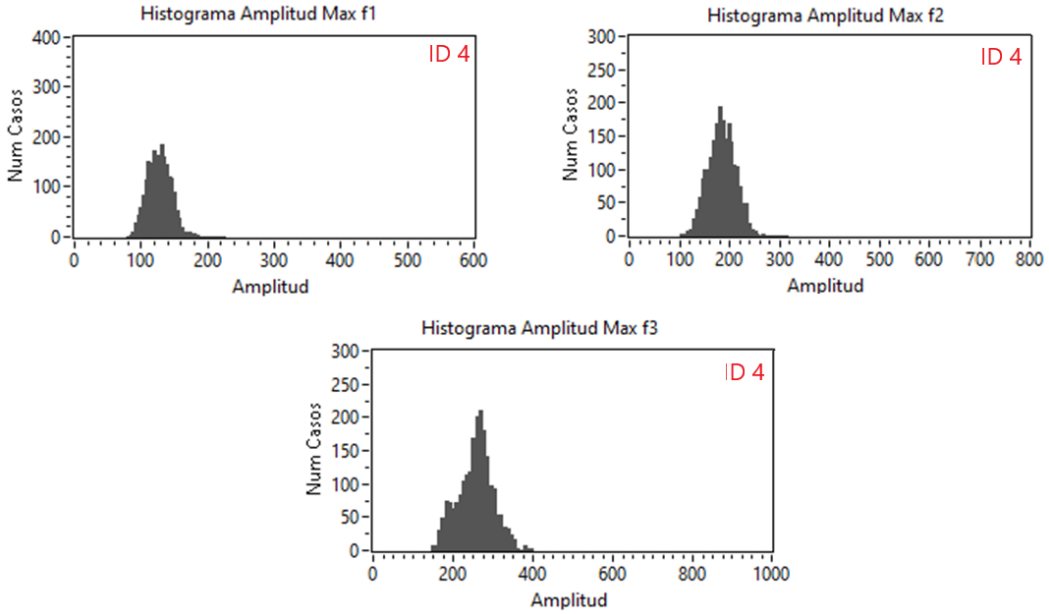


Figura 5.3: Histograma de amplitudes máximas para el individuo 4 (f_1,f_2 y f_3)

Un gráfico construido a partir del promedio de las amplitudes máximas valida la discusión previa (Figura 5.4). En dicho gráfico, se puede observar cómo la amplitud máxima media varía para los distintos individuos en las 3 frecuencias. Como resumen, en todos los casos las frecuencias más altas producen amplitudes máximas medias mayores y las diferencias de amplitud máxima media entre individuos sí parecen ser significativas, superando 200 unidades en el mayor caso.

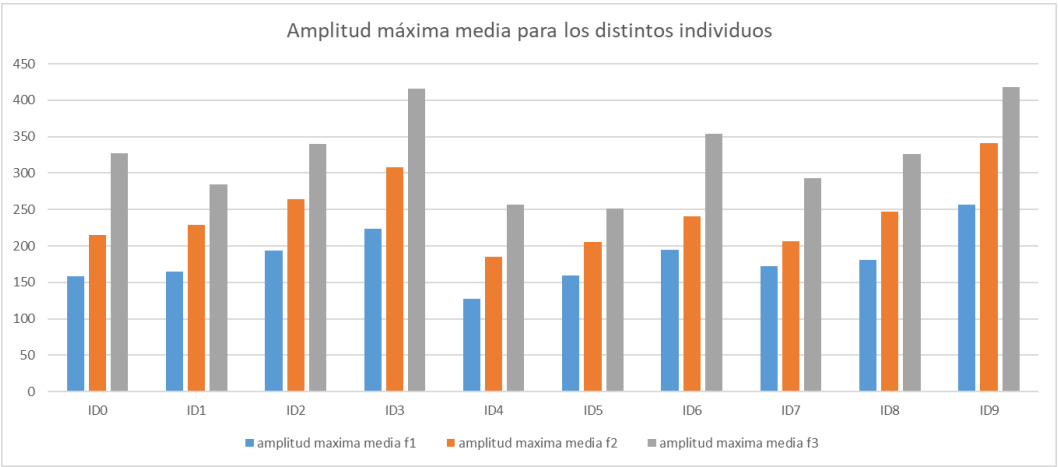


Figura 5.4: Amplitud máxima media por individuo y frecuencia

Otro objetivo de estudio ha sido la ubicación de los máximos absolutos en cada imagen (Figura 5.5). Esto permite identificar cuales son los mayores elementos de reflexión en los individuos, su recurrencia y de este modo las diferencias entre individuos. En rojo se muestra la ubicación de los máximos percibidos a $f_1 = 16kHz$, mientras que en gris se muestra la silueta del individuo para tomarla como referencia a la hora de asociar los máximos con las partes del cuerpo.

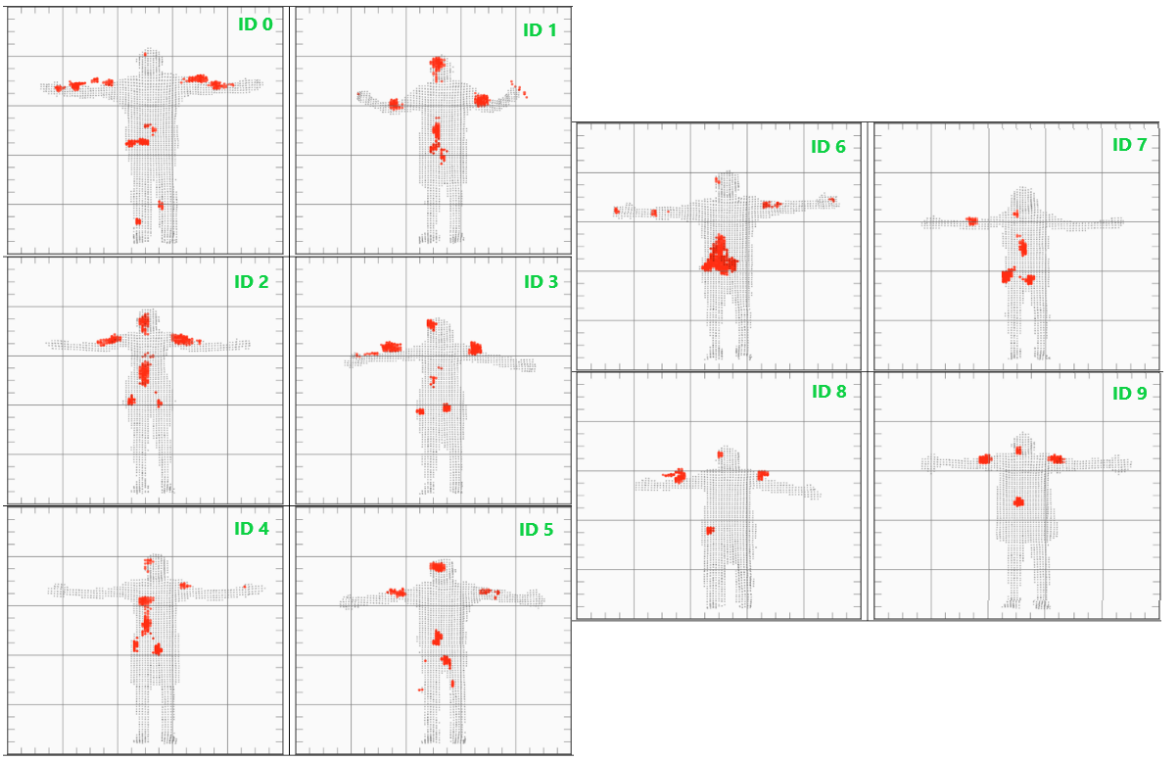


Figura 5.5: Elementos de máxima reflexión en cada individuo (a $f_1 = 16kHz$)

Como primera conclusión, estos elementos de máxima reflexión no corresponden en general a puntos en concreto sino a distintas zonas. Típicamente torso, cabeza y hombros, aunque en algunas ocasiones menos comunes también se producen en las manos y en el caso de los sujetos 0 y 5 en las piernas.

Es posible establecer ciertas relaciones de predisposición a reflejar el máximo de energía en distintas zonas del cuerpo ya que por ejemplo el individuo 0 o el 7 no reflejan máximos de energía en la cabeza mientras que los individuos 1,2 o el 5 lo hacen en numerosas ocasiones. También parece muy característico el torso como elemento de reflexión en el individuo 6.

Otro dato significativo son las reflexiones producidas sobre el individuo 1 en el pliegue de los brazos. Se ha observado que este sujeto arquea ligeramente los brazos en las capturas produciendo una superficie de reflexión puntual en la zona del codo reflejando así más comúnmente máximos de energía en ese punto.

Finalmente, al igual que ha sucedido con los histogramas anteriores, se ha comprobado que la posición de estos elementos de reflexión es también similar en las 3 frecuencias empleadas. A modo de ejemplo se muestran estos resultados para los individuos 0 y 4 (Figuras 5.6 y 5.7).

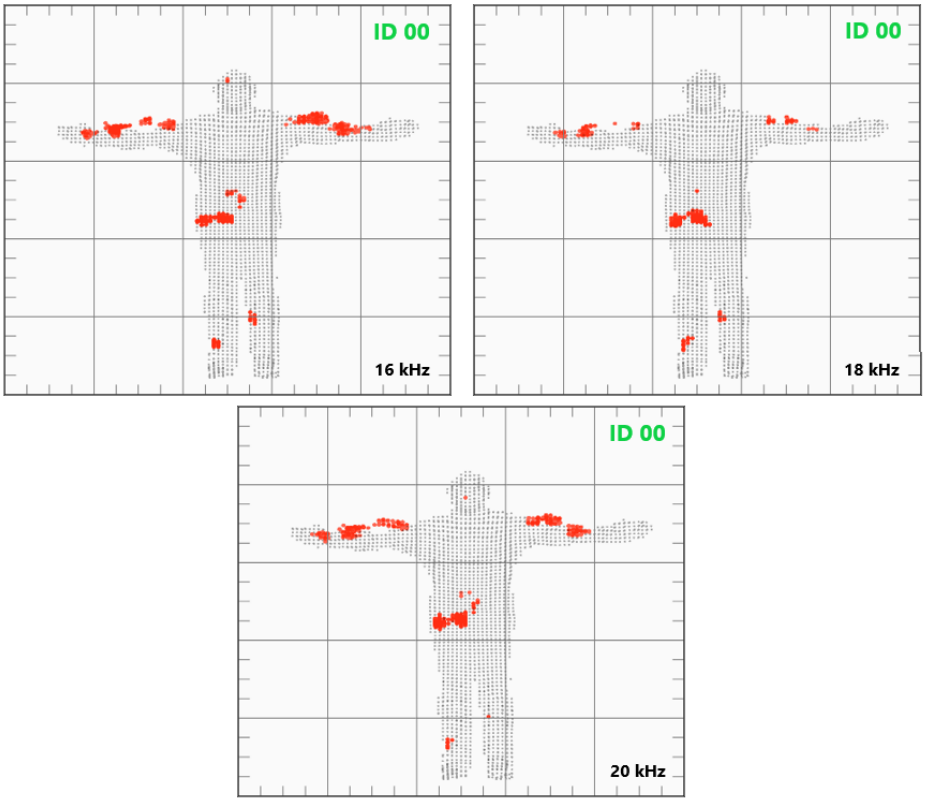


Figura 5.6: Elementos de máxima reflexión para el individuo 0 (f_1,f_2 y f_3)

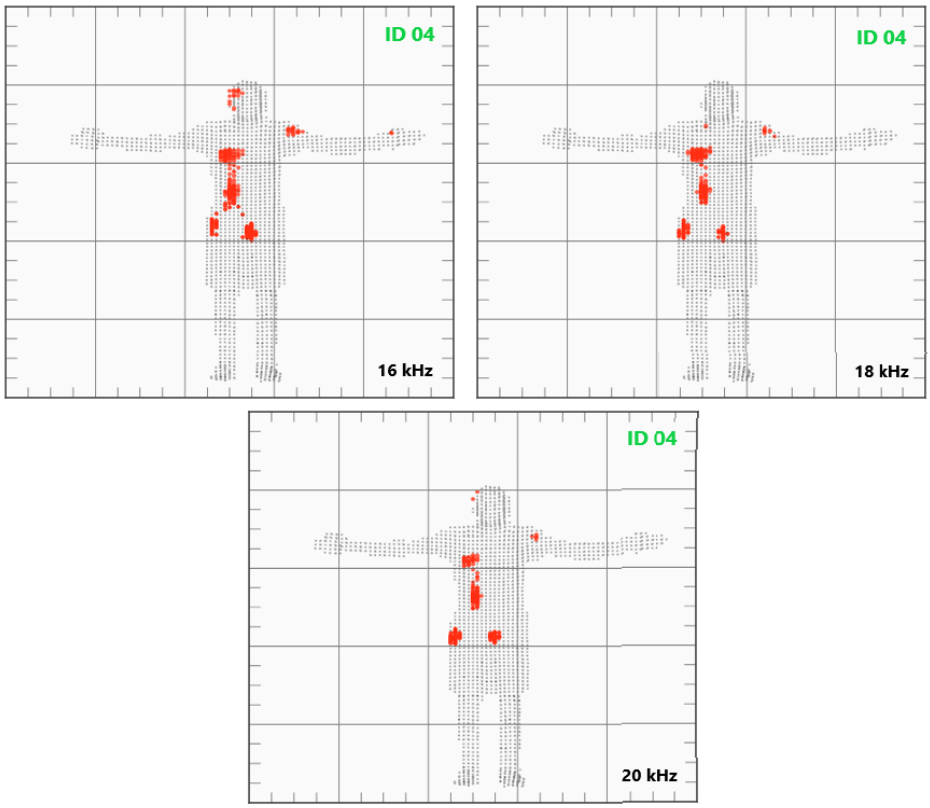


Figura 5.7: Elementos de máxima reflexión para el individuo 4 (f_1,f_2 y f_3)

5.2. Planteamiento del algoritmo

Al planificar el algoritmo a utilizar se debe tener en cuenta el tipo de datos del que se parte. En este caso, no se trata de un volumen sino de una superficie 3D que contiene en cada punto información acústica sobre la energía recogida. Además se dispone de 3 imágenes independientes por cada muestra (para las frecuencias de 16, 18 y 20 kHz).

La clasificación de imágenes 3D se encuentra profundamente ligada a la representación 3D de los datos. La separación principal al tratar con representaciones 3D radica en la categorización entre datos euclidianos y no euclidianos. En particular, únicamente se dispone de datos con una estructura euclidiana cuando pueden ser tratados como puntos en un espacio tridimensional [63].

Centrando el problema en los datos disponibles, la presencia de valores de energía en cada vóxel agrega una dimensión adicional al problema. De esta forma, no solo se trata de la ubicación espacial en 3D, sino también de la magnitud de la energía en cada punto. En este contexto, es importante comprender que los datos 3D todavía pueden tener una estructura euclidiana en términos de su disposición espacial en un espacio tridimensional. No obstante, la información adicional de energía agrega complejidad al problema.

Una de las formas de tratar datos 3D es la proyección en un espacio 2D. Esta proyección convierte al objeto 3D en una cuadrícula 2D con características específicas conservadas dependiendo del tipo de proyección. Una práctica común para representar datos 3D ha sido el dominio esférico [64]. Esto facilita el procesamiento de los datos 3D debido a la estructura de cuadrícula euclidiana de las proyecciones resultantes y permite la utilización de modelos de aprendizaje bien estudiados [63], [65].

Muchas representaciones complejas son posibles a la hora de proyectar los datos 3D, sin embargo, el uso de CNNs se encuentra ampliamente extendido para clasificar dichas imágenes proyectadas [63].

De esta manera y desde la perspectiva descrita en otros estudios que abordan problemas similares, en el proyecto se ha decidido transformar las imágenes 3D en proyecciones 2D para aplicar después una red neuronal convolucional bidimensional.

No obstante, es importante mantener el enfoque en el objetivo principal: la elaboración de un sistema biométrico. Los sistemas biométricos pueden operar en 3 modos distintos: registro, identificación y verificación. El modo registro (ya implementado) consiste en generar una base de datos con la que se comparan los datos de entrada al sistema. Por el contrario, los otros dos modos surgen de la forma en la que se realiza esta comparación [66].

El modo identificación consiste en decidir la identidad de la persona realizando una comparación "uno-a-muchos" y sin que el usuario proporcione previamente una identidad pretendida. Además dependiendo del resultado proporcionado por el sistema es posible diferenciar dos grupos principales: si tiene la capacidad de identificar que ninguna identidad de la base de datos corresponde con los datos adquiridos, se considera de identificación en conjunto abierto (*open set*); por el contrario si el sistema no cuenta con esta capacidad se considera de conjunto cerrado (*closed set*) [66].

Por otra parte, en el modo verificación es necesario proporcionar los datos de entrada y el modelo al que supuestamente pertenece dicha información. El sistema valida la identidad del individuo realizando una comparación entre ambos y se indica si los datos pertenecen a la identidad reclamada o a un impostor, en función de si se supera o no cierto umbral de similitud [66].

Dado que se trata de un primer enfoque al problema y debido a la limitación en número de muestras y recursos de computación, se ha elegido enfocar el sistema desde una perspectiva de verificación de conjunto cerrado, evitando así una costosa comparación del tipo "1 a N" realizada en problemas de identificación.

La similitud siempre ha sido un aspecto clave en computación y estadística. Muchos son los enfoques de similitud diferentes que permiten comparar dos vectores (distancia euclidea, coeficiente de correlación de Pearson, coeficiente de correlación de rangos de Spearman...). Sin embargo, ante muestras de datos más complejas como las disponibles en el presente estudio, estas medidas son inadecuadas. En estos casos, desde su aparición en 1994 para detectar firmas fraudulentas [67], una red neuronal siamesa puede ser la mejor opción [68].

Una red siamesa consta de dos redes neuronales artificiales idénticas que comparten pesos y emplean la retropropagación durante el entrenamiento para actualizar sus pesos. Ambas redes trabajan en paralelo y comparan sus salidas al final a través de una métrica de similitud como la distancia euclídea o la "similitud coseno" [68].

Uno de los primeros modelos con estructura siamesa surge para verificación facial y sigue una estructura que se ha mantenido a lo largo de los años bajo su forma básica (Figura 5.8): una parte simétrica e independiente junto con una comparación entre ambas que produce una métrica de similitud [69]. En concreto, X_1 y X_2 representan las imágenes de entrada, $G_w(X_1)$ y $G_w(X_2)$ son los vectores de características obtenidos a partir de cada red, y E_w representa la métrica de similitud obtenida.

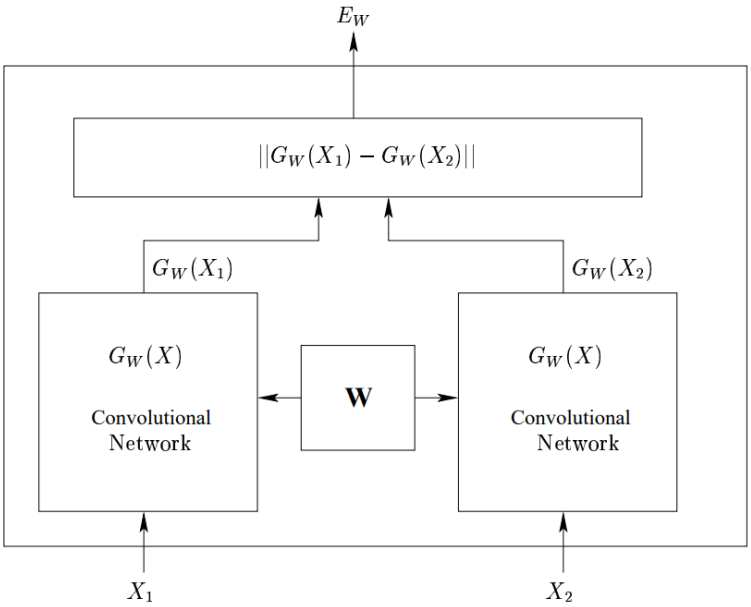


Figura 5.8: Arquitectura genérica de una red siamesa [69]

En línea con el enfoque orientado a verificación y con estas ideas planteadas, se ha propuesto el uso de una red neuronal convolucional siamesa, para la cuál es preciso también generar las proyecciones 2D que sirvan de entrada a la red.

Tras este largo proceso de documentación, se ha decidido tomar como referencia la red siamesa "SigNet" descrita por Dey Sounak, et al. en [70]. Este modelo ha sido diseñado con la finalidad de identificar falsificaciones de firmas, por lo que aunque la finalidad no es la misma, la estructura puede ser válida para el problema planteado. De hecho "SigNet" a su vez, se inspira en la arquitectura propuesta por Krizhevsky et al. [71] orientada a reconocimiento de imágenes.

5.2.1. Descripción del algoritmo

Optimización y función de pérdida

El algoritmo utiliza un optimizador Adam (explicado previamente en la sección 2.3.2) y una función de pérdida contrastiva (*contrastive loss*) definida de la siguiente manera:

$$L(s_1, s_2, y) = \alpha(1-y)D_w^2 + \beta y \,\text{máx}(0, m - D_w)^2 \tag{5.1}$$

donde "s_1" y "s_2" representan los vectores de muestras, "y" la etiqueta que determina si las dos muestras pertenecen a la misma clase ($y = 0$) o no ($y = 1$), m es el margen entre clases, α y β son dos constantes y por último "D_w" es la distancia euclídea definida como:

$$D_w = \|f(s_1; w_1) - f(s_2; w_2)\|_2 \tag{5.2}$$

El cálculo de la pérdida contrastiva cuenta en realidad con 2 ecuaciones diferentes que proporcionan valores de pérdida lógicos dependiendo de si el individuo es un impostor o no:

1. **Si el individuo es quien dice ser ($y = 0$):** $L(s_1, s_2, y) = \alpha \cdot D_w^2$. Esto quiere decir que las representaciones que estén lejos entre sí en el espacio vectorial van a ser penalizadas (dado que siendo el mismo individuo deberían estar cercanas) elevando las pérdidas.
2. **Si el individuo es un impostor ($y = 1$):** $L(s_1, s_2, y) = \beta \cdot \text{máx}(0, m - D_w)^2$. En este caso, Si la distancia euclídea es menor que el margen (es decir las representaciones no se encuentran lo suficiente alejadas), se aplica una penalización positiva al término mientras que de lo contrario la pérdida será 0. Esto significa que si las representaciones de ejemplos de diferentes individuos están demasiado cerca, la pérdida aumentará.

Los parámetros α y β establecen una relación de compromiso en el cálculo de la pérdida. En concreto, permiten controlar la importancia de la pérdida correspondiente a ejemplos del mismo individuo (α más grande) o de diferentes individuos (β más grande). Es decir, si se busca un modelo que se centre más en aprender representaciones para ejemplos del mismo individuo (verdaderos positivos), es posible aumentar α y/o

disminuir β. Análogamente, aumentar β y/o disminuir α consigue un modelo donde la separación de ejemplos de diferentes individuos sea mejor (verdaderos negativos).

Por último, el margen entre clases "m" actúa únicamente ante ejemplos de diferentes individuos y controla cómo de lejos deben estar las representaciones para que la diferencia máx$(0, m - D_w)^2$ sea 0 o aumente. Un m grande requiere de representaciones de diferentes individuos muy distintas, mientras que un m pequeño permite representaciones de diferentes individuos más cercanas facilitando el entrenamiento.

Arquitectura

El algoritmo programado en python y ejecutado en "Google Colab" cuenta con una arquitectura similar a la descrita en [71]. A continuación se describe esta arquitectura (Figura 5.9):

1. Capas convolucionales + ReLU
    a) Primera capa de convolución (96 kernels de tamaño 11x11) con ReLU (representada en azul)
    b) Segunda capa de convolución (256 kernels de tamaño 5x5) con ReLU (representada en verde)
    c) Tercera capa de convolución (384 kernels de tamaño 3x3) con ReLU (la primera representada en azul turquesa)
    d) Cuarta capa de convolución (256 kernels de tamaño 3x3) con ReLU (la segunda representada en azul turquesa)
2. Capas de normalización de la respuesta local (representadas en morado)
3. Capas de *dropout* (representadas en gris)
4. Capas *max pooling* (2x2) (en marrón)
5. Capa *fully connected* + ReLU + *dropout* (en rojo)
6. Capa *fully connected* + ReLU (en naranja)

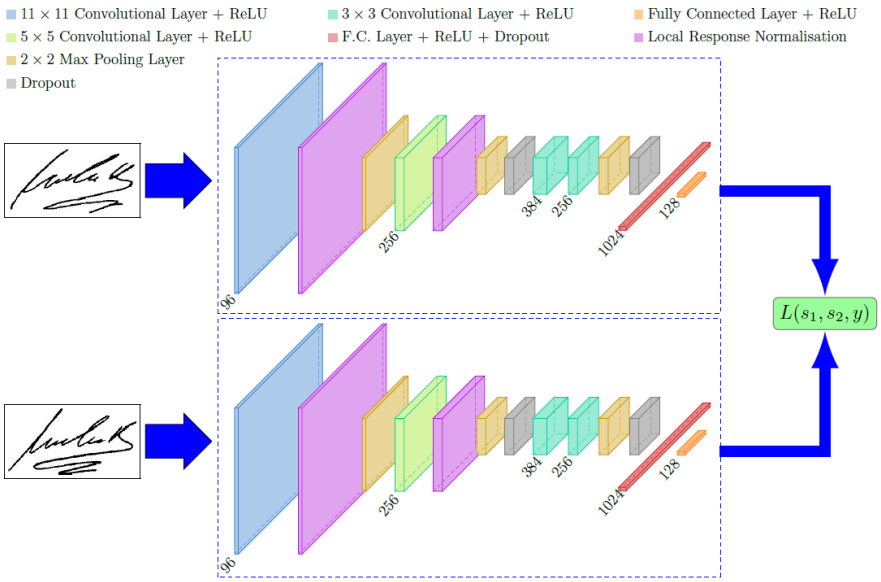


Figura 5.9: Arquitectura del modelo planteado en [70]

No obstante, algunos cambios se han realizado sobre la arquitectura descrita en el *paper*. En base a correcciones aplicadas posteriormente por el autor, se sustituyen las capas de normalización de la respuesta local por capas de normalización por lotes o *batch normalization* y el tamaño de las capas de *max pooling* se incrementa a 3x3.

5.2.2. Hiperparámetros configurables

Aparte de modificar la arquitectura, son varios los hiperparámetros que se pueden ajustar para mejorar el desempeño de la red. A modo de resumen:

- *Número de épocas*: Un número insuficiente de épocas puede llevar a cometer *underfitting*, mientras que si es demasiado grande el modelo puede ajustarse excesivamente a los datos produciendo *overfitting*.
- *Tamaño del lote*: Un tamaño de lote más grande puede acelerar el entrenamiento, pero también puede requerir más memoria. Un tamaño de lote más pequeño puede ayudar a la convergencia, pero puede ser más lento
- *Tasa de aprendizaje "lr" [optimizador]*: Como en cualquier optimizador, aumentar el paso o en este caso la tasa de aprendizaje acelera la convergencia aunque incrementa las posibilidades de no converger. Por el contrario, disminuir "lr" permite una convergencia más estable aumentando el tiempo necesario para ello. A mayores, se debe tener en cuenta que un "lr" demasiado bajo podría provocar que el optimizador quede atrapado en óptimos locales.
- α *[función de pérdida]*: Como se ha comentado en la sección 5.2.1, un α más grande da más importancia a los errores ante ejemplos del mismo individuo respecto a errores cometidos en ejemplos de diferentes individuos.

- *β [función de pérdida]*: Al igual que en el caso anterior, como se discute en la sección 5.2.1, un β más alto da más importancia a los errores ante ejemplos del distinto individuo respecto a errores cometidos en ejemplos del mismo individuo.
- *Margen "m" [función de pérdida]*: Se discute en la sección 5.2.1. En resumen, un "m" menor permite representaciones de diferentes individuos más cercanas, mientras que un "m" grande requiere de representaciones más distintas para reconocerlos como diferentes.
- *Decaimiento de pesos [optimizador]*: El decaimiento de pesos o *Weight Decay* actúa como mecanismo de regularización sobre los pesos de la red. De esta manera, aumentar el valor ayuda a combatir el sobreajuste mientras que disminuirlo se traduce en aplicar una regularización más suave y por ende permitir un mejor ajuste a los datos de entrenamiento.

5.3. Obtención de sets de datos

5.3.1. Generación de imágenes 2D

El entrenamiento de una red neuronal generalmente necesita imágenes del mismo tamaño y de ser factible, cuadradas. De esta manera, aunque antes de pasar las imágenes por la red se realiza un preprocesado que ajusta el tamaño de las imágenes a 105x105, se han generado imágenes de dimensiones cuadradas a partir de los datos 3D disponibles. Además, la red emplea imágenes monocroma o de un sólo canal de tal forma que los datos de entrada deben contar con unas dimensiones de 105x105x1.

Para obtener las representaciones 2D, se han realizado proyecciones de tipo MIP (*Maximum Intensity Proyection*) sobre los 3 ejes de coordenadas para cada una de las frecuencias. En un inicio, se planteó incluir en la misma imagen información de las 3 frecuencias generando así una imagen RGB de 3 canales. Sin embargo, finalmente se han utilizado las muestras generadas en cada frecuencia por separado. Como la red toma como entrada imágenes de un solo canal, emplear imágenes en color obligaría a incluir una etapa de preprocesado que las transforme en monocroma. Esta transformación no es óptima ya que tiene en cuenta, bajo diferentes pesos según la percepción humana, las aportaciones de cada canal y por ende de cada frecuencia $Y = 0{,}299R + 0{,}587G + 0{,}114B$. Aunque no se haya empleado, una forma de tener en cuenta bajo los mismos pesos la información de las 3 frecuencias en la misma imagen consiste simplemente en calcular para cada captura el promedio de las 3 imágenes (f_1, f_2 y f_3).

Respecto al número de proyecciones incluidas a la hora de construir las imágenes, 2 opciones se han planteado:

1. **Utilizar las 3 proyecciones (Figura 5.10):** De esta manera se incluye más información aunque el tamaño de la imagen es mayor (222x222 píxeles) por lo que se debe redimensionar en preprocesado, perdiendo así cierta resolución. Además, puede que parte de la información incluida en las distintas proyecciones pueda ser redundante entre sí.

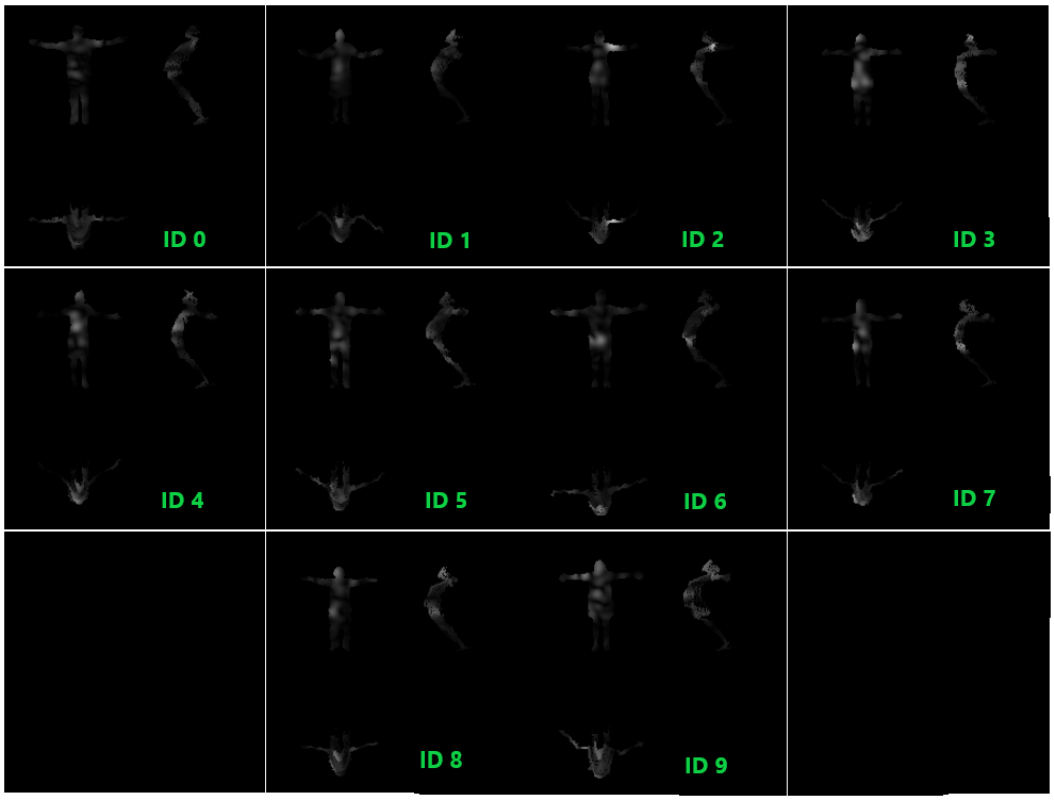


Figura 5.10: Imágenes 2D generadas empleando 3 proyecciones

2. **Utilizar únicamente la proyección frontal (Figura 5.11):** Con el objetivo de reducir gran parte de la información a la entrada se selecciona una única proyección que contenga el máximo de información posible, esto es la proyección frontal. De esta manera se construyen imágenes de tamaño 91x91 píxeles que además de no perder resolución al ser redimensionadas, abren la posibilidad de modificar la arquitectura para reducir el número de datos a la entrada simplificando así el modelo.

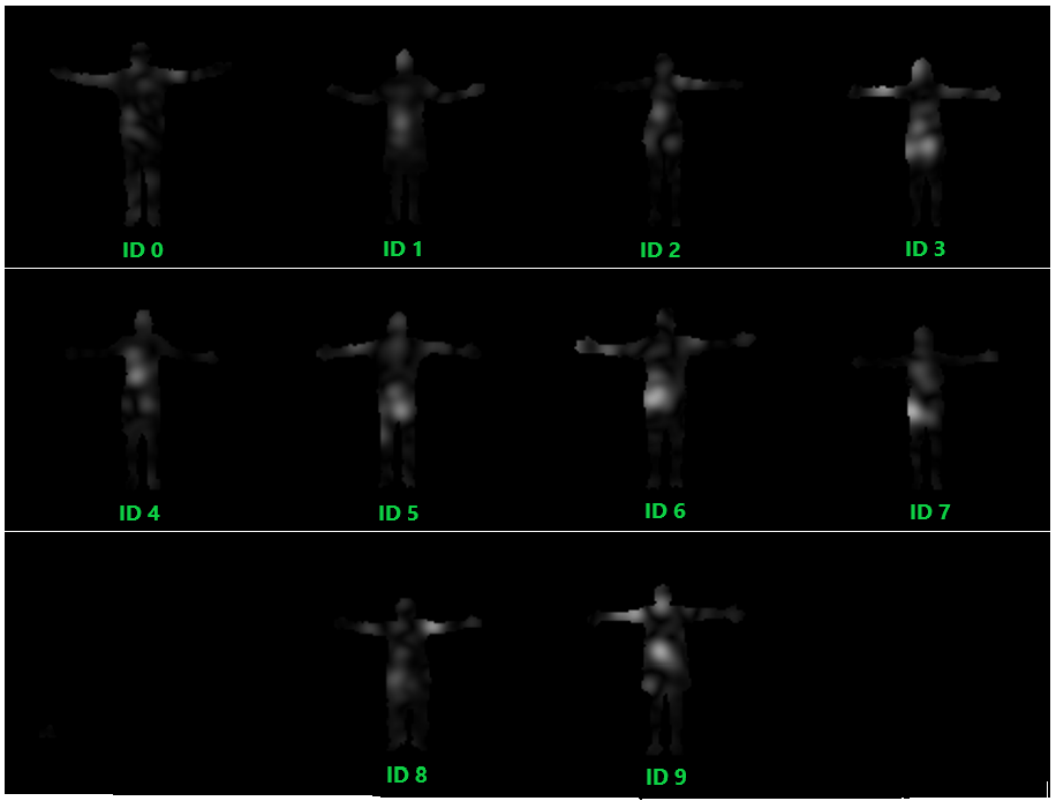


Figura 5.11: Imágenes 2D generadas empleando la proyección frontal

Otro aspecto relevante al construir las imágenes es asegurar el rango de valores. Al trabajar con imágenes en formato de 8 bits, se utiliza el rango [0,255] por lo que todos los valores de los píxeles deben estar comprendidos en ese rango. Sin embargo, los valores de energía capturados en las muestras sobrepasan en muchos casos estos valores. Una solución clásica consistiría en normalizar cada imagen dividiendo entre su valor de energía máximo. De esta forma, todas las imagenes contarían con valores comprendidos entre 0 y 255 aprovechando así todo el rango disponible.

Sin embargo, esta opción tiene un grave inconveniente para el caso de estudio. Tal como se ha demostrado en la sección 5.1, la información de energía absoluta es relevante y puede ayudar a identificar a los individuos por lo que perderla sería perder información importante. Por ejemplo, normalizar todas las imágenes por su valor máximo no permitiría al modelo saber si una persona típicamente refleja energías más altas que otra.

No obstante, existe una relación de compromiso acorde a este tema puesto que normalizar por el valor máximo posible registrable en una captura limitaría la gran mayoría de las imágenes a utilizar una parte pequeña del rango disponible. De esta forma, se ha tomado la decisión de calcular para cada individuo y cada frecuencia su percentil 95 de valor máximo de energía (Tabla 5.1) y se ha escogido el mayor de ellos como factor de normalización. De esta manera, se asegura que en el 95 % de las imágenes de cada individuo no se satura la imagen, reduciendo así un poco el factor respecto a escoger el máximo absoluto.

No hay que olvidar que estos estadísticos deben ser calculados únicamente empleando

ID	f_1	f_2	f_3
ID0	210.83	278.54	464.78
ID1	211.21	291.22	378.23
ID2	282.77	384.79	489.33
ID3	325.57	453.64	542.91
ID4	154.89	228.55	330.00
ID5	202.21	309.63	351.98
ID6	291.22	406.75	563.48
ID7	250.62	293.55	400.33
ID8	230.28	322.20	411.98
ID9	374.46	504.39	589.10
MAX	374.46	504.39	589.10

Tabla 5.1: Percentiles 95 de amplitud máxima

el conjunto de datos de entrenamiento para mantener la información de los datos de test al margen del entrenamiento. No obstante, una vez calculado el factor, este debe ser aplicado también en el conjunto de prueba para evitar problemas de desalineación.

Como consideración final, en la clasificación de datos 3D, una práctica recurrente consiste en generar datos sintéticos obtenidos a partir de modelos CAD. Objetivamente, es preferible utilizar datos del mundo real; sin embargo, la recopilación de datos reales es en general costosa por lo que los datos sintéticos pueden producir una gran cantidad de datos sin problemas de ruido [63]. No obstante, en este proyecto, generar muestras sintéticas no es tan sencillo como partir de modelos CAD. Dado el carácter novedoso de la fuente de datos empleada, un estudio más profundo sería necesario para poder generar imágenes acústicas de diferentes personas de manera sintética.

5.3.2. Construcción de pares

Primeramente, el conjunto de datos se distribuye entre los conjuntos de entrenamiento, validación y el de test. El primero se emplea en la obtención del clasificador, el segundo sirve para ajustar el modelo y el último se emplea para evaluar su desempeño. Se ha elegido repartir el conjunto de datos en aproximadamente un 60 % de datos de entrenamiento, 20 % de datos de validación y 20 % de datos de test para evitar problemas de sobreajuste, por lo que finalmente se obtiene un *train set* compuesto de 12600 imágenes, y un *test set* y *evaluation set* compuestos cada uno por 4140 imágenes. En ambos casos los ejemplos se encuentran escogidos aleatoriamente para eliminar las dependencias temporales entre imágenes y repartidos de forma igualitaria entre los 10 individuos para no desbalancear ninguna clase.

En realidad, las redes neuronales siamesas pueden dividirse en dos modelos principales según el número de parámetros de la red: pares y tripletes. En caso de usar pares se utiliza típicamente la entropía cruzada binaria y la pérdida contrastiva como funciones de pérdida mientras que al utilizar tripletes es común emplear la función *triplet loss* dado que la entrada consiste en 3 imágenes. En este caso se ha implementado una

arquitectura de red basada en pares ya que se comparan únicamente dos imágenes de entrada a la red. Este tipo de redes siamesas reciben comúnmente el nombre de redes neuronales gemelas y requieren un proceso previo de etiquetado de pares [72].

Este proceso requiere de la construcción de pares positivos (pertenecientes a la misma clase y etiquetados con $y = 0$) y pares negativos (pertenecientes a distintas clases y etiquetados con $y = 1$).

De esta forma, se ha diseñado un método determinista que permite construir estos pares respetando ciertos criterios:

- Mantener el número de pares positivos y negativos equilibrado con el objetivo de obtener un set balanceado entre las dos posibles clases.
- Emplear cada imagen en el mismo número de pares con el fin de no sobre utilizar unas imágenes frente a otras.
- A la hora de construir los pares negativos, comparar cada sujeto el mismo número de veces con cada uno de los otros individuos de forma que las comparaciones negativas no sean siempre entre las mismas personas. Esto permite eliminar la posibilidad de mezclar siempre individuos que casualmente sean físicamente más o menos similares.

El número máximo de pares positivos construidos sin repetición limita el número de pares negativos. No obstante, aunque siempre bajo este límite, el algoritmo diseñado es escalable y según el modo seleccionado permite construir distinto número de pares incrementando en 4 el número de pares en los que se emplea cada imagen. Esto permite variar el número de ejemplos disponibles para la red a coste de incrementar el uso de cada imagen, por ejemplo:

- **Modo 1:**
  - 25200 pares de entrenamiento
  - 8280 pares de evaluación
  - 8280 pares de test
  - cada imagen es utilizada en 4 pares (2 pares positivos y 2 pares negativos)
- **Modo 2:**
  - 50400 pares de entrenamiento
  - 16560 pares de evaluación
  - 16560 pares de test
  - cada imagen es utilizada en 8 pares (4 pares positivos y 4 pares negativos)
- **...**
- **Modo 5:**
  - 126000 pares de entrenamiento
  - 41400 pares de evaluación

- 41400 pares de test
- cada imagen es utilizada en 20 pares (10 pares positivos y 10 pares negativos)

- ...

El algoritmo genera unos archivos en formato ".csv" en los que incluye una lista de los pares generados. No obstante, antes de generar este archivo, se comprueba que ningún par está repetido, que todas las imagenes se han empleado el mismo número de veces y que la relación entre pares positivos y negativos en cada conjunto es del 50 %.

Posteriormente, los archivos generados "train_data.csv", "eval_data.csv" y "test_data.csv" se emplean para cargar el conjunto de datos de entrenamiento, validación y test en la red. Estos archivos contienen en cada línea 3 argumentos, el path relativo de las dos imágenes que constituyen el par y por último una etiqueta que determina si corresponden al mismo individuo (0) o no (1). Un ejemplo del contenido de estos ficheros se muestra a continuación (Figura 5.12):

```
6103 f1/ID09/N832_ID09.png,f1/ID08/N730_ID08.png,1
6104 f1/ID09/N1027_ID09.png,f1/ID08/N1989_ID08.png,1
6105 f1/ID09/N281_ID09.png,f1/ID08/N1033_ID08.png,1
6106 f1/ID09/N203_ID09.png,f1/ID08/N1874_ID08.png,1
6107 f1/ID09/N114_ID09.png,f1/ID08/N1647_ID08.png,1
6108 f1/ID09/N437_ID09.png,f1/ID08/N1589_ID08.png,1
6109 f1/ID09/N1665_ID09.png,f1/ID08/N222_ID08.png,1
6110 f1/ID09/N1668_ID09.png,f1/ID08/N371_ID08.png,1
6111 f1/ID09/N1966_ID09.png,f1/ID08/N1034_ID08.png,1
6112 f1/ID09/N1044_ID09.png,f1/ID08/N190_ID08.png,1
6113 f1/ID09/N218_ID09.png,f1/ID08/N1075_ID08.png,1
6114 f1/ID09/N1654_ID09.png,f1/ID08/N1987_ID08.png,1
6115 f1/ID09/N565_ID09.png,f1/ID08/N1891_ID08.png,1
6116 f1/ID09/N146_ID09.png,f1/ID08/N076_ID08.png,1
6117 f1/ID09/N951_ID09.png,f1/ID08/N491_ID08.png,1
6118 f1/ID09/N1454_ID09.png,f1/ID08/N1254_ID08.png,1
6119 f1/ID09/N1724_ID09.png,f1/ID08/N981_ID08.png,1
6120 f1/ID09/N2070_ID09.png,f1/ID08/N253_ID08.png,1
6121 f1/ID00/N917_ID00.png,f1/ID00/N1564_ID00.png,0
6122 f1/ID00/N1244_ID00.png,f1/ID00/N917_ID00.png,0
6123 f1/ID00/N1406_ID00.png,f1/ID00/N1244_ID00.png,0
6124 f1/ID00/N1147_ID00.png,f1/ID00/N1406_ID00.png,0
6125 f1/ID00/N554_ID00.png,f1/ID00/N1147_ID00.png,0
6126 f1/ID00/N131_ID00.png,f1/ID00/N554_ID00.png,0
6127 f1/ID00/N544_ID00.png,f1/ID00/N131_ID00.png,0
6128 f1/ID00/N1303_ID00.png,f1/ID00/N544_ID00.png,0
6129 f1/ID00/N107_ID00.png,f1/ID00/N1303_ID00.png,0
```

Figura 5.12: Lista de algunos pares contenidos en "train_data.csv"

Por último, se ha verificado que los pares y sus respectivas etiquetas se cargan correctamente al ejecutar el algoritmo visualizando un *batch* de 8 elementos (Figura 5.13).

```
[[0.]
 [0.]
 [1.]
 [1.]
 [0.]
 [0.]
 [1.]
 [0.]]
```

Figura 5.13: *batch* de 8 pares de imágenes con sus respectivas etiquetas

Capítulo 6. Resultados

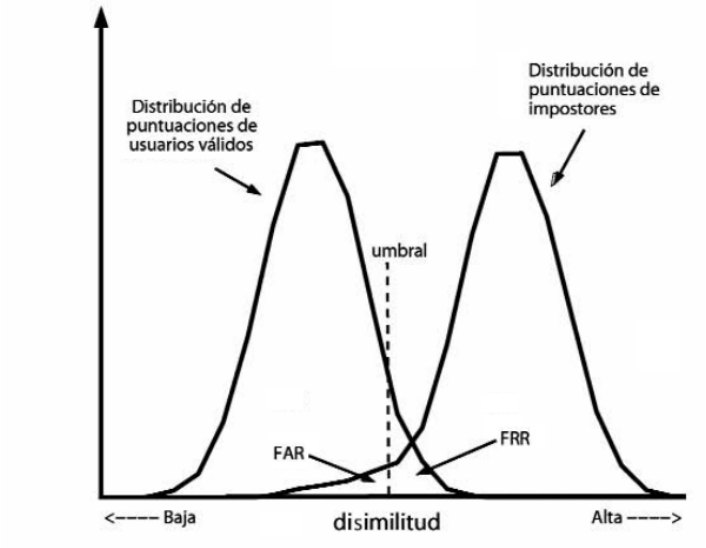
6.1. Consideraciones previas

Para obtener los resultados, se han entrenado los modelos en "Google Colab" utilizando una GPU Tesla T4 que cuenta con 15 GB de RAM. Esta restricción representa una limitación, ya que no es posible realizar entrenamientos que requieran una capacidad de RAM superior. De hecho, la mayoría de los entrenamientos ejecutados ya consumen prácticamente la totalidad de la memoria disponible (Figura 6.1).

Figura 6.1: Uso de RAM durante un entrenamiento con Google Colab

La salida del sistema biométrico es la distancia euclídea entre las características obtenidas para cada par de ejemplos, es decir una puntuación de disimilitud. Esto es una salida común en sistemas biométricos en modo verificación ya sea a través de una puntuación de similitud o disimilitud [66]. La distribución de puntuaciones generadas por los pares positivos se denomina distribución de usuarios válidos mientras que la distribución de puntuaciones generadas por pares negativos recibe el nombre de distribución de impostores.

En un sistema ideal, las puntuaciones obtenidas para usuarios válidos e impostores son perfectamente separables mediante un umbral de decisión dado que no existe solapamiento entre sus distribuciones [66]. No obstante, en un sistema real existe siempre una región en la que ambas distribuciones se solapan (Figura 6.2).



De esta manera, el desempeño del algoritmo cambia en función de la elección del umbral. Todas las puntuaciones de disimilitud por debajo del umbral se consideran como pares positivos (baja disimilitud) mientras que los casos por encima del umbral se consideran pares negativos (elevada disimilitud). Sin embargo, ante un caso real en el que las distribuciones no son perfectamente separables, existe parte de la distribución de impostores que queda debajo del umbral etiquetándose erróneamente como par positivo, este área se conoce como Tasa de Falsa Aceptación (FAR). Lo mismo ocurre cuando parte de la distribución de usuarios válidos sobrepasa el umbral determinando así la Tasa de Falso Rechazo (FRR)

Finalmente, los modelos descritos en las secciones subsiguientes han sido entrenados mediante la utilización de pares generados empleando el "modo 1" y empleando proyecciones MIP para la obtención de las imágenes 2D.

6.2. Modelo inicial

Inicialmente se ha empleado la arquitectura planteada en [70] y descrita en la sección 5.2.1. Por otra parte, los hiperparámetros empleados (Tabla 6.1) son los que se describen en el artículo a excepción del tamaño del lote. El tamaño del lote es 144 dado que los datos no son divisibles por 128 y las librerías de Pytorch empleadas esperan que los datos estén organizados en lotes del mismo tamaño para el procesamiento eficiente.

Parámetro	Valor
Tasa de aprendizaje (lr)	$1e-4$
Tamaño del lote	144
Número de épocas	20
Weight Decay	0.0005
Margen	1
α	1
β	1

Tabla 6.1: Hiperparámetros empleados en el modelo inicial

Tras el entrenamiento, se ha diseñado un script para analizar los resultados y se ha evaluado el funcionamiento del modelo representando las distribuciones de usuarios válidos (pares positivos) e impostores (pares negativos) obtenidas empleando los datos de test (Figura 6.3).

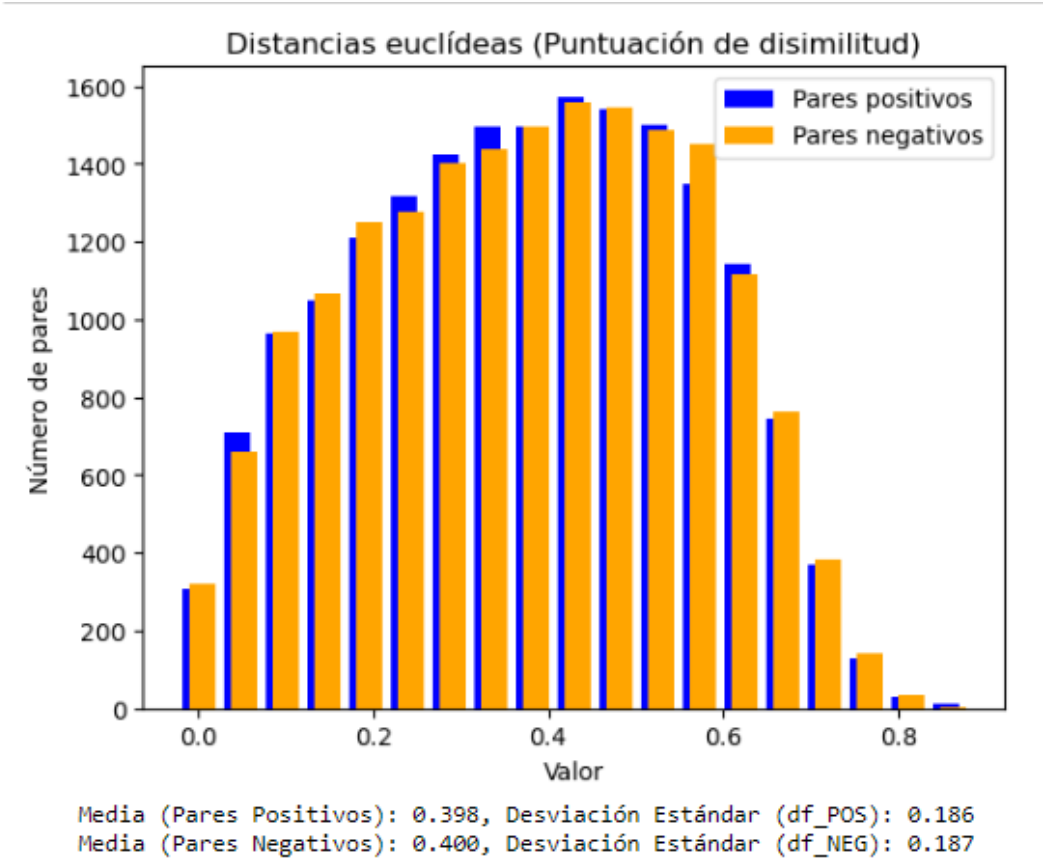


Figura 6.3: Distribuciones de disimilitud en datos de test para el modelo inicial

Estas distribuciones no tienen nada que ver con lo esperable y descrito en el apartado 6.1 dado que las distribuciones son prácticamente idénticas y por ende la tasa de falsa aceptación y de falso rechazo serán muy elevadas independientemente del umbral escogido.

Ante esta problemática, se representa también la evolución de las pérdidas de entrenamiento y validación al entrenar el modelo con 100 épocas (Figura 6.4).

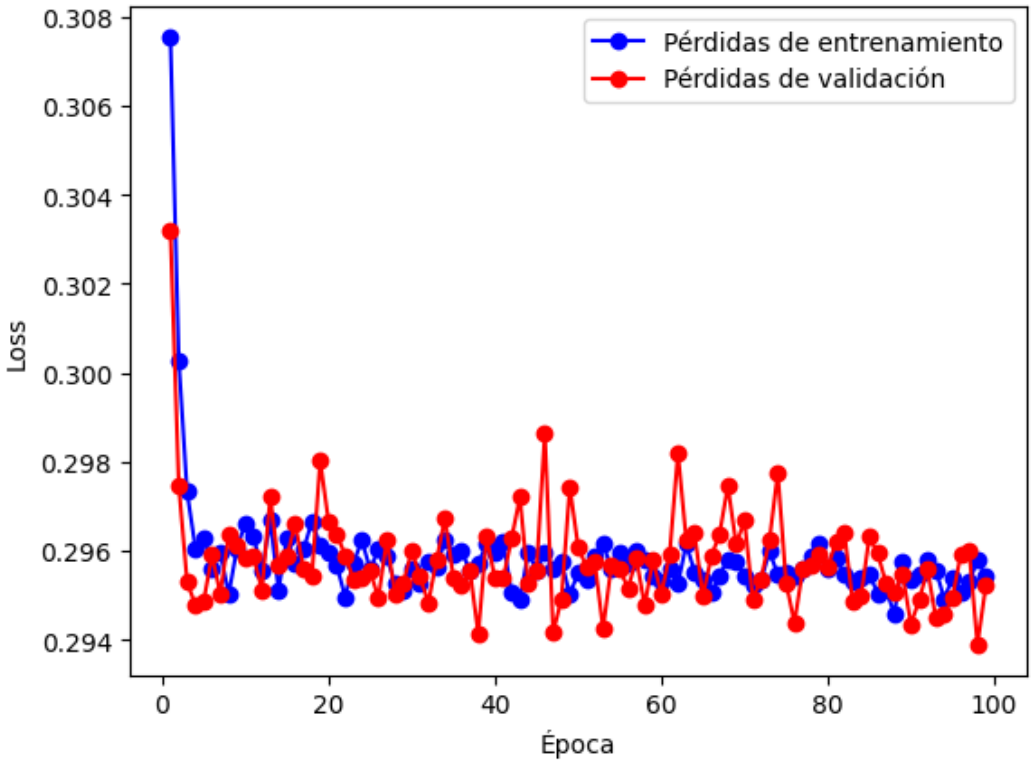


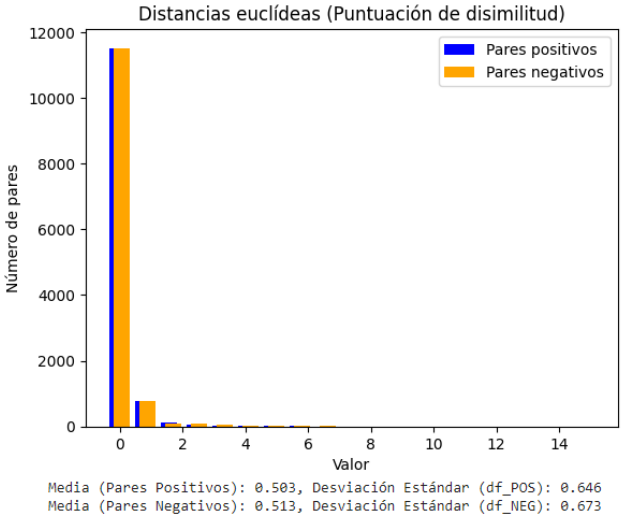
Figura 6.4: Pérdida contrastiva de entrenamiento y validación registrada por época

En la gráfica se observa que a partir de las primeras épocas, las pérdidas sobre el conjunto de entrenamiento y sobre el conjunto de validación exhiben cierta estabilidad. Es decir, sin experimentar variaciones significativas a la alza o a la baja.

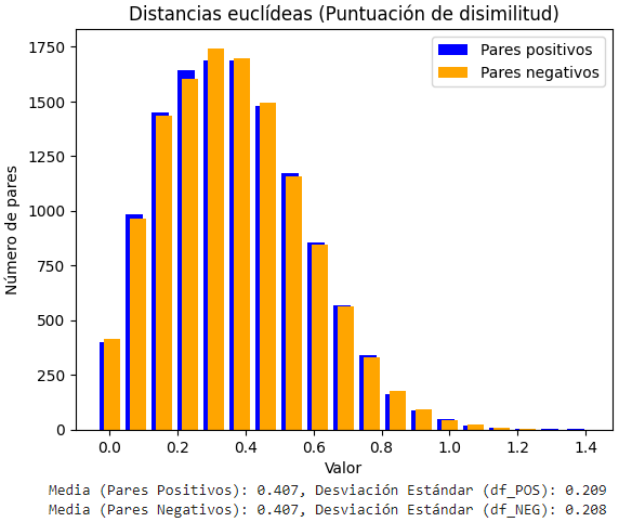
Ante este resultado, es posible descartar la posibilidad de que el modelo se esté sobreajustando a los datos de entrenamiento, ya que, en caso contrario, las pérdidas sobre el conjunto de entrenamiento disminuirían con el número de épocas mientras que las del conjunto de validación aumentarían.

De este modo, para comprender mejor el problema, se ha reentrenado la red sobre un menor número de épocas y pares analizando las distribuciones que el modelo predice sobre los datos de entrenamiento en cada época (Figura 6.5). El razonamiento consiste en comprobar si el modelo es capaz de clasificar correctamente los datos de entrenamiento ya que de lo contrario, su capacidad de generalización a nuevos datos sería limitada justificando así los resultados insatisfactorios obtenidos sobre el conjunto de test.

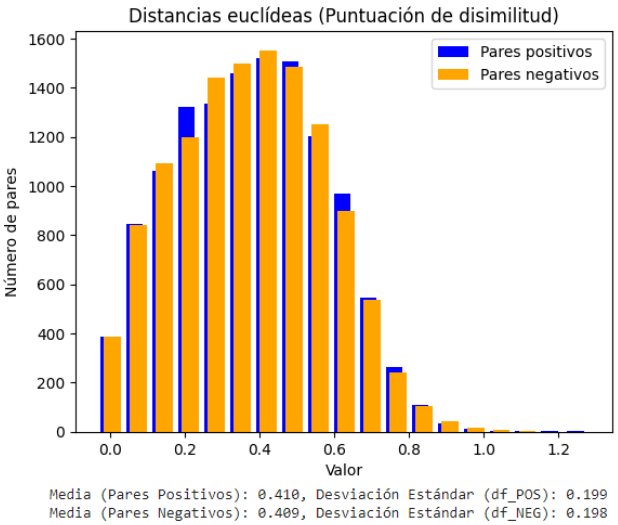
Las representaciones gráficas presentadas (Figura 6.5) respaldan la hipótesis formulada. En todos los casos, independientemente del número de épocas, las distribuciones se encuentran virtualmente superpuestas por lo que el modelo no adquiere de manera adecuada el conocimiento inherente a los datos de entrenamiento.



(a) Tras época 1



(b) Tras época 2



(c) Tras época 10

Figura 6.5: Distribuciones de disimilitud en entrenamiento para el modelo inicial

Estos resultados, además de proporcionar una explicación de los resultados observados, indican que se trata de un problema de subajuste (*underfitting*). Esto sugiere que la arquitectura propuesta en [70] no presenta la suficiente complejidad requerida para abordar el problema planteado en este estudio.

Finalmente, es pertinente señalar que también se han modificado los hiperparámetros descritos en la Tabla 6.1 y el número de pares empleados sin observar cambios significativos en los resultados.

6.3. Evolución del modelo

6.3.1. Primera aproximación

Considerando que el modelo no presenta riesgo de sobreajuste, a modo de enfoque inicial, se han eliminado las capas de *dropout* que actúan como regularización. Esta acción posibilita un incremento en la complejidad del modelo por lo que si existe verdaderamente un problema de subajuste, esta modificación puede conducir a mejoras en los resultados. Además, se conservan los hiperparámetros inicialmente especificados (Tabla 6.1) con el propósito de evaluar de manera aislada el impacto en la modificación de la arquitectura.

Empleando este nuevo modelo se obtienen los siguientes resultados en validación (Figura 6.6) y en entrenamiento (Figura 6.7).

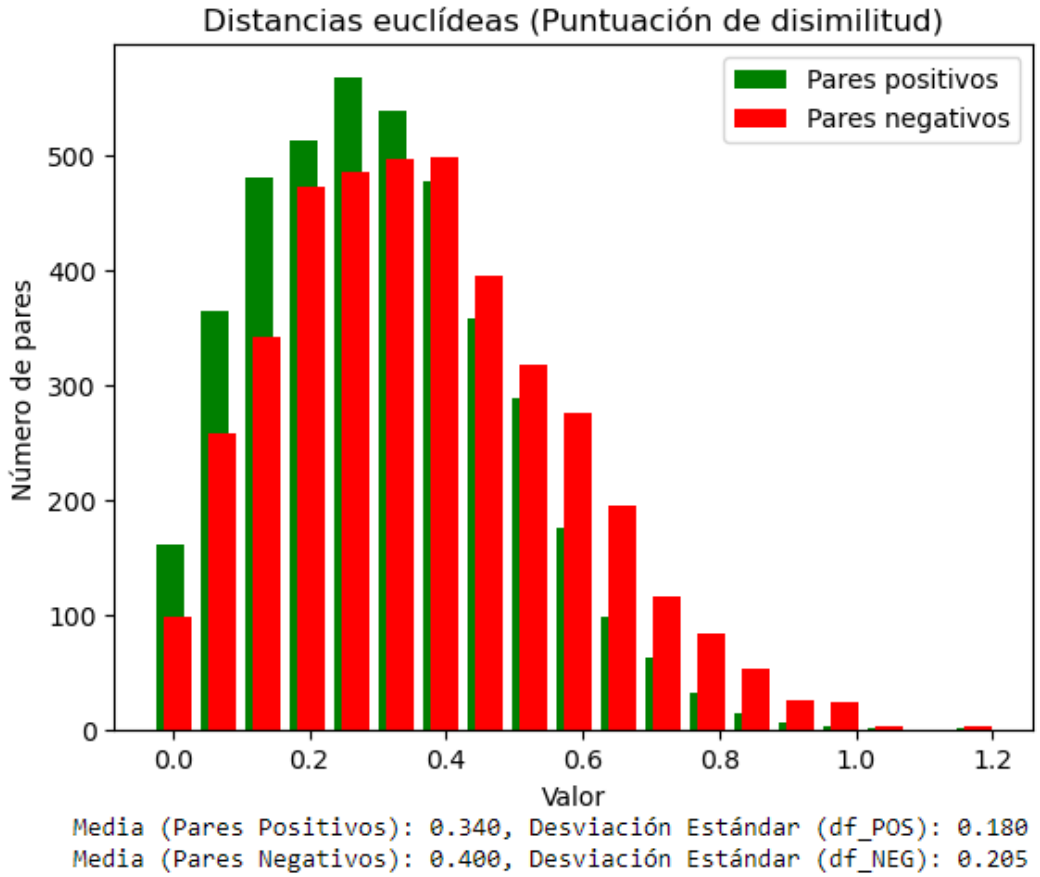
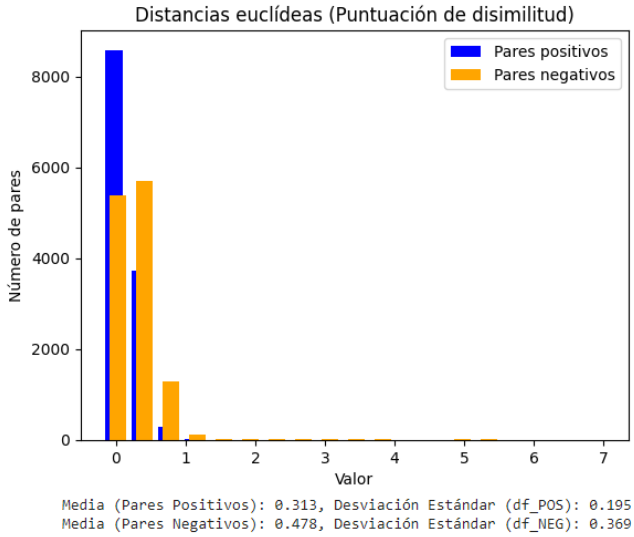
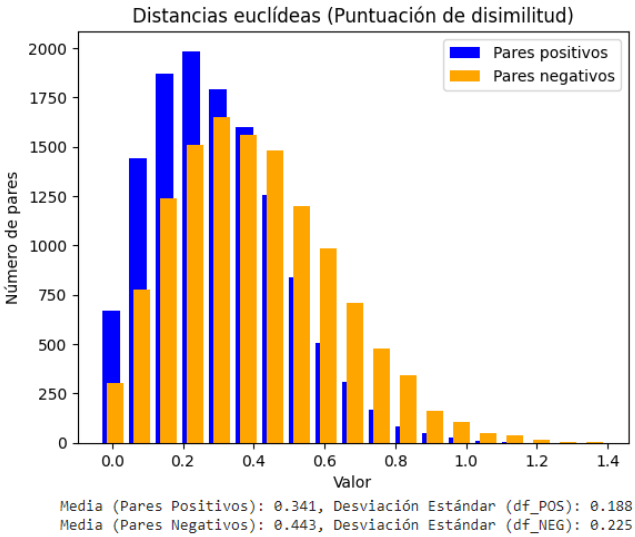


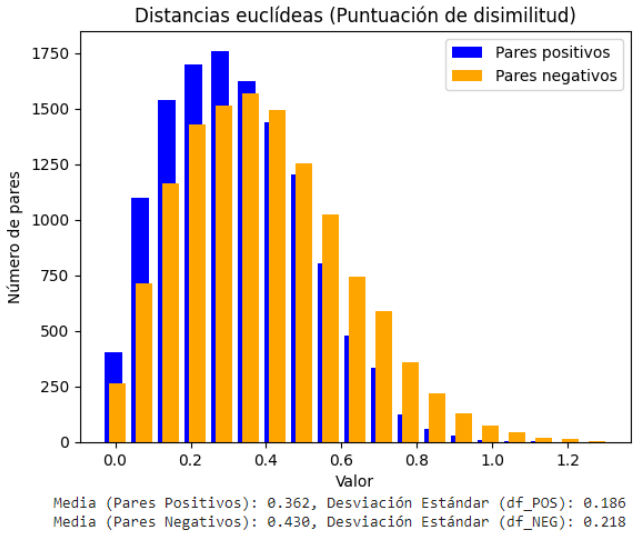
Figura 6.6: Distribuciones de disimilitud en datos de validación para el modelo inicial



(a) Tras época 1



(b) Tras época 2



(c) Tras época 10

Figura 6.7: Distribuciones de disimilitud en entrenamiento para el modelo sin *dropout*

Si bien estos resultados muestran un modelo incapaz de distinguir con precisión a los impostores de los usuarios legítimos, sí contribuyen a corroborar la conclusión expuesta en la sección previa. Al aumentar ligeramente a complejidad mediante la eliminación de las capas de "dropout", se aprecia una mejora en los resultados de entrenamiento que además se refleja en una mejora de los resultados de validación.

6.3.2. Mejor caso

Evaluando distintas combinaciones de hiperparámetros e incluso ligeros cambios en la arquitectura, los mejores resultados se han obtenido empleando los siguientes hiperparámetros (Tabla 6.2):

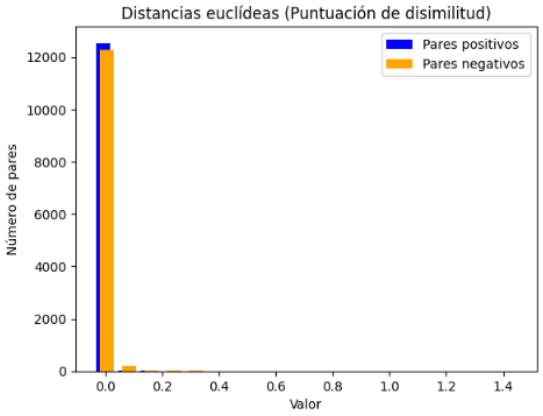
Parámetro	**Valor**
Tasa de aprendizaje (lr)	$1e-4$
Tamaño del lote	144
Número de épocas	10
Weight Decay	0.0005
Margen	0.01
α	1
β	1

Tabla 6.2: Hiperparámetros empleados en el mejor modelo

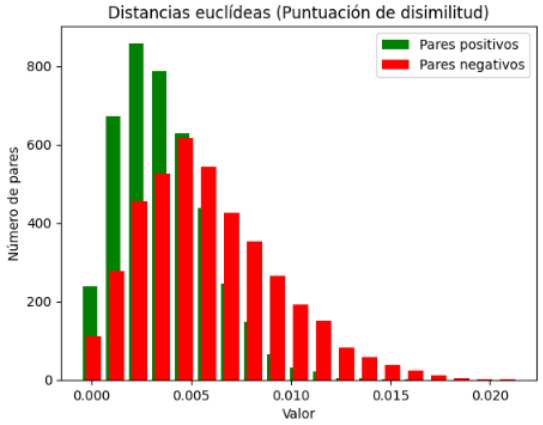
Con excepción del margen y el número de épocas, los demás parámetros son fundamentalmente idénticos. Los ajustes más significativos se han producido al modificar el valor del margen. Como se detalla en la sección 5.2.1, la reducción del margen permite representaciones de diferentes individuos más cercanas lo que mejora el rendimiento dado el nivel de complejidad y la semejanza inherente de los datos.

Los resultados obtenidos se detallan a continuación:

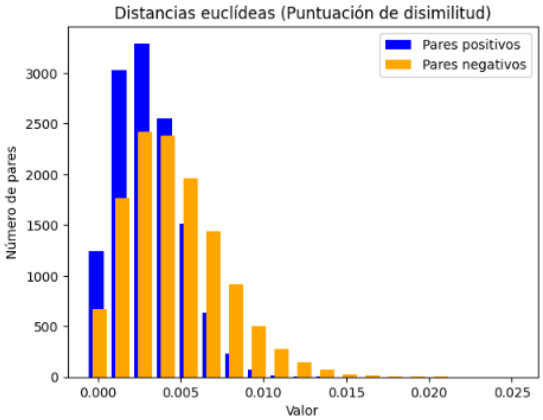
- Distribuciones de disimilitud obtenidas sobre los datos de entrenamiento y validación por épocas (Figura 6.8).
- Distribuciones obtenidas sobre los datos test (Figura 6.9).
- Curva ROC (Figura 6.10).
- Precision(umbral) y Recall(umbral) (Figura 6.11).
- Curva Precision-Recall (Figura 6.12).
- Valor-F1(umbral) (Figura 6.13).



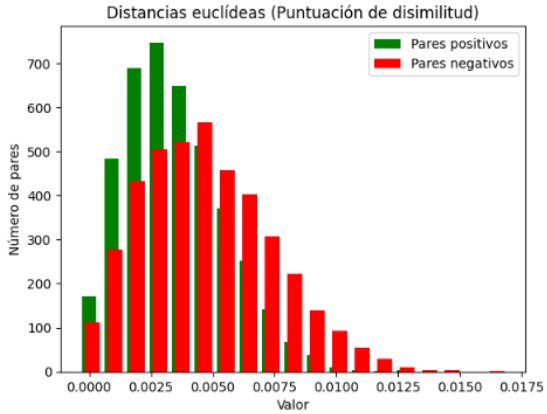
(a) Tras época 1 (entrenamiento)



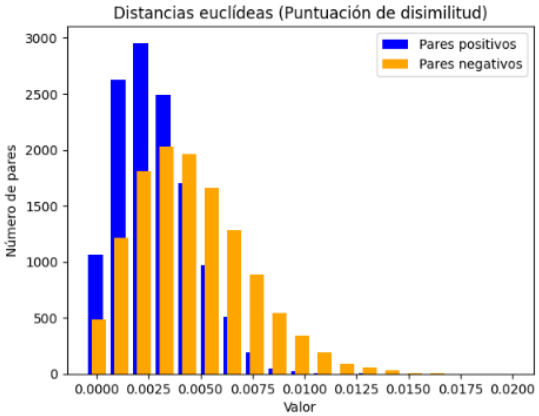
(b) Tras época 1 (validación)



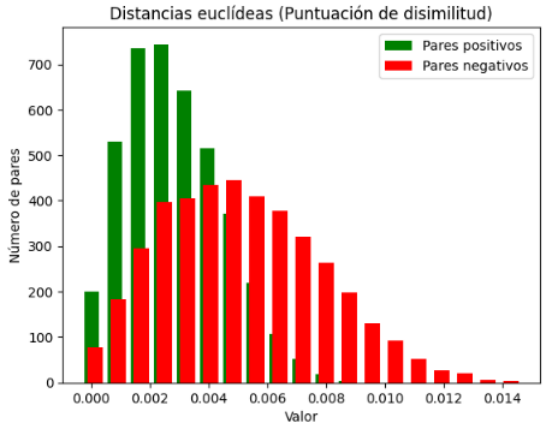
(c) Tras época 2 (entrenamiento)



(d) Tras época 2 (validación)



(e) Tras época 10 (entrenamiento)



(f) Tras época 10 (validación)

Figura 6.8: Distribuciones de disimilitud para el mejor modelo

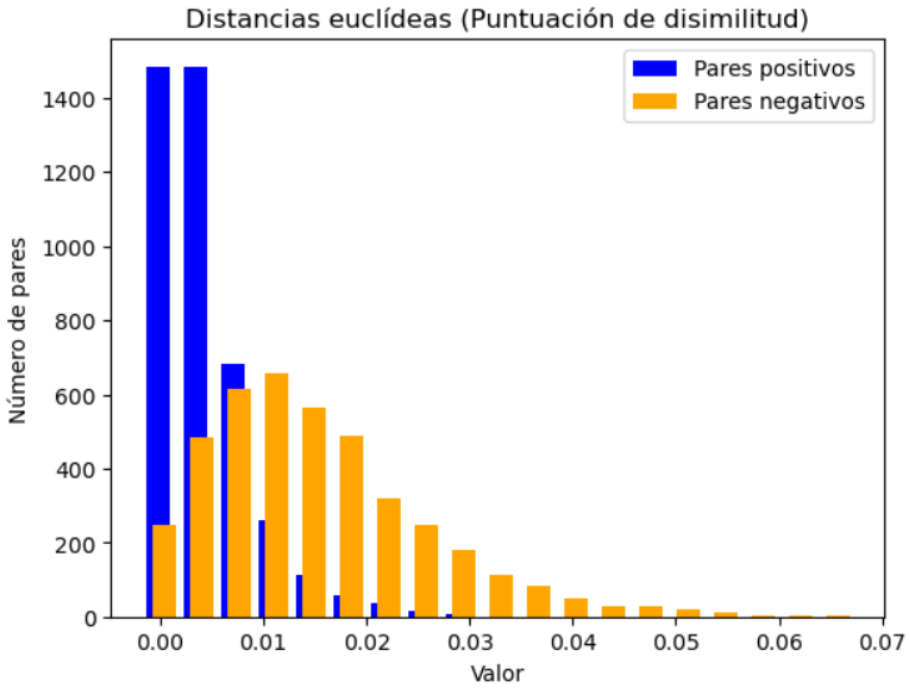


Figura 6.9: Distribuciones de disimilitud para el mejor modelo sobre los datos de test

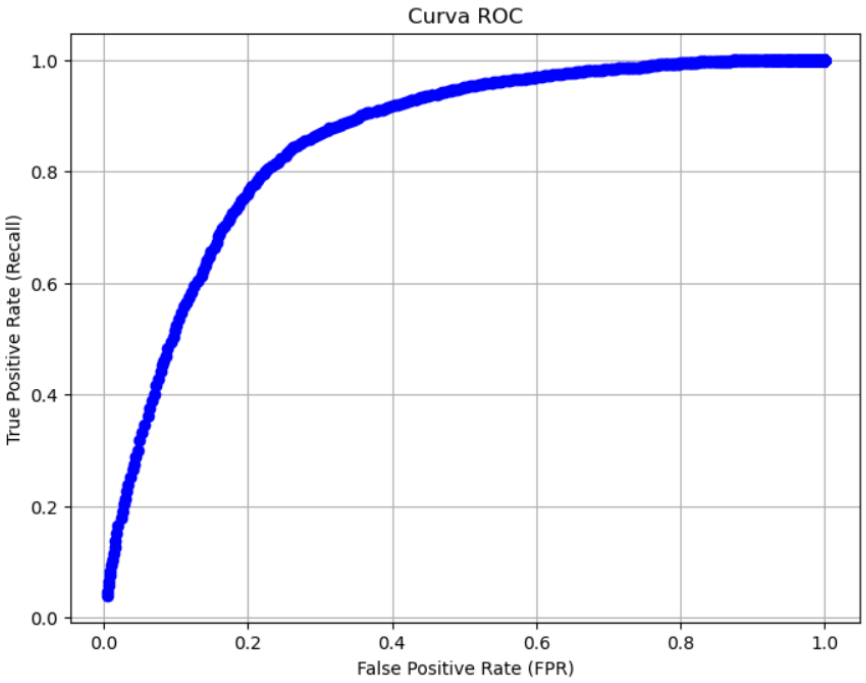


Figura 6.10: Curva ROC

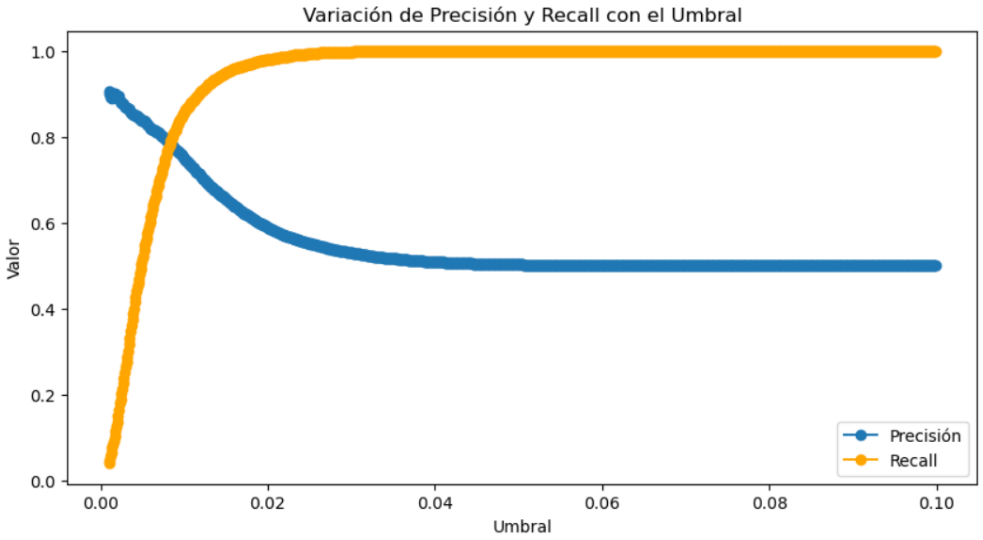


Figura 6.11: Curvas de precisión (umbral) y recall (umbral)

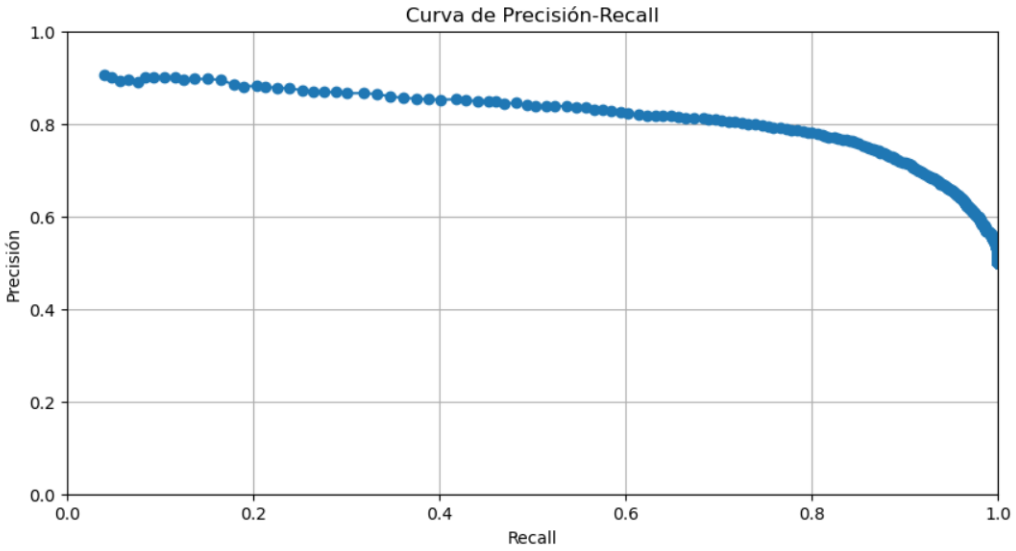


Figura 6.12: Curva precision-recall

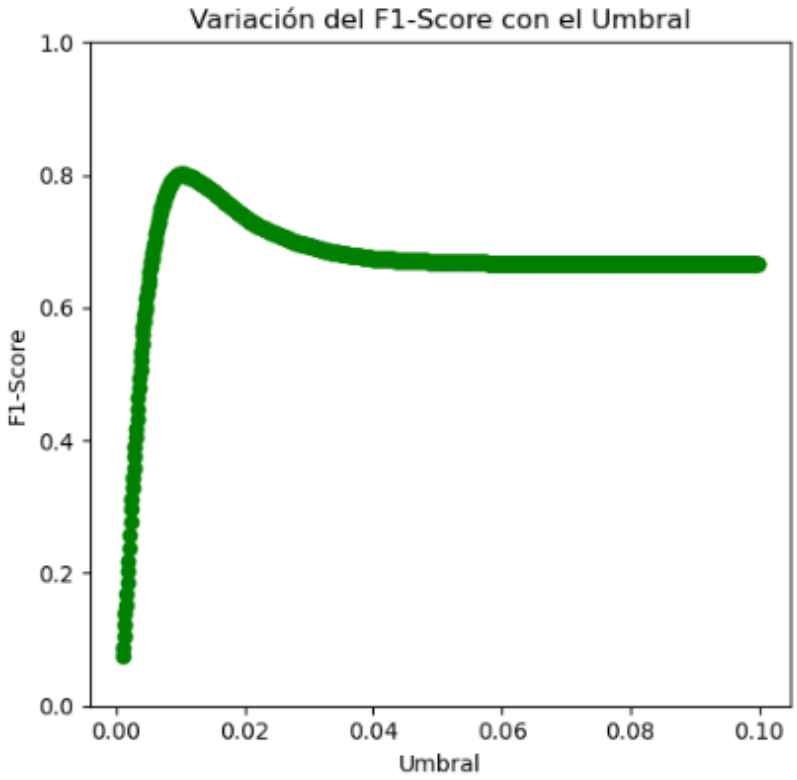


Figura 6.13: F1-Score (umbral)

Con el propósito de proporcionar una visión más concisa acerca de los resultados obtenidos, se ha escogido el umbral de 0.0102 el cual corresponde al punto donde el valor-F1 alcanza su valor máximo de 0.801. Por lo tanto, para este valor específico del umbral, se obtienen las siguientes métricas:

- Precisión = 0.746
- Recall = 0.865
- Accuracy = 0.785
- Error rate = 0.215
- Specificity = 0.705

Además, se ha considerado la inclusión de un ejemplo representativo junto con estos resultados. A continuación se muestran 2 pares de imágenes seleccionados aleatoriamente del conjunto de test (uno positivo y otro negativo), junto con sus respectivos vectores de características y distancias euclídeas calculadas (Figura 6.14). Se puede observar cómo para el umbral escogido (0.0102) ambos pares serían correctamente clasificados.

```
Distancia euclidia predicha: 0.0008067121962085366
Etiqueta Real: Par positivo
Vector de caracteristicas 1: tensor([[-0.0064,  0.0084]], device='cuda:0', grad_fn=<AddmmBackward0>)
Vector de caracteristicas 2: tensor([[-0.0064,  0.0076]], device='cuda:0', grad_fn=<AddmmBackward0>)
```

```
Distancia euclidia predicha: 0.016008571479469538
Etiqueta Real: Par negativo
Vector de caracteristicas 1: tensor([[-0.0124,  0.0086]], device='cuda:0', grad_fn=<AddmmBackward0>)
Vector de caracteristicas 2: tensor([[-0.0079,  0.0046]], device='cuda:0', grad_fn=<AddmmBackward0>)
```

Figura 6.14: Predicciones en 2 pares seleccionados aleatoriamente del conjunto de test

Finalmente, a pesar de las mejoras observadas en los resultados, el modelo evolucionado no alcanza aún un nivel de desempeño adecuado para su aplicación en seguridad biométrica. En consecuencia, se descarta de manera definitiva la arquitectura previamente propuesta.

Capítulo 7. Conclusiones

Se ha implementado con éxito un sistema de adquisición y procesado que posibilita la captura síncrona y la fusión en una única imagen de datos acústicos y LiDAR. Este sistema efectúa además una segmentación inteligente sobre las imágenes LiDAR empleadas con fines biométricos.

Este software habilita la realización de nuevas capturas acústicas y LiDAR, lo que facilita la generación de datos adicionales para continuar la presente investigación. Además, la existencia de este sistema de adquisición, no solo resulta relevante en el contexto de la biometría acústica, sino que también permite estudiar de manera sencilla la respuesta acústica de objetos diversos, filtrando espacialmente los lóbulos secundarios y las reflexiones producidas por otros elementos.

Se ha construido una base de datos multifrecuencial que comprende un total de 63000 imágenes distribuidas en tres frecuencias (21000 imágenes por frecuencia), y recolectadas sobre un conjunto de 10 individuos. Esta base de datos sirve como cimiento fundamental para que investigaciones subsiguientes puedan centrarse directamente en el análisis de estas imágenes.

Por otra parte, en relación con el estudio de las imágenes acústicas, los resultados obtenidos a partir del análisis de los máximos absolutos de energía en cada imagen, señalan que cada sujeto exhibe un histograma de amplitudes máximas singular. Este hallazgo presenta gran importancia ya que respalda la relevancia de la información de energía absoluta en la diferenciación entre sujetos.

Adicionalmente, se han identificado como principales áreas de reflexión en los individuos: el torso la cabeza y los hombros. Paralelamente, se han establecido relaciones que indican la predisposición de ciertos sujetos a reflejar energía en ciertas zonas del cuerpo. También es esencial destacar que estos elementos de máxima reflexión no se corresponden necesariamente con puntos específicos, sino que abarcan diferentes áreas del cuerpo, lo que sugiere la complejidad inherente a la reflexión acústica en el contexto de la identificación biométrica.

Asimismo, en lo que respecta a la información multifrecuencial, se ha observado que tanto la posición espacial como la distribución de energía es similar en las tres frecuencias empleadas, lo que indica cierta uniformidad en las reflexiones respecto a la frecuencia utilizada. En resumen, estos hallazgos contribuyen a comprender mejor las características acústicas y su potencial aplicabilidad en la identificación biométrica.

El entorno de ejecución proporcionado por "Google Colab" es una solución ágil para entrenar modelos de aprendizaje automático. No obstante, conlleva limitaciones en términos de recursos, lo que puede resultar una restricción relevante en función de la naturaleza del problema a resolver.

Finalmente, como se ha demostrado, la arquitectura "SigNet" no es lo suficientemente compleja como para aprender la profundidad de los datos en el marco de un problema de verificación.

7.1. Líneas Futuras

La escasa investigación existente en la literatura para la identificación biométrica mediante imágenes acústicas, unido a la amplia gama de decisiones tomadas durante el curso de esta investigación revela la presencia de numerosas oportunidades de desarrollo e investigación en esta línea.

La existencia del sistema de adquisición diseñado representa una ventaja considerable al simplificar la recolección de datos adicionales que permitan ampliar la base de datos. Para ello, sería de interés considerar la posibilidad de adquirir imágenes de sujetos adicionales o de los mismos en diferentes días y con diversas vestimentas. Este enfoque permitiría no sólo expandir el volumen de datos disponible, sino también ampliar el análisis para examinar la hipótesis relacionada con la variabilidad en las capturas asociada a la vestimenta y los movimientos y validar así la conveniencia del uso de imágenes acústicas para identificación de individuos.

A pesar de la inversión inicial, una alternativa más adecuada para entrenar los algoritmos sería la implementación de un servidor privado preparado para ejecutar código Python. Esta opción ofrece una mayor disponibilidad de los recursos en comparación con los servicios en la nube, como "Google Colab". Además, proporciona la capacidad de configurar el servidor y la asignación de recursos de manera flexible adaptándose a las necesidades específicas del proyecto.

Finalmente, muchas investigaciones inician su trayecto resolviendo problemas de clasificación o verificación basándose en datos existentes. En este contexto, la existencia de la base de datos construida en este proyecto proporciona un punto de partida sobre el cuál futuros trabajos pueden recurrir para fundamentar sus investigaciones.

De esta manera, futuros proyectos pueden abordar la tarea desde una perspectiva centrada en la búsqueda y comparación de algoritmos de inteligencia artificial de alta complejidad para la problemática en cuestión. Entre las posibles soluciones se pueden considerar enfoques como la utilización de arquitecturas más sofisticadas, la aplicación de *transfer learning* mediante el uso de redes preentrenadas, o incluso la exploración de otras alternativas realizando distinto tipo de proyecciones o la utilización directa de datos tridimensionales mediante redes convolucionales en tres dimensiones (CNNs 3D).

De este modo, este trabajo ha sentado las bases para el establecimiento de una nueva dirección de investigación en el ámbito de la identificación biométrica a través de imágenes acústicas de individuos. Las conclusiones y lineas futuras expuestas ofrecen una sólida estructura de trabajo para futuros proyectos que busquen explorar el potencial de esta tecnología que aún cuenta con limitada investigación al respecto.

Apéndice A. Bibliografía

[1] Ming Gao, Xihong Hu, Bo Cao, and Dianxin Li. Fingerprint sensors in mobile devices. In *2014 9th IEEE Conference on Industrial Electronics and Applications*, pages 1437–1440, 2014.

[2] M. I. Jiménez y J. J. Villacorta A. Izquierdo Fuente, L. del Val. Performance evaluation of a biometric system based on acoustic images. *Sensors*, 11(2):9499–9519, 2011.

[3] Andrés Martín Yeves. *Parametrización y clasificación de imágenes acústicas para biometría*. Trabajo fin de grado, Universidad de Valladolid, 2022.

[4] A.K. Jain, A. Ross, and S. Prabhakar. An introduction to biometric recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):4–20, 2004.

[5] Cafer Budak, Abidin Çalışkan, and Emrullah Acar. Comparison of multiple biometric identification with a single biometric identification system. 05 2018.

[6] A.K. Jain, A. Ross, and S. Prabhakar. An introduction to biometric recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):4–20, 2004.

[7] James Crowley. *Convolutional Neural Networks*, pages 67–80. Springer-Verlag, 04 2023.

[8] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*, 33(12):6999–7019, 2022.

[9] Bhavesh Pandya, Georgina Cosma, Ali A. Alani, Aboozar Taherkhani, Vinayak Bharadi, and T.M McGinnity. Fingerprint classification using a deep convolutional neural network. In *2018 4th International Conference on Information Management (ICIM)*, pages 86–91, 2018.

[10] Musab Coşkun, Ayşegül Uçar, Özal Yildirim, and Yakup Demir. Face recognition based on convolutional neural network. In *2017 International Conference on Modern Electrical and Energy Systems (MEES)*, pages 376–379, 2017.

[11] Rig Das, Emanuela Piciucco, Emanuele Maiorana, and Patrizio Campisi. Convolutional neural network for finger-vein-based biometric identification. *IEEE Transactions on Information Forensics and Security*, 14(2):360–373, 2019.

[12] Rig Das, Emanuele Maiorana, and Patrizio Campisi. Visually evoked potential for eeg biometrics using convolutional neural network. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 951–955, 2017.

[13] M. Viberg M. Moebus, A. M. Zoubir. Parametrization of acoustic images for the detection of human presence by mobile platforms. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, page 3538–3541, 2010.

[14] A. M. Zoubir M. Moebus. Three-dimensional ultrasound imaging in air using a 2d array on a fixed platform. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2, 2007.

[15] M. Moebus. *Object Detection and Classification for Mobile Platforms Using 3D Acoustic Imaging*. Tesis, Technischen Universität Darmstadt, 2011.

[16] Ahmet Çalışkan, Cüneyt Budak, and Eylem Acar. Comparison of multiple biometric identification with a single biometric identification system, 2018. Preprint.

[17] Levente Tamas, M. Popa, Gh Lazea, I. Szoke, and Andras Majdik. Lidar and vision based people detection and tracking. *Journal of Control Engineering and Applied Informatics*, 12:30–35, 06 2010.

[18] Abraham Gamarra and José Pinto. Reconocimiento de personas en ambiente con emisiones de humo usando sensor laser y redes neuronales convolucionales desde nube de puntos 3d. parte 1. *BISTUA REVISTA DE LA FACULTAD DE CIENCIAS BASICAS*, 17:179, 12 2018.

[19] Jun Ogata M. Szarvas, U. Sakai. Real-time pedestrian detection using lidar and convolutional neural networks. *2006 IEEE Intelligent Vehicles Symposium*, pages 213–218, 2006.

[20] J. J. Villacorta y M. Raboso L. del Val, A. Izquierdo Fuente. Acoustic biometric system based on preprocessing techniques and linear support vector machines. *Sensors 2015*, 15(6):14241–14260, 2015.

[21] Vladimir Manasson, Robert Mino, and Lev Sadovnik. Spinning grating antenna for mmw imaging. *Proceedings of SPIE - The International Society for Optical Engineering*, 06 1997.

[22] Thomas Nyland, John Mattoon, Eric Herrgesell, and Erik Wisner. *Physical Principles, Instrumentation, and Safety of Diagnostic Ultrasound*, pages 1–CP1. Saunders, 12 2002.

[23] M. I. Skolnik. *Introduction to Radar Systems*. McGraw-Hill, 1981.

[24] Seemin Rubab and Ajaz Mir. Biometrics verification: a literature survey. 5:67, 12 2011.

[25] J. Andress. *The basics of information security: understanding the fundamentals of InfoSec in theory and practice*. Syngress, 2014.

[26] A. Carmona Benjumea. *Aspectos antropométricos de la población laboral española aplicados al diseño industrial*. Instituto Nacional de Seguridad e Higiene en el Trabajo. Ministerio de Trabajo e inmigración, 2003.

[27] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *ArXiv e-prints*, 11 2015.

[28] Carmelo Bonilla Carrión. *Redes Convolucionales*. Trabajo fin de grado, Universidad de Sevilla, 2020.

[29] Oliver Layton. Cs343: Neural networks – convolution and max pooling. Diapositivas, Otoño 2019. [Consultado el 3 de Septiembre de 2023].

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[31] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.

[32] Alfonso Hernández Silva. *Diseño de redes neuronales antagónicas para estegoanálisis*. Trabajo fin de master, Universidad Politécnica de Madrid, 2020.

[33] Ph.D. Natalie Parde. Backpropagation and convolutional neural networks. Diapositivas, Primavera 2020. [Consultado el 3 de Septiembre de 2023].

[34] Kai Ming Ting. *Confusion Matrix*, pages 209–209. Springer US, Boston, MA, 2010.

[35] Mohammad Hossin and Sulaiman M.N. A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining and Knowledge Management Process*, 5:01–11, 03 2015.

[36] Kai Ming Ting. *Precision and Recall*, pages 781–781. Springer US, Boston, MA, 2010.

[37] Kai Ming Ting. *Sensitivity and Specificity*, pages 901–902. Springer US, Boston, MA, 2010.

[38] Gireen Naidu, Tranos Zuva, and Elias Sibanda. *A Review of Evaluation Metrics in Machine Learning Algorithms*, pages 15–25. Springer, 07 2023.

[39] scikit-learn developers. scikit-learn, métricas y puntuación: cuantificar la calidad de las predicciones, 2020.

[40] Peter Flach and Meelis Kull. Precision-recall-gain curves: Pr analysis done right. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[41] National Instruments. Jeff kodosky - national instruments, último acceso Sept,2023.

[42] Jeffrey Kodosky. Labview. *Proc. ACM Program. Lang.*, 4(HOPL), jun 2020.

[43] National Instruments. Labview tutorial, último acceso Sept,2023.

[44] John Hunt. *A Beginners Guide to Python 3 Programming*. Springer, 08 2019.

[45] The Linux Foundation. Pytorch, último acceso Sept,2023.

[46] Scikit-learn, último acceso Sept,2023.

[47] Tensor flow, último acceso Sept,2023.

[48] Poornima Naik, Girish Naik, and Mr M.B.Patil. *Conceptualizing Python in Google COLAB*. Shashwat Publication, 01 2022.

[49] Intel. *Intel RealSenseTM LiDAR Camera L515 - Datasheet*, revisión 003 edition, 2021. Hoja de especificaciones correspondiente al uso del sensor LiDAR L515.

[50] V. A. Sankar Ponnapalli and P. V. Y. Jayasree. *Fractal Array Antennas and Applications*, chapter 2. IntechOpen, Rijeka, 2018.

[51] Lara Del Val Puente y Luis Suárez Alberto Izquierdo, Juan José Villacorta. Design and evaluation of a scalable and reconfigurable multi-platform system for acoustic imaging. *Sensors*, 16(10):1688, 2016.

[52] Juan José Villacorta y Luis Suarez Lara Del Val, Alberto Izquierdo-Fuente. Using a planar array of mems microphones to obtain acoustic images of a fan matrix. *Journal of Sensors*, 2017:1–10, 10 2017.

[53] STMicroelectronics. *MP34DT01 - Datasheet*, revisión 9 edition, 2012. Hoja de especificaciones de los sensores MEMS MP34DT01.

[54] NI. *sbRIO-9607 Specifications*, 2023. Hoja de especificaciones de sbRIO-9607.

[55] Intel. *Intel RealSense D400 Series/SR300 Viewer - User Guide*, revisión 002 edition, 2018. Manual de usuario de la aplicacion RealSense Viewer.

[56] Intel Corporation. Response to issue 8058, 2020.

[57] Guo Xutao, Qingxiang Zhang, and Wei Wang. *Frequency Diverse Array Radar Beamforming Algorithm Based on FFT*, pages 1125–1132. Springer, 01 2019.

[58] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, and A. Frenkel. On the segmentation of 3d lidar point clouds. In *2011 IEEE International Conference on Robotics and Automation*, pages 2798–2805, 2011.

[59] Ali Abdelkader and Mohamed Moustafa. *Camera and LiDAR Fusion for Point Cloud Semantic Segmentation*, pages 499–508. Springer, 01 2023.

[60] Charles M Anderson, David Saloner, Jay S Tsuruda, Lorraine G Shapeero, and Ralph E Lee. Artifacts in maximum-intensity-projection display of mr angiograms. *AJR. American journal of roentgenology*, 154(3):623–629, 1990.

[61] M Prokop, H O Shin, A Schanz, and C M Schaefer-Prokop. Use of maximum intensity projections in ct angiography: a basic review. *RadioGraphics*, 17(2):433–451, 1997. PMID: 9084083.

[62] David Freedman and Persi Diaconis. On the histogram as a density estimator: l^2 theory. *Z. Wahrscheinlichkeitstheorie verw Gebiete*, 57(4):453–476, 1981.

[63] Eman Ahmed, Alexandre Saint, Abd El Rahman Shabayek, Kseniya Cherenkova, Rig Das, Gleb Gusev, Djamila Aouada, and Bjorn Ottersten. A survey on deep learning advances on different 3d data representations, 2019.

[64] Zhangjie Cao, Qixing Huang, and Karthik Ramani. 3d object classification via spherical projections, 2017.

[65] Vishakh Hegde and Reza Zadeh. Fusionnet: 3d object classification using multiple data representations. *CoRR*, abs/1607.05695, 2016.

[66] J. Ortega-Garcia, Fernando Alonso-Fernandez, and R. Coomonte-Belmonte. *Seguridad Biométrica*. Fundación Rogelio Segovia para el Desarrollo de las Telecomunicaciones, 05 2008.

[67] Jane Bromley, Isabelle Guyon, Yann LeCun, et al. Signature verification using a "siamese" time delay neural network. *Adv Neural Inf Process Syst 6*, 6:737–744, 1994.

[68] Davide Chicco. *Siamese Neural Networks: An Overview*, pages 73–94. Springer US, New York, NY, 2021.

[69] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, 07 2005.

[70] Sounak Dey, Anjan Dutta, Juan Ignacio Toledo, Suman K. Ghosh, Josep Lladós, and Umapada Pal. Signet: Convolutional siamese network for writer independent offline signature verification. *CoRR*, abs/1707.02131, 2017.

[71] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[72] Nicolás Serrano and Alejandro Bellogín. Siamese neural networks in recommendation. *Neural Computing and Applications*, 35(19):13941–13953, July 01 2023.