Universidad de Alcalá
Escuela Politécnica Superior

Máster Universitario en Ingeniería de Telecomunicación

Trabajo Fin de Máster

Diseño, implementación y evaluación de un sistema de detección y seguimiento de la pose tridimensional de personas basado en "Deep Learning"

Autor: Irene Guardiola Luna

Tutor: Javier Macías Guarasa

Cotutora: Leticia Monasterio Expósito

2022

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Máster Universitario en Ingeniería de Telecomunicación

Trabajo Fin de Máster

Diseño, implementación y evaluación de un sistema de detección y seguimiento de la pose tridimensional de personas basado en "Deep Learning"

Autora: Irene Guardiola Luna

Tutor: Javier Macías Guarasa

Cotutora: Leticia Monasterio Expósito

Tribunal:

Presidente: Álvaro Hernández Alonso

Vocal 1º: Ignacio Parra Alonso

Vocal 2º: Javier Macías Guarasa

Fecha de depósito: 22 de septiembre de 2022

"Dime y lo olvido, enséñame y lo recuerdo, involúcrame y lo aprendo."
Benjamin Franklin

Agradecimientos

Primero que todo, quiero agradecer a mis padres, sin los que no habría llegado hasta aquí, y a mi familia por todo el apoyo recibido. Por escucharme y estar siempre ahí, tanto en los buenos como en los malos momentos.

A continuación, quiero dar las gracias a todas las amistades conseguidas en estos dos últimos años. Vosotros habéis conseguido que este corto pero intenso proceso de aprendizaje fuese más llevadero. También agradecer a los antiguos y nuevos amigos por todo el apoyo, las conversaciones y tiempo juntos que pasamos.

Finalmente, gracias a mi tutor de TFM Javier por dejarme aprender involucrándome en el proyecto. Gracias a ello me he introducido en una rama de conocimiento que desconocía y que me ha despertado un gran interés. Además, agradecer tanto a Javier como a mi cotutora Leticia, por guiarme durante todo el proceso y ayudarme a mejorar.

Ah! se me olvidaba alguien más. Gracias a mí misma. Gracias por tomar la decisión de comenzar el máster y por la dedicación y esfuerzo empleado. No ha sido en vano.

Resumen

En el presente trabajo se va a abordar el tema de la detección de las articulaciones y la pose tridimensional del cuerpo humano en el contexto de la valoración funcional de personas. El objetivo final sería evaluar el grado de dependencia, discapacidad y/o limitaciones que las personas puedan llegar a tener, en particular las de avanzada edad, al realizar actividades de la vida diaria, con especial énfasis en las actividades básicas, como pueden ser: comer, lavarse los dientes, sentarse, limpiar, etc. Con esto se pretende ayudar a los terapeutas ocupacionales a detectar limitaciones de forma temprana y obtener valoraciones objetivas a través del sistema automático, eliminando la subjetividad del personal evaluador y las interferencias que la presencia de este pueda ejercer en las personas que están siendo evaluadas. En el trabajo se estudiarán y aportarán algoritmos que proporcionen parámetros de destreza y funcionalidad. Para ello se han analizado las redes neuronales y las posibles arquitecturas que se podrían aplicar para resolver el problema mencionado. A tal efecto, se ha indagado en las redes que sean capaces de estimar la posición tridimensional de las articulaciones del cuerpo humano a partir de imágenes de profundidad y en RGB, con el fin de evaluar funcionalmente a las personas y obtener una valoración clínica valida.

Palabras clave: Estimación de pose 3D, evaluación funcional, aprendizaje profundo, imágenes de profundidad.

Abstract

This project will address the issue of joint detection and the three-dimensional pose of the human body in the context of the functional assessment of people. The final objective would be to evaluate the degree of dependence, disability and/or limitations that people may have, particularly the elderly, when performing activities of daily living, with special emphasis on basic activities, such as: eating, brushing teeth, sitting, cleaning, etc. With this we aim to help occupational therapists to detect limitations early and obtain objective evaluations through the automatic system, eliminating the subjectivity of the evaluating personnel and the interferences that the presence of the latter may exert on the people being evaluated. This proyect will study and provide algorithms that offer dexterity and functionality parameters. For this purpose, neural networks and the possible architectures that could be applied to solve the aforementioned problem has been analyzed. To this end, we have investigated networks capable of estimating the three-dimensional position of the joints of the human body from depth and RGB images, in order to functionally evaluate people and obtain a valid clinical assessment.

Keywords: 3D pose estimation, functional evaluation, deep learning, depth images.

Índice general

Índice de figuras

Índice de tablas

Lista de acrónimos

A2J	Anchor-to-Joint.
Adam	Adaptive Moment Estimation.
ADLs	Activities of Daily Living.
AMPS	Assessment of Motor and Process Skills.
BS	Base de Sustentación.
BSP	Body Segment Parameters.
CG	Centro de Gravedad.
CMC	centro de masas corporal.
CNN	Redes Neuronales Convolucionales.
CoM	Centro de masas.
FCN	Fully Convolutional Network.
GD	Descenso de Gradiente.
GPU	Unidad de Procesamiento Gráfico.
H	Momento Angular.
HDF	Hierarchical Data Format.
I	Momento de Inercia.
IADLs	Instrumental Activities of Daily Living.
ICCV	International Conference on Computer Vision.
ILSVRC	ImageNet Large Scale Visual Recognition Challenge.
IoU	Intersection over Union.
mAP	Mean Average Precision.
MSE	Error Cuadrático Medio.
NAG	Nesterov Accelerated Gradient.
NMS	Non-Máximum-Supression.
SGD	Descenso de Gradiente Estocástico.
SGN	Semantics-Guided Neural Networks.
SVM	Máquinas de Vector Soporte.

Capítulo 1

Introducción

1.1 Contexto

Son muchas las aplicaciones que se pueden desarrollar a partir del seguimiento de las personas y sus movimientos, pero este trabajo se va a centrar en ofrecer un primer enfoque y en estudiar las bases de la detección y seguimiento de la pose tridimensional de personas con el objetivo final de evaluar su grado de dependencia, discapacidad y/o limitación funcional a la hora de realizar actividades de la vida diaria (Activities of Daily Living (ADLs)). El término ADLs fue propuesto por Katz, al igual que el índice que recibe su nombre [1], con la motivación de servir como elemento de referencia para evaluar el grado de dependencia de las personas al realizar actividades cotidianas [2]. Hay otros indicadores para medir las ADLs, como son: el índice de Barthel [3] y la escala de Rosow y Breslau [4]. Otra escala que puede ser de utilidad, y que utilizan los terapeutas ocupacionales, es la Evaluación de Habilidades Motoras y de Procesamiento, Assessment of Motor and Process Skills (AMPS) [5], la cual se basa en la observación, reuniendo información relacionada con las habilidades motoras y de procesamiento que muestra una persona al ejecutar alguna actividad de la vida diaria. La calidad de la ejecución de las ADLs se valora mediante la calificación del esfuerzo, eficiencia, seguridad e independencia a la hora de realizar una tarea [6]. Además de estudiar las ADLs, también se suelen analizar las actividades instrumentales de la vida diaria, Instrumental Activities of Daily Living (IADLs), las cuales hacen referencia a la interacción con el medio. Para ello se suele utilizar la escala de Lawton y Brody [4].

Este trabajo se encuadra en el proyecto EYEFUL-UAH [7], realizado por el grupo de investigación GEINTRA, del departamento de Electrónica de la Universidad de Alcalá, en colaboración con un equipo de investigación en terapia ocupacional de la Universidad Rey Juan Carlos.

La temática del proyecto aún no se ha llegado a abordar en profundidad en la literatura, pero es de gran interés para poder detectar limitaciones en el movimiento de las personas de forma temprana, pudiendo deberse a una patología concreta o a los cambios físicos y cognitivos provocados por el envejecimiento.

Con un sistema automático de este estilo, se pretende obtener valoraciones objetivas clínicamente validadas, eliminando la subjetividad del personal evaluador y las interferencias que la presencia de este ejerce sobre las personas que están desempeñando las pruebas pertinentes.

El objetivo final que se plantea en el proyecto en el que se enmarca este Trabajo Fin de Máster es abordar la cuantificación de dicha capacidad funcional de forma automática, a partir de la información captada por distintos sensores, y utilizando técnicas de aprendizaje profundo. Particularmente, el presente trabajo tendrá que brindar información sobre la evaluación de la funcionalidad de las personas y aportar unos primeros algoritmos que ofrezcan información de ciertos parámetros de destreza a partir de la

localización de las articulaciones del cuerpo humano captadas por cámaras de profundidad. Además se va a escoger una red neuronal válida para este tipo de aplicaciones.

1.2 Objetivos

En este trabajo se plantea el diseño, la implementación y evaluación de estrategias y algoritmos para la detección y seguimiento en 3D de las articulaciones del cuerpo de una persona usando técnicas de Deep Learning aplicadas a grabaciones de vídeo o de profundidad. Para el correcto desarrollo del trabajo, se abordará la revisión bibliográfica del estado del arte en dicha temática (incluyendo la recopilación de bases de datos y algorítmica de partida), la propuesta de diseño de soluciones algorítmicas, y la generación de una implementación final que será evaluada de forma rigurosa sobre las bases de datos disponibles.

La aplicación final partirá de los resultados obtenidos en este trabajo, ya que estará orientada a la medida de parámetros relacionados con la destreza de personas en la realización de actividades, en un contexto de valoración funcional de sus capacidades motoras y cognitivas. Será a partir de estos parámetros medidos que se establecerá el grado de discapacidad que presenta la persona analizada.

El trabajo implicará la utilización de técnicas de procesamiento de vídeo combinadas con las de inteligencia artificial basadas en redes neuronales profundas. El entorno de desarrollo se apoyará en una plataforma GNU/Linux, sobre el lenguaje de programación Python, usando librerías avanzadas de procesamiento.

En la figura 1.1 se muestra a grandes rasgos un diagrama de bloques de las etapas del sistema a diseñar, las cuales incluyen:

- La detección y seguimiento de las articulaciones y las manos de las personas a partir de vídeos de personas moviéndose y realizando actividades cotidianas.
- Extracción de características a partir de los *datasets* y la etapa de seguimiento.
- En la siguiente etapa se tomaría una decisión en función de los datos obtenidos en las fases anteriores.
- Finalmente se obtendría la información del grado de dependencia de la persona estudiada.

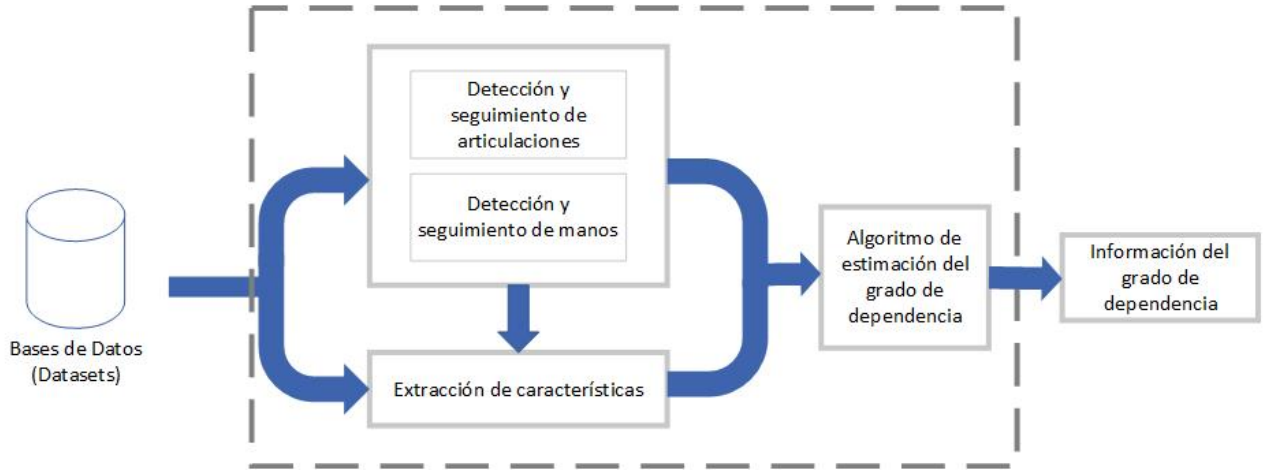


Figura 1.1: Diagrama de bloques del sistema.

En base a lo descrito, los objetivos específicos de este proyecto son:

- Realizar un estudio de alternativas del estado del arte de sistemas de detección y seguimiento de articulaciones [8,9] (3D, RGB monocular y profundidad).
- Seleccionar los sistemas apropiados y adaptarlos, en caso de ser necesario, a las restricciones del problema.
- Diseñar los algoritmos de estimación de parámetros de destreza y funcionalidad.
- Seleccionar bases de datos adecuadas para entrenar al sistema y verificar el correcto funcionamiento de éste [10–13].
- Documentar rigurosamente el trabajo realizado y las decisiones tomadas.

1.3 Organización de la memoria

En base a los objetivos propuestos descritos en la sección 1.2, se ha dividido este documento en diferentes capítulos, afrontando en cada uno unas tareas específicas y comentando los detalles más relevantes. En definitiva, el contenido del proyecto se ha organizado de la siguiente manera:

- **Capítulo 1**. En él se hace una introducción al tema a investigar en el trabajo junto con la motivación de dicha investigación y los objetivos a desarrollar. Adicionalmente se incluye una sinopsis de los contenidos que se van a tratar en cada capítulo.
- **Capítulo 2**. En este capítulo se introduce el estudio teórico relacionado con los parámetros útiles para poder valorar la capacidad funcional de una persona. Además de enumerar los distintos sistemas de detección y seguimiento de articulaciones existentes con código abierto y las bases de datos que serían convenientes para entrenar el modelo final. También se ha indagado dentro del aprendizaje automático y especialmente en las redes neuronales convolucionales.
- **Capítulo 3**. En el que se detalla la red utilizada para solucionar el problema planteado.
- **Capítulo 4**. En este capítulo se describen todos los experimentos realizados, tanto con la red como con los algoritmos de estimación de parámetros de destreza y funcionalidad desarrollados. A su vez, después de la descripción de cada experimento, se muestran los resultados obtenidos.
- **Capítulo 5**. En el que finalmente se detallan las principales conclusiones del trabajo y se describen algunas de las líneas futuras que se podrían seguir para continuar o mejorar el trabajo.

Capítulo 2

Estado del Arte

2.1 Introducción

El capítulo se ha dividido en dos áreas temáticas: la primera parte corresponde a la valoración de la capacidad funcional y al estudio de valores cinemáticos del ser humano desde el punto de vista de la biomecánica; y en la segunda parte se desarrollan los temas más técnicos, como son los sistemas de detección y seguimiento de articulaciones, bases de datos y la inteligencia artificial.

2.2 Estudio teórico

2.2.1 Valoración de la capacidad funcional

La valoración de la capacidad funcional se realiza para determinar las limitaciones que presenta un sujeto a la hora de realizar ciertas actividades como consecuencia de su condición de salud actual. Es especialmente importante en la valoración geriátrica integral [14], en la que se estudian las capacidades motoras, la capacidad de autocuidado, el ejercicio físico y la independencia en el medio ambiente, etc. Al realizar esta valoración se consigue detectar problemas y discapacidades, y aplicar medidas preventivas y terapéuticas.

La valoración de la capacidad funcional parte de la medida de la discapacidad que pueda tener una persona a la hora de realizar las ADLs y de la evaluación de las limitaciones funcionales. Las ADLs se dividen en actividades básicas, de interacción con instrumentos y avanzadas. Dentro de las primeras entran las funciones elementales para el autocuidado y son las últimas que se pierden (el baño, el aseo, el uso del retrete, la movilidad, la capacidad de comer y vestirse, y las continencias). Las IADLs son algo más complicadas que las anteriores porque dependen de la función cognitiva, pero son indispensables para la independencia en el medio ambiente, y son las actividades que incluyen algún tipo de interacción con un objeto, como son las actividades domésticas (limpiar, lavar, cocinar y comprar) y no domésticas (manejo del dinero, control de la medicación, uso del teléfono y de un medio de transporte). Por último, las actividades avanzadas, las cuales no son indispensables para vivir de forma autónoma y consisten en actividades sociales, ocio, etc.

2.2.2 Valores cinemáticos del ser humano

Los estudios que aportan valores cinemáticos del ser humano son analizados mayoritariamente para aplicarlos en las áreas de la robótica, la animación por ordenador y la biomecánica. En el área de la robótica se pretende sustituir o ayudar al humano en ciertas tareas, y en el caso de los robots humanoides, intentan que sus movimientos sean lo más parecido a los de un humano. En la animación pretenden simular en personajes humanos o similares el movimiento de los mismos. Y en la biomecánica se intenta detectar y solucionar las alteraciones anatómicas y de movimiento que surgen por el efecto que tienen diversas condiciones de salud en el cuerpo humano, el propio envejecimiento del cuerpo y por las circunstancias a las que se somete al cuerpo durante la realización de las actividades de la vida cotidiana [15].

Desde el punto de vista de la Biomecánica Deportiva, el cuerpo humano se puede aproximar a un sistema formado por un conjunto de segmentos sobre los que actúan tanto fuerzas externas como internas [16]. Este sistema se puede diseñar a partir de dos enfoques:

1. Considerar que los segmentos son rígidos y articulados, suponiendo que las articulaciones son puntuales. Esta suposición es microscópicamente errónea, ya que el movimiento de rotación que realizan las articulaciones implican la existencia de un centro de giro, el cual cambia continuamente de ubicación, pero este rango de desplazamiento está limitado por la cavidad articular, siendo este mínimo. Por ello, se puede estimar la posición del centro de giro en el centro de la articulación, planteamiento que se aplica en la mayoría de estudios.

2. Considerar las capacidades elásticas y deformadoras de los materiales que componen el cuerpo humano.

El cuerpo humano se puede segmentar de diversas maneras, aunque habitualmente se utilizan modelos con 14 segmentos (ver figura 2.1).

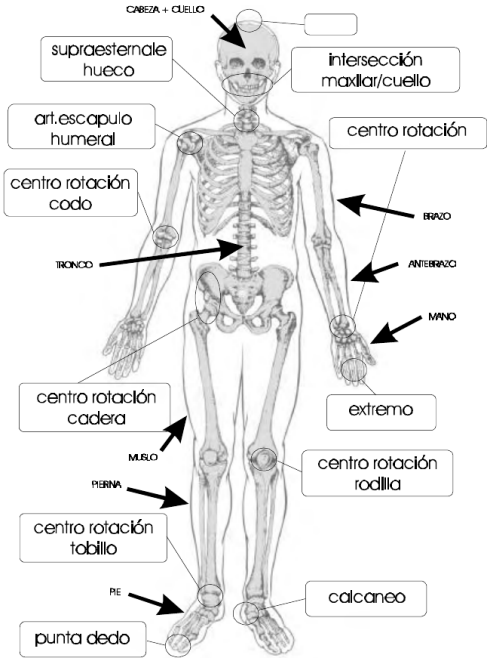


Figura 2.1: Puntos anatómicos y segmentos más habituales utilizados en los modelos biomecánicos [16].

Para analizar el movimiento de los humanos y obtener parámetros inerciales, se utilizan una serie de parámetros muy importantes: el centro de masa segmentario, el cual centra la masa de todo un segmento

en un punto; el CoM corporal; el Momento de Inercia (I); y el Momento Angular (H) segmentario y corporal. De estos parámetros, el más importante es el CoM.

Según Cappozzo y Berme [17], para representar las propiedades inerciales vinculadas a los segmentos corporales se utiliza la masa del segmento absoluta (en kg) y relativa (en %), el vector posición del CoM, los ejes referidos al momento de inercia y el I. Estos valores se pueden adquirir mediante un análisis estadístico o con estimaciones del individuo. Para determinarlos se pueden aplicar métodos directos o indirectos:

- **Métodos directos**. Se utiliza técnicas experimentales en sujetos vivos, obteniéndose valores particulares. Estos métodos se basan en técnicas de inmersión en agua, vibraciones mecánicas, utilización de escáner, etc.
- **Métodos indirectos**. Se utilizan algoritmos basados en la obtención de medidas antropométricas, es decir, parámetros de segmentos corporales, BSP. Se divide en dos subgrupos:
  - Métodos basados en aproximaciones geométricas. Se simula la morfología humana a partir de figuras geométricas descritas matemáticamente.
  - Métodos con base en las ecuaciones de regresión. Este es el método más manejado, en el cual los datos utilizados se obtienen de poblaciones. En los primeros estudios [18] se utilizaban poblaciones de cadáveres, pero éstas no eran grandes (13 cadáveres como máximo) debido a la dificultad que suponía tener que ir diseccionando en segmentos el cuerpo, pesarlos, y tomar medidas antropométricas (perímetros, pliegues, índices corporales, etc). Años más tarde también se empezaron a hacer estudios con personas vivas.

En este trabajo se va a descartar el uso de los valores de cadáveres ya que presenta varios inconvenientes: la muestra de sujetos es muy reducida, existe una perturbación de los tejidos del cuerpo tras haber fallecido y la ausencia de datos de cuerpos femeninos. Estas limitaciones podrían solventarse utilizando técnicas basadas en modelos matemáticos individualizados, pero esto supondría la necesidad de una gran cantidad de medidas. Por ello, De Leva [19] adaptó los parámetros inerciales de los estudios de Zatsiorsky [20] utilizando medidas antropométricas obtenidas de la base de datos del ejército de EE.UU de 1988. Estos parámetros son los más empleados en la actualidad y se han indicado en la tabla 2.1 los correspondientes al porcentaje de la distancia que hay desde el CoM propio de cada segmento hasta el punto de referencia, y el porcentaje de peso de cada segmento.

Mediante estos parámetros y las coordenadas del segmento, se puede calcular la ubicación del CoM [22] a partir de la siguiente ecuación matemática:

$$CGT = \sum Gpi - [Ki(Gpi - Gdi)] \cdot Pi \tag{2.1}$$

Donde:
CGT = centro de gravedad total.
Gpi y Gdi = coordenadas (x, y o z) proximal y distal respectivamente.
Ki = la distancia desde el punto proximal al CoM del segmento, en tanto porciento de la distancia total de cada segmento.
Pi = porcentaje de peso de cada segmento.

En este caso, el centro de gravedad total es igual al centro de masas del cuerpo, ya que al ser el campo gravitatorio uniforme, en la Tierra se puede considerar así, ambos parámetros se localizan en la misma posición a pesar de ser diferentes. Es por ello que se podrá hablar de ellos indistintamente. El CG de

Tabla 2.1: BSP. Las masas de los segmentos son relativas a la masa corporal total, y las posiciones del CoM de cada segmento son relativos a su longitud a puntos finales proximales [21].

Segmento	Masa (%)		CoM (%)	
	Hombre	Mujer	Hombre	Mujer
Cabeza	6,94	6,68	59,76	58,94
Tronco	43,46	42,57	43,10	37,82
Brazo	2,71	2,55	57,72	57,54
Antebrazo	1,62	1,38	45,74	45,59
Mano	0,61	0,56	79,00	74,74
Muslo	14,16	14,78	40,95	36,12
Pierna	4,33	4,81	44,59	44,16
Pie	1,37	1,29	44,15	40,14

un cuerpo es el punto respecto al cual la fuerza que la gravedad ejerce sobre los diferentes puntos que constituyen al cuerpo producen un momento resultante nulo y en el que el peso corporal se equilibra. El CoM, como ya se ha indicado, es el punto en el que se concentra la masa completa del cuerpo.

2.2.3 Postura

El control postural es uno de los factores determinantes de la postura. Constituye la habilidad de lograr mantener la estabilidad durante una postura estática o dinámica de todo el cuerpo o de algunos de los segmentos corporales, en respuesta a fuerzas de oposición que provocan un desequilibrio, como por ejemplo la gravedad o las irregularidades del terreno [23].

Durante el análisis de la postura, el evaluador debe ser capaz de reconocer y determinar si un segmento o articulación está desviado con respecto al alineamiento postural óptimo normal. Para ello también se ayudan de la base de sustentación, la cual se encarga de distribuir el peso en un área específica. El polígono de soporte está delimitado por el área comprendida entre los pies. Para asegurar la estabilidad del cuerpo, el CG o el CoM y la línea vertical que lo atraviesa debe caer dentro del polígono de soporte, por lo que una salida brusca del CoM acarrea una caída inmediata. El equilibrio se puede definir como [24]: "la habilidad de mantener la proyección del CMC dentro de los límites de la Base de Sustentación (BS). Necesariamente, a medida que ocurre la marcha, la posición de la BS cambia, así como la posición del CMC". Es por esto que el equilibrio se puede estimar cuantificando la proyección del CMC con respecto a la BS.

2.2.4 Marcha

La marcha es el paso bípedo que utilizan los humanos para desplazarse. Es un movimiento inestable y cada ser humano tiene su propio modelo de marcha característico. Durante el análisis de la marcha, el evaluador debe identificar las deficiencias y limitaciones presentes en la actividad y en caso de que el sujeto necesite utilizar algún auxiliar de la marcha (muletas, bastón, andador, etc), debe distinguir el grado de independencia, el tipo de auxiliar de la marcha que necesita, su aplicación y si le resulta útil [23].

2.2.4.1 Ciclo de la marcha

El ciclo de la marcha hace referencia al tiempo que transcurre entre el contacto inicial del pie con el suelo y el siguiente contacto inicial de ese mismo pie [23]. Este ciclo se divide en dos periodos:

- **Periodo de soporte**. Es el tiempo que transcurre mientras el pie esté tocando el suelo. Suele constituir aproximadamente el 62 % del ciclo total.
- **Periodo de balanceo**. Es el tiempo durante el cual el pie se mantiene en el aire para avanzar hacia adelante. Este constituye el 38 % restante del ciclo total.
- **Periodo de doble soporte**. Tiempo en el que los dos pies se encuentran apoyados en el suelo. Ocurre en dos ocasiones durante la marcha, al inicio de la fase de soporte y al terminar. Este periodo supone el 12.5 % del ciclo, por lo que los dos periodos constituyen el 25 % del periodo de soporte.

Del ciclo de marcha, además de los parámetros anteriores, se obtienen los siguientes:

- **Periodo de paso**. Es el intervalo temporal entre el primer contacto de un pie con el suelo y el primer contacto del pie contrario.
- **Frecuencia o cadencia**. Es el número de pasos que se da por minuto. Habitualmente es un valor de entre 90 y 140 pasos por minuto.
- **Velocidad**. Relación de la distancia recorrida por unidad de tiempo.
- **Aceleración**. En relación al cambio de la velocidad.
- **Longitud de paso largo o zancada**. Distancia en metros entre dos eventos iguales y sucesivos en una misma extremidad, por ejemplo, la distancia desde un primer contacto de un pie con el suelo hasta el siguiente contacto inicial del mismo pie.
- **Longitud de paso o paso corto**. Distancia en metros entre dos eventos iguales con una misma extremidad. Normalmente, se utiliza la distancia entre un contacto inicial de un pie hasta el contacto inicial del otro pie.
- **Ancho de paso**. Ancho en centímetros entre dos puntos iguales de los pies.
- **Ángulo de paso**. Es la orientación que adopta el pie durante el apoyo. Generalmente es de 5^o y 8^o con la línea de dirección de la marcha.

2.2.4.2 Desplazamientos del CG o CMC durante la marcha

Durante la marcha el CG sufre desplazamientos tanto horizontales como verticales (ver figura 2.2), proyectándose siempre sobre la base de soporte y describiendo una curva sinusoidal suave tanto en el eje vertical como en el horizontal [23]. Estos desplazamientos no exceden la excursión máxima normal, la cual es en promedio de 3 a 5 cm. Para acompañar estos movimientos, la pelvis rota 4^o hacia adelante y 4^o hacia atrás, y el tórax gira en sentido contrario para mantener el equilibrio. Además, la pelvis del lado sin apoyo durante el balanceo, desciende 5^o con respecto a la horizontal. Cuando el pie que avanza toca el suelo, la rodilla está prácticamente extendida completamente, seguidamente la rodilla se flexiona aproximadamente 15^o, luego se extiende y mantiene 5^o de flexión, para finalmente extenderse en el soporte terminal y volver a flexionarse 35^o en el prebalanceo.

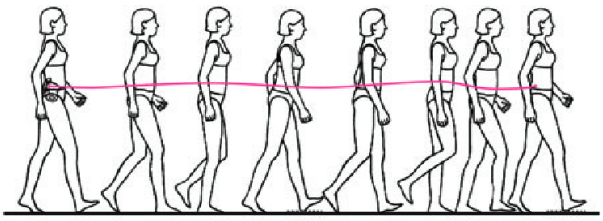


Figura 2.2: Recorrido del centro de gravedad en sentido vertical [23].

2.3 Sistemas de detección y seguimiento de articulaciones

En esta sección se van a exponer los distintos sistemas de código abierto de detección y seguimiento de articulaciones estudiados, con especial énfasis en los que se han llevado a la práctica. En un primer momento la búsqueda se centró en los sistemas de estimación de la pose 3D que podrían ser de utilidad, ya que al inicio se pensó en aplicar un primer "bloque" que fuera capaz de clasificar la actividad que la persona está realizando, para a partir de ahí determinar si las medidas de los parámetros de destreza se encuentran dentro de un rango establecido en función de la actividad realizada. Los sistemas encontrados son los siguientes:

- El estudio [25] realiza un sistema de reconocimiento de la acción humana basado en las coordenadas tridimensionales del esqueleto humano.
- En el *paper* [26] proponen un sistema de reconocimiento de ADLs a partir de las posiciones 2D del esqueleto humano. Como propiedad principal aseguran la robustez ante variaciones temporales que surgen por las distintas longitudes que pueda tener cada acción realizada.
- En el proyecto [27] plantean una red neuronal guiada por la semántica (Semantics-Guided Neural Networks (SGN)), la cual explota explícitamente la semántica y la dinámica para realizar un reconocimiento de la acción basado en el esqueleto. Es decir, explota las correlaciones espaciales y temporales a nivel de articulación y de *frame*.
- En [28] intentan aliviar los efectos de las variaciones de la vista al capturar los datos, que es uno de los problemas que existe en los sistemas de reconocimiento de la acción humana basado en el esqueleto.

Finalmente, se descartó el profundizar en la idea anterior para dar mayor preferencia a los sistemas que cumplen con los requisitos especificados, es decir, que sean capaces de detectar las articulaciones a partir de una imagen de profundidad y monocular.

Como ya se ha mencionado, la detección de la pose y su seguimiento se lleva utilizando ampliamente en distintas aplicaciones como pueden ser: la interacción persona-ordenador, el control de robots, la creación de animaciones en 3D, el entretenimiento doméstico, etc.

La propuesta que hacen en [29] es a partir de una imagen RGB extraer o detectar la localización en 2D de K articulaciones del cuerpo humano haciendo uso de una red neuronal. La ventaja que ofrecen respecto a otros estudios es que el mapa de calor de las articulaciones predichas es potencialmente más exacta y espacialmente más preciso.

En [30] han conseguido crear un sistema que se encarga de predecir las posiciones en 3D de las articulaciones dadas las ubicaciones de estas en 2D utilizando una red neuronal "simple", aplicando *batch normalization*, *dropout*, ReLUs y conexiones residuales, consiguiendo que sea más fácil de optimizar.

En el trabajo [31] proponen una red que mantiene las representaciones a alta resolución durante todo el proceso de aprendizaje de la estimación de la pose humana con la motivación de generar un mapa térmico de puntos clave estimados potencialmente más exacto.

El estudio [32] presenta dos soluciones distintas para estimar la pose de un humano en 3D a partir de una o más vistas combinadas, obteniendo información 3D de imágenes 2D. La primera solución se apoya en la triangulación algebraica junto con la adición de pesos de confianza estimados a partir de las imágenes de entrada. La segunda solución, se basa en un novedoso método de agregación volumétrica a partir de mapas intermedios de características en 2D, produciendo finalmente mapas de calor en 3D de las distintas articulaciones del cuerpo.

Finalmente, el estudio [33], el cual parte de [34], propone una estimación de la pose de la mano y del cuerpo humano a partir de una imagen de profundidad. El enfoque utiliza puntos de "anclaje", los cuales capturan la información del contexto espacial global-local al establecerse densamente en la imagen de profundidad como regresores locales para las articulaciones. Como predicen las posiciones de las articulaciones de forma conjunta, mejoran la capacidad de generalización de la red neuronal empleada. La red que utilizan como red troncal para impulsar la red A2J es una CNN 2D, en concreto la ResNet-50 [35], sin necesidad de utilizar capas convolucionales o deconvolucionales 3D que consumen mucho tiempo.

De entre todos los sistemas analizados, se ha decidido utilizar el denominado **A2J: Anchor-to-Joint Regression Network for 3D Articulated Pose Estimation from a Single Depth Image** [33], debido al doble uso que ofrece de detectar las articulaciones de tanto manos como del cuerpo completo, además de ser uno de los sistemas con mayor precisión y eficiencia, y porque también se puede aplicar en casos en los que la entrada sea una imagen RGB. Además, fueron subcampeones en el *HANDS2019 3D hand pose estimation Challenge*, en International Conference on Computer Vision (ICCV) de 2019.

2.4 Sistemas automáticos de estimación de destreza

Se ha realizado una amplia búsqueda de sistemas capaces de estimar la capacidad funcional de las personas, sin embargo, no se ha encontrado ninguno que estime exactamente ni en su totalidad la capacidad funcional. Dentro de los sistemas automáticos de estimación de destreza con código abierto se ha encontrado un estudio [36], en el que clasifica diferentes niveles de deterioro de las extremidades superiores (sanos, ligeramente deteriorados o moderadamente deteriorados), con una precisión media de 91,7 %.

Los siguientes estudios que se van a anunciar no son exactamente sistemas automáticos de estimación de destreza, pero si ofrecen estudios de evaluación de parámetros que podrían ser útiles para determinar el grado de destreza de una persona en el momento de hacer algún tipo de movimiento.

El estudio [37] evalúa los ángulos en las articulaciones del codo y del hombro en cuatro ejercicios típicos de rehabilitación de las extremidades superiores: flexión lateral del codo, flexión del codo, extensión del hombro y abducción del hombro.

El estudio [38] proporciona un conjunto de parámetros cinemáticos obtenidos con Kinect V2 al realizar los sujetos un conjunto de movimientos de la extremidad superior: la evaluación de Fugl-Meyer de la extremidad superior, el movimiento de alcanzar y mover la mano a la boca. En primer lugar, calcularon los parámetros cinemáticos de los sujetos sanos y de pacientes que han sufrido un accidente cerebrovascular, incluido el rango de movimiento de la articulación del hombro, la articulación del codo y el tronco, el desplazamiento relativo de la articulación del codo y la muñeca con respecto al marco de coordenadas local, la suavidad del movimiento y la coordinación entre articulaciones. Después analizaron la normalidad

y la varianza de estos parámetros y para cada paciente los cuantificaron mediante un coeficiente cualificado (QC). Los resultados mostraron que cuanto más grave es la enfermedad, menor es el QC.

El estudio [39] analizaron la cinemática de las articulaciones en las fases de alcance y retorno durante la acción de beber agua y encender una luz.

En [40] han investigado como disminuye el control postural con el envejecimiento de las personas, a partir del análisis de las coordenadas de 15 articulaciones proporcionadas al grabar la acción de mantenerse de pie con una cámara Kinect. Con ello pretenden identificar la idoneidad, la eficiencia y la estabilidad o la rapidez con la que un sujeto realiza o mantiene una posición de equilibrio. Algo parecido hacen en [41], pero en este caso determinan la estabilidad postural mientras la persona está sentada.

En el estudio [42] analizan la marcha de personas con la motivación de ofrecer unos índices posturales y de la marcha que se puedan medir para evaluar automáticamente el síndrome de fragilidad, determinado por la pérdida de capacidades físicas y psicológicas.

El trabajo [43] presenta una medida personalizada del equilibrio que considera las variaciones de la masa corporal junto con el seguimiento del movimiento mediante la cámara Kinectv2. Basándose en los datos obtenidos, y mediante el proceso de extracción de polígonos de apoyo (para definir la base de sustentación), han realizado un sistema en tiempo real para evaluar el equilibrio personalizado y la visualización del riesgo de caída sin necesidad de utilizar una plataforma de fuerza.

En el proyecto [44] han formulado una puntuación de estabilidad estática basada en la lógica difusa (*fuzzy*) para la población con ictus y la geriátrica, calculada a partir de un conjunto seleccionado de medidas derivadas de datos del esqueleto obtenido con Kinect durante el experimento de "mantenerse de pie con una pierna" (Single Limb Stance). El conjunto de parámetros analizados son: tiempo en el que se mantienen apoyado en una sola pierna, variación en el índice de vibración y desplazamiento del CoM.

2.5 Bases de Datos

En esta sección se va a aportar una tabla (2.2) con una serie de bases de datos que podrían ser de utilidad para probar el sistema. Estas bases de datos deben contener datos RGB y de profundidad, además de las posiciones de las articulaciones de las personas, y las actividades que desarrollen en ellos tienen que ser de la vida cotidiana, es por ello que en la tabla también se han añadido las actividades básicas generales que contienen cada base de datos.

- **NTU RGB+D y NTU RGB+D 120** [45, 46]. La primera contiene 60 clases diferentes y 56.880 muestras de vídeo. La segunda es una expansión de la primera, contiene 120 clases y 114.480 muestras de vídeo en total. Cuenta con 82 clases de ADLs, 12 condiciones médicas y 26 acciones e interacciones con otra persona. Para grabar los vídeos han utilizado 3 cámaras Kinect V2.

- **UTKinect-Action3D** [47]. Esta base de datos recoge los vídeos de 10 personas diferentes realizando 10 acciones una tras otra en dos ocasiones. Han grabado los vídeos utilizando una cámara Kinect V1.

- **Human3.6M** [48, 49]. Cuenta con 3.6 millones de poses 3D y sus correspondientes imágenes RGB. Cada conjunto de poses forma un escenario de entre los 17 posibles, y están ejecutados por 11 actores profesionales (6 hombres y 5 mujeres). Cada escenario lo han grabado con 4 cámaras digitales colocadas cada una en una esquina de la habitación; 1 sensor *Time-of-Flight*, los cuales utilizan la tecnología llamada Lidar para medir el fondo de varios puntos de una imagen a partir de la iluminación de la zona con luz infrarroja.

- **MSR DailyActivity 3D** [50, 51]. Esta base de datos cuenta con 16 actividades distintas de la vida cotidiana, realizadas por 10 sujetos dos veces cada uno. En total hay 320 archivos de datos de mapas de profundidad, posiciones del esqueleto y vídeos, los cuales han sido obtenidos a través de una cámara Kinect.

- **Northwestern-UCLA Multiview Action 3D Dataset** [52]. Contiene grabaciones simultáneas de tres cámaras Kinect, en las que 10 personas han realizado 10 actividades diarias distintas como: sentarse, levantarse, tirar la basura, lanzar, etc.

- **UTD Multimodal Human Action Dataset (UTD-MHAD)** [53]. En este caso han utilizado tanto una cámara Kinect como un sensor inercial colocado en la muñeca o muslo derecho del sujeto en función de la implicación de estos en la acción ha realizar, y han recogido 17 acciones diferentes ejecutadas 4 veces cada una por 8 personas distintas (4 hombres y 4 mujeres). Los datos obtenidos son de profundidad, vídeos RGB, posiciones de las articulaciones del esqueleto y secuencias de aceleración.

- **TST Fall detection dataset v2** [54]. La base de datos la forman las grabaciones de profundidad, las posiciones de las articulaciones del esqueleto y vídeo por una cámara Kinect v2, de 11 voluntarios que han realizado 4 ADLs y 4 caídas tres veces cada una. Además, también añade medidas de aceleración de la cadera y la muñeca derecha de cada persona tomadas por una Unidad de Medición Inercial (IMU).

- **PKU-MMD: A Large Scale Benchmark for Continuous Multi-Modal Human Action Understanding** [55, 56]. Este *dataset* se ha grabado con la cámara Kinect v2. Contiene 1076 secuencias de vídeo que incluyen 51 acciones diversas realizadas por 66 sujetos distintos desde tres puntos de vista diferentes. Las acciones se dividen en 41 actividades diarias y 10 actividades de interacción con otro humano. Ofrece mapas de profundidad con valores en milímetros, las posiciones de las articulaciones de cuerpo en 3D, las grabaciones en RGB y las secuencias de infrarrojo.

- **ETRI-Activity3D** [26]. Esta base de datos contiene actividades de vida diaria realizadas por 50 personas de edad avanzada, entre 64 y 88 años, y 50 jóvenes en su veintena. Hay un total de 112.620 secuencias formadas por 55 acciones tomadas a través de un sensor Kinect v2. El contenido se divide en vídeos RGB, mapas de profundidad, posiciones 3D de las articulaciones del cuerpo humano e imágenes segmentadas, las cuales están formadas por dos segmentos, uno es el que contiene a la persona y el otro contiene la habitación u entorno en el que esta se encuentra. Las grabaciones las han realizado en distintas habitaciones de una casa, en la cocina, sala de estar y dormitorio, además de estar tomadas desde 4 puntos de vista diferentes y a dos alturas del suelo distintas cada uno.

- **ITOP** [57,58]. Contiene 100.000 imágenes de profundidad grabadas con una Asus Xtion PRO desde un lateral y desde arriba, además de las posiciones 3D de 15 articulaciones del esqueleto del sujeto. Han participado 20 personas realizando 15 acciones secuencialmente cada una.

- **Jointly Learning Heterogeneous Features for RGB-D Activity Recognition (SYSU 3D)** [59]. Este *dataset* está enfocado en la interacción del humano con un objeto (Human-Object Interaction, HOI), como por ejemplo, la interacción con escobas, fregonas, tazas, etc. Contiene 480 vídeos de 12 actividades distintas realizadas por 40 personas.

Tabla 2.2: Bases de datos de ADLs.

Características Base de Datos				**Actividades que incluye**					
Nombre	**Resolución y FPS**	**nº *joints***	**Variabilidad**	**Organización de espacio y objetos**	**Movilidad**	**Comida**	**Aseo**	**Vestido/ Arreglo**	**Traslado cama/sillón**
RGB+D 120	RGB: 1920 × 1080 Depth, IR: 512 × 424 30 fps	25	Grabaciones desde tres puntos de vista	X	X	X	X	X	X
UTKinect-Action3D	RGB: 640x480 Depth: 320 × 240 30 fps	20	Grabaciones desde un punto de vista	X	X				X
Human3.6M	50 fps	24	Grabaciones desde 4 puntos de vista		X	X			X
MSR Daily Activity 3D	RGB: 640x480 Depth: 320 × 240 30 fps	20	Grabaciones desde un punto de vista		X	X			X
Multiview Action 3D	RGB: 640x480 Depth: 320 × 240 30 fps	20	Grabaciones desde tres puntos de vista	X	X				X
UTD-MHAD	RGB: 640x480 Depth: 320 × 240 30 fps	25	Grabaciones desde un punto de vista	X	X				X
TST Fall detection	RGB: 1920x1080 Depth: 512 × 424 30 fps	25	Grabaciones desde distintos puntos de vista		X				X
PKU-MMD	RGB: 1920x1080 Depth: 512 × 424 30 fps	25	Grabaciones desde distintos puntos de vista		X	X		X	X
ETRI-Activity3D	RGB: 1920x1080 Depth: 512 × 424 30 fps	25	Grabaciones desde distintos puntos de vista	X		X	X	X	X
ITOP	RGB: 640x480 Depth: 320 × 240 30 fps	15	Grabaciones desde dos puntos de vista		X	X			
SYSU 3D	RGB: 1920x1080 Depth: 320 × 240 30 fps	15	Grabaciones desde varios puntos de vista		X	X			

2.6 Aprendizaje automático (*Machine Learning*)

Ya que en el proyecto es necesario utilizar un modelo que aprenda a generar estimaciones (de posiciones de articulaciones) a partir de datos (vídeo o profundidad), hay que estudiar qué es el aprendizaje automático. Se puede pensar en *Machine Learning* como el uso de algoritmos para adquirir descripciones estructurales a partir de ejemplos de datos en bruto [60]. Las descripciones estructurales son otro término para referirse a los modelos que construimos para contener la información extraída de los datos brutos, y podemos utilizar esas estructuras o modelos para predecir datos desconocidos. Estos modelos pueden adoptar muchas formas, y cada uno de ellos aplica de forma diferente ciertas reglas a los datos conocidos para predecir datos desconocidos, como los siguientes ejemplos:

- **Árboles de decisión**. Crean un conjunto de reglas en forma de estructura de árbol.
- **Regresión lineal**. Crean un conjunto de parámetros para representar los datos de entrada. La regresión se refiere a funciones que intentan predecir una salida de valor real. Este tipo de función estima la variable dependiente conociendo la variable independiente. El tipo de regresión más común es la regresión lineal, la cual intenta generar una función que describa la relación entre "x" y "y", y para valores conocidos de "x" predice los valores de "y" que resultan precisos.
- **Pesos de las redes neuronales**. Las redes neuronales tienen lo que se llama un vector de parámetros que representa los pesos en las conexiones entre los nodos de la red.

2.6.1 Redes neuronales

Las redes neuronales son un modelo computacional que comparte algunas propiedades con el cerebro animal, en el cual muchas unidades simples trabajan en paralelo sin una unidad de control centralizada [60]. La primera propiedad principal que sigue es que la unidad más básica de la red neuronal es la neurona artificial o nodo. Estas neuronas se basan en el comportamiento de las neuronas biológicas del cerebro, por lo que son estimuladas por entradas y transmiten parte de la información que reciben a otras neuronas artificiales. La segunda propiedad principal es que pueden ser entrenadas para únicamente transmitir las señales útiles para lograr un objetivo.

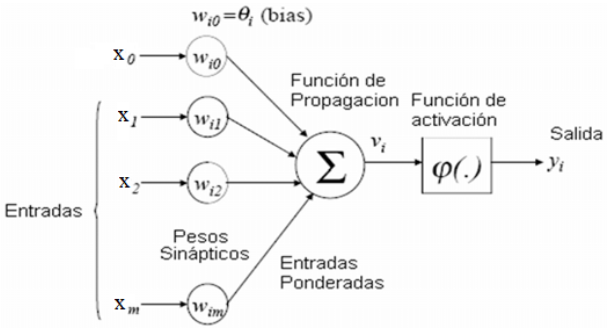


Figura 2.3: Modelo de una neurona artificial [61].

Las neuronas artificiales están compuestas por un núcleo, en el que pondera las entradas, se aplica la función de propagación y de activación; una red o vector de conexiones de entrada; y una salida. Su funcionamiento consiste en aplicar la función de propagación al vector de entrada, lo que implica realizar la suma ponderada de las componentes del vector de entrada y al resultado se le suma un *bias*. Seguidamente se aplica la función de activación, la cual determina el estado de activación de la neurona, obteniéndose finalmente la salida.

2.6.1.1 Pesos (*Weights*)

Los pesos de las conexiones (*weights*) en una red neuronal son coeficientes que escalan (amplifican o minimizan) la señal de entrada a una neurona dada en la red. Se modifican durante el proceso de aprendizaje o entrenamiento, del cual hablaremos más adelante. Cada conjunto de pesos representa una hipótesis específica sobre lo que significan las entradas, es decir, cómo se relacionan con los significados contenidos en las etiquetas. Los pesos representan conjeturas sobre las correlaciones entre la entrada de las redes y las etiquetas objetivo que buscan adivinar. Todos los pesos posibles y sus combinaciones se pueden escribir como el espacio de hipótesis de un problema. El intento de formular la mejor hipótesis es una cuestión de buscar en ese espacio de hipótesis, pudiendo hacerlo mediante el uso de algoritmos de error y optimización.

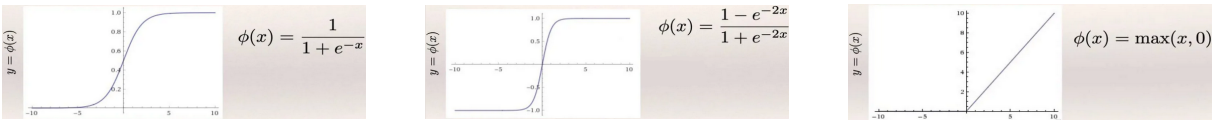
2.6.1.2 Sesgos (*Biases*)

Los sesgos (*biases*) son valores escalares que se agregan a la entrada para garantizar que al menos se activen algunas neuronas por cada capa, independientemente de la intensidad de la señal. Permiten que la red pruebe nuevas interpretaciones o comportamientos, y se modifican durante el proceso de aprendizaje.

2.6.1.3 Funciones de activación

Las funciones de activación deben ser no lineales y provocan la transformación de la combinación de entradas, pesos y sesgos. Los productos de estas transformaciones se ingresan en la siguiente capa de neuronas. Cuando una neurona artificial transmite un valor distinto de cero a otra neurona artificial, se dice que está activada. Las funciones de activación más utilizadas en *deep learning* son: (ver figura 2.4)

- **Sigmoide**. Mapea cualquier rango de valores de entrada a valores entre 0 y 1 a la salida. Cuando la entrada es muy alta o muy baja la salida se satura, complicando la convergencia del algoritmo (en el caso de la minimización del error con el Gradiente Descendente). Esta función es muy útil en la capa de salida de la red neuronal.
- **Tangente hiperbólica**. Mapea cualquier rango de valores de entrada a valores entre -1 y +1 a la salida. Al igual que con la función sigmoide, la salida puede saturar, pero tiene la ventaja de ser simétrica por lo que facilita el entrenamiento.
- **ReLU (Unidad Lineal Rectificada)**. Toma el valor 0 para cualquier valor negativo de "x" y cuando este es positivo toma su propio valor. Al no saturar, el algoritmo convergerá rápidamente, facilitando el entrenamiento.



(a) sigmoide (b) tanh (c) ReLU

Figura 2.4: Funciones de activación más usadas [62].

2.6.1.4 Arquitectura de las redes neuronales

Una red neuronal es la unión de múltiples neuronas que se combinan formando estructuras en capas, y su comportamiento viene definido por su arquitectura: el número de neuronas, el número de capas y el tipo de conexiones entre capas. Esta puede contener tantas capas y neuronas dentro de ellas como deseemos. En la figura 2.5 se puede observar que hay tres capas que desempeñan funciones diferentes:

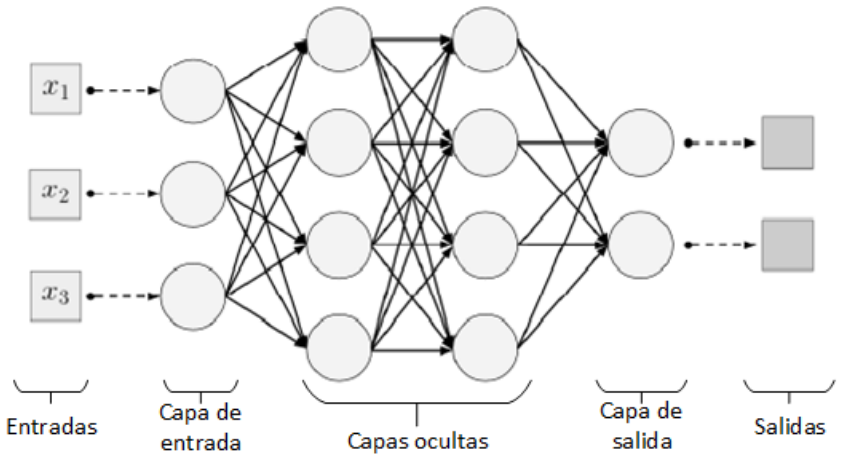


Figura 2.5: Modelo de una red neuronal [60].

- **Capa de entrada**. Es en esta capa en la que las neuronas reciben los datos o señales del exterior y los propaga desde cada neurona de esta capa a las de la siguiente capa. Usualmente el número de neuronas de esta capa es igual al número de características de entrada a la red.
- **Capas ocultas**. Se sitúan entre la capa de entrada y la de salida y el número de estas puede ser desde cero (Red Neuronal monocapa) a cualquier otro valor. Estas se encargan de recibir la información de otras neuronas y de procesarla. Se diferencia de la otras capas en que no tiene conexión directa con el exterior.
- **Capa de salida**. Es la última capa de la red y su función es aportar un resultado al exterior a partir de la información procesada que ha recibido de las capas anteriores. En esta capa habrá tantas neuronas como clases a clasificar.

Las redes neuronales multicapa pueden ser no recurrentes o recurrentes en función del tipo de conexión, y totalmente o parcialmente conectadas en función de su grado de conexión [63]. Las características de cada arquitectura son las siguientes:

- **No recurrentes**. En este caso la información sólo se propaga en un sentido, por lo que no contienen memoria.
- **Recurrentes**. Contiene lazos de realimentación entre las neuronas de una misma capa, entre neuronas de distintas capas, o entre una misma neurona.
- **Totalmente conectadas**. Todas las neuronas de una misma capa se mantienen conectadas con las todas las neuronas de la siguiente capa en las redes no recurrentes, o con las neuronas de la capa anterior en redes recurrentes.
- **Parcialmente conectadas**. Las neuronas de diferentes capas no se conectan en su totalidad con las de otras capas.

Como se ha comentado, se puede escoger el número de capas ocultas y de neuronas en cada capa en función del problema, pero hay que ser cauteloso en la selección, ya que un número alto de capas ocultas y neuronas nos permite obtener un mejor resultado, pero hay que tener en cuenta que superado un número de neuronas la solución empezará a ser cada vez más específica, perdiendo generalización con el conjunto de *test*, lo cual puede suponer un *overfitting* o sobreajuste, concepto que significa que la red se ha sobreentrenado y funcionará muy bien únicamente con los datos utilizados, pero muy mal con cualquier otro. Para que la red sea capaz de generalizar, habrá que entrenarla utilizando un elevado número de datos incluyendo la mayor variabilidad de estos posible. Alguna de las técnicas más comunes que solucionan el problema mencionado son las de **regularización**, las cuales consisten en minimizar la complejidad del modelo y la función de coste, como [64]:

- **Regularización L2 o Ridge**. Implementa una penalización al cuadrado sobre los pesos, lo que significa que cuanto más altos sean los pesos, mayor será la penalización. Esto garantiza que, una vez entrenada la red, el valor de los pesos óptimos sea bajo, lo cual implica que la red utilizará todas sus entradas adecuadamente y estará más diversificada. Minimiza el efecto de la correlación entre los atributos de entrada.

- **Regularización L1 o Lasso**. Uno de los problemas de la regularización L2 es que aunque los pesos resultantes sean más pequeños, la red toma múltiples entradas, aunque tengan menor peso, lo cual supone un problema cuando se trata de entradas ruidosas o irrelevantes. La motivación de esta técnica es eliminar por completo la toma de entradas ruidosas o irrelevantes e, idealmente, asignar un peso nulo a dichas entradas. En este caso, se añade una penalización de primer orden sobre los pesos, de esta forma, lo pesos aprendidos son más dispersos, que son más robustos a las entradas ruidosas. También se puede combinar la regularización L1 y L2, conocido como regularización de red elástica (ElasticNet).

  Los métodos tradicionales para manejar el sobreajuste y realizar la selección de características, como *cross-validation* o *stepwise regression*, funcionan muy bien cuando el conjunto de características es pequeño, pero las dos técnicas anteriores son una gran alternativa cuando estamos ante un gran conjunto de características.

- **Regularización restringida por la norma máxima**. Esta técnica se basa en limitar la norma máxima posible de los vectores de pesos para que se tomen un valor preestablecido como $||\hat{W}||_2 \leq k$, lo cual asegura que los pesos y las actualizaciones de la red estén siempre acotados y no dependan de factores como la velocidad de aprendizaje de la red.

- ***Dropout regularization***. La idea es utilizar un parámetro, p, que define la probabilidad con la que se utilizaran las activaciones de ciertas neuronas de la siguiente capa. Dado que se eliminan las activaciones durante el entrenamiento, es necesario escalar las activaciones adecuadamente para que la fase de prueba no cambie. Para ello se hace un *dropout* invertido, que escala la activación por un factor de $1/p$.

2.6.1.5 Entrenamiento de las redes neuronales

Otro aspecto importante sobre las redes es su entrenamiento. El desarrollo de aprendizaje de cualquier algoritmo de aprendizaje que utilice pesos ponderados, consiste en el proceso de reajustar estos pesos y sesgos haciendo algunos más grandes y otros más pequeños, asignando de esta forma mayor importancia a algunos datos y menor a otros, minimizando la función de coste. Esto ayuda al modelo a aprender que características están vinculadas a qué resultados (etiquetas), y ajusta los pesos y sesgos en consecuencia.

La función que cuantifica cuan cerca está la red del resultado ideal es la **función de pérdida** de los algoritmos de optimización, como por ejemplo el Descenso de Gradiente Estocástico (SGD), los cuales recompensan a la red por las buenas suposiciones y penalizan las malas, desplazando los parámetros de la red hacia la realización de buenas predicciones y alejándose de las malas. Todas las redes aprenden del error y luego actualizan sus pesos para reflejar los errores basándose en una función de coste determinada. La relación entre los pesos de la red y su error se representa mediante la pendiente denominada gradiente. Mediante los errores, el algoritmo de *backpropagation* trabaja hacia atrás para actualizar los pesos de la red en la dirección de los gradientes del error.

La función de pérdida se puede definir de múltiples maneras, dos de las más utilizadas para el entrenamiento de las CNNs son [64]:

- **Pérdidas de *Cross-entropy***. Es una métrica utilizada para medir el rendimiento de un modelo de clasificación[1] como un número entre 0 y 1, siendo 0 un modelo perfecto [65]. Mide la diferencia entre la distribución de probabilidad descubierta de un modelo de clasificación de aprendizaje automático y la distribución predicha. La función de pérdidas es:

$$L = -\sum_{c=1}^{N} y_c log(p_c) \tag{2.2}$$

  siendo N el número de clases.

- **Pérdidas de Hinge**. Esta métrica se utiliza en clasificadores que buscan maximizar el margen, como las Máquinas de Vector Soporte (SVM). Con maximizar el margen se entiende que la distancia mínima entre el hiperplano[2] y las observaciones sea lo más grande posible. La función de pérdidas se define como

$$L_i = \sum_{j \neq y_i} max(0, w_j x_i - w_{y_i} x_i + \delta), \tag{2.3}$$

  donde $\delta = 1$, y es el objetivo, w_y y w_j son los parámetros del modelo.

A continuación se van a comentar los algoritmos de optimización más utilizados en las CNNs [66–69]:

- **SGD**. Este es el algoritmo de optimización más común en el aprendizaje profundo. Es una variante del Gradient Descent[3], pero a diferencia de su predecesor, intenta actualizar los parámetros del modelo con mayor frecuencia, se modifican tras el cálculo de la pérdida en cada ejemplo de entrenamiento. Al actualizarlos con mayor frecuencia la convergencia es mucho más rápida que en el Descenso de Gradiente estándar o que en el Descenso de Gradiente por lotes, el cual sólo actualiza los pesos al finalizar cada época.

  Para mejorar la búsqueda del mínimo, se puede aplicar:

  - **Momentum**. Se inventó para reducir la alta varianza en Descenso de Gradiente (GD) y suavizar la convergencia. Acelera la convergencia hacia la dirección relevante y reduce las fluctuaciones hacia la dirección irrelevante. En este método se utiliza un hiperparámetro conocido como *momentum*, que se suele fijar en 0.9. Este hiperparámetro aumenta cuando el gradiente apunta en la misma dirección y disminuye cuando el gradiente cambia de dirección.

[1] La clasificación es un modelado basado en la delineación de clases de salida en función de algún conjunto de características de entrada [60].

[2] Un hiperplano es una construcción matemática que divide un espacio n-dimensional en partes separadas, por ello es útil en clasificación [60].

[3] Es un método para minimizar la función objetivo a partir de la actualización de los pesos en el sentido opuesto al gradiente de la función objetivo [69]. Es decir, se van actualizando los pesos hasta que se alcanza un mínimo.

  - **Nesterov**. El anterior método puede ser bueno, pero si el *momentum* llega a ser demasiado alto, el algoritmo puede pasar por alto los mínimos locales y seguir aumentando. Para resolver este problema se desarrolló el algoritmo Nesterov Accelerated Gradient (NAG), el cual consiste en un método de anticipación de la posición del siguiente peso, dónde intenta corregir el lugar de cálculo del gradiente, siendo esta la diferencia con el método anterior.

- **AdaGrad**. Una de las desventajas de los optimizadores explicados es que la tasa de aprendizaje es constante para todos los pesos y para cada ciclo. En cambio, por cada paso de tiempo este optimizador disminuye la tasa de aprendizaje para los parámetros asociados a las características que se producen con mayor frecuencia y la aumenta cuando los parámetros asociados a las características son poco frecuentes. Es por ello que funciona muy bien con datos dispersos.

- **RMSProp**. Surge para corregir el decrecimiento drástico de las tasas de aprendizaje. En comparación con AdaGrad, se añade un nuevo meta-parámetro que determina la longitud media del promedio de la estimación de los gradientes al cuadrado. En la práctica, este algoritmo proporciona buenos resultados en el entrenamiento de modelos profundos, siendo hoy en día uno de los métodos más utilizados.

- **Adaptive Moment Estimation (Adam)**. Es otro de los algoritmos con una tasa de aprendizaje adaptativa, se mantiene una tasa de aprendizaje para cada peso de la red y se adapta por separado a medida que se desarrolla el aprendizaje. Este algoritmo puede considerarse como la combinación de dos algoritmos, el de *momentum* y RMSProp. Este algoritmo utiliza directamente la media móvil exponencial del gradiente y del gradiente al cuadrado. Las ventajas de este algoritmo son que reduce la velocidad un poco para realizar una búsqueda cuidadosa, converge muy rápido y que rectifica la tasa de aprendizaje cuando se pasa. Pero esto conlleva un alto coste computacional.

Por último, respecto a las entradas introducidas en el algoritmo de entrenamiento, estas no pueden ser datos en bruto (*raw*) debido a que el *machine learning* está basado en álgebra lineal y en la resolución de *sets* de ecuaciones, las cuales esperan números con coma flotante como entrada (*floating-points*).

2.6.2 Aprendizaje profundo (*Deep Learning*)

El concepto de *Deep Learning* se puede definir como una red neuronal con un gran número de parámetros y capas, que cuenta con las siguientes arquitecturas principales [60]:

- **Redes Neuronales Recurrentes (RNR)**. Permite conexiones arbitrarias entre las neuronas de diferentes capas, pudiendo crear ciclos, lo cual permite tener memoria y analizar secuencias temporales. Es un modelo adecuado para el procesamiento de datos con estructura secuencial.

- **Redes Neuronales Convolucionales (CNN o ConvNet)**. Introducidas en la sección 2.7.

Deep Learning tiene la ventaja sobre otros algoritmos tradicionales de aprendizaje al poder extraer automáticamente las características (*features*). Esto se refiere al proceso en el que la red neuronal decide cuales son las características dentro de una base de datos (*dataset*) que se pueden utilizar para etiquetar los datos de manera fiable.

2.7 Redes Neuronales Convolucionales

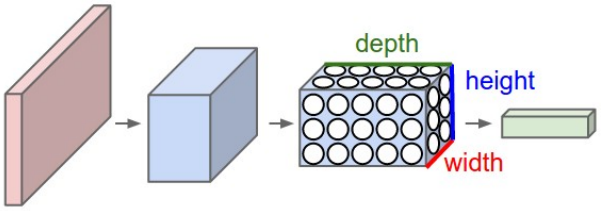
El origen de las CNN surgió a partir de la inspiración en la corteza visual primaria (V1) de los animales, que es la zona del cerebro que se encarga de decodificar la percepción y de transformarla en visión [70].

Las células de la corteza visual están adaptadas para explotar la fuerte correlación espacial local que se encuentra en los tipos de imágenes que nuestro cerebro procesa, y actúan como filtros locales sobre el espacio de entrada.

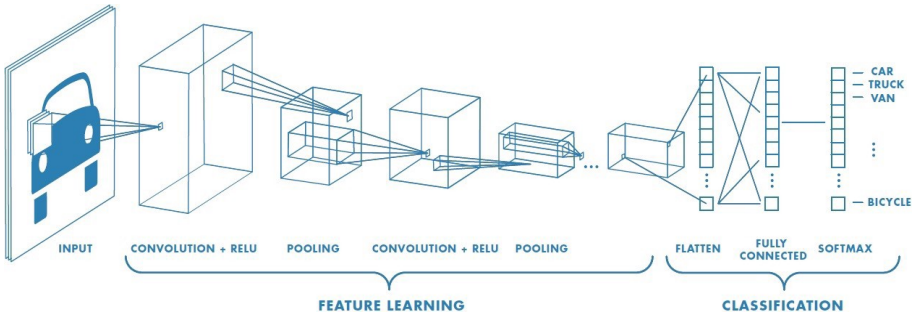
El objetivo de las CNN es aprender características de mayor orden a partir de la convolución de los datos. Este tipo de redes son capaces de capturar las dependencias espaciales y temporales de los datos aplicando filtros, por lo que son especialmente útiles en el procesamiento de señales de audio y de imágenes. Funcionan muy bien para detectar objetos y seres vivos en imágenes y también desempeñan un buen trabajo en el análisis de palabras y de sonidos, aunque es más eficaz en reconocimiento, clasificación y segmentación de imágenes.

2.7.1 Arquitectura de las CNNs

A diferencia de las redes neuronales tradicionales, las CNN organiza sus neuronas en 3 dimensiones: ancho, alto y profundidad (número de canales), que en el caso de las imágenes RGB sería equivalente al color. Por ejemplo: una imagen de 32 píxeles de ancho por 32 píxeles de alto se separaría a la entrada de la red en tres canales, uno por cada color primario de la luz, teniendo los datos de entrada un tamaño de $32 \times 32 \times 3$, lo cual supondría el uso de $32 \cdot 32 \cdot 3 = 3072$ neuronas en la capa de entrada.



(a) Esquema de una CNN [71].



(b) Esquema de las capas de una CNN [72].

Figura 2.6: Red Neuronal Convolucional.

Con imágenes de altas dimensiones el coste computacional también sería alto, así que las CNN tienen como función reducir las imágenes a un formato más sencillo de procesar sin perder las características fundamentales para obtener una buena predicción.

En la figura 2.6b se puede apreciar las dos fases de la red empleada para la clasificación: las fase de extracción de características y la de clasificación. Las principales capas que forman la CNN son: la capa de entrada, la cual contiene los valores en bruto de los píxeles de una imagen; la capa de convolución; la capa de activación; la capa *pooling*; la capa *fully-connected*; y la capa de salida. A continuación se entrará en el detalle de estas capas.

2.7.1.1 Capa de convolución

La capa de convolución es el núcleo de las CNN. Esta capa es la que se encarga de realizar las convoluciones, es decir, toma un grupo de píxeles de la imagen de entrada y realiza la operación del producto escalar por una matriz que constituye los pesos, deslizando la matriz por todo lo ancho y alto de la imagen de entrada, como se puede observar en la figura 2.7. La matriz mencionada se denomina *kernel* y al conjunto de *kernels*, filtro. Estos habitualmente tienen un tamaño en anchura y altura menor que la imagen de entrada y el mismo en profundidad. A medida que se va deslizando el filtro por la imagen, se va a producir un mapa de activación (*feature mapping*) que indica dónde están localizadas las características de la imagen. El número de veces que se desplazará el filtro y a los píxeles a los que afectará dependerá de un factor denominado *stride*, el cual establece el número de píxeles que se saltará el filtro a la hora de realizar la convolución.

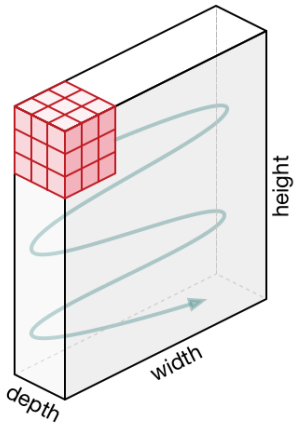


Figura 2.7: Movimiento del filtro [72].

Los hiperparámetros que dictan la disposición espacial y el tamaño del volumen de salida de una capa convolucional son:

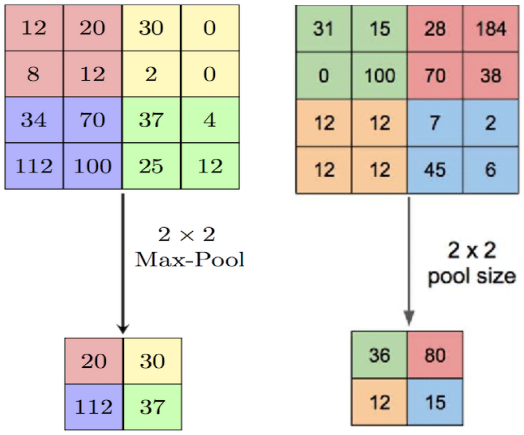
- **Tamaño del filtro**. Anchura y altura del filtro.
- **Profundidad de la salida**. Controla el número de neuronas de la capa convolucional que se conectan a la misma región del volumen de entrada.
- **Stride**. Ajustando el parámetro a valores bajos se asignarán más columnas de profundidad en el volumen de salida. Esto también producirá un mayor solapamiento de los campos receptivos entre las columnas, lo que conduce a volúmenes de salida más grandes. Lo contrario ocurre cuando especificamos valores de *stride* más altos. Estos valores más altos nos dan menos solapamiento y volúmenes de salida más pequeños.
- **Zero-padding**. Con este parámetro se indica el número de filas y columnas de ceros a añadir antes y después de los datos de entrada. Permite controlar el tamaño espacial del volumen de salida

2.7.1.2 Capa pooling

Este tipo de capas se suelen introducir entre capas convolucionales sucesivas, con el objetivo de reducir progresivamente el tamaño espacial (tanto anchura como altura) de los datos.

Las dos operaciones más típicas de *downsampling* son: la operación *max pooling* y el *average pooling*. La primera consiste en calcular el valor máximo de las celdas de los mapas de características para generar

un mapa más reducido [73]. Y el segundo radica en calcular el valor medio de las celdas de un mapa de características para crear otro más reducido. Este último consigue extraer las características más suaves, mientras que el *max pooling* obtiene las características más pronunciadas, como pueden ser los bordes [74].



(a) Operación *max pooling* [73].

(b) Operación *average pooling* [74].

Figura 2.8: Operaciones típicas de *downsampling*.

2.7.1.3 Capa completamente conectada (*fully-connected*)

Se utiliza esta capa para calcular las puntuaciones de las clases que se utilizan a la salida de la red. Las dimensiones de esta capa de salida es $1 \times 1 \times N$, siendo N el número de clases que se está evaluando. Esta capa está completamente conectada con todas sus neuronas y con las neuronas de la capa anterior.

2.7.2 Arquitecturas de CNNs más conocidas

Las siguientes arquitecturas son algunas de las más destacadas dentro de las CNN por su gran utilidad y la influencia que han tenido a lo largo de los años [60, 75].

- **LeNet** [76]. Desarrollada por Yann LeCun en 1989 fue una de las primeras arquitecturas exitosa de CNN propuesta. Originalmente se utilizaba para leer dígitos escritos a mano en imágenes.
- **AlexNet** [77]. La desarrolló Alex Krizhevsky, Ilya Sutskever y Geoff Hinton basándose en la arquitectura de LeNet. Ganó el ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 y fue una de las primeras arquitecturas en utilizar la Unidad de Procesamiento Gráfico (GPU) para aumentar su rendimiento. Ayudó a popularizar las CNN en visión por ordenador, y a día de hoy el *paper* que publicaron es considerado uno de los más influyentes en el área de la visión por ordenador.
- **ZF Net** [78]. La crearon Matthew Zeiler y Rob Fergus, ganando el ILSVRC 2013. Se trata de una mejora de AlexNet mediante el ajuste de los hiperparámetros de la arquitectura, en particular ampliando el tamaño de las capas convolucionales intermedias y reduciendo el tamaño del *stride* y del filtro en la primera capa. Se basa en el modelo de Zeiler y Fergus, que fue entrenado en el conjunto de datos ImageNet. Este modelo es computacionalmente más eficiente que AlexNet, gracias a la introducción de una etapa de inferencia aproximada mediante capas deconvolutivas en medio de la red.

- **GoogLeNet** [79]. Desarrollado por Christian Szegedy y su equipo en Google, ganó la tarea de clasificación de ILSVRC 2014. Introduce una tasa de error bastante menor que los anteriores ganadores (ZF Net y AlexNet). Consigue una arquitectura más profunda empleando una serie de técnicas distintas, como la convolución 1×1 y el *average pooling*. Para reducir el número de parámetros que debe aprender utiliza capas *unpooling* encima de la red para eliminar la redundancia espacial durante el entrenamiento y también presenta conexiones de acceso directo entre las dos primeras capas convolucionales previo a la adición de nuevos filtros en las capas posteriores de la red.

- **VGGNet** [80]. Creado por Karen Simonyan y Andrew Zisserman fue subcampeón en la tarea de clasificación de ILSVRC 2014. Demostró que la profundidad de la red era un factor crítico para el buen rendimiento de esta. Su modelo es computacionalmente eficiente y sirve como una fuerte línea base para muchas aplicaciones en visión por ordenador debido a su aplicabilidad en numerosas tareas, incluyendo la detección de objetos. Sus representaciones de características profundas se utilizan en múltiples arquitecturas de redes neuronales como YOLO, SSD, etc.

- **ResNet** [81]. Desarrollado por Shaoqing Ren, Kaiming He, Jian Sun y Xiangyu Zhang, ganando la tarea de clasificación de ILSVRC 2015. La red tiene 152 capas y más de un millón de parámetros, lo que se considera bastante profundo. Las CNN se utilizan sobre todo para tareas de clasificación de imágenes con 1000 clases, pero ResNet demuestra que las CNN también pueden utilizarse con éxito para resolver problemas de procesamiento del lenguaje natural, como la finalización de frases o la comprensión automática, donde fue utilizada por el equipo de Microsoft Research Asia en 2016 y 2017 respectivamente. Esta arquitectura es eficiente desde el punto de vista computacional y puede ampliarse o reducirse para adaptarse a la potencia de cálculo de las GPUs.

Capítulo 3

Desarrollo

3.1 Introducción

En este capítulo se mostrará la arquitectura de la red CNN escogida y mencionada en el apartado 2.3, la cual nos servirá para obtener las posiciones tridimensionales de ciertas articulaciones del cuerpo humano, a partir de las cuales se calcularán en el capítulo siguiente estimaciones de destreza, que incluyen detector de caídas y distintas métricas agregadas.

3.2 Descripción de la red aplicada

Como se mencionó en la sección 2.3, en el presente trabajo se propone seguir la arquitectura de la red presentada en el *paper* **"A2J: Anchor-to-Joint Regression Network for 3D Articulated Pose Estimation from a Single Depth Image"** [33].

Tal y como introducen en el estudio, la mayoría de los enfoques más avanzados de estimación de la postura de la mano y del cuerpo en 3D se basan en el aprendizaje profundo, sin embargo, estas siguen sufriendo algunos defectos. El primero siendo los métodos Fully Convolutional Network (FCN)[1] basados en el codificador-decodificador, los cuales se entrenan generalmente con mapas térmicos gausianos no adaptativos para diferentes articulaciones, soportando una carga computacional relativamente alta. Además, la mayoría de ellos no pueden ser entrenados completamente de principio a fin para la tarea de estimación de la pose en 3D. Es por ello que han abordado el problema de la estimación 3D de la postura de la mano y del cuerpo utilizando un nuevo enfoque denominado red de regresión Anchor-to-Joint (A2J). Esta red tiene la capacidad de aprender de extremo a extremo. La principal idea de A2J es predecir la posición de las articulaciones en 3D mediante la agregación de los resultados de estimación de múltiples puntos de anclaje, con el objetivo de mejorar la generalización del aprendizaje. En concreto, los puntos de anclaje pueden considerarse como los regresores locales hacia las articulaciones desde diferentes puntos de vista y distancias. Estos están densamente establecidos en la imagen de profundidad para capturar la información del contexto espacial global-local. Cada uno de ellos contribuirá a la regresión de las posiciones de todas las articulaciones en función de sus pesos. Finalmente, la articulación se localiza agregando los resultados de todos los puntos de anclaje.

[1] Es una arquitectura que principalmente se utiliza para la segmentación semántica, la cual consiste en clasificar los píxeles de una imagen de forma que los píxeles que pertenecen a una misma clase se asocien con su misma etiqueta. Es equivalente a una CNN sin capas totalmente conectadas [82].

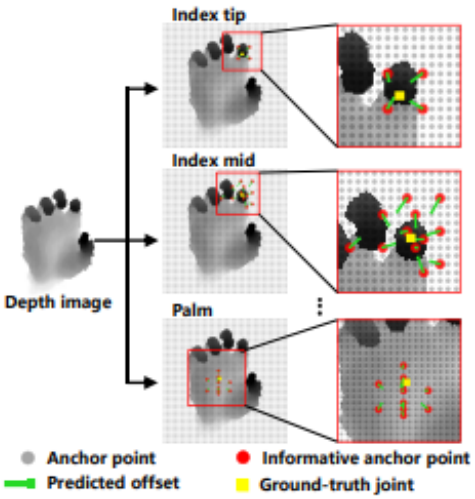


Figura 3.1: Representación de la idea de los puntos de anclaje [33].

Para cada articulación no todos los puntos de anclaje contribuyen por igual. Por consiguiente, proponen un procedimiento para descubrir los puntos de anclaje informativos hacia la determinada articulación mediante la asignación de pesos.

Durante el entrenamiento se tienen en cuenta dos factores: el error de estimación de los puntos de anclaje y la disposición espacial de los puntos de anclaje informativos. Los puntos de anclaje informativos escogidos son alentados a rodear uniformemente la articulación correspondiente para aliviar el *overfitting*. Se puede observar en la figura 3.1, que diferentes articulaciones poseen distintos puntos de anclaje informativos. Si la articulación es pequeña, como la de la punta del índice (*index tip*), esta tiene pocos puntos de anclaje informativos. Mientras que una articulación situada en una zona más amplia, como la articulación de la palma de la mano (*palm*), poseen muchos más, con el fin de capturar mayor información espacial.

Para entrenar la red tuvieron en cuenta a los dos enfoques siguientes, descartando finalmente el segundo por sus desventajas.

El enfoque basado en el aprendizaje 2D

Este enfoque tiene una gran capacidad de ajustarse para caracterizar patrones visuales. Recientemente se han introducido las CNNs en el dominio 3D mediante la regresión global y la detección local. La regresión global no consigue mantener bien la información del contexto espacial local debido a la operación de agregación de características globales dentro de las capas totalmente conectadas. Para mejorar el rendimiento se suele optar por abordar este problema mediante un modelo de codificador-decodificador (por ejemplo, FCN), estableciendo un mapa de calor local para cada articulación, sin embargo, el ajuste del mapa de calor sigue sin ser adaptativo para las diferentes articulaciones. Además, la operación de deconvolución requiere de mucho tiempo, aparte de que la mayoría de los métodos basados en el codificador-decodificador no pueden ser entrenados completamente de extremo a extremo.

El enfoque basado en el aprendizaje 3D

Para revelar mejor la propiedad 3D dentro de una imagen de profundidad y mejorar el rendimiento, últimamente se tiende a recurrir a las CNN 3D y a los conjuntos de puntos. Los métodos basados en

CNN 3D realizan una voxelización[2] de la imagen de profundidad en una representación volumétrica, para seguidamente ejecutar una operación de convolución o deconvolución 3D de forma que se capturen las características visuales 3D. Sin embargo, la CNN 3D es relativamente difícil de ajustar debido al gran número de parámetros convolucionales. Además, la operación de voxelización en 3D conlleva una gran carga computacional tanto en el almacenamiento de la memoria como en el tiempo de ejecución. Con respecto a la red de puntos, la cual transfiere la imagen de profundidad a una nube de puntos como entrada, requiere de varios procedimientos que consumen mucho tiempo (como el muestreo de puntos o la búsqueda de los k vecinos más próximos), por lo que es menos eficiente.

3.2.1 Arquitectura de la red

La arquitectura de la red A2J consiste en 3 ramas y una red troncal convolucional 2D, en concreto la red ResNet-50 [35]. Las 3 ramas se encargan de predecir en el plano los *offsets* entre los puntos de anclaje y las articulaciones, de estimar el valor de la profundidad de la posición de las articulaciones, y de proponer puntos de anclaje informativos respectivamente.

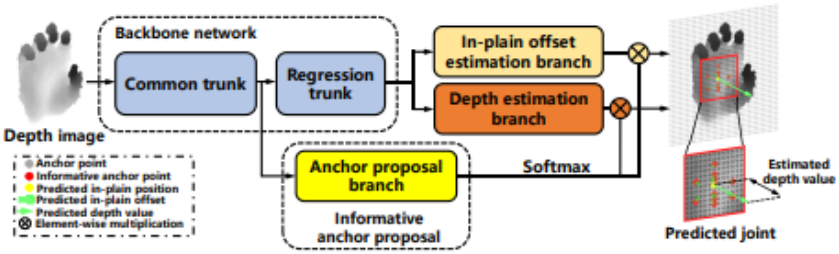


Figura 3.2: Red A2J. Consta de una red troncal y tres ramas funcionales: la rama de predicción de *offsets* en el plano, la rama de estimación de profundidad y la rama de propuesta de anclaje informativo [33].

La motivación que tuvieron para construir la red A2J sobre una CNN 2D fue porque:

1. La información 3D ya está incluida en una imagen de profundidad. Al utilizar la CNN 2D, esta todavía puede revelar las características 3D de los datos de la imagen de profundidad original.
2. A diferencia de una CNN 3D y de una red de puntos, la CNN 2D se puede preentrenar con conjuntos de datos a gran escala (por ejemplo, ImageNet [84]), lo cual puede llegar a mejorar la capacidad de la red de capturar patrones visuales en imágenes de profundidad.
3. La CNN 2D es altamente eficiente sin la operación de convolución 3D, que consume tiempo, y sin los procedimientos de preprocesamiento (por ejemplo, voxelización y muestreo de puntos).

3.2.1.1 Rama de predicción de los *offsets* en el plano y de estimación de la profundidad

Estas dos ramas se encargan, *grosso modo*, de predecir la posición 3D de las articulaciones del cuerpo humano. Están colocados a la salida de la red troncal (ver figura 3.2). Están formadas por cuatro capas convolucionales intermedias de tamaño 3×3 con 256 canales, incluyendo *batch normalization* y ReLU. Debido a que el mapa de características es un *downsampling* de 16 de la imagen de profundidad de entrada, y el ajuste del *stride* es igual a 4, un punto del mapa de características corresponderá a $4 \cdot 4 = 16$ puntos de anclaje en la imagen de profundidad. Suponiendo K articulaciones, la salida de la rama de

[2]Referido al proceso de conversión y segmentación de las imágenes que contienen información geométrica en una imagen rasterizada (o mapa de bits) [83]

predicción en el plano de los *offsets* entre los puntos de anclaje y las articulaciones tiene $16 \times K \times 2$ canales y la de la rama de estimación de la profundidad $16 \times K \times 1$.

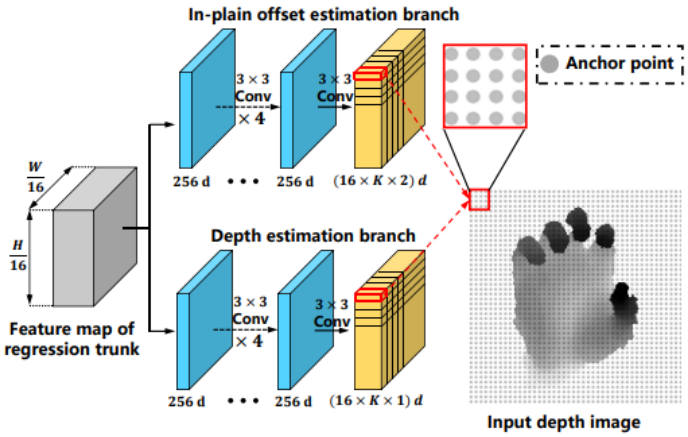


Figura 3.3: Rama de predicción en el plano de los *offsets* entre los puntos de anclaje y las articulaciones y de estimación del valor de la profundidad de la posición de las articulaciones (W y H son la anchura y altura de la imagen de profundidad respectivamente) [33].

3.2.1.2 Rama de propuesta de puntos de anclaje informativos

Esta rama se encarga de descubrir los puntos de anclaje informativos para una articulación determinada mediante la asignación de pesos. La decisión de cuales son los puntos informativos se toma siguiendo la ecuación 3.3 mostrada en la sección 3.6. Está colocada a la salida del mapa de características de la rama troncal común, previa a la regresión (ver figura 3.2). Al igual que las ramas anteriores, esta también cuenta con 4 capas convolucionales intermedias y una de salida, consiguiendo a la salida $16 \times K \times 1$ canales.

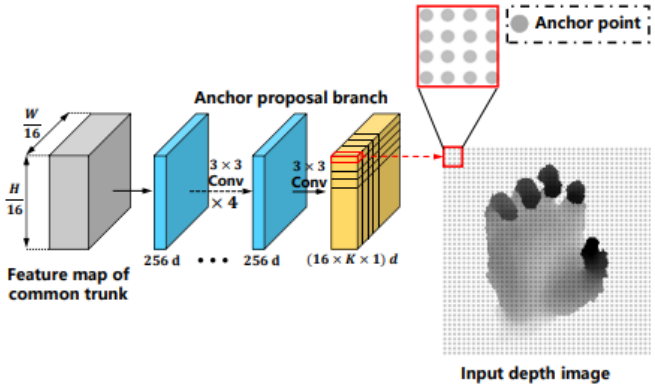


Figura 3.4: Rama de propuesta de puntos de anclaje informativos (W y H son la anchura y altura de la imagen de profundidad respectivamente) [33].

3.2.1.3 Red troncal, ResNet-50

La red A2J hace uso de la red ResNet-50, como su red troncal. Las capas de convolución de la 0 a la 3 corresponden a la rama troncal común, y la capa 4 a la regresión. Para adecuar la red ResNet-50 para que pudiera estimar poses, modificaron el tamaño del *stride* a 1 en la capa 4, consiguiendo a la salida un tamaño mayor al que proporcionaba la red originalmente, de esta forma se mantendrá mejor

la información espacial. Mientras tanto, la operación de convolución dentro de la capa 4 se sustituye por una convolución dilatada, con una dilatación de 2 para ampliar el campo receptivo. Suponiéndose un tamaño del *kernel* de 3×3, al utilizar una dilatación de 2, el campo receptivo será más grande, el mismo que si se utilizase un *kernel* de 5×5, pero sólo se requerirá de 9 parámetros en vez de 25 (los 16 píxeles restantes del *kernel* se consideran nulos). Este tipo de convolución se utiliza a menudo en el área de la segmentación de imágenes en tiempo real cuando la capa de red necesita un campo receptivo más grande, pero los recursos informáticos son limitados y el número o tamaño del núcleo de convolución no se puede aumentar [85].

3.3 Entrenamiento de la red

Las imágenes de entrada a la red A2J se generan recortando la zona de la imagen donde se encuentra el cuerpo de la persona utilizando un *bounding box* y se escala a un tamaño fijo de 288×288, el cual en el supuesto de ser una mano la imagen de entrada, esta se escala a un tamaño de 176×176.

Para la articulación j, el punto objetivo en el plano, T_j^i, indica el *ground truth* en coordenadas de píxel transformadas según la región recortada. Para que T_j^i y el objetivo en la tercera dimensión (profundidad), T_j^d, estén en magnitudes comparables, transforman la profundidad del *ground truth* de la articulación j, Z_j, como:

$$T_j^d = \mu(Z_j - \theta) \tag{3.1}$$

donde μ y θ son los parámetros de transformación. En el caso de la pose del cuerpo humano, $\mu = 50$ y $\theta = 0$, el cual representa la profundidad del punto central y que tiene sentido usarlo de ser la estimación de la pose de la mano, ya que esta se encontrará centrada y ese punto corresponderá al punto central de la mano. El valor de μ es igual a 0 en el supuesto de la estimación de la pose de la mano y 50 el otro, y es un factor de escalado establecido empíricamente que utilizan para optimizar el modelo. Al final, durante la fase de prueba, el resultado de la predicción será devuelto a las coordenadas del mundo real.

Para supervisar el entrenamiento de extremo a extremo de la red se utilizan las funciones de pérdidas de estimación de la posición de las articulaciones y la del entorno de los puntos de anclaje, esta última ayudando especialmente a reducir el sesgo del punto de vista. Están definidas en la sección 3.6.

Previamente a la fase de entrenamiento se realiza una serie de procesos y transformaciones para mejorar el resultado de este, los cuales se han ampliado a continuación [64]:

3.3.1 Preprocesado

El primer proceso de transformación realizado es el preprocesado, el cual incluye la acción de estandarización que consiste en:

1. **Sustracción de la media**. En este paso se calcula la media de la profundidad de todas las imágenes de la base de datos, para posteriormente restar esta media calculada a cada imagen del conjunto de datos, con el objetivo de centrar los datos en el origen a lo largo de cada una de las dimensiones de las características.

2. **Normalización**. Seguido del paso anterior, se escalan todas las dimensiones de las características. Proceso conseguido al dividir cada característica por la desviación estándar del conjunto de datos.

3.3.2 *Data augmentation*

Uno de los recursos más comunes para mejorar el rendimiento de la red cuando el conjunto de datos de entrenamiento es pequeño (del orden de 10^2 a 10^3) es aumentar los datos de entrenamiento alterando ligeramente las imágenes accesibles de la base de datos. Existen múltiples estrategias para conseguir este efecto y en este modelo aplican de las más comunes, que son [64]:

- **Traslación y rotación**. Para que la red detecte un mismo objeto independientemente de la traslación y rotación, se transforma la perspectiva de las imágenes de diferentes formas. Se puede voltear vertical y horizontalmente, rotar un cierto ángulo y desplazar unos píxeles vertical y horizontalmente una imagen y después añadirla al conjunto de datos.
- **Invariabilidad de escala**. Una de las limitaciones de las CNNs es su ineficacia para reconocer objetos a diferentes escalas. Para enmendar este problema, se suele aumentar el conjunto de entrenamiento realizando recortes de las imágenes accesibles.
- **Adición de ruido Gausiano (*Random Erasing*)**. Añade distorsión a las imágenes, pero la red neuronal aprenderá a pasar por alto el ruido. Consiste en seleccionar aleatoriamente una región rectangular de una imagen y borrar sus píxeles con valores aleatorios [86], consiguiendo un resultado similar al del ruido sal y pimienta. Teniendo en cuenta que demasiado poco ruido no tiene ningún efecto, mientras que mucho ruido hace que la función de mapeo sea demasiado difícil de aprender.
- **Perturbación del color**. Tal y como indica el nombre, este proceso consiste en perturbar directamente los valores de color de la imagen de entrada.

En nuestro caso concreto, al trabajar con imágenes de profundidad, la operación de modificar el color de las imágenes de entrada no es plausible, por lo que únicamente se aplica la rotación, el escalado y el ruido. En cuanto al ruido añadido, este se añade con una probabilidad del 50 %.

3.3.3 Ajuste fino (*Fine Tuning*) de CNNs

A menudo, nos encontramos con un conjunto de datos pequeño y el entrenamiento de una CNN en tales conjuntos de datos puede conducir a un exceso de ajuste en los datos de entrenamiento. El *fine tuning* de una CNN es una de las técnicas que pretende solucionar este problema. Esta técnica consiste en entrenar la CNN partiendo de un modelo previamente entrenado, de modo que, durante el proceso de entrenamiento que se ejecuta, se adaptan y cambian los pesos del modelo para que estos se ajusten mejor al contexto de la aplicación [64]. Esta estrategia tiene múltiples ventajas:

- Explota el gran número de modelos preentrenados disponibles para adaptación.
- Reduce el tiempo de cálculo, ya que la red ya ha aprendido filtros estables, y converge rápidamente para refinar los pesos en un nuevo conjunto de datos.
- Consigue evitar el sobreajuste aunque trabaje con un conjunto de datos de entrenamiento pequeño.

Existen múltiples maneras de aplicar este método, entre las que podemos destacar:

- **Extractor de características CNN**. Cuando nos enfrentamos a una tarea de clasificación con un número específico de clases, se puede aprovechar los modelos preentrenados existentes con hasta 1000 clases, como AlexNet, y eliminar la última capa *fully connected*, manteniendo el resto de la

red tal y como está. A continuación, se realiza un *forward pass*[3] para cada imagen de entrada y se capturan las activaciones de todas las capas convolucionales. El vector de características obtenido se puede utilizar ahora con cualquier clasificador de aprendizaje automático.

- **Adaptación de la CNN**. En ocasiones se desea aprovechar las capas totalmente conectadas de las propias redes para la tarea de clasificación. En estos casos, se puede sustituir la última capa totalmente conectada de la red preentrenada por una propia que tenga el número correcto de clases de salida para la aplicación. Una vez configurada esta nueva red, se copian los pesos de la red preentrenada y se utilizan para inicializar la nueva red. Por último, se ejecuta la nueva red a través del algoritmo *backpropagation* para adaptar los nuevos pesos de la red a su aplicación y datos particulares.

- **Reentrenamiento de la CNN**. Esta estrategia es útil cuando se necesita entrenar una CNN desde cero. Ya que el entrenamiento de una red CNN puede llegar a durar varios días, se recomienda inicializar los pesos de la red con un modelo preentrenado y comenzar el entrenamiento de la red desde el punto en que la red preentrenada detuvo su entrenamiento.

En la red A2J han aplicado la adaptación de la CNN modificando el tamaño de la última capa y parte de los pesos de un modelo preentrenado en la red ResNet50 con la base de datos ImageNet.

3.3.4 Inicialización de la red

Uno de los aspectos cruciales del entrenamiento de las redes convolucionales es la inicialización de la red [64]. Ya se mencionó que cada capa de la CNN tiene unos pesos que se entrenan sobre el conjunto de entrenamiento, y que para ello se utiliza un algoritmo de optimización. Pues las entradas al algoritmo incluyen un conjunto inicial de pesos, una función de pérdida y datos de entrenamiento etiquetados. El algoritmo utilizará los pesos iniciales para calcular un valor de pérdida teniendo en cuenta las etiquetas de los datos de entrenamiento y ajustará su peso para reducir la pérdida. Este peso ajustado pasará a la siguiente iteración, en la que el proceso anterior continuará hasta alcanzar la convergencia. Como puede verse en este proceso, la elección del peso inicial para la inicialización de la red desempeña un papel crucial en la calidad y la velocidad de convergencia del entrenamiento de la red. Por ello, se pueden aplicar varias estrategias para abordar esta cuestión. Algunas de ellas son las siguientes:

- **Inicialización aleatoria**. Consiste en asignar inicialmente a todos los pesos un valor aleatorio. Preferiblemente, que las muestras aleatorias procedan de una distribución gausiana de media cero y desviación estándar unitaria. Con este método, al asignar los pesos de forma asimétrica, se puede asegurar la diversidad de la red, pero hay que tener en cuenta que la distribución de la salida de las neuronas tendrá una varianza mucho mayor, por lo que se recomienda normalizar los pesos con el número de entradas a la capa.

- **Inicialización dispersa**. En este caso, se escoge una neurona al azar y se conecta aleatoriamente a K neuronas de la capa anterior, siendo los pesos de cada neurona aleatorios, como se ha descrito anteriormente. La idea de este método es desacoplar el número de conexiones a cada neurona del número de neuronas en las capas anteriores.

- **Normalización por lotes (*batch*)**. Pretende ser robusta ante los problemas de una mala inicialización de la red. El concepto es forzar a toda la red a normalizarse en una distribución gausiana unitaria al comienzo de la fase de entrenamiento, generando una versión normalizada por lotes.

[3] Se refiere al proceso de cálculo de los valores de las capas de salida a partir de los datos de entrada. Atraviesa todas las neuronas desde la primera a la última capa.

- **Inicialización Xavier**. Su principal objetivo era evitar los problemas de desaparición del gradiente y de pesos demasiado grandes, ya que el *backpropagation* crece proporcionalmente al valor de los pesos. Es decir, si los pesos de una red comienzan siendo demasiado pequeños, la señal se reduce al pasar por cada capa hasta que es demasiado pequeña para ser útil, sin embargo, si los pesos de una red comienzan siendo demasiado grandes, la señal crece a medida que pasa por cada capa hasta que es demasiado grande para ser útil.

En A2J se inicializan las capas convolucionales del modelo con Xavier y en las de *batch normalization* los pesos se inicializan a 1 y el *bias* a 0.

3.3.5 Optimizador aplicado

Como ya se introdujo en la subsección 2.6.1, la función que cuantifica cuan cerca está la red durante el entrenamiento del resultado ideal, es la función de pérdida de los algoritmos de optimización. Este es el encargado de ir mejorando los pesos en cada época. El optimizador que aplican es Adam, el cual se explicó en la subsección 2.6.1.

3.4 Tasa de aprendizaje

La tasa de aprendizaje es fundamental para una convergencia adecuada, y determina en cada época cuánto deben ser actualizados los pesos del modelo proporcionalmente al Gradiente de la función de pérdidas (ver figura 3.5).

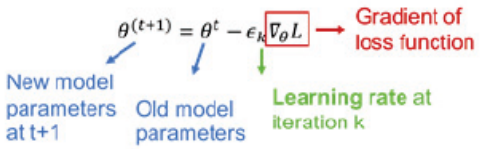


Figura 3.5: Actualización de los pesos del modelo [64].

Escoger la tasa de aprendizaje adecuada puede resultar complicado. Una tasa de aprendizaje demasiado pequeña conduce a un entrenamiento y una convergencia muy lentos, mientras que una tasa de aprendizaje demasiado alta puede hacer que la función de pérdidas fluctúe en torno a un mínimo local o incluso diverja, como se muestra en la figura 3.6.

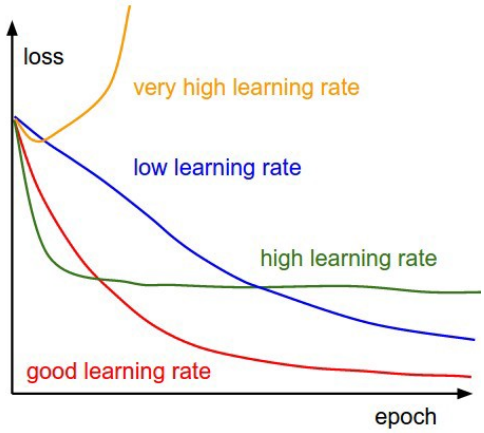


Figura 3.6: Efecto de las tasas de aprendizaje en la convergencia [87].

Hay varias formas de configurar la tasa de aprendizaje [64]:

- **Constante**. Se mantiene la tasa de aprendizaje igual para todas las épocas.
- **Decaída por pasos**. La tasa de aprendizaje decae cada cierto número de épocas.
- **Decaída inversa**. La tasa de aprendizaje decae como la inversa del tiempo.
- **Decaída exponencial**. La tasa de aprendizaje decae como una función exponencial natural.

A menudo se prefiere la decaída por pasos, ya que es el más sencillo, intuitivo y funciona bien empíricamente. Además sólo hay que especificar el hiperparámetro t, el cual hace referencia al número de épocas que tienen que pasar para reducir la tasa de aprendizaje un valor especificado. Precisamente esta es la configuración que siguen en A2J.

3.5 Parada temprana (*Early stopping*)

La parada temprana se refiere a la forma en que entrenamos la red controlando el rendimiento con un conjunto de validación separado y deteniendo el entrenamiento cuando las pérdidas de validación dejan de mejorar. Esta forma de regularización no la tienen aplicada en la red, si no que han configurado el número de épocas a base de pruebas. En el capítulo siguiente se mostrarán los resultados con la detención temprana añadida.

3.6 Planteamiento matemático

Una determinada articulación se localiza agregando las salidas de todos los puntos de anclaje. Dado que no todos los puntos de anclaje contribuyen por igual a una determinada articulación, se les asigna un peso a través de la rama de "propuesta de anclaje para descubrir los más informativos". Como consecuencia, la posición en el plano y el valor de profundidad de la articulación j pueden obtenerse como la media ponderada de los resultados de todos los puntos de anclaje:

$$\begin{cases} \hat{S}_j = \sum_{a \in A} \hat{P}_j(a)(S(a) + O(a)) \\ \hat{D}_j = \sum_{a \in A} \hat{P}_j(a) D_j(a) \end{cases} \tag{3.2}$$

donde $\hat{S}_j$ y $\hat{D}_j$ indican la posición del plano estimada y la profundidad de la articulación j. $S(a)$, $O_j(a)$ y $D_j(a)$ significa la posición en el plano del punto de anclaje a, la diferencia en distancia estimada entre el punto de anclaje a y la articulación j, y la profundidad predicha de la articulación j a partir del punto de anclaje a, respectivamente. $\hat{P}_j(a)$ es el peso normalizado entre el punto de anclaje a y la articulación j, y se obtiene aplicando la función *softmax*:

$$\hat{P}_j(a) = \frac{e^{\hat{P}_j(a)}}{\sum_{a \in A} e^{\hat{P}_j(a)}} \tag{3.3}$$

Como regla empírica, se toma que un punto de anclaje a con $\hat{P}_j(a) > 0{,}02$ será seleccionado como un punto de anclaje informativo para la articulación j.

A continuación se van a mostrar las pérdidas de estimación de la posición de las articulaciones (ecuación 3.4) y la del entorno de los puntos de anclaje (ecuación 3.6):

$$loss_1 = \alpha \sum_{j \in J} L_{\tau_1} \left(\sum_{a \in A} \tilde{P}_j(a)(S(a) + O_j(a)) - T_j^i \right) + \sum_{j \in J} L_{\tau_2} \left(\sum_{a \in A} \tilde{P}_j(a) D_j(a) - T_j^d \right) \tag{3.4}$$

donde $\alpha = 0{,}5$ y es el factor que equilibra el desplazamiento en el plano y la estimación de la profundidad. T_j^i es el *ground truth* en 2D de una articulación j acorde a la región recortada y T_j^d hace referencia al valor de profundidad de la articulación j objetivo.

$$L_\tau(x) = \begin{cases} \frac{1}{2\tau} x^2, & \text{cuando } |x| < \tau \\ \\ |x| - \frac{\tau}{2}, & \text{en cualquier otro caso} \end{cases} \tag{3.5}$$

En la ecuación 3.4 cuando $\tau = \tau_1$, τ toma el valor 1, y se procede a calcular las pérdidas $smooth_{L1}$ mediante la ecuación 3.5. Cuando $\tau = \tau_2$, este adquiere el valor 3, e igual que antes, se continúa evaluando las pérdidas utilizando ese valor de τ. Estos valores los han obtenido empíricamente y los han aplicado porque los datos de profundidad son relativamente ruidosos.

La siguiente definición de función de pérdida surge para mejorar la capacidad de generalización de A2J, situando los puntos de anclaje informativos alrededor de las articulaciones con el fin de observarlas desde múltiples puntos de vista simultáneamente. Por lo tanto, las pérdidas de los anclajes informativos las han definido como:

$$loss_2 = \sum_{j \in J} L_{\tau_1} \left(\sum_{a \in A} \tilde{P}_j(a) S(a) - T_j^i \right) \tag{3.6}$$

Finalmente, las dos funciones de pérdidas anteriores supervisan el procedimiento de aprendizaje con la siguiente formula:

$$loss = \lambda loss_1 + loss_2 \tag{3.7}$$

en la que $\lambda = 3$ es un factor que equilibra ambas pérdidas, estimado empíricamente.

Capítulo 4

Resultados

4.1 Introducción

En este capítulo se describirán todas las pruebas realizadas a lo largo del proyecto junto con los resultados obtenidos. La organización de este capítulo es la siguiente:

1. El capítulo comienza con la descripción del entorno experimental utilizado e indicando las bases de datos que se han usado en los experimentos de éste capítulo.

2. En el apartado 4.2.2 se procede a verificar la puesta en marcha del sistema A2J de estimación de la posición de las articulaciones del cuerpo en las condiciones del sistema de partida. Es decir, se prueba el sistema utilizando el modelo ofrecido por ellos y la base de datos ITOP.

3. Una vez se ha verificado el punto anterior, se pasa a la subsección 4.2.3, en la que se realiza el testeo del sistema utilizando una nueva base de datos, lo que supondrá que habrá que adaptar los datos de esta nueva base de datos a las entradas del sistema A2J.

4. En la sección 4.3 se van a describir todos los experimentos que se han abordado utilizando la base de datos ETRI. Éstos, a grandes rasgos, se han dividido en 3 entrenamientos generales sobre los que se aplicarán modificaciones en los parámetros de la red.

    - En el primero sólo se utiliza las grabaciones de 50 personas de edad avanzada y se dividirán las personas en un ratio de 80 %/10 %/10 % para *train*, *validation* y *test* respectivamente.
    - En el segundo se usan las grabaciones de 100 personas, tanto de edad avanzada como jóvenes, las cuales se dividirán en un ratio de 80 %/10 %/10 % para *train*, *validation* y *test* respectivamente.
    - En el último se utilizan las mismas grabaciones que en el anterior punto, pero esta vez se dividen en un ratio de 90 %/10 % para *train* + *validation* y *test* respectivamente. El 90 % de entrenamiento se dividirá en un 90 % de los *frames* para *train* y el 10 % restante parra validación.

5. En la sección 4.5 se explica por qué se ha tenido que dar "un paso atrás" para volver a utilizar la base de datos ITOP, en base a la evidencia de que algo en el código de entrenamiento no estaba bien.

6. Finalmente, en la sección 4.6 se exponen los resultados y los algoritmos generados para obtener la estimación de ciertos parámetros de destreza y funcionalidad.

4.2 Entorno experimental

En esta sección cabe destacar que todas las pruebas mencionadas a lo largo del capítulo se han realizado en un entorno virtual python creado en un ordenador con sistema operativo Ubuntu 20.04.4 LTS y GPU NVIDIA GeForce RTX 3090. El entorno virtual permite mantener separadas las dependencias requeridas por diferentes proyectos.

4.2.1 Bases de datos utilizadas

En esta sección se van a mencionar las bases de datos que se han utilizado al realizar las pruebas y a lo largo del capítulo se irá comentando qué resultados pertenecen a cada una de ellas. De entre las bases de datos expuestas en la sección 2.5, las elegidas han sido:

1. **ITOP**. Se ha escogido ésta de entre las 4 que utilizan en el estudio A2J, porque es la base de datos que contiene imágenes de profundidad y las posiciones 3D de las articulaciones del cuerpo completo.
2. **UTKinect-Action3D**. Se ha seleccionado ya que es de las que más actividades de la vida diaria cubre y porque el tamaño de las imágenes de profundidad coincide con el de las imágenes de la base de datos ITOP.
3. **ETRI-Activity3D**. Ésta se ha elegido ya que en las grabaciones, además de jóvenes, también participan personas de edad avanzada (mayores de 60 años) en tres salas de una casa, lo cual se asemeja a las condiciones que se van a aplicar al proyecto EYEFUL, encontrarse en un entorno "no controlado" y que éste sea una vivienda para poder observar como se desenvuelven las personas al realizar las ADLs.

4.2.2 Verificación de la puesta en marcha del sistema en las condiciones del sistema de partida

El primer paso es intentar replicar los mismos resultados que han conseguido en la publicación [33], utilizando una de las bases de datos que usan ellos. Entre las dos bases de datos de la posición del humano que utilizan ellos (K2HPD e ITOP), se ha escogido ITOP [88] ya que la base de datos contiene imágenes de profundidad y las posiciones 3D de las articulaciones, mientras que en K2HPD, están disponibles las imágenes de profundidad y las posiciones de las articulaciones son en 2D. Los pasos que se han seguido para imitar los resultados son los que ellos han propuesto en su repositorio oficial [89], que consisten en: una vez descargados los datos, ejecutar el código `data_preprosess.py` para convertir las imágenes de profundidad en archivos `.mat` y después ejecutar el código `itop_side.py` con las imágenes de test convertidas, obteniéndose los resultados de la tabla 4.1.

Al comparar los valores de la tabla 4.1 con los resultados proporcionados en la publicación, se puede comprobar que son prácticamente idénticos, por lo que se puede asegurar que se ha podido replicar el sistema correctamente. Se puede verificar con la figura 4.1, que además muestra la precisión (mAP) de otros métodos de estimación de la posición de las articulaciones del cuerpo humano. La métrica de evaluación mAP la aplican con una norma de 10 cm, que quiere decir que sólo se considerarán como verdadero positivo a las articulaciones que se encuentren dentro de un área menor de $1\,cm^2$ alrededor del *ground truth*. La fórmula utilizada es la mostrada en la ecuación 4.1, donde TP son los positivos verdaderos, y GT (*ground truth*) es el número total de articulaciones etiquetadas por la dimensión y el número de *frames*.

$$mAP = \frac{\sum TP_i}{GT} \tag{4.1}$$

Tabla 4.1: Resultados *test* A2J con la base de datos ITOP. La D hace referencia a derecha y la I a izquierda.

Número de la articulación	**Articulación**	***Accuracy* (mAP)**	**MSE** (m^2)
0	Cabeza	0,985	$6{,}966x10^{-8}$
1	Cuello	0,992	$8{,}733x10^{-8}$
2	Hombro D	0,965	$2{,}067x10^{-7}$
3	Hombro I	0,960	$1{,}939x10^{-7}$
4	Codo D	0,794	$1{,}307x10^{-6}$
5	Codo I	0,785	$1{,}956x10^{-6}$
6	Mano D	0,696	$1{,}897x10^{-6}$
7	Mano I	0,671	$9{,}108x10^{-7}$
8	Tronco	0,985	$1{,}467x10^{-6}$
9	Cadera D	0,919	$8{,}151x10^{-6}$
10	Cadera I	0,898	$5{,}987x10^{-6}$
11	Rodilla D	0,913	$6{,}077x10^{-6}$
12	Rodilla I	0,902	$9{,}614x10^{-6}$
13	Pie D	0,874	$1{,}019x10^{-5}$
14	Pie I	0,864	$7{,}980x10^{-6}$
Media		0,880	$3{,}737x10^{-6}$

$$TP_i = \sum_{i=0}^{N}\sum_{j=0}^{K}\left(output_j^i - label_j^i\right)^2 = \begin{cases} 1, \text{ cuando } TP < 0{,}1^2 \\ \\ 0, \text{ en cualquier otro caso} \end{cases} \tag{4.2}$$

También se ha añadido la métrica *MSE*, la cual se encarga de medir el promedio de los errores al cuadrado. Su ecuación es la 4.3.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(output_i - label_i\right)^2 \tag{4.3}$$

	mAP (front-view)							
Method	RF [32]	RTW [48]	IEF [4]	VI [19]	CMB [39]	REN-9x6x6 [17]	V2V* [25]	A2J (Ours)
Head	63.8	97.8	96.2	98.1	97.7	98.7	98.29	98.54
Neck	86.4	95.8	85.2	97.5	98.5	99.4	99.07	99.20
Shoulders	83.3	94.1	77.2	96.5	75.9	96.1	97.18	96.23
Elbows	73.2	77.9	45.4	73.3	62.7	74.7	80.42	78.92
Hands	51.3	70.5	30.9	68.7	84.4	55.2	67.26	68.35
Torso	65.0	93.8	84.7	85.6	96.0	98.7	98.73	98.52
Hips	50.8	90.3	83.5	72.0	87.9	91.8	93.23	90.85
Knees	65.7	68.8	81.8	69.0	84.4	89.0	91.80	90.75
Feet	61.3	68.4	80.9	60.8	83.8	81.1	87.60	86.91
mean	65.8	80.5	71.0	77.4	83.3	84.9	**88.74**	88.0

Figura 4.1: Comparación de las estimaciones con los métodos más avanzados [33].

4.2.3 Experimento sobre una nueva base de datos: UTKinect-Action3D

Una vez se ha comprobado que el sistema funciona correctamente, el siguiente paso es probar la red con la base de datos UTKinect-Action3D [47]. Para ello, primero hay que adaptar los datos de esta base de datos a las entradas del sistema A2J. Esto se lleva acabo realizando las acciones que se describen en los siguientes apartados.

4.2.3.1 Generación del archivo con los *keypoints*

Una de las primeras cosas que hay que hacer es generar un archivo `.mat` con las posiciones 3D (x, y, depth), referenciadas al eje de la cámara, de las 15 articulaciones necesarias (*keypoints*). Observando los valores de ITOP, se puede comprobar que tienen almacenados los datos en metros y en las coordenadas del plano imagen. Las posiciones de las articulaciones en UTK vienen guardadas en un archivo de texto por cada persona, y debido a que los datos utilizados en la prueba contenían algunos frames con múltiples informaciones del esqueleto con ubicaciones de las articulaciones ligeramente diferentes, se han eliminado los valores de *frames* repetidos, previamente al guardado en un `.mat`. Adicionalmente, se ha tenido que reordenar las posiciones de las articulaciones para que coincidan con el orden establecido en A2J, ya que la base de datos de UTK se ha grabado con una KinectV1 y ésta genera el esqueleto con las posiciones de 20 articulaciones enumeradas de forma diferente al esqueleto generado en ITOP, tal y como se muestra en la figura 4.2.

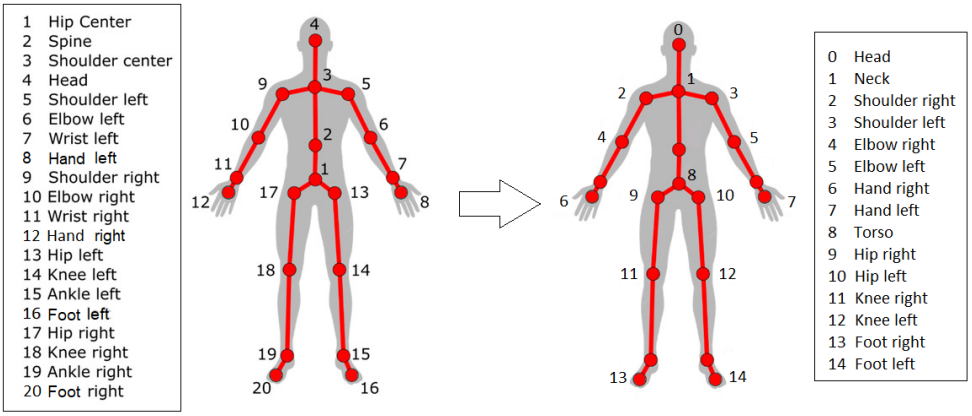


Figura 4.2: Posiciones de las articulaciones en la base de datos UTK (izquierda) y en la red A2J (derecha).

4.2.3.2 Generación del archivo con las imágenes de profundidad

En este caso hay que convertir los mapas de profundidad almacenados en ficheros `.xml` al formato Hierarchical Data Format (HDF) para posteriormente generar los `.mat` con el programa proporcionado por A2J. En esta conversión hay que tener en cuenta en qué unidades han almacenado las distancias, siendo estas habitualmente metros o milímetros, pero en este caso, al observar los valores, estos van de los 15000 a los 32000 (excluyendo los ceros que se dan cuando un objeto está muy cerca o más lejos de la distancia máxima que es capaz de detectar el sensor, que en este caso son 4 metros), por lo que al almacenarlos en el fichero `.h5` se dividen las distancias por un factor 10000 para que la unidad de medida sea metros.

4.2.3.3 Generación del *bounding box*

En este paso es necesario generar un *bounding box*, que es un recuadro delimitador que servirá para recortar la zona del mapa de profundidad en la que se encuentra la persona. Para ello se han evaluado dos aproximaciones distintas que se describen a continuación.

4.2.3.3.1 Utilizando un sistema de detección de objetos

Este tipo de sistemas consisten en detectar en dónde se encuentra un objeto determinado en una imagen, independientemente del tamaño de este, del entorno en el que se encuentre, de la posición en la que está en la propia imagen y de si está parcialmente cubierto o no por otro objeto.

De entre todas las opciones, los más implementados según [90] son YOLO, en especial la versión 3 [91]; RetinaNet [92]; y Faster R-CNN [93]. En vista de que YOLOv3 está a la par con RetinaNet en precisión (mAP) y es bastante más rápido que este y que Faster R-CNN (ver figura 4.3), en una implementación en tiempo real, como la que se busca como futuro de este proyecto, es preferible YOLO aunque no sea igual de preciso que los demás.

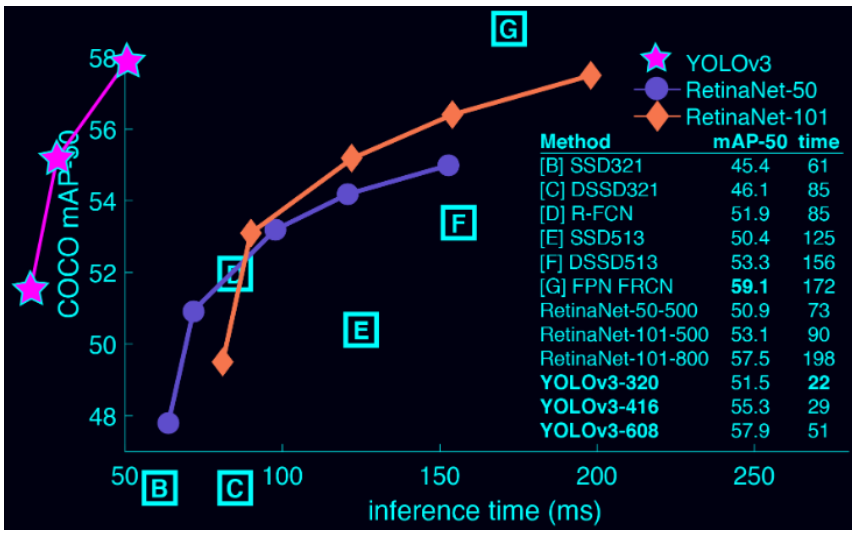


Figura 4.3: Comparación entre sistemas de detección de objetos en tiempo real [94].

YOLO es un algoritmo que es capaz de detectar numerosos objetos en una imagen, en concreto, en la implementación realizada puede reconocer hasta 80 objetos distintos. La velocidad de detección que alcanza es fundamentalmente debido a que el algoritmo sólo emplea una única *forward propagation* a través de la CNN para detectar los objetos, es decir, el algoritmo sólo se ejecuta una vez. La CNN que emplea la utiliza para predecir simultáneamente varias probabilidades de clase y cuadros delimitadores.

Primero divide las imágenes con un *grid* de tamaño fijo e intenta detectar objetos en cada una de las celdas a partir de *anchor boxes* de diferentes tamaños (ver figura 4.4). Seguidamente hace uso de la métrica Intersection over Union (IoU) para evitar una detección múltiple del mismo objeto, obteniendo con esta métrica un porcentaje de acierto del área predicha dónde se encuentra el objeto frente el *bounding-box* real. Conjuntamente aplica NMS [95], que consiste en seleccionar la caja delimitadora que detecta al objeto con mayor precisión de entre todas las cajas superpuestas, descartando las que están por debajo de un límite de probabilidad establecido. Con las cajas restantes, se eligen sucesivamente las dela probabilidad más alta, se establecen como predicción y se descarta cualquier caja restante con una $IoU \geq 0{,}5$ con la caja generada en el paso anterior.

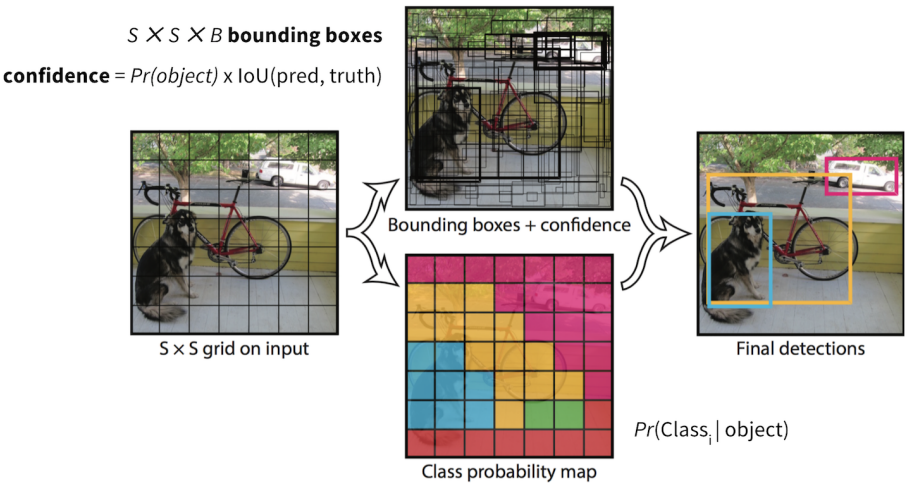


Figura 4.4: Esquema funcionamiento YOLO [96].

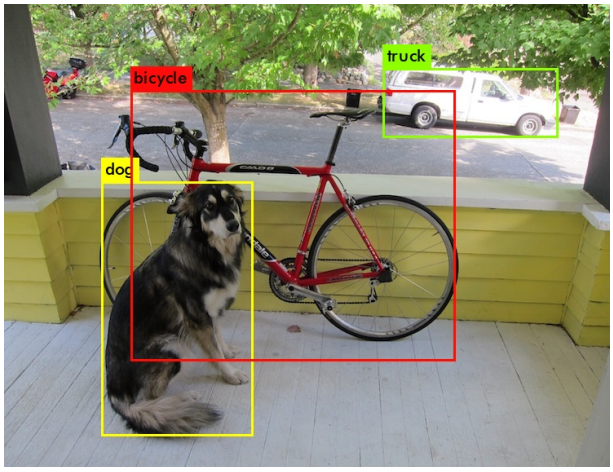


Figura 4.5: Ejemplo resultado YOLO [94].

Resultados obtenidos

YOLO se ha aplicado haciendo uso de la red Darknet con un modelo preentrenado en la base de datos COCO [97], la cual contiene más de $2x10^5$ imágenes etiquetadas de 80 objetos distintos.

Se hizo una primera prueba para observar que es lo qué detectaba del entorno dentro de las imágenes RGB de la base de datos de UTK (ver figura 4.6), comprobando que sí detecta a las personas, pero también a otros objetos que se encuentran en la sala mostrada en la imagen, como la silla. Suceso que no nos interesa que ocurra, por el momento sólo nos concierne que detecte a los humanos. Como modificar la configuración de la red no es tan sencillo, se optó por generar un único recuadro delimitador con Python, utilizando OpenCV [98–100]. Simplemente hay que leer del fichero `coco.names` las clases que hay, y cargar el archivo de configuración `yolov3.cfg` y el modelo preentrenado `yolov3.weights`. Después se ejecuta la red y se recogen las predicciones de la capa de salida, para obtener el *bounding box* si el *confidence score*[1] es mayor de 0,5, y seguidamente se realiza una NMS para eliminar las cajas superpuestas redundantes. Finalmente se comprueba que la clase detectada es persona y dibujamos el recuadro en la imagen, obteniendo el resultado de la figura 4.7.

[1] Es la probabilidad de que la imagen sea detectada correctamente por el algoritmo.

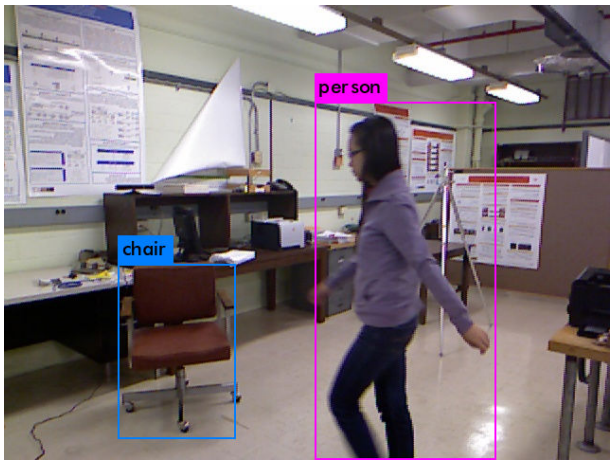


Figura 4.6: Detección de YOLO en imágenes de la base de datos UTK.

Figura 4.7: Detección de YOLO con OpenCV en imágenes de la base de datos UTK.

Al observar las salidas de la red, se pudo comprobar que no en todas las imágenes detectaba a la persona, por lo que se probó con distintos tamaños de anchura y altura de entrada y con umbrales de NMS diferentes, sin ser muy altos para evitar detecciones redundantes, observando el comportamiento de la tabla 4.2.

Tabla 4.2: Número de imágenes sin detección en función del tamaño de la entrada y el umbral NMS.

Tamaño de anchura y altura de entrada (píxeles)	416	608	608	416
Umbral NMS	0,1	0,1	0,2	0,2
Nº de imágenes sin detección de la persona	24	7	3	0

Una vez se han fijado los parámetros anteriores a un tamaño de 416×416 y un umbral de 0,1, hay que comprobar si el *bounding box* limita a la persona entera en la imagen de profundidad. Para ello se ha utilizado un editor de imágenes y se ha sobrepuesto una sobre la otra, obteniéndose la figura 4.8.

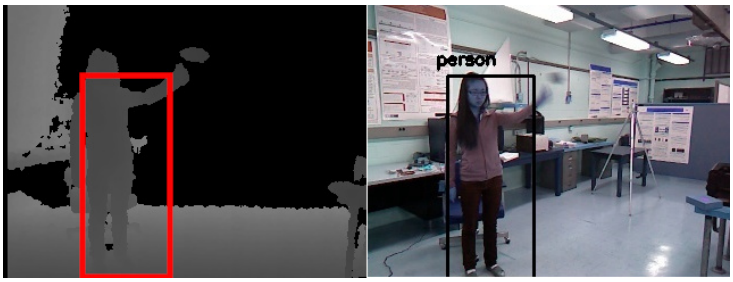


Figura 4.8: Detección de YOLO en imágenes de la base de datos UTK.

El anterior paso nos recuerda que como las imágenes de profundidad tienen una resolución mayor (320×240) que las RGB (640×480), es necesario redimensionar la imagen antes de la detección con YOLO para que así el *bounding box* generado a partir de la imagen RGB capte a la persona entera en la imagen de profundidad. Para ello se ha aplicado la función `cv2.resize` con el método del vecino más cercano. Añadir, que para que este proceso sea correcto, el tamaño de las imágenes de profundidad debe mantener la proporción adecuada. A continuación, en el editor de imágenes se superpone la imagen de profundidad a la de RGB y bajando la opacidad al mínimo se alinean para poder calcular cuántas filas y columnas de información hay que recortar de las imágenes de profundidad (ver figura 4.9).

Figura 4.9: Recorte de información de la imagen de profundidad.

Un punto negativo que ya se ha podido apreciar en la figura 4.9 es que a pesar de que el *bounding box* rodea a la persona, hay muchas ocasiones en las que alguna extremidad del cuerpo queda fuera del recuadro, y en caso de ampliar el cuadro en todas las direcciones un cierto número de píxeles, no funcionaría porque este no siempre está centrado y podría seguir quedando alguna extremidad fuera, y además hay ocasiones en las que el tamaño de la extensión no sería suficiente, como en el caso de la figura 4.10 y en otros momentos podría ser excesiva, lo que aumentaría la influencia del fondo en la detección, complicando tanto la tarea de entrenamiento como la propia detección.

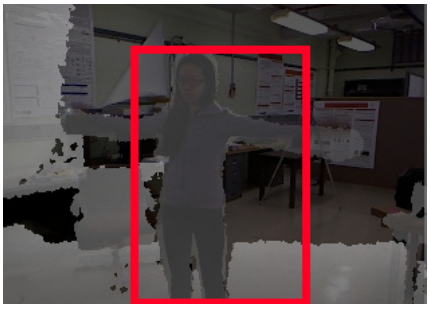


Figura 4.10: Detección de YOLO en imágenes de la base de datos UTK.

Debido a lo comentado, en un futuro habría que estudiar la posibilidad de utilizar un sistema de detección de objetos con mayor precisión a la hora de generar el recuadro delimitador del objeto a detectar, pero por el momento se va a utilizar el método que se va a comentar seguidamente, a pesar de que no tiene validez en un sistema en tiempo real, ya que no se tendría acceso a las coordenadas verdaderas de las articulaciones de las personas.

Utilizando las posiciones de las articulaciones.

Finalmente, esta es la opción que se ha utilizado para generar el *bounding box* que se aplicará a la red A2J. Consiste en crear el recuadro delimitador del área que ocupa una persona en una imagen de profundidad a partir de la información de las posiciones de las articulaciones. De entre las 15 articulaciones posibles se han escogido las dos que en el plano imagen tengan el menor valor y las dos mayores, limitando con los primeros la coordenada (u,v) del vértice superior izquierdo del recuadro y con los dos últimos la coordenada (u,v) del vértice inferior derecho. A partir de estos dos vértices se obtienen los otros dos restantes y se amplia el tamaño 50 píxeles en el lado derecho e inferior y se disminuye 50 píxeles en el lateral izquierdo y superior, para dar algo más de margen. Teniendo en cuenta que como máximo se podrá ampliar hasta el tamaño de la imagen y disminuir hasta 0.

Figura 4.11: Comparación entre recuadros delimitadores.

En la figura 4.11 se muestra una comparación entre el *bounding box* que utiliza A2J en color rojo y el que se ha generado en color negro, sobre las imágenes de la base de datos de ITOP. Respecto al recuadro en la base de datos de UTK, el generado a partir de las articulaciones es el de color verde y el de negro es el originado por YOLO, visible en la figura 4.12. Pudiendo observar una gran mejoría entre uno y otro.

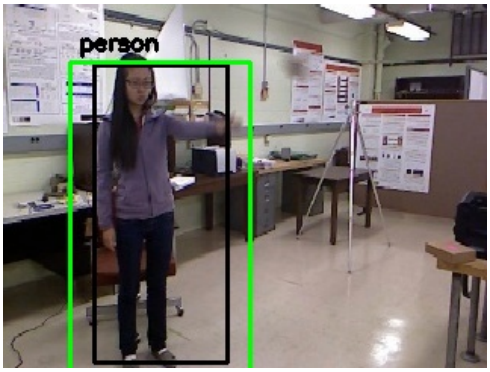


Figura 4.12: Comparación entre recuadros delimitadores.

4.2.3.4 Generación de la media y la desviación estándar

El último paso que quedaría para poner a prueba el funcionamiento de la red A2J con otra base de datos sería generar la media y desviación estándar de todos los datos de profundidad que se encuentran dentro del *bounding box* de todas las imágenes utilizadas en el entrenamiento. Antes de generar los resultados con la base de datos de UTK, se intentan calcular los de la base de datos ITOP. Cabe destacar que cuando se calcula la media y la desviación estándar, los valores de profundidad superiores e inferiores al valor de profundidad del punto central del recuadro más/menos un factor, se igualan al valor de la profundidad del punto central. De esta forma no se tienen en cuenta los valores nulos debido a objetos situados o muy cerca o más lejos de la distancia máxima en la que el sensor es capaz de detectar objetos. Este factor A2J lo considera como 200 mm en el caso de la estimación de las articulaciones de las manos, y en el caso del cuerpo completo no indican nada, por lo cual se ha estimado a $2\,m$ con ensayo y error a partir del conjunto de entrenamiento de la base de datos de ITOP, que es de la que se tiene el valor de la media y desviación estándar calculados por A2J, tabla 4.3.

Tabla 4.3: Media y desviación estándar de la base de datos ITOP.

	Media (m)	Desviación estándar (m)
Valor obtenido por A2J	3,367	0,456
Valor calculado	3,354	0,430

A continuación, se ha realizado el *test* de estimación de las articulaciones con la base de datos ITOP, pero utilizando los *bounding boxes* y media y desviación estándar calculados, para poder comparar los resultados con los originales. Ver tabla 4.4.

Al comparar las tablas 4.1 y 4.4, se puede comprobar que el mAP de todas las articulaciones sólo ha bajado en un 14,4 %, y este sigue siendo razonablemente alto, con un valor de precisión del 73,6 %. A su vez, la media del error ha subido ligeramente, en $0{,}8x10^{-6}\,m^2$. Por lo que se podría afirmar que el método utilizado de obtención de los datos de entrada al sistema A2J es correcto.

A continuación se muestran los valores calculados de la media y desviación estándar de profundidad del conjunto de datos de *test* de UTK (tabla 4.5) y la precisión y el error en la estimación de las articulaciones (tabla 4.6).

Tabla 4.5: Media y desviación estándar de la base de datos UTK.

	Media (m)	Desviación estándar (m)
Valor calculado	2,443	0,402

Tabla 4.4: Resultados *test* con la base de datos ITOP y media, desviación estándar y*bounding box* propio. La D hace referencia a derecha y la I a izquierda.

Número de la articulación	**Articulación**	***Accuracy* (mAP)**	**MSE** (m^2)
0	Cabeza	0,910	$23{,}254x10^{-8}$
1	Cuello	0,952	$40{,}751^{-8}$
2	Hombro D	0,792	$8{,}545x10^{-7}$
3	Hombro I	0,830	$8{,}523x10^{-7}$
4	Codo D	0,6084	$3{,}095x10^{-6}$
5	Codo I	0,611	$2{,}074x10^{-6}$
6	Mano D	0,490	$1{,}572x10^{-6}$
7	Mano I	0,509	$17{,}929x10^{-7}$
8	Tronco	0,935	$2{,}295x10^{-6}$
9	Cadera D	0,831	$7{,}129x10^{-6}$
10	Cadera I	0,787	$7{,}468x10^{-6}$
11	Rodilla D	0,848	$7{,}314x10^{-6}$
12	Rodilla I	0,819	$12{,}519x10^{-6}$
13	Pie D	0,551	$0{,}950x10^{-5}$
14	Pie I	0,572	$10{,}976x10^{-6}$
Media		0,736	$4{,}539x10^{-6}$

Tabla 4.6: Resultados *test* con la base de datos UTK. La D hace referencia a derecha y la I a izquierda.

Número de la articulación	**Articulación**	***Accuracy* (mAP)**	**MSE** (m^2)
0	Cabeza	0,011	0,052
1	Cuello	0,083	0,039
2	Hombro D	0,0	0,090
3	Hombro I	0,0	0,066
4	Codo D	0,002	0,112
5	Codo I	0,0	0,091
6	Mano D	0,007	0,146
7	Mano I	0,002	0,120
8	Tronco	0,0	0,064
9	Cadera D	0,0	0,064
10	Cadera I	0,0	0,052
11	Rodilla D	0,0	0,071
12	Rodilla I	0,0	0,101
13	Pie D	0,0	0,086
14	Pie I	0,0	0,103
Media		0,007	0,084

Los resultados obtenidos son en cierto modo los que se esperaban, ya que la red A2J no ha tenido imágenes parecidas a las de la base de datos de UTK para entrenar, por lo que no es capaz de estimar correctamente las posiciones de las articulaciones. Aún así, se han representado las imágenes de profundidad que se introducen a la red junto con el *ground truth*, justo después del preprocesado, para verificar que los malos resultados no tuvieran que ver con datos erróneos a la entrada. También se ha representado el esqueleto en 3D a partir de las articulaciones reales y las conseguidas a la salida de la red.

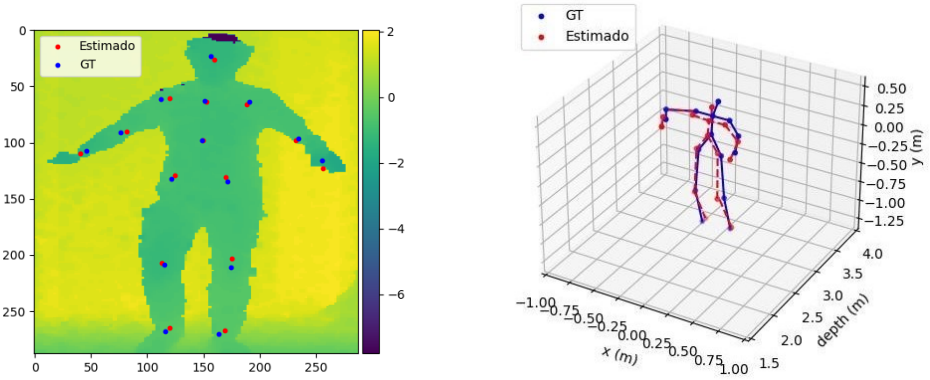


Figura 4.13: Imagen de profundidad preprocesada y el esqueleto estimado. Base de datos ITOP.

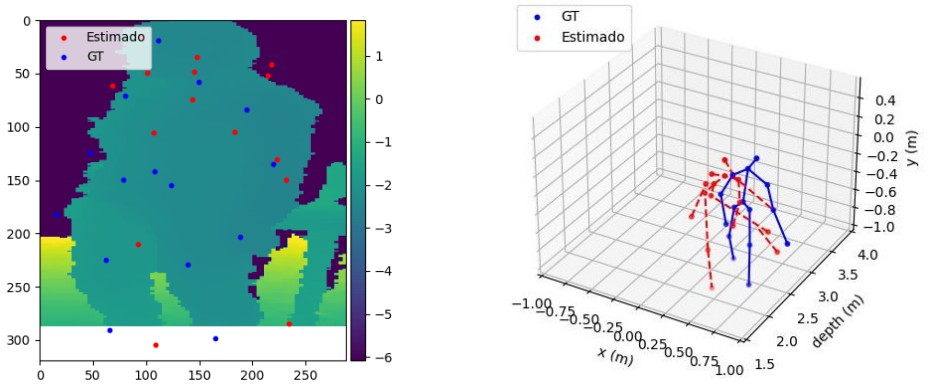


Figura 4.14: Imagen de profundidad preprocesada y el esqueleto estimado. Base de datos UTK.

4.3 Resultados experimentales ETRI

En esta sección se van a mostrar todos los experimentos que se han abordado con la base de datos de ETRI. De entre todas las actividades que contiene, se ha hecho una preselección de las principales y/o más relevantes y aplicables en las pruebas de AMPS indicando también el ID que le corresponde, las cuales son:

- Cepillado de dientes – ID 10
- Ponerse/quitarse zapatos – ID 20
  - Sentarse/Levantarse – ID 54
- Aspirar el suelo – ID 23
- Poner la mesa:

  - Mirar alrededor buscando algo – ID 30
  - Meter/Sacar comida de la nevera – ID 5
  - Limpiar la mesa – ID 25
- Preparar cereales y bebida:
  - Comer con tenedor – ID 1
  - Verter agua en un vaso – ID 2
  - Beber agua – ID 4
- Caerse al suelo – ID 53

Como una primera toma de contacto con el esquema de entrenamiento, se ha realizado una serie de pruebas únicamente con una de las acciones, siendo la escogida la actividad de meter y sacar comida de la nevera. Y únicamente se han utilizado las imágenes grabadas desde un punto de vista y desde una única altura en la posición de la cámara.

La forma de tratar los datos antes de introducirlos en la red es la misma que la desarrollada en el apartado 4.2.2, con excepción del factor por el que se dividen los datos de profundidad antes de almacenarlos en un fichero `.h5`. En este caso la unidad de medida son los metros, por lo que no es necesario dividir los datos por ningún factor.

Antes de entrenar, se realizó un *test* con el modelo entrenado en la base de datos de ITOP, obteniéndose los siguientes valores de mAP y MSE, tabla 4.7. En este caso, al igual que al utilizar la base de datos UTK, no se han conseguido unos buenos resultados, lo cual es normal, ya que el modelo utilizado está entrenado en otra base de datos (ITOP) con unas condiciones distintas a las de la base de datos ETRI.

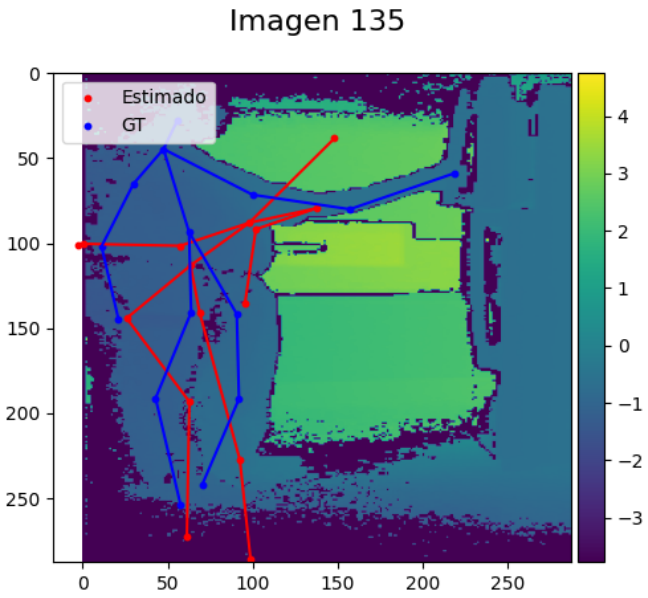


Figura 4.15: Imagen de profundidad preprocesada y el esqueleto estimado. Base de datos ETRI.

En la figura 4.15 se ha representado las articulaciones estimadas junto con las etiquetadas, observándose el error en la estimación.

Tabla 4.7: Resultados *test* con la base de datos ITOP. La D hace referencia a derecha y la I a izquierda.

Número de la articulación	**Articulación**	***Accuracy* (mAP)**	**MSE** (m^2)
0	Cabeza	0,0	0,619
1	Cuello	0,0	0,545
2	Hombro D	0,0	0,419
3	Hombro I	0,0	0,672
4	Codo D	0,0	0,328
5	Codo I	0,0	0,562
6	Mano D	0,0	0,416
7	Mano I	0,0	0,494
8	Tronco	0,0	0,490
9	Cadera D	0,0	0,508
10	Cadera I	0,0	0,560
11	Rodilla D	0,0	0,552
12	Rodilla I	0,0	0,539
13	Pie D	0,0	0,684
14	Pie I	0,0	0,721
Media		0,0	0,541

4.3.1 Entrenamiento con una acción realizada por 50 personas

En este primer entrenamiento se ha utilizado las imágenes de profundidad grabadas con una cámara en una única posición de las 50 personas mayores de 60 años participantes realizando la actividad mencionada anteriormente. Este conjunto de imágenes se ha dividido aleatoriamente en un 80 % para el conjunto de entrenamiento, un 10 % para el de validación y otro 10 % para el de *test*. Por tanto, el número de imágenes del conjunto de entrenamiento son 4516, del de validación 479 y del de *test* 512.

Respecto a los parámetros de la red: el tamaño del *batch* se mantiene a 64; el *learning rate* también se deja a 0,00035; y la caída de los pesos a $1x10^{-4}$ (entre 0,00035/3 y 0,00035/3), que cumple con una asignación habitual que dice que suele funcionar bien tenga un valor de entre la $tasa\_de\_aprendizaje/3$ y $tasa\_de\_aprendizaje/4$ [101].

Los valores que se utilizan para aumentar el número de datos en función del método son:

- Al realizar un escalado se asigna un valor aleatorio menor que 1 y al que se le suma 0,5.
- Recorte o ampliación del *bounding box*. Se recorta o amplia un valor de píxeles entero aleatorio entre -5 y 5 a cada lateral del bounding box.
- Rotación. Se rota la imagen un valor aleatorio entre -180^o y 180^o.

- Finalmente, también se aplica un ruido gaussiano con una probabilidad del 50 %.

El código abierto de entrenamiento que ofrecen es para estimar las poses de la mano, por lo que se ha tenido que adaptar a nuestra aplicación, la estimación de la pose del cuerpo humano. Además, se ha añadido un *checkpoint* para poder ir guardando únicamente el modelo de la época que genera menores pérdidas. En total se ha puesto un máximo de 300 épocas pero con una paciencia de 35 épocas, es decir, sí las pérdidas no disminuyen durante 35 épocas seguidas, se termina el entrenamiento.

A continuación, se van a listar los distintos entrenamientos realizados junto con sus resultados:

4.3.1.1 Sin aumento de datos

Este experimento consisten en entrenar a la red sin aplicar ningún método de aumento de datos. Como se puede observar en las figuras 4.16, 4.17 y 4.18, los resultados no son nada buenos. El error MSE en validación es excesivo (aproximadamente $15\,m^2$) y es raro que el error de validación sea mucho menor que el de *train*, el cual es exageradamente grande. En base a estos valores, es normal que la precisión sea inexistente. Respecto a la curva de pérdidas, no es bueno que la curva de validación esté tan alejada de la de entrenamiento.

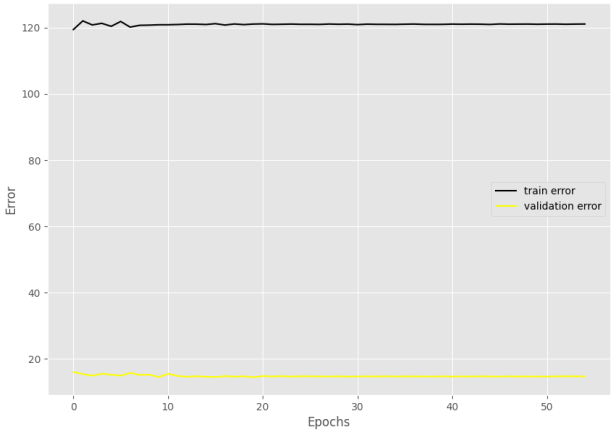


Figura 4.16: Error MSE (m^2) durante el entrenamiento con 50 personas y sin aumento de datos.

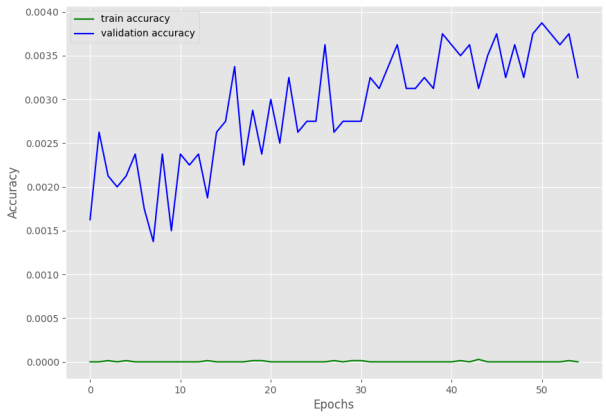


Figura 4.17: Precisión (mAP) durante el entrenamiento con 50 personas y sin aumento de datos.

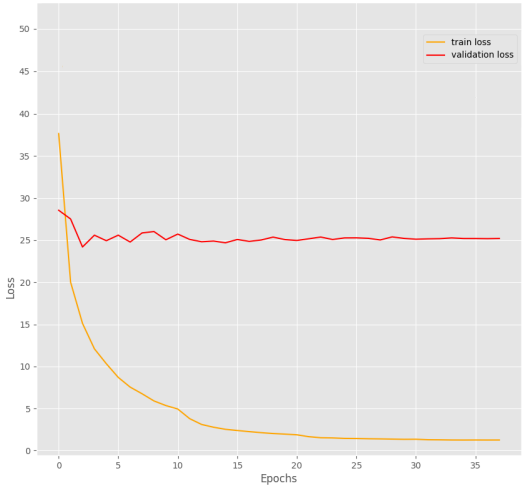


Figura 4.18: Pérdidas durante el entrenamiento con 50 personas y sin aumento de datos.

4.3.2 Con aumento de datos

Este experimento consiste en aplicar todos los métodos de aumento de datos a todo el conjunto de entrenamiento, de esta forma se puede ver la diferencia entre utilizar aumento de datos y no utilizarlo. En comparación a los resultados anteriores, el error MSE a disminuido ligeramente, aunque sigue siendo muy grande (figura 4.19), por ello la precisión se mantiene nula (figura 4.20), y las pérdidas de validación se mantienen iguales (figura 4.21).

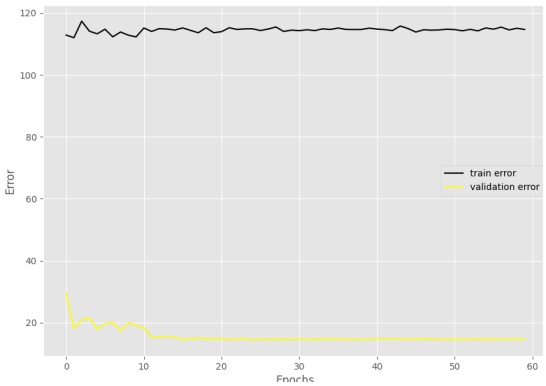


Figura 4.19: Error MSE (m^2) durante el entrenamiento con 50 personas y aumento de datos.

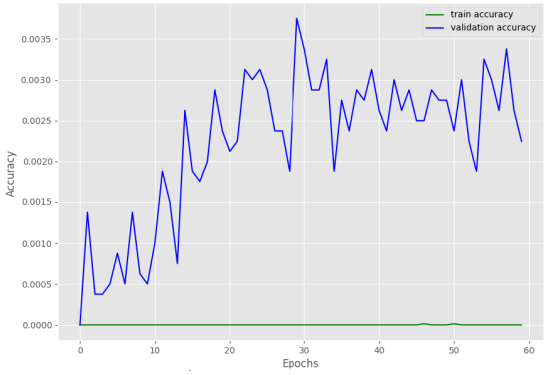


Figura 4.20: Precisión (mAP) durante el entrenamiento con 50 personas y aumento de datos.

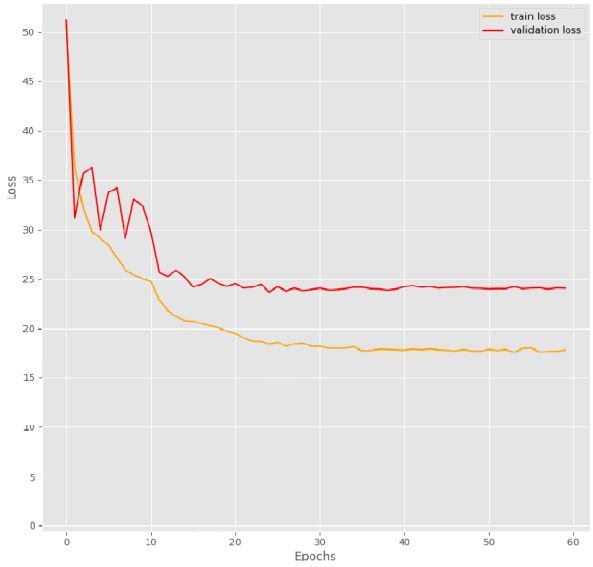


Figura 4.21: Pérdidas durante el entrenamiento con 50 personas y aumento de datos.

Tabla 4.8: Comparación de los entrenamientos con 50 personas.

		MSE (m^2)	*Accuracy* (mAP)	Pérdidas
Sin aumento de datos	Entrenamiento	122,0	0,0	0,51
	Validación	14,537	0,0037	24,489
Con aumento de datos	Entrenamiento	114, 359	0,0	17, 977
	Validación	15, 214	0,0029	24, 045

En cualquiera de las dos pruebas se obtienen malos resultados, ya que tanto las pérdidas como el error es muy alto, y la precisión muy baja. Esto se podría deber a que el número de imágenes para entrenar es bastante reducido, así que en el siguiente entrenamiento se aumentará el número de datos.

4.3.3 Entrenamiento con 100 personas con distribución de los datos en un ratio de 80/10/10

Como segundo entrenamiento también se han utilizado las imágenes de profundidad grabadas con una cámara en una única posición, pero esta vez utilizando las grabaciones de todos los sujetos participantes, 100 personas (50 personas de edad avanzada y 50 jóvenes en su veintena), realizando la misma actividad anterior. Este conjunto de imágenes se ha dividido aleatoriamente en un 80 % para el conjunto de entrenamiento, un 10 % para el de validación y otro 10 % para el de *test*. Ahora bien, se ha forzado que en cada conjunto de datos hubiese la mitad de jóvenes que de mayores. Por consiguiente, el número de imágenes del conjunto de entrenamiento son 8580, del de validación 1083 y del de *test* 941.

4.3.3.1 Sin aumento de datos

En este caso sólo se va a realizar un experimento y será sin aumentar los datos, simplemente para verificar si con el aumento de imágenes mejora el entrenamiento. En la tabla 4.9 se han agrupado los valores de

pérdidas, error y precisión al entrenar con 100 personas y distribuyendo los datos de la forma indicada anteriormente.

Tabla 4.9: resultados entrenamiento con 100 personas sin aumento de datos.

		MSE (m^2)	*Accuracy* (mAP)	Pérdidas
Sin aumento de datos	Entrenamiento	136,596	0,0	1,694
	Validación	11,915	0,0037	21,529

En este entrenamiento se ha conseguido bajar bastante las pérdidas de entrenamiento con respecto al anterior, pero las pérdidas de validación no siguen la curva de las de entrenamiento, lo cual indica que no está aprendiendo bien, además de que el error el muy grande, no puede ser que haya un error de $12\,m^2$ si en las imágenes de la base de datos la habitación tiene un tamaño menor de $4 \times 4\,m^2$. Con esto ya nos podemos hacer una idea de que el problema se encuentra en el entrenamiento y no en el conjunto de datos que se utiliza, aún así, se van a reducir las posibles condiciones de error modificando un poco el conjunto de entrenamiento para que a la red le sea más fácil aprender.

4.3.4 Entrenamiento 100 personas con distribución de los datos en un 90/10

En los entrenamientos anteriores se han utilizado a personas distintas en cada uno de los conjuntos (validación, entrenamiento y *test*), lo cual le complica mucho el aprendizaje de la red. Una forma de ayudar al entrenamiento es utilizando en el conjunto de validación a una persona que la red haya visto ya en el entrenamiento, aunque las de *test* son personas a las que no ha visto en ningún momento. Es por ello que se ha modificado la distribución del conjunto de datos. Ahora se ha realizado una primera separación en un 90 % de las personas para el conjunto de entrenamiento + validación y el 10 % sobrante para el conjunto de *test*. A su vez, los *frames* correspondientes al 90 % se han dividido en un 10 % para la validación y el restante para el entrenamiento. Una vez más, en la primera división realizada se ha forzado que cada uno de los conjuntos tenga la mitad de personas jóvenes y de adultos mayores de 60. A causa del reparto efectuado, el número de *frames* utilizados en el conjunto de entrenamiento son 8651, en validación 962 y en *test* 991.

4.3.4.1 Sin aumento de datos

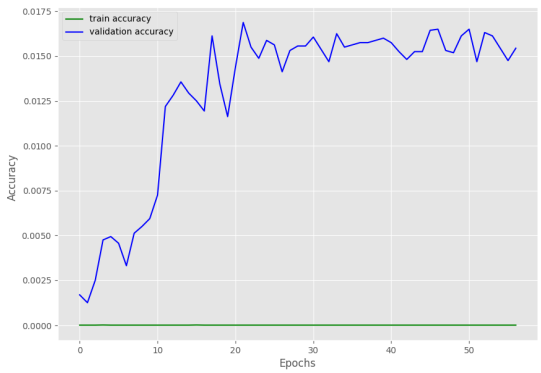


Figura 4.23: Precisión (mAP) durante el entrenamiento con 100 personas con distribución del conjunto en 90/10 y sin aumento de datos.

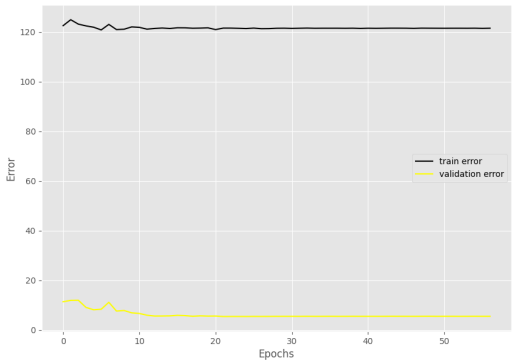


Figura 4.22: Error MSE (m^2) durante el entrenamiento con 100 personas con distribución del conjunto en 90/10 y sin aumento de datos.

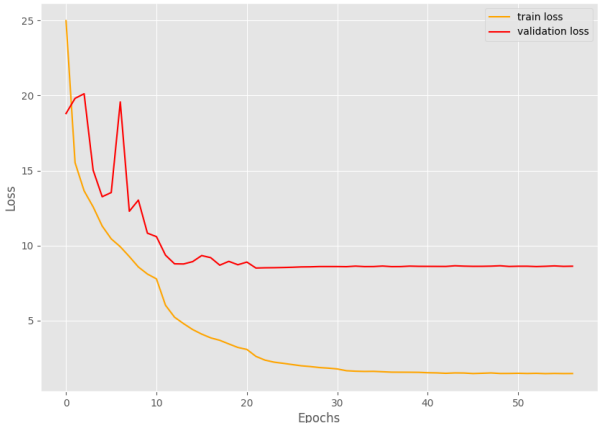


Figura 4.24: Pérdidas durante el entrenamiento con 100 personas con distribución del conjunto en 90/10 y sin aumento de datos.

A partir de las figuras de la 4.22 a 4.24, se ha podido comprobar que efectivamente, al modificar la repartición de los datos, el entrenamiento ha mejorado un poco, pero sigue sin ser suficiente, por lo que a continuación se va a experimentar diferentes combinaciones del uso de los métodos de aumento de datos para intentar mejorar el aprendizaje de la red, de forma que los métodos utilizados no supongan un error añadido y finalmente intentar encontrar el error que hay en el entrenamiento.

Tabla 4.10: Comparación de los entrenamientos con 100 personas con distribución del conjunto en 90/10. Con y sin aumento de datos.

		MSE (m^2)	*Accuracy* (mAP)	Pérdidas
Sin aumento de datos	Entrenamiento	121,60	0,0	1,587
	Validación	5,60	0,0161	8,651
Con aumento de datos	Entrenamiento	120,180	0,0	20,593
	Validación	14,595	0,0015	23,852

En la tabla 4.10 se muestra una comparación de los resultados obtenidos al entrenar sin utilizar los métodos de aumento de datos y aplicándolos todos a cada *frame* del conjunto de entrenamiento. Se puede observar que el resultado no ha mejorado, por lo que a continuación, se va a intentar disminuir las

pérdidas y aumentar la precisión eliminando la rotación y el escalado como método de aumento de datos, ya que estos añaden valores nulos que dificultan el entrenamiento. Un ejemplo de este efecto se muestra en la figura 4.25.

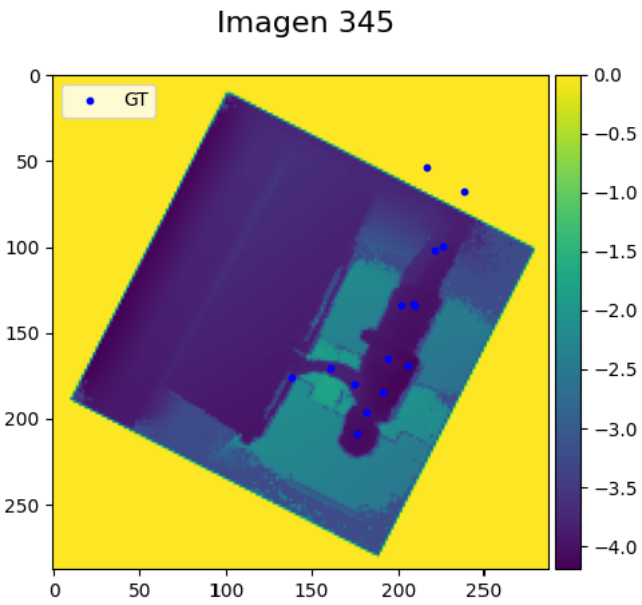


Figura 4.25: Imagen tras rotación y escalado.

4.3.4.2 Con aumento de datos, pero sin rotación

La siguiente prueba que se realizó para intentar disminuir las pérdidas, fue eliminar el paso de la rotación a la hora de aumentar los datos, puesto que al rotar una imagen en la que se encuentra una persona, ésta no acabaría en una posición realista, por lo que no añade valor. Al comparar los resultados de las figuras 4.26-4.28 con los de la tabla 4.10, se puede observar que las pérdidas, el MSE y la precisión ha mejorado ligeramente.

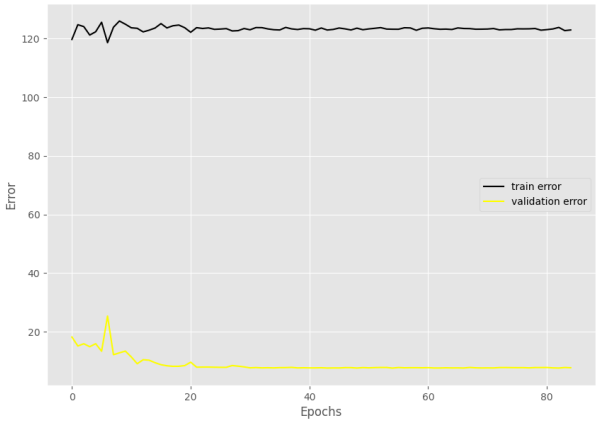


Figura 4.26: Error MSE (m^2) durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aumento de datos, pero sin rotación.

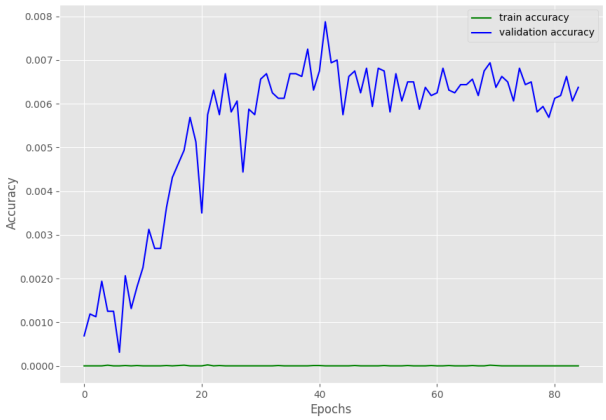


Figura 4.27: Precisión (mAP) durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aumento de datos, pero sin rotación.

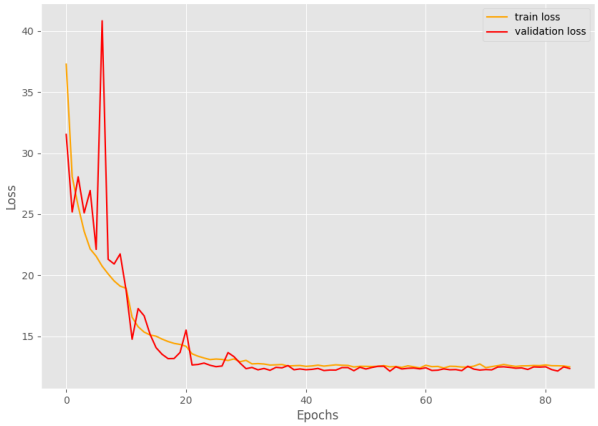


Figura 4.28: Pérdidas durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aumento de datos, pero sin rotación.

4.3.4.3 Con aumento de datos, pero sin rotación ni escalado

En este caso, además de haber eliminado la fase de rotación, también se ha quitado la de escalado, ya que aplicar un factor de escalado menor de 1 provoca un recuadro exterior de ceros, lo cual afecta bastante a la red y le impide aprender correctamente. En la figura 4.29 se muestra la curva del error en m^2, en la figura 4.30 la curva de precisión y en la figura 4.31 las pérdidas. En todas ellas se puede ver que básicamente no ha mejorado nada, por lo que ya se sabe seguro que el error está en el código de entrenamiento, como se intuía en un inicio.

4.3.4.4 Con aleatoriedad en el aumento de datos

Finalmente, se ha añadido que la forma de generar nuevos datos sea aleatoria, es decir, que no se apliquen todos los métodos de aumento de datos a todas las imágenes, si no que se vayan aplicando cada uno de forma aleatoria. Los resultados se han añadido a la tabla 4.11, junto con los resultados con aumento de datos sin rotación y sin escalado ni rotación.

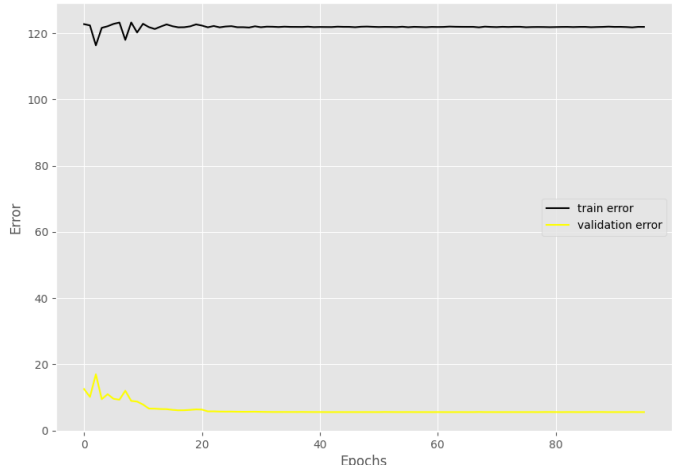


Figura 4.29: Error MSE (m^2) durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aumento de datos, pero sin rotación ni escalado.

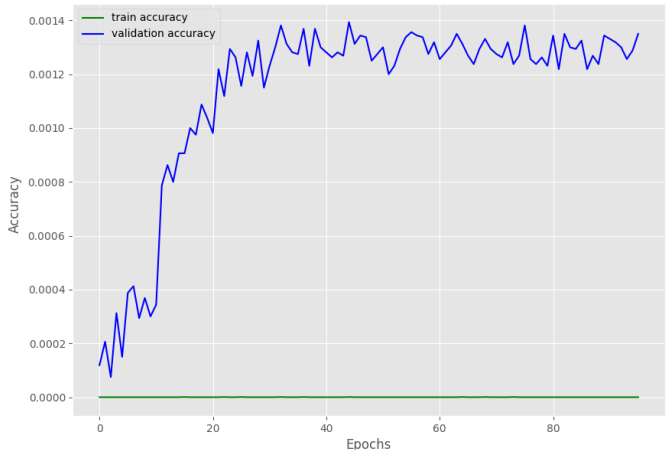


Figura 4.30: Precisión (mAP) durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aumento de datos, pero sin rotación ni escalado.

Tabla 4.11: Entrenamiento 3. Comparación de los entrenamientos con 100 personas.

		MSE (m^2)	*Accuracy* (mAP)	Pérdidas
Con aumento de datos sin rotación	Entrenamiento	123,366	0,0	12,641
	Validación	7,723	0,0064	12,50
Con aumento de datos sin rotación ni escalado	Entrenamiento	122,574	0,0	8,165
	Validación	5,544	0,00123	8,741
Con aleatoriedad en el aumento de datos	Entrenamiento	7916	0,0	7,409
	Validación	25,049	0,0019	8,371

Otro experimento que se realizó para observar el efecto que tenía, es modificar la regla de mAP para que la comparación no sea con $1\,cm^2$, si no con $4\,cm^2$ y de esta forma ver si aumenta la precisión. Aunque se ha podido comprobar que no ha sido útil, ver figuras 4.32, 4.33 y 4.34. Sí se ha conseguido mejorar

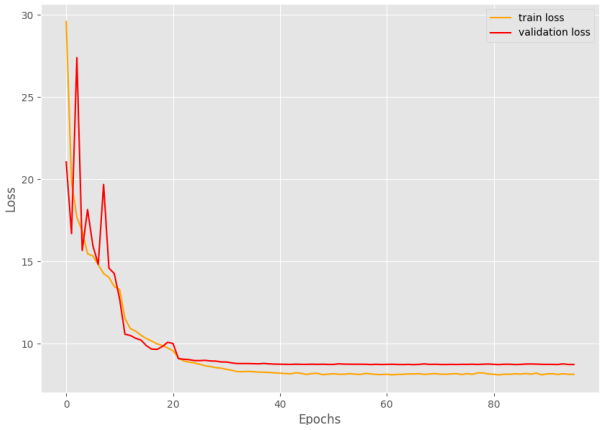


Figura 4.31: Pérdidas durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aumento de datos, pero sin rotación ni escalado.

la precisión, pero no han mejorado las pérdidas ni el error. Observando la curva del error se aprecia que ésta está creciendo en vez de disminuyendo, lo cual no es coherente.

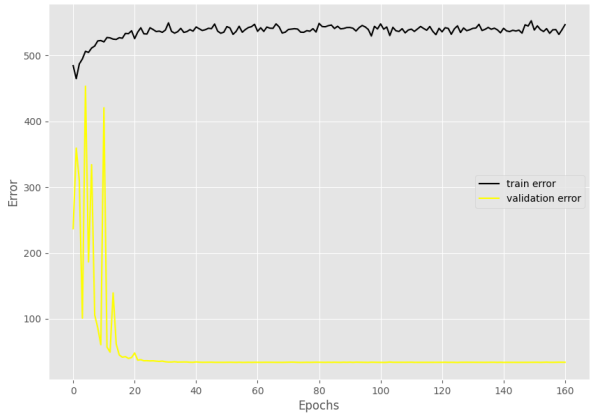


Figura 4.32: Error MSE (m^2) durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aleatoriedad en el aumento de datos. Regla mAP<20cm.

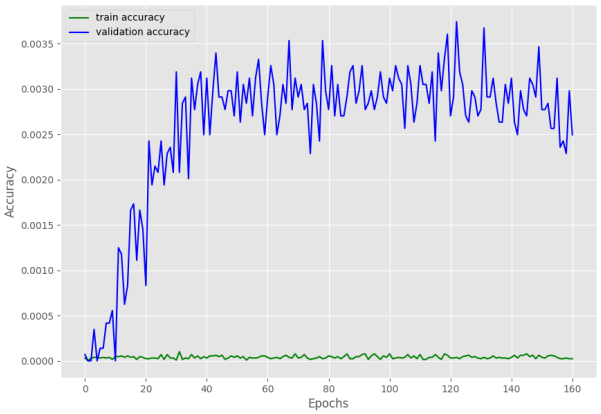


Figura 4.33: Precisión (mAP) durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aleatoriedad en el aumento de datos. Regla mAP<20cm.

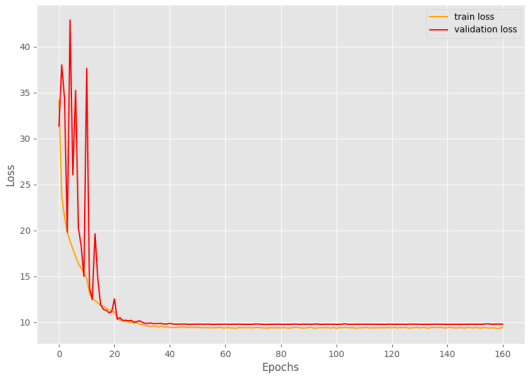


Figura 4.34: Pérdidas durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aleatoriedad en el aumento de datos. Regla mAP<20cm.

Adicionalmente, se ha cambiado la función de pérdidas que usan por una métrica muy utilizada, que es la MSE. Esto se ha aplicado con la función propia de la librería de PyTorch, `torch.nn.MSELoss`. En los resultados obtenidos (figuras 4.35, 4.36 y 4.37) se puede comprobar que la función de pérdidas que aportan y aplican en el estudio A2J es bastante mejor que la MSE para este tipo de aplicaciones. Aunque es muy probable que los resultados sean tan malos debido a algún fallo en el entrenamiento, que aún no se ha conseguido localizar, por lo que no se podría tener este resultado muy en cuenta.

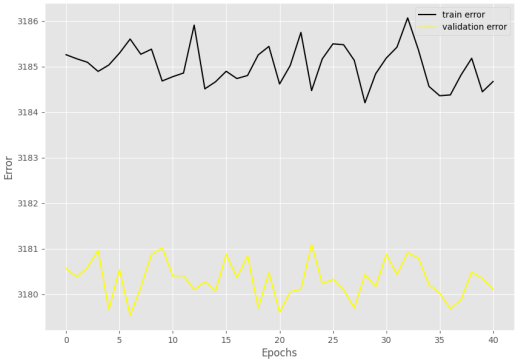


Figura 4.35: Error MSE (m^2) durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aleatoriedad en el aumento de datos. MSELoss.

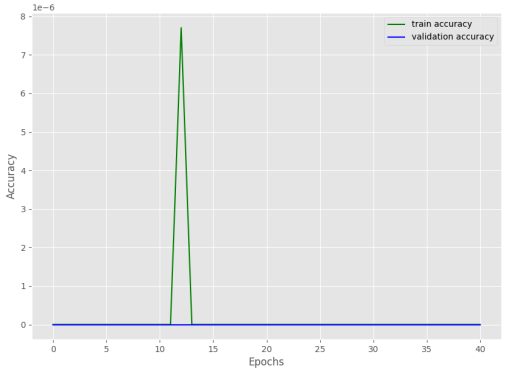


Figura 4.36: Precisión (mAP) durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aleatoriedad en el aumento de datos. MSELoss.

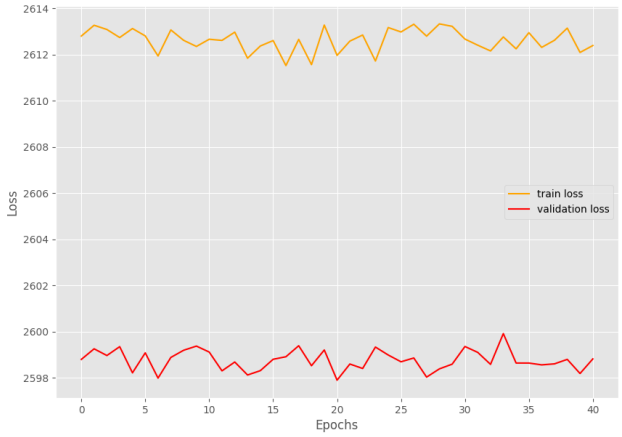


Figura 4.37: Pérdidas durante el entrenamiento con 100 personas y distribución del conjunto en 90/10. Con aleatoriedad en el aumento de datos. MSELoss.

Los resultados de las pruebas a comentar a continuación se han agrupado en la tabla 4.12 y en todas se han aplicando el aumento de datos aleatorio, que es el que se ha comprobado que funciona mejor, siempre teniendo presente que hay un error en el entrenamiento que se está tratando de resolver.

El experimento utilizando el modelo preentrenado en la base de datos ITOP a partir del cual inicializar los pesos del entrenamiento actual, *fine tuning*, como era de esperar, no ha ofrecido ninguna mejora ya que en la base de datos en la que se entrenó mantenían unas condiciones de entorno completamente distintas a las de la base de datos ETRI.

Como última prueba, se ha intentado un entrenamiento desde cero, es decir, sin partir del modelo preentrenado en la base de datos ImageNet, lo cual tampoco ha mejorado el resultado.

Tabla 4.12: Comparación de los entrenamientos con 100 personas y distribución del conjunto en 90/10.

		MSE (m^2)	*Accuracy* (mAP)	Pérdidas
Con aleatoriedad en el aumento de datos. Regla mAP<20cm.	Entrenamiento	541,42	0,0	9,388
	Validación	33,935	0,0003	9,787
Con aleatoriedad en el aumento de datos. *Fine tuning.*	Entrenamiento	532,70	0,0	8,252
	Validación	31,601	0,0006	9,559
Con aleatoriedad en el aumento de datos. Entrenamiento desde cero.	Entrenamiento	548,57	0,0	9,356
	Validación	33,536	0,0005	9,730

4.4 Análisis de los erróres

En esta sección se va a tratar de analizar los errores para tener pistas sobre dónde se encuentra el fallo, ya que modificando los entrenamientos no se ha hallado. Se ha analizado la distribución del error MSE (figura 4.38) y del mAP (figura 4.39) para ver si sólo son algunos *frames* los que perjudican el entrenamiento o todos contribuyen.

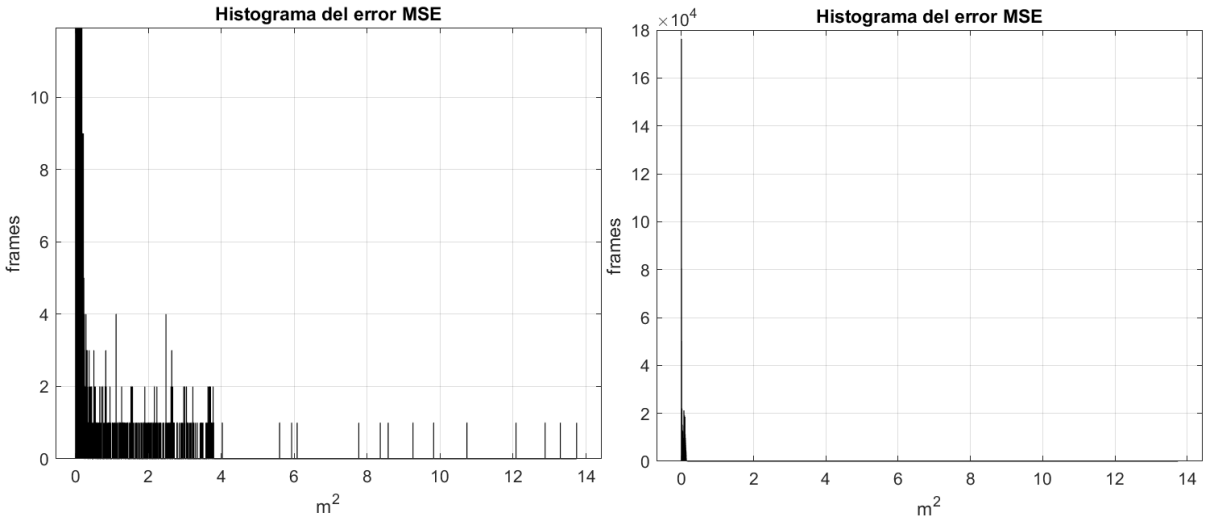


Figura 4.38: Histograma del error MSE con y sin *zoom*.

La mediana del error MSE de todos los *frames* de todas las épocas que se utilizan en un entrenamiento es igual a 0,024 m^2, 0,15 m, lo cual no es excesivo, pero hay *frames* en los que el error MSE llega los 15 m^2, lo cual supondría que la estimación de las articulaciones las está colocando en los puntos más alejados de la habitación de la imagen.

Se ha comprobado que una imagen que en un inicio genera muy poco error hay épocas muy posteriores en las que provoca mucho error, se puede deber a que la red durante el aprendizaje ha ido moviendo los pesos y en las épocas finales ya se ha alejado del mínimo inicial sin poder volver. Este comportamiento no es usual por lo que demuestra que el entrenamiento no está funcionando bien.

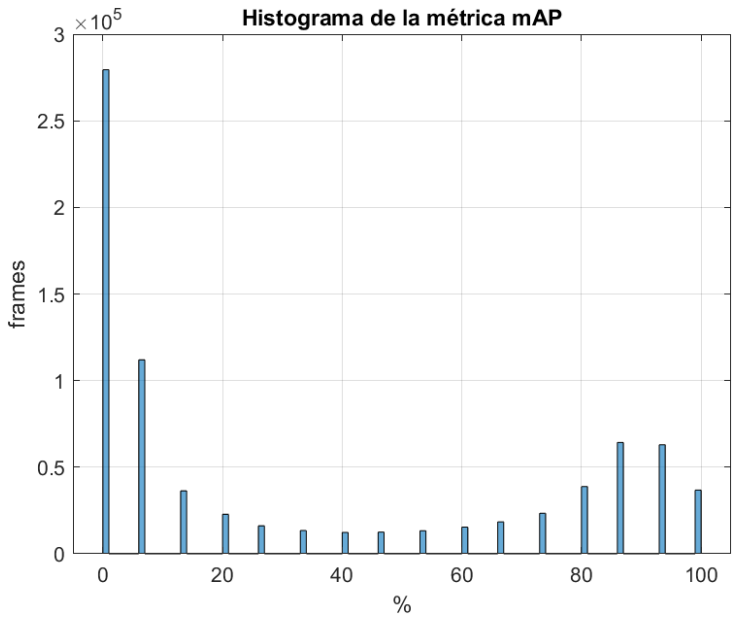
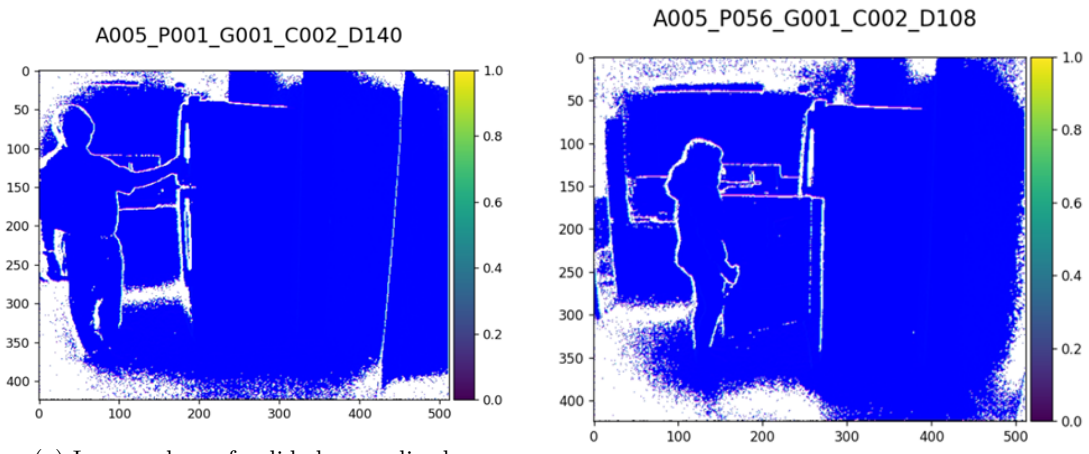


Figura 4.39: Histograma de la métrica de precisión mAP.

Se ha podido verificar que los *frames* con errores más altos son los que tienen alguna articulación obstruida por algún objeto o por su propio cuerpo, y los que menor error tienen son los que se ven de frente sin obstrucciones. De las imágenes utilizadas de la base de datos ETRI, una de las que menor error tiene es la de la figura 4.40a y de las que más la de la figura 4.40b.



(a) Imagen de profundidad normalizada con mínimo de obstrucciones.

(b) Imagen de profundidad normalizada con obstrucciones.

Figura 4.40: Imagen de profundidad preprocesada y el esqueleto estimado. Base de datos ITOP.

Viendo como el aprendizaje no se ha realizado correctamente, se va a entrenar la red con la misma base de datos que utilizan en el estudio A2J, la base de datos ITOP, para de esta forma comprobar definitivamente si el problema reside en la base de datos utilizada o si es porque el entrenamiento está mal implementado.

4.5 Resultados experimentales ITOP

Como se ha introducido, se ha dejado de realizar experimentos con la base de datos ETRI porque el entrenamiento no funcionaba correctamente y se obtenían unos resultados muy malos, por lo que se ha retrocedido un paso, eliminando fuentes de variabilidad para poder realizar una experimentación más amplia modificando parámetros de la red, la arquitectura, etc. Los experimentos a realizar con la base de datos ITOP se indicarán a continuación, y únicamente se mostrará las curvas de aprendizaje en el caso del entrenamiento con las mismas condiciones que el sistema original. El resto de resultados se aportarán a través de una tabla.

1. Entrenamiento en las mismas condiciones que el sistema original.

2. Entrenamiento desde un modelo preentrenado. En este experimento se han utilizado para inicializar este entrenamiento los pesos que obtuvieron A2J al entrenar la red con la base de datos ITOP.

3. Variaciones en el *drop out*. En esta prueba se han utilizado tres valores distinto de *drop out*: 25, 50 y 75 %.

4. Variaciones en la función de activación.

5. Modificaciones en la arquitectura. En este caso, como todas las capas intermedias de la red son iguales, se va a probar a quitar una de estas y a añadir una intermedia exactamente igual.

6. Cambios en el tamaño del *batch*. Se va a entrenar con dos tamaños distintos de *batch* (16 y 32), a parte del inicial de 64. También se intentó con 128, pero no se podía porque durante la ejecución se quedaba sin memoria en la GPU.

4.5.1 Entrenamiento en las mismas condiciones que el sistema original

En esta prueba los parámetros utilizados son los mismos que se utilizaron en la primera prueba de entrenamiento con la base de datos ETRI, los cuales se mencionan en la subsección 4.3.1. El resto de experimentos mantendrán todos los valores iniciales, excepto el parámetro que se está variando para observar como afecta en el resultado.

En este experimento no se añade *dropout*, se utiliza la función ReLU en la capa de activación, la arquitectura mantiene sus 4 capas convolucionales intermedias iguales y el tamaño del *batch* a 64.

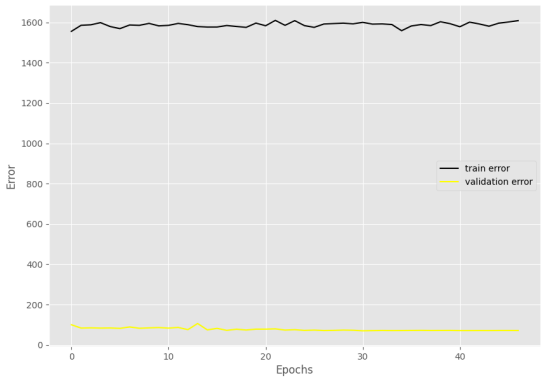


Figura 4.41: Entrenamiento ITOP. Error MSE (m^2) durante el entrenamiento con parámetros iniciales.

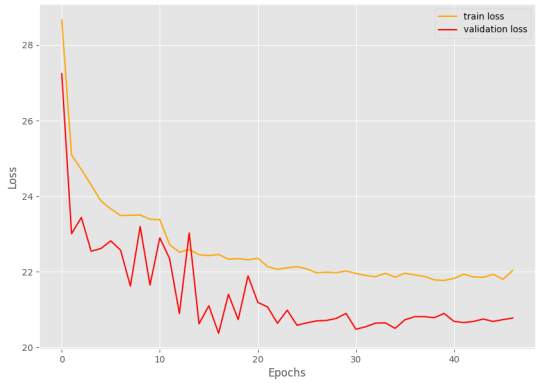


Figura 4.42: Entrenamiento ITOP. Pérdidas durante el entrenamiento con parámetros iniciales.

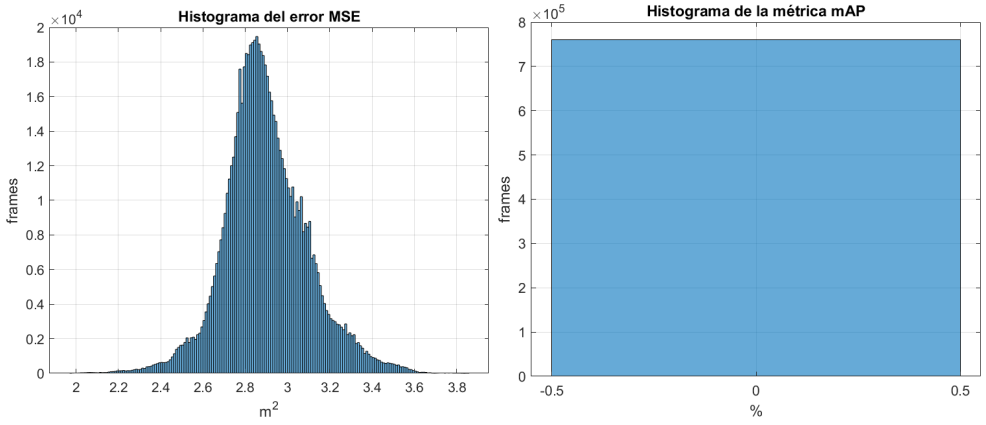


Figura 4.43: Histograma del error MSE y de la métrica de precisión mAP.

En este caso la distribución es gaussiana con media de error entre todas las épocas de 2,898 m^2, un valor muy alto, por lo que es normal que la precisión sea nula para todos los *frames*.

En la figura 4.42 se puede ver que las pérdidas de validación son inferiores a las de entrenamiento, lo cual es extraño. Con este resultado ya es definitivo que hay un fallo grave en el entrenamiento. En un inicio se fueron haciendo pequeños cambios y se corrigieron algunos errores, pero no se solucionó el problema principal, por lo que se pasó a reescribir la parte de código del tratamiento de los datos de forma que se replicara exactamente el mismo procedimiento que siguen A2J en el código de *test*. Una vez hecho esto, ya se empezaron a obtener unos resultados coherentes. En las figuras 4.44, 4.45 y 4.46 se muestran las curvas de entrenamiento de las pérdidas, del MSE y de la precisión obtenidas.

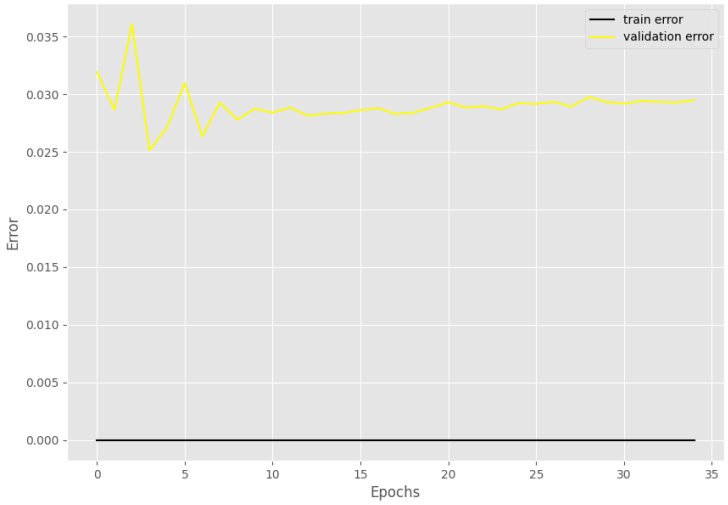


Figura 4.44: Correcto entrenamiento ITOP. Error MSE (m^2) durante el entrenamiento con parámetros iniciales.

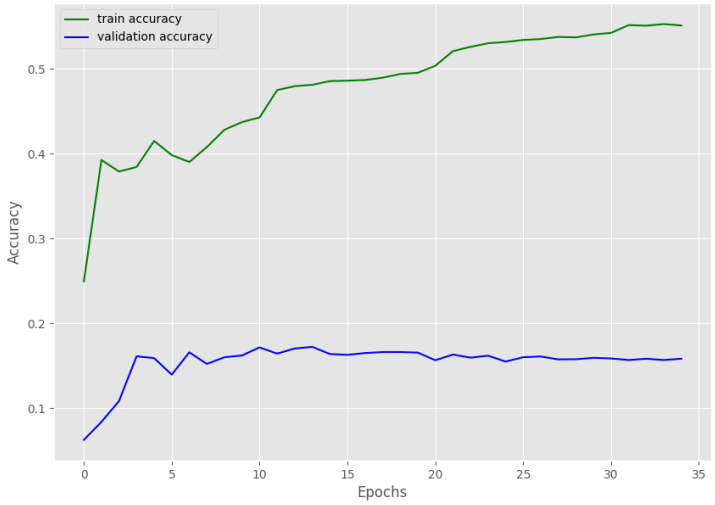


Figura 4.45: Correcto entrenamiento ITOP. Precisión (mAP) durante el entrenamiento con parámetros iniciales.

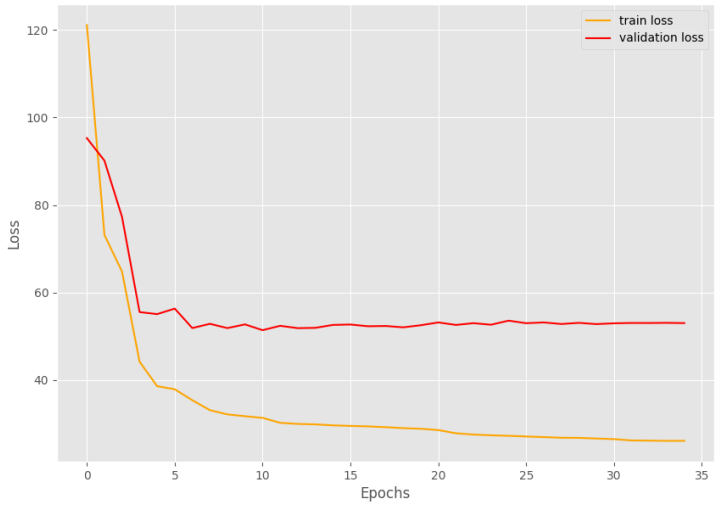


Figura 4.46: Correcto entrenamiento ITOP. Pérdidas durante el entrenamiento con parámetros iniciales.

Como conclusión, se va a añadir una tabla con todos los resultados obtenidos en esta sección para poder compararlos mejor. En vista de que se ha solucionado el problema en el último momento, los resultados de la tabla 4.13 se han obtenido limitando el número de épocas a 30, por lo que posiblemente no sean los valores óptimos.

Tabla 4.13: Entrenamiento ITOP. Comparación de los entrenamientos.

		Drop out			Activación	Arquitectura		*Batch*		Original
		25	50	75	Leaky ReLU	−1 capa	+1 capa	16	32	
Accuracy (mAP)	Entrenamiento	0,484	0,386	0,336	0,476	0,474	0,420	0,440	0,443	0,551
	Validación	0,0003	0,158	0,156	0,166	0,161	0,168	0,161	0,158	0,156
Pérdidas	Entrenamiento	28,647	33,486	36,003	30,084	30,525	33,497	36,040	36,028	26,112
	Validación	51,519	50,840	51,202	51,472	52,727	41,172	55,460	55,328	53,022
MSE (m^2)	Entrenamiento	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
	Validación	0,029	0,025	0,025	0,027	0,028	0,025	0,026	0,026	0,029

Como se puede observar, ahora los resultados si son coherentes, pero no se ha conseguido obtener una alta precisión. Se podría definir el mejor entrenamiento, fijándonos en los resultados por cada parámetro, de forma que éste tenga menores pérdidas y error, y una alta precisión:

- *Drop out* a 50 %.
- En el caso de la función de activación, la ReLU es ligeramente peor que la Leaky ReLU con una pendiente negativa de 0,01.
- Respecto al número de capas a utilizar, aumentando en 1 las capas intermedias se consigue una ligera mejoría en los resultados.
- El tamaño del *batch* es mejor si es de 16.

Al ejecutar el *test* con el mejor modelo entrenado se han conseguido los resultados de estimación de la pose de las articulaciones de la tabla 4.14. Comparándolos con la tabla 4.4, se puede observar que no coinciden con los obtenidos mediante el modelo entrenado por A2J. En este caso se ha conseguido una precisión del 32 % y un error de aproximadamente 14 cm, valores que no terminan de ser buenos. El análisis del motivo por el cual no se consigue un resultado cercano al de A2J se dejará como línea de investigación futura.

Tabla 4.14: Resultados *test* con la base de datos ITOP. La D hace referencia a derecha y la I a izquierda.

Número de la articulación	**Articulación**	***Accuracy* (mAP)**	**MSE** (m^2)
0	Cabeza	0,398	0,008
1	Cuello	0,409	0,007
2	Hombro D	0,390	0,009
3	Hombro I	0,407	0,010
4	Codo D	0,178	0,019
5	Codo I	0,148	0,024
6	Mano D	0,045	0,060
7	Mano I	0,057	0,075
8	Tronco	0,407	0,007
9	Cadera D	0,389	0,008
10	Cadera I	0,411	0,009
11	Rodilla D	0,455	0,009
12	Rodilla I	0,452	0,011
13	Pie D	0,295	0,011
14	Pie I	0,378	0,013
Media		0,321	0,019

En el último experimento relacionado con los entrenamientos de la red se ha utilizado la base de datos ETRI y se ha manteniendo las condiciones de la red como las iniciales, pero añadiendo la aplicación del aumento de datos de forma aleatoria, como en el caso anterior. Los resultados se muestran en la tabla 4.15 y se puede ver que estos no son del todo buenos, ya que el error es de aproximadamente 17 cm y la precisión del 14,6 %. La mejora del sistema se propondrá como línea futura.

Tabla 4.15: Entrenamiento correcto con ETRI.

		MSE (m^2)	*Accuracy* (mAP)	Pérdidas
Con aleatoriedad en el aumento de datos	Entrenamiento	0,0	0,259	674,83
	Validación	0,032	0,146	724,56

Añadir que también se modificó el *learning rate* a valores mayores y menores del escogido por A2J (0,001 y 0,0001), pero no hubo mejora, por lo que se mantuvieron todas las pruebas con el mismo *learning rate* de 0,00035.

4.6 Estimación de parámetros de destreza y funcionalidad

Uno de los principales objetivos de este proyecto era estimar parámetros que pudieran ser de utilidad a los fisioterapeutas, ayudándoles en la tarea de determinar el grado de destreza que tiene una determinada persona.

Centrándonos en el tren superior del cuerpo humano, el principal factor que demuestra torpeza en la destreza es el modo de andar o marcha y el equilibrio, en concreto la posición del CG. La marcha se ha analizado tomando en consideración el periodo del paso, el ángulo de rotación de la cadera, el de la rodilla y el ángulo de la caída de la pelvis. El equilibrio, y por ende la caída, se ha examinado a partir de la proyección del CG sobre la base de sustentación, el cambio de velocidad de un *frame* a otro y la alteración en el contorno del cuerpo.

Añadir, que todas las estimaciones se realizarán sobre las posiciones reales de las articulaciones, el *ground truth*.

4.6.1 Equilibrio

Comenzando por la estimación del nivel de equilibrio, éste se puede determinar en función de lo alejada que se encuentre la proyección del centro de gravedad en el suelo respecto al centro de la base de sustentación, la cual se puede aproximar a la circunferencia que forman los pies al estar de pie. El diámetro de la circunferencia sería la distancia que hay entre la parte exterior de cada pie, donde se encuentra el meñique. Mientras mayor sea la separación entre los pies, menor posibilidad de caerse tiene uno, ya que la proyección del centro de gravedad, si no se hacen posturas extrañas, es muy probable que se encuentre dentro de la circunferencia. Mientras la proyección del CG se mantenga dentro de la circunferencia, el individuo se mantendrá estable y en el momento en el que haya una salida brusca de este punto, la persona habrá perdido el equilibrio y muy probablemente se habrá caído al suelo.

Para conseguir un centro de masas preciso en 3D se necesitaría conocer el peso de la persona, pero como no se dispone de ese dato, se ha realizado una estimación aproximada. Para determinar la posición del CG se utilizaron los valores de la tabla 2.1 y las coordenadas de las articulaciones, aplicándolos en la ecuación 4.4. Al emplear esa fórmula y comprobar los desplazamientos del CG, este mostraba valores de metros, lo cual no era posible, ya que revisando los *frames* en los que ocurría, la persona no estaba haciendo ningún movimiento extraño, simplemente estaba de pie. Debido a ello se modificó la ecuación y se aplicó la mostrada en [102], ecuación 4.4, consiguiendo esta vez valores del orden de centímetros. El punto negativo es que el cálculo es en 2D, no se tiene en cuenta la profundidad, pero al proyectar el CG en el suelo se podría asignar como coordenada "z" el valor de profundidad del punto medio de la distancia entre los pies, que estando la persona en una posición vertical sería válido. A parte, como norma se ha decidido añadir/disminuir 10 cm a la coordenada "x" o "z", en función de en qué dirección esté mirando, de cada pie. Al más alejado se le suman y al más cercano se le restan.

Respecto al origen del sistema de referencia, éste se encuentra en la localización de la cámara de profundidad y los ejes de coordenadas parten de ella, como se muestra en la figura 4.47

Figura 4.47: Definición de los ejes y origen de coordenadas del sistema de referencia [103].

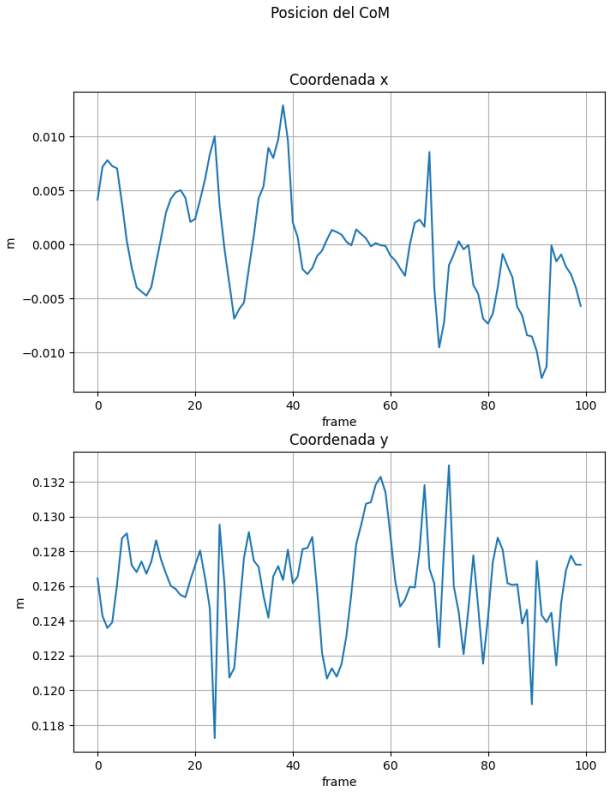
$$CGT = \sum [Ki(Gpi - Gdi)] \cdot Pi \tag{4.4}$$

Como se indicó en la ecuación 2.1, *Gpi* hace referencia a las coordenadas proximales y *Gdi* a las distales de un segmento. Estos puntos en función del segmento al que corresponden serían los siguientes (tabla 4.16):

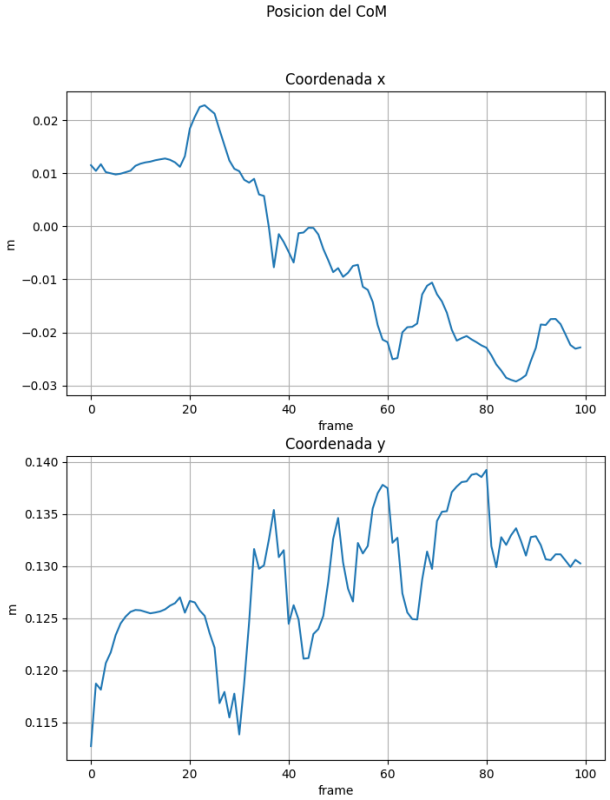
Tabla 4.16: Asignación de los segmentos corporales.

Segmento	**Pto. Proximal**	**ID**	**Pto. distal**	**ID**
Cabeza	Cabeza	0	Cuello	1
Tronco	Cuello	1	Torso	8
Brazo D	Hombro D	2	Codo D	4
Brazo I	Hombro I	3	Codo I	5
Antebrazo D	Codo D	2	Mano D	6
Antebrazo I	Codo I	5	Mano I	7
Mano D	Mano D	6	-	-
Mano I	Mano I	7	-	-
Muslo D	Cadera D	9	Rodilla D	11
Muslo I	Cadera I	10	Rodilla I	12
Pierna D	Rodilla D	11	Pie D	13
Pierna I	Rodilla I	12	Pie I	14
Pie D	Pie D	13	-	-
Pie I	Pie I	14	-	-

En la figura 4.48a se muestra el movimiento del CG tanto en el eje “x”, como en el y a lo largo de los primeros 100 *frames* de la base de datos ITOP. En dichas imágenes hay un hombre mirando al frente que está levantando y bajando una pierna y después ha girado 90^o sobre si mismo para luego volver a la posición inicial. En la figura 4.48b también se muestra el desplazamiento del CG a lo largo de los primeros 100 *frames* de la base de datos UTK. En este caso hay una mujer realizando la acción de caminar de derecha a izquierda.



(a) ITOP



(b) UTK

Figura 4.48: Movimiento del centro de gravedad.

Un dato clave para saber si una persona tiene cierta discapacidad al andar, es si durante la marcha obtiene un desplazamiento del CG mayor de 5 cm [23]. En el caso de ITOP, la desviación estándar de la

altura del CG es de 0,28 cm y en el eje x es de 0,5 cm. En UTK en el eje x es de 1,5 cm y en el y 0,59 cm. Así que las personas anteriores tienen una marcha normal.

A continuación, se ha calculado el porcentaje que se desvía el CG respecto a la posición óptima, siendo ésta el punto medio de la recta que forma la separación entre los pies. Se ha probado con las distintas actividades que realizan 2 hombres diferentes en la base de datos ITOP y una mujer en la de UTK.

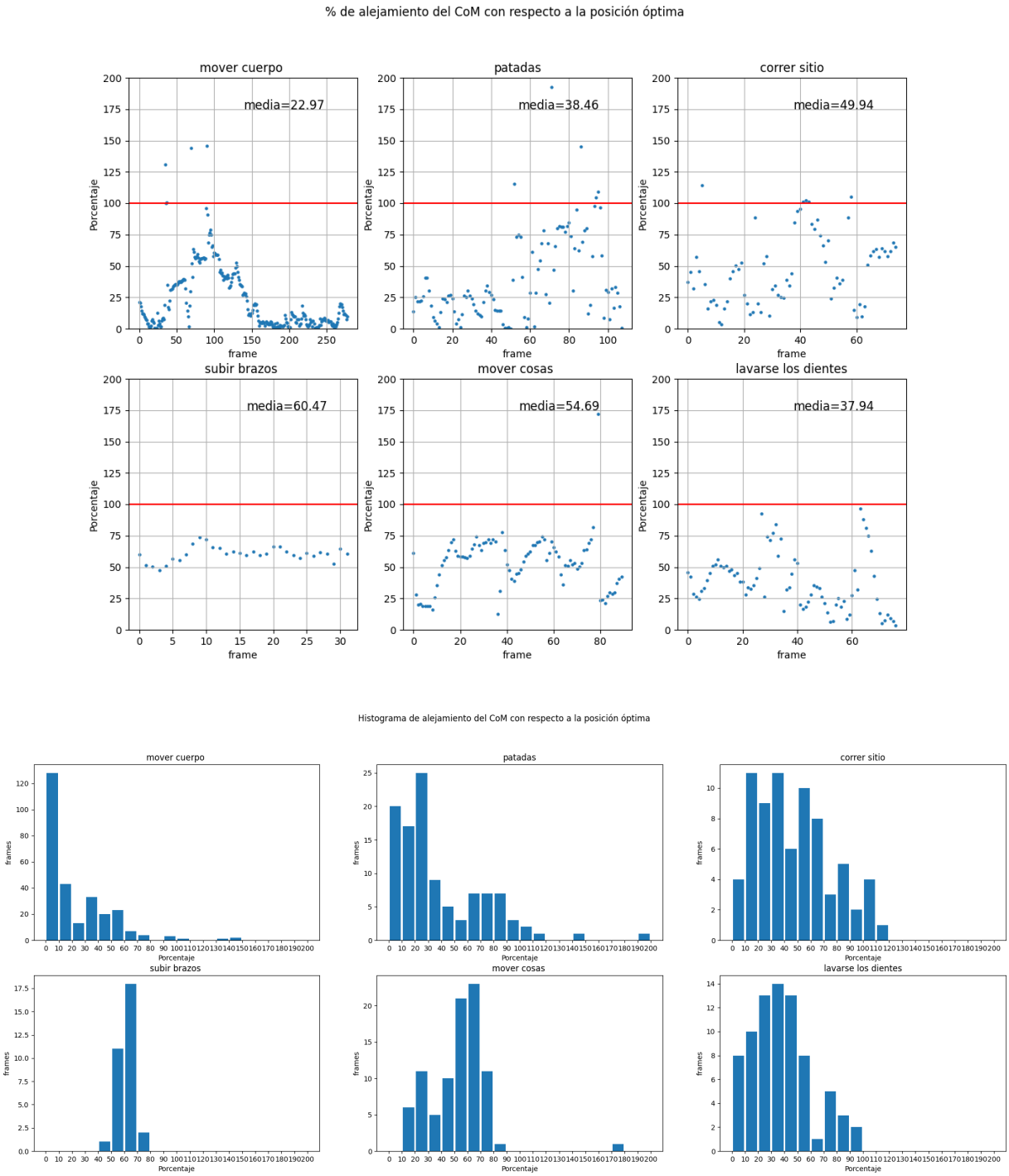


Figura 4.49: Porcentaje que se aleja el CG con respecto a la posición óptima. Hombre 1.

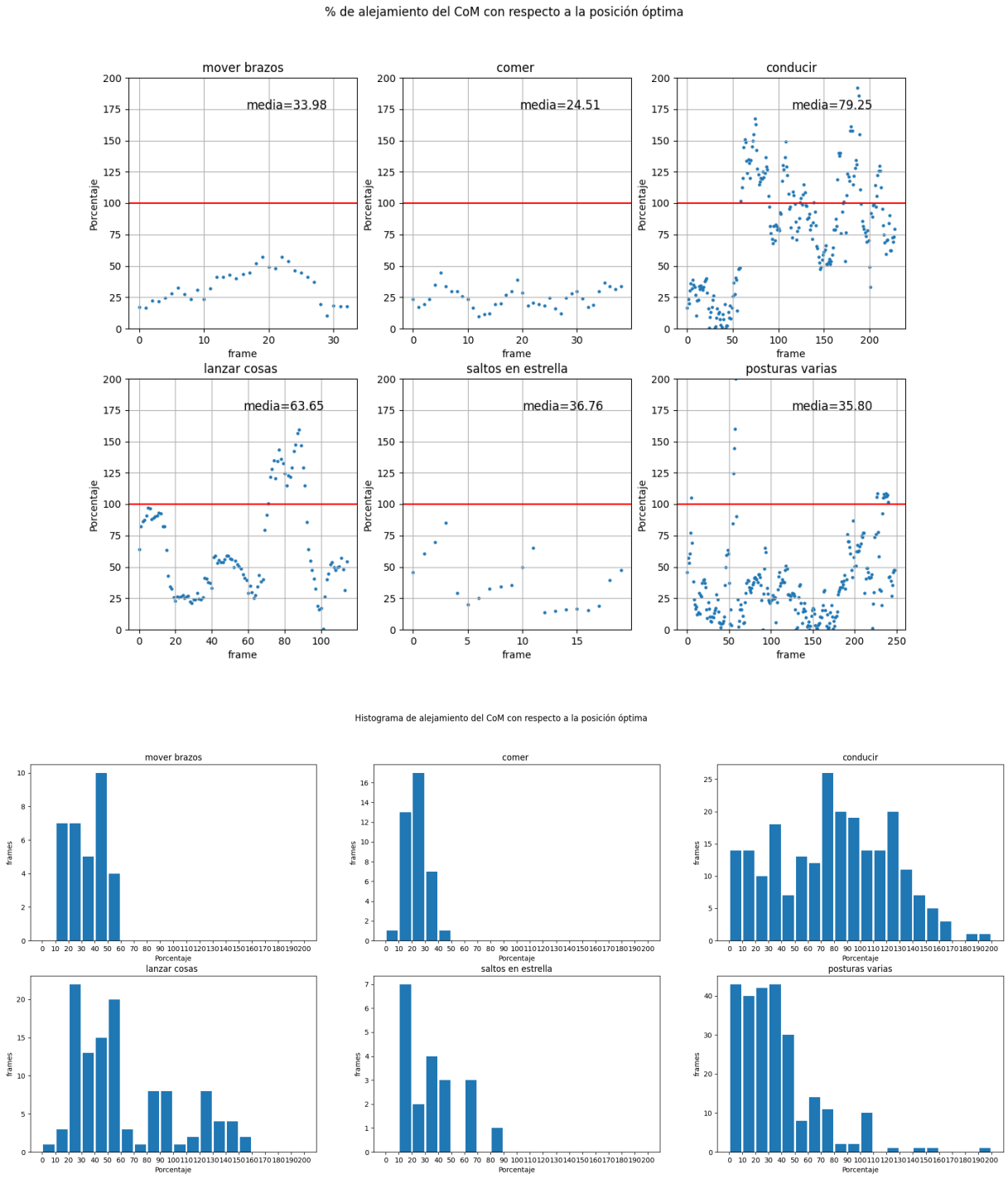


Figura 4.50: Porcentaje que se aleja el CG con respecto a la posición óptima. Hombre 1.

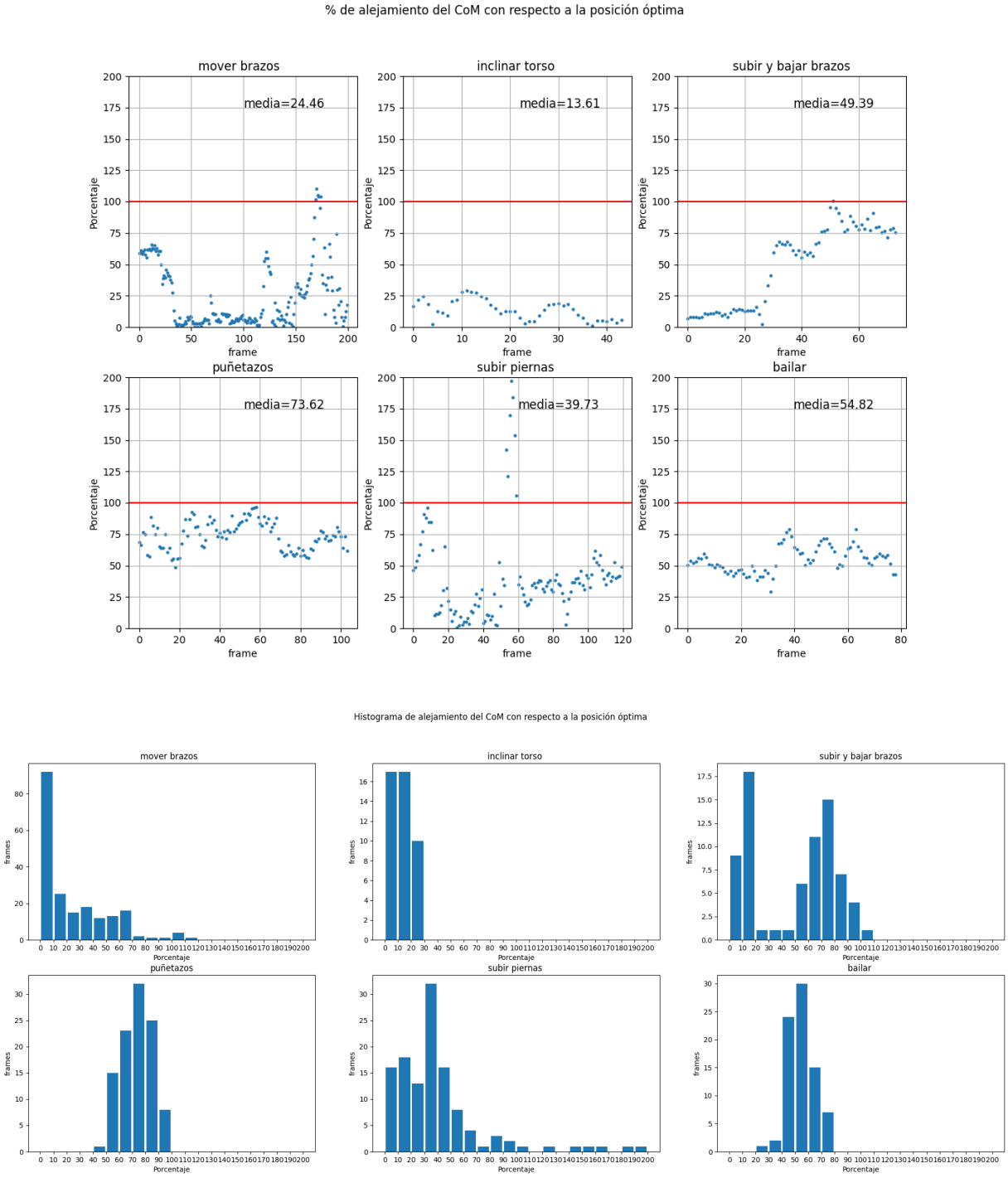


Figura 4.51: Porcentaje que se aleja el CG con respecto a la posición óptima. Hombre 2.

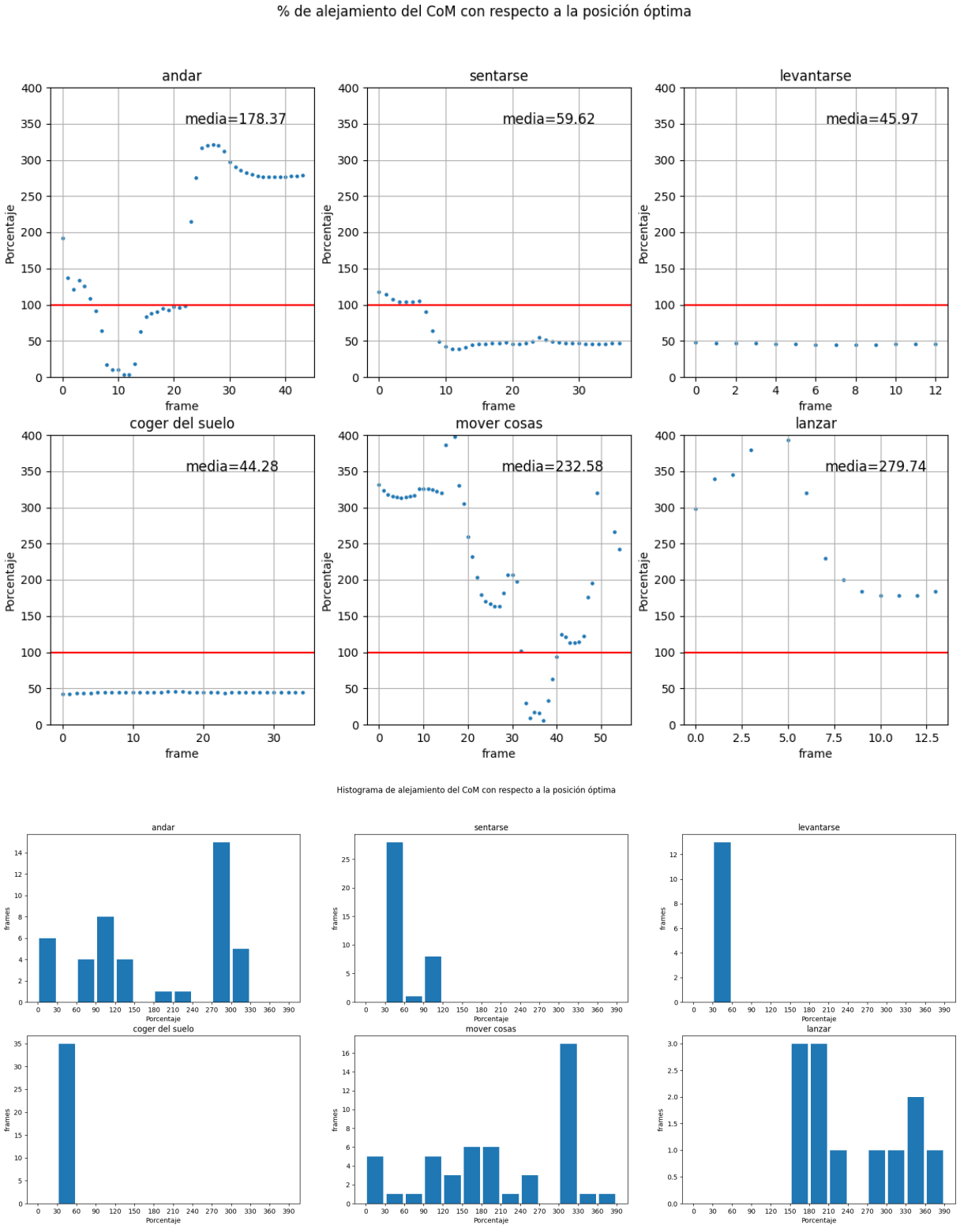


Figura 4.52: Porcentaje que se aleja el CG con respecto a la posición óptima. UTK.

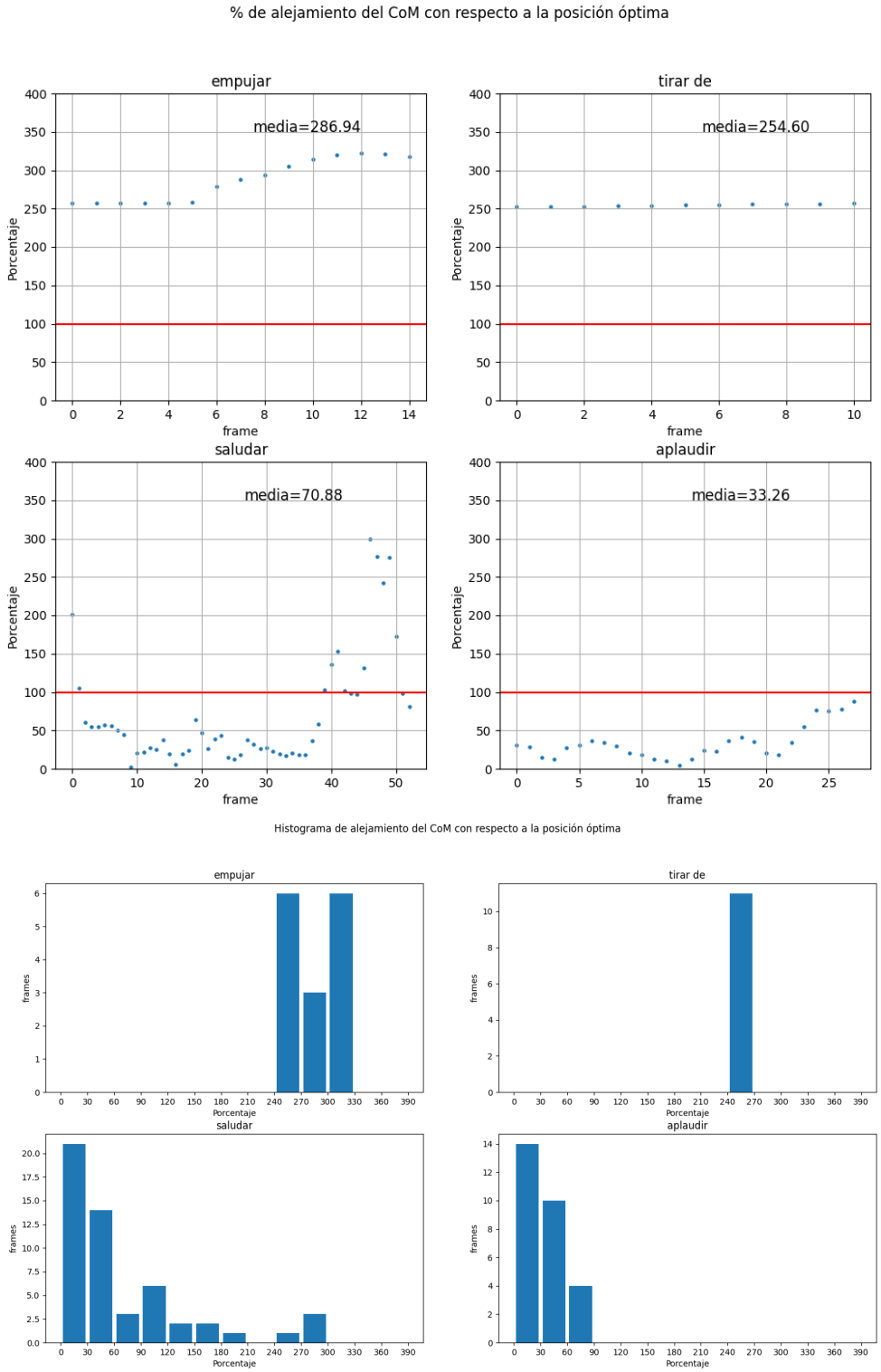


Figura 4.53: Porcentaje que se aleja el CG con respecto a la posición óptima. UTK.

A partir de las figuras anteriores (de la 4.49 a la 4.53) se puede verificar que en las acciones en las que sólo se usan las manos, como aplaudir o mover brazos, el CG se mantiene muy estable, a diferencia de cuando se añade un movimiento de tronco y piernas, en los que la proyección del CG puede llegar a salir de la base de sustentación.

Al estar tomando el valor de profundidad del punto medio de la distancia entre los pies como coordenada z, con este método sólo se podrá saber si la persona ha perdido el equilibrio en el eje al que apunta la puntera de los pies. Por ejemplo, si está mirando a la cámara, se sabrá si ha perdido el equilibrio en el eje "x", es decir, hacia alguno de sus laterales, pero si se cae hacia atrás o adelante no se podrá determinar. Por ello se ha dividido el proceso de detección de caídas en distintas fases, siendo la primera la anterior y las siguientes se han tomado de la referencia [104], donde identifican la caída a partir de tres parámetros críticos: la velocidad de descenso en el centro de la articulación de la cadera, el ángulo de la línea central del cuerpo con el suelo, y la relación entre la anchura y altura del rectángulo del cuerpo humano.

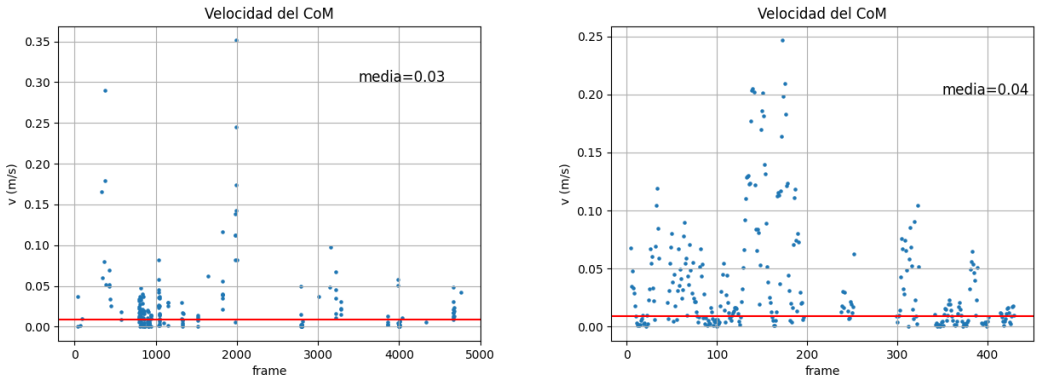
4.6.1.1 Primera condición

Durante el proceso de una caída repentina, el CG del cuerpo va a cambiar en la dirección vertical. Se podría utilizar el punto central de la cadera como CG para reflejar esta característica, pero mejor se va a utilizar el punto calculado. El paso de estar de pie a la caída es muy corto, así que se va ha detectar cada 5 *frames* adyacentes con un intervalo de tiempo de $0,033$ s ($= 1/30$; en ITOP y UTK utilizan 30fps). Ver figura 4.54.

La velocidad de descenso se puede calcular como [104]:

$$v = \frac{|y_{t2} - y_{t1}|}{\delta_t} \tag{4.5}$$

Si v es mayor que una velocidad crítica, se considera caída. En el estudio mencionado han definido experimentalmente esta velocidad como $0,09$ m/s.



(a) ITOP (b) UTK

Figura 4.54: Velocidad de descenso del CG.

4.6.1.2 Segunda condición

En el proceso de caída, la característica más evidente del cuerpo humano es la inclinación del propio cuerpo, y su continuo aumento. Se define la linea central del cuerpo como L, y es la conexión entre el punto medio entre el pie y la rodilla (p.e. izquierda) con la cabeza. A partir de esas coordenadas se calcula el ángulo como:

$$\theta = arctan \frac{|y_0 - y_t|}{|x_0 - x_t|} \tag{4.6}$$

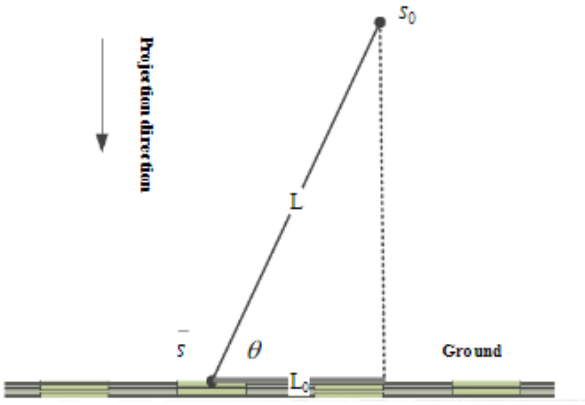
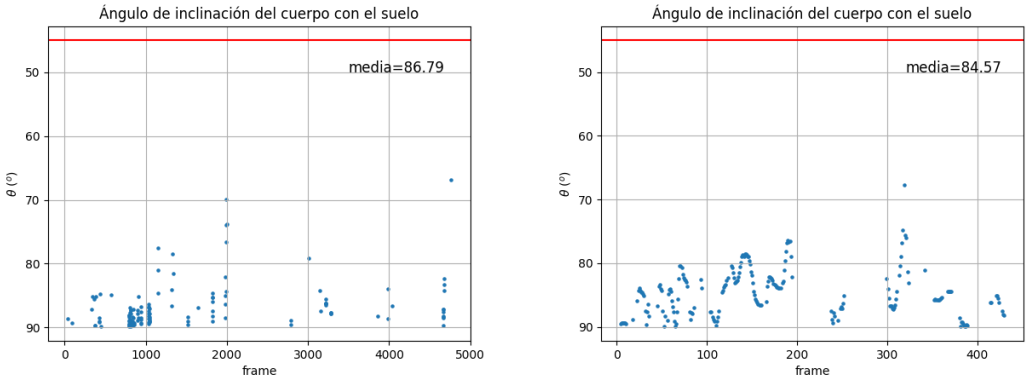


Figura 4.55: Ángulo entre la linea central del cuerpo y el suelo [104].

En la figura 4.55, $\bar{s}$ es el punto medio entre la línea que forman la rodilla (p.e. derecha) con el pie (p.e. derecho) y s_0 es la cabeza. Si el ángulo entre la linea central del cuerpo y el suelo (θ) es menor de 45^o, se puede considerar caída.



(a) ITOP (b) UTK

Figura 4.56: Inclinación del cuerpo.

En ninguno de los dos casos (ver figura 4.56) el ángulo del cuerpo baja de 75^o por lo que no se cumple con la condición de caída. De hecho, las dos personas mantienen de media aproximadamente el mismo ángulo de inclinación del cuerpo cuando están erguidos.

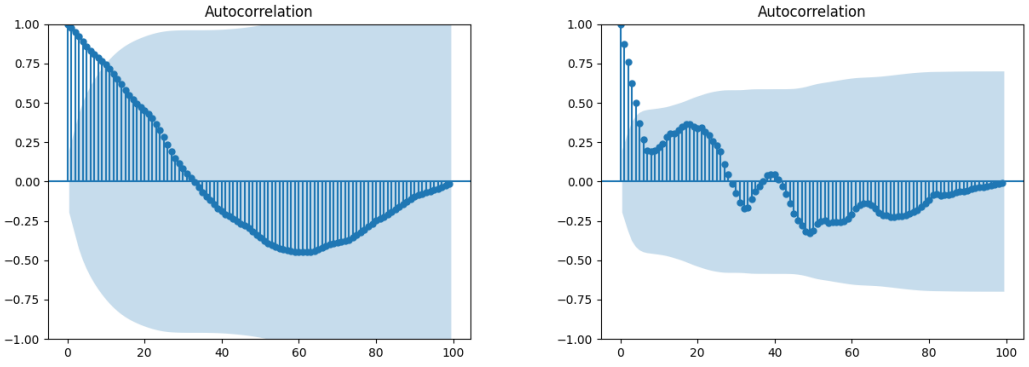
4.6.1.3 Tercera condición

Cuando se detecta una caída, el contorno del cuerpo cambia, por lo que se va a detectar el comportamiento de la caída a partir del cambio de la altura y anchura del contorno rectangular del sujeto. Para ello se hace uso del *bounding box* estimado en la sección anterior. El ratio entre la altura y anchura de la caja es $P = anchura/altura$. Normalmente cuando un humano camina el ratio P es menor de 1, mientras que cuando cae es mayor o igual a 1 [104].

Si se cumplen todas las condiciones, se podrá determinar que la persona evaluada se ha caído.

4.6.2 Marcha

Otro parámetro de importancia es el periodo del paso, el cual se puede obtener a partir de la función de autocorrelación, ya que con ella se puede encontrar patrones repetitivos en una señal. A partir de los valores de las coordenadas "x" e "y" del CMC se aplica la función `sm.tsa.acf` de python y al representarlo se obtiene la figura 4.57. Los datos utilizados han sido los primeros 100 *frames* de la base de datos UTK, en los cuales aparece una mujer caminando de derecha a izquierda.



(a) Eje x (b) Eje y

Figura 4.57: Autocorrelación de los primeros 100 *frames* de UTK.

En la figura 4.57b se aprecia perfectamente la periodicidad y la forma sinusoidal que provoca de la marcha. A partir de esos datos, se calcula el periodo como la diferencia entre los dos primeros máximos, obteniéndose, en este caso concreto, un periodo de 22 *frames*, lo cual son 0,733 segundos. Adicionalmente, se ha verificado de forma visual mirando las imágenes RGB, contando los *frames* que pasan mientras la mujer está andando. De forma visual, o incluso a través de las coordenadas de las articulaciones, se puede estimar como el tiempo que pasa desde un *heel strike* hasta el siguiente con la misma pierna. El *heel strike* se podría evaluar como la distancia entre las rodillas cuando esta es máxima o como el instante en el que la distancia entre el esternón y el tobillo del pie adelantado es máxima e incluso como el instante en el que la altura del centro de masas alcanza un mínimo local [105]. Esto último se puede comprobar en la figura 4.57b, entre mínimo local y mínimo local hay aproximadamente 20 *frames*.

En la figura 4.57 la zona azul de la imagen indica el intervalo de confianza. Cualquier valor en ese área representa una medida que no tiene correlación significativa con la medida más reciente.

Un dato extra es que la frecuencia de la marcha como máximo es de 6 Hz [104], lo que supondría terminar el ciclo completo en 5 *frames* ($1/6 = 0,1667 \cdot 30 fps = 5\, frames$). En el caso de los *frames* utilizados, la frecuencia de la marcha es de 1,364 Hz, valor normal.

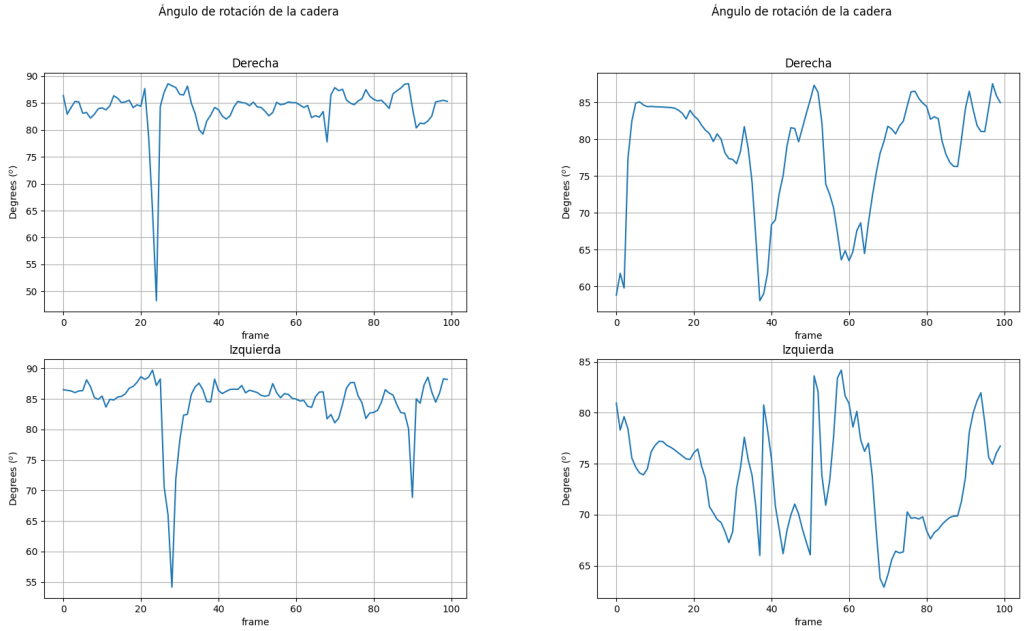
4.6.2.1 Rotación de la cadera

El siguiente parámetro de gran utilidad es el grado de rotación de la cadera durante la marcha. El ángulo de rotación de esta se estima como el ángulo que forma el segmento muslo con respecto al plano que forma la articulación de la cadera [106].

Para realizar el cálculo se define el plano en $y = 1$; el vector normal al plano, $\overrightarrow{k} = (0, 1, 0)$; el vector director, $\overrightarrow{V_r} = (cadera_x, cadera_y, cadera_z) - (rodilla_x, rodilla_y, rodilla_z)$. Finalmente, el ángulo entre el

segmento “muslo” y el plano es:

$$sen(\alpha) = \frac{|\overrightarrow{V_r} \cdot \overrightarrow{k}|}{|\overrightarrow{V_r}| \cdot |\overrightarrow{k}|} = \frac{|\overrightarrow{V_x} \cdot \overrightarrow{k_x} + \overrightarrow{V_y} \cdot \overrightarrow{k_y} + \overrightarrow{V_z} \cdot \overrightarrow{k_z}|}{\sqrt{V_x^2 + V_y^2 + V_z^2} \cdot \sqrt{k_x^2 + k_y^2 + k_z^2}} \tag{4.7}$$



(a) ITOP (b) UTK

Figura 4.58: Ángulo de caída de la cadera.

La desviación estándar del ángulo de rotación de la cadera derecha e izquierda en ITOP es de 4,6^o y 4,84^o respectivamente. En UTK es de 7,59^o en la cadera derecha y 5,01^o en la cadera izquierda. Se puede comprobar en la figura 4.58. La pelvis suele rotar 4^o hacia delante y otros 4^o hacia atrás [104], por lo que los valores anteriores se encuentran dentro de la norma.

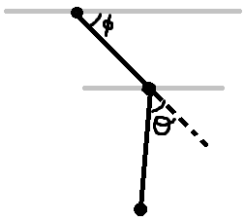
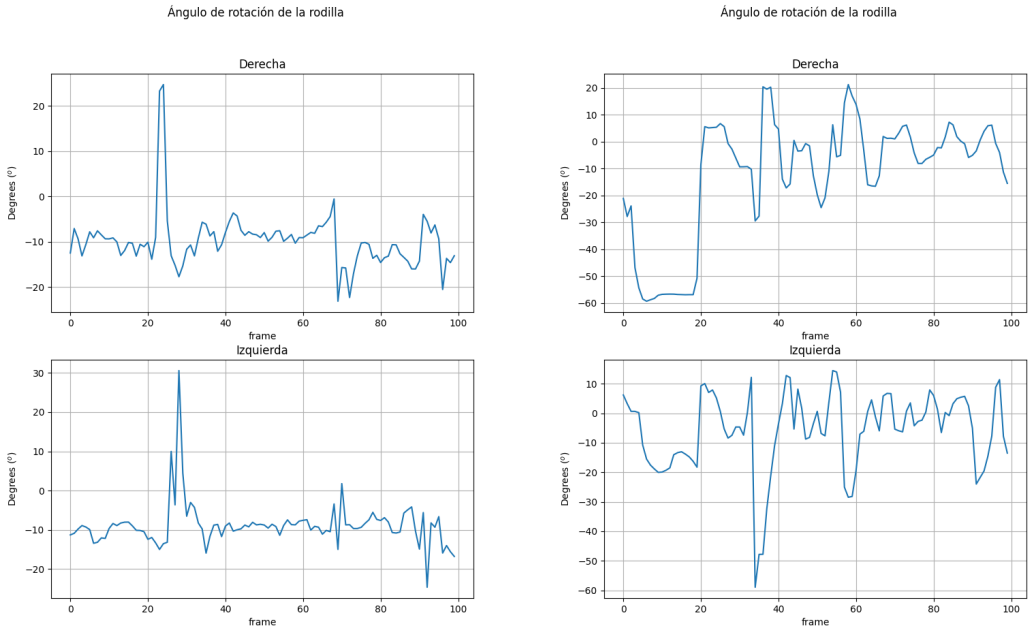


Figura 4.59: Esquema ángulos de rotación durante la marcha.

En la figura 4.59 se muestra un pequeño esquema de los ángulos de rotación de la cadera y la rodilla (ϕ y θ), junto dichas articulaciones (dibujadas con círculos) y los planos (en gris) utilizados para calcular los ángulos.

4.6.2.2 Rotación de la rodilla

La rotación de la rodilla se estima como el ángulo que forma el segmento “pierna” con el plano $y = 1$ menos el ángulo de rotación de la cadera. Obteniéndose la figura 4.60.



(a) ITOP (b) UTK

Figura 4.60: Ángulo de rotación de la rodilla.

La rodilla derecha en ITOP tiene una desviación típica de rotación de 6,15^o en la rodilla derecha y 5,68^o en la izquierda. En el caso de UTK la rotación es de 22,44^o en la derecha y 13,31^o en la izquierda. Se puede comprobar en la figura 4.60. Durante una marcha estándar, el ángulo puede variar hasta los 35^o [104], por lo que es normal que la persona en UTK tenga valores altos, a diferencia de la de ITOP, ya que en esta última no llega a caminar.

4.6.2.3 Caída de la cadera

Durante la marcha, cuando se está sobre una pierna, es muy común que la pelvis descienda hacia el lado contrario (ver figura 4.61). Es la forma natural de mantener el equilibrio para después poder impulsarse hacia delante. Sin embargo, la caída excesiva de la pelvis es un problema. Cuando la pelvis desciende, el centro de masa es arrastrado hacia el mismo lado, por lo que el tronco se inclina naturalmente hacia el lado opuesto para evitar la caída, lo cual suele provocar la rotación de la rodilla hacia dentro, pudiendo provocar una lesión [107]. Esta caída puede observarse tanto desde la vista frontal como desde la posterior. El rango óptimo para los hombres es menor de 5^o y para las mujeres menor de 7^o.

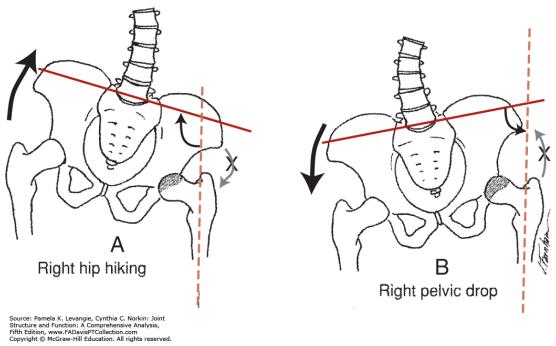
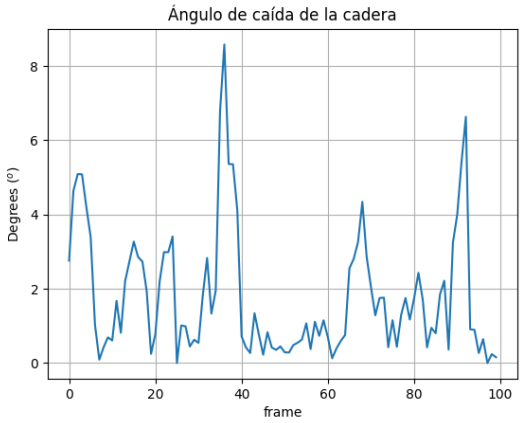
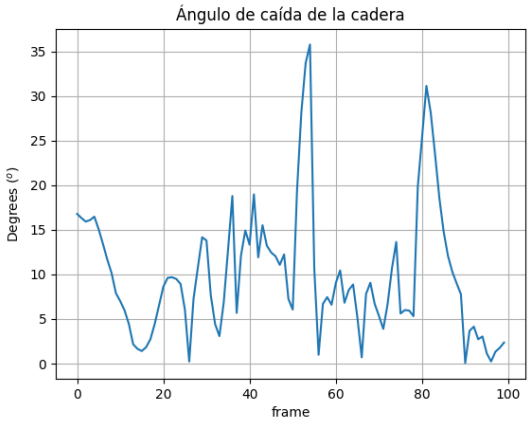


Figura 4.61: Esquema caída de la cadera [108].

El ángulo de caída de la pelvis se mide como el ángulo con respecto al plano horizontal.



(a) ITOP



(b) UTK

Figura 4.62: Ángulo de caída de la cadera.

La desviación típica de la caída de la pelvis en ITOP es de 1,71^o y en UTK 7,30^o. Observable en la figura 4.62. Esto nos dice que el hombre de ITOP tiene la pelvis muy equilibrada y que la mujer de UTK la tiene ligeramente superior al ángulo óptimo, pero no supone ningún problema.

En base a todos los análisis realizados a lo largo de esta sección, se puede afirmar que las dos personas estudiadas, una de cada base de datos, no tienen ninguna discapacidad, al menos en el tren inferior.

Capítulo 5

Conclusiones y líneas futuras

En este último capítulo se resumen las conclusiones obtenidas y se proponen futuras líneas de investigación que se deriven del trabajo.

5.1 Conclusiones

En el presente Trabajo de Fin de Máster se ha realizado un estudio del aprendizaje automático y las redes neuronales desde un punto de vista general, y se ha profundizado en la redes neuronales convolucionales, ya que son capaces de capturar las dependencias espaciales y temporales de los datos, por ello son las que más se utilizan en el procesamiento de imágenes. Este estudio teórico previo se ha realizado para poder entender adecuadamente todos los proyectos investigados basados en arquitecturas con CNNs. Se ha realizado una amplia búsqueda de sistemas para poder escoger una red ya creada que fuese capaz de estimar las posiciones de las articulaciones del cuerpo humano y de las manos a partir de imágenes de profundidad. También se han buscado numerosas bases de datos con grabaciones de profundidad de personas realizando actividades de la vida diaria, habiendo encontrado una base de datos que cumplía con todos los requisitos: contenía grabaciones de profundidad y RGB, las acciones eran ADLs y además de participar gente joven, también participa personas de edad avanzada.

Después de todo el análisis del estado del arte relacionado con las redes neuronales, se realizaron una serie de experimentos probando y entrenando la red con diferentes bases de datos. Al utilizar una de las bases de datos en el entrenamiento, se fueron modificando los parámetros empleados de uno en uno mientras los demás de mantenían iguales, así se pudo observar el efecto que cada uno de ellos provocaba. A pesar de que durante gran parte de los experimentos el sistema no ha funcionado adecuadamente, al final se pudo solucionar el problema y realizar algún experimento más, aunque los resultados conseguidos no son igual de buenos que los que se esperaban. Gracias a ello se ha podido indagar en las redes convolucionales y sus distintos enfoques de entrenamiento y configuraciones.

Por otra parte, se ha hecho un estudio teórico profundo de los diferentes métodos que tienen los terapeutas ocupacionales de determinar el grado de discapacidad o de destreza que tienen las personas en la realización de las tareas de la vida cotidiana, como pueden ser: comer, beber, lavarse los dientes, ponerse los zapatos, etc. Centrándose en los umbrales establecidos de diferentes parámetros, como ángulos de giro, velocidades, etc. Además, se ha creado un algoritmo que aporta parámetros de destreza que podrían ser de gran utilidad en la asignación del nivel de habilidad de las personas. Asimismo aporta conocimiento sobre el estado de equilibrio de la persona y si ésta ha sufrido una caída.

5.2 Líneas futuras

Este trabajo tiene un gran margen de ampliación, por lo que se van a plantear una serie de líneas futuras que se pueden llevar a cabo:

- Analizar en profundidad los sistemas de entrenamiento utilizados y las adaptaciones realizadas, para conseguir buenos resultados al aplicarlas a nuevas bases de datos, fundamentalmente ETRI, y la que se grabará en el proyecto EYEFUL.
- Seguidamente se podría investigar el uso de otros sistema de detección de objetos automático como FastRCNN, que como se comentó, tiene mayor precisión que YOLO. A parte de este tipo de sistemas, se podría estudiar la posibilidad de utilizar la segmentación por instancias, que es una forma especial de segmentación de imágenes que se ocupa de detectar instancias de objetos y delimitar sus fronteras en las imágenes [109], lo cual nos podría ser muy útil para precisar el *bounding box* de la persona con mayor exactitud.
- Seguir buscando alternativas de sistemas de estimación de la pose humana con imágenes RGB, Depth y RGBD.
- Respecto a los parámetros de estimación de la destreza de las personas, por el momento sólo se ha generado el algoritmo para el tren inferior del cuerpo humano, el siguiente paso sería conseguir más métricas de destreza relacionadas con el tren superior, en el cual también habría que incluir la interacción de las manos y brazos con los objetos. Igualmente se podría seguir ampliando los parámetros de destreza útiles del tren inferior del cuerpo.
- Ampliar el estudio de la estimación 3D del CoM, ya que actualmente se está estimando en 2D.
- Cuando se consiga una buena estimación de la localización de las articulaciones del cuerpo, habrá que realizar un estudio detallado de esquemas de determinación de la funcionalidad a partir de parámetros factibles de posición, movimiento, secuencia y planificación, mediante la coordinación estrecha con el grupo de terapeutas ocupacionales y médicos implicados en el proyecto EYEFUL.

Bibliografía

[1] M. Wallace and M. Shelkey, "Katz Index of Independence in Activities of Daily Living (ADL)."

[2] Mariana López Ortega, "Limitación funcional y discapacidad: conceptos, medición y diagnóstico. Una introducción a la situación en México," http://inger.gob.mx/pluginfile.php/1682/mod_resource/content/10/Repositorio_Cursos/Archivos/Cuidamhe/MODULO_I/UNIDAD_2/Limitaci%C3%B3n.pdf [Último acceso 02/febrero/2022].

[3] "Índice de Barthel," https://infogerontologia.com/documents/vgi/escalas/indice_barthel.pdf [Último acceso 02/febrero/2022].

[4] "Escala de Rosow y Breslau," https://www.bmj.com/content/suppl/2013/07/23/bmj.f4240.DC1/artf010761.ww1_default.pdf [Último acceso 02/febrero/2022].

[5] "Evaluación de Terapia Ocupacional. Evaluación de las Habilidades Motoras y de Procesamiento (AMPS)," https://vsip.info/evaluacion-de-habilidades-motoras-y-procesamiento-amps-4-pdf-free.html [Último acceso 07/febrero/2022].

[6] "Evaluación de las Habilidades Motoras y de Procesamiento (AMPS)," http://www.terapia-ocupacional.com/articulos/AMPS.shtml [Último acceso 07/febrero/2022].

[7] "Información del proyecto EYEFUL," https://www.geintra-uah.org/eyeful/es/informaci%C3%B3n [Último acceso 02/febrero/2022].

[8] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep High-Resolution Representation Learning for Human Pose Estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[9] J. Martinez, R. Hossain, J. Romero, and J. J. Little, "A Simple yet Effective Baseline for 3D Human Pose Estimation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[10] "(GitHub) Deep High-Resolution Representation Learning for Human Pose Estimation (CVPR 2019)," https://github.com/leoxiaobin/deep-high-resolution-net.pytorch [Último acceso 02/febrero/2022].

[11] "(GitHub) A Simple yet Effective Baseline for 3D Human Pose Estimation," https://github.com/una-dinosauria/3d-pose-baseline [Último acceso 02/febrero/2022].

[12] M. Contributors, "OpenMMLab Pose Estimation Toolbox and Benchmark," https://github.com/open-mmlab/mmpose, 2020.

[13] B. Ni, G. Wang, and P. Moulin, "Rgbd-hudaact: A color-depth video database for human daily activity recognition," in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 2011, pp. 1147–1153.

[14] P. A. Soler, S. A. A. Silguero, and J. L. N. López, "Valoración funcional en el anciano," http://inger.gob.mx/pluginfile.php/1682/mod_resource/content/19/Repositorio_Cursos/Archivos/Alzheimer/MODULO_II/UNIDAD_3/Valoracion.pdf [Último acceso 06/septiembre/2022].

[15] "Biomecánica," https://www.fisioterapia-online.com/glosario/biomecanica [Último acceso 24/junio/2022].

[16] V. M. Soto and M. Gutiérrez, "Parámetros inerciales para el modelado biomecánico del cuerpo humano," *Revista Motricidad*, vol. 2, pp. 169–189, 1996. [Online]. Available: https://dialnet.unirioja.es/descarga/articulo/2278423.pdf

[17] A. Cappozzo and N. Berme, *Biomechanics of Human Movement: Applications in Rehabilitation, Sports and Ergonomics*. Bertec Corporation, 1990. [Online]. Available: https://books.google.es/books?id=NVRlGQAACAAJ

[18] W. T. Dempster, "The anthropometry of body action," *Dynamic Anthropometry*, vol. 63, no. 4, pp. 435–636, 1955.

[19] P. de Leva, "Joint center longitudinal positions computed from a selected subset of chandler's data," *Journal of Biomechanics*, vol. 29, no. 9, pp. 1231–1233, 1996. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021929096000218

[20] V. M. Zaciorskij, V. Seluyanov, and L. Chugunova, "Methods of determining mass-inertial characteristics of human body segments," *Contemporary Problems of Biomechanics*, pp. 272–291, 1990.

[21] M. Virmavirta and J. Isolehto, "Determining the location of the body?s center of mass for different groups of physically active people," *Journal of Biomechanics*, vol. 47, no. 8, pp. 1909–1913, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021929014002103

[22] J. B. Frutos, J. M. P. Andrés, and J. L. L. Elvira, "Análisis cinemático tridimensional: Aspectos metodológicos," *Motricidad. European Journal of Human Movement*, vol. 29, pp. 75–94, 2012. [Online]. Available: https://www.redalyc.org/articulo.oa?id=274224827006

[23] J. D. Lesmes, *Evaluación clínica-funcional del movimiento corporal humano*. Editorial Médica Internacional, 2007.

[24] C. G. Fábrica, A. Rey, P. V. González, D. Santos, and D. Ferraro, "Evaluación del equilibrio durante la marcha a velocidad autoseleccionada en jóvenes saludables, adultos mayores no caedores y adultos mayores con alto riesgo de caídas," *Revista Médica del Uruguay*, vol. 27, pp. 147 – 154, 09 2011. [Online]. Available: http://www.scielo.edu.uy/scielo.php?script=sci_arttext&pid=S1688-03902011000300004&nrm=iso

[25] C. Li, Q. Zhong, D. Xie, and S. Pu, "Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, ser. IJCAI'18. AAAI Press, 2018, pp. 786–792.

[26] J. Jang, D. Kim, C. Park, M. Jang, J. Lee, and J. Kim, "ETRI-activity3d: A large-scale RGB-d dataset for robots to recognize daily activities of the elderly," in *2020 IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS). IEEE, oct 2020. [Online]. Available: https://doi.org/10.1109%2Firos45743.2020.9341160

[27] P. Zhang, C. Lan, W. Zeng, J. Xing, J. Xue, and N. Zheng, "Semantics-guided neural networks for efficient skeleton-based human action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[28] P. Zhang, C. Lan, J. Xing, W. Zeng, J. Xue, and N. Zheng, "View adaptive neural networks for high performance skeleton-based human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[29] S. ke, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," 02 2019.

[30] J. Martinez, R. Hossain, J. Romero, and J. J. Little, "A simple yet effective baseline for 3d human pose estimation," in *ICCV*, 2017.

[31] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *CVPR*, 2019.

[32] K. Iskakov, E. Burkov, V. Lempitsky, and Y. Malkov, "Learnable triangulation of human pose," in *International Conference on Computer Vision (ICCV)*, 2019.

[33] F. Xiong, B. Zhang, Y. Xiao, Z. Cao, T. Yu, J. Zhou Tianyi, and J. Yuan, "A2j: Anchor-to-joint regression network for 3d articulated pose estimation from a single depth image," in *Proceedings of the IEEE Conference on International Conference on Computer Vision (ICCV)*, 2019.

[34] G. Moon, J. Chang, and K. M. Lee, "V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[36] B. Dehbandi, A. Barachant, D. Harary, J. D. Long, K. Z. Tsagaris, S. J. Bumanlag, V. He, and D. Putrino, "Using data from the microsoft kinect 2 to quantify upper limb behavior: A feasibility study," *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 5, pp. 1386–1392, 2017.

[37] Ó. G. Hernández, V. Morell, J. L. Ramon, and C. A. Jara, "Human pose detection for robotic-assisted and rehabilitation environments," *Applied Sciences*, vol. 11, no. 9, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/9/4183

[38] K. Zhao, C. Guo, H. Bian, J. Yu, H. Wen, T. Wang, and Z. Zhang, "Upper extremity kinematic parameters: Reference ranges based on kinect v2," in *2021 27th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, 2021, pp. 30–35.

[39] I. A. Mesquita, P. F. P. da Fonseca, M. Borgonovo-Santos, E. Ribeiro, A. R. V. Pinheiro, M. V. Correia, and C. Silva, "Comparison of upper limb kinematics in two activities of daily living with different handling requirements," *Human Movement Science*, vol. 72, p. 102632, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167945720300361

[40] P. Lee, T.-B. Chen, C.-Y. Wang, S.-Y. Hsu, and C.-H. Liu, "Detection of postural control in young and elderly adults using deep and machine learning methods with joint-node plots," *Sensors*, vol. 21, no. 9, 2021.

[41] B. Dehbandi, A. Barachant, A. H. Smeragliuolo, J. D. Long, S. J. Bumanlag, V. He, A. Lampe, and D. Putrino, "Using data from the microsoft kinect 2 to determine postural stability in healthy subjects: A feasibility trial," *PLOS ONE*, vol. 12, no. 2, pp. 1–19, 02 2017. [Online]. Available: https://doi.org/10.1371/journal.pone.0170890

[42] E. Gianaria, M. Grangetto, M. Roppolo, A. Mulasso, and E. Rabaglietti, "Kinect-based gait analysis for automatic frailty syndrome assessment," in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 1314–1318.

[43] M. Hayashibe, A. González, and M. Tournier, "Personalized balance and fall risk visualization with kinect two," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2020, pp. 4863–4866.

[44] O. Mazumder, K. Chakravarty, D. Chatterjee, A. Sinha, and A. Das, "Posturography stability score generation for stroke patient using kinect: Fuzzy based approach," in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2017, pp. 3052–3056.

[45] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "Ntu rgb+d: A large scale dataset for 3d human activity analysis," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1010–1019, 2016.

[46] J. Liu, A. Shahroudy, M. Perez, G. Wang, L.-Y. Duan, and A. C. Kot, "Ntu rgb+d 120: A large-scale benchmark for 3d human activity understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 10, pp. 2684–2701, 2020. [Online]. Available: https://doi.org/10.1109/TPAMI.2019.2916873

[47] L. Xia, C. Chen, and J. Aggarwal, "View invariant human action recognition using histograms of 3d joints," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*. IEEE, 2012, pp. 20–27.

[48] C. S. Catalin Ionescu, Fuxin Li, "Latent structured models for human pose estimation," in *International Conference on Computer Vision*, 2011.

[49] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1325–1339, jul 2014.

[50] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining actionlet ensemble for action recognition with depth cameras." IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), June 2012, pp. 1290–1297.

[51] ——, "MSRDaily Activity3D (MSR Daily Activity 3D Dataset)," https://sites.google.com/view/wanqingli/data-sets/msr-dailyactivity3d [Último acceso 13/julio/2022].

[52] J. Wang and X. Nie, "Northwestern-UCLA Multiview Action 3D Dataset)," https://wangjiangb.github.io/my_data.html [Último acceso 13/julio/2022].

[53] C. Chen, R. Jafari, and N. Kehtarnavaz, "Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor," 09 2015.

[54] S. Gasparrini, E. Cippitelli, E. Gambi, S. Spinsante, J. Wåhslén, I. Orhan, and T. Lindh, "Proposal and experimental evaluation of fall detection solution based on wearable and depth data fusion," in *ICT Innovations 2015*, ser. Advances in Intelligent Systems and Computing,

S. Loshkovska and S. Koceski, Eds. Springer International Publishing, 2016, vol. 399, pp. 99–108. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-25733-4_11

[55] L. Chunhui, H. Yueyu, L. Yanghao, S. Sijie, and L. Jiaying, "Pku-mmd: A large scale benchmark for continuous multi-modal human action understanding," *ACM Multimedia workshop*, 2017.

[56] ——, "(Dataset) PKU-MMD: A Large Scale Benchmark for Continuous Multi-Modal Human Action Understanding," http://39.96.165.147/Projects/PKUMMD/PKU-MMD.html [Último acceso 17/julio/2022].

[57] A. Haque, B. Peng, Z. Luo, A. Alahi, S. Yeung, and L. Fei-Fei, "Itop dataset," Oct. 2016. [Online]. Available: https://doi.org/10.5281/zenodo.3932973

[58] ——, "Towards Viewpoint Invariant 3D Human Pose Estimation," in *European Conference on Computer Vision*, October 2016.

[59] J. Hu, W. Zheng, J. Lai, and J. Zhang, "Jointly learning heterogeneous features for rgb-d activity recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 11, pp. 2186–2200, 2017.

[60] J. Patterson and A. Gibson, *Deep Learning: A Practitioner?s Approach.* O'Reilly Media, Inc., 2017.

[61] J. Carlos, J. Jauregui, O. Eduardo, G. Guerrero, and J. Diaz Rodriguez, "Desarrollo de un modelo de regresión con redes neuronales artificiales para estimar la resistencia rotórica de un motor de inducción," 07 2022.

[62] José David Villanueva García, "Introducción a redes neuronales," 10 2020, https://www.iartificial.net/redes-neuronales-desde-cero-i-introduccion/ [Último acceso 19/07/2022].

[63] L. Manjarrez, "Relaciones neuronales para determinar la atenuación del valor de la aceleración máxima en superficie de sitios en roca para zonas de subducción," Ph.D. dissertation, 06 2014.

[64] W. Di, A. Bhardwaj, and J. Wei, *Deep Learning Essentials.*

[65] "Cross-Entropy Loss," https://wandb.ai/sauravmaheshkar/cross-entropy/reports/What-Is-Cross-Entropy-Loss-A-Tutorial-With-Code--VmlldzoxMDA5NTMx [Último acceso 25/08/2022].

[66] H. Farnaz, S. Asadollah, and B. Peyman, "Adaptahead optimization algorithm for learning deep cnn applied to mri segmentation," *Journal of Digital Imaging*, vol. 32, no. 1, p. 105?115, Feb. 2019. [Online]. Available: https://doi.org/10.1007/s10278-018-0107-6

[67] "Various Optimization Algorithms For Training Neural Network," https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6 [Último acceso 06/09/2022].

[68] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[69] S. Ruder, "An overview of gradient descent optimization algorithms," 2016, https://ruder.io/optimizing-gradient-descent/ [Último acceso 06/09/2022].

[70] "Claves para entender la visión III: el cerebro," 02 2016, https://saludocular.com/tag/corteza-visual/ [Último acceso 25/07/2022].

[71] "Convolutional Neural Networks for Visual Recognition," https://cs231n.github.io/convolutional-networks/ [Último acceso 25/07/2022].

[72] "A Comprehensive Guide to Convolutional Neural Networks ? the ELI5 way," 2018, https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 [Último acceso 25/07/2022].

[73] "Max Pooling," https://paperswithcode.com/method/max-pooling [Último acceso 10/08/2022].

[74] "Average Pooling," https://paperswithcode.com/method/average-pooling [Último acceso 10/08/2022].

[75] "Different Types of CNN Architectures," https://vitalflux.com/different-types-of-cnn-architectures-explained-examples/ [Último acceso 11/08/2022].

[76] Y. le Cun, "Generalization and network design strategies," Departmen of Computer Science, University of Toronto, Tech. Rep., 1989.

[77] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[78] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *CoRR*, vol. abs/1311.2901, 2013. [Online]. Available: http://arxiv.org/abs/1311.2901

[79] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014. [Online]. Available: http://arxiv.org/abs/1409.4842

[80] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[81] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[82] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, 2017.

[83] "Voxelización," https://towardsdatascience.com/how-to-voxelize-meshes-and-point-clouds-in-python-ca94d403f81d [Último acceso 24/08/2022].

[84] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[85] "Dilated Convolution," https://paperswithcode.com/method/dilated-convolution [Último acceso 25/08/2022].

[86] "Random Erasing," https://paperswithcode.com/method/random-erasing [Último acceso 29/08/2022].

[87] "Learning Rate," https://towardsdatascience.com/https-medium-com-dashingaditya-rakhecha-understanding-learn [Último acceso 06/09/2022].

[88] "ITOP Dataset," https://zenodo.org/record/3932973#.YxeJsnZBzIV [Último acceso 06/09/2022].

[89] "(GitHub) A2J," https://github.com/zhangboshen/A2J [Último acceso 06/09/2022].

[90] "Real-Time Object Detection," https://paperswithcode.com/task/real-time-object-detection#datasets [Último acceso 06/09/2022].

[91] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. [Online]. Available: http://arxiv.org/abs/1804.02767

[92] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *CoRR*, vol. abs/1708.02002, 2017. [Online]. Available: http://arxiv.org/abs/1708.02002

[93] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015. [Online]. Available: http://arxiv.org/abs/1506.01497

[94] "YOLO: Real-Time Object Detection," https://pjreddie.com/darknet/yolo/ [Último acceso 06/09/2022].

[95] "Non Maximum Suppression," https://paperswithcode.com/method/non-maximum-suppression [Último acceso 06/09/2022].

[96] "Modelos de Detección de Objetos," https://www.aprendemachinelearning.com/modelos-de-deteccion-de-objetos/ [Último acceso 06/09/2022].

[97] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[98] "OpenCV," https://opencv.org/ [Último acceso 06/09/2022].

[99] luaffjk, "(GitHub) Image-Processing," 2019, https://github.com/luaffjk/Image-Processing/blob/master/detecting.ipynb [Último acceso 06/09/2022].

[100] Sunita Nayak, "Deep Learning based Object Detection using YOLOv3 with OpenCV," 2018, https://learnopencv.com/deep-learning-based-object-detection-using-yolov3-with-opencv-python-c/ [Último acceso 06/09/2022].

[101] Aditya Rakhecha, "Understanding Learning Rate," 2019, https://towardsdatascience.com/https-medium-com-dashingaditya-rakhecha-understanding-learning-rate-dd5da26bb6de [Último acceso 06/09/2022].

[102] J. Muñoz, R. Cassibba, H. Castro, W. Holtz, and A. Muñoz, "Errores en la determinación del centro de gravedad del cuerpo humano mediante el stick figure (2D)," *Anales AFA*, vol. 21, 01 2009.

[103] "Definición de los ejes de una cámara de profundidad," https://external-content.duckduckgo.com/iu/?u=https%3A%2F%2Fi.pinimg.com%2Foriginals%2Fc7%2F80%2F40%2Fc78040f26e8026aef4fe7ac718ff8e20.png&f=1&nofb=1 [Último acceso 06/09/2022].

[104] W. Chen, Z. Jiang, H. Guo, and X. Ni, "Fall detection based on key points of human-skeleton using openpose," *Symmetry*, vol. 12, p. 744, 05 2020.

[105] J. Latorre, R. Llorens, C. Colomer, and M. Alcañiz, "Reliability and comparison of kinect-based methods for estimating spatiotemporal gait parameters of healthy and post-stroke individuals," *Journal of Biomechanics*, vol. 72, pp. 268–273, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002192901830160X

[106] D. Cunado, M. S. Nixon, and J. N. Carter, "Automatic extraction and description of human gait models for recognition purposes," *Computer Vision and Image Understanding*, vol. 90, no. 1, pp. 1–41, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314203000080

[107] "Pelvic Drop," 2021, https://geeksonfeet.com/run/pelvicdrop/ [Último acceso 06/09/2022].

[108] P. K. Levangie and C. C. Norkin, *Joint Structure and Function: A Comprehensive Analysis*. F.A. Davis Company, 2011.

[109] H. Bandyopadhyay, "Instance Segmentation," 2022, https://www.v7labs.com/blog/instance-segmentation-guide#:~:text=Segmentation%20refers%20to%20the%20task,objects%20and%20demarcating%20their%20boundaries. [Último acceso 06/09/2022].

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR

Universidad
de Alcalá