UNIVERSIDAD DE CHILE
FACULTAD DE MEDICINA
ESCUELA DE POSTGRADO

CONTEO AUTOMÁTICO DE ACCIONES TÉCNICAS DINÁMICAS MEDIANTE VISIÓN COMPUTACIONAL EN LA EVALUACIÓN DEL RIESGO BIOMECÁNICO DE TRASTORNOS MUSCULOESQUELÉTICOS RELACIONADOS AL TRABAJO DE EXTREMIDAD SUPERIOR

DIEGO JOSUE VICENTE CONTRERAS

TESIS PARA OPTAR AL GRADO DE MAGISTER EN INFORMÁTICA MÉDICA

Director de Tesis: Dr.rer.nat. Victor Castañeda Zeman
Co-Director de Tesis: Prof.Dr. Steffen Härtel

2023

Dedico este trabajo a mis padres, por su apoyo, amor y paciencia.
A mis abuelas y mi tata, por siempre en mi corazón.
A mis hermanos, sobrinos, mis tíos y mis primos.

Agradecimientos

Quiero agradecer a toda mi familia, por su apoyo incondicional, por su comprensión y por permitirme dedicar todo el tiempo necesario a este trabajo.

Quiero dar un especial agradecimiento al Prof.Dr. Steffen Härtel, director del Magíster en Informática Médica (MIM), por sostener y transmitir valores que orientan hacia la excelencia. Por su valiosísima crítica y, por invitarme a participar del Espacio Tesistas MIM, lugar donde este trabajo fue enriquecido de manera importante.

Agradezco al Servicio Alemán de Intercambio Académico, en alemán, Deutscher Akademischer Austauschdienst (DAAD), por otorgarme una beca que permitió realizar una pasantía en la Universidad de Heilbronn (Hochschule Heilbronn). En dicho contexto, agradezco al Prof. Dr.-Ing. Gerrit Meixner por su generosa invitación a trabajar en el Usability and Interaction Technology Laboratory (UniTyLab), lugar donde desarrollamos una investigación junto a Mario Schwarz, a quien agradezco su trabajo, la cual culminó en una publicación científica[1].

Agradezco a Janice Kerr, mi jefatura directa en el Instituto de Seguridad del Trabajo, por apoyarme y comprender mi necesidad de tiempo. Al Dr. Guillermo Farmer y Paulina Cuadra por su confianza al extender recomendaciones en mi favor.

Agradezco a mi profesor director de tesis: Dr.rer.nat. Victor Castañeda, por su guía y orientación técnica en cuanto al análisis llevado a cabo en este trabajo y la estructura del documento.

Por último, quiero agradecer a los amigos que aportaron el consejo técnico, las palabras de apoyo y alegría en este período, en particular, a Cristian Ruz, Camila Less, Waldo Altermatt, Gustavo Sánchez, Pamela Muñoz, Paulina Caro y Cristian Vergara.

[1] Artículo disponible en: https://link.springer.com/chapter/10.1007/978-3-031-35741-1_20

Índice

Índice de figuras

Índice de tablas

Índice de Código

Glosario de Términos

Acciones Técnicas: «Acciones manuales elementales que son necesarias para cumplir las funciones dentro del ciclo de trabajo. Esta acción implica una actividad de las extremidades superiores, que debe ser identificada como un conjunto de movimientos, de uno o varios segmentos articulares, que permiten realizar una operación laboral simple. No debe ser identificada sólo con el movimiento articular en sí, sino con el conjunto de movimientos, de uno o varios segmentos articulares, que permiten realizar una operación laboral simple». La acción técnica puede ser estática (postura fija, por ejemplo al sostener un objeto en la mano) o dinámica (asociada a movimiento).

Acciones Técnicas Dinámicas («ATD»): Son aquellas que están determinadas por la constante movilidad de la extremidad superior.

Conteo Automático de Acciones Técnicas Dinámicas: Número de acciones técnicas por minuto en cada extremidad, en cada video, como resultado de utilizar el contador automático propuesto por Taborri et al., adaptado en esta tesis para su utilización con visión computacional.

Conteo Humano de Consenso ($ConteoH_{Consenso}$): Número de acciones técnicas por minuto en cada extremidad en cada video, como resultado del promedio obtenido entre los evaluadores participantes especialistas en ergonomía.

Factor Frecuencia (FF): Valor obtenido mediante el número de acciones técnicas por minuto en el ciclo de trabajo, teniendo en cuenta la posibilidad de breves interrupciones. Es un factor sumando para obtener el nivel de riesgo OCRA Checklist.

Occupational Repetitive Actions Checklist (OCRA Checklist): método de primer nivel para la evaluación del riesgo por movimientos repetitivos de las extremidades superiores.

Trastrornos Musculoesqueléticos Relacionados al Trabajo (TMERT): Si el trastorno musculoesquelético se relaciona con factores de riesgo presentes en el trabajo, se habla de trastornos musculoesquelético relacionados al trabajo, y en tal caso, surgen como resultado de una sobrecarga mecánica sobre una parte específica del sistema musculoesquelético capaz de provocar un trauma acumulado en el tiempo, asociado a descansos insuficientes para una recuperación fisiológica, o bien, por una sobrecarga puntual que supera la capacidad de resistencia del tejido provocando una lesión

Trastornos Musculoesqueléticos: Amplio rango de condiciones inflamatorias y degenerativas que afectan músculos, tendones, ligamentos, articulaciones, nervios periféricos y vasos sanguíneos. Pueden surgir como resultado de enfermedades al sistema musculoesquelético, de actividades deportivas o del trabajo.

Umbral Amplitud (UA): Parámetro de entrada, para configurar el funcionamiento del contador automático de acciones técnicas dinámicas. Representa la distancia mínima, en grados, entre el último valor mínimo y el máximo presente para que éste sea considerado un nuevo máximo válido.

Umbral Temporal (UT): Parámetro de entrada, en segundos, para configurar el funcionamiento del contador automático de acciones técnicas dinámicas. Representa la distancia temporal mínima, entre el último valor máximo valido y el máximo presente para que éste sea considerado un nuevo máximo válido.

1. Resumen

Problema. Para abordar las limitaciones de la observación al evaluar el riesgo biomecánico de Trastornos Musculoesqueléticos Relacionados al Trabajo (TMERT), se utilizan sensores inerciales, de profundidad o visión computacional para capturar o estimar la postura y calcular las variables del movimiento requeridas por el método observacional de ergonomía. Taborri et al. propuso un algoritmo basado en umbrales para automatizar el conteo de «Acciones Técnicas Dinámicas» (ATD) con sensores inerciales y así calcular el «Factor Frecuencia» (FF) del método Occupational Repetitive Actions Checklist (OCRA Checklist). Sin embargo, el uso de sensores inerciales es poco común en el contexto ocupacional, ya que requiere de conocimiento especializado y calibración y además, pueden interferir el movimiento de la persona, alterando los registros y el trabajo. La visión computacional permite estimar las coordenadas articulares en 3D para calcular los ángulos requeridos en la propuesta de Taborri et al. Sin embargo, no se halló literatura sobre el conteo de ATD utilizando visión computacional y, por lo tanto, es importante evaluar propuestas dirigidas a mejorar la reproducibilidad de las evaluaciones del riesgo de TMERT.

Solución. Adaptar el algoritmo de Taborri et al. para el conteo automático de ATD y respectiva determinación del FF a partir de videos, utilizando la solución de visión computacional, Mediapipe Pose para comparar los resultados con un «conteo humano de consenso» ($ConteoH_{Consenso}$), estimando su validez en la base de datos «University of Washington Indoor Object Manipulation Dataset» (en adelante, Dataset).

La hipótesis de este trabajo es que el conteo automático de «acciones técnicas dinámicas», en las distintas modalidades, determinará un «factor frecuencia» similar al «conteo humano de consenso» en el Método Occupational Repetitive Actions Checklist.

Método. Se usaron videos de tareas repetitivas y la anotación de las coordenadas articulares 3D incluidas en el citado Dataset para efectuar el estudio. Tres evaluadores con formación acreditada en el Método OCRA Checklist contaron las ATD en los videos, para definir un $ConteoH_{Consenso}$ con el cual comparar los resultados de los conteos automáticos. Se realizó un análisis comparativo, visual y estadístico, de los ángulos del movimiento anotado en el Dataset con la estimación postural de Mediapipe, mediante la dispersión de las curvas, coeficientes de concordancia de Lin y correlación r de pearson. Se adaptó el algoritmo de Taborri et al. en Python, y se realizaron pruebas para determinar la combinación óptima de umbrales en el conteo de ATD y el FF en cada grupo de datos en base a la comparación con el conteo humano de consenso.

Resultados. El movimiento humano reconstruido a partir de la estimación postural de Mediapipe Pose es más semejante al movimiento humano observado en los videos del Dataset que el movimiento humano reconstruido a partir de la postura anotada en el mismo Dataset. En ambas extremidades, el coeficiente de concordancia de Lin entre las curvas del movimiento de Dataset suavizado y Mediapipe suavizado presentó el siguiente orden decreciente: flexión hombro, abducción hombro, flexión codo, rotación hombro, flexión muñeca. El contador automático de ATD puede configurarse con una combinación de umbrales «óptimos» que ofrecen un error $RMSE_{Conteo}$ inferior a 6 ATD por minuto entre los 20 videos, lo que es inferior a la desviación estándar propia del $ConteoH_{Consenso}$. Además, en la asignación del FF respectivo, el conteo automático arroja un error $RMSE_{frec}$ menor a 0.6 para la extremidad derecha y menor a 1.2 para la extremidad izquierda, que también es inferior a la desviación estándar propia del FF entregado por consenso en cada extremidad. En ambos casos, mediante la prueba U de Mann-Whitney, no se halló diferencia estadísticamente significativa entre los conteos automáticos y la asignación del FF realizados con umbrales óptimos y lo obtenido por consenso de los evaluadores.

Conclusión. Se adaptó para visión computacional un algoritmo diseñado inicialmente para sensores inerciales para realizar el conteo automático de acciones técnicas dinámicas, cuya configuración óptima ofrece, para los grupos

de datos analizados, resultados estadísticamente indiferenciables del $ConteoH_{Consenso}$. La evidencia descrita entrega argumentos suficientes para aceptar la hipótesis planteada. Este trabajo es promisorio en cuanto al desarrollo de soluciones tecnológicas para aumentar la objetividad al momento de evaluar la exposición ocupacional a repetitividad en extremidades superiores y el riesgo de desarrollar trastornos musculoesqueléticos relacionados al trabajo.

2. Abstract

Problem. To address the limitations of human observation in assessing the biomechanical risk of work-related musculoskeletal disorders, inertial sensors, depth sensors, or computer vision are commonly used to capture or estimate posture and calculate the motion variables required by the observational ergonomics method. Taborri et al. proposed an threshold-based algorithm to automate the counting of «dynamic technical actions» using inertial sensors and calculate the «Frequency Factor» of the Occupational Repetitive Actions Checklist method (OCRA Checklist). However, the use of sensors is uncommon in occupational contexts due to the requirement of specialized knowledge and calibration. Moreover, they can interfere with a person's movement, disrupting records and work. Computer vision allows for the estimation of 3D joint coordinates to calculate the angles required in Taborri et al.'s proposal. Literature regarding «dynamic technical actions» counting using computer vision was not found. It is important to assess proposals aimed at improving the reproducibility of biomechanical risk assessment of work-related musculoskeletal disorders.

Solution. Adapt and estimate the validity of an algorithm developed by Taborri et al. for the automatic counting of «dynamic technical actions» and the respective determination of the «frequency factor» from videos, using a computer vision solution, Mediapipe Pose, to compare the results with a «human consensus count», estimating its validity in the «University of Washington Indoor Object Manipulation Dataset».

The hypothesis suggests that the automatic counting of «dynamic technical actions», in different modalities, will determine a «frequency factor» similar to the «human consensus count» in the Occupational Repetitive Actions Checklist method.

Method. Videos of repetitive tasks and the annotation of 3D joint coordinates included in the mentioned dataset were used for the study. Three evaluators with accredited training in the OCRA Checklist Method counted the «dynamic technical actions» in the videos to establish a «human consensus count» for comparison with the results of the automatic counting. A comparative analysis, both visual and statistical, was conducted on the movement angles annotated in the dataset and the postural estimation from Mediapipe, using curve dispersion, Lin's concordance and Pearson's r correlation coefficients. The algorithm created by Taborri et al. was adapted in Python to determine the optimal combination of thresholds for counting «dynamic technical actions» and the «frequency factor» for each data group. The automatic counts of «dynamic technical actions» and «frequency factor» were compared with the «human consensus count».

Results. Human motion reconstructed from Mediapipe Pose postural estimation is more similar to the human motion observed in the Dataset videos than the human motion reconstructed from the annotated posture in the same Dataset. In both extremities, the Lin's concordance coefficient between the smoothed Dataset movement curves and smoothed Mediapipe showed the following decreasing order: shoulder flexion, shoulder abduction, elbow flexion, shoulder rotation, wrist flexion. The automatic «dynamic technical actions» counter can be configured with a combination of «optimal thresholds» that offer an $RMSE_{Conteo}$ error of less than 6 «dynamic technical actions» per minute among the 20 videos, which is lower than the own standard deviation of the human counting consensus. Additionally, in the assignment of the respective Frequency Factor, the automatic count yields an $RMSE_{frec}$ error of less than 0.6 for the right extremity and less than 1.2 for the left extremity, which is also lower than the own standard deviation of the Frequency Factor provided by consensus in each extremity. In both cases, using the Mann-Whitney U test, no statistically significant difference was found between the automatic counts and Frequency Factor assignment performed with optimal thresholds and what was obtained by consensus of the evaluators.

Conclusion. An algorithm initially designed for inertial sensors was adapted for computer vision to enable auto-

matic counting of «dynamic technical actions». Its optimal configuration yielded results statistically indistinguishable from the «human consensus count» for the analyzed data groups. This work shows promising for the development of technological solutions to enhance objectivity in evaluating occupational exposure to upper limb repetitiveness and the risk of developing work-related musculoskeletal disorders.

3. Introducción

3.1. Antecedentes

Los Trastornos Musculoesqueléticos (TME) incluyen un amplio rango de condiciones inflamatorias y degenerativas que afectan músculos, tendones, ligamentos, articulaciones, nervios periféricos y vasos sanguíneos [1, 2]. Pueden surgir como resultado de enfermedades al sistema musculoesquelético, de actividades deportivas o del trabajo. Si el TME se relaciona con factores de riesgo presentes en el trabajo, se habla de Trastornos Musculoesquelético Relacionados al Trabajo (TMERT), y en tal caso, surgen como resultado de una sobrecarga mecánica sobre una parte específica del sistema musculoesquelético capaz de provocar un trauma acumulado en el tiempo, asociado a descansos insuficientes para una recuperación fisiológica, o bien, por una sobrecarga puntual que supera la capacidad de resistencia del tejido provocando una lesión [3]. Entre los TMERT se incluyen tenosinovitis, tendinitis y/o tendinosis, como el síndrome del manguito rotador, epicondilitis lateral, epicondilitis medial, síndrome del túnel carpiano, tenosinovitis De Quervain, dedo en gatillo, entre otros. Se distinguen de las enfermedades profesionales musculoesqueléticas donde hay un diagnóstico médico de patología musculoesquelética y, en Chile, se determina una relación causal directa con el trabajo[2,3][4-6].

En la Unión Europea y en Estados Unidos (EE.UU.) son el problema de salud más común relacionado al trabajo con una incidencia anual de un 40 % a 30 %, respectivamente [7, 8]. En Chile, fueron la causa más común de denuncias de enfermedades profesionales entre el 2017 y el 2019, entre un 45 % y un 43 % del total, cuya proporción respecto del total se altera desde el año 2020, por la aparición del COVID-19 (ver Figura 1), alcanzando 8 % en 2020 y 2021, y un 10 % en el año 2022. Sin embargo la cantidad total de denuncias de enfermedad profesional musculoesquelética no se redujo, pues se mantuvo entre las 16 mil y 20 mil por año [6].

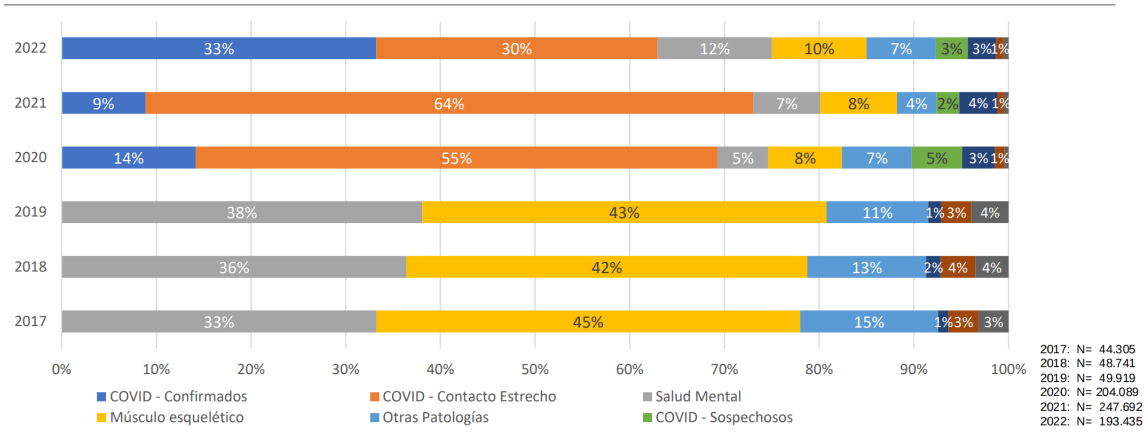


Figura 1: Distribución de los diagnósticos del total de denuncias individuales de enfermedades profesionales, Mutualidades e ISL (2017 - 2022). Extraído de [6].
Previo al período de pandemia por COVID, los TMERT fueron la principal causa de denuncias de enfermedades profesionales en Chile. En relación al total, se calcula que cada años se han realizado alrededor de 20 mil denuncias, sin mayor diferencia antes y después de la pandemia.

[2] Artículo 7° de la Ley 16.744: Es enfermedad profesional la causada de una manera directa por el ejercicio de la profesión o el trabajo que realice una persona y que le produzca incapacidad o muerte.

[3] Ante una denuncia de una presunta enfermedad profesional, un comité de calificación de las mutualidades, administradores de la Ley 16.744, dictamina el origen médico-legal de la patología, en base a protocolos de evaluación de la exposición ocupacional a factores de riesgo de TMERT, establecidos por la Superintendencia de Seguridad Social.

En EE.UU. se estimó un costo directo[4] anual de USD $13 billones, mientras que el costo total (directo + indirecto[5]) se calculó en USD $100 billones en 1996 [9]. En países nórdicos, se estimó un costo directo entre un 2,7 % y un 5,2 % del Producto Interno Bruto, y entre un 15,2 % y 22,2 % en 2003, si se incluyen todos los costos asociados [9]. Por lo tanto, los TMERT constituyen un costo enorme para las economías y toda la sociedad en su conjunto.

Entre los rubros con mayor trabajo repetitivo destaca la industria y servicios, al procesar alimentos, por ejemplo, al despinar peces o desconchar; al cosechar o seleccionar frutas y hortalizas; al deshuesar y despostar carne; al reparar equipos y maquinarias; en labores de aseo y limpieza [10]. El comercio electrónico puede llevar a altas demandas al sector, y en pequeñas y medianas empresas, donde las promociones de temporada tipo «Cyber» llevan a preparar pedidos a una alta velocidad en los centros de distribución, donde se contrata a un alto número de trabajadores transitorios sin un adecuado entrenamiento motor en las tareas a desempeñar, lo que plantea desafíos para proteger la salud, haciendo que la definición de un «trabajo seguro» sea aún más compleja.

3.1.1. Factores de riesgo de TMERT

Entre los determinantes causales de los TMERT, existen varias clases de factores de riesgo, entre los que destacan los biomecánicos, tales como la fuerza requerida, la repetitividad, posturas anatómicamente no neutras, las posturas estáticas mantenidas y las fuerzas compresivas externas. También son importantes los factores de riesgo psicosocial y organizacional[6]. Además, son importantes los factores de riesgo ambientales, como el frío o la vibración. Y por último, también destacan los factores de riesgo extra laborales y personales, tales como la edad, historia clínica previa, capacidad física, género, actividades deportivas y recreativas, hábitos posturales inadecuados, entre otros [1, 2, 9].

La repetitividad como factor de riesgo de TMERT. La repetitividad es uno de los factores de riesgo biomecánico más relevantes. De ahí que, en la denominación de muchos TMERT se utilizó el término «lesiones por esfuerzos repetitivos», en ingles, «repetitive strain injuries», síndromes por sobreuso, entre otros. La Tabla 1 muestra la evidencia de relación causal en los TMERT más frecuentes y la presencia de repetitividad en el trabajo.

[4] El costo directo de los TMERT incluye la pérdida funcional, a veces progresiva y de larga rehabilitación, gasto de bolsillo, disminución de capacidad de ganancia.

[5] El costo indirecto incluye la atención de salud, contratación y formación de reemplazos en el trabajo, interrupción de procesos productivos, costo judicial e investigaciones asociadas, disminución de la calidad de los productos y posición en el mercado [2, 9].

[6] Condiciones de empleo, duración de las tareas y jornadas, ritmo impuesto por líneas de producción, sistemas de remuneración (trabajo a trato o por producción, etc.), baja capacidad de desición, entre otros. Trabajadores expuestos en simultáneo a factores de riesgo físico y psicosocial / organizacional poseen mayor probabilidad de reportar síntomas comparado a una exposición por separado [9].

Segmento Corporal Afectado	Evidencia de no efecto (-)	Evidencia insuficiente (+/0)	Evidencia (++)	Evidencia fuerte (+++)
Codo		✓		
Cuello y Cuello / Hombro			✓	
Hombro			✓	
Mano / Muñeca (Tendinitis)			✓	
Mano / Muñeca (Síndrome del túnel carpiano STC)			✓	
Codo combinado con fuerza y postura				✓
Mano / Muñeca (Tendinitis) combinado con fuerza y postura				✓
Mano / Muñeca (STC) combinado con fuerza y postura				✓

Tabla 1: Evidencia de la relación causal entre repetitividad y trastornos musculoesqueléticos relacionados al trabajo. Adaptado de [1].
Se destacan aquellos segmentos corporales en que la repetitividad combinada con postura y fuerza presentan una evidencia fuerte (+++) de relación causal de TMERT.

En el análisis ergonómico del trabajo, las tareas se deben desglosar secuencialmente y temporalmente en «operaciones[7]», y éstas a su vez, se deben desglosar en «acciones técnicas[8]», las cuales suelen involucrar el movimiento simultáneo de las articulaciones del hombro, codo y muñeca. Además, puede existir un «ciclo fundamental» en alguna de las operaciones que componen la tarea. Con fines explicativos, la Tabla 2 presenta una tarea del rubro agrícola: su ciclo está compuesto por cuatro operaciones, donde se detalla el desglose de acciones técnicas de la operación N°2, la que posee un «ciclo fundamental», pues, dichas acciones técnicas se repiten «más del 50 % de la duración del ciclo» hasta llenar la caja. No se incluyen tiempos dedicados a cada acción técnica, cada operación, cada ciclo y la tarea acumulada en el día, sin embargo, aquello es necesario en estudios de ergonomía.

[7] Conjunto de movimientos necesarios para efectuar una transformación en un producto [11]. También pueden llamarse llamadas «subtareas» o «fases».

[8] «Acciones manuales elementales que son necesarias para cumplir las funciones dentro del ciclo de trabajo. Esta acción implica una actividad de las extremidades superiores, que debe ser identificada como un conjunto de movimientos, de uno o varios segmentos articulares, que permiten realizar una operación laboral simple. No debe ser identificada sólo con el movimiento articular en sí, sino con el conjunto de movimientos, de uno o varios segmentos articulares, que permiten realizar una operación laboral simple». La acción técnica puede ser estática (postura fija, por ejemplo al sostener un objeto en la mano) o dinámica (asociada a movimiento). Extraído de [11].

Tarea	Operación	Acciones Técnicas (extremidad derecha)	Acciones Técnicas (extremidad izquierda)
Llenar cajas con manzanas	1) Abrir una caja de cartón	1.1) ... 1.2) ... 1.3)... 1.n)	1.1) ... 1.2) ... 1.3) ... 1.n)
	2) Ingresar manzanas a la caja	2.1) Tomar manzana (de la cinta transportadora) 2.2) Acercar la manzana (hacia la caja) 2.3) Colocar la manzana (al interior de la caja)	2.1) Tomar manzana (de la cinta transportadora) 2.2) Acercar la manzana (hacia la caja) 2.3) Colocar la manzana (al interior de la caja)
	3) Cerrar caja	3.1) ... 3.2) ... 3.3)... 3.n)	3.1) ... 3.2) ... 3.3)... 3.n) ....
	4) Sellar caja	4.1) ... 4.2) ... 4.3)... 4.n)	4.1) ... 4.2) ... 4.3)... 4.n) ....

Tabla 2: Desglose de una tarea con 4 operaciones y el detalle de acciones técnicas de la operación n°2. La tarea «llenar cajas con manzanas» posee 4 operaciones: abrir una caja de cartón, ingresar manzanas a la caja, cerrar caja y sellar caja. Cada operación posee un grupo de acciones técnicas. Con fines explicativos sólo se detallan las acciones técnicas de la operación N°2: Tomar manzana, Acercar manzana (hacia la caja), colocar la manzana (al interior de la caja). Dado que dichas acciones técnicas se repiten «más del 50 % de la duración del ciclo» hasta llenar la caja. En ergonomía, se considera que la tarea posee un «ciclo fundamental».

La repetitividad está presente en operaciones de «tareas cíclicas», como en el ejemplo de la Tabla 2, en la operación n°2. Sin embargo, también puede estar presente en operaciones puntuales de tareas que no presentan ciclos de trabajo claramente definidos, «tareas acíclicas», donde existen operaciones compuestas por patrones de movimientos similares o movimientos repetitivos que se ejecutan una y otra vez con poca variación en el tiempo. Por ejemplo, un mecánico automotriz realiza la tarea de «desmontar un motor», cuya secuencia de operaciones no sigue un ciclo particular, pero en varias oportunidades realiza la operación «usar un atornillador», la cual se desglosa en secuencias de movimientos manuales elementales llamados «acciones técnicas» que son repetitivas. En este contexto, Barbara Silverstein [12] proporcionó la definición operacional más citada: una tarea es «altamente repetitiva» cuando presenta «ciclos de duración de 30 segundos o menos, o cuando más del 50 % del tiempo de ciclo se emplea realizando un *ciclo fundamental* o el mismo patrón de movimientos».

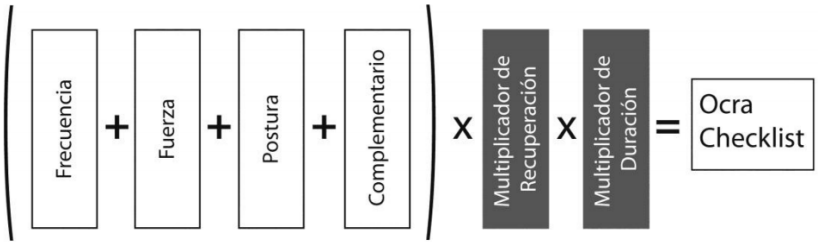
3.1.2. Evaluación del riesgo por movimientos repetitivos

La Norma «ISO 11228-3. Ergonomics - Manual Handling - Part 3: Handling of Low Loads at High Frequency» [13], recomienda utilizar el método Occupational Repetitive Actions, llamado «OCRA Index» para evaluar del riesgo biomecánico asociado al trabajo repetitivo. El método OCRA Index (1998) exige un extenso proceso de observación y análisis de las tareas. Según sus autores, permite «la evaluación precisa del riesgo de forma analítica, aplicado al diseño y rediseño de puestos de trabajo»[11]. Dicho índice se define como la proporción entre el número de acciones técnicas contabilizadas $(nATC)$ y el número de acciones técnicas recomendadas $(nATR)$ para prevenir los TMERT. Los valores de $(nATC)$ y $(nATR)$ deben ser comparados para cada extremidad considerando la duración de la tarea en todo el día.

$$OCRA\ Index = \frac{nATC}{nATR}$$

En el año 2007, con la finalidad de reducir los tiempos de evaluación, se diseñó una versión simplificada del OCRA Index: el «OCRA Checklist» [11], ver Figura 2, el cual permite, según describen sus autores, «la evaluación del riesgo en primer nivel, mediante el mapa de riesgo y la aproximación a la gestión

y disminución del riesgo». Su aplicación implica asignar puntajes individuales a cada factor de riesgo, para ser sumados y, luego, multiplicados con factores de recuperación y duración, y con ello, obtener el puntaje $OCRA_{Checklist}$ (ver Figura 2a), el cual se asocia a un nivel de riesgo biomecánico, permitiendo predecir la incidencia esperada de trabajadores afectados por patologías asociadas a TMERT (Figura 2b). Ambos métodos han demostrado un nivel de correlación importante, con equivalencias en el puntaje final obtenido (Figura 2b).



(a)

CHECKLIST	INDICE OCRA	NIVEL	RIESGO	Previsión de patológicos TME (%)
< 7,5	<2,2	Verde	Riesgo aceptable	< 5,3
7,6 – 11,0	2,3 – 3,5	Amarillo	Riesgo muy leve	5,3 - 8,4
11,1 – 14.0	3,6 - 4,5	Rojo Suave	Riesgo medio leve	8,5- 10,7
14,1 – 22,5	4,6 – 9,0	Rojo	Riesgo medio	10,8- 21,5
≥ 22,6	≥ 9,1	Violeta	Riesgo elevado	>21,5

(b)

Figura 2: Cálculo de Occupational Repetitive Actions Checklist (OCRA Checklist), en Figura 2a y, niveles de riesgo equivalentes con el índice OCRA en Figura 2b.
En la Figura 2a se presenta el cálculo del puntaje que determina el nivel de riesgo en el Método OCRA Checklist. Extraído de [11]. La suma de los puntajes asignados a cada factor de riesgo (Frecuencia, Fuerza, Postura y Complementarios), es multiplicado con los factores de recuperación y duración, para obtener el puntaje OCRA Checklist. A su vez, dicho puntaje se relaciona con un nivel de riesgo, el cual se detalla en la Figura 2b, donde además, se presentan las equivalencias entre el índice OCRA y OCRA Checklist, criterio de clasificación de riesgo e incidencia esperada (%) de los trabajadores afectados con patologías en la extremidad superior. Extraído de [11].

Cuando se aplica el OCRA Checklist los factores de riesgo a evaluar son: «ausencia de tiempo para la recuperación», «frecuencia de movimientos», «fuerza», «posturas forzadas (considerando la estereotipia)» y «factores de riesgo complementarios» tales como «vibración», «ambiente frío inferior a los 0 ℃», «trabajo de precisión», «contragolpes», «uso de guantes inadecuados», y la «duración neta del trabajo repetitivo». Cada factor de riesgo posee una hoja de evaluación que presenta diversas situaciones que ejemplifican los niveles de riesgo. Por lo tanto, los evaluadores deben estimar una equivalencia entre lo observado y lo descrito en la respectiva hoja de evaluación, y para ello, entrevistan y observan el trabajo directamente o mediante videos. Por razones de interés en esta investigación, no se presentan todas las hojas de evaluación sino que sólo el FF (ver Tabla 3), el cual depende del número de acciones técnicas por minuto $(N°acciones_{minuto})$ equivalente a la proporción entre el número de acciones observadas y el tiempo de

observación en minutos [11].

$$N°acciones_{minuto} = \frac{(N°acciones)}{tiempo\ obs(min)}$$

Acciones Técnicas Dinámicas por minuto	A - Factor Frecuencia cuando interrupciones breves SON posibles	B- Factor Frecuencia cuando interrupciones breves NO SON posibles
< 22.5	0.0	0.0
>= 22.5 a < 27.5	0.5	0.5
>= 27.5 a < 32.5	1	1
>= 32.5 a < 37.5	2	2
>= 37.5 a < 42.5	3	4
>= 42.5 a < 47.5	4	5
>= 47.5 a < 52.5	5	6
>= 52.5 a < 57.5	6	7
>= 57.5 a < 62.5	7	8
>= 62.5 a < 67.5	8	9
>= 67.5 a < 72.5	9	10
>= 72.5	9	10

Tabla 3: Asignación del FF para el OCRA Checklist, dependiente del número de acciones técnicas por minuto. Adaptado de Daniela Colombini, Enrico Occhipinti y Enrique Álvarez-Casado en [11].

3.1.3. Algoritmo para el conteo automático de «Acciones Técnicas Dinámicas»

Taborri et al. [14] propuso un algoritmo para contar automáticamente ATD utilizando sensores inerciales y, con ello, obtener el FF del Método OCRA Checklist. El algoritmo utiliza las curvas que describen los ángulos de las articulaciones del hombro, codo y muñeca de ambas extremidades respecto a los tres planos anatómicos (sagital, frontal o coronal y transversal[9]), obteniendo 9 curvas en cada extremidad. Se identifican todas las máximas y mínimas de las curvas de cada ángulo, y luego, los umbrales del algoritmo contabilizan cada máxima $(\mathrm{Max_i})$ que cumpla con dos condiciones: (*i*) la distancia en amplitud con el mínimo sucesivo $(\mathrm{Min_i})$ es mayor que el umbral de amplitud (UA) y además (*ii*), la distancia temporal entre sucesivos máximos $T\left(\mathrm{Max_i}\right) - T\left(\mathrm{Max_{i-1}}\right)$, es mayor que el Umbral Temporal (UT) . Por lo tanto, para cada $j - ésimo$ curva, el número de máximos que pasan los dos umbrales representa el número de máximos válidos en j. Finalmente, el valor promedio del número de acciones $(N°acciones_{estimado})$ se considera como ATD. Lo descrito, se presenta en el diagrama de la Figura 3.

[9]La explicación del movimiento descrito en los tres planos anatómicos se explica en detalle la sección: 3.1.4 Planos anatómicos y descripción del movimiento humano.

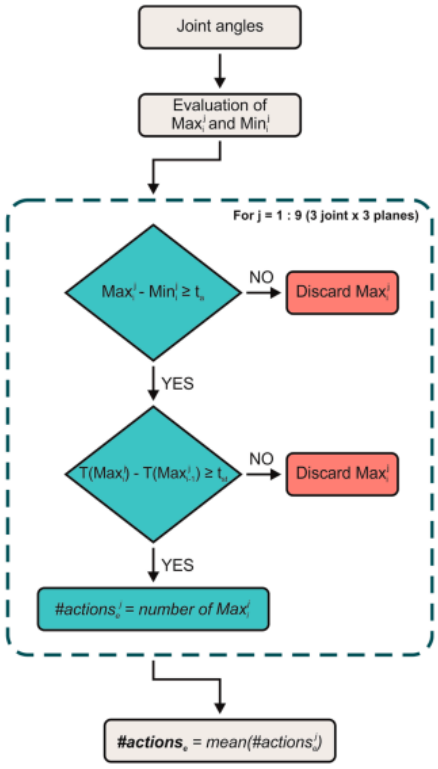


Figura 3: Diagrama de funcionamiento del algoritmo basado en umbrales para el conteo automático de ATD. Extraído de Taborri et al [14].

De manera complementaria, y con fines explicativos, se presenta la Figura 4, donde se observa que entre los 52 y 56 segundos ninguna máxima y mínima cumple ambas condiciones. Sólo se contabiliza las maximas n° 7 y n°10. La máxima n° 11 no es contabilizada debido a que no cumple el UT.

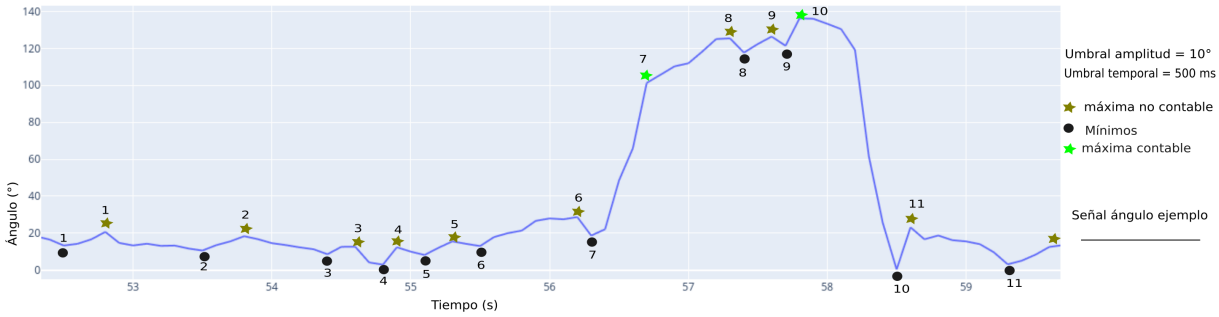


Figura 4: Contabilización de máximas en una señal aplicando el algoritmo propuesto por Taborri et al. La curva representa el movimiento en un plano, en grados, respecto al tiempo. Se enumeran las máximas y mínimas por separado y se evalúa acaso cumplen las 2 condiciones: i) la diferencia en amplitud entre la $(\mathrm{Max_i})$ y la $(\mathrm{Min_i})$, es mayor al UA, de 10° en el ejemplo, y ii) la diferencia temporal entre la máxima presente y la máxima válida anterior es superior al UT $(T\left(\mathrm{Max_i}\right) - T\left(\mathrm{Max_{i-1}}\right)$) de 500 ms. En el ejemplo, ninguna combinación de máxima y mínima cumple las dos condiciones de UT y UA entre el segundo 52 y 56. Sólo se contabilizan las maximas n° 7 y n°10. La máxima n° 11 no cumple con el UT.

En su publicación [14] comparó el $(N°acciones_{estimado})$ con el número real de ATD contadas mediante el

análisis de video ($N°acciones_{contado}$), variando 3 umbrales de amplitud, a $3°$, $5°$ y $10°$, y también, variando el UT entre $100\ ms$, $300\ ms$ y $500\ ms$ en 4 escenarios de evaluación: baja frecuencia, frecuencia intermedia, alta frecuencia, más alta frecuencia. El procedimiento se aplicó variando las posibles combinaciones de UA y UT generando nueve (3×3) versiones del algoritmo para realizar la batería de pruebas. El desempeño del algoritmo se evaluó como el error relativo del FF estimado por el algoritmo ($FF_{estimado}$) en función del FF contado por humanos (FF_{real}) según indica la siguiente fórmula:

$$Error\ Relativo(\%) = \frac{|FF_{real} - FF_{estimado}|}{FF_{real}} * 100 \tag{3.1}$$

El error relativo ($Error\ Relativo(\%)$) puede asumir cualquier valor partiendo desde 0 % cuando se obtiene el mismo valor entre FF_{real} y $FF_{estimado}$, por tanto un valor mayor indica mayor error[10]. Entre las desventajas se encontró que los umbrales deben ser configurados teniendo en cuenta las características de la tarea a evaluar, los autores recomiendan configurarlos cuidadosamente, ya que el algoritmo presentó desempeños significativamente distintos, reflejados en la oscilación del error relativo: entre un 10 % a un 30 % en la asignación del FF $FF_{estimado}$ comparado al $FF_{contado}$. La mejor combinación de umbrales temporal y amplitud que se obtuvo en los 4 escenarios de evaluación de la publicación original se presenta en la Tabla 4.

		Baja frecuencia	**Frecuencia intermedia**	**Alta frecuencia**	**Más alta frecuencia**
Extremidad Derecha	**Combinación de umbrales** Error Relativo (%)	**500 ms - 10°** 5,6 (2,5)	**500 ms - 3°** 5,7 (1,3)	**100 ms - 10°** 2,2 (1,2)	**100 ms - 3°** 4,4 (2,9)
Extremidad Izquierda	**Combinación de umbrales** Error Relativo (%)	**500 ms - 10°** 5,3 (1,2)	**500 ms - 10°** 2,6 (2,4)	**500 ms - 10°** 4,0 (3,2)	**500 ms - 10°** 5,7 (2,1)

Tabla 4: Mejores combinaciones de UT y UA para el conteo automático. Extraído de Taborri et al. [14]
La tabla muestra para cada extremidad la mejor combinacioń en los 4 escenarios de prueba realizados en la publicación [14]. El UT se establece en milisegundos (ms), y el UA, en grados (°). Los resultados muestran el valor promedio y desviación estándar del error relativo en la asignación del FF. La publicación no señala el número de acciones técnicas dinámicas obtenidas con el contador automático basado en umbral.

3.1.4. Estimación postural monocular 3D basada en vision computacional

La estimación postural monocular 3D basada en visión computacional es un campo de investigación muy activo para las ciencias de la computación por lo que existen muchas modalidades de implementación, para la estimación en 2D y en 3D [15, 16]. Estos métodos utilizan como entrada «$input$» las imágenes capturadas con cámaras monoculares digitales, y entregan como resultado las coordenadas de los puntos anatómicos en el dominio de la imagen en 2D, es decir, en qué pixeles se ubican los puntos de referencia, o también pueden entregar las coordenadas en metros, si se tiene un método de calibración como referencia. Cuando se trata de la estimación en 3D, la modalidad más común es la estimación de dos pasos: primero, se estima la postura en 2D diseñando y entrenando un modelo de redes neuronales

[10]Este cálculo de error relativo se indetermina cuando el FF real (humano) es cero. En consecuencia, en este trabajo propone una alternativa que normaliza el error relativo en función del máximo valor posible del FF. Ver detalle en la sección 6.2 Comparación entre las modalidades de conteo automático y el «conteo humano de consenso».

convolucionales profundas (CNN), y luego, se utiliza otra red neuronal de regresión para estimar la postura 3D a partir de las coordenadas 2D detectadas, como mapas de probabilidad [15, 16]. Un problema conocido es la sensibilidad a la oclusión del segmento corporal frente a la cámara, y en consecuencia, están sujetos a errores [15-19]. Sin embargo, el uso de diferentes técnicas como las redes neuronales Long-Short Term Memory (LSTM), entre otras, pueden mejorar su desempeño frente oclusiones, y hacen que la exactitud del estado del arte haya mejorado rápidamente cada año alcanzando nuevos récords [16].

Las métricas de desempeño más utilizadas describen principalmente 3 criterios: la exactitud, velocidad y robustez del modelo [16]. Sólo considerando la exactitud, en inglés «accuracy», los autores no suelen utilizar las mismas métricas en sus publicaciones, tampoco utilizan los mismos dataset ni el mismo equipamiento en cuanto a hardware computacional, por lo que la comparación es compleja. Entre las más utilizadas, el error promedio de la posición articular, en inglés, Mean Per Joint Position Error (MPJPE) y el error promedio de ángulo articular, en inglés, Mean Per Joint Angle Error (MPJAE).

$$MPJPE(x, \hat{x}) = \frac{1}{N} \sum_{i=1}^{N} \|m_{\hat{x}}(i) - m_x(i)\| \tag{3.2}$$

$$MPJAE(x, \hat{x}) = \frac{1}{3N} \sum_{i=1}^{3N} \|(m_{\hat{x}}(i) - m_x(i) \text{mod} \pm 180)\| \tag{3.3}$$

Comúnmente, en las ecuaciones 3.2 y 3.3, se utiliza la información cinemática de 16 puntos anatómicos [16], donde $\hat{x}$ representa los valores estimados por visión computacional, mientras que x representa los valores del «ground truth» del dataset, y resultan de la sumatoria de la distancia euclidiana entre los valores obtenidos por el método comparado con los datos reales disponibles en un dataset, como «ground truth». Otra métrica, define un umbral para considerar que el punto ha sido correctamente detectado, entonces se calcula la estadísticas de las articulaciones correctamente detectadas en el set de imagenes: el Porcentaje de Articulaciones Detectadas, en inglés, Percent of Detected Joints, también llamado Porcentaje de Puntos Correctos en 2D o 3D, en inglés, Percent of Correct Points, el cual se asocia a un porcentaje de tolerancia, por ejemplo, el 20 % de tolerancia (PCK3D@0.2), donde el punto está correctamente detectado cuando la distancia euclidiana es menor al 20 % del tamaño del torso de la persona [16]. En cuanto a la velocidad, una métrica es el número de operaciones por segundo requeridos por el modelo, a mayor cantidad de cálculos, más lento será el modelo; otra, el tiempo requerido para procesar 1 cuadro, en milisegundos, o bien el número de cuadros procesados por segundo, en inglés, Frames per second (FPS). En cuanto a la robustez del modelo, no existe métrica específica pero se suele deducir en evaluando la exactitud en múltiples datasets [16].

Comparación entre métodos de estimación postural basada en visión computacional. Entre los métodos más utilizados para la estimación postural 3D basada en visión computacional, se encuentran OpenPose (2018) [20] y Mediapipe Pose (2020) [21], sin embargo, el rápido avance del estado del arte hace que puedan considerarse prácticamente obsoletos, en cuanto a exactitud. Se ha demostrado que OpenPose posee un desempeño de exactitud superior a Mediapipe Pose en el AR Dataset, en cambio, en

Yoga Dataset, presentan desempeño similar. Sin embargo, la velocidad de Mediapipe Pose es superior en términos de FPS al procesar con CPU, lo cual puede ser observado en las Tablas 5 y 6, donde además se incluye al método MMPose (2020) para su comparación. Dicha Tabla 6 incluye el año de publicación, métricas de desempeño disponibles en los artículos de su publicación, por lo que no siempre es posible comparar directamente entre métodos. Si bien, Mediapipe Pose no dispone de métricas de exactitud en términos de MPJPE, podemos suponer que MMPose (MPJPE = 17,57 mm) es superior a ambos métodos, ya que supera con creces a OpenPose-OpenVINO (MPJPE = 100 mm) en el dataset CMU Panoptic. En cuanto a la velocidad de procesamiento en CPU, destaca Mediapipe Pose, por sobre los demás porque habilita su utilización en tiempo real, en computadores de uso personal, incluso se ha probado en teléfonos móviles y equipos como Raspberry Pi 4. Por último, se detallan las posibilidades de instalación y preparación para la versión en CPU, concluyéndose que Mediapipe Pose es la opción de más simple utilización, seguido de MMPose el cual requiere de un equipamiento de mayor capacidad para su uso, pues, no ofrece más de 13 FPS, en CPU Intel(R) Core(TM) i7-8700 CPU @ 3.2GHz.

Modelo	**FPS**	**AR Dataset, PCK@0.2**	**Yoga Dataset PCK@0.2**
OpenPose	0.4[a]	87.8	83.4
BlazePose Full (Mediapipe Pose)	10[a]	84.1	84.5
BlazePose Lite (Mediapipe Pose)	31[b]	79.6	77.6

Tabla 5: Comparativa entre soluciones de visión computacional: BlazePose 2D (Mediapipe Pose) y OpenPose.
[a]CPU con 20 núcleos (Intel i9-7900X); [b]Teléfono Pixel 2 Single Core. Extraído de [21]

Variable Comparación	Real-time 3D Multi-person Pose Estimation OpenPose-OpenVINO[a]	Mediapipe Pose (BlazePose GHUM)[b]	MMPose[c]
Año de publicación	2019	2020	2020
N° de Puntos anatómicos estimados	18	33	133
MPJPE	100 mm (CMU Panoptic dataset)	No disponible. Según se informa, la estimación 3D se basa en datos sintéticos.	17.57 mm (CMU Panoptic dataset) 46.8 mm MPJPE (Human3.6M)
Velocidad (FPS) en CPU.	No disponible (revisar tabla 5)	**Modelo Lite** ~44 FPS en CPU via XNNPac; **Modelo Full** ~18 FPS en CPU via XNNPack TFLite; **Modelo Heavy** ~4 FPS en CPU via XNNPack TFLite, con el siguiente equipamiento: MacBook Pro (15-inch 2017)	Entre 0,01 y 13 FPS en CPU con el siguiente equipamiento: CPU Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz
Multipersona	Si	No	Si
Sistema Operativo	Ubuntu, Windows, Mac OSX	Ubuntu, Windows, Mac OSX, Android	Ubuntu, Windows, Mac OSX
Requisitos hardware (versión CPU)	Al menos 8GB de memoria RAM libre; CPU con al menos 8 núcleos (altamente recomendado)	No indicado explícitamente, aunque la solución es múlti plataforma: GPU, CPU, teléfonos móviles, Raspberry Pi 4 y Jeton Xavier NX	No indicado

Posibilidades de Instalación y preparación (versión CPU)	**Windows:** Versión ejecutable instalación directa.
	Ubuntu y MacOS: Versión ejecutable no disponible.
	Compilación del modelo pre-entrenado: pre-requisitos de instalación[d]. Creación de ambiente, descarga del modelo pre-entrenado.
	Entrenamiento del modelo:
	Clonar repositorio, descargar dataset, entrenar y testear el modelo.
		Windows, Ubuntu y MacOS:
		Paquete con la solución compilada en Python 3:
		Crear un ambiente virtual Python 3 e instalación directa mediante:
		«pip install mediapipe»
		Detalle de instrucciones en <Instalación de Mediapipe>
			Windows, Ubuntu y MacOS:
			Paquete con la solución compilada en Python 3: Crear un ambiente virtual Python 3.6+ y PyTorch 1.5+ (para CPU).
			Instalación directa mediante:
			$ pip install openmim
			$ mim install mmcv-full
			$ #Clonar repositorio e instalar
			$ git clone https://github.com/open-mmlab/ $ mmpose.git
			$ cd mmpose
			$ pip install -e .
			Compilación del modelo pre-entrenado: Se provee código para evaluar modelos pre-entrenados. Requiere descargar Dataset y configurar rutas a las dependencias.
			Entrenamiento del modelo:
			Clonar repositorio, descarga de dataset, permite entrenar y testear el modelo.

Tabla 6: Comparación de métodos de visión computacional para la estimación postural en 3D.
[a]Información disponible en <OpenPose-OpenVINO>;
[b]Información disponible en <Model Card Mediapipe BlazePose> y <mediapipe en google.github.io>;
[c]Información disponible en: <Resumen de velocidad de MMPose> y <enlace a MMPose respecto a Panoptic-dataset>;
[d]Librerías y dependencias para Ubuntu 20: «OpenCV» (versiones compatibles 2.X y 3.X). Librería «Caffe» y todas sus dependencias;

Planos anatómicos y descripción del movimiento humano. El estudio del movimiento humano se suele describir en relación a 3 planos anatómicos imaginarios, los cuales se corresponden, cada uno, con un eje de movimiento que es perpendicular a dicho plano y lo define. Como se expone en la Figura 5, adaptado de Norkin y White [22], se presentan los ejes x, y, z y los planos de movimiento para la articulación del hombro, donde se esquematiza en cada imagen un plano adicional, en color verde, respecto de los movimientos que dichos planos definen. Es importante señalar que los planos anatómicos se vuelven a definir en las extremidades en función de los movimientos que naturalmente describen las articulaciones, y la disposición que tenga en ese momento la extremidad. Esto quiere decir, que la flexión del codo siempre se define en el plano sagital, independiente de la orientación de la extremidad en el espacio, entonces existe un sistema de coordenadas locales en cada articulación:

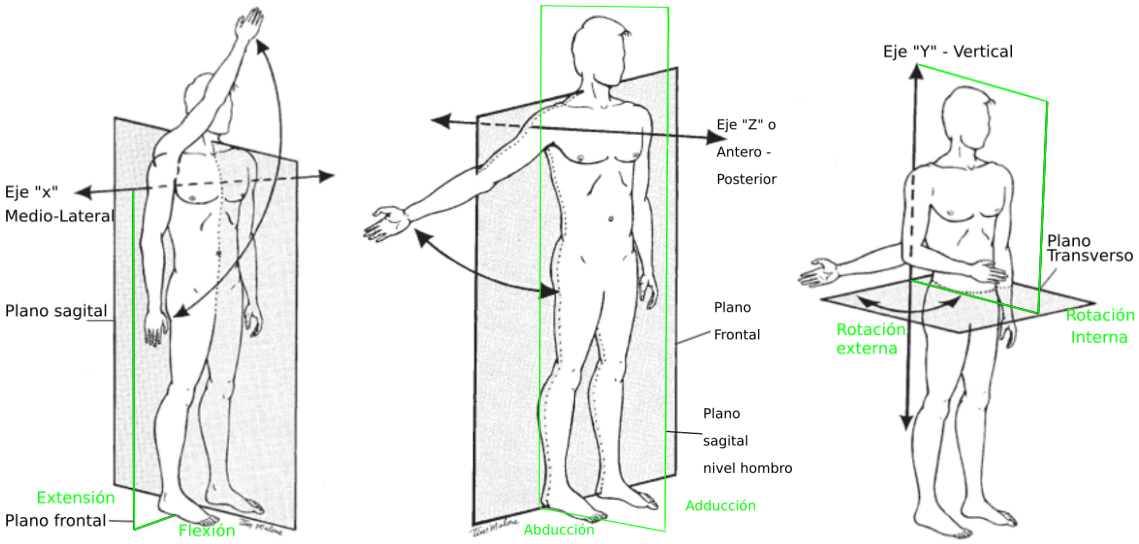
- El Plano Medio Sagital, es un plano vertical que divide el cuerpo en dos mitades, derecha e izquierda. Los planos sagitales, son planos paralelos al plano medio sagital y suelen definirse en función de una referencia anatómica, por ejemplo: El movimiento de flexión-extensión del hombro, se describe en un plano sagital a nivel del hombro, y se suele describir como aquel movimiento que utiliza como eje x o medio-lateral y, de forma imaginaria, podría «limpiar el plano sagital». La flexión se define como el movimiento hacia la parte delantera del cuerpo (delimitado por el plano frontal a nivel de hombro) y la extensión se define como el movimiento hacia la parte trasera del cuerpo, llevando el brazo hacia atrás. Lo cual se onserva en la Figura 5a.

- Los Planos Frontales (o coronales), son planos verticales que atraviesan el cuerpo y lo dividen en dos partes: anterior (frontal) y posterior (dorsal), y forma un ángulo recto con el plano sagital. En el hombro, la abducción-adducción utiliza como eje Antero-Posterior, eje z en nuestro sistema de referencia, y se describe en función del plano frontal o coronal. Se considera abducción cuando el movimiento se ejecuta en dirección externa del cuerpo y se considera adducción el movimiento descrito en sentido contrario, ver Figura 5b (en verde). Cabe destacar que este último movimiento sólo puede ejecutarse en combinación con flexión o extensión del hombro, debido a la presencia del torso del tronco de la persona.

- Los Planos Transversos, son planos horizontales que dividen el cuerpo en dos partes: superior e inferior y atraviesan en ángulo recto los planos sagital y frontal. La rotación interna y externa se describen en función del plano transverso. Cuando la rotación se realiza en dirección hacia la línea media del cuerpo respecto al plano sagital a nivel de hombro, se habla de rotación interna. Y a la inversa, cuando el movimiento se describe en sentido contrario, se habla de rotación externa (ver Figura 5c). En este trabajo, se asignó valores negativos de la rotación de hombro como distintivo de la rotación interna y los valores positivos deben considerarse rotación externa.

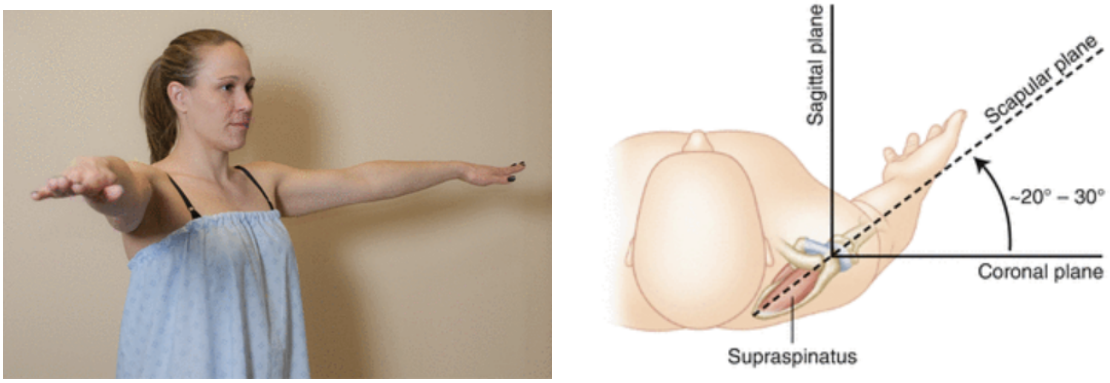
La mayoría de movimientos naturales ocurren como una combinación de movimientos en los 3 ejes o 3 planos, si se quiere, los que constituyen componentes de dicho movimiento. Por ejemplo, en la Figura 6 los brazos realizan una abducción fuera del plano coronal y se combina con una flexión y rotación. En tal caso, si la persona es observada desde el lado derecho, desde una perspectiva paralela al plano coronal, el ángulo que se forma entre la vertical (eje Y) y la proyección del brazo sobre el Plano Sagital, tal como si se iluminara provocando una sombra, corresponde al componente flexor del movimiento. Lo mismo ocurre al observar desde el frente: el ángulo entre la vertical y la proyección del brazo sobre el plano coronal determina la componente de abducción del hombro, y por último, si se observa desde una

perspectiva superior (mirando el plano transverso), se observa el componente rotacional, que para este caso, se trata de una rotación externa (Figura 6b).



(a) flexión-extensión (b) abducción-adducción (c) rotación interna y externa

Figura 5: Definición de los movimientos de hombros en función de los ejes y planos movimiento. Adaptado de Norkin y White [22].

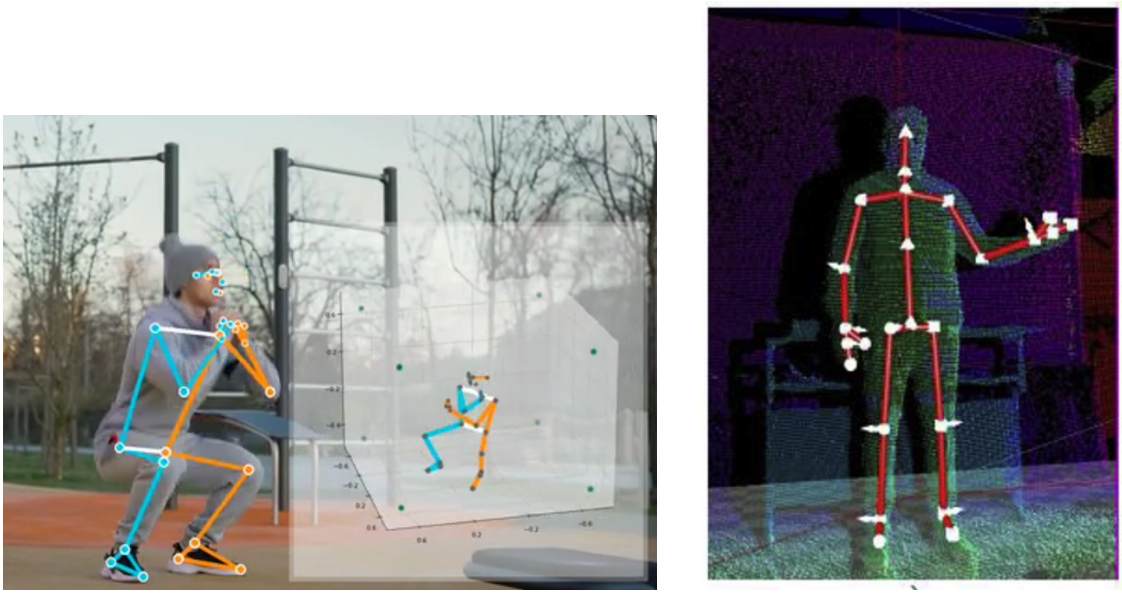


(a) (b)

Figura 6: Figuras a y b muestran la abducción de hombros en el plano escapular, que posee un componente flexor, abductor y de rotación. Imágenes extraídas de https://musculoskeletalkey.com/range-of-motion/.

La Figura 6a muestra que la perspectivas de observación revelan distintos componentes del movimiento en hombros: desde una vista lateral, el ángulo entre la vertical y la proyección del brazo sobre el plano sagital a nivel del hombro representa la flexión; mirado desde el frente, el ángulo entre la vertical y la proyección del brazo sobre el plano coronal indica el ángulo de abducción. Por último, desde una vista superior 6b, la rotación interna ocurre cuando el brazo supera el plano sagital que se ubica a nivel de hombro, en dirección hacia el plano medio sagital; y, por el contrario, la rotación externa, cuando el brazo se aleja del plano sagital ubicado a nivel del hombro.

Modelamiento humano y esqueleto vectorial 3D. La estimación en 3D permite ubicar en el espacio la posición de los puntos anatómicos de la persona bajo análisis, lo que suele representarse en pixeles o en metros, si se utiliza una referencia como calibración. Esto habilita la construcción del esqueleto humano como modelo virtual, donde los puntos anatómicos estimados indican la posición de las articulaciones que permiten construir los segmentos corporales o «huesos», como representación vectorial con el origen definido como el punto medio entre las caderas. Esto permite fijar las coordenadas en el modelo humano, es decir, los términos «izquierda», «derecha», «adelante», «atrás», entre otros, se refieren a la reconstrucción. A continuación, junto con las coordenadas del origen se utiliza el punto medio entre los hombros para definir el segmento «tronco», como línea media del cuerpo y eje vertical y; a su vez, cada segmento corporal resulta de la resta vectorial de las posiciones en 3D de las articulaciones que le dan origen, por ejemplo, el vector que representa el segmento brazo derecho resulta de restar las coordenadas del hombro derecho y el codo derecho. La Figura 7a muestra los puntos anatómicos de Mediapipe Pose requeridos para construir un esqueleto virtual 3D similar al que se produce la aplicación de Kinect for Windows SDK 2.0 expuesto en la Figura 7b. Al definir segmentos corporales mediante la resta vectorial, se definen relaciones de dependencia jerarquizada de las coordenadas y segmentos corporales construidos: por ejemplo, la posición de la mano (vector) y muñeca (coordenadas), dependen de la posición del antebrazo (vector) y codo (coordenadas); éstos, a su vez, dependen de la posición del brazo (vector) y el hombro (coordenadas), y así sucesivamente, hasta que se define la relación con el origen del sistema de referencia.



(a) (b)

Figura 7: Reconstrucción de las coordenadas en espacio vectorial 3D.
En la Figura 7a, se presenta la estimación postural 3D basada en Mediapipe Pose. Extraído de [23]. En la Figura 7b, el esqueleto vectorial basado en la topología de Kinect V2. Extraído de [23].

Como se ha mencionado, el punto medio entre las caderas (vector 1) y el punto medio entre los hombros (vector 2) definen el segmento «tronco» (vector 3), lo que permite determinar el plano «Frontal» (o

«Coronal») del sistema de referencia[11] y, con ello, calcular el vector ortogonal al plano y habilitar la definición de los planos «Sagital» y «Transversal» del modelo virtual 3D (para mayor información referirse al Planos anatómicos y descripción del movimiento humano). Esto último, permite además, calcular los ángulos de las articulaciones del hombro (segmento brazo), codo (antebrazo) y muñeca (mano) de ambos miembros superiores respecto a los tres planos anatómicos, obteniendo las curvas en cada extremidad, que requeriría el algoritmo de Taborri et al para su aplicación (ver la sección: Algoritmo para el conteo automático de «Acciones Técnicas Dinámicas»).

3.2. Problema

Existen limitaciones humanas al aplicar métodos observacionales en ergonomía para evaluar el riesgo biomecánico de TMERT, tales como: tiempo excesivo [24, 25], alto costo y reproducibilidad [24, 26], por ejemplo, en el Método OCRA Checklist se ha reportado un tiempo de aplicación total promedio de 16 min (SD=5.9, rango [6-31]) y una reproducibilidad en el FF del 30 % y 23 % intra e inter evaluador, respectivamente [25]. Para superar dichas limitaciones se utiliza instrumentación para medir o capturar las variables físicas habitualmente estimadas. Para ello, se captura o se estima la postura del trabajador utilizando sensores inerciales, sensores de profundidad (RGB-D) [23] o mediante visión computacional [17-19] en un abordaje combinado (se combina el método observacional con la captura o estimación postural) para calcular variables del movimiento requeridas. Para evaluar el riesgo biomecánico asociado al trabajo repetitivo, la propuesta de Taborri et al. [14] permite automatizar el conteo de ATD por minuto a partir de la medición, con sensores inerciales, de los ángulos que surgen entre los planos anatómicos: sagital, coronal y transversal, y los segmentos que unen las articulaciones del hombro, codo y muñeca en base a sus posiciones en 3D (x, y, z). Dichos sensores no son ampliamente utilizados por requerir formación especializada para su uso, y pueden limitar los movimientos de los trabajadores [17, 23, 27]. La visión computacional permite estimar las posiciones articulares en x, y, z y calcular los ángulos requeridos por el algoritmo de Taborri et al. permitiendo la posibilidad de implementarlo para calcular el número de ATD a partir de videos directamente. No se hallaron publicaciones asociadas a una implementación de esta naturaleza en cuanto a métodos observacionales dedicados a estudiar el riesgo de TMERT en el trabajo repetitivo, por lo tanto, es importante avanzar en el desarrollo de soluciones que no requieran de instrumentación adicional.

Considerando el análisis realizado, Mediapipe Pose es la herramienta de estimación postural basada en visión computacional de más simple de instalar y utilizar en el desarrollo de esta tesis. Adicionalmente, teniendo en cuenta la existencia de bases de datos, gratuitas y de acceso público que contienen videos junto con la anotación de la captura 3D de la postura humana en tareas del contexto ocupacional, se plantean las siguientes preguntas de investigación ¿Qué diferencia hay entre un conteo automático de ATD mediante visión computacional comparado con un $ConteoH_{Consenso}$ proporcionado por especialistas en ergonomía, a través del analisis de videos de un Dataset? Además, ¿Qué diferencia hay si ese conteo automático también se compara con el conteo obtenido a partir de anotaciones de la postura disponibles en el Dataset? ¿Cómo se comporta la asignación del FF entre las distintas modalidades de conteo? Para seleccionar la base de datos a utilizar, se elaboró la tabla 7. Finalmente, se optó por *«University of Washington Indoor Object Manipulation»* como base de datos a incluir en este trabajo, debido a la

[11] El producto cruz de dos vectores, en el ejemplo, Vector Tronco y Vector Caderas, produce un vector que es perpendicular a ambos, definiendo los puntos que dan origen a los planos anatómicos del modelo virtual 3D.

visibilidad del rostro de las personas filmadas en los videos del Dataset, ya que la anonimización es una práctica común en videos de acceso público, lo que limita la utilización de Mediapipe Pose para la estimación postural, toda vez que el funcionamiento de su red neuronal depende de la detección del rostro en las imágenes analizadas, tal como se explica en el Anexo: «Mediapipe Pose» - estimación postural basada en visión computacional

Base de datos	«University of Washington Indoor Object Manipula-tion»[28]	MOPED25 [29]	TUM Kitchen Dataset [30]	MPII Cooking Activities	KIT Whole-Body Human Motion Database[31]	AndyData-lab-onePerson [32][12]
Anotación de la Postura en 3D	SI	SI	SI	SI	SI	SI
Videos con rostro descubierto	SI	SI	SI	SI	NO	NO
Posibilidad de contar acciones técnicas	SI	NO	SI	SI	SI	SI
Método de Captura de movimiento	Kinect V2	Sistema Multicámara 37 Marcadores	Kinect V2	Visión Computacio-nal multicámara	Azure_Kinect; Sensores inerciales VICON	Sistema óptico (gold standard) y sensores inerciales
Duración de cada video	Alrededor de 3 minutos	5 a 30 segundos	1 a 4 minutos	2 a 40 minutos	8 a 30 segundos	2 a 4 minuto
Disponible para descarga	SI	SI	SI	SI	SI	SI
Motivo de descarte	-	Video registra 1 ejecución (no permite contar ATD)	Filmación desde esquina superior de la sala (poco usual, en el contexto de la ergonomía)	Dificultad para importar la postura 3D.	Video con rostro anonimizado; Acciones técnicas de 1 solo ciclo.	Video con rostro anonimizado

Tabla 7: Tabla de desición para seleccionar conjunto de datos a utilizar en la investigación.

[12] Se tomó contacto con los autores, con la finalidad de conseguir los videos sin anonimizar, pero indicaron que consentimiento informado esteblece su anonimización.

4. Hipótesis

El conteo automático de «acciones técnicas dinámicas», en las distintas modalidades, determinará un «factor frecuencia» similar al «conteo humano de consenso» en el Método Occupational Repetitive Actions Checklist.

5. Objetivo

5.1. Objetivo general

Comparar el conteo automático de «acciones técnicas dinámicas» con el «conteo humano de consenso» determinado por especialistas en los videos de tareas repetitivas de la base de datos «University of Washington Indoor Object Manipulation Dataset (Dataset)»

5.2. Objetivos específicos

1. Obtener y comparar las variables cinemáticas obtenidas a partir de la anotación de las coordenadas articulares del Dataset con aquellas variables cinemáticas obtenidas por Mediapipe Pose, en los respectivos videos del Dataset.
2. Definir el «conteo humano de consenso» entre evaluadores especialistas en cuanto al número de «acciones técnicas dinámicas» de extremidades superiores en videos de tareas repetitivas del Dataset.
3. Realizar un conteo automático de «acciones técnicas dinámicas» a partir de variables cinemáticas anotadas y a partir de la estimación postural de Mediapipe Pose en videos del Dataset.
4. Comparar el error relativo de las distintas modalidades de conteo automático con los resultados obtenidos mediante consenso por evaluadores especialistas.

6. Material y método

A continuación, se describen las herramientas utilizadas en esta investigación, junto con el detalle de cada uno de los pasos llevados a cabo en el procesamiento de los datos. En algunos pasos, a modo de profundizar la explicación, se agregan fragmentos del código utilizado, en otros, sólo se provee una descripción de la función utilizada en Python, incluyendo los parámetros de entrada y elementos de salida, sin entregar detalles del código, para reducir la extensión de la explicación. Sin embargo, en ambos casos, el detalle completo se dispone en los Anexos de este trabajo.

6.1. Base de datos *«University of Washington Indoor Object Manipulation»*

El conjunto de datos denominado «University of Washington Indoor Object Manipulation» [33], en adelante referido como Dataset, consiste en 20 videos junto con la detección de posturas humanas (en 2D y 3D) almacenadas en archivos de extensión .mat y etiquetas de actividad asignada a cada cuadro de los videos de veinte participantes con edades entre 18 y 25 años. Fue desarrollado para abordar la escasez de videos que capturen acciones de manipulación de objetos involucrando posturas incómodas y repeticiones, y la captura postural en 2D y 3D. Los videos fueron grabados con una cámara sensor de Kinect V2 para Xbox One entre 8 a 12 cuadros por segundo.

6.1.1. Características de la tarea disponible en el Dataset

Se registró una única tarea ejecutada tres veces por cada participante en un mismo registro, por lo que cada video tiene una duración aproximada de tres minutos: existe un estante donde se han ubicado cajas y barras en tres niveles, a diferentes alturas. Partiendo desde el nivel mas alto, toman la caja y la colocan sobre una mesa frente al estante y, luego, realiza lo mismo con la barra, esto se repite en el nivel medio y con los elementos del nivel inferior del estante. A continuación, la caja y la barra se vuelven a colocar en los respectivos niveles del estante, desde donde inicialmente se recogieron. Las cajas se manipulan con ambas manos y las varillas con una sola mano, sin embargo la estrategia motora de cada participante es variable, sin instrucción específica.

6.1.2. Utilización del Dataset

Los 20 videos del Dataset se utilizaron en conjunto con Mediapipe Pose para obtener la estimación postural en 3D y, además, para realizar el análisis visual por parte de los evaluadores especialistas. Igualmente, se utilizó la anotación de la postura en 3D que el grupo de datos incluye. Dichos datos se importaron a Python utilizando las librerías H5Py, para acceder a los archivos con extensión .mat (nativos de Matlab), Pandas y NumPy. De esa forma se cargaron y procesaron los datos de cada video con las funciones $PrepareAndLoadDataset$ o $PrepareAndLoadMediapipe$, según se señala en la subsección 11.5. Se desecharon los datos posturales anotados en el Dataset del video n°3 debido a las grandes desviaciones pesquizadas en la captura postural.

Mayor detalle en cuanto a los elementos componentes, características e ilustraciones del Dataset, se expone en el 11.2 Anexo: Características del grupo de datos: «University of Washington Indoor Object Manipulation Dataset».

6.2. «Conteo humano de consenso» de «acciones técnicas dinámicas» en videos del Dataset

El $ConteoH_{Consenso}$ se refiere al valor promedio de ATD por minuto, obtenido por los 3 participantes Kinesiólogos Ergónomos con formación acreditada en la aplicación del Método OCRA Checklist. Los profesionales recibieron una invitación vía correo electrónico y, luego de aceptar y firmar el consentimiento informado, se les envió un enlace web único a una hoja de cálculo de Google Drive. La hoja incluyó los campos necesarios para completar, en cada extremidad, el número de ATD en la totalidad del video, el número de ATD por minuto y un enlace directo para acceder a visualizar los videos. Posteriormente, estos datos se transcribieron a Python bajo el formato DataFrame de la librería Pandas para procesarlos. Con la finalidad de obtener el «conteo humano de consenso» $Conteo_{Consenso}$ para cada extremidad en cada uno de los videos, se promediaron los resultados de cada participante y, además, mediante la función $FuncionFFrecuencia$ (Código 1) se asignó el respectivo FF del Método OCRA Checklist. La Tabla 8 muestra la hoja de cálculo de Google Drive referida.

Código asignado al Participante	N° de Video	N° de acciones técnicas dinámicas		Tiempo de Observación (min:seg)	N° de acciones técnicas dinámicas por minuto.		Puntaje Factor Frecuencia (completa investigador)	
		Extremidad Izquierda	Extremidad derecha		Extremidad Izquierda	Extremidad derecha	Extremidad Izquierda	Extremidad derecha
	1							
	2							
	3							
	4							
	5							
	6							
	7							
	8							
	9							
	10							
	11							
	12							
	13							
	14							
	15							
	16							
	17							
	18							
	19							
	20							

N° de Video	Enlace
Video 1	https://drive.google.com/file/d/1gl95JkZzNKyA7Bp3UsD
Video 2	https://drive.google.com/file/d/1RMeODfYzu92uPbeQAp
Video 3	https://drive.google.com/file/d/1bZDUDp-KjSncnLPTe3P
Video 4	https://drive.google.com/file/d/10BHh4UmbPRwy2A0Xe
Video 5	https://drive.google.com/file/d/1hw8ao-DBAq35vSnf9zpf
Video 6	https://drive.google.com/file/d/18K4NEyoA6g_hS2WeK2
Video 7	https://drive.google.com/file/d/1-WjkHt_1NwrWWt6g4GN
Video 8	https://drive.google.com/file/d/1dWKkDChSl87jqQtM84r
Video 9	https://drive.google.com/file/d/17zdrn_IQiqvC1ZeWpUT-
Video 10	https://drive.google.com/file/d/1tBCviQyUmRmppU8qbr
Video 11	https://drive.google.com/file/d/1pFqaIC2h9TCBV5JzwG3
Video 12	https://drive.google.com/file/d/1FQPD1uwD93HZf6DSV
Video 13	https://drive.google.com/file/d/1lM_Q9fQLbP-0LVPouNj1
Video 14	https://drive.google.com/file/d/1SJGczI8MW2FBcrs2ZOp
Video 15	https://drive.google.com/file/d/1gUSXbnGcKFBJbMauns
Video 16	https://drive.google.com/file/d/1x_dEkH2T-JsYP6uHq0L
Video 17	https://drive.google.com/file/d/1Gd7NhmRCQRg1O2Dm
Video 18	https://drive.google.com/file/d/17RI-lOgvEl0FZsXEf2fRW
Video 19	https://drive.google.com/file/d/1AAAygGzTezo5x_Xfjvti5
Video 20	https://drive.google.com/file/d/1pVI41-14irPqPjHeZgf8oF

Tabla 8: Planilla a completar por los participantes y enlace para visualizar los videos.

Código 1 Función «$FuncionFFrecuencia$», que asigna el puntaje del FF del método de evaluación ergonómica OCRA Checklist.

```
def FuncionFFrecuencia(value):
    """
    Asigna el puntaje del Factor Frecuencia del método de evaluación ergonómica
       OCRA Checklist.
    Parámetros:
    value (float): Valor de entrada utilizado para determinar el puntaje.
    Retorna:
    float: El puntaje asignado según el valor de entrada.
    """
    if value  < 22.5:
        return 0.0
    elif value >= 22.5 and value < 27.5:
        return 0.5
    elif value >= 27.5 and value < 32.5:
        return 1
    elif value >= 32.5 and value < 37.5:
        return 2
    elif value >= 37.5 and value < 42.5:
        return 3
    elif value >= 42.5 and value < 47.5:
        return 4
    elif value >= 47.5 and value < 52.5:
        return 5
    elif value >= 52.5 and value < 57.5:
        return 6
    elif value >= 57.5 and value < 62.5:
        return 7
    elif value >= 62.5 and value < 67.5:
        return 8
    elif value >= 67.5 and value < 72.5:
        return 9
    elif value >= 72.5:
        return 9
```

6.3. Pre-procesamiento de los datos posturales

El pre-procesamiento consistió en obtener y almacenar el grupo de datos de la estimación postural 3D basada en visión computacional mediante Mediapipe Pose, así como también en construir los puntos anatómicos en espacio tridimensional y su respectivo esqueleto vectorial para el Dataset y Mediapipe, para realizar el cálculo de ángulos necesarios en esta tesis. Además, se prepararon los datos para el análisis mediante correcciones al sistema de referencias del Dataset y al componente «Z» de Mediapipe Pose, se sincronizaron temporalmente los registros y redujo la «agitación» en la ubicación 3D de las coordenadas de ambos conjuntos de datos.

6.3.1. Estimación postural 3D mediante Mediapipe Pose y almacenamiento

La estimación postural en 3D consistió en obtener las coordenadas articulares de las personas filmadas utilizando Mediapipe Pose con la librería OpenCV para procesar los 20 videos del Dataset y almacenarlas cuadro por cuadro, como un grupo de datos, en archivos $.csv$, donde el valor del componente «z» de las

coordenadas de cada punto anatómico obtenido por Mediapipe se escaló aplicando una división tipo «$floor$» por 3. Además, se capturó otro archivo $.csv$ con los cuadros en los que la persona estaba de espaldas a la cámara con la finalidad de corregir, más adelante, la lateralidad de los puntos anatómicos del Dataset, el cual define como eje de coordenadas al lente de la cámara que captura la escena. Esto supone dos desafíos que se resolvieron según se explica en las secciones siguientes, a saber: corregir el eje de coordenadas y la lateralidad de los puntos anatómicos. En el Dataset los términos «hombro izquierdo» o «derecho» son fijos puesto que están en referencia al campo visual de la escena. En cambio, en Mediapipe, el término «hombro izquierdo» se refiere al hombro izquierdo de la persona independiente de dónde ésta mire. El detalle completo del código utilizado en este paso, se presenta en la sección 11.1.1 Anexo: (Código Python) Estimación postural 3D mediante Mediapipe Pose y almacenamiento en $.csv$.

Limitaciones

Cabe mencionar que la topología de coordenadas presentes en el Dataset y Mediapipe no ofrecen la cantidad de puntos anatómicos necesarios para definir una reconstrucción fiel de los planos y ejes de coordenadas locales que recomienda la Sociedad Internacional de Biomecánica en su artículo de estandarización [34]. Por lo tanto, esta implementación replica lo descrito por Manguishi et al. en [23] quienes utilizaron una Kinect V2, para implementar el método de evaluación postural Rapid Upper Limb Assessment (RULA). Además, en la presente propuesta se hacen adaptaciones para los puntos de referencia anatómicos que ofrece Mediapipe.

Esta propuesta intenta emular la descripción biomecánica a partir de un análisis visual que realizarían profesionales especialistas en el análisis del movimiento humano, tales como especialistas en ergonomía, médicos, kinesiólogos, terapeutas ocupacionales, profesores de educación física, entre otros.

6.3.2. Reconstrucción de puntos anatómicos en espacio tridimensional de Unity Engine

Se construyó un modelo humano virtual por cada grupo de datos, Dataset y Mediapipe. Dicho modelamiento se refiere a representar en el espacio de simulación del Sofware Unity Engine 2021.3.14f[13] los puntos anatómicos o coordenadas, y conectarlos para su representación visual. El modelamiento también incluyó un esqueleto vectorial (ver Figura 8b), el cual no es visible en la representación, pero que permitió definir los sistemas coordenadas locales de cada articulación y el sistema de coordenadas global del esqueleto, lo que es necesario para efectuar el cálculo de ángulos sobre los tres planos de movimiento. El uso de Unity Engine permitió navegar el espacio de simulación para observar la reconstrucción y cambiar la perspectiva desde cualquier ángulo, mientras se definían los vectores del esqueleto y se llevaba a cabo la optimización de código. Para ello, además de los datos posturales disponibles, se utilizó la cámara web del computador personal junto con Mediapipe Pose para transmitir a Unity Engine la ubicación espacial de las coordenadas articulares estimadas en tiempo real y, con ello, se verificó que el código era consistente.

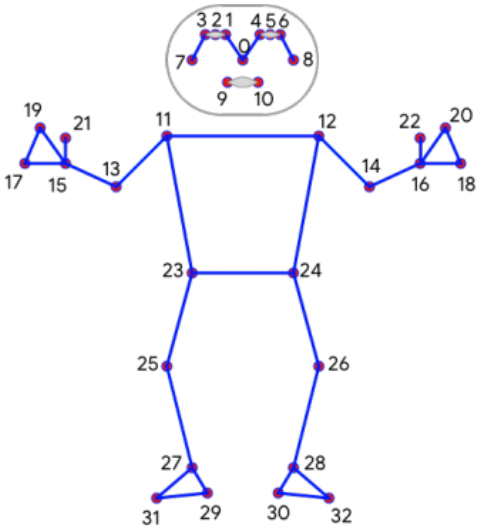
La implementación inicial de todos estos pasos se realizó en el lenguaje C#, de Unity Engine, lo que

[13] Es importante señalar, que se utilizó Unity Engine debido a que este autor estudió cursos gratuitos de visión computacional con Python: «Advance Computer Vision with Python» y «3D Hand Tracking», creados por Murtaza Hassan, disponibles en: https://www.computervision.zone/courses/advance-computer-vision-with-python/ y https://www.computervision.zone/courses/3d-hand-tracking/, respectivamente, donde se entregan los fundamentos del uso de Mediapipe Pose y, se utiliza Unity Engine para recibir y proyectar en tiempo real la estimación postural, entregando las bases de conocimiento y algunas herramientas que se utilizaron en este proyecto de investigación.

se detalla en la sección 11.3 Anexo: (Código $C\#$) Implementación en Unity Engine y (Código C#) Functions.cs. Sin embargo, dado que la implementación final se replicó por completo en Python, en adelante, las referencias específicas son presentadas en este último lenguaje de programación.

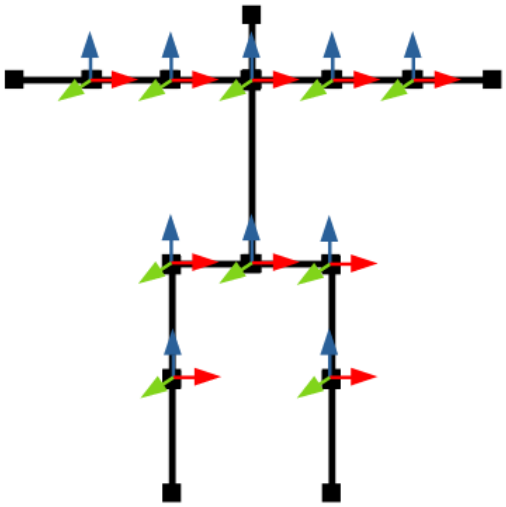
Esqueleto vectorial El primer paso para construir el esqueleto vectorial consistió en corregir el sistema de coordenadas del Dataset, cuyo origen predefinido es el lente de la cámara que captura el movimiento (ya descrito en la sección 6.3.1 Estimación postural 3D mediante Mediapipe Pose y almacenamiento) con la finalidad de hacerlo coincider con el sistema de referencias de Mediapipe Pose, situado en el punto medio entre las caderas. Para ello, se realizó una resta aritmética de cada componente x, y, z por cada uno de los puntos anatómico del Dataset con el respectivo componente x, y, z del punto medio entre ambas caderas, lo que redefine todas las coordenadas y asigna las coordenadas $0, 0, 0$ a este último.

Una vez que cada grupo de datos tiene su sistema de coordenadas situado en el punto medio entre las caderas, se promedió las posiciones de ambos hombros para obtener el punto medio y, a continuación, mediante una resta vectorial con el origen del sistema de coordenadas, se estableció el primer segmento corporal «el tronco» o «columna», que también da origen al Eje Y (vertical o cefalo-caudal) del sistema global (ver Figuras 8a y 8b). En tal caso, la resta vectorial de la posición de la cadera izquierda y el eje de coordendas entre las caderas, determina el Eje X del sistema (medio-lateral). Por último, el Eje Z (antero-posterior) se define como el producto vectorial cruz entre ambos $Z = X \otimes Y$, el cual apunta «hacia adelante» del modelo humano. En el hombro, el vector «brazo» se definió por las coordenadas del hombro y del codo, con su origen en el hombro, cuya dirección y sentido apunta hacia el codo. En tal caso, el Eje vertical Y del sistema global se traslada al hombro respectivo, mientras que el Eje X del sistema local se construye mediante la resta vectorial el punto medio de los hombros y el hombro respectivo.



Topología Mediapipe, en Inglés:
0. Nose; 1. Left eye inner; 2. Left eye; 3. Left eye outer; 4. Right eye inner; 5. Right eye; 6. Right eye outer; 7. Left ear; 8. Right ear; 9. Mouth left; 10. Mouth right; 11. Left shoulder; 12. Right shoulder; 13. Left elbow; 14. Right elbow; 15. Left wrist; 16. Right wrist; 17. Left pinky #1 knuckle; 18. Right pinky #1 knuckle; 19. Left index #1 knuckle; 20. Right index #1 knuckle; 21. Left thumb #2 knuckle; 22. Right thumb #2 knuckle; 23. Left hip; 24. Right hip; 25. Left knee; 26. Right knee; 27. Left ankle; 28. Right ankle; 29. Left heel; 30. Right heel; 31. Left foot index; 32. Right foot index.

(a)



(b)

Figura 8: Topología de Mediapipe Pose 8a y esqueleto vectorial 8b.

6.3.3. Cálculo de ángulos en 3D

Como se ha mencionado, este procedimiento se llevó a cabo inicialmente en el software Unity Engine, en C#, para visualizar la reconstrucción de las coordenadas en 3D. Se utilizaron las funciones $Vector3.Angle()$, $Vector3.ProjectOnPlane()$, $Vector3.SignedAngle()$ y $Vector3.Cross()$ para calcular los ángulos de cada articulación, verificando el resultado obtenido con la estimación postural en tiempo real de Mediapipe Pose y la cámara web. Finalmente se replicaron dichas funciones en Pyhton $Angle(vec1, vec2)$:, $ProjectOnPlane(vector, planeNormal)$:, $SignedAngle(vec1, vec2, axis)$: y $np.Cross()$ que es nativa de la librería NumPy. A continuación, con fines explicativos, se presentan extrac-

tos y adaptaciones del código presente en la sección 11.5 Anexo: (Código Python) FuncionesPipeline.py).

1. **Cálculo de ángulos de los hombros:** Basado en el abordaje presentado por [23, 35] manera similar al ejemplo descrito en el último párrafo de la sección 3.1.4 Planos anatómicos y descripción del movimiento humano, que hace referencia a la Figura 6, se utilizó la proyección del segmento «brazo» o «humero» con respecto a los 3 planos anatómicos para obtener flexión de hombros (respecto al plano sagital), abducción (respecto al plano frontal o coronal) y la rotación, respecto al plano transverso. En este último, se consideraron valores negativos en la rotación interna y valores positivos en la rotación externa, donde el punto neutro coincide con el plano sagital del hombro respectivo. El código 2 muestra las funciones y detalla, con fines explicativos, el código utilizadoen el hombro izquierdo:

Código 2 Cálculo de ángulos de los hombros.

```
# PASO NÚMERO 3: CÁLCULO DE ÁNGULOS ENTRE VECTORES ÓSEOS Y EJES DE COORDENADAS DE LAS ARTICULACIONES (PARA HOMBROS) O ENTRE HUESOS (PARA CODO Y MUÑECA).
# Hombro izquierdo
# Proyecta el vector del brazo izquierdo en el plano sagital del hombro usa el vector perpendicular al plano (vector x)
vector_proyectado_sagital_lhombro = ProjectOnPlane(vector_brazo_izquierdo, vectorx_hombros_medios)
# Calcula el ángulo sagital izquierdo como el valor absoluto del ángulo entre el vector proyectado sagital y el eje y de los hombros medios (no se busca la extensión de hombro)
angulo_sagital_izquierdo = abs(SignedAngle(vector_proyectado_sagital_lhombro, -vectory_hombros_medios, vectorx_hombros_medios))

# Proyecta el vector del brazo izquierdo en el plano frontal de los hombros medios
vector_proyectado_frontal_lhombro = ProjectOnPlane(vector_brazo_izquierdo, vectorz_hombros_medios)
# Calcula el ángulo frontal izquierdo como el valor absoluto del ángulo entre el vector proyectado frontal y el eje y de los hombros medios
angulo_frontal_izquierdo = abs(ProjectOnPlane(vector_proyectado_frontal_lhombro, -vectory_hombros_medios, vectorz_hombros_medios))

# Proyecta el vector del brazo izquierdo en el plano transversal
vector_proyectado_transversal_lhombro = ProjectOnPlane(vector_brazo_izquierdo, -vectory_hombros_medios)
angulo_rotacion_izquierda=SignedAngle(vector_proyectado_transversal_lhombro,vectorz_hombros_medios,-vectory_hombros_medios)
```

Limitaciones:

Cálculo de ángulos de codos: Dataset y Mediapipe proveen 1 coordenada para el codo y no se dispone de las coordenadas para el «epicóndilo lateral» y «epicóndilo lateral» (identificados como EM y EL, en color celeste, en la Figura 9) para definir los sistemas de coordenadas locales recomendado por la Sociedad Internacional de Biomecánica [34], según se observa en la Figura 10. En consecuencia, con las coordenadas de hombro, codo y muñeca se definieron los vectores «húmero» y su proyección en 180° hacia el antebrazo, y en referencia vector «antebrazo», se calculó la magnitud angular entre los dos vectores que representan la flexión

de codo[14].

Cálculo de ángulos de muñeca: De manera similar, Dataset y Mediapipe proveen 1 coordenada para la muñeca y no se dispone de las coordenadas para el «estiloides radial» y «estiloides ulnar» utilizados para definir el sistema de coordenadas locales (Figura 9, en celeste). Por lo tanto, se generó un vector que representa la proyección de antebrazo segmento «antebrazo» en 180° hacia la mano y se calculó el ángulo absoluto en 3D, acorde a lo representado en la Figura 9

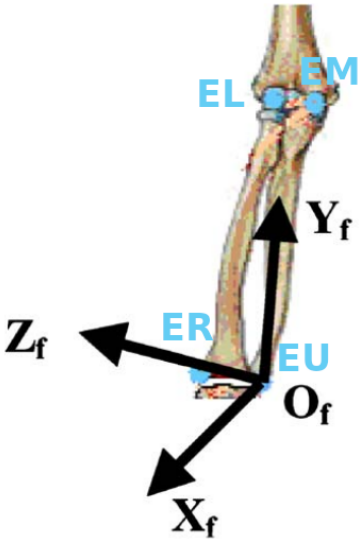


Figura 9: Definición del sistema de coordenadas locales para codos y muñecas, de acuerdo a la Sociedad Internacional de Biomecánica. Extraído de [34].
Los puntos anatómicos EL: Epicóndilo Lateral, EM: Epicóndilo Medial **(ambos ausentes en topología de Mediapipe y Dataset)**, permiten la definición del sistema de coordenadas locales en el codo. Por su parte, los puntos anatómicos ER: Estiloides Radial, EU: Estiloides Ulnar **(ambos ausentes en topología de Mediapipe y Dataset)**, permiten la definición del sistema de coordenadas locales de muñeca, el cual se presenta en detalle: se define el O_f origen estiloides ulnar, Y_f Vector Y: La línea que conecta la Estiloide Ulnar (EU) y el punto medio entre el epicóndilo lateral (EL) y el epicóndilo medial (EM), apuntando en dirección superior; X_f La línea perpendicular al plano formado por el Estiloide Ulnar (EU), el Estiloide Radial (EU) y el punto medio entre el Epicóndilo Lateral (EL) y el Epicóndilo Medial (EM), apuntando hacia adelante. Z_f Vector Z La línea común perpendicular al eje X_f y al eje Y_f, apuntando hacia la derecha. Extraído de [34]

[14]Para codo y muñeca, se replicó la función Vector3.Angle de Unity, cuyo resultado no proporciona información sobre la orientación relativa de los vectores en términos de derecha, izquierda, arriba o abajo. Simplemente representa la magnitud angular entre los dos vectores.

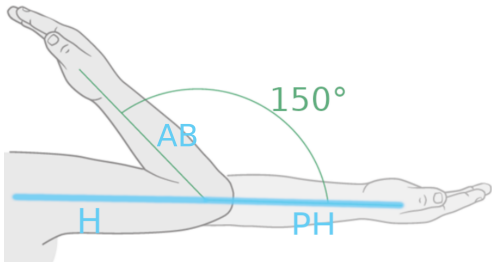


Figura 10: Cálculo de flexión de codo como ángulo absoluto entre la proyección del húmerio (PH) y el antebrazo (AB).

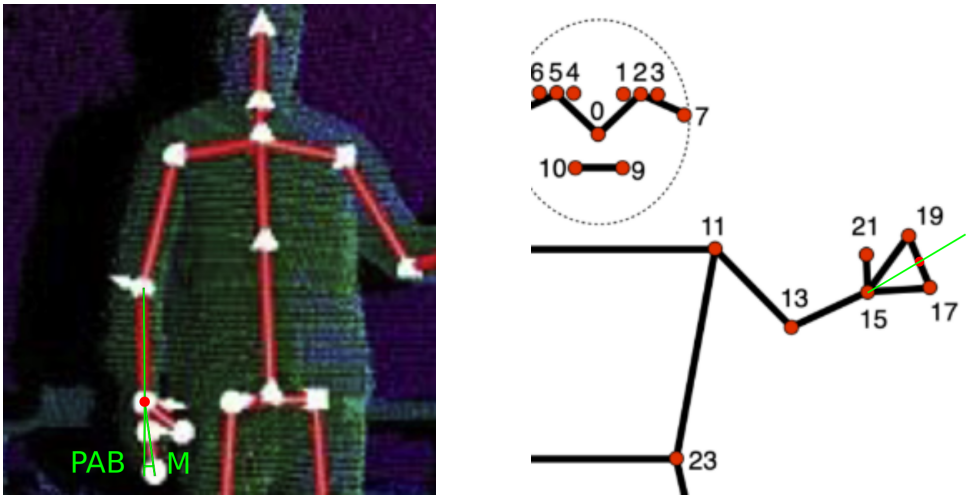
Debido a que en Mediapipe y Kinect V2 no se dispone de los puntos anatómicos Epicóndilo Lateral (EL) y Epicóndilo Medial (EM), que se detallan en la Figura 9 para la articulación del codo, se limita la creación del sistema de coordenadas locales indicados por la Sociedad Internacional de Biomecánica [34], por lo tanto, se calculó el ángulo absoluto la proyección del húmero, en la imagen PH: Proyección Húmero y el antebrazo (AB).

1. **Adaptaciones para Mediapipe en la topología en mano-dedos**: Es importante destacar que la topología de Dataset (Kinect V2) es distinta a Mediapipe en región las manos y dedos, por lo que en este último no se pudo replicar exactamente lo descripor por Manguishi et al. [23] en cuanto al cálculo de ángulo en muñeca, donde se utilizó el ángulo entre el vector que conecta el codo y la muñeca y el vector que conecta la muñeca con la punta de los dedos. En cambio en Mediapipe, se dispone del centro de la muñeca, la punta del dedo pulgar, la punta del dedo índice y la punta del dedo meñique. En este último grupo de datos, se definió el punto medio entre el índice y el dedo meñique como referecia para el cálculo de ángulo de flexión de muñeca.

Código 3 Cálculo de ángulos de codos y muñecas.

```
1
2 flexion_codo_izq= Angle(vector_antebrazo_izq, vector_brazo_izq_proyectado)
3 flexion_muneca_izq= Angle(vector_mano_izq,vector_antebrazo_izq_proyectado)
```

Calcula el ángulo de flexión del codo izquierdo como el ángulo entre el vector del antebrazo izquierdo y el vector brazo proyectado en 180° en dirección distal 10.



(a) (b)

Figura 11: Diferencias en la topología Muñeca-Mano-Dedos en Dataset 11a y Mediapipe 11b. Adaptación en Mediapipe para el cálculo del ángulo de muñeca.

Como se observa en la Figura 11a, se presenta la extremidad derecha de la reconstrucción del Dataset destacando el punto anatómico rojo como centro de la muñeca, la proyección del antebrazo (PAB) y la punta del dedo medio para cada extremidad (M). Por lo tanto, se consideró como ángulo de muñeca, el ángulo absoluto entre los vectores de la proyección del antebrazo (PAB) y el vector une el centro de la muñeca la punta del dedo medio (M), representados en las líneas de color verde. En cambio en Mediapipe, Figura 11b, se dispone del centro de la muñeca (15), la punta del dedo índice (19) y la punta del dedo meñique (17). En consecuencia, se definió como ángulo de muñeca el ángulo absoluto entre los vectores la proyección del antebrazo y el vector une el centro de la muñeca con el punto medio entre el índice y el dedo meñique (en la Figura 11b, ambos vectores quedan superpuestos representados en la línea de color verde).

Implementación para Mediapipe A continuación se presenta un extracto de la función Python llamada «$BodyPlanesAngles$», con su descripción de parámetros de entrada y retorno, la cual es resultado de replicar el código C# utilizado para elaborar el modelo humano virtual y calcular los ángulos de cada articulación en estudio. La versión completa se encuentra en la sección 11.5 Anexo: (Código Python) FuncionesPipeline.py.

Código 4 Extracto de la función Python $BodyPlanesAngles$ descripción de entrada, parámetros y retorno.

```
def BodyPlanesAngles(img, lmList, transparency=0.5, draw=False):
    """
    Calcula ángulos de los planos corporales a partir de las coordenadas de la
        postura.
    Parámetros:
    img (array): Imagen o video, para uso en tiempo real sobre videos o cámara
        web.
    lmList (list): Lista de coordenadas de puntos de referencia (landmarks) de
        la postura.
    transparency (float, opcional): Nivel de transparencia para la
        visualización (0-1) en videos.
    draw (bool, opcional): Si se deben dibujar los resultados en la imagen,
        para el uso en videos.

    Retorna:
    img (array): Imagen para su uso en videos o cálculo en tiempo real.
    angles (list): Lista de ángulos calculados en el siguiente orden:
    [sagital_derecho, frontal_derecho, transverso_derecho,
    derecho_codo_flexion, derecho_muneca_flexion,
    sagital_izquierdo, frontal_izquierdo, transverso_izquierdo,
    izquierdo_codo_flexion, izquierdo_muneca_flexion]
    """
```

Utilizado para elaborar modelo humano virtual y cálculo de ángulos. Puede usarse para proyectar las coordenadas en el video en tiempo real con el parámetro draw=True. La versión completa se encuentra en la sección 11.5 Anexo: (Código Python) FuncionesPipeline.py.

Por último, el mismo procesamiento se aplicó para el Dataset sin embargo, hubo que agregar las siguientes variaciones: se utilizó el archivo $.csv$ que detalla aquellos cuadros del video donde el participante se observaba de espaldas a la cámara y definieron 2 funciones casi idénticas, pero que cambian la lateralidad de izquierda a derecha y viceversa y, por último, estas funciones no permiten proyectar el resultado sobre los videos en tiempo real, debido a las diferencias en la unidad de medida (en metros y no en pixeles).

- La primera función «$BPAfront(lmList)$:» permite procesar aquellos cuadros en que el participante filmado se encuentra frente a la cámara, donde la izquierda y derecha de la persona no coinciden con la izquierda y derecha del campo visual.
- La segunda función «$BPABack(lmList)$:» permite procesar aquellos cuadros en que la izquierda y derecha del participante si coincide con la izquierda y derecha del campo visual, es decir, aquellos cuadros donde el sujeto filmado da la espalda a la cámara.

Ambas funciones, también presentes en la sección: 11.5 Anexo: (Código Python) FuncionesPipeline.py

6.3.4. Sincronización temporal

La sincronización temporal consistió en alinear el movimiento reproducido por modelo humano virtual del Dataset con el movimiento reproducido por el modelo humano de Mediapipe para que al comparar las variables cinemáticas coincidan temporalmente. Este paso fue necesario porque la cantidad de cuadros presentes en la anotación postural del Dataset era siempre menor a la cantidad de cuadros presentes en los videos, sobre los cuales Mediapipe Pose hizo su estimación de postura. Para llevar a cabo este

procedimiento, se utilizaron los archivos $.mat$ incluidos en el dataset que informan los 'timestamp' de los videos y de la postura anotada. Se buscó la diferencia entre ambos y dicha diferencia fue la cantidad de cuadros que se eliminaron al inicio del grupo de datos provenientes de Mediapipe, resultando curvas aparentemente alineadas. Sin embargo, la inspección visual detallada de cada curva hizo detectar que en algunos casos persistían 2 a 3 cuadros de diferencia, por lo que se creó un diccionario Python llamado $desfase\_dict$, con dicho ajuste final. La implementación de este procedimiento puede ser revisada en la sección: 11.5Anexo: (Código Python) FuncionesPipeline.py

6.3.5. Reducción de la agitación de las coordenadas

Al momento de reproducir el movimiento a través de los puntos anatómicos en espacio tridimensional de Unity Engine, fue posible constatar que cada punto anatómico se mostraba con un grado de agitación, oscilando en torno a cierta posición, incluso cuando la persona del video permanecía prácticamente inmóvil (corroborado al permanecer inmóvil y transmitir la señal de la cámar web en tiempo real). Para reducir dicha agitación, se promedió la ubicación de cada punto anatómico en 3 cuadros consecutivos, reduciendo considerablemente el grado de agitación y oscilación de la posición observada.

Para ello, dado que la estructura de datos del Dataset y Mediapipe son diferente en términos del número de puntos anatómicos en cada, se utilizaron las siguientes funciones:

Código 5 Función Python $SmoothDataset$ utilizada para suavizar la ubicación de las coordenadas 3D Dataset.

```
def SmoothDataset(body_logger_3d, window_size):
    """
    Suaviza los datos de un conjunto de coordenadas 3D a lo largo de una
        ventana deslizante.
    Entradas:
    - body_logger_3d (numpy.ndarray): Matriz tridimensional de coordenadas 3D.
    Cada dimensión representa [frames, participantes, landmarks].
    - window_size (int): Tamaño de la ventana deslizante para el suavizado.
    Salida:
    - smoothed_data (numpy.ndarray): Matriz suavizada de coordenadas 3D.
     Cada dimensión representa [frames_suavizados, participantes, landmarks].
    """
    total_frames = len(body_logger_3d)
    landmarks = len(body_logger_3d[0][0])
         if total_frames < window_size:
            raise ValueError("El tamaño de la ventana no puede ser mayor que el
                número de fotogramas.")                 smoothed_data =
                np.zeros((total_frames - window_size + 1,
                len(body_logger_3d[0]), landmarks))
        for i in range(total_frames - window_size + 1):
            for j in range(len(body_logger_3d[0])):
                for k in range(landmarks):
                coord_data = body_logger_3d[i:i+window_size, j, k]
                smoothed_data[i][j][k] = np.mean(coord_data)
    return smoothed_data
```

Código 6 Función Python $SmoothMediapipe$ utilizada para suavizar la ubicación de las coordenadas 3D Mediapipe.

```
def SmoothMediapipe(posture, window_size):
    """
    Suaviza los datos de postura detectados por el modelo Mediapipe a lo largo
        de una ventana deslizante.
    Entradas:
    - posture (list): Lista de datos de postura detectados por el modelo
        Mediapipe.
      Cada elemento de la lista representa una secuencia de fotogramas y
          contiene una matriz 3D (33 joints x 3 coordenadas).
    - window_size (int): Tamaño de la ventana deslizante para el suavizado.
    Salida:
    - posture_mediape_imported (list): Lista de datos de postura suavizados.
      Cada elemento de la lista contiene una secuencia de fotogramas
          suavizados, con una matriz 3D (33 joints x 3 coordenadas).
    """
    posture_mediape_imported = []
    for N_frames in range(len(posture)):
        if window_size <3:
            posture_mediape_imported.append(posture[N_frames])
        else:
            smoothed_posture = []
            for i in range(33):
                smoothed_joint = []
                for j in range(5):
                    smoothed_joint.append(sum([posture[k][i][j] for k in
                        range(N_frames-window_size+1, N_frames+1)]) //
                        window_size)
                smoothed_posture.append(smoothed_joint)
            posture_mediape_imported.append(smoothed_posture)
    return posture_mediape_imported
```

6.4. Procesamiento en Python

El procesamiento en Python consistió en obtener el cálculo de ángulos a partir de los 4 grupos de datos: $Dataset$, $Dataset_{Suavizado}$, $Mediapipe$ y $Mediapipe_{Suavizado}$, para ambas extremidades (replicando pre-procesaminto realizado en Unity Engine, como punto inicial). Luego, se comparó la cinemática de cada movimiento estudiado entre los 4 grupos de datos y se realizó un análisis comparativo respecto a los coeficientes de concordancia de Lin Pelliccia [36] y correlación r de Pearson en las curvas que describen los movimientos de hombros, codos y muñecas. Por último, con la finalidad de estudiar el comportamiento del contador automático de ATD y además, determinar la mejor combinación de parámetros de entrada UT y UA, para su posterior comparación estadística, se evaluaron 10 opciones de UT y 5 opciones de UA en los 4 grupos de datos, cuyos resultados se compararon con el «conteo humano de consenso», determinado por especialistas en ergonomía.

En cada paso mencionado, se prepararon gráficos, tablas comparativas y análisis estadísticos.

6.4.1. Dispersión de las curvas descritas por el movimiento de cada articulación.

Para comparar la cinemática de los movimientos de hombro (flexión, abducción y rotación), codo (flexión) y muñeca (flexión), se calculó el valor RMS de cada curva en los distintos grupos de datos y utilizó la librería $Seaborn$ de Python, para desplegar gráficos de cajas y para observar la dispersión de estos valores antes y después del suavizado, considerando los 20 videos.

6.4.2. Comparación de variables cinemáticas de la estimación de Mediapipe Pose y las *anotaciones presentes en el Dataset*

Con la finalidad de comparar el grado de concordancia entre los conjuntos de datos, se utilizó tanto el coeficiente estadístico de Concordancia de Lin, cuya implementación se presenta en la función del Código 7, como también el coeficiente de Correlación r de Pearson, utilizando el módulo $stats.pearsonr$ de la librería $SciPy$. Se comparó el movimiento de ambas extremidades mediante los ángulos hombros en la flexión (plano sagital), abducción (plano frontal) y rotaciones (plano transverso), flexión de codo y flexión de muñeca de los conjuntos de datos $Dataset_{Suavizado}$ y $Mediapipe_{Suavizado}$, ya que se tomó en consideración el efecto favorable del pre-procesamiento mediante el suavizado, entonces, no se incluyeron datos sin suavizar. Además, el valor obtenido en cada coeficiente se utilizó como parámetro de entrada en la función del Código 8 para clasificar ambos coeficientes.

Nota: Teniendo en consideración que el coeficiente de correlación r-de Pearson analiza la relación lineal entre los grupos de datos a comparar, lo cual escapa al propósito de este estudio, y además, estos datos tienen presencia de valores atípicos, se estimó que el coeficiente de concordancia de Lin es un mejor indicador para comparar las variables cinemáticas entre los grupos de datos. Los resultados se resumieron en una tabla para cada extremidad, que presenta los movimientos ordenados de manera decreciente en base a la cantidad de videos que caen en cada categoría de clasificación del coeficiente. Complementario a la sección de resultados, el detalle completo, que incluye ambos coeficientes y su respectiva clasificación, así como también el detalle de cada video y cada movimiento comparado se presentó en las secciones 11.7 Anexo: Detalle de resultados de concordancia y correlación entre ángulos de Dataset y Mediapipe y 11.7.3 Anexo: Orden decreciente de los videos según el el valor promedio de concordancia de Lin.

Código 7 Función para calcular el Coeficiente de Concordancia de Lin. Extraído de https://nirpyresearch.com/concordance-correlation-coefficient/

```python
def Lin_ccc(x, y):
    '''
    Calcula el Coeficiente de Concordancia de Lin (CCC).
    Parámetros:
    x (array-like): El primer conjunto de mediciones.
    y (array-like): El segundo conjunto de mediciones.
    Retorna:
    float: El Coeficiente de Concordancia de Lin (CCC) entre x e y.

    El CCC es una medida de la concordancia entre dos conjuntos de mediciones, x e y.
    Cuantifica el grado en que están de acuerdo y proporciona un valor entre -1 y 1,
    donde 1 indica una concordancia perfecta, 0 indica ninguna concordancia y -1 indica una discordancia perfecta.
    Extraído de https://nirpyresearch.com/concordance-correlation-coefficient/
    '''
    sxy = np.sum((x - x.mean()) * (y - y.mean())) / x.shape[0]
    lincc = 2 * sxy / (np.var(x) + np.var(y) + (x.mean() - y.mean())**2)
    return lincc
```

Código 8 Clasificación de los coeficientes de concordancia de Lin y correlación r-Pearson.

```python
def ClasificacionCoeficiente(r):
    """
    Clasifica un coeficiente de correlación en una categoría de acuerdo a su valor.
    Parámetros:
    r (float): Coeficiente de correlación a clasificar.
    Retorna:
    str: Categoría que describe el coeficiente de correlación.
    """
    if r >= 0.8:
        return "Alta (+)"
    elif r >= 0.4:
        return "Moderada (+)"
    elif r >= 0.2:
        return "Débil (+)"
    elif r >= 0:
        return "Muy Débil (+)"
    elif r >= -0.2:
        return "Muy Débil (-)"
    elif r >= -0.4:
        return "Débil (-)"
    elif r >= -0.8:
        return "Moderada (-)"
    else:
        return "Alta (-)"
```

De manera complementaria, incluyendo el coeficiente de concordancia de Lin y correlación r de Pearson, se prepararon gráficas para cada curva de movimiento y cada video, utilizando las librerías Matplotlib

y Plotly, las cuales se presentan en la sección 11.7.2 Anexo: Gráficos de las curvas del movimiento, ejemplos en base a la concordancia y correlación de este trabajo.

6.4.3. Contador automático de «acciones técnicas dinámicas»

Con la finalidad de llevar a cabo el conteo automático de ATD en cada extremidad a partir de los datos presentes en Dataset y Mediapipe, se utilizaron las curvas que describen el movimiento de hombro, codo y muñeca. Como se ha mencionado en la sección 3.1.3 Algoritmo para el conteo automático de «Acciones Técnicas Dinámicas», el algoritmo original de Taborri et al. requiere las curvas de los tres planos anatómicos de cada articulación, pero los grupos de datos utilizados no presentan la totalidad de puntos anatómicos requeridos para definir correctamente los sistemas de coordenadas locales, como se ha visto en la sección 6.3.3 Cálculo de ángulos en 3D (Figura 9). En consecuencia, en lugar de utilizar los movimientos en los tres planos para el codo y la muñeca, se implementó una adaptación del algoritmo para utilizar el ángulo absoluto del codo (entre los segmentos «brazo» y «antebrazo») y el ángulo absoluto en la muñeca (entre el «antebrazo» y la «mano»).

El funcionamiento del algoritmo adaptado consiste en identificar todas las máximas y mínimas de cada curva (columna) de los datos. Para ello, se utiliza la función $find\_peak$ de la librería $scipy.signal$[15], que también recibe umbrales como parámetros de entrada, los cuales fueron definidos en todo este estudio como $prominence = 10$ y $distance = 4$. A continuación, los parámetros de umbral del contador automático consideran válida cada máxima $(\mathrm{Max_i})$ que cumpla con las dos condiciones: (*i*) la distancia en amplitud con el mínimo sucesivo $(\mathrm{Min_i})$ es mayor que el UT $(\mathrm{t_a})$ y además (*ii*), la distancia temporal entre sucesivos máximos $T(\mathrm{Max_i}) - T(\mathrm{Max_{i-1}})$, es mayor que el UT $(\mathrm{t_t})$. Si el peak es válido, se almacena la posición temporal y la amplitud y se incrementa el contador de la acción técnica correspondiente y se almacenan en listas. Luego, se almacena el valor promedio del número de acciones $(N°acciones_{estimado})$ y finalmente, se retorna la lista de conteos promedio de acciones técnicas por video y la lista de resultados de ángulos procesados por video, la cual se utiliza más adelante para crear DataFrames y continuar el procesamiento en cuanto a los ángulos.

La adaptación del algoritmo de Taborri et al. se implementó en Python, mediante la función llamada $DynamicTechnicalActionsCounter$. Esta función inicia su procesamiento con los resultados del cálculo de ángulos provenientes de Dataset o Mediapipe, suavizados o sin suavizar, según corresponda. Recibe como parámetros de entrada, la función a importar los datos angulares, ya sea $PrepareAndLoadDataset$ o Mediapipe $PrepareAndLoadMediapipe$ (Código 9 o 10), indicando si dichos datos se van a suavizar o no. Además, se debe indicar el número de videos a analizar, el UT y UA a utilizar. Esta implementación se presenta y explica en el Código 11 (parte 1) y 12 (parte 2).

[15]para mayor detalles de la librería particular, se recomienda revisar la documentación en el sitio oficial https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.find_peaks.html.

Código 9 Función Python $PrepareAndLoadDataset$ descripción de entrada, parámetros y retorno.

```
def PrepareAndLoadDataset(i, side="derecho", Smooth=False,
    window_size=window_size):
    """
    Prepara y carga el grupo de datos del Dataset de un video.
    Parámetros:
    i (int): Número del video que se procesará.
    side (str, opcional): Lado del cuerpo a considerar para el análisis
        ("izquierdo" o "derecho").
    Smooth (bool, opcional): Si se debe suavizar la señal.
    window_size (int, opcional): Tamaño de ventana para el suavizado de la
        señal.

    Retorna:
    Ldatads_smooth (pd.DataFrame): DataFrame que contiene ángulos procesados
        y/o suavizados, según parámetro.
    technical_actions_ds_smooth (dict): Diccionario iniciado en cero para el
        conteo de acciones técnicas en el paso siguiente.
    """
```

Código 10 Función Python $PrepareAndLoadMediapipe$ descripción de entrada, parámetros y retorno.

```
def PrepareAndLoadMediapipe(i, side="izquierdo", Smooth=False,
    window_size=window_size):
    """
    Prepara y carga datos obtenidos de la detección de postura mediante
        Mediapipe para su análisis.
    Parámetros:
    i (int): Número del video que se procesará.
    side (str, opcional): Lado del cuerpo a considerar para el análisis
        ("izquierdo" o "derecho").
    Smooth (bool, opcional): Si se debe suavizar la señal.
    window_size (int, opcional): Tamaño de ventana para el suavizado de la
        señal.
    Retorna:
    Ldatamp_smooth (pd.DataFrame): DataFrame con ángulos procesados y
        suavizados (opcional) obtenidos de Mediapipe.
    technical_actions_mp_smooth (dict): Diccionario con valores técnicos
        relacionados con los ángulos.
    """
```

Es importante mencionar que la función $PrepareAndLoadDataset$ y $PrepareAndLoadMediapipe$ entregan como retorno los resultados en la misma estructura de datos.

Código 11 Función Python *DynamicTechnicalActionsCounter* Parte 1 - (continúa en la Página siguiente).

```
def DynamicTechnicalActionsCounter(LoadingFunction, n_VideosToAnalize,
    umbral_temporal, umbral_amplitud, dataset_type, side, Smooth=False,
    window_size=None, print_func=False):
    """
    Realiza el conteo de "acciones técnicas dinámicas» de la extremidad
        seleccionada basado en datos de Dataset o Mediapipe procesados y/o
        suavizados y genera resultados de análisis.
    Parámetros:
    LoadingFunction (función): Función que carga y procesa los datos para el
        análisis Dataset o Mediapipe.
    n_VideosToAnalize (int): Número de videos a analizar.
    umbral_temporal (float): Distancia temporal mínima (en segundos), entre
        sucesivos máximos válidos.
    umbral_amplitud (float): Distancia de amplitud (en grados) entre cada
        máximo y el sucesivo mínimo.
    dataset_type (str): Tipo de grupo de datos utilizado para el análisis
        "Dataset" o "Mediapipe".
    side (str): Estremidad a evaluar "izquierda" o "derecha".
    Smooth (bool, opcional): Si se debe suavizar la señal.
    window_size (int, opcional): Tamaño de ventana para el suavizado de la
        señal.
    print_func (bool): Indica si se deben imprimir resultados durante el
        análisis.
    Retorna:
    ConteoList (list): Lista de conteos promedio de acciones técnicas por video.
    ResultsList (list): Lista de resultados de ángulos procesados por video.
    """
    import pandas as pd
    from scipy.signal import find_peaks
    ConteoList = []
    ResultsList = []
    # Recorre cada video para realizar el análisis
    for i in range(n_VideosToAnalize):
        # Carga y procesa los datos del video utilizando la función de carga
            para dataset o mediapipe
        data, technical_actions = LoadingFunction(i, side=side, Smooth=Smooth,
            window_size=window_size)
        # Recorre cada columna (ángulo) en los datos, excepto la columna "time"
        for col in data.columns:
             if col == 'time':
                pass
            # Encuentra los máximos y mínimos de la señal
            max_peaks, _ = find_peaks(data[col], prominence=10, distance=4)
            min_peaks, _ = find_peaks(-data[col], prominence=10, distance=4)
            valid_peak = np.empty(0)
            # Recorre cada par de máximo y mínimo
            for f in range(min(len(min_peaks), len(max_peaks))):
                # Comprueba si la diferencia en amplitud entre los max(i) y
                    min(i) cumple con el umbral_amplitud
                # y si la distancia temporal entre max(i) y max(i-1) cumple con
                    el umbral_temporal.
                # Si el peak es válido, se almacena la posición temporal y la
                    amplitud
                # y se incrementa el contador de la acción técnica
                    correspondiente
```

Código 12 Función Python $DynamicTechnicalActionsCounter$ Parte2 - (continuación de la página anterior).

```
                if (data[col][max_peaks[f]] - data[col][min_peaks[f]] >
                    umbral_amplitud) and (data['time'][max_peaks[f]] -
                    data['time'][max_peaks[f - 1]] >
                    umbral_temporal):#type:ignore
                    valid_peak = np.append(valid_peak, max_peaks[f])
                    technical_actions[col] += 1
            # Agrega los resultados del video a la lista de resultados
            ResultsList.append({f"video {str(i+1)} {col}": data[col]})

            # Imprime los resultados, si se ha especificado
            if print_func:
                PrintAnglesOnBigLoop(data, col, i, max_peaks, min_peaks,
                    valid_peak, dataset_type, window_size)#type:ignore
        # Calcula el promedio de los conteos de acciones técnicas por video
        ConteoList.append(sum(technical_actions[key] for key in
            technical_actions) // len(technical_actions))
    # retorna las listas de resultados
    return ConteoList, ResultsList
```

6.4.4. Diseño de pruebas para el contador automático de acciones técnicas dinámicas

Debido a que el contador automático de ATD, requiere definir umbrales temporal y de amplitud, con la finalidad de determinar cuál combinación de umbrales presenta un menor error comparado al «conteo humano de consenso» y al FF que resulta en correspondencia a dicho conteo, se diseñaron pruebas para cada extremidad con los grupos de datos cinemáticos de los 20 videos disponibles en este estudio: $Dataset$, $Dataset_{Suavizado}$, $Mediapipe$ y $Mediapipe_{Suavizado}$.

Se utilizó la función $DynamicTechnicalActionsCounter$, que recibe como entre los parámetros de entrada UT y UA, y mediante un ciclo for se evaluaron todas las posibles combinaciones entre los siguientes umbrales: $umbral\ temporal(UT) = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]$ y $umbral\ amplitud(UA) = [5, 10, 15, 20, 25]$. En cada prueba, se realizó el conteo automático de ATD de cada extremidad para los 20 videos (40 conteos en cada prueba). Por lo tanto, los datos cinemáticos de $Dataset$, $Dataset_{Suavizado}$, $Mediapipe$ y $Mediapipe_{Suavizado}$ fueron utilizados para efectuar 50 pruebas de conteo automático de ATD con las distintas combinaciones de umbral. El resultado de las 400 pruebas se almacenaron en un archivo .csv para su análisis comparativo posterior.

El paso siguiente fue importar los conteos automáticos, resultantes de las pruebas, como objetos DataFrame de la librería Pandas. En cada extremidad (Izq y Der), se determinó el número de ATD por minuto Izq_{min} y Der_{min}, dividiento el respectivo conteo automático por la duración del video, en minutos. En consecuencia, el FF del Método OCRA Checklist se asignó en función de los conteos por minuto, en cada extremidad, utilizando la misma $FuncionFFrecuencia$ del Código: 1.

6.5. Análisis de resultados

El primer análisis fue exploratorio y visual, al reproducir el movimiento registrado en el Dataset y Mediapipe con las coordenadas articulares que dieron lugar a puntos anatómicos del espacio tridimensional

sincronizados en Unity Engine. Posteriormente, en Python, se replicó el pre-procesamiento y se utilizaron las librerías Plotly, Matplotlib, desplegar gráficos para comparar la dispersión de los ángulos de flexión de hombro, abducción de hombro, rotación de hombro, flexión de codo, flexión de muñeca, de cada extremidad en los 20 videos, en las distintas modalidades: $Dataset$, $Dataset_{Suavizado}$, $Mediapipe$ y $Mediapipe_{Suavizado}$ y visualizar el efecto del suavizado.

6.5.1. Análisis comparativo de variables cinemáticas entre Mediapipe *y Dataset*

Como se ha señalado en la subsección previa 6.4.2, para comparar el movimiento de ambas extremidades mediante los ángulos hombros en la flexión (plano sagital), abducción (plano frontal) y rotaciones (plano transverso), flexión de codo y flexión de muñeca $Dataset_{Suavizado}$ y $Mediapipe_{Suavizado}$ se utilizó el coeficiente estadístico de Concordancia de Lin y el coeficiente de Correlación r de Pearson. Además, el valor obtenido en cada coeficiente clasificó en categorías verbales, acorde a lo descrito en el Código 8.

6.5.2. Comparación entre las modalidades de conteo automático y el «conteo humano de consenso»

Los conteos de las pruebas realizadas con diferentes combinaciones de umbrales temporales y de amplitud, se importaron desde archivos $.CSV$ como objetos DataFrame de la biblioteca Pandas, en Python, y luego, se procesaron utilizando las bibliotecas NumPy, SciPy y Scikit-learn.

El primer paso en este punto, consistió en excluir del análisis al video n°3. Por lo tanto, cada prueba tuvo un tamaño muestral de 19 videos, procedimiento que se aplicó también al «conteo humano de consenso» ($ConteoH_{Consenso}$). Se aplicó la prueba de de normalidad de Shapiro-Wilk, con un valor $alpha = 0.05$, a todas las modalidades de conteo de forma individual, esto es: $ConteoH_{Consenso}$, $Dataset$, $Dataset_{Suavizado}$, $Mediapipe$ y $Mediapipe_{Suavizado}$, considerando todas las combinaciones de UT y UA.

El procesamiento incluyó cálculos comparativos entre el $Conteo_{Humano}$ con el resultado del conteo con todas combinaciones de umbrales temporal y de amplitud. Se utilizaron varios indicadores de error, tanto para los conteos de ATD como para la asignación respectiva del FF, tales como la raíz del error cuadrático medio, $RMSE$, en referencia a la abreviación inglesa del término Root Mean Square Error, el coeficiente de concordancia de Lin, el $error\ relativo$ y la prueba de significancia U de Mann-Whitney, cuya elección se debe al tamaño muestral (n=19) y la variabilidad de los conteos observada entre los evaluadores humanos. Estas pruebas se realizaron tanto para los conteos de ATD como para el FF resultante.

1. **Error relativo medio de los conteos:** representa una medida de discrepancia entre los datos, en términos porcentuales, comparando el «conteo humano de consenso» con las distintas modalidades de conteo automático.

$$Error\ Relativo\ Conteo(\%) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{|ConteoH_{Consenso} - Conteo_{GrupoConteo}|}{ConteoH_{Consenso}}\right) \times 100 \quad (6.1)$$

   Donde:
   - N es el número total de videos (conteos) a analizar en los grupos de conteo (19 al descontar el video n°3).
   - $ConteoH_{Consenso}$ es el resultado del Conteo Humano de Consenso en cada extremidad y en cada

video

- $Conteo_{GrupoConteo}$ representa el resultado del conteo automático llevado a cabo con cada extremidad y cada video del grupo de datos $Dataset$, $Dataset\_Suavizado$, $Mediapipe$ y $Mediapipe\_Suavizado$.

2. **Error relativo medio del «factor frecuencia» normalizado:** representa una medida de discrepancia en la asignación del FF, en términos porcentuales, comparando la asignación resultante del «conteo humano de consenso» con las distintas modalidades de conteo automático.

$$Error\ Relativo\ Frecuencia\ Norm(\%) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{|F.Frec_{Consenso} - F.Frec_{GrupoConteo}|}{9}\right) \quad (6.2)$$

Donde:

- N es el número total de videos (conteos) a analizar en los grupos de conteo (19 al descontar el video n°3).
- $F.Frec_{Consenso}$ es el FF que se asigna como resultado del Conteo Humano de Consenso en cada extremidad y cada video.
- $F.Frec_{GrupoConteo}$ representa el resultado del FF que se corresponde al resultado del conteo automático llevado a cabo con cada extremidad y cada video del grupo de datos $Dataset$, $Dataset\_Suavizado$, $Mediapipe$ y $Mediapipe\_Suavizado$.

   - **Nota:** Esta propuesta normaliza el error relativo en función del máximo valor posible en la asignación del FF, de esa forma se evita que el error se vuelva indefinido cuando el FF humano es cero (ver propuesta de Taborri et al. para el cálculo del error relativo en la Ecuación 3.1 de la sección 3.1.3 Algoritmo para el conteo automático de «Acciones Técnicas Dinámicas»).

3. **Error RMSE de los conteos (Error RMSE conteo):** se utilizó como medida de error entre los valores de ATD de cada grupo de datos y el conteo humano de consenso. Cuanto menor sea el RMSE, mejor el desempeño del contador automático y la combinación de umbrales utilizados.

$$Error\ RMSE\ conteo = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(ConteoH_{Consenso} - Conteo_{estimadoGrupoConteo})^2}$$

Donde:

- n es el número de muestras en el grupo de conteo, en este caso 19 videos (excluyendo el video n°3).
- $ConteoH_{Consenso}$ es el conteo humano de consenso para cada extremidad.
- $Conteo_{estimadoGrupoConteo}$ es conteo automático de cada extremidad en cada grupo de conteo, a saber: $Dataset$, $Dataset\_Suavizado$, $Mediapipe$ y $Mediapipe\_Suavizado$.

4. **Error RMSE de la frecuencia (Error RMSE frecuencia):** se utilizó como medida de error entre los valores de la asignación del FF de cada grupo de datos y el conteo humano de consenso.

$$Error\ RMSE\ frecuencia = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(F.Frec_{humano} - F.Frec_{estimadoGrupoConteo})^2}$$

Donde:
- n es el número de muestras en el grupo de conteo, en este caso 19 videos (excluyendo el video n°3).
- $F.Frec_{humano}$ es el FF resultado para cada extremidad del conteo humano de consenso.
- $Conteo_{estimadoGrupoConteo}$ es el FF resultante de cada extremidad en cada grupo de conteo automático, a saber: $Dataset$, $Dataset\_Suavizado$, $Mediapipe$ y $Mediapipe\_Suavizado$. A continuación se presenta el código Python 13 que contiene la implementación del análisis estadístico señalado.

Código 13 Comparación entre «conteo humano de consenso» y las modalidades de conteo automático de ATD. Parte 1 - (continúa en la Página siguiente).

```
import numpy as np
from FuncionesPipeline import Lin_ccc, ClasificacionCoeficiente, rounder,
    rmse_y_std
import pandas as pd
from scipy.stats import mannwhitneyu
if len(consenso.iloc[:,0])==20:
    consenso = consenso.drop(3)
# Diccionarios para cada resultado
ConteosAuto_Izq = {'Grupo': [], 'Extremidad': [], 'umbral_temporal': [],
    'umbral_amplitud': [],'Error RMSE conteo': [], 'Error RMSE Frecuencia':[],
                   'Lin_conteo':[],'ClasLin_conteo':[], 'Lin_frec':[],
                       'ClasLin_frec':[],'ErrorRel_conteo':[],
                       'ErrorRel_ffrec':[],'pValue_conteo':[],'pValue_ffrec':[]}
ConteosAuto_Der = {'Grupo': [], 'Extremidad': [], 'umbral_temporal': [],
    'umbral_amplitud': [], 'Error RMSE conteo': [], 'Error RMSE Frecuencia':[],
                  'Lin_conteo':[],'ClasLin_conteo':[], 'Lin_frec':[],
                     'ClasLin_frec':[],'ErrorRel_conteo':[],
                     'ErrorRel_ffrec':[],'pValue_conteo':[],'pValue_ffrec':[]}
# Iterar a través de las columnas de PruebasConteoAutomatico para efectuar los
    distintos cálculos.
for col in PruebasConteoAutomatico.columns:
    #lado izquierdo
    if "Izq" in col:
        error_relativoizq =  (abs(consenso['Izq_min'] -
            PruebasConteoAutomatico[col]) / consenso['Izq_min'] * 100).mean()
        stdizq =  (abs(consenso['Izq_min'] - PruebasConteoAutomatico[col]) /
            consenso['Izq_min'] * 100).std()
        error_relativoizqfrec = (abs(consenso['F.Frec.Izq'] - ffrec[col]) /
            9).mean()
        stdizqfrec = (abs(consenso['F.Frec.Izq'] - ffrec[col]) / 9).std()
        mae_izq, std_rmse_conteo_izq = rmse_y_std(consenso['Izq_min'],
            PruebasConteoAutomatico[col])
        ffrecuenciaizq, std_frec_izq = rmse_y_std(consenso['F.Frec.Izq'],
            ffrec[col])
        Lin_conteoizq = Lin_ccc(consenso['Izq_min'],
            PruebasConteoAutomatico[col])
        Lin_frecizq = Lin_ccc(consenso['F.Frec.Izq'], ffrec[col])
        clasLinconteoizq= ClasificacionCoeficiente(Lin_conteoizq)
        claslinfrecizq=ClasificacionCoeficiente(Lin_frecizq)
        estadistico_izq, pvalue_izq = mannwhitneyu(consenso['Izq_min'],
            PruebasConteoAutomatico[col])
        estadistico_izqfrec, pvalue_izqfrec =
            mannwhitneyu(consenso['F.Frec.Izq'], ffrec[col])
        # Agregar los valores correspondientes a los diccionarios
        ConteosAuto_Izq['Grupo'].append(col[0])
        ConteosAuto_Izq['Extremidad'].append(col[1])
        ConteosAuto_Izq['umbral_temporal'].append(f'{rounder(col[2])}')
        ConteosAuto_Izq['umbral_amplitud'].append(f'{rounder(col[3])}')
        ConteosAuto_Izq['Error RMSE conteo'].append(mae_izq)
        ConteosAuto_Izq['Error RMSE Frecuencia'].append(ffrecuenciaizq)
        ConteosAuto_Izq['Lin_conteo'].append(Lin_conteoizq)
        ConteosAuto_Izq['Lin_frec'].append(Lin_frecizq)
        ConteosAuto_Izq['ClasLin_conteo'].append(clasLinconteoizq)
        ConteosAuto_Izq['ClasLin_frec'].append(claslinfrecizq)
        ConteosAuto_Izq['ErrorRel_conteo'].append(error_relativoizq)
        ConteosAuto_Izq['ErrorRel_ffrec'].append(error_relativoizqfrec)
        ConteosAuto_Izq['pValue_conteo'].append(pvalue_izq)
        ConteosAuto_Izq['pValue_ffrec'].append(pvalue_izqfrec)
```

Código 14 Comparación entre «conteo humano de consenso» y las modalidades de conteo automático de ATD. Parte2 - (continuación de la página anterior).

```
    elif 'Der' in col:
        error_relativoder =  (abs(consenso['Der_min'] -
            PruebasConteoAutomatico[col]) / consenso['Der_min'] * 100).mean()
        stdDer =  (abs(consenso['Der_min'] - PruebasConteoAutomatico[col]) /
            consenso['Der_min'] * 100).std()
        error_relativoderfrec = (abs(consenso['F.Frec.Der'] - ffrec[col]) /
            9).mean()
        stdderfrec = (abs(consenso['F.Frec.Der'] - ffrec[col]) / 9).std()
        mae_der, std_rmse_conteo_der = rmse_y_std(consenso['Der_min'],
            PruebasConteoAutomatico[col])
        ffrecuenciader, std_frec_der = rmse_y_std(consenso['F.Frec.Der'],
            ffrec[col])
        Lin_conteoder = Lin_ccc(consenso['Der_min'],
            PruebasConteoAutomatico[col])
        Lin_frecder = Lin_ccc(consenso['F.Frec.Der'], ffrec[col])
        clasLinconteoder= ClasificacionCoeficiente(Lin_conteoder)
        claslinfrecder=ClasificacionCoeficiente(Lin_frecder)
        estadistico_der, pvalue_der = mannwhitneyu(consenso['Der_min'],
            PruebasConteoAutomatico[col])
        estadistico_derfrec, pvalue_derfrec =
            mannwhitneyu(consenso['F.Frec.Der'], ffrec[col])
        # Agregar los valores correspondientes a los diccionarios
        ConteosAuto_Der['Grupo'].append(col[0])
        ConteosAuto_Der['Extremidad'].append(col[1])
        ConteosAuto_Der['umbral_temporal'].append(f'{rounder(col[2])}')
        ConteosAuto_Der['umbral_amplitud'].append(f'{rounder(col[3])}')
        ConteosAuto_Der['Error RMSE conteo'].append(mae_der)
        ConteosAuto_Der['Error RMSE Frecuencia'].append(ffrecuenciader)
        ConteosAuto_Der['Lin_conteo'].append(Lin_conteoder)
        ConteosAuto_Der['Lin_frec'].append(Lin_frecder)
        ConteosAuto_Der['ClasLin_conteo'].append(clasLinconteoder)
        ConteosAuto_Der['ClasLin_frec'].append(claslinfrecder)
        ConteosAuto_Der['ErrorRel_conteo'].append(error_relativoder)
        ConteosAuto_Der['ErrorRel_ffrec'].append(error_relativoderfrec)
        ConteosAuto_Der['pValue_conteo'].append(pvalue_der)
        ConteosAuto_Der['pValue_ffrec'].append(pvalue_derfrec)
# Convertir los diccionarios en DataFrames
ConteosAuto_Izq_df = pd.DataFrame(ConteosAuto_Izq)
ConteosAuto_Der_df = pd.DataFrame(ConteosAuto_Der)
```

6.6. Aspectos éticos

Este trabajo fue aprobado por el Comité de Ética de Investigación en Seres Humanos de la Facultad de Medicina de la Universidad de Chile. La participación de evaluadores especialistas ha sido voluntaria y anónimizada sin registrar ninguna información personal, sin remuneración asociada y, con la posibilidad de desistir del proceso en cualquier instante previo a la publicación final de la tesis. Las herramientas e insumos tales como Python, junto con sus librerías y el Dataset utilizado, son gratuitos, de libre acceso y de código abierto, los cuales fueron obtenidas de las publicaciones directas de sus autores, en los sitios web oficiales, tales como MediapipePose, https://www.anaconda.com/products/distribution y https://data.mendeley.com/datasets/xwzzkxtf9s/2, respectivamente. El Software Unity Engine 2021.3.14f1 se utilizó la versión gratuita de uso personal.

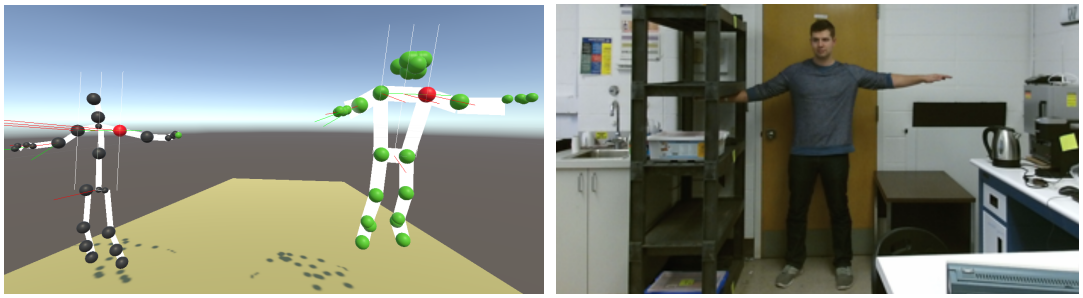
7. Resultados

7.1. Análisis cinemático

Para dar cumplimiento al objetivo específico n°1: «Obtener y comparar las variables cinemáticas obtenidas a partir de la anotación de las coordenadas articulares del Dataset con aquellas variables cinemáticas obtenidas por Mediapipe Pose, en los respectivos videos del Dataset», la primera comparación se realizó mediante la inspección visual del movimiento reconstruido por los puntos anatómicos en el espacio tridimensional de Dataset y Mediapipe en el software Unity Engine. A continuación, se analizó la dispersión de valor cuadrático médio, en inglés, root mean squared (RMS) de las curvas que describen los movimientos estudiados en grupos de datos suavizados y sin suavizar. Luego, se analizaron los Coeficientes de Concordancia de Lin y Correlación r-Pearson entre los ángulos obtenidos por Dataset Suavizado y Mediapipe Suavizado, analizando en qué movimientos, y en qué videos hubo mayor y menor concordancia entre los grupos de datos, lo que se expone en las Tablas que se despliegan en las secciones 11.7 Anexo: Detalle de resultados de concordancia y correlación entre ángulos de Dataset y Mediapipe y 11.7.3 Anexo: Orden decreciente de los videos según el el valor promedio de concordancia de Lin

7.1.1. El movimiento reconstruido por los puntos anatómicos de Mediapipe es más semejante al movimiento observado en los videos

La reconstrucción de los puntos anatómicos en el espacio 3D fue llevada a cabo en el software Unity Engine, lo que permitió ubicar, por separado, las coordenadas de los grupos de datos de Dataset y Mediapipe, a la vez que se trazaron líneas entre dichos puntos para definir tanto los segmentos corporales, así como también los vectores que definen los ejes de coordenadas locales para el análisis del movimiento las articulaciones estudiadas. Entre ellos, por ejemplo: el origen de cada reconstrucción es el punto medio entre las caderas. Los vectores que definen los ejes de coordenadas locales en hombros fueron destacados en color blanco (eje Y, vertical o céfalo-caudal), color verde (eje X, medio-lateral) y color rojo (eje Z, antero-posterior). Además, se destacó en color rojo el hombro izquierdo de ambos grupos de datos y, en particular en el Dataset, se destacó la punta de la mano izquierda de color verde para identificar visualmente la lateralidad. La reconstrucción de Dataset, en negro, y de Mediapipe, en verde, puede ser observada en la Figura 12a, lo que constituye una reconstrucción de la posición T (12b), adoptada al inicio de todos los videos.

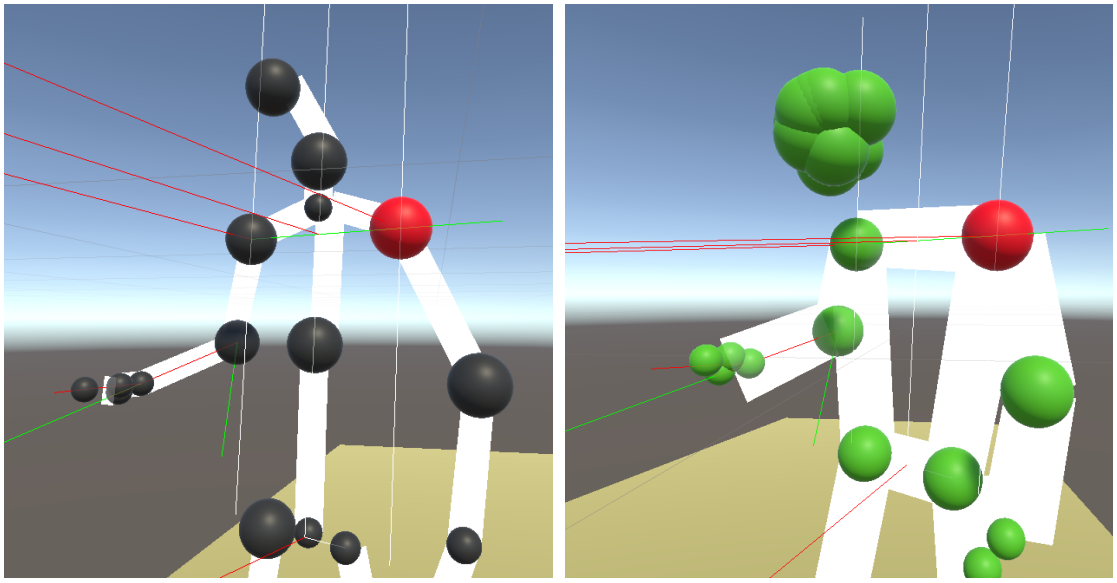


(a) (b)

Figura 12: Reconstrucciones de los puntos anatómicos en el espacio 3D de Unity Engine, en 12a, a partir de las coordenadas de Dataset, en negro, y Mediapipe, en verde, referente a la posición «T» de la escena 12b.

Con la finalidad de ejemplificar el grado de similitud entre el movimiento de las reconstrucciones y el movimiento observado en los videos, las imágenes de 12a y 12b representan la posición T adoptada al inicio de cada uno de los videos. La reconstrucciones de los puntos anatómicos de Dataset se identifican en color negro y aquellas de Mediapipe en color verde. Se observan líneas que sobresalen la reconstrucción para visualizar los vectores que defin en los Ejes de Movimiento: en color blanco,los ejes Y; en color verde, los ejes X del tronco y hombros, así como también, la proyección del segmento brazo y antebrazo; en color rojo, los ejes Z del tronco y hombros, junto con los segmentos antebrazo y mano. A partir de estos ejes se definen los planos anatómicos que permiten calcular los ángulos en los sistemas de coordenadas locales. 6.3.3 Cálculo de ángulos en 3D

Para analizar los movimientos del codo, se definieron vectores locales tales como la proyección distal del segmento brazo, en color verde, como prolongación en 180°, con su eje en el codo, para evaluar el ángulo respecto al antebrazo, en color rojo. Y en el caso de la muñeca, se evaluó el ángulo entre la prolongación del antebrazo, en color verde, con el segmento mano, en rojo. Los descrito, se observa en las Figuras 13a y 13b que representa los grupos Dataset y Mediapipe, mirando hacia su derecha para tomar o dejar una barra en nivel medio del estante, desde una perspectiva antero-lateral.



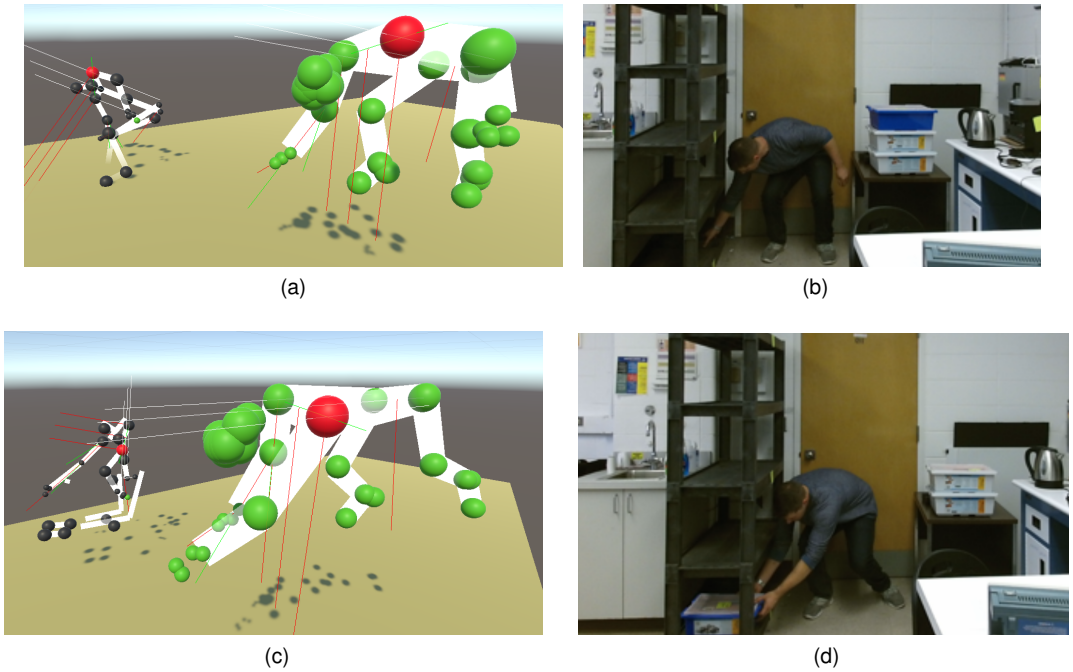
(a) (b)

Figura 13: Perspectiva cercana a la reconstrucción de Dataset, en negro, y Mediapipe, en verde, pertenecientes al video n°4.
En 13a y 13b, se observan las reconstrucciones desde una perspectiva más cercana para presentar en mayor detalle las líneas que sobresalen la reconstrucción, y que representan los vectores que definen los ejes de las coordenadas locales y planos de movimiento en hombros, codos y muñecas. En particular, se quiere destacar los vectores que dieron lugar a los ángulos en codos y muñecas para ambos grupos de datos y mostrar sus diferencias: en el caso de codos, el ángulo se calcula entre la proyección distal del brazo (verde) y el antebrazo (rojo); en el caso de la muñeca, la el ángulo se calcula entre proyección distal del antebrazo (verde) y el segmento mano (rojo). Cabe señalar, que cada grupo de datos posee puntos anatómicos distintos en el segmento mano-dedos. Por lo tanto, para el Dataset, el segmento mano une la muñeca con la punta del dedo medio, en cambio, en Mediapipe, el segmento mano une la muñeca con el punto medio entre el índice y el dedo meñique, cuyo detalle se expone en la sección de Métodos

La primera aproximación, previa al análisis estadístico, consistió en la inspección visual de la reconstrucción del movimiento en Unity con cada grupo de datos, desde múltiples perspectivas para identificar diferencias con el movimiento observado por las personas filmadas en los videos, así como también diferencias entre los grupos de datos. En primer lugar, se observó que cada punto anatómico se mostraba con un grado de agitación, oscilando en torno a cierta posición en ambos grupos de datos, y que dicha oscilación se reducía considerablemente al aplicar una media móvil de 3 cuadros, la cual se ajustó a 5 cuadros en el análisis final. En segundo lugar, las reconstrucciones del movimiento a partir de Mediapipe mostraban desplazamientos excesivos cuando los puntos se acercaban o se alejaban de la cámara, por lo tanto, se decidió reducir a $\frac{1}{3}$, aproximadamente, la magnitud del componente Z de cada una de las coordenadas presentadas por Mediapipe, mediante la división tipo $floor$, dividiendo por 3 dicho componente al momento de almacenar los datos en archivos $.CSV.$, lo cual se aplicó a todo el análisis realizado (ver sección 6.3.1). El efecto de el mencionado escalamiento al componente «Z» se puede visualizar en la sección 11.6 Anexo: Ajuste en el componente «Z» de las coordenadas de cada punto anatómico de Mediapipe Pose.

Mediapipe entregó reconstrucciones que se observaron estables y coherentes con la forma humana en todo momento. En cambio, las reconstrucciones a partir del Dataset mostraban grandes alteraciones

a la forma humana, en la mayoría de las posiciones agachado y, en menor grado, en posiciones de tronco erguido con los brazos elevados en abducción-flexión, para tomar o depositar la caja o la barra desde el nivel inferior o superior. La Figura 14a muestra una reconstrucción correcta y similar entre Dataset y Medipipe. En cambio, la Figura 14c presenta una recostrucción del Dataset alterada perdiendo la representación postural y, por su parte, en Mediapipe también se aprecia una desviación en la reconstrucción, exhibiendo al torso o tronco recto en el modelo humano virtual, lo que contrasta con la flexión a nivel de columna presentada en la imagen 14d, lo que ocurre debido a que Mediapipe posee la reconstrucción de puntos intermedios en la columna que permitan reproducir una flexión a ese nivel, en consecuencia, las flexiones de columna son ejecutadas a nivel de cadera, con la columna recta.



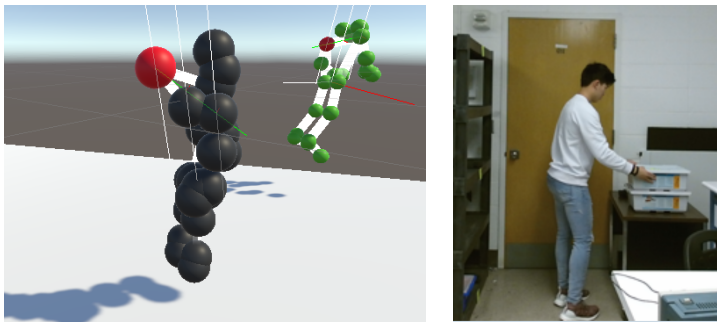
(a) (b)

(c) (d)

Figura 14: Reconstrucciones de Dataset y Mediapipe de la posición agachado en 2 instantes distintos del video n°4.
La Figura 14a presenta la reconstrucción de Dataset, en negro, y mediapipe, en verde, desde una perspectiva antero-lateral en referencia a la escena de la Figura 14b capturadas desde una perspectiva frontal. Es posible observar que los grupos de datos de Dataset y Mediapipe producen la reconstrucción de modelos humanos con un similitud entre ellos, y tienden a ser consistentes con la forma humana. La postura es agachado para acceder al nivel inferior del estante para manipular una barra.
En el caso de la reconstrucción de 14c, en referencia a la escena presentada en 14d, existen errores en la exactitud de la captura de Dataset, realizado con Kinect V2, los que llevan a importantes desalineaciones en la reconstrucción, se pierde la forma humana al unir los puntos anatómicos detectados. Y en el caso de Mediapipe, también se observa una reconstrucción algo inexacta, debido a que esta solución de visión computacional no ofrece puntos intermedios en la columna que permitan reproducir una flexión a ese nivel. En consecuencia, lo que se observa es que la flexión en la columna es representada en la articulación de caderas, aparentando un torso horizontalizado, distinto a lo observado en la escena 14d.

Por otro lado, se observaron defectos en la reconstrucción del Dataset cuando el punto anatómico no es capturado directamente por la cámara de la Kinect V2, debido a estar ocluido por otra parte del cuerpo o por algún elemento, como la caja que se transporta, donde la captura asigna valores inexactos, lo que a su vez afecta la reconstrucción, la definición de planos anatómicos y de los ángulos calculados. A modo

de ejemplo, si la persona se encuentra en perspectiva lateral respecto de la cámara capturando su perfil derecho, como extremidad más cercana a la cámara, ésta presenta una buena reconstrucción. En cambio la extremidad izquierda, que no se captura directemente, no obtiene una reconstrucción de calidad. Lo descrito, puede ser observado en la Figura 15a, que es una reconstrucción de la escena de la Figura 15b.



(a) (b)

Figura 15: Defecto de la oclusión en la reconstrucción del hombro izquierdo destacado en rojo en Dataset comparado a Mediapipe en 15a. Ambas en referencia a la escena de 15b.
En la Figura 15b, el hombro izquierdo no es capturado de manera directa por la cámara, siendo ocluído por el resto del cuerpo. Mediante la exploración y cambio de perspectiva del espacio virtual de Unity Engine se que, en este tipo de situaciones, Kinect V2 asigna valores inexactos a las coordenadas. En particular desde una perspectiva trasera, en la Figura 15a, se destaca la esfera de color rojo que representa el hombro izquierdo, desviado más a la izquierda de la posición natural afectando el modelo anatómico, lo que a su vez afecta la definición de ejes y planos anatómos y los ángulos calculados. Este problema es menos evidente en Mediapipe, donde la posición del hombro izquierdo (también en rojo) mantiene una posición coherente con el modelo anatómico, y se observa más robusto frente a la oclusión de segmentos corporales.

Tras visualizar la reconstrucción del movimiento hecha a partir de las coordenadas anotadas en el Dataset del video n°3, se decidió descartar el uso de estos datos para efectos estadísticos en el contexto de evaluar, más adelante, el conteo automático de ATD y su FF, ya que prácticamente toda la captura con Kinect V2 estuvo asociada a reconstrucciones que no reproducían un modelo humano (ver Figura 16). Por último, el suavizado de los datos con una media móvil de 5 cuadros consecutivos redujo notablemente la agitación de los puntos anatómicos y otorgó establilidad al movimiento de Dataset y Mediapipe modelado en Unity Engine.

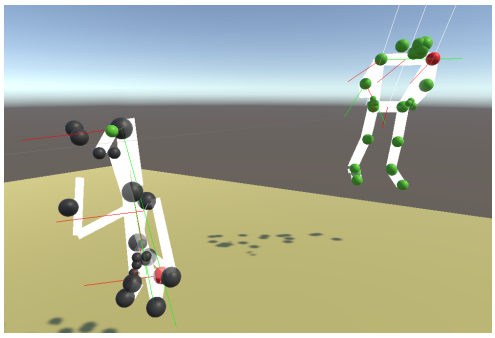
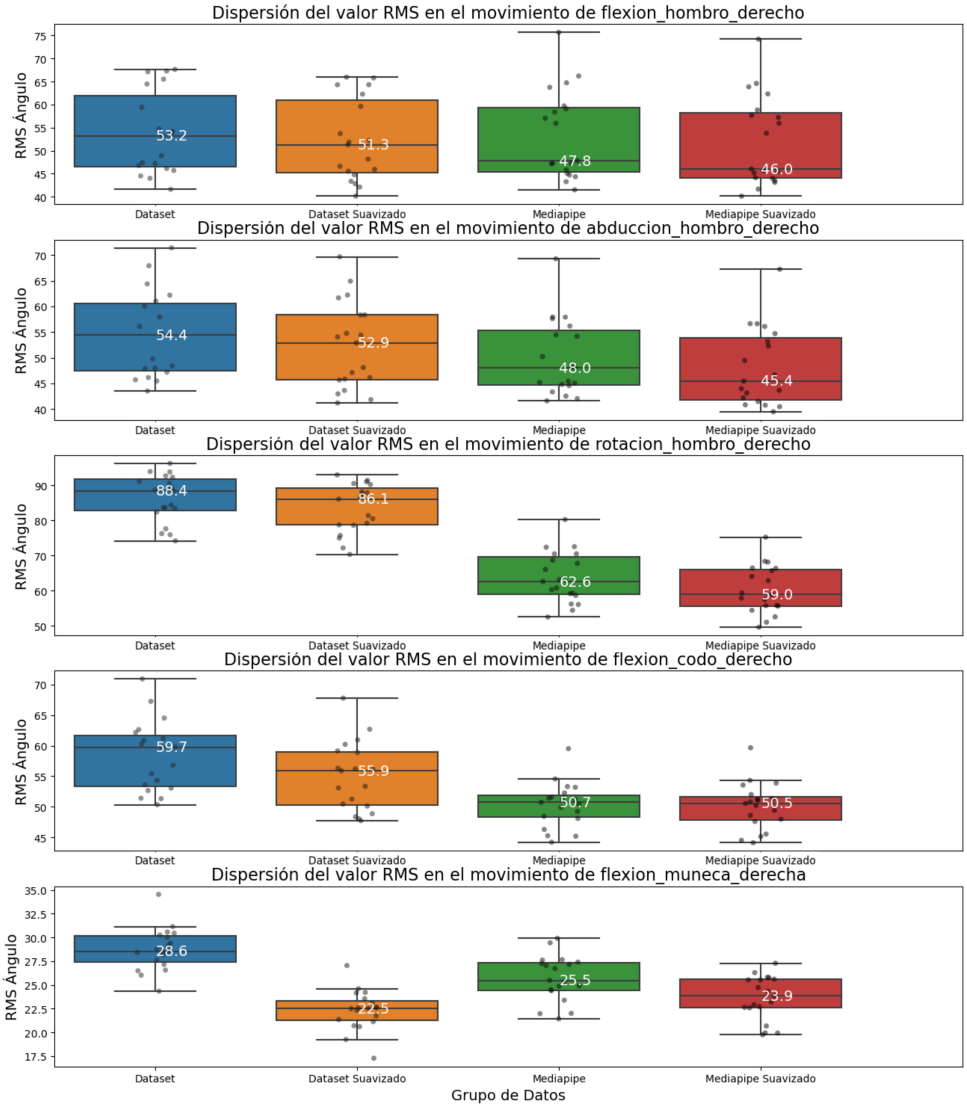


Figura 16: Errores en la captura postural realizada con Kinect V2 en la postura anotada en el Video3 del Dataset. Compárese con la reconstrucción de Mediapipe (verde)
Nótese, en color rojo, la ubicación errónea del hombro izquierdo de Dataset, casi a nivel de suelo. Estos errores de captura, hicieron descartar el video 3 para el análisis conjunto.

7.1.2. El efecto del suavizado y reducción del valor RMS de las curvas

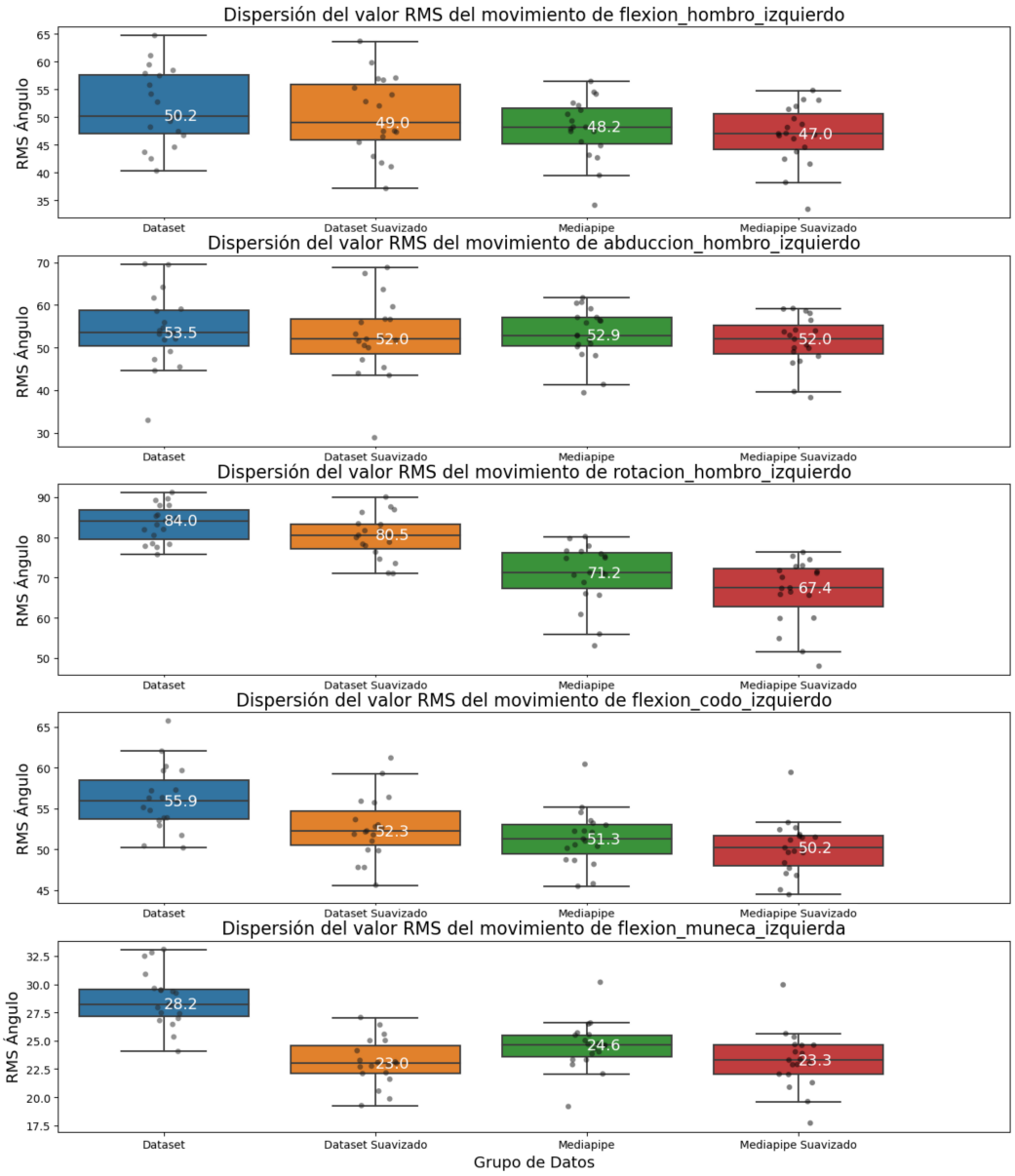
La dispersión del valor RMS de los ángulos de cada extremidad en cada movimiento estudiado, mostró el resultado de suavizar las curvas al aplicar una media móvil de 5 cuadros consecutivos en la etapa de pre-procesamiento: se redujo la mediana y los percentiles 25 y 75, tanto en el grupo de datos del Dataset como también en Mediapipe, lo cual puede ser observado en la Figura 18a y 18a y . Esto es coherente con la reconstrucción del movimiento con el modelamiento humano en Unity Engine, descrito en la sección anterior.



(a) Extremidad derecha.

Figura 17: Dispersión del valor RMS de las curvas y efecto del suavizado en los ángulos en Dataset y Mediapipe en extremidad derecha.

Al considerar los 19 videos (excluyendo el video n°3), se observa que al aplicar una media móvil de 5 cuadros en la ubicación de las coordenadas, se reducen respectivamente la mediana de los valores RMS de $Dataset$ comparado a $Dataset_{suavizado}$ y $Mediapipe$ comparado a $Mediapipe_{suavizado}$, así como también los percentiles 25 y 75. Esto es coherente con la reducción de la agitación observable en la reproducción del movimiento, descrita en la sección previa de resultados.



(a) Extremidad izquierda.

Figura 18: Dispersión del valor RMS de las curvas y efecto del suavizado en los ángulos en Dataset y Mediapipe en extremidad izquierda.

Al considerar los 19 videos (excluyendo el video n°3), se observa que al aplicar una media móvil de 5 cuadros en la ubicación de las coordenadas, se reducen respectivamente la mediana de los valores RMS de $Dataset$ comparado a $Dataset_{suavizado}$ y $Mediapipe$ comparado a $Mediapipe_{suavizado}$, así como también los percentiles 25 y 75. Esto es coherente con la reducción de la agitación observable en la reproducción del movimiento, descrita en la sección previa de resultados.

7.1.3. Tendencias en la concordancia de Lin entre el movimiento reconstruído por Dataset y Mediapipe.

Teniendo en consideración la agitación u oscilación en torno a cierta posición que se observó en los puntos anatómicos al momento de reproducir el movimiento de $Dataset$ (sin suavizar) y $Mediapipe$

(sin suavizar) en Unity Engine, se decidió hacer un análisis estadístico utilizando el coeficiente de concordancia de Lin y correlación r de Pearson entre las curvas obtenidas a partir de los ángulos que describen el movimiento de los puntos anatómicos en el espacio tridimensional construídos a partir del $Dataset_{Suavizado}$ y $Mediapipe_{Suavizado}$ en los 20 videos estudiados.

Ordenados de mayor a menor en base al coeficiente de concordancia de Lin, se observó que los movimientos de abducción y flexión de hombros, de ambas extremidades presentaron entre 18 a 19 de los registros en una clasifición de concordancia **alta (+)** o **moderada (+)**, mayor o igual a 0.8 o, mayor o igual a 0.4 respectivamente, las excepciones fueron los video n° 20 y n° 3 en ambas extremidades en categorías inferiores. A continuación, el movimiento de flexión de codo mostró 18 de sus registros en categoría de concordancia **moderada (+)** o **débil (+)**, mayor o igual a 0.2 y menor a 0.4, las excepciones fueron los videos n° 20 y n° 3 **muy débil (+)**, mayor o igual a 0 y menor a 0.2, y la extremidad izquierda mostró un registro en categoría **múy débil (-)**, mayor o igual a -0.2. Por su parte, los movimientos de rotación de hombro, se ubicaron con 19 o 18 de los registros en las categorías **débil (+)** o **muy débil (+)**, dependiendo de la extremidad. Mientras que el movimiento de muñecas tuvo un comportamiento distinto en cada extremidad: la muñeca derecha presentó 18 de sus registros en concordancia **muy débil (+)** y 2 registros en concordancia **muy débil (-)** y, la extremidad izquierda presentó 3 registros en concordancia **muy débil (+)**, 16 de sus registros en concordancia **muy débil (-)** y 1 registro en concordancia **débil (-)**, mayor o igual a -0.4 y menor a $-0,2$. Lo descrito puede ser observado en las Tablas 9b y 9a.

El detalle completo de resultados, referente al análisis de concordancia y correlación r de Pearson con su respectiva clasificación, se entrega en la sección 11.7 Anexo: Detalle de resultados de concordancia y correlación entre ángulos de Dataset y Mediapipe. Dicho Anexo, incorpora además, 2 gráficos por cada tipo de movimiento analizado con diferente coeficiente de concordancia y correlación, con la finalidad de entregar soporte visual al análisis realizado.

Movimiento	Clasificación Concordancia de Lin entre Ángulos de $Dataset_{Suavizado}$ y $Mediapipe_{Suavizado}$						
	Alta (+)	Moderada (+)	Débil (+)	Muy Débil (+)	Muy Débil (-)	Débil (-)	Total
Abduccion hombro der	3	15	1	1	-	-	20
Flexión hombro der	3	16	-	1	-	-	20
Flexión codo der	-	10	8	2	-	-	20
Rotación hombro der	-	1	10	8	1	-	20
Flexión muñeca der	-	-	-	18	2	-	20

(a) Extremidad derecha

Movimiento	Clasificación Concordancia de Lin entre Ángulos de $Dataset_{Suavizado}$ y $Mediapipe_{Suavizado}$						
	Alta (+)	Moderada (+)	Débil (+)	Muy Débil (+)	Muy Débil (-)	Débil (-)	Total
Abduccion hombro izq	6	12	1	1	-	-	20
Flexión hombro izq	6	13	-	1	-	-	20
Flexión codo izq	-	12	6	1	1	-	20
Rotación hombro izq	-	1	4	15	-	-	20
Flexión muñeca izq	-	-	-	3	16	1	20

(b) Extremidad izquierda

Tabla 9: Coeficiente de Concordancia de Lin entre los ángulos de obtenidos a partir de $Dataset_{Suavizado}$ y $Mediapipe_{Suavizado}$ en cada extremidad.
Las Tabla 9b presenta los resultados de extremidad izquierda y la Tabla 9a los resultados de la extremidad derecha. Las celdas indican la cantidad de videos cuya concordancia se clasifica en las respectivas categorías de Concordancia. Se observa que, para ambas extremidades, los movimientos de abducción y flexión tienen Alta (+) o Moderada (+) concordancia, mientras que los movimientos de flexión de codo se encuentran principalmente en las categorías Moderada (+) o Débil (+). Los movimientos de rotación tienden a tener concordancia Débil (+) o Muy Débil (+), y por último, la flexión de muñeca muestra una concordancia mayoritariamente Muy Débil (+) en extremidad derecha y mayoritariamente Muy Débil (-) en extremidad izquierda.

Para comparar, en conjunto, todos los movimientos estudiados en cada video entre los grupos de datos $Dataset_{Suavizado}$ y $Mediapipe_{Suavizado}$, se calculó el promedio y desviación estándar del coeficiente de concordancia de Lin del movimiento de hombro, codo y muñeca de cada extremidad, y se estableció un ranking de los 20 videos ordenados de mayor a menor, cuyo detalle se presenta en el la sección 11.7.3 Anexo: Orden decreciente de los videos según el el valor promedio de concordancia de Lin.

7.2. Conteo humano de consenso de Acciones Técnicas Dinámicas

Para dar cumplimiento al objetivo específico n°3, los resultados del conteo de ATD por minuto de cada extremidad, junto con su respectiva asignación del FF del OCRA Checklist, hecha por 3 profesionales especialistas en ergonomía a partir de la observación de los videos del Dataset, se presenta en la Tabla 10a. En complemento, el promedio y desviación estándar de los 3 participantes permitió establecer el $ConteoH_{Consenso}$ para cada extremidad, el cual se utilizó para calcular el respectivo FF, lo que es presentado en la Tabla 10b.

La prueba de normalidad de Shapiro-Wilk, con un valor $alpha = 0.05$, indica que el $ConteoH_{Consenso}$ de la extremidad derecha sigue una distribución similar a la normal ($p - value = 0.757$), en cambio, en la extremidad izquierda parece no seguir esta distribución $p - value = 0.010$. La prueba de normalidad aplicada a los conteos de cada participante tuvo un comportamiento similar en cada extremidad. Sin embargo, cuando se agruparon los conteos de cada extremidad de los 3 participantes en 2 conjuntos de 60 datos, la prueba de normalidad indica que ninguna extremidad tiene una distribución normal:

extremidad izquierda $p - value = 0.00018$ y extremidad derecha $p - value = 0.01084$

Los resultados del conteo humano de cada participante $P1, P2$ y $P3$, permiten señalar que el participante n°3 (P3), entregó conteos consistentemente más altos, para ambas extremidades y en todos los videos, que los otros dos participantes, lo cual puede ser observado en la Tabla 10a.

Los resultados del $ConteoH_{Consenso}$ indican que el mayor número de ATD se contabilizó en los video 9, video 6, video 4, video 1, con más de 30 ATD por minuto en la extremidad derecha, mientras que los videos 2, video 3, video 17, video 8 y video 19, presentaron menos de 20 ATD por minuto en promedio. Lo señalado, se presenta en la Tabla 10b. Cabe mencionar que los datos de ambas tablas han sido ordenadas de manera decreciente, en base al número de ATD de la extremidad derecha de la Tabla de consenso 10b, informado en la columna Der_min.

	P1				P2				P3			
n° video	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der
9	18	0.0	35	2.0	18	0.0	35	2.0	27	0.5	53	6.0
6	18	0.0	33	2.0	16	0.0	31	1.0	23	0.5	45	4.0
4	16	0.0	27	0.5	15	0.0	27	0.5	20	0.0	39	3.0
1	15	0.0	27	0.5	14	0.0	26	0.5	19	0.0	38	3.0
20	13	0.0	25	0.5	13	0.0	25	0.5	18	0.0	36	2.0
14	14	0.0	25	0.5	13	0.0	24	0.5	20	0.0	36	2.0
12	12	0.0	23	0.5	12	0.0	22	0.0	18	0.0	36	2.0
15	32	1.0	21	0.0	29	1.0	22	0.0	42	3.0	36	2.0
18	13	0.0	23	0.5	11	0.0	22	0.0	18	0.0	33	2.0
16	11	0.0	21	0.0	11	0.0	21	0.0	16	0.0	31	1.0
10	24	0.5	19	0.0	24	0.5	19	0.0	35	2.0	29	1.0
7	37	2.0	19	0.0	37	2.0	20	0.0	55	6.0	28	1.0
13	33	2.0	18	0.0	33	2.0	17	0.0	50	5.0	26	0.5
5	34	2.0	18	0.0	33	2.0	17	0.0	52	5.0	25	0.5
11	18	0.0	18	0.0	18	0.0	17	0.0	20	0.0	25	0.5
2	30	1.0	17	0.0	29	1.0	16	0.0	44	4.0	24	0.5
3	29	1.0	16	0.0	29	1.0	15	0.0	42	3.0	24	0.5
17	10	0.0	19	0.0	16	0.0	14	0.0	19	0.0	20	0.0
8	29	1.0	15	0.0	29	1.0	15	0.0	41	3.0	21	0.0
19	18	0.0	9	0.0	18	0.0	9	0.0	28	1.0	14	0.0

(a)

Conteo Humano de Consenso				
n° video	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der
9	21.0 ± 5.2	0.0	41.0 ± 10.4	3.0
6	19.0 ± 3.6	0.0	36.3 ± 7.6	2.0
4	17.0 ± 2.6	0.0	31.0 ± 6.9	1.0
1	16.0 ± 2.6	0.0	30.3 ± 6.7	1.0
20	14.7 ± 2.9	0.0	28.7 ± 6.4	1.0
14	15.7 ± 3.8	0.0	28.3 ± 6.7	1.0
12	14.0 ± 3.5	0.0	27.0 ± 7.8	0.5
15	34.3 ± 6.8	2.0	26.3 ± 8.4	0.5
18	14.0 ± 3.6	0.0	26.0 ± 6.1	0.5
16	12.7 ± 2.9	0.0	24.3 ± 5.8	0.5
10	27.7 ± 6.4	1.0	22.3 ± 5.8	0.0
7	43.0 ± 10.4	4.0	22.3 ± 4.9	0.0
13	38.7 ± 9.8	3.0	20.3 ± 4.9	0.0
5	39.7 ± 10.7	3.0	20.0 ± 4.4	0.0
11	18.7 ± 1.2	0.0	20.0 ± 4.4	0.0
2	34.3 ± 8.4	2.0	19.0 ± 4.4	0.0
3*	33.3 ± 7.5	2.0	18.3 ± 4.9	0.0
17	15.0 ± 4.6	0.0	17.7 ± 3.2	0.0
8	33.0 ± 6.9	2.0	17.0 ± 3.5	0.0
19	21.3 ± 5.8	0.0	10.7 ± 2.9	0.0

(b)

Tabla 10: Resultados de conteo humano ($P1$, $P2$ y $P3$) de «acciones técnicas dinámicas» por minuto y «Conteo Humano de Consenso», como promedio y desviación estándar, junto con el respectivo puntaje del FF correspondiente a cada extremidad, en cada uno de los videos.
Las Tablas 10a y 10b muestran el conteo de ATD por minuto para cada extremidad, junto con su asignación del FF del OCRA Checklist obtenidos por 3 profesionales especialistas en ergonomía $P1$, $P2$ y $P3$ a partir de la observación de los videos en el Dataset. Referente la Tabla 10a, el participante n°3 (P3), entregó conteos consistentemente más altos que los otros dos participantes, en cada extremidad y en todos los videos. La Tabla 10b presenta el promedio y la desviación estándar del consenso entre evaluadores para cada extremidad. Ambas tablas están ordenadas de manera decreciente según el número de ATD promedio (consenso) en la extremidad derecha Der_min. Los primeros 4 videos de la parte alta de la tabla superan las 30 ATD por minuto, cuyo máximo es 41 con una desviación estándar de ± 10 y los últimos 4 videos de la parte baja de la tabla no superan las 20 ATD en promedio. También es posible observar que las desviaciones estándar de la parte alta de la tabla tienden a ser mayores a la parte baja de la tabla. Es decir, que existe una tendencia a aumentar la dispersión de los conteos de consenso cuando la cantidad de ATD contabilizadas aumenta.
*Se destaca en color rojizo los resultados referentes al video n°3, ya que no se consideraron en la comparación con los grupos de datos de conteo automático (ver sección siguiente).

7.3. Conteo automático de «Acciones Técnicas Dinámicas»

En este trabajo se adaptó el algoritmo propuesto por Taborri et al. [14], para realizar la tarea de contar ATD con visión computacional, el detalle de su implementación se describe en sección 6.4.3 Contador

automático de «acciones técnicas dinámicas». En la Figura 19 se presenta como ejemplo gráfico, parte del proceso de conteo, a través de la identificación de peaks válidos, como «Valid_Peak», en amarillo, en una de las curvas que se analizaron para determinar las ATD en ese registro. Este ejemplo particular, utiliza el ángulo del hombro izquierdo en el movimiento de abducción, con parámetros de UT de 800 milisegundos y UA de 10°, pero deben incorporarse al mismo procesamiento el resto de movimientos de la extremidad, promediarlos en todo el registro, y dividirlo por la duración del registro, en minutos, para obtener el conteo de ATD por minuto.

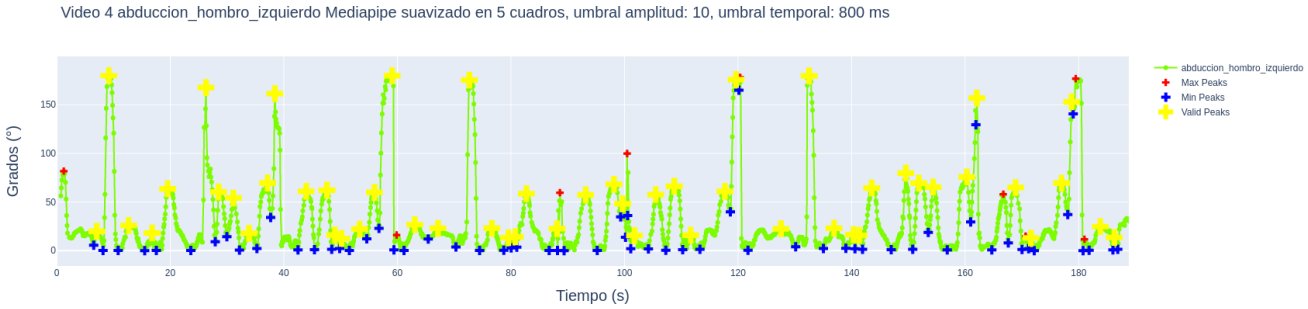


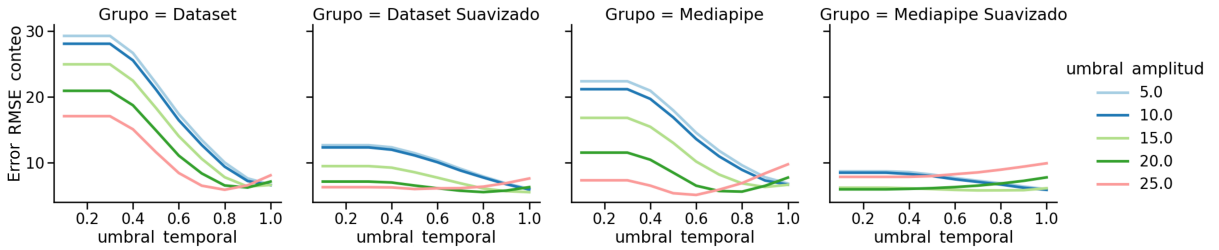
Figura 19: Ejemplo de detección máximos válidos «Valid Peaks» en una señal utilizando el contador automático.
La detección se realizó utilizando los datos angulares que describen movimiento de abducción hombro izquierdo del video n° 4 de $Mediapipe_{Suavizado}$. Las coordenadas articulares se obtuvieron con Mediapipe Pose, pre-procesado los datos con un suavizado de las coordenadas mediante un filtro media móvil de 5 cuadros consecutivos. Se configuraron los parámetros de UA en 10° y un UT de 800 milisegundos. El resultado de la detección destaca en amarillo los máximos válidos como $Valid\_Peaks$. Los máximos (Max_Peaks) que no cumplen con los dos criterios del contador automático (ver sección 6.4.3), se mantienen en color rojo, los peaks mínimos Min_Peaks se destacan en color azul. Por último, cabe señalar que para considerar una «acción técnica dinámica» válida para cada extremidad, se ejecuta este procedimiento con todas las curvas del hombro, codo y muñeca, respectivamente. Luego se promedian los valores obtenidos para cada extremidad, considerando dichas curvas, y dicho promedio es dividido por la duración del registro, en minutos, obteniéndose el número de acciones técnicas por minuto.

7.3.1. No se halló una combinación de umbral temporal y umbral amplitud con el mejor desempeño en todos los indicadores

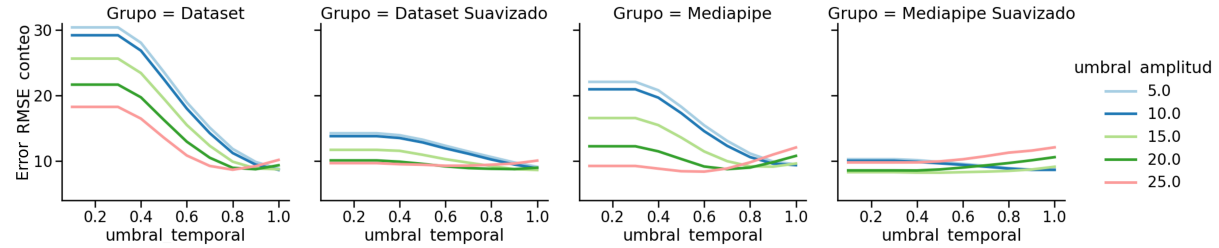
Con el propósito de buscar el menor error entre las posibles combinaciones de UT y de UA, y también, para determinar la combinación de umbrales de mejor desempeño para el contador automático, se llevaron a cabo pruebas con 50 combinaciones de umbrales en los 4 grupos de datos disponibles: $Dataset$, $Dataset_{Suavizado}$, $Mediapipe$ y $Mediapipe_{Suavizado}$, originando 400 pruebas (200 por cada extremidad). Cada prueba consistió en evaluar el conteo automático de cada extremidad en 19 videos, al excluir del análisis los resultados del video n°3.

La evaluación de normalidad de los datos, indicó que el $96.5\,\%(n = 386)$ de las pruebas de conteo automático parecen seguir una distribución normal, y el $4.5\,\%(n = 14)$ restante parecen no seguir una distribución normal (los 14 indicados, en la extremidad izquierda). Por su parte, en la asignación del FF, sólo el $15.75\,\%(n = 63)$ de las pruebas parecen seguir una distribución normal y el $84.25\,\%(n = 337)$ de las pruebas de conteo automáticos parecen no seguir una distribución normal. Los resultados se presentan en las Figuras 34 y 35 incluidas en la sección 11.8.1 Anexo: Análisis de normalidad en las pruebas combinación de umbral.

Los principales hallazgos indican que, para ambos grupos de datos: $Dataset$ y $Mediapipe$, el preprocesamiento para reducir la agitación de los puntos anatómicos aplicando un suavizado mediante una media móvil de las coordenadas de los puntos anatómicos de 5 cuadros consecutivos, es capaz de reducir la magnitud del máximo error $RMSE$ en la región de más bajo UT, en cada una de las curvas, donde cada curva refleja un UA distinto. Este procedimiento redujo la pendiente de dichas curvas de cada grupo suavizado, comparado al mismo grupo sin suavizar. Esto significa que el suavizado reduce la variación del error, haciéndolo más estable y menos dependiente del UT. Lo señalado es válido tanto para el RMSE de los conteos «$Error\ RMSE\ conteo$», como para el RMSE del FF «$Error\ RMSE\ frecuencia$» de ambas extremidades, lo cual puede ser observado en las Figuras 20 y 21.



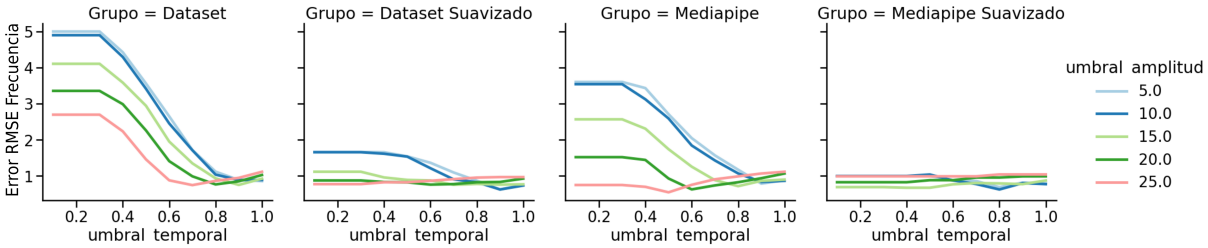
(a) Extremidad derecha



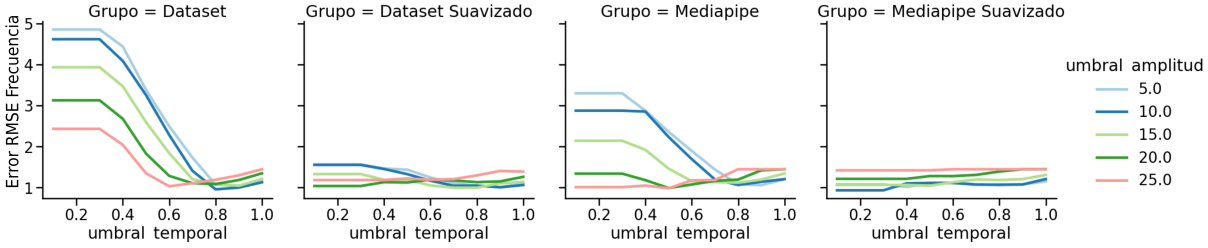
(b) Extremidad izquierda

Figura 20: Variación del error RMSE del conteo automático $Error\ RMSE\ conteo$, en cada extremidad, al combinar 10 umbrales temporales y 5 umbrales de amplitud, en 50 pruebas por cada grupo de datos.

En la Figura 20, se presenta una comparación visual de los errores en función de diferentes umbrales temporales y los diferentes umbrales de amplitud representados por el color de cada curva. Los gráficos de la Figura 20a muestran la extremidad derecha y los gráficos de la Figura 20b la extremidad izquierda. Cada subgráfico corresponde a un grupo de datos cinemáticos específicos $Dataset, Dataset_{Suavizado}, Mediapipe$ y $Mediapipe_{Suavizado}$ y representa el cómo varían los errores en función de los umbrales seleccionados, reflejando el desempeño del contador automático. Es posible observar que la variación en el error es menor al utilizar datos suavizados: la curva se vuelve más horizontal, en comparación a los datos sin suavizar, indicando que hay una menor dependencia del UT, a la vez que la diferencia entre los umbrales de amplitud se hace menor. Se observa también, que en datos sin suavizar el error es mayor cuando el UT es bajo (parte izquierda del gráfico) y presenta una pendiente negativa cuando aumenta el UT, hasta alcanzar su mínimo alrededor del UT de 0.8 aproximadamente. Respecto al UA, para Mediapipe suavizado, se logra observar que 10 y 15 grados suelen entregar los mejores resultados en la mayoría de las pruebas de UT. En cambio, para Dataset Suavizado, el mejor desempeño se da con umbrales más altos: 20 y 25 grados, con UT entre los 0.6 a 0.8 aproximadamente. Para el caso de UT mayores a 0.6 el error es menor con 10 y 15 grados de UA respectivamente.



(a) Extremidad derecha



(b) Extremidad Izquierda

Figura 21: Variación del $Error\ RMSE\ frecuencia$, referente al FF en cada extremidad, al combinar 10 umbrales temporales y 5 umbrales de amplitud, en 50 pruebas por cada grupo de datos.
En la Figura 21, se presenta una comparación visual de los errores RMSE de la asignación del FF del Método OCRA Checklist, en función de diferentes umbrales temporales y los diferentes umbrales de amplitud representados por el color de cada curva. Los gráficos de la Figura 21a muestran la extremidad derecha y los gráficos de la Figura 21b la extremidad izquierda. Cada subgráfico corresponde a un grupo de datos cinemáticos específicos $Dataset, Dataset_{Suavizado}, Mediapipe$ y $Mediapipe_{Suavizado}$ y representa el cómo varían los errores del FF en función de los umbrales seleccionados, reflejando el desempeño del contador automático. Es posible observar que la variación en el error es menor al utilizar datos suavizados: la curva se vuelve más horizontal, en comparación a los datos sin suavizar, indicando que hay una menor dependencia del UT, a la vez que la diferencia entre los umbrales de amplitud se hace menor. Se observa también, que en datos sin suavizar el error es mayor cuando el UT es bajo (parte izquierda del gráfico) y presenta una pendiente negativa cuando aumenta el UT, hasta alcanzar su mínimo alrededor del UT de 0.5 y 0.8 aproximadamente. Respecto al UA, para Mediapipe suavizado, se logra observar que 10 y 15 grados suelen entregar los mejores resultados en la mayoría de las pruebas de UT. En cambio, para Dataset Suavizado, el mejor desempeño se da con umbrales más altos: 20 y 25 grados con umbrales temporales bajos de 0.2 a 0.4, con UT entre los 0.8 a 0.9. En este caso, el error es menor con 10 y 15 grados de UA respectivamente.

Al explorar los resultados, dispuestos en tablas, fue posible hallar cuál combinación de umbrales presentó el menor error «$Error\ RMSE\ conteo$» y al error asociado al FF «$Error\ RMSE\ frecuencia$», sin embargo dicha combinación es única en referencia a cada grupo de datos para la extremidad respectiva y para el indicador de conteo o frecuencia respectivo. Dicho de otro modo, cada extremidad, cada grupo de datos y cada indicador de error presentó una combinación de umbrales distinta que arrojó el menor error. Además, hubo otras combinaciones de umbral con resultados cercanos en términos del error. Lo señalado se presenta en las Tablas de la sección 11.8 Anexo: Detalle de resultados conteo automático de «acciones técnicas dinámicas».

7.3.2. Selección de umbrales óptimos: un balance entre los indicadores de «Error RMSE conteo» y «Error RMSE frecuencia»

Como se ha señalado, no se pudo hallar una única combinación de UT y UA que ofrezca el menor error para ambas extremidades, entre todos los grupos de datos, ni para todos los indicadores de desempeño («$Error\ RMSE\ conteo$» y «$Error\ RMSE\ frecuencia$»). En consecuencia, se decidió

calcular los coeficientes de «concordancia de Lin de los conteos» (Lin_conteo) y «concordancia de Lin del FF» (Lin_frec), entre los conteos automáticos y el $ConteoH_{Consenso}$, con la finalidad de determinar qué combinación de umbrales es óptima en términos de ofrecer un balance entre los indicadores de error. Se ordenó de manera decreciente los resultados en función de los coeficientes Lin_conteo y Lin_frec y, se seleccionó para cada extremidad y en cada grupo de datos aquella combinación de umbrales con la clasificación más alta para ambos (Lin_conteo y Lin_frec), en simultáneo.

Para explicar esta selección, a modo de ejemplo, en el caso del grupo $Mediapipe_{Suavizado}$ la combinación de $UT = 0.6\ s$ y $UA = 15°$ entrega un «$Error\ RMSE\ conteo = 5.9 \pm 2.8$» y «$Error\ RMSE\ frecuencia = 0.8 \pm 0.5$», al revisar el respectivo puntaje y clasificación del coeficiente de $Lin\_conteo = 0.46$ y $Lin\_frec = 0.29$, las clasificaciones son $Moderada\ (+)$ y $Débil\ (+)$, por lo que su desempeño se juzgó menos balanceado que la combinación $UT = 0.5\ s$ y $UA = 15°$, que ofrece clasificaciones del coeficiente de concordancia de Lin $Moderada\ (+)$ y $Moderada\ (+)$ para los conteos $Lin\_conteo = 0.46$ y para la asignación del FF $Lin\_frec = 0.49$ (en este caso los valores de «$Error\ RMSE\ conteo =$» son 5.9 ± 2.8» y «$Error\ RMSE\ frecuencia = 0.7 \pm 0.4$».

Bajo el criterio descrito, se seleccionó la combinación de umbral óptima para cada extremidad y para cada grupo de datos. El mejor desempeño entre todos los grupos de datos se observó en $Mediapipe$, en término de los coeficientes de concordancia de Lin, con la combinación de $UT = 0.5\ s$ y $UA = 25°$ en la extremidadad derecha, y en la extremidad izquierda $UT = 0.1\ s$, $0.2\ s$ y $0.3\ s$ combinados con el $UA = 25°$, ya que los 3 umbrales temporales entregaron exactemente los mismos resultados. En segundo lugar, para $Mediapipe_{Suavizado}$ se seleccionaron los $UT = 0.5\ s$ y $UA = 15°$ y en la extremidad izquierda $UT = 0.4$ s y $UA = 15°$. En tercer lugar, para $Dataset_{Suavizado}$, se seleccionaron los $UT = 1\ s$ y $UA = 15°$ y la extremidad izquierda $UT = 1\ s$ y $UA = 10°$. Y por último, en $Dataset$, se seleccionaron los $UT = 0.8\ s$ y $UA = 25°$ y en la extremidad izquierda $UT = 1\ s$ y $UA = 10°$. Lo descrito, se presenta en la Tabla 11, destacando en **negrita** las señaladas combinaciones de umbral. Por último, el detalle completo de estos resultados se expone la Tabla 17 de la sección 11.8.3 Anexo: Resultados de las distintas combinaciones de Umbral Temporal y Umbral de Amplitud, destacando los mismos resultados en color verde.

	Variables	Extremidad	Dataset	Dataset Suavizado	Mediapipe	Mediapipe Suavizado
	Umbrales óptimos (segundos-grados)	**Der**	**0.8 s - 25°**	**1 s - 15°**	**0.5 s - 25°**	**0.5 s - 15°**
		Izq	**1 s - 10°**	**1 s - 10°**	**0.1 s, 0.2 s, 0.3 s - 25° ‡**	**0.4 s - 15°**
Conteos	Error RMSE conteo	**Der**	5.9 ± 3.4	5.5 ± 3	5.4± 2.5	5.9 ± 2.8
		Izq	8.6 ± 3.9	8.9 ± 3.7	9.3 ± 4.7	8.2 ± 3.9
	Error Relativo Conteo	**Der**	21.1 ± 17.9	20.3± 15.6	22.3 ± 16.7	24.2 ± 16.9
		Izq	37.9 ± 25.4	42 ± 30.4	44.2 ± 35	35 ± 23.6
	P-Value Conteos (U Mann-Whitney)$\sim$	**Der**	0.95	0.98	0.22	0.33
		Izq	0.17	0.14	0.07	0.29
	Concordancia Lin Conteo	**Der**	0.36 Débil (+)	0.47 Moderada (+)	0.58 Moderada (+)	0.46 Moderada (+)
		Izq	0.28 Débil (+)	0.31 Débil (+)	0.42 Moderada (+)	0.42 Moderada (+)
Factor Frecuencia	Error RMSE Frecuencia	**Der**	0.9 ± 0.7	0.7 ± 0.5	0.6 ± 0.3	0.7 ± 0.4
		Izq	1.1 ± 0.8	1.1 ± 0.6	1.1 ± 0.7	1.2 ± 0.9
	Error Relativo Frecuencia Norm	**Der**	5.6 ± 6.7	5.3 ± 5.4	4.9 ± 3.7	5.9 ± 4.3
		Izq	9.9 ± 9.9	10.2 ± 8.5	9.4 ± 8.0	9.1± 9.8
	P-Value Frecuencia (U Mann-Whitney)$\sim$	**Der**	0.69	0.91	0.11	0.19
		Izq	0.35	0.25	0.1	0.67
	Concordancia Lin Frecuencia	**Der**	0.1 Muy Débil (+)	0.38 Débil (+)	0.67 Moderada (+)	0.49 Moderada (+)
		Izq	0.18 Muy Débil (+)	0.24 Débil (+)	0.44 Moderada (+)	0.29 Débil (+)

Tabla 11: Umbrales óptimos para cada extremidad en cada grupo de datos, indicadores de error, concordancia de Lin y P-Value entre conteos automáticos y $ConteoH_{Consenso}$.
‡ El grupo de datos de Mediapipe, en la extremidad izquierda obtuvo 3 combinaciones con de UT con el mismo desempeño óptimo.
$\sim$ Las pruebas estadísticas U Mann-Whitney indican que al utilizar umbrales óptimos, las diferencias entre cada grupo de resultados de conteo automático y el grupo de $ConteoH_{Consenso}$ no son significativas, ya que todos los P-Value obtenidos son mayores al $\alpha = 0.05$.

La tabla 12a presenta los resultados del conteo automático utilizando la combinación de umbrales óptima para cada extremidad y para cada grupo de datos, tachando los resultados del video n° 3, debido a que no se consideraron para el análisis conjunto, ya que los datos de Dataset fueron de dicho video fueron descartados. Los resultados, nuevamente se han ordenado de mayor a menor en función del $ConteoH_{Consenso}$ de ATD de la extremidad derecha, columna Der_min de la tabla 12b. De manera adicional, se presenta la Tabla 12c, con la finalidad de exponer para cada extremidad, la diferencia entre $ConteoH_{Consenso}$ - Conteo $Mediapipe_{Suavizado}$ realizado con umbrales óptimos para la extremidad derecha. En esta última tabla, el orden de los videos se ha establecido de menor a mayor, en base al valor absoluto de la diferencia entre los conteos de la extremidad derecha.

	Dataset				Dataset Suavizado				Mediapipe				Mediapipe Suavizado			
n° video	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der
9	24.4	0.5	29.5	1.0	26.2	0.5	28.6	1.0	34.2	2.0	34.2	2.0	27.2	0.5	33.3	2.0
6	24.5	0.5	31.3	1.0	27.5	0.5	28.7	1.0	35.5	2.0	34.2	2.0	29.6	1.0	31.3	1.0
4	24.9	0.5	27.4	0.5	24.9	0.5	26.7	0.5	23.4	0.5	24.9	0.5	22.0	0.0	24.5	0.5
1	21.8	0.0	26.4	0.5	22.5	0.5	21.2	0.0	22.5	0.5	25.8	0.5	17.9	0.0	22.8	0.5
20	22.6	0.5	28.1	1.0	23.7	0.5	22.6	0.5	23.7	0.5	23.0	0.5	20.2	0.0	20.9	0.0
14	22.2	0.0	28.7	1.0	29.0	1.0	27.0	0.5	27.6	1.0	29.0	1.0	24.9	0.5	29.3	1.0
12	23.6	0.5	28.3	1.0	26.1	0.5	23.9	0.5	28.0	1.0	27.0	0.5	24.9	0.5	26.1	0.5
15	29.1	1.0	29.6	1.0	30.0	1.0	26.9	0.5	34.5	2.0	28.2	1.0	29.1	1.0	28.7	1.0
18	23.2	0.5	28.3	1.0	22.6	0.5	22.3	0.0	23.5	0.5	27.7	1.0	19.9	0.0	24.1	0.5
16	26.7	0.5	32.5	1.0	28.1	1.0	27.2	0.5	27.8	1.0	31.0	1.0	24.6	0.5	27.8	1.0
10	26.8	0.5	26.4	0.5	26.4	0.5	25.7	0.5	30.0	1.0	25.7	0.5	26.8	0.5	26.8	0.5
7	28.7	1.0	29.7	1.0	31.8	1.0	25.2	0.5	33.8	2.0	27.7	1.0	28.7	1.0	29.7	1.0
13	26.6	0.5	24.4	0.5	29.8	1.0	22.1	0.0	32.9	2.0	27.5	1.0	28.9	1.0	30.2	1.0
5	26.6	0.5	27.0	0.5	28.4	1.0	23.8	0.5	32.1	1.0	25.2	0.5	26.1	0.5	24.7	0.5
11	23.0	0.5	31.5	1.0	27.0	0.5	25.2	0.5	29.7	1.0	27.3	0.5	27.6	1.0	27.6	1.0
2	24.6	0.5	26.2	0.5	25.0	0.5	24.2	0.5	30.6	1.0	25.4	0.5	25.4	0.5	24.2	0.5
~~3~~	~~18.0~~	~~0.0~~	~~32.7~~	~~2.0~~	~~23.5~~	~~0.5~~	~~23.5~~	~~0.5~~	~~26.0~~	~~0.5~~	~~21.8~~	~~0.0~~	~~20.6~~	~~0.0~~	~~19.7~~	~~0.0~~
17	23.3	0.5	21.8	0.0	21.0	0.0	18.5	0.0	20.8	0.0	20.3	0.0	18.2	0.0	19.2	0.0
8	24.4	0.5	26.3	0.5	27.1	0.5	24.0	0.5	34.1	2.0	26.3	0.5	30.6	1.0	26.3	0.5
19	20.6	0.0	21.1	0.0	20.8	0.0	18.1	0.0	18.6	0.0	17.4	0.0	18.1	0.0	17.4	0.0

(a) Conteo automático con umbrales óptimos

	Conteo Humano de Consenso			
n° video	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der
9	21.0 ± 5.2	0.0	41.0 ± 10.4	3.0
6	19.0 ± 3.6	0.0	36.3 ± 7.6	2.0
4	17.0 ± 2.6	0.0	31.0 ± 6.9	1.0
1	16.0 ± 2.6	0.0	30.3 ± 6.7	1.0
20	14.7 ± 2.9	0.0	28.7 ± 6.4	1.0
14	15.7 ± 3.8	0.0	28.3 ± 6.7	1.0
12	14.0 ± 3.5	0.0	27.0 ± 7.8	0.5
15	34.3 ± 6.8	2.0	26.3 ± 8.4	0.5
18	14.0 ± 3.6	0.0	26.0 ± 6.1	0.5
16	12.7 ± 2.9	0.0	24.3 ± 5.8	0.5
10	27.7 ± 6.4	1.0	22.3 ± 5.8	0.0
7	43.0 ± 10.4	4.0	22.3 ± 4.9	0.0
13	38.7 ± 9.8	3.0	20.3 ± 4.9	0.0
5	39.7 ± 10.7	3.0	20.0 ± 4.4	0.0
11	18.7 ± 1.2	0.0	20.0 ± 4.4	0.0
2	34.3 ± 8.4	2.0	19.0 ± 4.4	0.0
3*	33.3 ± 7.5	2.0	18.3 ± 4.9	0.0
17	15.0 ± 4.6	0.0	17.7 ± 3.2	0.0
8	33.0 ± 6.9	2.0	17.0 ± 3.5	0.0
19	21.3 ± 5.8	0.0	10.7 ± 2.9	0.0

(b) $ConteoH_{Consenso}$

	Mediapipe Suavizado			
n° video	Izq_min	F.Frec.Izq	Der_min	F.Frec.Der
12	-10.9	-0.5	0.9	0.0
14	-9.2	-0.5	-1.0	0.0
3	12.7	2.0	-1.4	0.0
17	-3.2	0.0	-1.5	0.0
18	-5.9	0.0	1.9	0.0
15	5.2	1.0	-2.4	-0.5
16	-11.9	-0.5	-3.5	-0.5
10	0.9	0.5	-4.5	-0.5
5	13.6	2.5	-4.7	-0.5
6	-10.6	-1.0	5.0	1.0
2	8.9	1.5	-5.2	-0.5
4	-5.0	0.0	6.5	0.5
19	3.2	0.0	-6.7	0.0
7	14.3	3.0	-7.4	-1.0
1	-1.9	0.0	7.5	0.5
11	-8.9	-1.0	-7.6	-1.0
9	-6.2	-0.5	7.7	1.0
20	-5.5	0.0	7.8	1.0
8	2.4	1.0	-9.3	-0.5
13	9.8	2.0	-9.9	-1.0

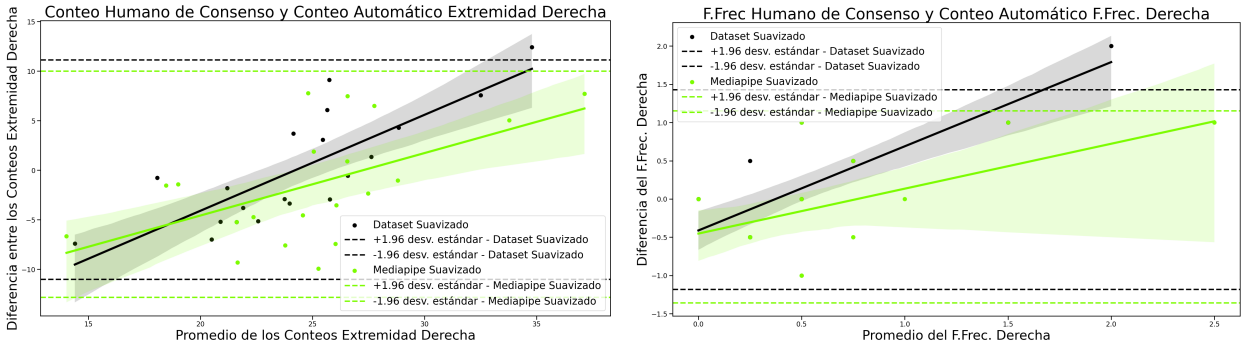
(c) Diferencia: Conteo Humano de Consenso - Conteo Automático Mediapipe Suavizado.

Tabla 12: Resultados de conteo automático y $ConteoH_{Consenso}$ utilizando las combinaciones óptimas de cada grupos de datos.

En la Tabla 12a se observa, para cada extremidad, el detalle de conteo automático de ATD y respectiva asignación del FF utilizando los umbrales que se han definido como óptimos. Se han tachado con color rojo los datos referentes al video n°3, ya que no se utilizaron en el análisis de conteo. En la Tabla 12b se observa el $ConteoH_{Consenso}$ de los 3 evaluadores participantes como promedio y desviación estándar de ATD, junto con su respectiva asignación del FF, en los 20 videos, ordenados de mayor a menor en base a la extremidad derecha. También se destacan los datos del video n°3 que no se utilizó para comparar con el conteo automático.

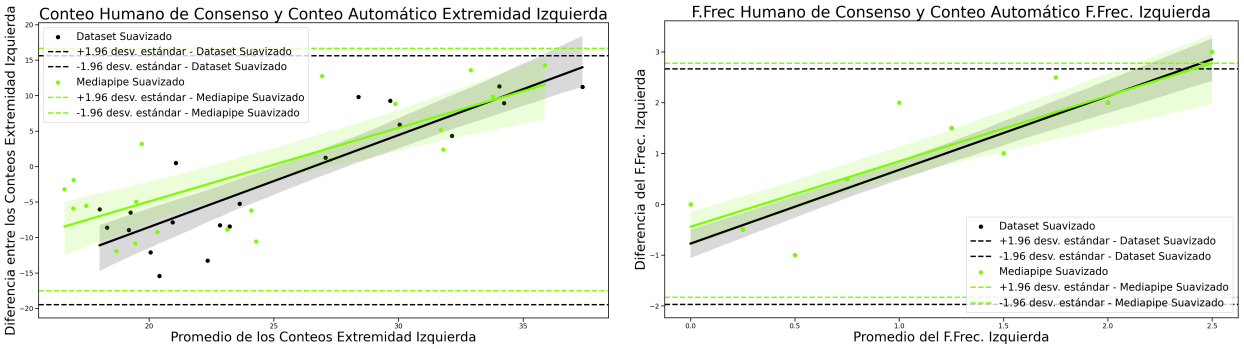
En la Tabla 12c, se observa, para cada extremidad, la diferencia entre $ConteoH_{Consenso}$ - $Mediapipe_{Suavizado}$ con umbrales óptimos para la extremidad derecha de Mediapipe Suavizado $UT = 0.5\ s$ y $UA = 15°$. Se destaca con un color verde transparente la columna Der_min, cuyo valor absoluto se utiliza como referencia para ordenar los resultados de manera creciente.

El análisis de discrepancia entre los resultados humanos de consenso y los resultados del contador automático configurado con umbrales óptimos, se llevó a cabo a través de los gráficos de bland-altman para ambas extremidades, tanto para los conteos como para la asignación del FF en los datos de Mediapipe Suavizado y Dataset Suavizado. Los cuales permiten observar cómo varía la diferencia entre los métodos de conteo en función del promedio de las mediciones con cada método. En general, se observó una línea de tendencia con una pendiente positiva para ambos grupos de datos en cada uno de los gráfico, donde la diferencia entre los métodos es negativa en una porción de la curva y es positiva a partir de cierto valor promedio de las mediciones. Lo señalado, es válido para los resultados de conteo automático y para la asignación del FF, en ambas extremidades y en ambos grupos de datos. Esto sugiere que el error entre los métodos de conteo no tiene un comportamiento constante dentro del rango de promedio evaluado. Lo descrito puede ser observado en las Figuras 22 y 23.



(a) (b)

Figura 22: Gráfico de Bland-Altman para los conteo 22a y asignación del FF 22b en Extremidad Derecha El análisis de Bland-Altman muestra una línea de tendencia con una pendiente positiva para ambos grupos de datos en cada uno de los gráfico, donde la diferencia entre los métodos es negativa en una porción de la curva, mientras que es positiva a partir de cierto valor promedio de las mediciones. Lo que sugiere que el error entre los métodos de conteo no tiene un comportamiento constante dentro del rango de promedio evaluado.



(a) (b)

Figura 23: Gráfico de Bland-Altman para los conteo 23a y asignación del FF 23b en Extremidad Izquierda El análisis de Bland-Altman muestra una línea de tendencia con una pendiente positiva para ambos grupos de datos en cada uno de los gráfico, donde la diferencia entre los métodos es negativa en una porción de la curva, mientras que es positiva a partir de cierto valor promedio de las mediciones. Lo que sugiere que el error entre los métodos de conteo no tiene un comportamiento constante dentro del rango de promedio evaluado.

Para finalizar, se recomienda revisar la sección 11.8.4 Anexo: Características de los videos, personas y estrategia de movimiento en relación al conteo de «acciones técnicas dinámicas», donde se describen en detalle las características generales de cada video, de las personas filmadas y caracterización de las estrategias del movimiento observado por cada persona.

8. Discusión

8.1. Optimización de umbrales y respuestas a las preguntas de investigación

En la sección introductoria, se planteó la necesidad de establecer estrategias para reducir la incidencia y prevalencia de los trastornos musculoesqueléticos relacionados al trabajo. Se hizo referencia a las brechas de conocimiento en cuanto al uso y utilidad de las herramientas de visión computacional en el estudio del movimiento humano para abordar esta problemática, y se plantearon tres preguntas que dirigieron esta investigación: ¿Qué diferencia hay entre un conteo automático de ATD mediante visión computacional comparado con un $ConteoH_{Consenso}$ proporcionado por especialistas en ergonomía, a través del analisis de videos de un Dataset? Además, ¿Qué diferencia hay si ese conteo automático también se compara con el conteo obtenido a partir de anotaciones de la postura disponibles en el Dataset? ¿Cómo se comporta la asignación del FF entre las distintas modalidades de conteo?

Se comparó el conteo automático de ATD con grupos de datos suavizados y sin suavizar con el $ConteoH_{Consenso}$ determinado por especialistas en los videos de tareas repetitivas de la base de datos «University of Washington Indoor Object Manipulation Dataset (Dataset)», utilizando Mediapipe Pose para estimar la postura de las personas incluidas en los videos del Dataset. Se adaptó el contador automático basado en umbral que propuso Taborri et. al [14] y se realizaron pruebas para analizar 50 combinaciones de umbral con los grupos de datos del Dataset y Mediapipe, con la finalidad de determinar la mejor combinación en términos del mínimo error RMSE (respecto al $ConteoH_{Consenso}$) de los conteos automáticos y el RMSE del FF que se asigna de manera respectiva.

La respuesta a la primera y segunda pregunta es que el contador automático de ATD puede configurarse con una combinación de umbrales «óptimos» que ofrecen un error $RMSE_{Conteo}$ inferior a 6 ATD por minuto considerando los 20 videos, lo que es inferior a la desviación estándar propia del $ConteoH_{Consenso}$ de este estudio. Además, al usar la prueba U de Mann-Whitney, no existe diferencia estadísticamente significativa entre los conteos automáticos realizados con umbrales óptimos y los $ConteoH_{Consenso}$. Sin embargo, una limitación del contador automático, es que no se halló una combinación de umbral óptima en todas las situaciones, es decir, cada grupo de datos y cada extremidad posee una combinación distinta de umbrales óptimos. La respuesta a la tercera pregunta es: si el conteo se configura con los umbrales que en este estudio se han definido como óptimos, la asignación del FF respectivo, arroja un error $RMSE_{frec}$ menor a 0.6 para la extremidad derecha y menor a 1.2 para la extremidad izquierda, que también es inferior a la desviación estándar propia del FF entregado por consenso en cada extremidad, y mediante la prueba U de Mann-Whitney es posible establecer que no existe diferencia estadísticamente significativa. En razón a lo planteado, la evidencia descrita entrega argumentos suficientes para aceptar la hipótesis planteada.

Por otro lado, es posible sostener que en este trabajo se obtuvieron resultados similares a los reportados por Taborri et al. en su artículo original, donde se reportó un error relativo del FF inferior al 6 %, respecto al «conteo humano de consenso», al configurar el contador automático con parámetros óptimos, en este trabajo se logró un $Error\ Relativo\ Frecuencia\ Norm$ también inferior al 6 %, respecto al máximo valor asignable para el FF. Sin embargo, como se ha señalado las definiciones entregadas en el artículo original y este trabajo, son levemente diferentes. Estos resultados son promisorios como propuesta para abordar el problema de investigación planteado en un inicio, el cual hace referencia a la baja reproducibilidad humana (alto grado de error) para asignar el FF, según lo reportado en la literatura [25], en torno al 30 % y 23 % intra e inter evaluador, respectivamente.

Por último, si bien el desempeño más óptimo, en términos de los indicadores de error utilizados, se halló con el grupo de datos Mediapipe (sin suavizar), este estudio sugiere que es preferible utilizar el contador automático con los datos suavizados ya que el grado de error observado en los conteos de ATD y en la respectiva asignación del FF es menos dependiente de la elección de umbrales que se haga. Dicho de otro modo, si se escogen equivocadamente los umbrales y se está trabajando con datos suavizados, en los videos utilizados en este estudio, el error observado es menor que si se escogen equivocadamente en datos sin suavizar.

8.2. Desafíos de los métodos de captura con Kinect y estimación Mediapipe

Como se ha señalado, previo al análisis estadístico, el primer análisis se realizó mediante inspección visual de la reconstrucción del movimiento registrado en los distintos grupos de datos en el espacio 3D de Unity Engine. La posibilidad de visualización y navegación desde cualquier perspectiva en el espacio, junto con la posibilidad de transmitir las coordenadas de Mediapipe Pose en tiempo real desde Python hacia en C# de Unity Engine, fue fundamental en etapas de desarrollo del código, pre-procesamiento y cálculo de ángulos, para detectar y corregir anomalías en las reconstrucciones, dependientes de la programación, ya que no era fácil de interpretar, sin visualizar, el movimiento y la disposición de vectores que conforman los ejes y planos de movimiento en los sistemas de coordenadas para calcular los ángulos. Esta herramienta, permitió constatar visualmente el efecto del suavizado de las coordenadas articulares con la aplicación de una media móvil para reducir la agitación de la ubicación de los puntos anatómicos

detectados y, por último, detectar y corregir la magnitud del componente «Z» de la estimación postural de Mediapipe Pose, para modelar el movimiento de manera mucho más proporcional durante los giros, al agacharse, al acercarse o alejarse de la cámara, volviéndose una reconstrucción altamente similar a lo observado en los videos.

Una limitación de este estudio es que el escalamiento realizado al componente «Z» se hizo en base a la observación únicamente y no basado en criterios de proporcionalidad anatómica, lo cual puede explorarse en trabajos futuros para comparar con capturas posturales de alta exactitud. La alteración de la proporcionalidad que motivó esta corrección en Mediapipe, también ha sido reportada en una publicación reciente de Dill et al. [37], donde se utilizaron estimaciones posturales con Mediapipe Pose con 2 cámaras en simultáneo, con diferentes ángulos respecto al plano visual de la persona filmada, hallándose que la estimación postural es altamente dependiente de la perspectiva de la cámara, afectando las proporciones corporales. En su estudio, los autores concluyen sugiriendo la necesidad de establecer reconstrucciones con Mediapipe basadas en 2 cámaras o basadas en la aplicación de modelos biomecánicos de movimiento, para ajustar la estimación postural y asegurar que las proporciones corporales se mantengan constantes.

La reconstrucción del movimiento de Mediapipe Pose se observó muy similar al movimiento de las personas en los videos, sin diferencias demasiado evidentes a la inspección visual, lo cual sugiere una mayor robustez frente a la oclusión comparado con la reconstrucción a partir de las coordenadas del Dataset, realizada con Kinect V2. En dicho grupo de datos, se observaron grandes diferencias y errores, comparado con la postura y el movimiento de las personas filmadas en los videos, principalmente al momento de agacharse o elevar los brazos, y también, al momento de existir oclusión de segmentos por otra parte del cuerpo o por los elementos manipulados en los videos, reconstruyendo posturas incomprensibles o sin entregar una forma de esqueleto. Estos errores son similares a lo reportado por Manguishi et al. en [23], donde se relata que, en ocasiones, no les fue posible capturar con Kinect V2 el «esqueleto» en la postura de una persona arrodillada en el suelo. De manera similar, el estudio publicado por Kim et al. [35], reportó este tipo de problemas, donde se comparó el sensor de profundidad de la Kinect V2, capturado a 30 Hz, con el sistema óptico de 8 cámaras infrarroja de Vicon MX40+ de 53 marcadores reflectantes, capturado a 120 Hz y, con el sistema de Xsens, basado en 17 sensores inerciales, capturado a 240 Hz en posturas estáticas para la evaluación del riesgo biomecánico de TMERT. Se reportaron múltiples problemas de captura en la Kinect, los que podían ser pequeñas diferencias en los ángulos evaluados hasta diferencias extremadamente grandes. Como ejemplo, en una de sus pruebas la persona estaba agachada, ocultando la extremidad del sensor, y en consecuencia, Kinect V2 asumió que la rodilla estaba extendida por completo, sin flexión, generando grandes diferencias con los otros sistemas de captura de movimiento. Los autores aconsejan tener precauciones en el uso de la Kinect para evaluaciones ergonómicas en situaciones de oclusión de segmentos frente al sensor.

A partir de lo descrito, se puede señalar que Mediapipe logró reproducir de manera satisfactoria y, con notable estabilidad, el movimiento en 3D a partir de videos que suponen un desafío importante para la estimación postural, los que poseen características similares a los que se obtienen en el ámbito ocupacional: no están en alta definición (576 × 324 pixeles), las personas filmadas realizan frecuentes giros, dando la espalda o el frente a la cámara, se agachan y manipulan objetos que ocluyen la captura directa de los segmentos anatómicos por parte de la cámara. Presentó reconstrucciones más robustas a la oclusión que aquellas realizadas a partir de Kinect V2. En consecuencia, se logró cumplir el objetivo

específico n°1 de esta investigación, referido a obtener y comparar las variables cinemáticas obtenidas a partir de la anotación de las coordenadas articulares del Dataset con aquellas variables cinemáticas obtenidas por Mediapipe Pose, en los respectivos videos del Dataset.

Por último, lo señalado deriva en una de las principales limitaciones de este estudio: la base de datos que se utilizaría como referencia «$ground\ truth$» respecto de las coordenadas articulares y el movimiento humano no ofreció una exactitud suficientemente estable y confiable en toda la duración de los registros. Esto ocurrió debido a que no se pudo hallar una base de datos cuya captura de movimiento sea de alta exactitud para servir como fuente de referencia confiable respecto del movimiento analizado y, que a la vez que incluya tareas repetitivas, con más de un ciclo de trabajo y, con el rostro de los participantes descubiertos para que sea compatible con Mediapipe. La creación de una propia base de datos con estos requisitos se estimó inviable en los plazos establecidos en el programa de Magister.

8.3. Necesidad de una base de datos de postura y video para el estudio del trabajo repetitivo

Para comparar las variables biomecánidas de los grupos de datos Dataset y Mediapipe, se desestimó el uso de indicadores comunes en la literatura, tales como *Error promedio de la posición articular (Mean Per Joint Position Error)* debido a la imposibilidad de establecer equivalencias entre los conjuntos de datos, ya que en Mediapipe las coordenadas son dadas en el dominio de la imágen, en pixeles, en cambio el Dataset presenta las coordenadas en metros. Tampoco se usó el Error promedio de ángulo articular (Mean Per Joint Angle Error), debido a que para llevar a cabo esta métrica de comparación, se requiere utilizar 17 articulaciones, entre las cuales no se pretendían incluir en este estudio, por ejemplo, extremidades inferiores y tronco. En su lugar, la comparación se realizó mediante el análisis de la concordancia de Lin de los ángulos del movimiento entre los grupos de datos.

Se observó que la abducción y flexión de hombros presentan en su mayoría concordancia alta (+) o moderada (+), seguido del movimiento de flexión de codo en la categoría moderada (+) o débil (+), y con más bajo grado de concordancia se encuentran los movimientos de rotación de hombros en las categorías débil (+) o muy débil (+) y flexión de muñeca, con una concordancia muy débil (+) en extremidad derecha o concordancia muy débil (-) en extremidad izquierda.

Existen varias causas que explican en mayor o menor medida la diferencia de resultados entre hombros, codos y muñecas. Se debe tener en consideración que el grado de concordancia entre los ángulos obtenidos depende de la exactitud de Kinect V2 y de Mediapipe, para reconstruir fielmente el movimiento de los puntos anatómicos de las personas, para definir correctamente los ángulos, y que sean «concordantes».

En primer lugar, la baja concordancia en muñecas incluso con valores negativos, puede atribuirse, en parte, a que Dataset y Mediapipe presentan diferentes puntos anatómicos en sus sistemas de datos a nivel de muñeca y manos, entonces se debió adaptar los datos de Mediapipe para efectuar un cálculo aproximado de la flexión de muñeca. En consecuencia, ambos grupos de datos evaluaron el ángulo de segmentos corporales cercanos, pero distintos.

En segundo lugar, la presencia de oclusiones afectó de manera muy considerable la exactitud de la captura en Dataset, no siendo tan evidente este efecto sobre la estimación postural de Mediapipe. Al respecto, el cálculo de ángulos de codos y muñecas fue más afectado, debido a que la tarea filmada

implicaba manipular cajas y realizar giros que terminan ocultando, de manera más frecuente, la captura directa de los codos y muñecas por parte del sensor en comparación a los hombros que están visibles la mayor parte del tiempo. De manera similar a este hallazgo, publicaciones tales como Dill et al. [37] y Moreno et al. [38], han reportado problemas de exactitud en la estimación postural de Mediapipe frente a oclusiones respecto de la cámara, que si bien fue menos evidente, contribuye en explicar diferencias en la concordancia y agregar incertidumbre respecto a las posibles causas.

En tercer lugar, frente a la captura o estimación postural levemente inexactas, los ángulos que se miden en articulaciones pequeñas, cuyos segmentos que conforman el ángulo a medir tiene un número pequeño de pixeles, como por ejemplo, en los dedos de las manos o en las muñecas, se ven más alterados en comparación al efecto de la misma inexactitud sobre articulaciones más grandes cuyos segmentos corporales tienen un gran número de pixeles, como los hombros. Esto es relevante, si además, se tiene en consideración que los videos descargados del Dataset presentan una baja resolución 576 × 324, por lo que se desconoce el impacto de dicha resolución en la exactitud de la estimación postural de Mediapipe Pose y en Dataset.

Es posible inferir que la exactitud ofrecida por Kinect V2 y Mediapipe sea aceptable a nivel de hombros, ya que los ángulos evaluados tienen en su mayoría concordancias altas y moderadas, resultado de una visibilidad mayor de los segmentos que dan lugar a la articulación y que los errores en exactitud afectan pocos pixeles en relación al número total de pixeles involucrados en los segmentos en estudio. A nivel de codos, existe incertidubre ya que no es posible establecer en qué grado el nivel de afectación de la concordancia es resultado de una baja exactitud de la captura, de la oclusión de los segmentos por la manipulación de cajas y los giros, o bien, acaso existe relación, o no, con la baja resolución de los videos disponibles. En el caso de muñecas, el grado de incertidumbre es mucho mayor, sobretodo para la Kinect que se muestra mucho más dependiente de la captura directa por parte del sensor frente a oclusiones, al tomar cajas y girar el tronco, además de que se requiere un grado de exactitud mayor en la captura y estimación postural en relación al tamaño de los segmentos y la influencia de la resolución del video.

Un aspecto que llama la atención, y que no se tiene una explicación, es el por qué la rotación de hombros obtuvo un grado de concordancia bajo comparado a la abducción y flexión de hombros, resultando incluso inferior a la flexión de codos. Una hipótesis podría estar relacionada con una determinación inexacta de la escalamiendo del componente de las coordenadas Z, que alteraría la profundidad y proyección del brazo sobre el plano transveso, donde se mide el ángulo, o bien, podría llegar a tener una explicación incluso a nivel de programación, lo cual debe ser investigado en mayor profundidad.

8.4. Potencial del contador automático como herramienta preventiva

En base a los resultados obtenidos, se considera que la adaptación del contador automático para visión computacional se implementó de manera exitosa, sin embargo, existen limitaciones en este estudio que condicionan la generalización de los resultados en su aplicabilidad real en el contexto ocupacional, pero que a la vez, sugieren interrogantes para nuevas investigaciones. Por ejemplo, los escenarios de prueba de este estudio tienen un número relativamente bajo de ATD por minuto, lo que se traduce en una baja exposición ocupacional a repetitividad: como máximo, se registraron 41 ATD por minuto en el $ConteoH_{Consenso}$, lo cual asigna un puntaje de 3 en el FF, donde el máximo puntaje es 10 cuando la tarea presenta 72.5 o más ATD por minuto, por lo tanto se desconoce el comportamiento de la propuesta en

situaciones de repetitividad más alta y de mayor relevancia epidemiológica referente al riesgo de TMERT. Una segunda limitación se halla en el bajo número de evaluadores participantes (n=3) para establecer el $ConteoH_{Consenso}$, lo que hace cuestionar la representatividad de estos resultados, pues, un participante otorgó conteos superiores a los otros participantes, en todos los videos analizados, y además, se observó una tendencia a aumentar la dispersión de los conteos humanos, reflejado en la desviación estándar de éstos, en los videos donde se contabilizó un número mayor de ATD por minuto, sugiriendo la existencia de los mismos problemas de reproducibilidad de la evaluación humana, reportados en la literatura. Lo señalado, son fuentes de incertidumbre que dificultan la interpretación de los resultados observados en el análisis de bland-altman, donde se observó que el error entre los métodos de conteo no tiene un comportamiento constante dentro del rango de promedio evaluado, pudiendo ser atribuible tanto al bajo número de evaluadores humanos, como a un desempeño variable del contador automático, haciendo necesario el desarrollo de nuevas investigaciones.

Una tercera limitación, guarda relación con cierto grado de incertidumbre que suscita la exactitud de estimación postural de Mediapipe Pose, versión 0.9.0, a nivel de muñecas y manos, sobretodo porque considera sólo 4 puntos anatómicos en los segmentos muñeca-mano, y existen oscilaciones en las coordenadas de los puntos anatómicos estimados, lo que puede ser insuficiente para tareas donde los movimientos son más detallados en muñeca, manos y dedos. Al plantear esta adaptación del contador automático para visión computacional, se hizo bajo el supuesto de que Mediapipe Pose entrega una estimación postural con grado de exactitud suficiente para evitar que oscilaciones azarosas en la estimación sean erróneamente detectadas, o no detectadas, por la combinación de umbrales temporal y de amplitud utilizados. Al plantear esto, no se tuvo en consideración que los ángulos evaluados en articulaciones pequeñas, como las muñecas, manos y dedos, pueden verse más afectados por dichas oscilaciones azarosas, por lo que la estimación postural en dichos segmentos requiere de un grado de exactitud superior al de articulaciones de mayor tamaño, y además, que dicha exactitud también tiene un grado de dependencia con la resolución del video en cuestión. Como se ha expuesto, la exactitud de la captura postural de Dataset no fue suficiente para considerarla una referencia válida con la cual comparar Mediapipe en los datos angulares de codos y muñecas, por lo tanto, la propuesta de contador automático posee grados de incertidumbre en cuanto a su funcionamiento. En este punto, cabe señalar que el 1 de Marzo de 2023, se publicó una nueva versión de Mediapipe Holistic[16], que combina la estimación 3D de Mediapipe Pose con la estimación 3D de Mediapipe Hand, que proporciona 21 puntos anatómicos por cada mano, y la estimación 3D de Mediapipe Face, con 468 puntos anatómicos, lo que se convierte en una alternativa valiosa para evaluar esta propuesta con una estimación postural de mayor exactitud y detalle postural a nivel de muñecas, manos y dedos.

En razón a lo expuesto, se requiere llevar a cabo un análisis donde se compare la estimación postural de Mediapipe Holistics con sistemas de captura de movimiento de alta exactitud, resilientes a la oclusión, tales como sensores inerciales o sensores ópticos multicámara, en un diseño experimental similar. Es esencial que dicho estudio considere la presencia de tareas similares a las observadas en el Dataset utilizado, agregando otros escenarios de manipulación con alternancia bimanual y utilizando herramientas, con un número de ATD superior a los 72.5 por minuto, ya que supone un acercamiento a las condiciones reales de los videos capturados en el contexto ocupacional.

[16]Al momento proponer y durante la mayor parte del desarrollo de este proyecto de investigación, Mediapipe Holistic no entregaba estimación postural 3D, sino que sólo entregaba estimación postural 2D. Detalles actualizados pueden ser revisados en https://github.com/google/mediapipe/blob/master/docs/solutions/holistic.md.

9. Conclusión

Este estudio propone y valida, para el conjunto de datos «University of Washington Indoor Object Manipulation», una adaptación del contador automático de «acciones técnicas dinámicas» basado en umbrales, propuesto por Taborri et al., para ser implementado con la herramienta de visión computacional Mediapipe Pose, el cual se configuró para obtener conteos automáticos indiferenciables, en términos estadísticos, del conteo humano de consenso, sirviendo como evidencia para aceptar la hipótesis de investigación. Este estudio sugiere que el contador automático debe utilizarse con datos de estimación postural suavizados, ya que los resultados del conteo y el grado de error observado es menos dependiente de la elección de umbrales que se haga. Sin embargo, se observó que el umbral óptimo es distinto en cada extremidad y en cada grupo de datos evaluado, lo cual guarda relación con el número de ATD realizado por cada extremidad.

Debido a que las tareas disponibles en el Dataset fueron evaluadas con una baja cantidad de ATD por minuto, los escenarios de prueba son insuficientes en cuanto al rango de exposición ocupacional a repetitividad como factor de riesgo de TMERT, lo que limita la generalización de estos resultados y la intepretación de su aplicabilidad real como herramienta de prevención.

Se concluye que es aconsejable utilizar herramientas de visualización y de modelamiento 3D, tales como Unity Engine, para observar desde cualquier perspectiva, variando la velocidad de la reconstrucción del movimiento a partir de Mediapipe. Se debe estudiar en mayor profundidad las alteraciones de proporcionalidad de la reconstrucción dependientes del escalamiento del componente Z de las coordenadas de Mediapipe.

La comparación de variables cinemáticas de Mediapipe y la reconstrucción del movimiento realizada a partir de Kinect arrojan concordancias altas (+) y moderadas (+) en hombros, concodancias moderadas (+) y débiles (+) en codos, y muy débiles (+) y (-) en muñecas derecha e izquierda respectivamente, concluyéndose que Kinect no ofrece un grado de exactitud suficiente para ser considerada una referencia válida para comparar la estimación postural de Mediapipe, en consecuencia, se requiere comparar con capturas de movimiento de alta exactitud, tales como sensores inerciales o sistemas ópticos multicámara como «ground truth». Asímismo, se requiere estudiar en mayor profundidad los movimientos de rotación interna y externa de hombros, teniendo como referencia datos posturales de alta exactitud.

Trabajo Futuro Esta investigación es promisoria, debido a que constituye un paso hacia reducir los tiempos de evaluación, la incertidumbre y la discrepancia asociada a evaluaciones ergonómicas con métodos observacionales dedicados a cuantificar la exposición ocupacional a riesgo biomecánico de TMERT asociado al trabajo repetitivo, por ejemplo, en el reconocimiento de la relación causal con el trabajo o en materia de vigilacia ocupacional. Sin embargo, esta propuesta debe evaluarse en diferentes escenarios, donde se incluya con la mayor exposición ocupacional posible. Además, esta investigación es promisoria si se considera que la estimación postural 3D basada en visión computacional es un campo muy activo, donde el mejoramiento de la exactitud de las herramientas, hace que la versión de Mediapipe Pose 0.9.0 utilizada en este estudio, pueda considerarse incluso obsoleta. De hecho la actualización de Mediapipe Holistic, entrega una estimación postural 3D de 21 puntos anatómicos por cada mano / dedos, lo que otorga oportunidad para evaluar esta propuesta con bases de datos cuya captura de movimiento sea de alta exactitud, y evitar las incertidumbres respecto de la validez de los ángulos del movimiento en

codos, muñecas y manos. En investigaciones futuras, se debería aumentar el número de participantes especialistas en ergonomía en referencia al conteo de ATD y el $ConteoH_{Consenso}$, en una variedad de escenarios más representativos de toda la gama de exposición ocupacional a repetitividad, oclusión y uso de elementos de protección personal, con el fin de enriquecer el análisis respecto a las propiedades de generalización de los resultados.

10. Bibliografia

[1] Bruce P Bernard y Vern Putz-Anderson. Musculoskeletal disorders and workplace factors; a critical review of epidemiologic evidence for work-related musculoskeletal disorders of the neck, upper extremity, and low back. *National Institute for Occupational Safety and Health*, (97B141), 1997. URL: https://www.cdc.gov/niosh/docs/97-141/pdfs/97-141.pdf?id=10.26616/NIOSHPUB97141.

[2] Laura Punnett y David H Wegman. Work-related musculoskeletal disorders: the epidemiologic evidence and the debate. *Journal of electromyography and kinesiology*, 14(1):13-23, 2004.

[3] Ministerio de Salud, Chile. Protocolos de vigilancia para trabajadores expuestos a factores de riesgo de trastornos musculoesqueléticos de extremidades superiores relacionados con el trabajo, 2012. URL: https://www.minsal.cl/portal/url/item/dbd6275dd3c8a29de040010164011886.pdf (visitado 21-05-2021).

[4] Ministerio del Trabajo y Previsión Social, Chile. Ley 16.744 establece normas sobre accidentes del trabajo y enfermedades profesionales, febrero de 1968. URL: https://www.bcn.cl/leychile/navegar?idNorma=28650 (visitado 21-05-2021).

[5] Superintendencia de Seguridad Social, Ministerio del Trabajo y Prevision Social, Chile. Compendio de normas del seguro social de accidentes del trabajo y enfermedades profesionales libro iii. denuncia, calificación y evaluación de incapacidades permanentes título iii. calificacion de enfermedades profesionales b. protocolo de patologías músculo esqueléticas de extremidad superior (mees). URL: https://www.suseso.cl/613/w3-propertyvalue-136546.html (visitado 13-01-2021).

[6] Superintendencia de Seguridad Social, Ministerio del Trabajo y Prevision Social, Chile. Informe Anual de Seguridad y Salud en el Trabajo 2022, Estadísticas de Accidentabilidad. Informe técnico, abril de 2023. URL: https://www.suseso.cl/607/articles-707000_archivo_01.pdf (visitado 01-09-2023).

[7] European Agency for Safety and Health at Work. and IKEI. and Panteia. *Work-related musculoskeletal disorders: prevalence, costs and demographics in the EU.* Publications Office, 2019. DOI: 10.2802/66947.

[8] Bureau of Labor Statistics, U.S. Department of Labor. Fact sheet | occupational injuries and illnesses resulting in musculoskeletal disorders (msds), mayo de 2020. URL: https://www.bls.gov/iif/oshwc/case/msds.htm (visitado 21-05-2021).

[9] Shengli Niu. Ergonomics and occupational safety and health: an ilo perspective. *Applied ergonomics*, 41(6):744-753, 2010.

[10] Instituto de Salud Pública de Chile. Trabajo Repetitivo de Miembros Superiores. Orientaciones para su Evaluación en Entornos Laborales. 2021. URL: https://www.ispch.cl/salud-de-los-trabajadores/ (visitado 21-05-2021).

[11] Daniela Colombini, Enrico Occhipinti y Enrique Álvarez-Casado. The revised ocra checklist method. *Editorial Factors Humans: Barcelona, Spain*, 2013.

[12] Barbara A Silverstein, Lawrence J Fine y Thomas J Armstrong. Hand wrist cumulative trauma disorders in industry. *Occupational and Environmental Medicine*, 43(11):779-784, 1986.

[13] ISO. ISO 11228-3. Ergonomics - Manual Handling - Part 3: Handling of Low Loads at High Frequency. Informe técnico, 2007.

[14] Juri Taborri, Marco Bordignon, Francesco Marcolin, Alessandro Bertoz, Marco Donati y Stefano Rossi. On the OCRA measurement: automatic computation of the dynamic technical action frequency factor. *Sensors*, 20(6):1643, 2020. DOI: 10.3390/s20061643.

[15] Steffi L. Colyer, Murray Evans, Darren P. Cosker y Aki I. T. Salo. A review of the evolution of vision-based motion analysis and the integration of advanced computer vision methods towards developing a markerless system. *Sports Medicine - Open*, 4(1), 2018. DOI: https://doi.org/10.1186/s40798-018-0139-y.

[16] Yann Desmarais, Denis Mottet, Pierre Slangen y Philippe Montesinos. A review of 3d human pose estimation algorithms for markerless motion capture, 2020. arXiv: 2010.06449 [cs.CV].

[17] Li Li, Tara Martin y Xu Xu. A novel vision-based real-time method for evaluating postural risk factors associated with musculoskeletal disorders. *Applied Ergonomics*, 87:103138, 2020.

[18] Prabesh Paudel y Kyoung-Ho Choi. A deep-learning based worker's pose estimation. En *International Workshop on Frontiers of Computer Vision*, páginas 122-135. Springer, 2020.

[19] Ze Li, Ruiqiu Zhang, Ching-Hung Lee y Yu-Chi Lee. An evaluation of posture recognition based on intelligent rapid entire body assessment system for determining musculoskeletal disorders. *Sensors*, 20(16):4414, 2020. DOI: doi:10.3390/s20164414.

[20] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei y Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1812.08008*, 2018.

[21] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang y Matthias Grundmann. Blazepose: on-device real-time body pose tracking, 2020. arXiv: 2006.10204 [cs.CV].

[22] Cynthia C Norkin y D Joyce White. *Measurement of joint motion: a guide to goniometry*. FA Davis, 2016.

[23] Vito Modesto Manghisi, Antonio Emmanuele Uva, Michele Fiorentino, Vitoantonio Bevilacqua, Gianpaolo Francesco Trotta y Giuseppe Monno. Real time rula assessment using kinect v2 sensor. *Applied ergonomics*, 65:481-491, 2017.

[24] Esa-Pekka Takala, Irmeli Pehkonen, Mikael Forsman, Gert-Ake Hansson, Svend Erik Mathiassen, W Patrick Neumann, Gisela Sjogaard, Kaj Bo Veiersted, Rolf H Westgaard y Jorgen Winkel. Systematic evaluation of observational methods assessing biomechanical exposures at work. *Scandinavian Journal of Work, Environment Health*, 36(1):3-24, 2010. ISSN: 03553140, 1795990X. URL: http://www.jstor.org/stable/40967825.

[25] Ida-Märta Rhén y Mikael Forsman. Inter-and intra-rater reliability of the ocra checklist method in video-recorded manual work tasks. *Applied Ergonomics*, 84:103025, 2020.

[26] Brian D Lowe, Patrick G Dempsey y Evan M Jones. Ergonomics assessment methods used by ergonomics professionals. *Applied ergonomics*, 81:102882, 2019.

[27] Nuno Ferrete Ribeiro y Cristina P. Santos. Inertial measurement units: a brief state of the art on gait analysis. *2017 IEEE 5th Portuguese Meeting on Bioengineering (ENBENG)*:1-4, 2017. URL: https://repositorium.sdum.uminho.pt/bitstream/1822/71728/1/ribeiro2017.pdf.

[28] Behnoosh Parsa. University of washington indoor object manipulation (uw iom) dataset, 2020.

[29] Li Li, Ziyang Xie y Xu Xu. MOPED25: a multimodal dataset of full-body pose and motion in occupational tasks. *Journal of Biomechanics*, 113:110086, 2020. DOI: https://doi.org/10.1016/j.jbiomech.2020.110086.

[30] Moritz Tenorth, Jan Bandouch y Michael Beetz. The tum kitchen data set of everyday manipulation activities for motion tracking and action recognition. En *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, páginas 1089-1096, 2009. DOI: 10.1109/ICCVW.2009.5457583.

[31] Franziska Krebs, Andre Meixner, Isabel Patzer y Tamim Asfour. The kit bimanual manipulation dataset. En *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, páginas 499-506, 2021.

[32] Pauline Maurice, Adrien Malaisé, Serena Ivaldi, Olivier Rochel, Clelie Amiot, Nicolas Paris, Guy-Junior Richard y Lars Fritzsche. Andydata-lab-oneperson. en, 2019. DOI: 10.5281/ZENODO.3254403. URL: https://zenodo.org/record/3254403.

[33] Konstantinos Peppas, Konstantinos Tsiolis, Ioannis Mariolis, Angeliki Topalidou-Kyniazopoulou y Dimitrios Tzovaras. 3d-hpe_certh_dataset, versión 1.0.0, enero de 2021. DOI: 10.5281/zenodo.4475685. URL: https://doi.org/10.5281/zenodo.4475685.

[34] Ge Wu, Frans CT Van der Helm, HEJ DirkJan Veeger, Mohsen Makhsous, Peter Van Roy, Carolyn Anglin, Jochem Nagels, Andrew R Karduna, Kevin McQuade, Xuguang Wang y col. Isb recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion—part ii: shoulder, elbow, wrist and hand. *Journal of biomechanics*, 38(5):981-992, 2005.

[35] Woojoo Kim, Chunxi Huang, Donghyeok Yun, Daniel Saakes y Shuping Xiong. Comparison of joint angle measurements from three types of motion capture systems for ergonomic postural assessment:3-11, 2020. DOI: https://doi.org/10.1007/978-3-030-51549-2_1.

[36] Daniel Pelliccia. The concordance correlation coefficient - nirpy research, 2021. URL: https://nirpyresearch.com/concordance-correlation-coefficient/.

[37] Sebastian Dill, Maurice Rohr, Gökhan Güney, Christoph Hoog Antink, Andreas Rösch, Luisa De Witte y Elias Schwartz. Accuracy evaluation of 3d pose estimation with mediapipe pose for physical exercises. En *Current Directions in Biomedical Engineering*, volumen 9 de número 1, páginas 563-566. De Gruyter, 2023.

[38] Gerardo Moreno, Cristian Pinzón y Manuel Ferre. Comparison of intrusive and non-intrusive motion capture technologies for upper body kinematic analysis, 2023.

[39] Hongyi Xu, Eduard Gabriel Bazavan, Andrei Zanfir, William T Freeman, Rahul Sukthankar y Cristian Sminchisescu. Ghum & ghuml: generative 3d human shape and articulated pose models. En *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, páginas 6184-6193, 2020.

11. Anexos

11.1. Anexo: «Mediapipe Pose» - estimación postural basada en visión computacional

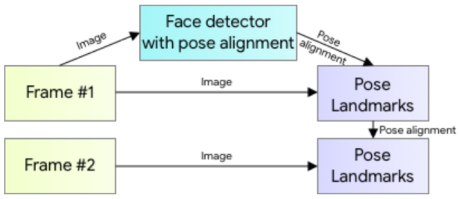
MediaPipe es un conjunto de soluciones de aprendizaje de máquinas multiplataforma (C++, Python, Android, iOS) listas para usar, gratis, de código abierto, extensibles y personalizables desarrolladas por la

compañía Google. Entre ellas, se encuentra *Mediapipe Pose* para la estimación postural de alta fidelidad, utilizando la investigación BlazePose [21] que también impulsa Google. El sistema toma, como entrada $(input)$, una imagen o cuadros de video RGB, y produce como salida $(output)$, una lista de las ubicaciones 3D `(pose_landmarks)` de los puntos anatómicos clave para una persona en la imagen, los cuales suelen indicar la posición de determinadas articulación, en el formato siguiente `[x, y, z, Visibility]`:

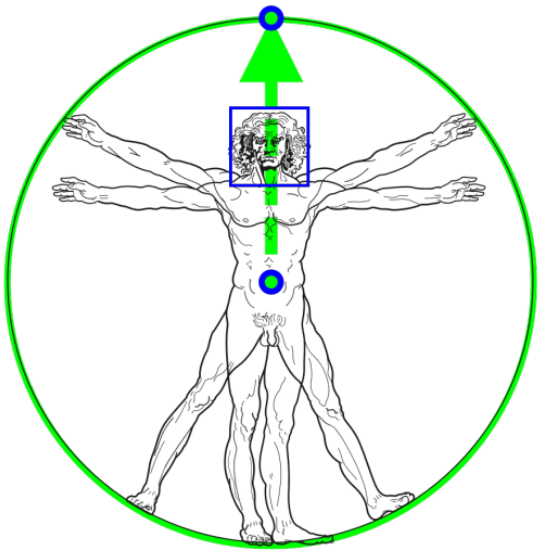
- `x` e `y`: Coordenadas normalizadas entre `[0.0, 1.0]` respecto al ancho y alto de la imagen $(w \times h)$ respectivamente.
- `z`: Representa la profundidad del punto anatómico clave, considerando el punto medio entre ambas caderas como el origen `[0.0, 0.0, 0.0]`. La coordenada `z` es un valor en «pixeles de la imagen» al igual que `x` e `y`[17]. Valores negativos informan de una posición entre la cámara y el punto medio entre las caderas; valores positivos informan de una posición detrás del punto medio entre las caderas.
- `Visibility`: Es un valor entre `[0.0, 1.0]` y denota la probabilidad de que el punto anatómico clave esté ubicado dentro del cuadro y no ocluído por otra parte del cuerpo u otro objeto.

El modelo que se observa en la Figura Mediapipe Pose. Extraído de [21]. rastrea la postura en dos pasos: usando un detector (Subfigure 24a y Subfigure 24b), el algoritmo ubica el rostro de una persona y su orientación dentro de la imagen, como región de interés (ROI) y, a partir de aquello, basado en las relaciones de proporción establecidas en el Vitruvio de Da Vinci estima los puntos de referencia anatómicos que dan origen a la estimación postural. Posteriormente, el rastreador predice los puntos de referencia de la pose y la máscara de segmentación dentro de la ROI utilizando el marco recortado de la ROI como entrada. En los casos de uso de video, el detector se invoca solo cuando es necesario, es decir, para el primer fotograma y cuando el rastreador ya no pudo identificar la presencia de la postura corporal en el fotograma anterior. Para otros cuadros del video, la canalización simplemente deriva la ROI de los puntos de referencia de pose del marco anterior.

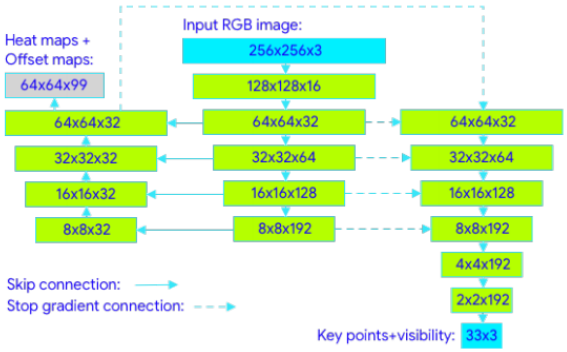
[17] La escala de la coordenada Z es similar a `x` e `y`, pero su naturaleza es diferente ya que no se obtiene en base a anotaciones humanas sino que mediante el ajuste de datos sintéticos provenientes de un modelo propuesto por Google en la investigación [39].



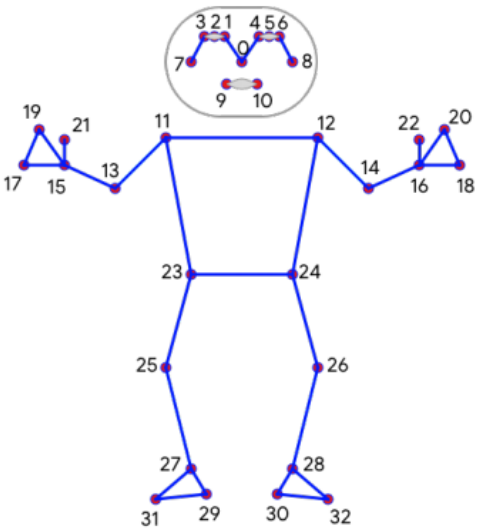
(a) Modelo de detección de Mediapipe Pose.



(b) Modelo de Detección y estimación de la ubicación y proporciones del cuerpo humano, basado en el Vitruvio de Da Vinci.



(c) Estructura de la red neuronal convolucional

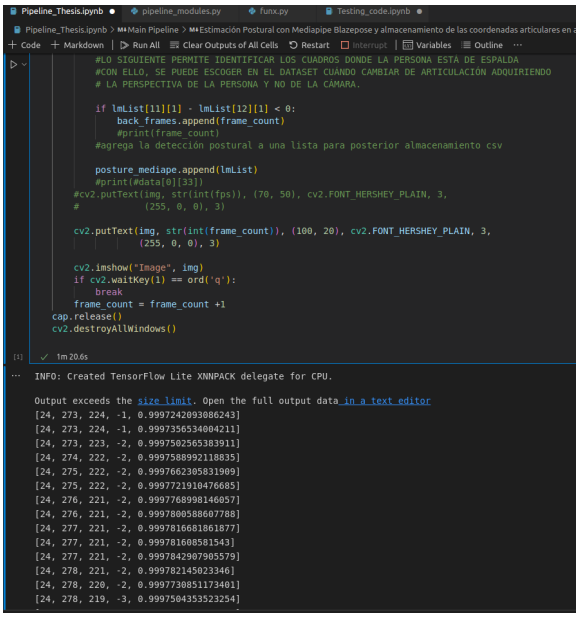


Topología Mediapipe, en Inglés:
0. Nose; 1. Left eye inner; 2. Left eye; 3. Left eye outer; 4. Right eye inner; 5. Right eye; 6. Right eye outer; 7. Left ear; 8. Right ear; 9. Mouth left; 10. Mouth right; 11. Left shoulder; 12. Right shoulder; 13. Left elbow; 14. Right elbow; 15. Left wrist; 16. Right wrist; 17. Left pinky #1 knuckle; 18. Right pinky #1 knuckle; 19. Left index #1 knuckle; 20. Right index #1 knuckle; 21. Left thumb #2 knuckle; 22. Right thumb #2 knuckle; 23. Left hip; 24. Right hip; 25. Left knee; 26. Right knee; 27. Left ankle; 28. Right ankle; 29. Left heel; 30. Right heel; 31. Left foot index; 32. Right foot index.

(d) Topología de puntos anatómicos clave «landmarks» estimados por Mediapipe.

Figura 24: Mediapipe Pose. Extraído de [21].

En cuanto a las especificaciones del modelo, las que pueden ser consultadas en el sitio oficial de Mediapipe:https://google.github.io/mediapipe/solutions/models.html, aunque se puede acceder directamente en <este hipervínculo>, es relevante informar que ha sido diseñado para aplicaciones de realidad aumentada, postura en 3D y reconocimiento de gestos, «Fitness[18]» y conteo de repeticiones, mediciones en 3D (ángulos / distancias); el modelo ha sido entrenado y evaluado en 30 mil imágenes de personas (bajo su consentimiento) usando una aplicación de realidad aumentada con cámara de celulares inteligentes, en ambientes naturales. La mayoría de imágenes de entrenamiento (85 mil) capturan un amplio rango de posturas de «Fitness»; de manera adicional, el modelo ha sido entrenado y evaluado en diferentes poblaciones equitativamente distibuidas y representativas de 14 sub-regiones geográficas: América central, Caribe, Sudamérica, Norteamérica, Asia Central, Asia Oriental, Sudeste Asiático, Asia Meridional, Asia Occidental, Africa del Norte África central Africa del Sur Australia y Nueva Zelanda, Europa (excluida la UE); se declara como fuera del alcance las apliaciones para trabajar con múltiples personas en una imagen, personas muy lejanas de la cámara (mayor a 4 metros), que la cabeza no sea visible, aplicaciones que requieran precisión en cuanto a las mediciones de profundidad y, cualquier forma de vigilancia o reconocimiento de identidad es explícitamente fuera del alcance, por lo que no es posible de realizar con esta tecnología. Adicionalmente, se declara que el modelo no ha sido diseñado para decisiones críticas para la vida humana, sino que para entretenimiento.



(a) Extracto de código del Pipeline de pruebas escrito en Python. Se imprime las coordenadas en 3D de la cadera derecha (landmark 24), con sus valores en el dominio de la imagen $(x, y, z, visibility)$

(b) Captura de pantalla: utilización de Mediapipe Pose para imprimir en tiempo real el cálculo de ángulos en 2D de las articulaciones de hombros, codos y muñecas de ambas extremidades (en rojo) y el desempeño en términos de FPS (en azul).

Figura 25: Aplicación de Mediapipe en videos del UW IOM Dataset.

[18]disciplina deportiva.

11.1.1. Anexo: (Código Python) Estimación postural 3D mediante Mediapipe Pose y almacenamiento en *.csv*

MediapipeToCSV.py

```python
import cv2
import time
import pipeline_modules as pm
import csv
import os
import pandas as pd
pathvideo='/home/diego/Documents/AT_Counting/Videos/' # Directorio donde se ubican los videos
for i in range(19):
    video = str(i)
    ext = '.avi'
    cap = cv2.VideoCapture(pathvideo+video+ext)
    pTime = 0
    detector = pm.poseDetector()
    posture_mediape = []     # para guardar las coordenadas
    back_frames = [] # Para guardar los cuadros donde la persona está de espalda al video
    frame_count = 0
    while cap.isOpened():
        success, img = cap.read()
        if not success:
            print("Ignoring empty camera frame.")
            break
        img = detector.findPose(img, False)
        lmList= detector.findPosition(img,False)
        cTime = time.time()
        rTime = cTime-pTime
        fps1 = 1 / rTime #frames per second
        pTime = cTime    # captura y muestra ángulos 3D en tiempo real de los segmentos en estudio
        shoulder_angle_right = detector.findAngle(img, 24,12,14, muneca_der=False, muneca_iz=False, distal=False, draw=True)
        shoulder_angle_left = detector.findAngle(img, 13,11,23, muneca_der=False, muneca_iz=False, distal=False, draw=True)
        elbow_angle_right = detector.findAngle(img, 12,14,16, muneca_der=False, muneca_iz=False, distal=True, draw=True)
        elbow_angle_left  = detector.findAngle(img, 11,13,15, muneca_der=False, muneca_iz=False, distal=True, draw=True)
        wrist_angle_right = detector.findAngle(img, 14,16,20, muneca_der=True, muneca_iz=False  ,  distal=True, draw=True)
        wrist_angle_left = detector.findAngle(img, 13,15,19, muneca_der=False, muneca_iz=True  , distal=True, draw=True)
        if len(lmList) !=0:
            """LO SIGUIENTE PERMITE IDENTIFICAR LOS CUADROS DONDE LA PERSONA ESTÁ DE ESPALDA
            CON ELLO, SE PUEDE ESCOGER EN EL DATASET CUÁNDO CAMBIAR DE ARTICULACIÓN ADQUIRIENDO
            LA PERSPECTIVA DE LA PERSONA Y NO DE LA CÁMARA."""
            if lmList[11][1] - lmList[12][1] < 0 and lmList[23][1] - lmList[24][1] < 0:
                back_frames.append(frame_count)
            posture_mediape.append(lmList)
        cv2.putText(img, str(int(fps1)), (100, 50), cv2.FONT_HERSHEY_PLAIN, 3, (255, 0, 0), 3)
        cv2.imshow("Image", img)
        if cv2.waitKey(1) == ord('q'):
            break
        if cv2.waitKey(1) == ord('p'):
            cv2.waitKey(-1)
        frame_count = frame_count +1
    cap.release()
    cv2.destroyAllWindows()
    path_save='Saved_files/'
    completeName = os.path.join()
    header = []
    for i in range(33):
            header.append(i)     # almacena la estimación postural en formato csv en path_save/thesis_txt_mediapipe
    with open(path_save+'thesis_txt_mediapipe/'+video+'.csv', 'w', newline='\n', encoding='UTF8') as f:
            writer = csv.writer(f)
            writer.writerow(header)
            writer.writerows(posture_mediape)
    my_df = pd.DataFrame(back_frames)     # almacena el archivo que indica los cuadros que dan la espalda a la cámara
    my_df.to_csv(path_save+'thesis_txt_mediapipe/'+video+'back_frames'+'.csv', index=False, header=False)
```

11.2. Anexo: Características del grupo de datos: «University of Washington Indoor Object Manipulation Dataset»

El conjunto de datos «University of Washington Indoor Object Manipulation» [33], es una base de datos creada para .... disponible en el sitio https://data.mendeley.com/datasets/xwzzkxtf9s/2, consiste en videos y la información de captura de la postura humana detectada en dichos videos (en 2D y 3D) de veinte participantes entre 18 a 25 años. Los videos fueron grabados con una cámara sensor de Kinect para Xbox One a una velocidad promedio de doce cuadros por segundo. El detalle de carpetas y archivos que trae el Dataset se observa en la Figura Carpetas y archivos incluidos en el Dataset [33]. y se describe a continuación:

- 1 carpeta «Videos», que contiene 20 archivos enumerados del 01 al 20, comprimidos usando la extensión «.7z,» que al extraer su contenido corresponde a los 20 videos de participantes en formato .avi con una resolución de 576 × 324 (8-bit no comprimidos).
    - Cantidad de hombres: 15
    - Cantidad de mujeres: 5
- 1 carpeta «VideoLabels», que contiene 20 archivos de texto (.txt) enumerados del 01 al 20, con etiquetas de actividad asignadas a cada cuadro del video en consideración.
- 1 carpeta «JointPositions», que contiene 20 archivos de extensión .mat, enumerados del 01 al 20, correspondiente a cada participante. Cada archivo .mat tiene las siguientes variables:
    - «videotimelogger»: marca de tiempo correspondiente a cada captura de cuadro RGB.
    - «bodytimelogger»: registro de tiempo correspondiente a cada captura de cuadro esquelético.
    - «bodylogger3D»: coordenadas x, y, z de 25 articulaciones en metros (con sensor kinect como origen) para cada cuadro capturado.
    - «pos2Dcolor»: coordenadas de píxeles de 25 articulaciones (en una imagen en color sin escala) correspondientes a cada cuadro en bodylogger3D.

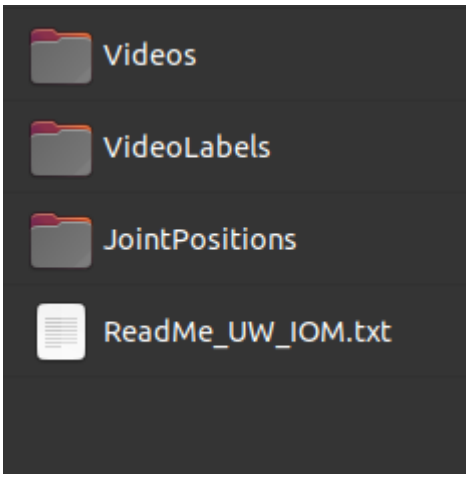


Figura 26: Carpetas y archivos incluidos en el Dataset [33].

11.2.1. Anexo: Descripción de la tarea realizada en el Dataset

Cada video dura aproximadamente tres minutos y cada participante realiza la misma tarea, tres veces: existe un estante donde se ha ubicado cajas y barras en tres niveles, a diferentes alturas. Partiendo desde el nivel mas alto, toman la caja y la colocan sobre una mesa frente al estante y, luego, realiza lo mismo con la barra, esto se repite en el nivel medio y con en el nivel inferior del estante. A continuación, la caja y la barra se vuelven a colocar en los respectivos niveles del estante, desde donde inicialmente se recogieron. Las cajas se manipulan con ambas manos y las varillas con una sola mano. Aparte de la anotación postural, la clasificación de las acciones tiene diecisiete etiquetas, donde cada etiqueta sigue una jerarquía de cuatro niveles: El primer nivel indica si se manipula la caja o la barra, el segundo nivel denota el movimiento humano (caminar, pararse y agacharse), el tercer nivel captura el tipo de manipulación de objetos, si corresponde (alcanzar, levantar, colocar y sostener), y el cuarto nivel representa la altura relativa de la superficie donde se realiza la manipulación (bajo, medio y alto).

(a) Fotograma del video del participante N°11, tomando la caja desde el nivel superior.

(b) Fotograma del video del participante N°2, tomando la barra desde el nivel central.

(c) Fotograma del video del participan N°19, tomando la caja desde el nivel superior.

(d) Fotograma del participante N°2, depositando la caja en el nivel inferior.

Figura 27: Ilustración de las tareas realizadas y características de los participantes.

11.3. Anexo: (Código $C\#$) Implementación en Unity Engine

Los archivos de código $C\#$: Anim_dataset.cs, AnimationCode.cs y Functions.cs (replicados en Python en la Sección 11.5 Anexo: (Código Python) FuncionesPipeline.py), se usaron para asignar las coordenadas articulares en 3D del Dataset y de Mediapipe, para reconstruirlas en el espacio virtual de Unity Engine.

~/Unity/Thesis/Assets/Anim_dataset.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.Linq;
using System.Threading;
using System;
using System.IO;
using static Functions;
public class Anim_dataset : MonoBehaviour
{
    public GameObject[] Body_dataset;
    List<string> linesds;
    List<string> backframes;
    int start = AnimationCode.start;
    public static int PointsxCount;
    public static int counter = 0;
    int one = -1;
    int two = -2;
    public static String video = "9";
    public static int sleep = 10;
    String ext = ".csv";
    List<String> lshoulderangles = new List<String>();
    public SetFrameRate setFrameRate;
    private Dictionary<string, int> desfaseDict = new Dictionary<string, int>
    {{ "1", 4 }, {"2", 3 },  {"3", 3 },   {"4", 4 },  {"5", 2 },  {"6", 3 },
      {"7", 3 },  {"8", 3 },  {"9", 3 },   {"10", 3 }, {"11", 4 }, {"12", 3 },
      {"13", 2 }, {"14", 4 }, {"15", 3 },  {"16", 3 }, {"17", 3 }, {"18", 3 },
      {"19", 4 }, {"20", 0 },  {"80", 3 }    };
    // Start is called before the first frame update
    void Start()
    {
        linesds = System.IO.File.ReadLines("Assets/Thesis_Export_csv/"+ video + ext).ToList();
        //PointsxCount = (linesds[0].Split(',').Length);
        PointsxCount = linesds.Count;
        backframes = System.IO.File.ReadLines("Assets/Thesis_Export_csv/" + video + "back_frames" + ext).ToList();
        int start =  AnimationCode.start + desfaseDict[video];
        if (start > 0)
        {
            int finish = start;
            for (int i = 0; i < backframes.Count; i++)
            {
                backframes[i] = (int.Parse(backframes[i]) - start).ToString();
            }
        }
        setFrameRate.SetTargetFrameRate();
    }
    // Update is called once per frame
    void LateUpdate()
    {
        //BodyPlaneAngles(Body_dataset);
        counter += 1   ;
        one = counter-1;
        two = counter-2;
        AssignCoordinatesDataset(linesds, counter, one, two, Body_dataset);
        if (backframes.Contains(counter.ToString()))
        {
            AnglesCalculationback(Body_dataset);
        }
        else
        {
            AnglesCalculationfront(Body_dataset);
        }
        print(counter);
        if (counter == PointsxCount)
       {
            UnityEditor.EditorApplication.isPlaying = false;
        }
    }
}
```

~/Unity/Thesis/Assets/AnimationCode.cs

```
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using UnityEngine;
using System.Threading;
using System;
using System.IO;
using static Functions;
public class AnimationCode : MonoBehaviour
{
    //public UDPReceive udpReceive;
    public GameObject[] Body;
    List<string> lines;
    public Vector3 normalized;
    public int lineCount;
    private string[] points;
    private string[] points_2;
    private string[] points_1;
    void AssignCoordinates_(List<string> lines, int counter, int counter_1, int counter_2, GameObject[] Body)
    {
        if (counter < 0)
            counter = 0;
        if (counter_1 < 0)
            counter_1 = 0;
        if (counter_2 < 0)
            counter_2 = 0;

        string[] points = lines[counter].Split(',');
        string[] points_2 = lines[counter_2].Split(',');
        string[] points_1 = lines[counter_1].Split(',');
        // Iterate through the points and assign the coordinates
                for (int i = 0; i <=32; i++)
        {
            float x0 = float.Parse(points[0 + (i*3)])/100;
            float x1 = float.Parse(points_1[0 + (i*3)])/100;
            float x2 = float.Parse(points_2[0 + (i*3)])/100;

            float y0 = float.Parse(points[1 + (i*3)])/100 -1.66f;
            float y1 = float.Parse(points_1[1 + (i*3)])/100 -1.66f;
            float y2 = float.Parse(points_2[1 + (i*3)])/100 -1.66f;

            float z0 = float.Parse(points[2 + (i*3)])/300 ;// - (float.Parse(points[2 + (3*23)])/330 + float.Parse(points[2 + (3*24)])/330)/2 ;
            float z1 = float.Parse(points_1[2 + (i*3)])/300 ;//- (float.Parse(points_1[2 + (3*23)])/330 + float.Parse(points_1[2 + (3*24)])/330)/2 ;
            float z2 = float.Parse(points_2[2 + (i*3)])/300 ;//- (float.Parse(points_2[2 + (3*23)])/330 +float.Parse(points_2[2 + (3*24)])/330)/2;

            Body[i].transform.localPosition = new Vector3((x0+x1+x2)/3,(y0+y1+y2)/3,(z0+z1+z2)/3);
            //Body[i].transform.localPosition = new Vector3(x0, y0,z0);
        }
    }
    public static int start;
    private static int datasetCounter;
    private static int datasetlength;
    int sleep = Anim_dataset.sleep;
    int counter = Anim_dataset.counter;
    int counter_1 = -1;
    int counter_2 = -2;
    float sagital_right_angle;
    String video = Anim_dataset.video;
    String ext = ".txt";
    public SetFrameRate setFrameRate;
    // Start is called before the first frame update
    void Start()
    {
        lines = System.IO.File.ReadLines("Assets/Mediapipe_txt/" + video + ext).ToList();
        lineCount = lines.Count;
        //string data = udpReceive.data;
        datasetlength = Anim_dataset.PointsxCount;
        int start = lineCount - Anim_dataset.PointsxCount ;
```

```
        setFrameRate.SetTargetFrameRate();
    }
    // Update is called once per frame
    void LateUpdate()
    {
        int start = lineCount - Anim_dataset.PointsxCount ;
        if (start<0)
        {
            start = 0;
        }
        // Assign the coordinates to the Body[i].transform.localPosition using a method
        counter =  Anim_dataset.counter + start;
        counter_1 = counter -1;
        counter_2 = counter -2;
        // Update is called once per frame and average 3 samples per frame
        AssignCoordinates_(lines, counter,counter_1, counter_2, Body);
        BodyPlaneAngles(Body);
        //Thread.Sleep(sleep);
        if (lineCount == datasetlength)
        {
            //System.IO.File.WriteAllLinesAsync("Angles_Mediapipe"+video+ext, lshoulderangles);
            UnityEditor.EditorApplication.isPlaying = false;
        }
        counter %= lineCount;
    }
}

```

11.3.1. (Código C#) Functions.cs

~/Unity/Thesis/Assets/Functions.cs

```
using System.Collections;
using System.Collections.Generic;
using System;
using UnityEngine;
public static class Functions
{
    public static void DrawPlane(Vector3 position,  Vector3 normal,  Vector3 front)
    {
         Vector3 v3;
         if (normal.normalized != Vector3.forward)
             v3 = Vector3.Cross(normal,  front).normalized * normal.magnitude;
         else
             v3 = Vector3.Cross(normal,  front).normalized * normal.magnitude;
         var corner0 = position + v3;
         var corner2 = position - v3;
         var q = Quaternion.AngleAxis(90.0f,  normal);
         Debug.DrawLine(corner0+front,  corner2+front,  Color.blue);
         Debug.DrawLine(corner0-front,  corner2-front,  Color.red);
         Debug.DrawLine(corner0+front,  corner0-front,  Color.green);
         Debug.DrawLine(corner2-front,  corner2+front, Color.black);

         Debug.DrawRay(position,  normal,  Color.red);
    }
    public static void AssignCoordinatesDataset(List<string> lines,  int counter,  int counter_1,  int counter_2,
GameObject[] Body)
    {
        // Initialize the Body array
        //Body = new GameObject[33];
        if (counter< 0)
        {
            counter = 0;
        }
        if (counter_1 < 0)
        {
            counter_1 = 0;
        }
        if (counter_2 < 0)
        {
            counter_2 = 0;
        }
        string[] points = lines[counter].Split(',');
        string[] points_2 = lines[counter_2].Split(',');
        string[] points_1 = lines[counter_1].Split(',');
          // Iterate through the points and assign the coordinates
        //int pointIndex = 0;
        for (int i = 0; i <=24; i++)
        {
            float x0 = float.Parse(points[0 + (i*3)]);
            float x1 = float.Parse(points_1[0 + (i*3)]);
            float x2 = float.Parse(points_2[0 + (i*3)]);

            float y0 = float.Parse(points[1 + (i*3)]);
            float y1 = float.Parse(points_1[1 + (i*3)]);
            float y2 = float.Parse(points_2[1 + (i*3)]);

            float z0 = float.Parse(points[2 + (i*3)]) - (float.Parse(points[2 + (3*23)]) + float.Parse(points[2 +
(3*24)])) / 2 ;
            float z1 = float.Parse(points_1[2 + (i*3)]) - (float.Parse(points_1[2 + (3*23)]) +
float.Parse(points_1[2 + (3*24)])) / 2 ;
            float z2 = float.Parse(points_2[2 + (i*3)]) - (float.Parse(points_2[2 + (3*23)])
+float.Parse(points_2[2 + (3*24)])) / 2 ;
            Body[i].transform.localPosition = new Vector3((x0+x1+x2)/3, (y0+y1+y2)/3, (z0+z1+z2)/3);
        }
    }
        public static void BodyPlaneAngles(GameObject[] Body)
        {
        Vector3 middhips = (Body[23].transform.localPosition + Body[24].transform.localPosition)/2f;
        Vector3 middshoulders = (Body[11].transform.localPosition + Body[12].transform.localPosition)/2f;
        // local coordinate system vectors: Hips; middshoulder; leftshoulder; right shoulder; define planes for
Vector3
        Vector3 vectorx_middhip = (Body[23].transform.localPosition - middhips);
```

```
        Vector3 vectory_middhip = middshoulders - middhips;
        Vector3 vectorz_middhip = Vector3.Cross(vectory_middhip, vectorx_middhip);
        Debug.DrawLine(middhips,  middhips+vectorx_middhip);
        Debug.DrawLine(middhips,  middhips+vectory_middhip, Color.green);
        Debug.DrawLine(middhips,  middhips+vectorz_middhip*10f, Color.red);
        //midd_shoulders
        Vector3 vectorx_middshoulders = Body[11].transform.localPosition - middshoulders;
        Vector3 vectory_middshoulders = - (middhips - middshoulders);
        Vector3 vectorz_middshoulders = Vector3.Cross(vectory_middshoulders, vectorx_middshoulders);

        // left SIDE
        // leftshoulder
        Vector3 vector_left_arm = Body[13].transform.localPosition - Body[11].transform.localPosition;
        Vector3 projectedvector_sagital_lshoulder = Vector3.ProjectOnPlane(vector_left_arm, vectorx_middshoulders);
        float sagital_left_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_sagital_lshoulder, -
(vectory_middshoulders), vectorx_middshoulders));
        Vector3 projectedvector_frontal_lshoulder = Vector3.ProjectOnPlane(vector_left_arm, vectorz_middshoulders);
        float frontal_left_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_frontal_lshoulder, -
vectory_middshoulders, vectorz_middshoulders));
        Vector3 projectedvector_transverse_lshoulder = Vector3.ProjectOnPlane(vector_left_arm,
vectory_middshoulders);
        float transverse_left_angle = Vector3.SignedAngle(projectedvector_transverse_lshoulder,
vectorz_middshoulders, vectory_middshoulders);
        // leftelbow
        Vector3 vector_left_forearm = Body[15].transform.localPosition - Body[13].transform.localPosition;
        Vector3 vector_left_elbowarm = -(Body[11].transform.localPosition - Body[13].transform.localPosition);
        float left_elbow_flexion = Vector3.Angle(vector_left_forearm, vector_left_elbowarm);
        // Leftwrist  // first we have to create the middfingers coordinates
        Vector3 lmiddfingers = (Body[19].transform.localPosition + Body[17].transform.localPosition)/2f;
        Vector3 vector_left_hand = (lmiddfingers - Body[15].transform.localPosition);
        Vector3 vector_left_forearmwrist = -(Body[13].transform.localPosition - Body[15].transform.localPosition);
        float left_wrist_flexion = Vector3.Angle(vector_left_hand, vector_left_forearmwrist);

        // RIGHT SIDE
        // rightshoulder
        Vector3 vector_right_arm = Body[14].transform.localPosition - Body[12].transform.localPosition;
        Vector3 projectedvector_sagital_rshoulder = Vector3.ProjectOnPlane(vector_right_arm, vectorx_middshoulders)
;
        var sagital_right_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_sagital_rshoulder,  -
(vectory_middshoulders), vectorx_middshoulders));
        Vector3 projectedvector_frontal_rshoulder = Vector3.ProjectOnPlane(vector_right_arm, vectorz_middshoulders)
;
        float frontal_right_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_frontal_rshoulder,  -
vectory_middshoulders, vectorz_middshoulders));
        Vector3 projectedvector_transverse_rshoulder = Vector3.ProjectOnPlane(vector_right_arm,
vectory_middshoulders);
        float transverse_right_angle = Vector3.SignedAngle(projectedvector_transverse_rshoulder,
vectorz_middshoulders,  -vectory_middshoulders);
        // rightelbow
        Vector3 vector_right_forearm = Body[16].transform.localPosition - Body[14].transform.localPosition;
        Vector3 vector_right_elbowarm = -(Body[12].transform.localPosition - Body[14].transform.localPosition);
        float right_elbow_flexion = Vector3.Angle(vector_right_forearm, vector_right_elbowarm);
        // rightwrist  // first we have to create the middfingers coordinates
        Vector3 rmiddfingers = (Body[20].transform.localPosition + Body[18].transform.localPosition)/2f;
        Vector3 vector_right_hand = (rmiddfingers - Body[16].transform.localPosition);
        Vector3 vector_right_forearmwrist = -(Body[14].transform.localPosition - Body[16].transform.localPosition);
        float right_wrist_flexion = Vector3.Angle(vector_right_hand, vector_right_forearmwrist);
        // draw left
        Debug.DrawLine(middshoulders,  middshoulders+vectory_middshoulders);
        Debug.DrawLine(middshoulders,  middshoulders+vectorx_middshoulders, Color.green);
        Debug.DrawLine(middshoulders,  middshoulders+vectorz_middshoulders*10f, Color.red); //
        Debug.DrawLine(middshoulders,  middhips); //
        Debug.DrawLine(Body[11].transform.localPosition,  Body[11].transform.localPosition + (-
vectory_middshoulders)   );
        Debug.DrawLine(Body[11].transform.localPosition,  Body[11].transform.localPosition+vectory_middshoulders);
        Debug.DrawLine(Body[11].transform.localPosition,  Body[11].transform.localPosition+vectorx_middshoulders,
Color.green);
        Debug.DrawLine(Body[11].transform.localPosition,  Body[11]
.transform.localPosition+vectorz_middshoulders*10f, Color.red); //
        Debug.DrawLine(middshoulders,  middhips); //
        Debug.DrawLine(Body[12].transform.localPosition,  Body[12].transform.localPosition + (-
vectory_middshoulders)   );
        Debug.DrawLine(Body[12].transform.localPosition,  Body[12].transform.localPosition+ vectory_middshoulders);
        Debug.DrawLine(Body[12].transform.localPosition,  Body[12].transform.localPosition+ vectorx_middshoulders,
Color.green);
        Debug.DrawLine(Body[12].transform.localPosition,  Body[12].transform.localPosition+
vectorz_middshoulders*10f, Color.red); //
```

```
        // codo derecho
        Debug.DrawLine(Body[14].transform.localPosition, Body[14].transform.localPosition +vector_right_elbowarm*1.5f, Color.green);
        Debug.DrawLine(Body[14].transform.localPosition,  Body[14].transform.localPosition +vector_right_forearm*1.5f, Color.red ); //
        //muñeca derecha
        Debug.DrawLine(Body[16].transform.localPosition, Body[16].transform.localPosition +vector_right_forearmwrist*1.5f, Color.green);
        Debug.DrawLine(Body[16].transform.localPosition,  Body[16].transform.localPosition +vector_right_hand*2.8f, Color.red );
        Debug.Log("MEDIAPIPE left shoulder sagital_left_angle: " + sagital_left_angle + " left shoulder frontal_left_angle: " + frontal_left_angle + " left shoulder transverse_left_angle: " + transverse_left_angle + " left_elbow_flexion: " + left_elbow_flexion +" left_wrist_flexion: " + left_wrist_flexion);
        // HOMBRO CORONAL: DESDE DELANTE
        Debug.DrawLine(middshoulders,  middshoulders+ vectory_middshoulders);
        Debug.DrawLine(middshoulders,  middshoulders+ vectorx_middshoulders, Color.green);
        Debug.DrawLine(middshoulders,  middshoulders+ vectorz_middshoulders, Color.red); //
        Debug.DrawLine(middshoulders,  middhips); //return data;
        Debug.DrawLine(Body[12].transform.localPosition,  Body[12].transform.localPosition + (-vectory_middshoulders));
        var data = string.Format(sagital_left_angle.ToString() + ", " + frontal_left_angle.ToString() + ", " + transverse_left_angle.ToString() +", "+ left_elbow_flexion.ToString()+", "+ left_wrist_flexion.ToString()+", "+ sagital_right_angle.ToString()+", "+ frontal_right_angle.ToString()+", "+ transverse_right_angle.ToString()+", "+ right_elbow_flexion.ToString()+", "+ right_wrist_flexion.ToString());
    }
        public static void AnglesCalculationfront(GameObject[] Body_dataset)
        {
            // middhips middshoulder
        Vector3 middhips = (Body_dataset[12].transform.localPosition + Body_dataset[16].transform.localPosition) /2f;
        Vector3 middshoulders = (Body_dataset[4].transform.localPosition + Body_dataset[8].transform.localPosition) /2f;
        Vector3 vectorx_middhip = (Body_dataset[16].transform.localPosition - middhips);
        Vector3 vectory_middhip = middshoulders - middhips;
        Vector3 vectorz_middhip = Vector3.Cross(vectory_middhip, vectorx_middhip);
        Debug.DrawLine(middhips,  middhips+vectorx_middhip);
        Debug.DrawLine(middhips,  middhips+vectory_middhip, Color.green);
        Debug.DrawLine(middhips,  middhips+vectorz_middhip*10f, Color.red);

        //midd_shoulders
        Vector3 left_shoulder = Body_dataset[8].transform.localPosition;
        Vector3 right_shoulder = Body_dataset[4].transform.localPosition;
        Vector3 vectorx_middshoulders = left_shoulder - middshoulders;
        Vector3 vectory_middshoulders =  -(middhips - middshoulders);
        Vector3 vectorz_middshoulders = Vector3.Cross(-vectorx_middshoulders, vectory_middshoulders);

        // left SIDE
        // leftshoulder
        Vector3 vector_left_arm =  Body_dataset[9].transform.localPosition - Body_dataset[8].transform.localPosition;
        Vector3 projectedvector_sagital_lshoulder = Vector3.ProjectOnPlane(vector_left_arm, vectorx_middshoulders);
        float sagital_left_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_sagital_lshoulder, -(vectory_middshoulders), vectorx_middshoulders));
        Vector3 projectedvector_frontal_lshoulder = Vector3.ProjectOnPlane(vector_left_arm, vectorz_middshoulders);
        float frontal_left_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_frontal_lshoulder, -vectory_middshoulders, vectorz_middshoulders));
        Vector3 projectedvector_transverse_lshoulder = Vector3.ProjectOnPlane(vector_left_arm,  -vectory_middshoulders);
        float transverse_left_angle = Vector3.SignedAngle(projectedvector_transverse_lshoulder, vectorz_middshoulders, -vectory_middshoulders);
        // leftelbow
        Vector3 vector_left_forearm = Body_dataset[10].transform.localPosition - Body_dataset[9].transform.localPosition;
        Vector3 vector_left_elbowarm = -(Body_dataset[8].transform.localPosition - Body_dataset[9].transform.localPosition);
        float left_elbow_flexion = Vector3.Angle(vector_left_forearm, vector_left_elbowarm);
        // leftwrist  // first we have to create the middfingers coordinates
        Vector3 lmiddfingers =  Body_dataset[23].transform.localPosition;
        Vector3 vector_left_hand = (lmiddfingers - Body_dataset[10].transform.localPosition);
        Vector3 vector_left_forearmwrist = -(Body_dataset[9].transform.localPosition - Body_dataset[10].transform.localPosition);
        float left_wrist_flexion = Vector3.Angle(vector_left_hand, vector_left_forearmwrist);

        // rightshoulder
        Vector3 vector_right_arm = Body_dataset[5].transform.localPosition - Body_dataset[4].transform.localPosition;
        Vector3 projectedvector_sagital_rshoulder = Vector3.ProjectOnPlane(vector_right_arm, vectorx_middshoulders);
```

```
        float sagital_right_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_sagital_rshoulder, -vectory_middshoulders, vectorx_middshoulders));
        Vector3 projectedvector_frontal_rshoulder = Vector3.ProjectOnPlane(vector_right_arm, vectorz_middshoulders);
        float frontal_right_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_frontal_rshoulder, -vectory_middshoulders, vectorz_middshoulders));
        Vector3 projectedvector_transverse_rshoulder = Vector3.ProjectOnPlane(vector_right_arm,  -vectory_middshoulders);
        float transverse_right_angle = Vector3.SignedAngle(projectedvector_transverse_rshoulder, vectorz_middshoulders, -vectory_middshoulders);
        // rightlbow
        Vector3 vector_right_forearm = Body_dataset[6].transform.localPosition - Body_dataset[5].transform.localPosition;
        Vector3 vector_right_elbowarm = -(Body_dataset[4].transform.localPosition - Body_dataset[5].transform.localPosition);
        float right_elbow_flexion = Vector3.Angle(vector_right_forearm, vector_right_elbowarm);
        // rightwrist  // first we have to create the middfingers coordinates
        Vector3 rmiddfingers = Body_dataset[21].transform.localPosition ;
        Vector3 vector_right_hand = (rmiddfingers - Body_dataset[6].transform.localPosition);
        Vector3 vector_right_forearmwrist = -(Body_dataset[5].transform.localPosition - Body_dataset[6].transform.localPosition);
        float right_wrist_flexion = Vector3.Angle(vector_right_hand, vector_right_forearmwrist);
        // draw left
        Debug.DrawLine(middshoulders,  middshoulders+vectory_middshoulders);
        Debug.DrawLine(middshoulders,  middshoulders+vectorx_middshoulders, Color.green);
        Debug.DrawLine(middshoulders,  middshoulders+vectorz_middshoulders*10f, Color.red); //
        Debug.DrawLine(middshoulders,  middhips); //
        Debug.DrawLine(left_shoulder,  left_shoulder + (-vectory_middshoulders)  );
        Debug.DrawLine(left_shoulder,  left_shoulder +vectory_middshoulders);
        Debug.DrawLine(left_shoulder,  left_shoulder+vectorx_middshoulders, Color.green);
        Debug.DrawLine(left_shoulder,  left_shoulder+vectorz_middshoulders*10f, Color.red); //
        Debug.DrawLine(middshoulders,  middhips); //
        Debug.DrawLine(right_shoulder,  right_shoulder + (-vectory_middshoulders)    );
        Debug.DrawLine(right_shoulder,  right_shoulder+vectory_middshoulders);
        Debug.DrawLine(right_shoulder,  right_shoulder+vectorx_middshoulders, Color.green);
        Debug.DrawLine(right_shoulder,  right_shoulder+vectorz_middshoulders*10f, Color.red); //
        // codo
        Debug.DrawLine(Body_dataset[5].transform.localPosition ,  Body_dataset[5].transform.localPosition +vector_right_forearm *1.5f, Color.red);
        Debug.DrawLine(Body_dataset[5].transform.localPosition ,  Body_dataset[5].transform.localPosition +vector_right_elbowarm*1.5f, Color.green); //
        // muñeca
        Debug.DrawLine(Body_dataset[6].transform.localPosition,  Body_dataset[6].transform.localPosition+vector_right_forearmwrist*1.5f, Color.green);
        Debug.DrawLine(Body_dataset[6].transform.localPosition,  Body_dataset[6].transform.localPosition+vector_right_hand*1.5f, Color.red); //
        //see results
        Debug.Log("left shoulder sagital_left_angle: " + sagital_left_angle + " left shoulder frontal_left_angle: " + frontal_left_angle + " left shoulder transverse_left_angle: " + transverse_left_angle + " left_elbow_flexion: " + left_elbow_flexion +" left_wrist_flexion: " + left_wrist_flexion);
        // draw right
        Debug.Log("DATASETFRONT: right shoulder sagital_right_angle: " + sagital_right_angle + " right shoulder frontal_right_angle: " + frontal_right_angle + " right shoulder transverse_right_angle: " + transverse_right_angle + " right_elbow_flexion: " + right_elbow_flexion +" right_wrist_flexion: " + right_wrist_flexion);
        // HOMBRO CORONAL: DESDE DELANTE
        Debug.DrawLine(middshoulders,  middshoulders+vectory_middshoulders);
        Debug.DrawLine(middshoulders,  middshoulders+vectorx_middshoulders, Color.green);
        Debug.DrawLine(middshoulders,  middshoulders+vectorz_middshoulders*10f, Color.red); //
        Debug.DrawLine(middshoulders,  middhips); //
        Debug.DrawLine(right_shoulder,  right_shoulder + (-vectory_middshoulders));
        }

    public static void AnglesCalculationback(GameObject[] Body_dataset)
        {
        // middhips middshoulder
        Vector3 middhips = (Body_dataset[12].transform.localPosition + Body_dataset[16].transform.localPosition)/2f;
        Vector3 middshoulders = (Body_dataset[4].transform.localPosition + Body_dataset[8].transform.localPosition)/2f;
        Vector3 vectorx_middhip = (Body_dataset[12].transform.localPosition - middhips);
        Vector3 vectory_middhip = middshoulders - middhips;
        Vector3 vectorz_middhip = Vector3.Cross(vectory_middhip, vectorx_middhip);
        Debug.DrawLine(middhips,  middhips+vectorx_middhip);
        Debug.DrawLine(middhips,  middhips+vectory_middhip, Color.green);
        Debug.DrawLine(middhips,  middhips+vectorz_middhip*10f, Color.red);
        //midd_shoulders
        Vector3 left_shoulder = Body_dataset[4].transform.localPosition;
```

```
        Vector3 right_shoulder = Body_dataset[8].transform.localPosition;
        Vector3 vectorx_middshoulders = left_shoulder - middshoulders;
        Vector3 vectory_middshoulders =  -(middhips - middshoulders);
        Vector3 vectorz_middshoulders = Vector3.Cross(-vectorx_middshoulders, vectory_middshoulders);
        // left SIDE
        // leftshoulder
        Vector3 vector_left_arm = Body_dataset[5].transform.localPosition - Body_dataset[4]
.transform.localPosition;
        Vector3 projectedvector_sagital_lshoulder = Vector3.ProjectOnPlane(vector_left_arm, vectorx_middshoulders);
        float sagital_left_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_sagital_lshoulder, -
(vectory_middshoulders), vectorx_middshoulders));
        Vector3 projectedvector_frontal_lshoulder = Vector3.ProjectOnPlane(vector_left_arm, vectorz_middshoulders);
        float frontal_left_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_frontal_lshoulder, -
vectory_middshoulders, vectorz_middshoulders));
        Vector3 projectedvector_transverse_lshoulder = Vector3.ProjectOnPlane(vector_left_arm,
vectory_middshoulders);
        float transverse_left_angle = Vector3.SignedAngle(projectedvector_transverse_lshoulder,
vectorz_middshoulders, vectory_middshoulders);
        // leftelbow
        Vector3 vector_left_forearm = Body_dataset[6].transform.localPosition - Body_dataset[5]
.transform.localPosition;
        Vector3 vector_left_elbowarm = -(Body_dataset[4].transform.localPosition - Body_dataset[5]
.transform.localPosition);
        float left_elbow_flexion = Vector3.Angle(vector_left_forearm, vector_left_elbowarm);
        // leftwrist  // first we have to create the middfingers coordinates
        Vector3 lmiddfingers = Body_dataset[21].transform.localPosition ;
        Vector3 vector_left_hand = (lmiddfingers - Body_dataset[6].transform.localPosition);
        Vector3 vector_left_forearmwrist = -(Body_dataset[5].transform.localPosition - Body_dataset[6]
.transform.localPosition);
        float left_wrist_flexion = Vector3.Angle(vector_left_hand, vector_left_forearmwrist);
        // rightshoulder
        Vector3 vector_right_arm = Body_dataset[9].transform.localPosition - Body_dataset[8]
.transform.localPosition;
        Vector3 projectedvector_sagital_rshoulder = Vector3.ProjectOnPlane(vector_right_arm, vectorx_middshoulders)
;
        float sagital_right_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_sagital_rshoulder, -
vectory_middshoulders, vectorx_middshoulders));
        Vector3 projectedvector_frontal_rshoulder = Vector3.ProjectOnPlane(vector_right_arm, vectorz_middshoulders)
;
        float frontal_right_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_frontal_rshoulder, -
vectory_middshoulders, vectorz_middshoulders));
        Vector3 projectedvector_transverse_rshoulder = Vector3.ProjectOnPlane(vector_right_arm,
vectory_middshoulders);
        float transverse_right_angle = Mathf.Abs(Vector3.SignedAngle(projectedvector_transverse_rshoulder,
vectorz_middshoulders, -vectory_middshoulders));
        // rightlbow
        Vector3 vector_right_forearm = Body_dataset[10].transform.localPosition - Body_dataset[9]
.transform.localPosition;
        Vector3 vector_right_elbowarm = -(Body_dataset[8].transform.localPosition - Body_dataset[9]
.transform.localPosition);
        float right_elbow_flexion = Vector3.Angle(vector_right_forearm, vector_right_elbowarm);
        // rightwrist  // first we have to create the middfingers coordinates
        Vector3 rmiddfingers = Body_dataset[23].transform.localPosition;
        Vector3 vector_right_hand = (rmiddfingers - Body_dataset[10].transform.localPosition);
        Vector3 vector_right_forearmwrist = -(Body_dataset[9].transform.localPosition - Body_dataset[10]
.transform.localPosition);
        float right_wrist_flexion = Vector3.Angle(vector_right_hand, vector_right_forearmwrist);
        // draw left
        Debug.DrawLine(middshoulders,  middshoulders+vectory_middshoulders);
        Debug.DrawLine(middshoulders,  middshoulders+vectorx_middshoulders, Color.green);
        Debug.DrawLine(middshoulders,  middshoulders+vectorz_middshoulders*10f, Color.red); //
        Debug.DrawLine(middshoulders,  middhips); //
        Debug.DrawLine(left_shoulder,  left_shoulder + (-vectory_middshoulders)  );
        Debug.DrawLine(left_shoulder,  left_shoulder +vectory_middshoulders);
        Debug.DrawLine(left_shoulder,  left_shoulder+vectorx_middshoulders, Color.green);
        Debug.DrawLine(left_shoulder,  left_shoulder+vectorz_middshoulders*10f, Color.red); //
        Debug.DrawLine(middshoulders,  middhips); //
        Debug.DrawLine(right_shoulder,  right_shoulder + (-vectory_middshoulders)    );
        Debug.DrawLine(right_shoulder,  right_shoulder+vectory_middshoulders);
        Debug.DrawLine(right_shoulder,  right_shoulder+vectorx_middshoulders, Color.green);
        Debug.DrawLine(right_shoulder,  right_shoulder+vectorz_middshoulders*10f, Color.red); //
        Debug.Log("DATASET BACK:left shoulder sagital_left_angle: " + sagital_left_angle + " left shoulder
frontal_left_angle: " + frontal_left_angle +" left shoulder transverse_left_angle: " + transverse_left_angle + "
left_elbow_flexion: " + left_elbow_flexion +" left_wrist_flexion: " + left_wrist_flexion);
        // HOMBRO CORONAL: DESDE DELANTE
        Debug.DrawLine(middshoulders,  middshoulders+vectory_middshoulders);
        Debug.DrawLine(middshoulders,  middshoulders+vectorx_middshoulders, Color.green);
```

11.4. Anexo: (Código iPython) Pipeline de Análisis cinemático del movimiento y Conteo Automático en los distintos grupos de datos

Pipeline_conteos

November 1, 2023

0.1 Pipeline de Análisis cinemático del movimiento y Conteo Automático en los distintos grupos de datos

[ ]:
```
from FuncionesPipeline import PrepareAndLoadDataset, PrepareAndLoadMediapipe,␣
 ↪FuncionFFrecuencia, DynamicTechnicalActionsCounter,␣
 ↪ClasificacionCoeficiente, Lin_ccc
import numpy as np
import pandas as pd
from scipy import stats
from scipy.stats import mannwhitneyu
import plotly.io as pio
import plotly.graph_objects as go
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
# Set default renderers and templates for Plotly
pio.renderers.default = 'vscode'
pio.templates.default = 'plotly'

# Define parametros
n_VideosToAnalize = 20
window_size = 5
Smooth = True
umbral_temporal = 0.5
umbral_amplitud = 15

# imprimir
printdataset = False
printdataset_smooth = False
printmediapipe = False
printmediapipe_smooth = False

resultadoconteo=[]
#Lado Izquierdo
ConteoLdataset, LDatasetResults  = DynamicTechnicalActionsCounter(
    PrepareAndLoadDataset,n_VideosToAnalize, umbral_temporal , umbral_amplitud,␣
 ↪"Dataset", "izquierdo", Smooth=False, print_func=printdataset)
```

```
# Procesa Smoothed Dataset
ConteoLdataset_smoothed, LDatasetResults_smoothed =␣
 ↪DynamicTechnicalActionsCounter(
    PrepareAndLoadDataset, n_VideosToAnalize, umbral_temporal,␣
 ↪umbral_amplitud,"Dataset", "izquierdo", Smooth=True,␣
 ↪window_size=window_size, print_func=printdataset)
# Procesa Mediapipe
ConteoLMediapipe, LMediapipeResults = DynamicTechnicalActionsCounter(
    PrepareAndLoadMediapipe, n_VideosToAnalize, umbral_temporal,␣
 ↪umbral_amplitud, "Mediapipe", "izquierdo", Smooth=False,␣
 ↪print_func=printmediapipe)
# Procesa Smoothed Mediapipe
ConteoLMediapipe_smoothed, LMediapipeResults_smoothed  =␣
 ↪DynamicTechnicalActionsCounter(
    PrepareAndLoadMediapipe, n_VideosToAnalize, umbral_temporal,␣
 ↪umbral_amplitud, "Mediapipe","izquierdo", Smooth=True,␣
 ↪window_size=window_size, print_func= printmediapipe_smooth)
print("ConteoLdataset:", ConteoLdataset)
print("ConteoLdataset_smoothed:", ConteoLdataset_smoothed)
print("ConteoLMediapipe:", ConteoLMediapipe)
print("ConteoLMediapipe_smoothed:", ConteoLMediapipe_smoothed)

# Lado derecho
ConteoRdataset, RDatasetResults  = DynamicTechnicalActionsCounter(
    PrepareAndLoadDataset, n_VideosToAnalize,umbral_temporal, umbral_amplitud,␣
 ↪"Dataset", "derecho", Smooth=False, print_func=printdataset)
# Process Smoothed Dataset
ConteoRdataset_smoothed, RDatasetResults_smoothed =␣
 ↪DynamicTechnicalActionsCounter(
    PrepareAndLoadDataset, n_VideosToAnalize, umbral_temporal, umbral_amplitud,␣
 ↪"Dataset", "derecho", Smooth=True, window_size=window_size, ␣
 ↪print_func=printdataset)
#Process Mediapipe
ConteoRMediapipe, RMediapipeResults = DynamicTechnicalActionsCounter(
    PrepareAndLoadMediapipe, n_VideosToAnalize, umbral_temporal,␣
 ↪umbral_amplitud, "Mediapipe", "derecho", Smooth=False,␣
 ↪print_func=printmediapipe)
# Process Smoothed Mediapipe
ConteoRMediapipe_smoothed, RMediapipeResults_smoothed  =␣
 ↪DynamicTechnicalActionsCounter(
    PrepareAndLoadMediapipe, n_VideosToAnalize,umbral_temporal,␣
 ↪umbral_amplitud, "Mediapipe", "derecho", Smooth=True,␣
 ↪window_size=window_size, print_func=printmediapipe)
print("ConteoRdataset:", ConteoRdataset)
print("ConteoRdataset_smoothed:", ConteoRdataset_smoothed)
print("ConteoRMediapipe:", ConteoRMediapipe)
```

```
print("ConteoRMediapipe_smoothed:", ConteoRMediapipe_smoothed)
resultadoconteo.
 ↪append(f'{ConteoLdataset},{ConteoLdataset_smoothed},{ConteoLMediapipe},␣
 ↪{ConteoLMediapipe_smoothed},␣
 ↪{ConteoRdataset},{ConteoRdataset_smoothed},{ConteoRMediapipe},␣
 ↪{ConteoRMediapipe_smoothed}')
```

[ ]:
```
dfLDatasetResults = pd.DataFrame()
dfLdatasetresults_smoothed = pd.DataFrame()
dfLMediapipeResults = pd.DataFrame()
dfLMediapipeResults_smoothed = pd.DataFrame()
for i in range(len(LDatasetResults)):
    for key, value in LDatasetResults[i].items():
        dfLDatasetResults[f"{key}"] = value
    for key, value in LDatasetResults_smoothed[i].items():
        dfLdatasetresults_smoothed[f"{key}"] = value
    for key, value in LMediapipeResults[i].items():
        dfLMediapipeResults[f"{key}"] = value
    for key, value in LMediapipeResults_smoothed[i].items():
        dfLMediapipeResults_smoothed[f"{key}"] = value

dfRDatasetResults = pd.DataFrame()
dfRdatasetresults_smoothed = pd.DataFrame()
dfRMediapipeResults = pd.DataFrame()
dfRMediapipeResults_smoothed = pd.DataFrame()
for i in range(len(RDatasetResults)):
    for key, value in RDatasetResults[i].items():
        dfRDatasetResults[f"{key}"] = value
    for key, value in RDatasetResults_smoothed[i].items():
        dfRdatasetresults_smoothed[f"{key}"] = value
    for key, value in RMediapipeResults[i].items():
        dfRMediapipeResults[f"{key}"] = value
    for key, value in RMediapipeResults_smoothed[i].items():
        dfRMediapipeResults_smoothed[f"{key}"] = value
```

0.2 Gráficos de sección 7.1.2. El efecto del suavizado y reducción del valor RMS de las curvas.

[ ]:
```
# Crear una lista de partes del cuerpo
partes_cuerpo_der = ['flexion_hombro_derecho', 'abduccion_hombro_derecho',␣
 ↪'rotacion_hombro_derecho', 'flexion_codo_derecho', 'flexion_muneca_derecha']
# Crear subplots para cada parte del cuerpo
fig, axs = plt.subplots(len(partes_cuerpo_der), 1, figsize=(13, 15),␣
 ↪sharex=True)

# Lista de videos a excluir
videos_a_excluir = [3]
```

```
# Iterar a través de las partes del cuerpo y generar los gráficos
for i, parte_cuerpo in enumerate(partes_cuerpo_der):
    # Crear un diccionario para almacenar los datos RMS por conjunto de datos
    rms_data_der = {
        'Dataset': dfRDatasetResults[[f'video {str(a)} {parte_cuerpo}' for a in␣
 ↪range(1, 21) if a not in videos_a_excluir]].apply(lambda x: np.sqrt(np.
 ↪mean(x**2))),
        'Dataset Suavizado': dfRdatasetresults_smoothed[[f'video {str(a)}␣
 ↪{parte_cuerpo}' for a in range(1, 21) if a not in videos_a_excluir]].
 ↪apply(lambda x: np.sqrt(np.mean(x**2))),
        'Mediapipe': dfRMediapipeResults[[f'video {str(a)} {parte_cuerpo}' for␣
 ↪a in range(1, 21) if a not in videos_a_excluir]].apply(lambda x: np.sqrt(np.
 ↪mean(x**2))),
        'Mediapipe Suavizado': dfRMediapipeResults_smoothed[[f'video {str(a)}␣
 ↪{parte_cuerpo}' for a in range(1, 21) if a not in videos_a_excluir]].
 ↪apply(lambda x: np.sqrt(np.mean(x**2)))}
    # Crear un DataFrame a partir del diccionario
    df_rms_der = pd.DataFrame(rms_data_der)
    sns.stripplot(data=df_rms_der, orient='v', ax=axs[i], color='black',␣
 ↪alpha=0.5, jitter=True)

    # Crear el gráfico de boxplot horizontal en el subplot correspondiente
    sns.boxplot(data=df_rms_der, orient='v', ax=axs[i], showfliers=False,␣
 ↪fliersize= 17)
    # Agregar la mediana al gráfico
    medianas = df_rms_der.median()
    for j, mediana in enumerate(medianas):
        axs[i].text(j, mediana, f'{mediana:.1f}', ha='left', va='center',␣
 ↪color='white', fontsize=14)
    axs[i].set_ylabel("RMS Ángulo",fontsize=14)
    axs[i].set_xlabel("Grupo de Datos",fontsize=14)
    axs[i].set_title(f"Dispersión del valor RMS en el movimiento de␣
 ↪{parte_cuerpo}", fontsize=16)
    axs[i].set_xlim(-0.5, 4)
plt.tight_layout()
plt.show()
```

[ ]:
```
#Lado Izquierdo
partes_cuerpo_izq = ['flexion_hombro_izquierdo', 'abduccion_hombro_izquierdo',␣
 ↪'rotacion_hombro_izquierdo', 'flexion_codo_izquierdo',␣
 ↪'flexion_muneca_izquierda']
fig, axs = plt.subplots(len(partes_cuerpo_izq), 1, figsize=(13, 15),␣
 ↪sharex=True)
videos_a_excluir = [3]
for i, parte_cuerpo in enumerate(partes_cuerpo_izq):
    rms_data_izq = {
```

```
        'Dataset': dfLDatasetResults[[f'video {str(a)} {parte_cuerpo}' for a in␣
↪range(1, 21) if a not in videos_a_excluir]].apply(lambda x: np.sqrt(np.
↪mean(x**2))),
        'Dataset Suavizado': dfLdatasetresults_smoothed[[f'video {str(a)}␣
↪{parte_cuerpo}' for a in range(1, 21) if a not in videos_a_excluir]].
↪apply(lambda x: np.sqrt(np.mean(x**2))),
        'Mediapipe': dfLMediapipeResults[[f'video {str(a)} {parte_cuerpo}' for␣
↪a in range(1, 21) if a not in videos_a_excluir]].apply(lambda x: np.sqrt(np.
↪mean(x**2))),
        'Mediapipe Suavizado': dfLMediapipeResults_smoothed[[f'video {str(a)}␣
↪{parte_cuerpo}' for a in range(1, 21) if a not in videos_a_excluir]].
↪apply(lambda x: np.sqrt(np.mean(x**2)))}
    df_rms_izq = pd.DataFrame(rms_data_izq)
    sns.stripplot(data=df_rms_izq, orient='v', ax=axs[i], color='black',␣
↪alpha=0.5, jitter=True)

    sns.boxplot(data=df_rms_izq, orient='v', ax=axs[i], showfliers=False,␣
↪fliersize= 17)
    medianas = df_rms_izq.median()
    for j, mediana in enumerate(medianas):
        axs[i].text(j, mediana, f'{mediana:.1f}', ha='left', va='center',␣
↪color='white', fontsize=14)
        axs[i].set_ylabel("RMS Ángulo",fontsize=14)
    axs[i].set_xlabel("Grupo de Datos",fontsize=14)
    axs[i].set_title(f"Dispersión del valor RMS del movimiento de␣
↪{parte_cuerpo}", fontsize=16)
    axs[i].set_xlim(-0.5, 4)
plt.tight_layout()
plt.show()
```

0.3 Concordancia y Correlación

0.3.1 Gráficos de Resuldados Anexo 11.7.2: Gráficos de las curvas del movimiento, ejemplos en base a la concordancia y correlación

[ ]:
```
i = 0
r_Llin_diff =[]
r_Lresults = []
lin_Lresults= []
for L in dfLMediapipeResults.columns:
    if "time" in L:
        continue
    else:
        # Truncate datasets to have the same length
        min_len = min(len(dfLdatasetresults_smoothed[L].dropna()),␣
↪len(dfLMediapipeResults_smoothed[L].dropna()))
```

```
        dfLdatasetresults_smoothed_truncated = dfLdatasetresults_smoothed[L][:
↪min_len].dropna()
        dfLMediapipeResults_smoothed_truncated =␣
↪dfLMediapipeResults_smoothed[L][:min_len].dropna()

        # R pearson
        r = stats.pearsonr(dfLdatasetresults_smoothed_truncated,␣
↪dfLMediapipeResults_smoothed_truncated)[0]
        # Lin's concordance correlation coefficient
        lin_cc = Lin_ccc(dfLdatasetresults_smoothed_truncated ,␣
↪dfLMediapipeResults_smoothed_truncated)
        r_Lresults.append({'Video-segmento corporal': L, 'Correlación r': r,␣
↪'Categoría Correlación': ClasificacionCoeficiente(r)})
        lin_Lresults.append({'Video-segmento corporal': L,'Concordancia Lin':␣
↪lin_cc, 'Categoría Concordancia': ClasificacionCoeficiente(lin_cc)})

        fig = go.Figure()
        # línea del movimiento
        fig.add_trace(go.Scatter(
            y=dfLdatasetresults_smoothed_truncated,
            mode='lines+markers',
            name='Dataset Suav (Der)',
            x=dfLdatasetresults_smoothed[f'video {i+1} time'][:min_len],
            line=dict(color='black')))
        fig.add_trace(go.Scatter(
            y=dfLMediapipeResults_smoothed_truncated,
            mode='lines+markers',
            name='Mediapipe Suav (Der)',
            x=dfLMediapipeResults_smoothed[f'video {i+1} time'][:min_len],
            line=dict(color='lawngreen')))
        max_time = max(dfLdatasetresults_smoothed[f'video {i+1} time'][:
↪min_len])
        fig.update_xaxes(range=[0, max_time])
        fig.update_layout(title=f'Ángulos de {L} Coeficiente Lin\'s CC: {lin_cc␣
↪:.3f} Pearson (r):{r :.3f} ',
                          xaxis_title='Tiempo (s)',
                          yaxis_title='Grados (°)',
                          title_font=dict(size=20),legend=dict(
                          orientation="h",  # Configura la orientación horizontal
                          x=0.6,  # Ajusta la posición horizontal
                          y=-0.05, font=dict(size=17))) # Ajusta la posición␣
↪vertical (debajo del gráfico)
        #fig.show()
        i += 1
        if i == n_VideosToAnalize:
            i = 0
```

[ ]:
```
i = 0
r_Rlin_diff =[]
r_Rresults = []
lin_Rresults= []
for R in dfRMediapipeResults.columns:
    if "time" in R:
        continue
    else:
        min_len = min(len(dfRdatasetresults_smoothed[R].dropna()),␣
↪len(dfRMediapipeResults_smoothed[R].dropna()))
        dfRdatasetresults_smoothed_truncated = dfRdatasetresults_smoothed[R][:
↪min_len].dropna()
        dfRMediapipeResults_smoothed_truncated =␣
↪dfRMediapipeResults_smoothed[R][:min_len].dropna()
        # R pearson
        r = stats.pearsonr(dfRdatasetresults_smoothed_truncated,␣
↪dfRMediapipeResults_smoothed_truncated)[0]
        # Calculate Lin's concordance correlation coefficient
        lin_cc =␣
↪Lin_ccc(dfRdatasetresults_smoothed_truncated,dfRMediapipeResults_smoothed_truncated)
        r_Rresults.append({'Video-segmento corporal': R, 'Correlación r': r,␣
↪'Categoría Correlación': ClasificacionCoeficiente(r)})
        lin_Rresults.append({'Video-segmento corporal': R,'Concordancia Lin':␣
↪lin_cc, 'Categoría Concordancia': ClasificacionCoeficiente(lin_cc)})

        fig = go.Figure()
        fig.add_trace(go.Scatter(
            y=dfRdatasetresults_smoothed_truncated,
            mode='lines+markers',
            name='Dataset Suav (Der)',
            x=dfRdatasetresults_smoothed[f'video {i+1} time'][:min_len],
            line=dict(color='black')))
        fig.add_trace(go.Scatter(
            y=dfRMediapipeResults_smoothed_truncated,
            mode='lines+markers',
            name='Mediapipe Suav (Der)',
            x=dfRMediapipeResults_smoothed[f'video {i+1} time'][:min_len],
            line=dict(color='lawngreen')))
        max_time = max(dfRdatasetresults_smoothed[f'video {i+1} time'][:
↪min_len])
        fig.update_xaxes(range=[0, max_time])
        fig.update_layout(title=f'Ángulos de {R} Coeficientes Concordancia de␣
↪Lin\'s CC: {lin_cc:.3f} y Correlación de Pearson (r):{r:.3f} ',#y␣
↪Coeficiente de Bland-Altman {bland_altman_coefficient:.3f}
                          xaxis_title='Tiempo (s)',
                          yaxis_title='Grados (°)',
```

```
                        title_font=dict(size=20),legend=dict(
                        orientation="h",  # Configura la orientación horizontal
                        x=0.6,  # Ajusta la posición horizontal
                        y=-0.05, font=dict(size=17))) # Ajusta la posición␣
↪vertical (debajo del gráfico)
        #fig.show()
        i += 1
        if i == n_VideosToAnalize:
            i = 0
```

0.3.2 Tablas de Anexo: 11.7.1. Tablas de concordancia y correlación concordancia y correlación

[ ]:
```
# Create a DataFrame to store the correlation results
correlation_dfR = pd.DataFrame(r_Rresults)
concordanciaLin_dfR = pd.DataFrame(lin_Rresults)

# Ordena 'Correlation' de mayor a menor
correlation_dfR = correlation_dfR.sort_values(by='Correlación r',␣
↪ascending=False)
concordanciaLin_dfR = concordanciaLin_dfR.sort_values(by='Concordancia Lin',␣
↪ascending=False)
#Guarda en dataframe
correlation_dfL = pd.DataFrame(r_Lresults)
concordanciaLin_dfL = pd.DataFrame(lin_Lresults)
#lado izquerdo
correlation_dfL = correlation_dfL.sort_values(by='Correlación r',␣
↪ascending=False)
concordanciaLin_dfL = concordanciaLin_dfL.sort_values(by='Concordancia Lin',␣
↪ascending=False)

concordancia_correlacion_izq = pd.concat([concordanciaLin_dfL,␣
↪correlation_dfL], axis=1, join='inner')
columnas = ['Video-segmento corporal', 'Concordancia Lin',  'Categoría␣
↪Concordancia','Correlación r', 'Categoría Correlación']
concordancia_correlacion_izq = concordancia_correlacion_izq[columnas]
concordancia_correlacion_izq = concordancia_correlacion_izq.iloc[:,1:]
#print(LLC.to_latex())
concordancia_correlacion_izq
```

[ ]:
```
concordancia_correlacion_der = pd.concat([concordanciaLin_dfR,␣
↪correlation_dfR], axis=1, join='inner')
columnas = ['Video-segmento corporal', 'Concordancia Lin',  'Categoría␣
↪Concordancia','Correlación r', 'Categoría Correlación']
concordancia_correlacion_der = concordancia_correlacion_der[columnas]
concordancia_correlacion_der = concordancia_correlacion_der.iloc[:,1:]
```

```
#print(LLC.to_latex())
concordancia_correlacion_der
```

0.3.3 Tabla 15: del Anexo 11.7.3: Orden decreciente de los videos según el el valor promedio de concordancia de Lin

[ ]:
```
def clean_data(correlation_dfL, lado):
    """ crea columna 'N° de Video' a partir de 'Video-segmento corporal'"""
    def N_de_Video(Video_segmento_corporal):
        index1 = [i for i in range(len(Video_segmento_corporal)) if␣
 ↪Video_segmento_corporal.startswith(" ", i)][1]
        return Video_segmento_corporal[:index1]
    correlation_dfL.insert(1, "N° de Video", correlation_dfL.apply(lambda row :␣
 ↪N_de_Video(row["Video-segmento corporal"]), axis=1))
    # Performed 1 aggregation grouped on columns: 'N° de Video', 'Segmento␣
 ↪Corporal'
    correlation_dfL = correlation_dfL.groupby(['N° de Video']).
 ↪agg({'Concordancia Lin': ['mean', 'std']}).reset_index()

    # Sort by column: 'Correlacion_r_mean' (descending)
    correlation_dfL = correlation_dfL.sort_values(by=('Concordancia Lin',␣
 ↪'mean'), ascending=False)
    correlation_dfL.rename(columns={'Concordancia Lin': f'Concordancia Lin␣
 ↪{lado}'}, inplace=True)
    return correlation_dfL

concordanciaLin_dfL_clean = clean_data(concordanciaLin_dfL.copy(), "izquierdo")
concordanciaLin_dfR_clean = clean_data(concordanciaLin_dfR.copy(), "derecho")
Concordancias=pd.concat([concordanciaLin_dfL_clean,concordanciaLin_dfR_clean.
 ↪iloc[:,1:]], axis=1)
pd.set_option('display.float_format', '{:.2f}'.format)

Concordancias[('Concordancia Lin izquierdo','Clasificación de Concordancia␣
 ↪Lin')] = Concordancias[('Concordancia Lin izquierdo','mean')].
 ↪apply(ClasificacionCoeficiente)
Concordancias[('Concordancia Lin derecho','Clasificación de Concordancia Lin')]␣
 ↪= Concordancias[('Concordancia Lin derecho','mean')].
 ↪apply(ClasificacionCoeficiente)
Concordancias = Concordancias[[('N° de Video', ''), ('Concordancia Lin␣
 ↪izquierdo', 'mean'), ('Concordancia Lin izquierdo', 'std'), ('Concordancia␣
 ↪Lin izquierdo','Clasificación de Concordancia Lin'),
           ('Concordancia Lin derecho', 'mean'), ('Concordancia Lin derecho',␣
 ↪'std'), ('Concordancia Lin derecho', 'Clasificación de Concordancia Lin')]]
Concordancias
```

0.3.4 Tabla 9: Coeficiente de Concordancia de Lin entre los ángulos de obtenidos a partir de DatasetSuavizado y MediapipeSuavizado en cada extremidad. Sección de Resultados 7.1.3. Concordancia y correlación entre ángulos 3D del Dataset y Mediapipe

[ ]:
```
group_keywords = ['flexion_hombro', 'abduccion_hombro', 'rotacion_hombro',␣
 ↪'flexion_codo', 'flexion_muneca']
# Extrae categoría de la columna 'Video-segmento corporal' basado en palabras␣
 ↪claves
def extract_category(segmento_corporal):
    for keyword in group_keywords:
        if keyword in segmento_corporal:
            return keyword
    return "Otro"
concordanciaLin_dfL['Segmento Corporal'] = concordanciaLin_dfL['Video-segmento␣
 ↪corporal'].apply(extract_category)
concordanciaLin_dfR['Segmento Corporal'] = concordanciaLin_dfR['Video-segmento␣
 ↪corporal'].apply(extract_category)

# Agrupa por Categorías, and calcula promedio y conteo de videos según␣
 ↪'Correlación'
grouped_dfL_concordancia = concordanciaLin_dfL.groupby(['Segmento Corporal',␣
 ↪'Categoría Concordancia'], as_index=False).agg({
    'Concordancia Lin': 'mean',
    'Video-segmento corporal': 'count'})
grouped_dfR_concordancia = concordanciaLin_dfR.groupby(['Segmento␣
 ↪Corporal','Categoría Concordancia'], as_index=False).agg({
    'Concordancia Lin': 'mean',
    'Video-segmento corporal': 'count'})

# Cambia el nombre de 'Video-segmento corporal' --> 'Cantidad de videos
grouped_dfL_concordancia = grouped_dfL_concordancia.
 ↪rename(columns={'Video-segmento corporal': 'Cantidad de videos'})
grouped_dfL_concordancia = grouped_dfL_concordancia.
 ↪sort_values(by='Concordancia Lin', ascending=False)
grouped_dfR_concordancia = grouped_dfR_concordancia.
 ↪rename(columns={'Video-segmento corporal': 'Cantidad de videos'})
grouped_dfR_concordancia = grouped_dfR_concordancia.
 ↪sort_values(by='Concordancia Lin', ascending=False)
# Reordena las columnas y ordena los niveles
Concordancia_izq = pd.pivot_table(grouped_dfL_concordancia, index= ['Segmento␣
 ↪Corporal'], columns='Categoría Concordancia', values='Cantidad de videos' ,␣
 ↪aggfunc='sum')
orden_columna_izq = ["Alta (+)", "Moderada (+)","Débil (+)","Muy Débil (+)",␣
 ↪"Muy Débil (-)", "Débil (-)"]
Concordancia_izq = Concordancia_izq[orden_columna_izq]
```

```
Concordancia_izq
```

[ ]:
```
#Extremidad derecha
Concordancia_der = pd.pivot_table(grouped_dfR_concordancia, index= ['Segmento␣
 ↪Corporal'], columns='Categoría Concordancia', values='Cantidad de videos' ,␣
 ↪aggfunc='sum')
orden_columna_der = ["Alta (+)", "Moderada (+)","Débil (+)","Muy Débil (+)",␣
 ↪"Muy Débil (-)"]
Concordancia_der = Concordancia_der[orden_columna_der]
Concordancia_der
```

0.4 Resultados Sección 7.2. «Conteo humano de consenso» (ConteoHConsenso) de «Acciones Técnicas Dinámicas» («ATD»)

0.4.1 Tabla 10a y Tabla 10b Resultados de conteo humano (P 1, P 2 y P 3) de «acciones técnicas dinámicas» por minuto y «Conteo Humano de Consenso», como promedio y desviación estándar, junto con el respectivo puntaje del "FF" correspondiente a cada extremidad, en cada uno de los videos.

[ ]:
```
from FuncionesPipeline import FuncionFFrecuencia
import numpy as np
import pandas as pd
from scipy import stats
from scipy.stats import mannwhitneyu
import plotly.io as pio
import plotly.graph_objects as go
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
pd.set_option('display.float_format', '{:.1f}'.format)

A5CP = pd.DataFrame({'Izq_min':␣
 ↪[15,30,29,16,34,18,37,29,18,24,18,12,33,14,32,11,10,13,18,13], 'Der_min':
 ↪[27,17,16,27,18,33,19,15,35,19,18,23,18,25,21,21,19,23,9,25]})
A5CP.index = range(1,21)
C6VC = pd.DataFrame({'Izq_min':␣
 ↪[19,44,42,20,52,23,55,41,27,35,20,18,50,20,42,16,19,18,28,18], 'Der_min':
 ↪[38,24,24,39,25,45,28,21,53,29,25,36,26,36,36,31,20,33,14,36]})
C6VC.index = range(1,21)
B8MP = pd.DataFrame({'Izq_min':
 ↪[14,29,29,15,33,16,37,29,18,24,18,12,33,13,29,11,16,11,18,13],'Der_min':
 ↪[26,16,15,27,17,31,20,15,35,19,17,22,17,24,22,21,14,22,9,25]})
B8MP.index = range(1,21)
timevideos= pd.DataFrame({'timevideos':['3:04','2:31','2:23','2:44','2:11','2:
 ↪22','1:59','2:35','2:08','2:32','3:18','3:13','2:13','2:58','2:14','3:27','3:
 ↪54','3:19','4:02','2:55']})
```

```
timevideos['decimal_minutes'] = timevideos['timevideos'].apply(lambda x: int(x.
 ↪split(':')[0]) + int(x.split(':')[1])/60)
timevideos.index = range(1,21)

consenso = pd.DataFrame()
consenso.index = range(1, 21)
ConteoHumano= pd.concat([A5CP,B8MP,C6VC], axis=1, keys=['A5CP','B8MP','C6VC'])
for col in ['Izq_min', 'Der_min']:
    consenso[f'{col}'] = ConteoHumano.loc[:, (slice(None), col)].mean(axis=1).
 ↪round(decimals=1)
consenso['F.Frec.Der'] = consenso['Der_min'].apply(FuncionFFrecuencia)
consenso['F.Frec.Izq'] = consenso['Izq_min'].apply(FuncionFFrecuencia)

ConteoHConsenso_imprimir = pd.DataFrame()
ConteoHConsenso_imprimir.index =range(1, 21)
for col in ['Izq_min', 'Der_min']:
    if 'FF' not in col:
        mean_col = ConteoHumano.loc[:, (slice(None), col)].mean(axis=1)
        std_col = ConteoHumano.loc[:, (slice(None), col)].std(axis=1)
        ConteoHConsenso_imprimir[f'{col}'] = [f'{mean :.1f} ± {std :.1f}' for␣
 ↪mean, std in zip(mean_col, std_col)]
    else:
        mean_col = ConteoHumano.loc[:, (slice(None), col)].mean(axis=1)
        std_col = ConteoHumano.loc[:, (slice(None), col)].std(axis=1)
        ConteoHConsenso_imprimir[f'{col}'] = [f'{mean :.1f} ± {std :.1f}' for␣
 ↪mean, std in zip(mean_col, std_col)]
ConteoHConsenso_imprimir = ConteoHConsenso_imprimir[['Izq_min', 'Der_min']]
ConteoHConsenso_imprimir =pd.concat([ConteoHConsenso_imprimir,consenso.iloc[:,2:
 ↪]], axis=1)
indice = [9, 6, 4, 1, 20, 14, 12, 15, 18, 16, 10, 7, 13, 5, 11, 2, 3, 17, 8, 19]
ConteoHConsenso_imprimir['n° video'] = [i for i in range(1,21)]
ConteoHConsenso_imprimir=ConteoHConsenso_imprimir[['n° video','Izq_min',  'F.
 ↪Frec.Izq', 'Der_min','F.Frec.Der']]
ConteoHConsenso_imprimir = ConteoHConsenso_imprimir.reindex(index=indice)
ConteoHConsenso_imprimir =pd.concat([ConteoHConsenso_imprimir], axis=1, ␣
 ↪keys=['Conteo Humano de Consenso'])

renamecolumns = ConteoHumano.rename(columns = {"A5CP":"P1",          "B8MP":
 ↪"P2", "C6VC":"P3"} , inplace=True)
conteo_imprimir = ConteoHumano.copy()
indice = [9, 6, 4, 1, 20, 14, 12, 15, 18, 16, 10, 7, 13, 5, 11, 2, 3, 17, 8, 19]
conteo_imprimir[("",'n° video')] = range(1,21)
conteo_imprimir = conteo_imprimir.reindex(indice)
consenso_copy = consenso.copy()
```

```
conteo_imprimir[[(  '', 'n° video'), ('P1',  'Izq_min'), ('P1',  'Der_min'),␣
 ↪('P2',  'Izq_min'), ('P2',  'Der_min'),('P3',  'Izq_min'), ('P3', ␣
 ↪'Der_min')]]
```

[ ]:
```
ConteoHConsenso_imprimir
```

0.5 Pruebas en 50 combinaciones de umbral

0.6 Descrito en la sección de Métodos:

0.7 6.4.4. Diseño de pruebas para el contador automático de acciones técnicas dinámicas

[ ]:
```
"""from FuncionesPipeline import PrepareAndLoadDataset,␣
 ↪PrepareAndLoadMediapipe, DynamicTechnicalActionsCounter
import plotly.io as pio
from itertools import product
import gc
# Set default renderers and templates for Plotly
pio.renderers.default = 'vscode'
pio.templates.default = 'plotly'

# Define parameters
n_VideosToAnalize = 20
window_size = 5
Smooth = True
umbral_temporal = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1]
umbral_amplitud = [5,10,15,20,25]

# imprimir
printdataset = False
printdataset_smooth = False
printmediapipe = False
printmediapipe_smooth = False

resultadoConteo =[]
for ut, ua in product(umbral_temporal,umbral_amplitud):
    #Lado Izquierdo
    ConteoLdataset, LDatasetResults  = DynamicTechnicalActionsCounter(
        PrepareAndLoadDataset,n_VideosToAnalize, ut , ua, "Dataset",␣
 ↪"izquierdo", Smooth=False, print_func=printdataset)
    # Process Smoothed Dataset
    ConteoLdataset_smoothed, LDatasetResults_smoothed =␣
 ↪DynamicTechnicalActionsCounter(
        PrepareAndLoadDataset, n_VideosToAnalize, ut, ua,"Dataset",␣
 ↪"izquierdo", Smooth=True, window_size=window_size, print_func=printdataset)
    # Process Mediapipe
    ConteoLMediapipe, LMediapipeResults = DynamicTechnicalActionsCounter(
```

```
        PrepareAndLoadMediapipe, n_VideosToAnalize, ut, ua, "Mediapipe",␣
↪"izquierdo", Smooth=False, print_func=printmediapipe)
    # Process Smoothed Mediapipe
    ConteoLMediapipe_smoothed, LMediapipeResults_smoothed  =␣
↪DynamicTechnicalActionsCounter(
        PrepareAndLoadMediapipe, n_VideosToAnalize, ut, ua,␣
↪"Mediapipe","izquierdo", Smooth=True, window_size=window_size, print_func=␣
↪printmediapipe_smooth)

    # Lado derecho
    ConteoRdataset, RDatasetResults  = DynamicTechnicalActionsCounter(
        PrepareAndLoadDataset, n_VideosToAnalize,ut, ua, "Dataset", "derecho",␣
↪Smooth=False, print_func=printdataset)
    # Process Smoothed Dataset
    ConteoRdataset_smoothed, RDatasetResults_smoothed =␣
↪DynamicTechnicalActionsCounter(
        PrepareAndLoadDataset, n_VideosToAnalize, ut, ua, "Dataset", "derecho",␣
↪Smooth=True, window_size=window_size,  print_func=printdataset)
    #Process Mediapipe
    ConteoRMediapipe, RMediapipeResults = DynamicTechnicalActionsCounter(
        PrepareAndLoadMediapipe, n_VideosToAnalize, ut, ua, "Mediapipe",␣
↪"derecho", Smooth=False, print_func=printmediapipe)
    # Process Smoothed Mediapipe
    ConteoRMediapipe_smoothed, RMediapipeResults_smoothed  =␣
↪DynamicTechnicalActionsCounter(
        PrepareAndLoadMediapipe, n_VideosToAnalize,ut, ua, "Mediapipe",␣
↪"derecho", Smooth=True, window_size=window_size,␣
↪print_func=printmediapipe_smooth)

    resultadoConteo.append([f"{ut}, {ua}, {ConteoLdataset},␣
↪{ConteoLdataset_smoothed}, {ConteoLMediapipe}, {ConteoLMediapipe_smoothed},␣
↪{ConteoRdataset}, {ConteoRdataset_smoothed}, {ConteoRMediapipe},␣
↪{ConteoRMediapipe_smoothed}"])
    resultadoConteo[f"ConteoLdataset {ut} , {ua}"] = ConteoLdataset
    resultadoConteo[f"ConteoLdataset_smoothed {ut} , {ua}"] =␣
↪ConteoLdataset_smoothed
    resultadoConteo[f"ConteoLMediapipe {ut} , {ua}"] = ConteoLMediapipe
    resultadoConteo[f"ConteoLMediapipe_smoothed {ut} , {ua}"] =␣
↪ConteoLMediapipe_smoothed
    resultadoConteo[f"ConteoRdataset {ut} , {ua}"] = ConteoRdataset
    resultadoConteo[f"ConteoRdataset_smoothed {ut} , {ua}"] =␣
↪ConteoRdataset_smoothed
    resultadoConteo[f"ConteoRMediapipe {ut} , {ua}"] = ConteoRMediapipe
    resultadoConteo[f"ConteoRMediapipe_smoothed {ut} , {ua}"] =␣
↪ConteoRMediapipe_smoothed
```

```
    with open(r'/home/diego/Documents/AT_Counting/Saved_files/ConteoAutomatico/
↪'+'conteounicodfadfa.txt', 'w') as fp:
        fp.writelines(["%s\n" % item for item in resultadoConteo])
        gc.collect()"""
```

[ ]:
```
# Ruta del archivo CSV y lectura de los resultados de la celda comentada.
from FuncionesPipeline import  FuncionFFrecuencia, ClasificacionCoeficiente,␣
↪Lin_ccc
import numpy as np
import pandas as pd
from scipy import stats
from scipy.stats import mannwhitneyu
import plotly.io as pio
import plotly.graph_objects as go
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

path = '/home/diego/Documents/AT_Counting/Saved_files/'
txt = 'ConteoAutomatico/conteo.txt'
umbral_temporal = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1]
umbral_amplitud = [5,10,15,20,25]

# Lee el archivo CSV, transpone el DataFrame y recorta el dato
conteos = pd.read_csv(path+txt, sep=',', header=None)
nombrecolumna = conteos.T.iloc[:2,:].values.tolist()
conteos = conteos.T.iloc[2:,:]
conteos.columns = nombrecolumna

# Divide el DataFrame en las 50 columnas correspondientes
filas_por_conteo = 20  # Cantidad de filas por cada conteo
exConteoLdataset = conteos.iloc[0:filas_por_conteo,:]
exConteoLdataset.index = range(1,21)
exConteoLdataset_smoothed = conteos.iloc[filas_por_conteo:2 * filas_por_conteo,:
↪]
exConteoLdataset_smoothed.index = range(1,21)
exConteoLMediapipe = conteos.iloc[ 2 * filas_por_conteo:3 * filas_por_conteo,:]
exConteoLMediapipe.index = range(1,21)
exConteoLMediapipe_smoothed = conteos.iloc[3 * filas_por_conteo:4 *␣
↪filas_por_conteo,:]
exConteoLMediapipe_smoothed.index = range(1,21)

exConteoRdataset = conteos.iloc[4 * filas_por_conteo:5 * filas_por_conteo,:]
exConteoRdataset.index = range(1,21)
exConteoRdataset_smoothed = conteos.iloc[ 5 * filas_por_conteo:6 *␣
↪filas_por_conteo,:]
```

```
exConteoRdataset_smoothed.index = range(1,21)
exConteoRMediapipe = conteos.iloc[6 * filas_por_conteo:7 * filas_por_conteo,:]
exConteoRMediapipe.index = range(1,21)
exConteoRMediapipe_smoothed = conteos.iloc[ 7 * filas_por_conteo:8 *␣
 ↪filas_por_conteo,:]
exConteoRMediapipe_smoothed.index = range(1,21)

# concatena en un solo dataframe todos los grupos de datos.
dataframe = pd.concat([exConteoLdataset, exConteoLdataset_smoothed,␣
 ↪exConteoLMediapipe,
                    exConteoLMediapipe_smoothed, exConteoRdataset,␣
 ↪exConteoRdataset_smoothed,
                    exConteoRMediapipe, exConteoRMediapipe_smoothed], axis=1,␣
 ↪join= 'inner', keys=[('Dataset','Izq'),('Dataset Suavizado','Izq'),
                    ('Mediapipe','Izq'), ('Mediapipe Suavizado','Izq'),␣
 ↪('Dataset','Der'),
                    ('Dataset Suavizado','Der'),('Mediapipe','Der'),␣
 ↪('Mediapipe Suavizado','Der')])

PruebasConteoAutomatico = pd.DataFrame()
ffrec = pd.DataFrame()
#Crea la columna con el conteo por minuto al dividir cada conteo automático por␣
 ↪la duración del video
for col in dataframe.columns:
    PruebasConteoAutomatico[col] = dataframe[col]/timevideos['decimal_minutes']
    ffrec[col] = PruebasConteoAutomatico[col].apply(FuncionFFrecuencia)
#redefine el nombre de las columnas
PruebasConteoAutomatico.columns=  dataframe.columns
ffrec.columns = dataframe.columns
```

0.8 Test estadístico Shapiro-Wilk evalúa acaso la muestra distribuye normal.

0.9 Gráficos de Anexo 11.8.1 Análisis de normalidad en las pruebas combinación de umbral.

[ ]:
```
shapiro_conteos={'Grupo': [],'Extremidad':[],'umbral_temporal':
 ↪[],'umbral_amplitud':[],'shapiro':[]}
for col in PruebasConteoAutomatico.columns:
    res = stats.shapiro(PruebasConteoAutomatico[col])
    shapiro_conteos['Grupo'].append(col[0])
    shapiro_conteos['Extremidad'].append(col[1])
    shapiro_conteos['umbral_temporal'].append(col[2])
    shapiro_conteos['umbral_amplitud'].append(col[3])
    shapiro_conteos[f'shapiro'].append(res[1])
    #print(f'{col}={res}')
dfshapiro_conteos= pd.DataFrame(shapiro_conteos)
```

```
shapiro_FactorFrecuencia={'Grupo': [],'Extremidad':[],'umbral_temporal':
 ↪[],'umbral_amplitud':[],'shapiro':[]}
for col in ffrec.columns:
    res = stats.shapiro(ffrec[col])
    shapiro_FactorFrecuencia['Grupo'].append(col[0])
    shapiro_FactorFrecuencia['Extremidad'].append(col[1])
    shapiro_FactorFrecuencia['umbral_temporal'].append(col[2])
    shapiro_FactorFrecuencia['umbral_amplitud'].append(col[3])
    shapiro_FactorFrecuencia[f'shapiro'].append(res[1])
dfshapiro_shapiro_FF= pd.DataFrame(shapiro_FactorFrecuencia)

g = sns.FacetGrid(dfshapiro_conteos, col='Extremidad', row='Grupo',␣
 ↪margin_titles=True)
g.map(sns.histplot, 'shapiro', kde=True)
g.set_axis_labels('Valor p de Shapiro-Wilk', 'Frecuencia')
g.set_titles(col_template="{col_name}", row_template="{row_name}")
for ax in g.axes.flat:
    ax.axvline(x=0.02, color='red', linestyle='--')
plt.subplots_adjust(top=0.9)
g.fig.suptitle('Histogramas de normalidad Shapiro-Wilk de conteo automático en␣
 ↪50 combinaciones de umbral')
plt.show()


g = sns.FacetGrid(dfshapiro_shapiro_FF, col='Extremidad', row='Grupo',␣
 ↪margin_titles=True)
g.map(sns.histplot, 'shapiro', kde=True)
g.set_axis_labels('Valor p de Shapiro-Wilk', 'Frecuencia')
g.set_titles(col_template="{col_name}", row_template="{row_name}")
for ax in g.axes.flat:
    ax.axvline(x=0.02, color='red', linestyle='--')
plt.subplots_adjust(top=0.9)
g.fig.suptitle('Histogramas Shapiro-Wilk de Factor Frecuencia en 50␣
 ↪combinaciones de umbral')
plt.show()
```

[ ]:
```
print(len(dfshapiro_shapiro_FF)) #n° de pruebas de conteo automático.
print(f'Conteos automático no distribuyen normal␣
 ↪{len(dfshapiro_conteos[dfshapiro_conteos["shapiro"]<=0.05])}')
print(f'Conteos automáticos distribuyen normal␣
 ↪{len(dfshapiro_conteos[dfshapiro_conteos["shapiro"]>0.05] )}')
print(f'Factor Frecuencia no distribuyen normal␣
 ↪{len(dfshapiro_shapiro_FF[dfshapiro_shapiro_FF["shapiro"]<=0.05])}')
print(f'Factor Frecuencia distribuyen normal␣
 ↪{len(dfshapiro_shapiro_FF[dfshapiro_shapiro_FF["shapiro"]>0.05])}')␣
 ↪#dfshapiro_conteos
```

```
400
Conteos automático no distribuyen normal 14
Conteos automáticos distribuyen normal 386
Factor Frecuencia no distribuyen normal 337
Factor Frecuencia distribuyen normal 63
```

[ ]:
```python
from FuncionesPipeline import rounder, rmse_y_std

if len(consenso.iloc[:,0])==20:
    consenso = consenso.drop(3)
# Diccionarios para cada resultado
ConteosAuto_Izq = {'Grupo': [], 'Extremidad': [], 'umbral_temporal': [],
 ↪'umbral_amplitud': [],'Error RMSE conteo': [], 'Error RMSE Frecuencia':[],
                  'Lin_conteo':[],'ClasLin_conteo':[], 'Lin_frec':[],
 ↪'ClasLin_frec':[],'ErrorRel_conteo':[], 'ErrorRel_ffrec':[],'pValue_conteo':
 ↪[],'pValue_ffrec':[]}
ConteosAuto_Der = {'Grupo': [], 'Extremidad': [], 'umbral_temporal': [],
 ↪'umbral_amplitud': [], 'Error RMSE conteo': [], 'Error RMSE Frecuencia':[],
                  'Lin_conteo':[],'ClasLin_conteo':[], 'Lin_frec':[],
 ↪'ClasLin_frec':[],'ErrorRel_conteo':[], 'ErrorRel_ffrec':[],'pValue_conteo':
 ↪[],'pValue_ffrec':[]}

# Iterar a través de las columnas de PruebasConteoAutomatico para efectuar los
 ↪distintos cálculos.
for col in PruebasConteoAutomatico.columns:
    #lado izquierdo
    if "Izq" in col:
        error_relativoizq =  (abs(consenso['Izq_min'] -
 ↪PruebasConteoAutomatico[col]) / consenso['Izq_min'] * 100).mean()
        stdizq =  (abs(consenso['Izq_min'] - PruebasConteoAutomatico[col]) /
 ↪consenso['Izq_min'] * 100).std()
        error_relativoizqfrec = (abs(consenso['F.Frec.Izq'] - ffrec[col]) / 9).
 ↪mean()
        stdizqfrec = (abs(consenso['F.Frec.Izq'] - ffrec[col]) / 9).std()
        mae_izq, std_rmse_conteo_izq = rmse_y_std(consenso['Izq_min'],
 ↪PruebasConteoAutomatico[col])
        ffrecuenciaizq, std_frec_izq = rmse_y_std(consenso['F.Frec.Izq'],
 ↪ffrec[col])
        Lin_conteoizq = Lin_ccc(consenso['Izq_min'],
 ↪PruebasConteoAutomatico[col])
        Lin_frecizq = Lin_ccc(consenso['F.Frec.Izq'], ffrec[col])
        clasLinconteoizq= ClasificacionCoeficiente(Lin_conteoizq)
        claslinfrecizq=ClasificacionCoeficiente(Lin_frecizq)
        estadistico_izq, pvalue_izq = mannwhitneyu(consenso['Izq_min'],
 ↪PruebasConteoAutomatico[col])
```

```
        estadistico_izqfrec, pvalue_izqfrec = mannwhitneyu(consenso['F.Frec.
↪Izq'], ffrec[col])
        # Agregar los valores correspondientes a los diccionarios
        ConteosAuto_Izq['Grupo'].append(col[0])
        ConteosAuto_Izq['Extremidad'].append(col[1])
        ConteosAuto_Izq['umbral_temporal'].append(f'{rounder(col[2])}')
        ConteosAuto_Izq['umbral_amplitud'].append(f'{rounder(col[3])}')
        ConteosAuto_Izq['Error RMSE conteo'].append(mae_izq)
        ConteosAuto_Izq['Error RMSE Frecuencia'].append(ffrecuenciaizq)
        ConteosAuto_Izq['Lin_conteo'].append(Lin_conteoizq)
        ConteosAuto_Izq['Lin_frec'].append(Lin_frecizq)
        ConteosAuto_Izq['ClasLin_conteo'].append(clasLinconteoizq)
        ConteosAuto_Izq['ClasLin_frec'].append(claslinfrecizq)
        ConteosAuto_Izq['ErrorRel_conteo'].append(error_relativoizq)
        ConteosAuto_Izq['ErrorRel_ffrec'].append(error_relativoizqfrec)
        ConteosAuto_Izq['pValue_conteo'].append(pvalue_izq)
        ConteosAuto_Izq['pValue_ffrec'].append(pvalue_izqfrec)

    elif 'Der' in col:
        error_relativoder =  (abs(consenso['Der_min'] -␣
↪PruebasConteoAutomatico[col]) / consenso['Der_min'] * 100).mean()
        stdDer =  (abs(consenso['Der_min'] - PruebasConteoAutomatico[col]) /␣
↪consenso['Der_min'] * 100).std()
        error_relativoderfrec = (abs(consenso['F.Frec.Der'] - ffrec[col]) / 9).
↪mean()
        stdderfrec = (abs(consenso['F.Frec.Der'] - ffrec[col]) / 9).std()
        mae_der, std_rmse_conteo_der = rmse_y_std(consenso['Der_min'],␣
↪PruebasConteoAutomatico[col])
        ffrecuenciader, std_frec_der = rmse_y_std(consenso['F.Frec.Der'],␣
↪ffrec[col])
        Lin_conteoder = Lin_ccc(consenso['Der_min'],␣
↪PruebasConteoAutomatico[col])
        Lin_frecder = Lin_ccc(consenso['F.Frec.Der'], ffrec[col])
        clasLinconteoder= ClasificacionCoeficiente(Lin_conteoder)
        claslinfrecder=ClasificacionCoeficiente(Lin_frecder)
        estadistico_der, pvalue_der = mannwhitneyu(consenso['Der_min'],␣
↪PruebasConteoAutomatico[col])
        estadistico_derfrec, pvalue_derfrec = mannwhitneyu(consenso['F.Frec.
↪Der'], ffrec[col])
        ConteosAuto_Der['Grupo'].append(col[0])
        ConteosAuto_Der['Extremidad'].append(col[1])
        ConteosAuto_Der['umbral_temporal'].append(f'{rounder(col[2])}')
        ConteosAuto_Der['umbral_amplitud'].append(f'{rounder(col[3])}')
        ConteosAuto_Der['Error RMSE conteo'].append(mae_der)
        ConteosAuto_Der['Error RMSE Frecuencia'].append(ffrecuenciader)
        ConteosAuto_Der['Lin_conteo'].append(Lin_conteoder)
```

```
        ConteosAuto_Der['Lin_frec'].append(Lin_frecder)
        ConteosAuto_Der['ClasLin_conteo'].append(clasLinconteoder)
        ConteosAuto_Der['ClasLin_frec'].append(claslinfrecder)
        ConteosAuto_Der['ErrorRel_conteo'].append(error_relativoder)
        ConteosAuto_Der['ErrorRel_ffrec'].append(error_relativoderfrec)
        ConteosAuto_Der['pValue_conteo'].append(pvalue_der)
        ConteosAuto_Der['pValue_ffrec'].append(pvalue_derfrec)
# Convertir los diccionarios en DataFrames
ConteosAuto_Izq_df = pd.DataFrame(ConteosAuto_Izq)
ConteosAuto_Der_df = pd.DataFrame(ConteosAuto_Der)
```

0.10 Gráficos de Resultados

0.10.1 Sección 7.3. Conteo automático de «Acciones Técnicas Dinámicas».

Subsección 7.3.1. No se halló una combinación de umbral_temporal y umbral_amplitud con el mejor desempeño en todos los indicadores

Figura 20: Variación del error RMSE del conteo automático Error RM SE conteo, en cada extremidad, al combinar 10 umbrales temporales y 5 umbrales de amplitud, en 50 pruebas por cada grupo de datos.

[ ]:
```
# Estilo de los gráficos
sns.set_context("talk", font_scale=1, rc={"lines.linewidth": 3.5})
# Crear el primer gráfico Extremidad Derecha
g1 = sns.relplot(
    data=ConteosAuto_Der_df, kind="line",
    x="umbral_temporal", y="Error RMSE conteo",
    hue="umbral_amplitud", col="Grupo", palette="Paired")
g1.set(ylim=(4, 31))
# Crear el segundo gráfico Extremidad Izquierda
g2 = sns.relplot(
    data=ConteosAuto_Izq_df, kind="line",
    x="umbral_temporal", y="Error RMSE conteo",
    hue="umbral_amplitud", col="Grupo", palette="Paired")
g2.set(ylim=(4, 31))
# Mostrar los gráficos
plt.show()
```

[ ]:
```
sns.set_context("talk", font_scale=1, rc={"lines.linewidth": 3.5}, )
# Crear el primer gráfico Extremidad Derecha
g1 = sns.relplot(
    data=ConteosAuto_Der_df, kind="line",
    x="umbral_temporal", y="Error RMSE Frecuencia",
    hue="umbral_amplitud", col="Grupo", palette="Paired")
g1.set(ylim=(0, 5.5))
# Crear el segundo gráfico Extremidad Izquierda
g2 = sns.relplot(
    data=ConteosAuto_Izq_df, kind="line",
```

```
    x="umbral_temporal", y="Error RMSE Frecuencia",
    hue="umbral_amplitud", col="Grupo", palette="Paired")
g2.set(ylim=(0, 5.5))
# Mostrar los gráficos
plt.show()
```

0.11 Definición de umbrales óptimos

0.12 Tablas de resultados finales

0.13 Sección Anexo 11.8.3. Resultados de las distintas combinaciones de umbral temporal y umbral amplitud.

0.13.1 Tabla 17: Resultados de las distintas combinaciones de umbral temporal y umbral amplitud. Ordenados de manera decreciente en base al coeficiente de concordancia de Lin de los conteos de «ATD» y del «FF».

[ ]:
```
pd.set_option('display.max_columns', None)
pd.set_option('display.max_row', None)
if len(consenso.iloc[:,0])==20:
    consenso = consenso.drop(3)
# Inicializar diccionarios para cada resultado
ConteosAuto_Izq = {'Grupo': [], 'Extremidad': [], 'umbral_temporal': [],␣
 ↪'umbral_amplitud': [],'Error RMSE conteo': [], 'Error RMSE Frecuencia':[],
                   'Lin_conteo':[],'ClasLin_conteo':[], 'Lin_frec':[],␣
 ↪'ClasLin_frec':[],'ErrorRel_conteo':[], 'ErrorRel_ffrec':[],'pValue_conteo':
 ↪[],'pValue_ffrec':[]}
ConteosAuto_Der = {'Grupo': [], 'Extremidad': [], 'umbral_temporal': [],␣
 ↪'umbral_amplitud': [], 'Error RMSE conteo': [], 'Error RMSE Frecuencia':[],
                   'Lin_conteo':[],'ClasLin_conteo':[], 'Lin_frec':[],␣
 ↪'ClasLin_frec':[],'ErrorRel_conteo':[], 'ErrorRel_ffrec':[],'pValue_conteo':
 ↪[],'pValue_ffrec':[]}

# Iterar a través de las columnas de PruebasConteoAutomatico
for col in PruebasConteoAutomatico.columns:
    #lado izquierdo
    if "Izq" in col:
        error_relativoizq =  (abs(consenso['Izq_min'] -␣
 ↪PruebasConteoAutomatico[col]) / consenso['Izq_min'] * 100).mean()
        stdizq =  (abs(consenso['Izq_min'] - PruebasConteoAutomatico[col]) /␣
 ↪consenso['Izq_min'] * 100).std()
        error_relativoizqfrec = ((abs(consenso['F.Frec.Izq'] - ffrec[col]) /␣
 ↪9)*100).mean()
        stdizqfrec = ((abs(consenso['F.Frec.Izq'] - ffrec[col]) / 9)*100).std()
        mae_izq, std_rmse_conteo_izq = rmse_y_std(consenso['Izq_min'],␣
 ↪PruebasConteoAutomatico[col])
        ffrecuenciaizq, std_frec_izq = rmse_y_std(consenso['F.Frec.Izq'],␣
 ↪ffrec[col])
```

```
        Lin_conteoizq = Lin_ccc(consenso['Izq_min'],␣
↪PruebasConteoAutomatico[col])
        Lin_frecizq = Lin_ccc(consenso['F.Frec.Izq'], ffrec[col])
        clasLinconteoizq= ClasificacionCoeficiente(Lin_conteoizq)
        claslinfrecizq=ClasificacionCoeficiente(Lin_frecizq)
        estadistico_izq, pvalue_izq = mannwhitneyu(consenso['Izq_min'],␣
↪PruebasConteoAutomatico[col])
        estadistico_izqfrec, pvalue_izqfrec = mannwhitneyu(consenso['F.Frec.
↪Izq'], ffrec[col])
        # Agregar los valores correspondientes a los diccionarios
        ConteosAuto_Izq['Grupo'].append(col[0])
        ConteosAuto_Izq['Extremidad'].append(col[1])
        ConteosAuto_Izq['umbral_temporal'].append(f'{rounder(col[2])}')
        ConteosAuto_Izq['umbral_amplitud'].append(f'{rounder(col[3])}')
        ConteosAuto_Izq['Error RMSE conteo'].append(f'{rounder(mae_izq)} ± {␣
↪rounder(std_rmse_conteo_izq)}')
        ConteosAuto_Izq['Error RMSE Frecuencia'].
↪append(f'{rounder(ffrecuenciaizq)} ± {rounder(std_frec_izq)}')
        ConteosAuto_Izq['Lin_conteo'].append(f'{rounder(Lin_conteoizq)}')
        ConteosAuto_Izq['Lin_frec'].append(f'{rounder(Lin_frecizq)}')
        ConteosAuto_Izq['ClasLin_conteo'].append(clasLinconteoizq)
        ConteosAuto_Izq['ClasLin_frec'].append(claslinfrecizq)
        ConteosAuto_Izq['ErrorRel_conteo'].
↪append(f'{rounder(error_relativoizq)} ± {rounder(stdizq)}')
        ConteosAuto_Izq['ErrorRel_ffrec'].
↪append(f'{rounder(error_relativoizqfrec)} ± {rounder(stdizqfrec)}')
        ConteosAuto_Izq['pValue_conteo'].append(f'{rounder(pvalue_izq)}')
        ConteosAuto_Izq['pValue_ffrec'].append(f'{rounder(pvalue_izqfrec)}')

    elif 'Der' in col:
        error_relativoder =  (abs(consenso['Der_min'] -␣
↪PruebasConteoAutomatico[col]) / consenso['Der_min'] * 100).mean()
        stdDer =  (abs(consenso['Der_min'] - PruebasConteoAutomatico[col]) /␣
↪consenso['Der_min'] * 100).std()
        error_relativoderfrec = ((abs(consenso['F.Frec.Der'] - ffrec[col]) /␣
↪9)*100).mean()
        stdderfrec = ((abs(consenso['F.Frec.Der'] - ffrec[col]) / 9)*100).std()
        mae_der, std_rmse_conteo_der = rmse_y_std(consenso['Der_min'],␣
↪PruebasConteoAutomatico[col])
        ffrecuenciader, std_frec_der = rmse_y_std(consenso['F.Frec.Der'],␣
↪ffrec[col])
        Lin_conteoder = Lin_ccc(consenso['Der_min'],␣
↪PruebasConteoAutomatico[col])
        Lin_frecder = Lin_ccc(consenso['F.Frec.Der'], ffrec[col])
        clasLinconteoder= ClasificacionCoeficiente(Lin_conteoder)
        claslinfrecder=ClasificacionCoeficiente(Lin_frecder)
```

```
        estadistico_der, pvalue_der = mannwhitneyu(consenso['Der_min'],␣
↪PruebasConteoAutomatico[col])
        estadistico_derfrec, pvalue_derfrec = mannwhitneyu(consenso['F.Frec.
↪Der'], ffrec[col])
        # Agregar los valores correspondientes a los diccionarios
        ConteosAuto_Der['Grupo'].append(col[0])
        ConteosAuto_Der['Extremidad'].append(col[1])
        ConteosAuto_Der['umbral_temporal'].append(f'{rounder(col[2])}')
        ConteosAuto_Der['umbral_amplitud'].append(f'{rounder(col[3])}')
        ConteosAuto_Der['Error RMSE conteo'].append(f'{rounder(mae_der)} ±␣
↪{rounder(std_rmse_conteo_der)}')
        ConteosAuto_Der['Error RMSE Frecuencia'].
↪append(f'{rounder(ffrecuenciader)} ± {rounder(std_frec_der)}')
        ConteosAuto_Der['Lin_conteo'].append(f'{rounder(Lin_conteoder)}')
        ConteosAuto_Der['Lin_frec'].append(f'{rounder(Lin_frecder)}')
        ConteosAuto_Der['ClasLin_conteo'].append(clasLinconteoder)
        ConteosAuto_Der['ClasLin_frec'].append(claslinfrecder)
        ConteosAuto_Der['ErrorRel_conteo'].
↪append(f'{rounder(error_relativoder)} ± {rounder(stdDer)}')
        ConteosAuto_Der['ErrorRel_ffrec'].
↪append(f'{rounder(error_relativoderfrec)} ± {rounder(stdderfrec)}')
        ConteosAuto_Der['pValue_conteo'].append(f'{rounder(pvalue_der)}')
        ConteosAuto_Der['pValue_ffrec'].append(f'{rounder(pvalue_derfrec)}')
# Convertir los diccionarios en DataFrames
ConteosAuto_Izq_df = pd.DataFrame(ConteosAuto_Izq)
ConteosAuto_Der_df = pd.DataFrame(ConteosAuto_Der)
stats_total = pd.concat([ConteosAuto_Der_df, ConteosAuto_Izq_df], axis=0)
```

[ ]:
```
#explorar los resultados de cada grupo de datos:
stats_total[['Grupo', 'Extremidad', 'umbral_temporal', 'umbral_amplitud',
       'Error RMSE conteo', 'Error RMSE Frecuencia', 'Lin_conteo',␣
↪'ClasLin_conteo',
       'Lin_frec', 'ClasLin_frec','ErrorRel_conteo',␣
↪'ErrorRel_ffrec','pValue_conteo', 'pValue_ffrec']].
↪sort_values(['Lin_conteo','Lin_frec'], ascending=False)
#stats_total[stats_total['Grupo']== 'Dataset']
#stats_total[stats_total['Grupo']== 'Dataset Suavizado']
#stats_total[stats_total['Grupo']== 'Mediapipe Suavizado']
#stats_total[stats_total['Grupo']== 'Mediapipe']
stats_total.head()
```

0.14 Umbrales Óptimos

0.15 Tabla 12c: Resultados de conteo automático y «conteo humano de consenso» utilizando las combinaciones óptimas de cada grupos de datos umbral_temporal = 0.5 y umbral_amplitud = 15

[ ]:
```
pd.set_option('display.float_format', '{:.1f}'.format)
orden = [('','n° video'), ('Dataset', 'Izq_min'), ('Dataset', 'F.Frec.Izq'),␣
 ↪('Dataset', 'Der_min'),('Dataset', 'F.Frec.Der'),
          ('Dataset Suavizado', 'Izq_min'), ('Dataset Suavizado', 'F.Frec.
 ↪Izq'), ('Dataset Suavizado', 'Der_min'), ('Dataset Suavizado', 'F.Frec.Der'),
          ('Mediapipe','Izq_min'), ('Mediapipe', 'F.Frec.Izq'), ('Mediapipe',␣
 ↪'Der_min'), ('Mediapipe', 'F.Frec.Der'), ('Mediapipe Suavizado', 'Izq_min'),
          ('Mediapipe Suavizado', 'F.Frec.Izq'), ('Mediapipe Suavizado',␣
 ↪'Der_min'), ('Mediapipe Suavizado', 'F.Frec.Der')]
umbrales_optimos = pd.concat([PruebasConteoAutomatico['Dataset']['Izq'][(1.0,␣
 ↪10.0)],ffrec['Dataset']['Izq'][(1.0, 10.0)],
                           PruebasConteoAutomatico['Dataset']['Der'][(0.7, 25.
 ↪0)],ffrec['Dataset']['Der'][(0.7, 25.0)],
                           PruebasConteoAutomatico['Dataset␣
 ↪Suavizado']['Izq'][(1.0, 10.0)],ffrec['Dataset Suavizado']['Izq'][(1.0, 10.
 ↪0)],
                           PruebasConteoAutomatico['Dataset␣
 ↪Suavizado']['Der'][(1.0, 15.0)],ffrec['Dataset Suavizado']['Der'][(1.0, 15.
 ↪0)],
                           PruebasConteoAutomatico['Mediapipe']['Izq'][(0.1,␣
 ↪25.0)],ffrec['Mediapipe']['Izq'][(0.1, 25.0)],
                           PruebasConteoAutomatico['Mediapipe']['Der'][(0.5,␣
 ↪25.0)],ffrec['Mediapipe']['Der'][(0.5, 25.0)],
                           PruebasConteoAutomatico['Mediapipe␣
 ↪Suavizado']['Izq'][(0.4, 15.0)],ffrec['Mediapipe Suavizado']['Izq'][(0.4, 15.
 ↪0)],
                           PruebasConteoAutomatico['Mediapipe␣
 ↪Suavizado']['Der'][(0.5, 15.0)],ffrec['Mediapipe Suavizado']['Der'][(0.5, 15.
 ↪0)]],
                           axis=1, keys=orden[1:])
umbrales_optimos = umbrales_optimos.reindex(index= indice)
umbrales_optimos[(  '',   'n° video')] = indice
umbrales_optimos = umbrales_optimos[orden]
umbrales_optimos
```

[ ]:
```
orden_1=["Izq_min", "F.Frec.Izq", "Der_min", "F.Frec.Der"]
diffDataset = consenso_copy - umbrales_optimos['Dataset']
diffDataset =diffDataset[orden_1]
diffDatasetS = consenso_copy - umbrales_optimos['Dataset Suavizado']
diffDatasetS = diffDatasetS[orden_1]
diffMediapipe = consenso_copy - umbrales_optimos['Mediapipe']
```

```
diffMediapipe = diffMediapipe[orden_1]
diffMediapipeS = consenso_copy - umbrales_optimos['Mediapipe Suavizado']
diffMediapipeS = diffMediapipeS[orden_1]

diff = pd.concat([diffDataset, diffDatasetS, diffMediapipe, diffMediapipeS],␣
 ↪axis = 1, keys=["Dataset", "Dataset Suavizado", "Mediapipe", "Mediapipe␣
 ↪Suavizado"])
diff[(  '',   'n° video')] = indice
Mp_SuavizadoDiff_abs = diff[[col for col in diff.columns if 'Mediapipe␣
 ↪Suavizado' in col[0]]].abs().sort_values(by=('Mediapipe Suavizado',   ␣
 ↪'Der_min'), ascending=True)
```

[ ]:
```
OrdenDecrecienteError = [12,14,3,17,18,15,16,10,5,6,2,4,19,7,1,11,9,20,8,13]

Mediapipe_Suavizado_differencia = diff[[col for col in diff.columns if␣
 ↪'Mediapipe Suavizado' in col[0]]].sort_values(by=('Mediapipe Suavizado',   ␣
 ↪'Der_min'), ascending=True).reindex(OrdenDecrecienteError)
Mediapipe_Suavizado_differencia[(  '',   'n° video')] = indice
Mediapipe_Suavizado_differencia[[('', 'n° video'), ('Mediapipe␣
 ↪Suavizado','Izq_min'), ('Mediapipe Suavizado', 'F.Frec.Izq'), ('Mediapipe␣
 ↪Suavizado','Der_min'), ('Mediapipe Suavizado', 'F.Frec.Der')]]
```

11.5. Anexo: (Código Python) FuncionesPipeline.py

El archivo de código llamado $FuncionesPipeline.py$ contiene las principales funciones que se importan desde el pipeline principal de Procesamiento en Python.

FuncionesPipeline.py

```python
import numpy as np
def PrepareAndLoadDataset(i, side = "derecho", Smooth = False, window_size=3, datasetdata=False):
    """
    Prepara y carga el grupo de datos del Dataset de un video.
    Parámetros:
    i (int): Número del video que se procesará.
    side (str, opcional): Lado del cuerpo a considerar para el análisis ("izquierdo" o "derecho").
    Smooth (bool, opcional): Si se debe suavizar la señal.
    window_size (int, opcional): Tamaño de ventana para el suavizado de la señal.
    Retorna:
    Ldatads_smooth (pd.DataFrame): DataFrame que contiene ángulos procesados y/o suavizados, según parámetro.
    technical_actions_ds_smooth (dict): Diccionario iniciado en cero para el conteo de acciones técnicas en el paso siguiente.
    """
    import numpy as np
    import h5py
    import pandas as pd
    #Para correr el archivo en Google Colab
    """import sys
    path_to_module = '/content/mnt/MyDrive/Thesis_Github/'
    sys.path.append(path_to_module)
    import io
    mat = h5py.File("/content/mnt/MyDrive/Thesis_Github/JointPositions/"+str(i+1)+'.mat', 'r')
    """
    #Carga los cuadros donde Dataset da la espalda a la Cámara.
    pathbackframes = '/home/diego/Documents/AT_Counting/Saved_files/thesis_txt_mediapipe/'+str(i+1)+'back_frames.csv'
    back_frames_df = pd.read_csv(pathbackframes, sep=',', header=None)
    back_frames_biglist =back_frames_df.values.tolist() #aparece con un [] demás
    back_frames = [value for sublist in back_frames_biglist for value in sublist]
    #abre el archivo mat usando librería h5py; bodylogger3d contiene Postura 3D
    mat = h5py.File("JointPositions/"+str(i+1)+'.mat', 'r')

    list(mat.keys())
    # Smooth permite suavizar la señal, acorde al parámetro window_size (refernte al n° de cuadros a promediar)
    if Smooth == True:
        bodylogger3d=SmoothDataset(mat['bodylogger3D'][:],window_size) # type: ignore
    if Smooth == False:
        bodylogger3d=mat['bodylogger3D'][:] # type: ignore
    img=np.zeros(5)
    angle_ds_imported = []
    timelogger =[]

    #timelogger es utilizado para imprimir la señal respecto al tiempo basado en video.
    for k in range(len(bodylogger3d)): # type: ignore
        timelogger.append(mat['videotimelogger'][k][0]) # type: ignore
    for frame in range(len(bodylogger3d)): # type: ignore
        #BPAfront
        if frame in back_frames:
            img2, angles = BPABack(img, bodylogger3d[frame], 0.4, draw=False) # type: ignore
        else:
            img2, angles = BPAfront(img, bodylogger3d[frame], 0.4, draw=False) # type: ignore
        angle_ds_imported.append(angles)

    if side.lower() == "izquierdo":
        sagital_izquierdo_angle = []
        frontal_izquierdo_angle = []
        transverse_izquierdo_angle = []
        izquierdo_codo_flexion = []
        izquierdo_muneca_flexion = []
        for l in range(len(angle_ds_imported)):
            sagital_izquierdo_angle.append(angle_ds_imported[l][5])
            frontal_izquierdo_angle.append(angle_ds_imported[l][6])
            transverse_izquierdo_angle.append(angle_ds_imported[l][7])
            izquierdo_codo_flexion.append(angle_ds_imported[l][8])
            izquierdo_muneca_flexion.append(angle_ds_imported[l][9])
        Ldatads_smooth = pd.DataFrame({'flexion_hombro_izquierdo': sagital_izquierdo_angle,
                                       'abduccion_hombro_izquierdo': frontal_izquierdo_angle,
                                       'rotacion_hombro_izquierdo': transverse_izquierdo_angle,
                                       'flexion_codo_izquierdo':izquierdo_codo_flexion,
```

```
                                    'flexion_muneca_izquierda':izquierdo_muneca_flexion})
        Ldatads_smooth['time'] = timelogger
        technical_actions_ds_smooth = {'flexion_hombro_izquierdo': 0,
                                    'abduccion_hombro_izquierdo': 0,
                                    'rotacion_hombro_izquierdo': 0,
                                    'flexion_codo_izquierdo': 0,
                                    'flexion_muneca_izquierda': 0}
    elif side.lower() == "derecho":
        sagital_izquierdo_angle = []
        frontal_izquierdo_angle = []
        transverse_izquierdo_angle = []
        izquierdo_codo_flexion = []
        izquierdo_muneca_flexion = []
        for l in range(len(angle_ds_imported)):
            sagital_izquierdo_angle.append(angle_ds_imported[l][0])
            frontal_izquierdo_angle.append(angle_ds_imported[l][1])
            transverse_izquierdo_angle.append(angle_ds_imported[l][2])
            izquierdo_codo_flexion.append(angle_ds_imported[l][3])
            izquierdo_muneca_flexion.append(angle_ds_imported[l][4])

        Ldatads_smooth = pd.DataFrame({'flexion_hombro_derecho': sagital_izquierdo_angle,
                                    'abduccion_hombro_derecho': frontal_izquierdo_angle,
                                    'rotacion_hombro_derecho': transverse_izquierdo_angle,
                                    'flexion_codo_derecho':izquierdo_codo_flexion,
                                    'flexion_muneca_derecha':izquierdo_muneca_flexion})
        Ldatads_smooth['time'] = timelogger
        technical_actions_ds_smooth = {'flexion_hombro_derecho': 0,
                                    'abduccion_hombro_derecho': 0,
                                    'rotacion_hombro_derecho': 0,
                                    'flexion_codo_derecho': 0,
                                    'flexion_muneca_derecha': 0}
    else:
        print(f"Must specify side as either izquierdo or derecho")
        Ldatads_smooth = None
        technical_actions_ds_smooth = None
    return Ldatads_smooth, technical_actions_ds_smooth

def PrepareAndLoadMediapipe(i, side="izquierdo", Smooth=False, window_size=3):
    """
    Prepara y carga datos obtenidos de la detección de postura mediante Mediapipe para su análisis.
    Parámetros:
    i (int): Número del video que se procesará.
    side (str, opcional): Lado del cuerpo a considerar para el análisis ("izquierdo" o "derecho").
    Smooth (bool, opcional): Si se debe suavizar la señal.
    window_size (int, opcional): Tamaño de ventana para el suavizado de la señal.
    Retorna:
    Ldatamp_smooth (pd.DataFrame): DataFrame con ángulos procesados y suavizados (opcional) obtenidos de Mediapipe.
    technical_actions_mp_smooth (dict): Diccionario con valores técnicos relacionados con los ángulos.
    """
    import h5py
    import numpy as np
    import pandas as pd
    #Se carga información de Dataset, para utilizar videotime y bodytime
    mat = h5py.File("JointPositions/" + str(i + 1) + '.mat', 'r')
    video = str(i + 1)
    posture_mediapipe_imported = importDataMediapipe(video)

    if Smooth == True:
        posture_mediapipe_imported = SmoothMediapipe(posture_mediapipe_imported, window_size)
    img = np.zeros(5)
    angles_mediapipe = []
    timelogger = []
    bodysmoothed = []
    angle_mp_imported = []
    #Desfase entre video y dataset declarado en propio Dataset
    videotime =mat['videotimelogger'][:] # type: ignore
    bodytime =mat['bodytimelogger'][:] # type: ignore

    desfase_dict = {1:4,2:3,3:3,4:4,5:2,6:3,7:3,8:3,9:3,10:3,11:4,12:3,13:2,14:4,15:3,16:3,17:3,18:3,19:4,20:2}
    number = desfase_dict[i+1]
    #PERMITE INICIAR el video desde el frame "start"
    start = len(videotime)-len(bodytime) + number # type: ignore
    #PERMITE INICIAR el video desde el frame "start"
```

```
    start = len(videotime)-len(bodytime) + number # type: ignore
    if start>0:
        finish= start
        posture_mediapipe_imported=posture_mediapipe_imported[start:]  # type: ignore
    else:
        #permite finalizar mediapipe en frame # finish
        finish = -(len(posture_mediapipe_imported)) # type: ignore

    for k in range(min(len(posture_mediapipe_imported), len(mat['bodylogger3D'][:]))): # type: ignore
        img, angles = BodyPlanesAngles(img, posture_mediapipe_imported[k])  # type: ignore
        angle_mp_imported.append(angles)
        timelogger.append(mat['videotimelogger'][k][0]) # type: ignore

    if side.lower() == "izquierdo":
        sagital_izquierdo_angle = []
        frontal_izquierdo_angle = []
        transverse_izquierdo_angle = []
        izquierdo_codo_flexion = []
        izquierdo_muneca_flexion = []
        for l in range(len(angle_mp_imported)):
            sagital_izquierdo_angle.append(angle_mp_imported[l][5])
            frontal_izquierdo_angle.append(angle_mp_imported[l][6])
            transverse_izquierdo_angle.append(angle_mp_imported[l][7])
            izquierdo_codo_flexion.append(angle_mp_imported[l][8])
            izquierdo_muneca_flexion.append(angle_mp_imported[l][9])
        Ldatamp_smooth = pd.DataFrame({'flexion_hombro_izquierdo': sagital_izquierdo_angle[:-finish],
                                       'abduccion_hombro_izquierdo': frontal_izquierdo_angle[:-finish],
                                       'rotacion_hombro_izquierdo': transverse_izquierdo_angle[:-finish],
                                       'flexion_codo_izquierdo':izquierdo_codo_flexion[:-finish],
                                       'flexion_muneca_izquierda':izquierdo_muneca_flexion[:-finish]})
        Ldatamp_smooth['time'] = timelogger[:-finish]
        technical_actions_mp_smooth = {'flexion_hombro_izquierdo': 0,
                                       'abduccion_hombro_izquierdo': 0,
                                       'rotacion_hombro_izquierdo': 0,
                                       'flexion_codo_izquierdo': 0,
                                       'flexion_muneca_izquierda': 0}
    elif side.lower() == "derecho":
        sagital_izquierdo_angle = []
        frontal_izquierdo_angle = []
        transverse_izquierdo_angle = []
        izquierdo_codo_flexion = []
        izquierdo_muneca_flexion = []
        for l in range(len(angle_mp_imported)):
            sagital_izquierdo_angle.append(angle_mp_imported[l][0])
            frontal_izquierdo_angle.append(angle_mp_imported[l][1])
            transverse_izquierdo_angle.append(angle_mp_imported[l][2])
            izquierdo_codo_flexion.append(angle_mp_imported[l][3])
            izquierdo_muneca_flexion.append(angle_mp_imported[l][4])
        Ldatamp_smooth = pd.DataFrame({'flexion_hombro_derecho': sagital_izquierdo_angle[:-finish],
                                       'abduccion_hombro_derecho': frontal_izquierdo_angle[:-finish],
                                       'rotacion_hombro_derecho': transverse_izquierdo_angle[:-finish],
                                       'flexion_codo_derecho':izquierdo_codo_flexion[:-finish],
                                       'flexion_muneca_derecha':izquierdo_muneca_flexion[:-finish]})
        Ldatamp_smooth['time'] = timelogger[:-finish]
        technical_actions_mp_smooth = {'flexion_hombro_derecho': 0,
                                       'abduccion_hombro_derecho': 0,
                                       'rotacion_hombro_derecho': 0,
                                       'flexion_codo_derecho': 0,
                                       'flexion_muneca_derecha': 0}
    else:
        print(f"Must specify side as either izquierdo or derecho")
        Ldatamp_smooth = None
        technical_actions_mp_smooth =  None
    return Ldatamp_smooth, technical_actions_mp_smooth

def PrintAnglesOnBigLoop(Ldatads, col, i, max_peaks, min_peaks, valid_peak, x, window_size=None, umbral_amplitud=
None, umbral_temporal=None):
    import matplotlib.pyplot as plt
    import plotly.express as px
    import plotly.io as pio
    import plotly.graph_objects as go
    if 'time' in col:
        pass
```

```
    else:
        fig = go.Figure()
        if window_size:
            fig.update_layout(title=f'Video {i+1} {col} {x} suavizado en {window_size} cuadros, umbral amplitud: {umbral_amplitud}, umbral temporal: {umbral_temporal*1000 :.0f} ms', title_font=dict(size=20)) #type:ignore
        else:
            fig.update_layout(title=f'Video {i+1} {x}')
        fig.add_trace(go.Scatter(
            y=Ldatads[col],
            mode='lines+markers',
            name=col,
            x=Ldatads['time'],
            line=dict(color='lawngreen')))
        fig.add_trace(go.Scatter(
            x=Ldatads['time'][max_peaks],
            y=Ldatads[col][max_peaks],
            mode='markers',
            marker=dict(
                size=8,
                color='red',
                symbol='cross'
            ),
            name='Max Peaks'))
        fig.add_trace(go.Scatter(
            x=Ldatads['time'][min_peaks],
            y=Ldatads[col][min_peaks],
            mode='markers',
            marker=dict(
                size=10,
                color='blue',
                symbol='cross'
            ),
            name='Min Peaks'))
        fig.add_trace(go.Scatter(
            x=Ldatads['time'][valid_peak],
            y=Ldatads[col][valid_peak],
            mode='markers',
            marker=dict(
                size=18,
                color='yellow',
                symbol='cross'
            ),
            name='Valid Peaks'))
        max_time = max(Ldatads['time'])
        fig.update_xaxes(range=[0, max_time])
        fig.update_xaxes(title_text='Tiempo (s)', title_font=dict(size=20))
        fig.update_yaxes(title_text='Grados (°)', title_font=dict(size=20))
        fig.show()

def DynamicTechnicalActionsCounter(LoadingFunction, n_VideosToAnalize, umbral_temporal, umbral_amplitud, dataset_type, side, Smooth=False, window_size=None, print_func=False):
    """
    Realiza el conteo de acciones técnicas dinámicas de la extremidad seleccionada basado en datos de Dataset o Mediapipe procesados y/o suavizados y genera resultados de análisis.
    Parámetros:
    LoadingFunction (función): Función que carga y procesa los datos para el análisis Dataset o Mediapipe.
    n_VideosToAnalize (int): Número de videos a analizar.
    umbral_temporal (float): Distancia temporal mínima, en segundos, entre sucesivos máximos.
    umbral_amplitud (float): Distancia de amplitud entre cada máximo y el sucesivo mínimo.
    dataset_type (str): Tipo de conjunto de datos utilizado para el análisis "Dataset" o "Mediapipe".
    side (str): Estremidad a evaluar "izquierda" o "derecha"
    Smooth (bool, opcional): Si se debe suavizar la señal.
    window_size (int, opcional): Tamaño de ventana para el suavizado de la señal.
    print_func (bool): Indica si se deben imprimir resultados durante el análisis.
    Retorna:
    ConteoList (list): Lista de conteos promedio de acciones técnicas por video.
    ResultsList (list): Lista de resultados de ángulos procesados por video.
    """
    import pandas as pd
    from scipy.signal import find_peaks
    ConteoList = []
    ResultsList = []
    # Recorre cada video para realizar el análisis
    for i in range(n_VideosToAnalize):
```

```
        # Carga y procesa los datos del video utilizando la función de dataset o mediapipe
        data, technical_actions = LoadingFunction(i, side=side, Smooth=Smooth, window_size=window_size)
        # Recorre cada columna (ángulo) en los datos
        for col in data.columns:
            if col == 'time':
                pass
            # Encuentra los máximos y mínimos de la señal
            max_peaks, _ = find_peaks(data[col], prominence=10, distance=4)
            min_peaks, _ = find_peaks(-data[col], prominence=10, distance=4)

            valid_peak = np.empty(0)
            for f in range(min(len(min_peaks), len(max_peaks))):
                """ Comprueba si la diferencia en amplitud entre los max(i) y min(i) cumple con el umbral_amplitud
                y si la distancia temporal entre max(i) y max(i-1) cumple con el umbral_temporal.
                Si el peak es válido, se almacena la posición temporal y la amplitud
                y se incrementa el contador de la acción técnica correspondiente"""
                if (data[col][max_peaks[f]] - data[col][min_peaks[f]] > umbral_amplitud) and (data['time'][max_peaks[f]] - data['time'][max_peaks[f - 1]] > umbral_temporal):#type:ignore
                    valid_peak = np.append(valid_peak, max_peaks[f])
                    technical_actions[col] += 1
            ResultsList.append({f"video {str(i+1)} {col}": data[col]})
            if print_func:
                PrintAnglesOnBigLoop(data, col, i, max_peaks, min_peaks, valid_peak, dataset_type, window_size, umbral_amplitud, umbral_temporal)
        ConteoList.append(sum(technical_actions[key] for key in technical_actions) // len(technical_actions))
    return ConteoList, ResultsList

def filtrar_frecuencias(signal, frecuencia_muestreo, frecuencia_corte):
    """Filtro pasa bajo, utilizando la transformada de Fourier, llevando a cero los componentes de frecuencia por sobre la frecuencia de corte
    Parámetros:
    signal (numpy.ndarray): señal a filtrar.
    frecuencia_muestreo (int): frecuencia de muestreo de la señal.
    frecuencia_corte (int): frecuencia de corte para filtrar la señal.
    Retorna:
    numpy.ndarray: La señal filtrada.
    """
    # Paso 1: Realiza la Transformada de Fourier
    espectro = np.fft.fft(signal)
    # Paso 2: Identifica las frecuencias a mantener y establece en cero las demás
    num_puntos = len(signal)
    frecuencias = np.fft.fftfreq(num_puntos, 1 / frecuencia_muestreo)
    # Establece en cero las frecuencias mayores que la frecuencia de corte
    espectro[np.abs(frecuencias) > frecuencia_corte] = 0
    # Paso 3: Aplica la Transformada Inversa de Fourier
    signal_filtrada = np.fft.ifft(espectro)
    return np.real(signal_filtrada)

video_fps_dict = {1:8,2:10,3:8,4:11,5:11,6:12,7:12,8:12,9:12,10:12,11:12,12:12,13:12,14:12,15:12,16:12,17:8,18:8,19:8,20:8}

def FuncionFFrecuencia(value):
    if value  < 22.5:
        return 0.0
    elif value >= 22.5 and value < 27.5:
        return 0.5
    elif value >= 27.5 and value < 32.5:
        return 1
    elif value >= 32.5 and value < 37.5:
        return 2
    elif value >= 37.5 and value < 42.5:
        return 3
    elif value >= 42.5 and value < 47.5:
        return 4
    elif value >= 47.5 and value < 52.5:
        return 5
    elif value >= 52.5 and value < 57.5:
        return 6
    elif value >= 57.5 and value < 62.5:
        return 7
    elif value >= 62.5 and value < 67.5:
        return 8
    elif value >= 67.5 and value < 72.5:
        return 9
```

```
    elif value >= 72.5:
        return 9

def rounder(a):
    if isinstance(a, int):
        return int(a)
    else:
        return np.round(a,3)

def rmse_y_std(y_true, y_pred):
    """
    Calcula el RMSE (Root Mean Squared Error) y la desviación estándar de la media
    de los errores entre y_true y y_pred.

    Parametros:
    y_true (array-like): Valores reales u observados.
    y_pred (array-like): Valores predichos o estimados.

    Retorna:
    rmse (float): RMSE entre y_true y y_pred.
    std_of_mean (float): Desviación estándar de la media de los errores.
    """
    errores = np.sqrt((y_true - y_pred) ** 2)
    rmse = np.sqrt(np.mean(errores ** 2)).mean()
    std = np.std(errores)
    return rmse, std

def caderaCentered(bodylogger3d):
    for k in range(len(bodylogger3d)): # type: ignore
        for landmark in range(len(bodylogger3d[k])):

            bodylogger3d[k][landmark][0],bodylogger3d[k][landmark][1], bodylogger3d[k][landmark][2] = bodylogger3d[k][landmark][0] - (bodylogger3d[k][12][0] + bodylogger3d[k][16][0])/2, bodylogger3d[k][landmark][1] - (bodylogger3d[k][12][1] + bodylogger3d[k][16][1])/2, bodylogger3d[k][landmark][2] - (bodylogger3d[k][12][2] + bodylogger3d[k][16][2])/2
    return bodylogger3d

def SmoothDataset(body_logger_3d, window_size):
    """
    Suaviza los datos de un conjunto de coordenadas 3D a lo largo de una ventana deslizante.
    Entradas:
    - body_logger_3d (numpy.ndarray): Matriz tridimensional de coordenadas 3D.
      Cada dimensión representa [frames, participantes, landmarks].
    - window_size (int): Tamaño de la ventana deslizante para el suavizado.
    Salida:
    - smoothed_data (numpy.ndarray): Matriz suavizada de coordenadas 3D.
      Cada dimensión representa [frames_suavizados, participantes, landmarks].
    """
    total_frames = len(body_logger_3d)
    landmarks = len(body_logger_3d[0][0])
    if total_frames < window_size:
        raise ValueError("El tamaño de la ventana no puede ser mayor que el número de fotogramas.")
    smoothed_data = np.zeros((total_frames - window_size + 1, len(body_logger_3d[0]), landmarks))
    for i in range(total_frames - window_size + 1):
        for j in range(len(body_logger_3d[0])):
            for k in range(landmarks):
                coord_data = body_logger_3d[i:i+window_size, j, k]
                smoothed_data[i][j][k] = np.mean(coord_data)
    return smoothed_data

def SmoothMediapipe(posture, window_size):
    """
    Suaviza los datos de postura detectados por el modelo Mediapipe a lo largo de una ventana deslizante.
    Entradas:
    - posture (list): Lista de datos de postura detectados por el modelo Mediapipe.
      Cada elemento de la lista representa una secuencia de fotogramas y contiene una matriz 3D (33 joints x 5 coordenadas).
    - window_size (int): Tamaño de la ventana deslizante para el suavizado.
    Salida:
    - posture_mediape_imported (list): Lista de datos de postura suavizados.
      Cada elemento de la lista contiene una secuencia de fotogramas suavizados, con una matriz 3D (33 joints x 5 coordenadas).
    """
    posture_mediape_imported = []
```

```
    for N_frames in range(len(posture)):
        if window_size <3:
            posture_mediape_imported.append(posture[N_frames])
        else:
            smoothed_posture = []
            for i in range(33):
                smoothed_joint = []
                for j in range(5):
                    smoothed_joint.append(sum([posture[k][i][j] for k in range(N_frames-window_size+1, N_frames+1)]
) // window_size)
                smoothed_posture.append(smoothed_joint)
            posture_mediape_imported.append(smoothed_posture)
    return posture_mediape_imported

def Lin_ccc(x, y):
    '''
    Calcula el Coeficiente de Concordancia de Lin (CCC).
    Parámetros:
    x (array-like): El primer conjunto de mediciones.
    y (array-like): El segundo conjunto de mediciones.
    Retorna:
    float: El Coeficiente de Concordancia de Lin (CCC) entre x e y.
    El CCC es una medida de la concordancia entre dos conjuntos de mediciones, x e y.
    Cuantifica el grado en que están de acuerdo y proporciona un valor entre -1 y 1,
    donde 1 indica una concordancia perfecta, 0 indica ninguna concordancia y -1 indica
    una discordancia perfecta.
    Extraído de https://nirpyresearch.com/concordance-correlation-coefficient/
    '''
    sxy = np.sum((x - x.mean()) * (y - y.mean())) / x.shape[0]
    lincc = 2 * sxy / (np.var(x) + np.var(y) + (x.mean() - y.mean())**2)
    return lincc

def ClasificacionCoeficiente(r):
    """
    Clasifica un coeficiente de correlación en una categoría de acuerdo a su valor.
    Parámetros:
    r (float): Coeficiente de correlación a clasificar.
    Retorna:
    str: Categoría que describe el coeficiente de correlación.
    """
    import pandas as pd
    if isinstance(r, pd.Series):
        # Para una Serie de Pandas, aplicar la función a cada elemento y devolver una Serie de resultados
        return r.apply(ClasificacionCoeficiente) # type: ignore
    else:
        if r >= 0.8:
            return "Alta (+)"
        elif r >= 0.4:
            return "Moderada (+)"
        elif r >= 0.2:
            return "Débil (+)"
        elif r >= 0:
            return "Muy Débil (+)"
        elif r >= -0.2:
            return "Muy Débil (-)"
        elif r >= -0.4:
            return "Débil (-)"
        elif r >= -0.8:
            return "Moderada (-)"
        else:
            return "Alta (-)"

def ProjectOnPlane(vector, plane_normal):
    sqr_mag = np.dot(plane_normal, plane_normal)
    if sqr_mag < np.finfo(float).eps:
        return vector
    dot = np.dot(vector, plane_normal)
    return vector - plane_normal * dot / sqr_mag

def Angle(vec1, vec2):
    vec1 = np.reshape(vec1, vec2.shape)
    denominator = np.sqrt(np.dot(vec1, vec1) * np.dot(vec2, vec2))
    if denominator < np.finfo(float).eps:
```

```
        return 0
    dot = np.clip(np.dot(vec1, vec2) / denominator, -1, 1)
    return np.arccos(dot) * 180 / np.pi

def SignedAngle(vec1, vec2, axis):
    unsigned_angle = Angle(vec1, vec2)
    cross = np.cross(vec1, vec2)
    sign = np.sign(np.dot(axis, cross))
    return unsigned_angle * sign

def importDataMediapipe(video):
    import pandas as pd
    path_read = 'Saved_files/thesis_txt_mediapipe/'
    file_path = path_read + video + '.csv'
    try:
        epbcv = pd.read_csv(file_path, lineterminator="\n", sep=',')
    except FileNotFoundError:
        print(f"File not found: {file_path}")
        return None
    epbcv.columns = [str(i) for i in range(33)]
    epbcv.index = range(len(epbcv))  # type: ignore
    posture_mediape_imported = []
    for N_frames in range(len(epbcv)):
        posture_mediape_imported.append(clean_imported(epbcv.iloc[N_frames]))
    return posture_mediape_imported

def clean_imported(text):
    """Removes unwanted characters from a list of strings and converts it to a formatted list of integers and floats."""
    cleaned_list = []
    for item in text:
        values = item.replace('[', '').replace(']', '').replace("'", '').split(',')
        if len(values) == 5:
            values = [int(values[0]), int(values[1]), int(values[2]), int(values[3]), float(values[4])]
            cleaned_list.append(values)
        else:
            print(f"Invalid input format: {item}")
    return cleaned_list

def BodyPlanesAngles(img, lmList, transparency=0.5, draw=False):
        """Calcula ángulos de los planos corporales a partir de las coordenadas de la postura.
        Parámetros:
        img (array): Imagen o video, para uso en tiempo real sobre videos o cámara web.
        lmList (list): Lista de coordenadas de puntos de referencia (landmarks) de la postura.
        transparency (float, opcional): Nivel de transparencia para la visualización (0-1) en videos.
        draw (bool, opcional): Si se deben dibujar los resultados en la imagen, para el uso en videos.
        Retorna:
        img (array): Imagen para su uso en videos o cálculo en tiempo real.
        angles (list): Lista de ángulos calculados en el siguiente orden:
        [sagital_derecho, frontal_derecho, transverso_derecho,
        derecho_codo_flexion, derecho_muneca_flexion,
        sagital_izquierdo, frontal_izquierdo, transverso_izquierdo,
        izquierdo_codo_flexion, izquierdo_muneca_flexion]   """
        import numpy as np
        #NOW 3 STEPS
        #  STEP NUMBER 1 : SET JOINT LANDMARKS COORDINATES POSITION.
        if len(lmList) !=0:
            #first define points then vectors by substraction between them.
            izquierdo_cadera_x, izquierdo_cadera_y, izquierdo_cadera_z = lmList[23][1:4]
            derecho_cadera_x, derecho_cadera_y, derecho_cadera_z = lmList[24][1:4]
            izquierdo_cadera = np.array([izquierdo_cadera_x, izquierdo_cadera_y, izquierdo_cadera_z])
            derecho_cadera = np.array([derecho_cadera_x, derecho_cadera_y, derecho_cadera_z])
            middcaderas = np.array([(izquierdo_cadera_x+derecho_cadera_x)/2, (izquierdo_cadera_y+derecho_cadera_y)/2, (izquierdo_cadera_z+derecho_cadera_z)/2])

            #izquierdohombro and derechohombro
            izquierdo_hombro_x, izquierdo_hombro_y, izquierdo_hombro_z = lmList[11][1:4]
            derecho_hombro_x, derecho_hombro_y, derecho_hombro_z = lmList[12][1:4]
            derecho_hombro = np.array([derecho_hombro_x, derecho_hombro_y, derecho_hombro_z])
            izquierdo_hombro = np.array([izquierdo_hombro_x, izquierdo_hombro_y, izquierdo_hombro_z])
            #middle point between izquierdo and derecho hombro
            middhombros = np.array([(izquierdo_hombro_x+derecho_hombro_x)/2, (izquierdo_hombro_y+derecho_hombro_y)/2, (izquierdo_hombro_z+derecho_hombro_z)/2])
            #codos munecas and tip of izquierdo and derecho hand
```

```
            izquierdo_codo_x, izquierdo_codo_y, izquierdo_codo_z = lmList[13][1:4]
            derecho_codo_x, derecho_codo_y, derecho_codo_z = lmList[14][1:4]
            derecho_codo = np.array([derecho_codo_x, derecho_codo_y, derecho_codo_z])
            izquierdo_codo = np.array([izquierdo_codo_x, izquierdo_codo_y, izquierdo_codo_z])
            izquierdo_muneca_x, izquierdo_muneca_y, izquierdo_muneca_z = lmList[15][1:4]
            derecho_muneca_x, derecho_muneca_y, derecho_muneca_z = lmList[16][1:4]
            derecho_muneca = np.array([derecho_muneca_x, derecho_muneca_y, derecho_muneca_z])
            izquierdo_muneca = np.array([izquierdo_muneca_x, izquierdo_muneca_y, izquierdo_muneca_z])
            lmiddfingers = np.array([(lmList[19][1] + lmList[17][1])/2,(lmList[19][2] + lmList[17][2])/2,
(lmList[19][3] + lmList[17][3])/2])
            rmiddfingers = np.array([(lmList[18][1] + lmList[20][1])/2,(lmList[18][2] + lmList[20][2])/2,
(lmList[18][3] + lmList[20][3])/2])

        # STEP NUMBER 2 : DEFINES JOINT LOCAL AXIS COORDINATE SYSTEM  BY USING NP.SUBSTRACT FOR CALCULARIONS
            #middcadera
            vx_middcadera = np.subtract(izquierdo_cadera, middcaderas)
            vy_middcadera = np.subtract(middhombros, middcaderas)
            vz_middcadera = np.cross(vy_middcadera, vx_middcadera)
            #middhombros
            vectorx_middhombros = np.subtract(izquierdo_hombro, middhombros)
            vectory_middhombros = -np.subtract(middcaderas,middhombros) #the same as vy_middcadera but starting
from hombro position and points up
            vectorz_middhombros = np.cross(vectorx_middhombros,vectory_middhombros)
            # izquierdo_
            vector_izquierdo_arm = np.subtract(izquierdo_codo, izquierdo_hombro)
            vector_izquierdo_forearm = np.subtract(izquierdo_muneca, izquierdo_codo)
            vector_izquierdo_codoarm = - np.subtract(izquierdo_hombro,izquierdo_codo)
            # izquierdomuneca  # first we have to create the middfingers coordinates
            vector_izquierdo_hand = np.subtract(lmiddfingers, izquierdo_muneca)
            vector_izquierdo_forearmmuneca = - np.subtract(izquierdo_codo, izquierdo_muneca)

            # derecho SIDE
            # derechohombro
            vector_derecho_arm = np.subtract(derecho_codo,derecho_hombro)
            # derechocodo
            vector_derecho_forearm = np.subtract(derecho_muneca , derecho_codo)
            vector_derecho_codoarm = - np.subtract(derecho_hombro,derecho_codo)
            # derechomuneca  # first we have to create the middfingers coordinates
            vector_derecho_hand = np.subtract(rmiddfingers , derecho_muneca)
            vector_derecho_forearmmuneca = - np.subtract(derecho_codo, derecho_muneca)

        # STEP NUMBER 3: ANGLE CALCULATIONS BETWEEN BONE VECTORS AND JOINT COORDINATES AXIS (FOR hombroS) OR
BETWWEN BONES (FOR codoS AND muneca).
          # ANGLES izquierdo SIDE
            #angles L hombro
            projectedvector_sagital_lhombro = ProjectOnPlane(vector_izquierdo_arm, vectorx_middhombros)
            sagital_izquierdo_angle = abs(SignedAngle(projectedvector_sagital_lhombro,-vectory_middhombros,
vectorx_middhombros))
            projectedvector_frontal_lhombro = ProjectOnPlane(vector_izquierdo_arm, vectorz_middhombros)
            frontal_izquierdo_angle = abs(SignedAngle(projectedvector_frontal_lhombro,-vectory_middhombros,
vectorz_middhombros))
            projectedvector_transverse_lhombro = ProjectOnPlane(vector_izquierdo_arm, -vectory_middhombros)
            transverse_izquierdo_angle = SignedAngle(projectedvector_transverse_lhombro,vectorz_middhombros,-
vectory_middhombros)
            # codo and muneca
            izquierdo_codo_flexion = Angle(vector_izquierdo_forearm,vector_izquierdo_codoarm)
            izquierdo_muneca_flexion = Angle(vector_izquierdo_hand,vector_izquierdo_forearmmuneca)
        # derecho SIDE
            # hombro
            projectedvector_sagital_rhombro = ProjectOnPlane(vector_derecho_arm, vectorx_middhombros)
            sagital_derecho_angle = abs(SignedAngle(projectedvector_sagital_rhombro,-vectory_middhombros,
vectorx_middhombros))
            projectedvector_frontal_rhombro = ProjectOnPlane(vector_derecho_arm, vectorz_middhombros)
            frontal_derecho_angle = abs(SignedAngle(projectedvector_frontal_rhombro,-vectory_middhombros,
vectorz_middhombros))
            projectedvector_transverse_rhombro = ProjectOnPlane(vector_derecho_arm, -vectory_middhombros)
            transverse_derecho_angle = SignedAngle(projectedvector_transverse_rhombro,vectorz_middhombros,
vectory_middhombros)
            #codo and muneca
            derecho_codo_flexion = Angle(vector_derecho_forearm,vector_derecho_codoarm)
            derecho_muneca_flexion = Angle(vector_derecho_hand,vector_derecho_forearmmuneca)
            angles = [sagital_derecho_angle, frontal_derecho_angle, transverse_derecho_angle, derecho_codo_flexion,
derecho_muneca_flexion, sagital_izquierdo_angle, frontal_izquierdo_angle, transverse_izquierdo_angle,
izquierdo_codo_flexion, izquierdo_muneca_flexion]
            """
            Comentado, no utilizado en la tesis
```

```
            if draw:
                overlay = img.copy()
                # Draw 3 transparent spheres on the frame
                center1 = (int(izquierdo_hombro[0]), int(izquierdo_hombro[1]))
                radius1 = size
                color1 = (240, 0, 0)
                thickness1 = -1
                cv2.circle(overlay, np.multiply(center1,100), radius1, color1, thickness1)
                center2 = (int(derecho_hombro[0]), int(derecho_hombro[1]))
                radius2 = size
                color2 = (0, 255, 0)
                thickness2 = -1
                cv2.circle(overlay,  np.multiply(center2,100), radius2, color2, thickness2)
                #cv2.putText(img, "sagital derecho"+str(int(sagital_derecho_angle)), (center1[0] - 20, center1[1] + 20),
                                #cv2.FONT_HERSHEY_PLAIN,1, (0, 0, 255), 1)
                center3 = (int(List[13][1]), int(List[13][2]))
                radius3 = size
                color3 = (0, 0, 255)
                thickness3 = -1
                cv2.circle(overlay,  np.multiply(center3,100), radius3, color3, thickness3)
                # Add the spheres to the frame with a transparency of 0.5
                # Set the center and radius of the sphere
                radius = size
                # Set the axes and orientation of the sphere
                axes = (radius, radius)
                angle = 0
                start_angle = 0
                end_angle = 180
                # Set the color and thickness of the sphere
                color = (255, 0, 0)
                thickness = -1
                # Draw the sphere
                cv2.ellipse(overlay,  np.multiply(center1,100), axes, angle, start_angle, end_angle, color, thickness)
                img2 = cv2.addWeighted(overlay, transparency , img, 1 - transparency, 0)#, dtype=cv2.CV_8U)
            """
        else:
            lmlist=0
            img = img
            angles=0
            return  img, angles
        return  img, angles
if __name__ == '__main__':
    pass


def import_posture_dataset(namevideo):
    import pandas as pd
    import numpy as np
    path='/home/diego/Documents/AT_Counting/Saved_files/'
    df='Thesis_Export_csv/bodylogger3D_'+namevideo+'.csv'

    dataframe = pd.read_csv(path+df, sep=',', header=None).values.tolist()
    timepath = "Thesis_Export_csv/bodytimelogger_"+namevideo+'.csv'
    timelogger = pd.read_csv(path+timepath, sep=',', header=None)
    back_frames_df = pd.read_csv(path+'thesis_txt_mediapipe/'+namevideo+'back_frames.csv', sep=',', header=None)
    back_frames =back_frames_df.values.tolist() #aparece con un [] demás
    a=[]
    for i in range(len(back_frames)):
        a.append(back_frames[i][0])
    back_frames = a
    posture_dataset_3D=[]
    List=[]
    j=0
    for i in range(len(dataframe[0])):
        x,y,z = dataframe[0][i], dataframe[1][i], dataframe[2][i]
        List.append([x,y,z])
        j=j+1
        if j==25:
            posture_dataset_3D.append(List)
            List=[]
            j=0
```

```
    posture_dataset_3D = np.asanyarray(posture_dataset_3D)
    ###### import 2D ##############################################################################
    df_2d ='Thesis_Export_csv/pos2Dcolor_'+namevideo+'.csv'
    p2ddataframe_2d = pd.read_csv(path+df_2d, sep=',', header=None).values.tolist()
    posture_dataset_2D=[]
    body2d=[]
    j2d=0
    for k in range(len(p2ddataframe_2d[0])):
        f,g = p2ddataframe_2d[0][k]//3.3333333333333335, p2ddataframe_2d[1][k]//3.3333333333333335
        body2d.append([f,g])
        #print(body)
        j2d=j2d+1
        if j2d==25:
            posture_dataset_2D.append(body2d)
            body2d=[]
            j2d=0
    lendataset3d = len(dataframe[0])//25
    return posture_dataset_3D , posture_dataset_2D , timelogger , back_frames, lendataset3d
if __name__ == '__main__':
    pass

def BPAfront(img, lmList, transparency=0.5, draw=True):
        import numpy as np
        import cv2
        #NOW 3 STEPS
        #  STEP NUMBER 1 : SET JOINT LANDMARKS COORDINATES POSITION.
        if len(lmList) !=0:
            #first define points then vectors by substraction between them.
            izquierdo_cadera_x, izquierdo_cadera_y, izquierdo_cadera_z = lmList[16][0:3]
            derecho_cadera_x, derecho_cadera_y, derecho_cadera_z = lmList[12][0:3]
            izquierdo_cadera = np.array([izquierdo_cadera_x, izquierdo_cadera_y, izquierdo_cadera_z])
            derecho_cadera = np.array([derecho_cadera_x, derecho_cadera_y, derecho_cadera_z])
            middcaderas = np.array([(izquierdo_cadera_x+derecho_cadera_x)/2, (izquierdo_cadera_y+derecho_cadera_y)
/2, (izquierdo_cadera_z+derecho_cadera_z)/2])
            #izquierdohombro and derechohombro
            izquierdo_hombro_x, izquierdo_hombro_y, izquierdo_hombro_z = lmList[8][0:3]
            derecho_hombro_x, derecho_hombro_y, derecho_hombro_z = lmList[4][0:3]
            derecho_hombro = np.array([derecho_hombro_x, derecho_hombro_y, derecho_hombro_z])
            izquierdo_hombro = np.array([izquierdo_hombro_x, izquierdo_hombro_y, izquierdo_hombro_z])
            #middle point between izquierdo and derecho hombro
            middhombros = np.array([(izquierdo_hombro_x+derecho_hombro_x)/2, (izquierdo_hombro_y+derecho_hombro_y)
/2, (izquierdo_hombro_z+derecho_hombro_z)/2])
            #codos munecas and tip of izquierdo and derecho hand
            izquierdo_codo_x, izquierdo_codo_y, izquierdo_codo_z = lmList[9][0:3]
            derecho_codo_x, derecho_codo_y, derecho_codo_z = lmList[5][0:3]
            derecho_codo = np.array([derecho_codo_x, derecho_codo_y, derecho_codo_z])
            izquierdo_codo = np.array([izquierdo_codo_x, izquierdo_codo_y, izquierdo_codo_z])
            izquierdo_muneca_x, izquierdo_muneca_y, izquierdo_muneca_z = lmList[10][0:3]
            derecho_muneca_x, derecho_muneca_y, derecho_muneca_z = lmList[6][0:3]
            derecho_muneca = np.array([derecho_muneca_x, derecho_muneca_y, derecho_muneca_z])
            izquierdo_muneca = np.array([izquierdo_muneca_x, izquierdo_muneca_y, izquierdo_muneca_z])
            lmiddfingers = np.array([lmList[23][0:3]])
            rmiddfingers = np.array([lmList[21][0:3]])

        # STEP NUMBER 2 : DEFINES JOINT LOCAL AXIS COORDINATE SYSTEM  BY USING NP.SUBSTRACT FOR CALCULARIONS
            #middcadera
            vx_middcadera = np.subtract(izquierdo_cadera, middcaderas)
            vy_middcadera = np.subtract(middhombros, middcaderas)
            vz_middcadera = np.cross(vy_middcadera, vx_middcadera)
            #middhombros
            vectorx_middhombros = np.subtract(izquierdo_hombro, middhombros)
            vectory_middhombros = -np.subtract(middcaderas,middhombros) #the same as vy_middcadera but starting
from hombro position and points up
            vectorz_middhombros = np.cross(-vectorx_middhombros,vectory_middhombros)
            # izquierdo_
            vector_izquierdo_arm = np.subtract(izquierdo_codo, izquierdo_hombro)
            vector_izquierdo_forearm = np.subtract(izquierdo_muneca, izquierdo_codo)
            vector_izquierdo_codoarm = - np.subtract(izquierdo_hombro,izquierdo_codo)
            # izquierdomuneca  # first we have to create the middfingers coordinates
            vector_izquierdo_hand = np.subtract(lmiddfingers, izquierdo_muneca)
            vector_izquierdo_forearmmuneca = - np.subtract(izquierdo_codo, izquierdo_muneca)
            # derecho SIDE
            # hombro
            vector_derecho_arm = np.subtract(derecho_codo,derecho_hombro)
```

```
            # codo
            vector_derecho_forearm = np.subtract(derecho_muneca , derecho_codo)
            vector_derecho_codoarm = - np.subtract(derecho_hombro,derecho_codo)
            # muneca  # first we have to create the middfingers coordinates
            vector_derecho_hand = np.subtract(rmiddfingers , derecho_muneca)
            vector_derecho_forearmmuneca = - np.subtract(derecho_codo, derecho_muneca)

        # STEP NUMBER 3: ANGLE CALCULATIONS BETWEEN BONE VECTORS AND JOINT COORDINATES AXIS (FOR hombroS) OR BETWWEN BONES (FOR codoS AND muneca).
          # ANGLES izquierdo SIDE
            #angles L hombro
            projectedvector_sagital_lhombro = ProjectOnPlane(vector_izquierdo_arm, vectorx_middhombros)
            sagital_izquierdo_angle = abs(SignedAngle(projectedvector_sagital_lhombro,-vectory_middhombros, vectorx_middhombros))
            projectedvector_frontal_lhombro = ProjectOnPlane(vector_izquierdo_arm, vectorz_middhombros)
            frontal_izquierdo_angle = abs(SignedAngle(projectedvector_frontal_lhombro,-vectory_middhombros, vectorz_middhombros))
            projectedvector_transverse_lhombro = ProjectOnPlane(vector_izquierdo_arm, -vectory_middhombros)
            transverse_izquierdo_angle = -SignedAngle(projectedvector_transverse_lhombro,vectorz_middhombros,-vectory_middhombros)
            # codo and muneca
            izquierdo_codo_flexion = Angle(vector_izquierdo_forearm,vector_izquierdo_codoarm)
            izquierdo_muneca_flexion = Angle(vector_izquierdo_hand,vector_izquierdo_forearmmuneca)

        # derecho SIDE
            # hombro
            projectedvector_sagital_rhombro = ProjectOnPlane(vector_derecho_arm, vectorx_middhombros)
            sagital_derecho_angle = abs(SignedAngle(projectedvector_sagital_rhombro,-vectory_middhombros, vectorx_middhombros))
            projectedvector_frontal_rhombro = ProjectOnPlane(vector_derecho_arm, vectorz_middhombros)
            frontal_derecho_angle = abs(SignedAngle(projectedvector_frontal_rhombro,-vectory_middhombros, vectorz_middhombros))
            projectedvector_transverse_rhombro = ProjectOnPlane(vector_derecho_arm, -vectory_middhombros)
            transverse_derecho_angle = SignedAngle(projectedvector_transverse_rhombro,vectorz_middhombros,-vectory_middhombros)
            #codo and muneca
            derecho_codo_flexion = Angle(vector_derecho_forearm,vector_derecho_codoarm)
            derecho_muneca_flexion = Angle(vector_derecho_hand,vector_derecho_forearmmuneca)
            angles = [sagital_derecho_angle, frontal_derecho_angle, transverse_derecho_angle, derecho_codo_flexion, derecho_muneca_flexion, sagital_izquierdo_angle, frontal_izquierdo_angle, transverse_izquierdo_angle, izquierdo_codo_flexion, izquierdo_muneca_flexion]
        else:
            img = img
            angles=0
            return  img, angles
        return  img, angles


def BPABack(img, lmList, transparency=0.5, draw=True, datasetdata=False):
        import numpy as np
        import cv2
        #NOW 3 STEPS
        #  STEP NUMBER 1 : SET JOINT LANDMARKS COORDINATES POSITION.
        if len(lmList) !=0:
            #first define points then vectors by substraction between them.
            izquierdo_cadera_x, izquierdo_cadera_y, izquierdo_cadera_z = lmList[12][0:3]
            derecho_cadera_x, derecho_cadera_y, derecho_cadera_z = lmList[16][0:3]
            izquierdo_cadera = np.array([izquierdo_cadera_x, izquierdo_cadera_y, izquierdo_cadera_z])
            derecho_cadera = np.array([derecho_cadera_x, derecho_cadera_y, derecho_cadera_z])
            middcaderas = np.array([(izquierdo_cadera_x+derecho_cadera_x)/2, (izquierdo_cadera_y+derecho_cadera_y)/2, (izquierdo_cadera_z+derecho_cadera_z)/2])
            #izquierdohombro and derechohombro
            izquierdo_hombro_x, izquierdo_hombro_y, izquierdo_hombro_z = lmList[4][0:3]
            derecho_hombro_x, derecho_hombro_y, derecho_hombro_z = lmList[8][0:3]
            derecho_hombro = np.array([derecho_hombro_x, derecho_hombro_y, derecho_hombro_z])
            izquierdo_hombro = np.array([izquierdo_hombro_x, izquierdo_hombro_y, izquierdo_hombro_z])
            #middle point between izquierdo and derecho hombro
            middhombros = np.array([(izquierdo_hombro_x+derecho_hombro_x)/2, (izquierdo_hombro_y+derecho_hombro_y)/2, (izquierdo_hombro_z+derecho_hombro_z)/2])
            #codos munecas and tip of izquierdo and derecho hand
            izquierdo_codo_x, izquierdo_codo_y, izquierdo_codo_z = lmList[5][0:3]
            derecho_codo_x, derecho_codo_y, derecho_codo_z = lmList[9][0:3]
            derecho_codo = np.array([derecho_codo_x, derecho_codo_y, derecho_codo_z])
            izquierdo_codo = np.array([izquierdo_codo_x, izquierdo_codo_y, izquierdo_codo_z])
            izquierdo_muneca_x, izquierdo_muneca_y, izquierdo_muneca_z = lmList[6][0:3]
```

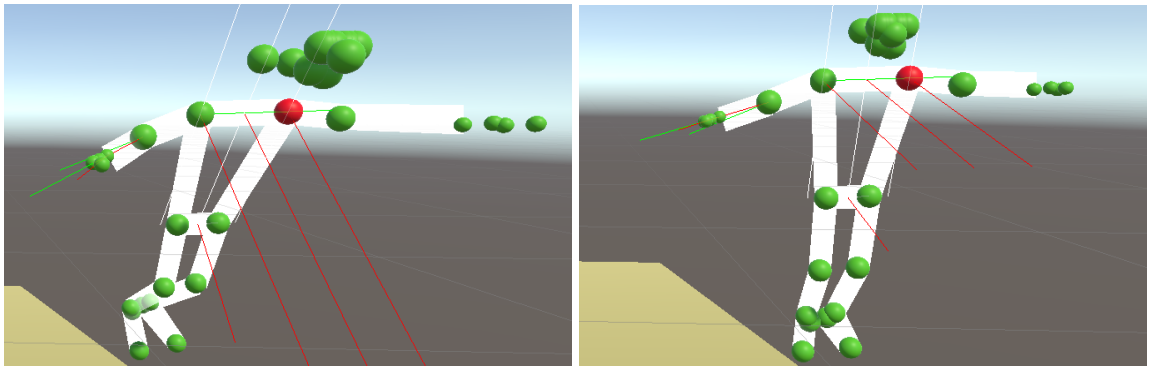
```
            derecho_muneca_x, derecho_muneca_y, derecho_muneca_z = lmList[10][0:3]
            derecho_muneca = np.array([derecho_muneca_x, derecho_muneca_y, derecho_muneca_z])
            izquierdo_muneca = np.array([izquierdo_muneca_x, izquierdo_muneca_y, izquierdo_muneca_z])
            lmiddfingers = np.array([lmList[21][0:3]])
            rmiddfingers = np.array([lmList[23][0:3]])

        # STEP NUMBER 2 : DEFINES JOINT LOCAL AXIS COORDINATE SYSTEM  BY USING NP.SUBSTRACT FOR CALCULARIONS
            #middcadera
            vx_middcadera = np.subtract(izquierdo_cadera, middcaderas)
            vy_middcadera = np.subtract(middhombros, middcaderas)
            vz_middcadera = np.cross(vy_middcadera, vx_middcadera)
            #middhombros
            vectorx_middhombros = np.subtract(izquierdo_hombro, middhombros)
            vectory_middhombros = -np.subtract(middcaderas,middhombros) #the same as vy_middcadera but starting
from hombro position and points up
            vectorz_middhombros = np.cross(-vectorx_middhombros,vectory_middhombros)
            # izquierdo_
            vector_izquierdo_arm = np.subtract(izquierdo_codo, izquierdo_hombro)
            vector_izquierdo_forearm = np.subtract(izquierdo_muneca, izquierdo_codo)
            vector_izquierdo_codoarm = - np.subtract(izquierdo_hombro,izquierdo_codo)
            # izquierdomuneca  # first we have to create the middfingers coordinates
            vector_izquierdo_hand = np.subtract(lmiddfingers, izquierdo_muneca)
            vector_izquierdo_forearmmuneca = - np.subtract(izquierdo_codo, izquierdo_muneca)
            # derecho SIDE
            # derechohombro
            vector_derecho_arm = np.subtract(derecho_codo,derecho_hombro)
            # derechocodo
            vector_derecho_forearm = np.subtract(derecho_muneca , derecho_codo)
            vector_derecho_codoarm = - np.subtract(derecho_hombro,derecho_codo)
            # derechomuneca  # first we have to create the middfingers coordinates
            vector_derecho_hand = np.subtract(rmiddfingers , derecho_muneca)
            vector_derecho_forearmmuneca = - np.subtract(derecho_codo, derecho_muneca)

        # STEP NUMBER 3: ANGLE CALCULATIONS BETWEEN BONE VECTORS AND JOINT COORDINATES AXIS (FOR hombroS) OR
BETWWEN BONES (FOR codoS AND muneca).
          # ANGLES izquierdo SIDE
            #angles L hombro
            projectedvector_sagital_lhombro = ProjectOnPlane(vector_izquierdo_arm, vectorx_middhombros)
            sagital_izquierdo_angle = abs(SignedAngle(projectedvector_sagital_lhombro,-vectory_middhombros,
vectorx_middhombros))
            projectedvector_frontal_lhombro = ProjectOnPlane(vector_izquierdo_arm, vectorz_middhombros)
            frontal_izquierdo_angle = abs(SignedAngle(projectedvector_frontal_lhombro,-vectory_middhombros,
vectorz_middhombros))
            projectedvector_transverse_lhombro = ProjectOnPlane(vector_izquierdo_arm, vectory_middhombros)
            transverse_izquierdo_angle = -SignedAngle(projectedvector_transverse_lhombro,vectorz_middhombros,
vectory_middhombros)
            # codo and muneca
            izquierdo_codo_flexion = Angle(vector_izquierdo_forearm,vector_izquierdo_codoarm)
            izquierdo_muneca_flexion = Angle(vector_izquierdo_hand,vector_izquierdo_forearmmuneca)
        # derecho SIDE
            # hombro
            projectedvector_sagital_rhombro = ProjectOnPlane(vector_derecho_arm, vectorx_middhombros)
            sagital_derecho_angle = abs(SignedAngle(projectedvector_sagital_rhombro,-vectory_middhombros,
vectorx_middhombros))
            projectedvector_frontal_rhombro = ProjectOnPlane(vector_derecho_arm, vectorz_middhombros)
            frontal_derecho_angle = abs(SignedAngle(projectedvector_frontal_rhombro,-vectory_middhombros,
vectorz_middhombros))
            projectedvector_transverse_rhombro = ProjectOnPlane(vector_derecho_arm, vectory_middhombros)
            transverse_derecho_angle = SignedAngle(projectedvector_transverse_rhombro,vectorz_middhombros,-
vectory_middhombros)
            #codo and muneca
            derecho_codo_flexion = Angle(vector_derecho_forearm,vector_derecho_codoarm)
            derecho_muneca_flexion = Angle(vector_derecho_hand,vector_derecho_forearmmuneca)
            angles = [sagital_derecho_angle, frontal_derecho_angle, transverse_derecho_angle, derecho_codo_flexion,
derecho_muneca_flexion, sagital_izquierdo_angle, frontal_izquierdo_angle, transverse_izquierdo_angle,
izquierdo_codo_flexion, izquierdo_muneca_flexion]
        else:
            lmlist=0
            img = img
            angles=0
            return  img, angles
        return  img, angles
```

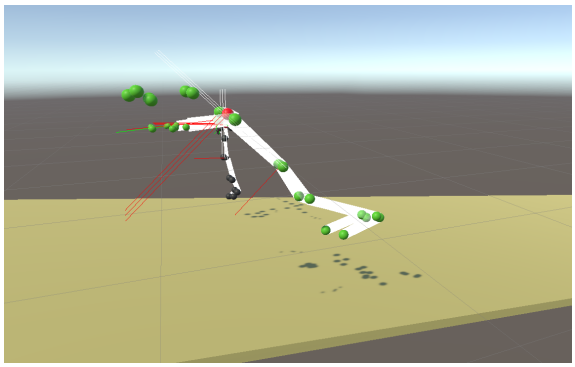
11.6. Anexo: Ajuste en el componente «Z» de las coordenadas de cada punto anatómico de Mediapipe Pose

Desde una perspectiva frontal no se observa gran variación en la estimación del componente Z de Mediapipe, sin embargo, al cambiar la perspectiva se aprecia una distorsión en la proporcion de los segmentos corporales, por lo que se llevaron a cabo ajustes a aproximadamente $\frac{1}{3}$ del valor entregado por Mediapipe.

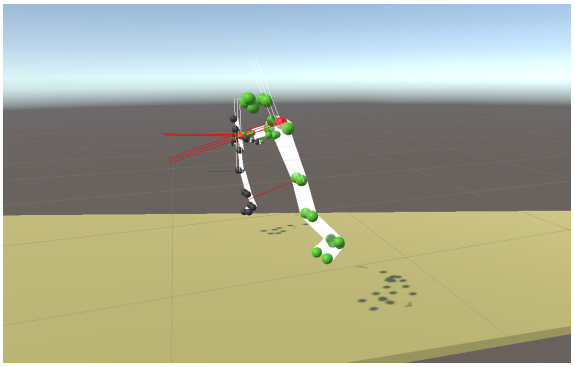


(a) Componente «Z» no escalado

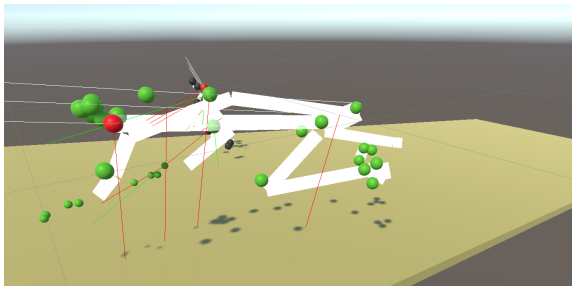
(b) Componente «Z» escalado



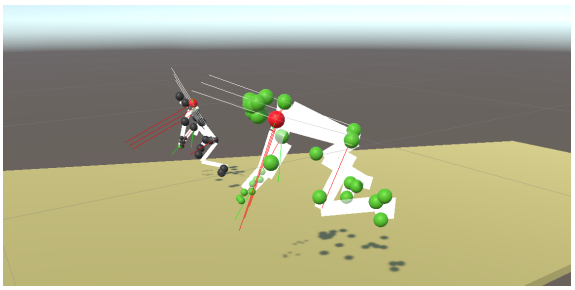
(c) Componente «Z» no escalado



(d) Componente «Z» escalado



(e) Componente «Z» no escalado



(f) Componente «Z» escalado

Figura 28: Efecto de escalar a $\frac{1}{3}$ el componente «Z» de las coordenadas estimadas por Mediapipe Pose (en verde) en la reconstrucción de los puntos anatómicos en el espacio 3D del software Unity Engine.
Las Figuras 28a, 28b, muestran cuadro n°6 del video n°9 en la posición «T» desde una vista frontal, con la finalidad de comparar la reconstrucción de Mediapipe Pose sin escalar mediante la división tipo $floor$ el componente Z de cada punto anatómico. Las Figuras 28a y 28b muestran la misma posición desde una perspectiva lateral, donde se observa la reconstrucción sin escalar con una postura con inclinación anterior de tronco (el componente Z de los vectores reconstruidos, tienden a apuntar hacia abajo), en cambio, al escalar dicha componente la reconstrucción se observa más vertical (y el componente Z se observa más horizontal. Por último, se presenta las Figuras 28e y 28f donde muestra la posición agachado en el cuadro n° 335 del video n°9, con una reconstrucción más proporcional a la figura humana cuando se escala el componente en 28f, en comparación a la reconstrucción presente en 28e.

11.7. Anexo: Detalle de resultados de concordancia y correlación entre ángulos de Dataset y Mediapipe

11.7.1. Anexo: Tablas de concordancia y correlación

La Tabla 13 y la Tabla 14 muestran la fuerza de correlación entre el ángulo calculado a partir de ambos conjuntos de datos, ordenados de manera decreciente.

Tabla 13: Concordancia y Correlación entre Ángulos de Dataset y Mediapipe (Lado Derecho)

Video-segmento corporal	Concordancia Lin	Categoría Concordancia	Correlación r-Pearson	Categoría Correlación
video 9 flexion_hombro_derecho	0.911943	Alta (+)	0.912134	Alta (+)
video 9 abduccion_hombro_derecho	0.896010	Alta (+)	0.912358	Alta (+)
video 6 abduccion_hombro_derecho	0.891577	Alta (+)	0.894910	Alta (+)
video 4 abduccion_hombro_derecho	0.880467	Alta (+)	0.890299	Alta (+)
video 4 flexion_hombro_derecho	0.859924	Alta (+)	0.871281	Alta (+)
video 6 flexion_hombro_derecho	0.850969	Alta (+)	0.852322	Alta (+)
video 18 abduccion_hombro_derecho	0.774875	Moderada (+)	0.792825	Moderada (+)
video 12 flexion_hombro_derecho	0.768245	Moderada (+)	0.774799	Moderada (+)
video 18 flexion_hombro_derecho	0.765898	Moderada (+)	0.766021	Moderada (+)
video 5 flexion_hombro_derecho	0.754012	Moderada (+)	0.757162	Moderada (+)
video 16 flexion_hombro_derecho	0.742726	Moderada (+)	0.746648	Moderada (+)
video 5 abduccion_hombro_derecho	0.726380	Moderada (+)	0.739352	Moderada (+)
video 10 flexion_hombro_derecho	0.724650	Moderada (+)	0.725725	Moderada (+)
video 1 flexion_hombro_derecho	0.697512	Moderada (+)	0.713902	Moderada (+)
video 14 flexion_hombro_derecho	0.695291	Moderada (+)	0.698080	Moderada (+)
video 15 abduccion_hombro_derecho	0.693634	Moderada (+)	0.696558	Moderada (+)
video 12 abduccion_hombro_derecho	0.691057	Moderada (+)	0.704929	Moderada (+)
video 7 flexion_hombro_derecho	0.686841	Moderada (+)	0.702434	Moderada (+)
video 15 flexion_hombro_derecho	0.683095	Moderada (+)	0.695757	Moderada (+)
video 7 abduccion_hombro_derecho	0.680884	Moderada (+)	0.706473	Moderada (+)
video 4 flexion_codo_derecho	0.679270	Moderada (+)	0.753970	Moderada (+)
video 1 abduccion_hombro_derecho	0.674227	Moderada (+)	0.674517	Moderada (+)
video 13 flexion_hombro_derecho	0.673849	Moderada (+)	0.678286	Moderada (+)
video 13 abduccion_hombro_derecho	0.654153	Moderada (+)	0.654452	Moderada (+)
video 16 abduccion_hombro_derecho	0.633103	Moderada (+)	0.639803	Moderada (+)
video 10 abduccion_hombro_derecho	0.629908	Moderada (+)	0.633406	Moderada (+)
video 14 abduccion_hombro_derecho	0.613232	Moderada (+)	0.628994	Moderada (+)
video 2 flexion_hombro_derecho	0.613206	Moderada (+)	0.622991	Moderada (+)
video 17 flexion_hombro_derecho	0.586185	Moderada (+)	0.586822	Moderada (+)
video 19 flexion_hombro_derecho	0.580931	Moderada (+)	0.581307	Moderada (+)
video 17 flexion_codo_derecho	0.579032	Moderada (+)	0.604152	Moderada (+)
video 2 abduccion_hombro_derecho	0.573832	Moderada (+)	0.597344	Moderada (+)
video 9 flexion_codo_derecho	0.564503	Moderada (+)	0.587091	Moderada (+)

Continúa en la página siguiente

Tabla 13 – Continuación de la página anterior

Video-segmento corporal	Concordancia Lin	Categoría Concordancia	Correlación	Categoría Correlación
video 6 flexion_codo_derecho	0.539441	Moderada (+)	0.608329	Moderada (+)
video 19 abduccion_hombro_derecho	0.534811	Moderada (+)	0.550519	Moderada (+)
video 15 flexion_codo_derecho	0.525028	Moderada (+)	0.562499	Moderada (+)
video 11 flexion_hombro_derecho	0.512375	Moderada (+)	0.525920	Moderada (+)
video 1 flexion_codo_derecho	0.506591	Moderada (+)	0.533077	Moderada (+)
video 11 abduccion_hombro_derecho	0.504568	Moderada (+)	0.514414	Moderada (+)
video 12 flexion_codo_derecho	0.500464	Moderada (+)	0.532626	Moderada (+)
video 16 flexion_codo_derecho	0.495252	Moderada (+)	0.518187	Moderada (+)
video 4 rotacion_hombro_derecho	0.483334	Moderada (+)	0.577795	Moderada (+)
video 13 flexion_codo_derecho	0.467982	Moderada (+)	0.497723	Moderada (+)
video 17 abduccion_hombro_derecho	0.462197	Moderada (+)	0.465262	Moderada (+)
video 20 abduccion_hombro_derecho	0.450735	Moderada (+)	0.455939	Moderada (+)
video 8 flexion_hombro_derecho	0.443863	Moderada (+)	0.458154	Moderada (+)
video 20 flexion_hombro_derecho	0.443240	Moderada (+)	0.456982	Moderada (+)
video 19 flexion_codo_derecho	0.438438	Moderada (+)	0.458072	Moderada (+)
video 9 rotacion_hombro_derecho	0.391417	Débil (+)	0.442376	Moderada (+)
video 8 abduccion_hombro_derecho	0.366196	Débil (+)	0.377821	Débil (+)
video 10 flexion_codo_derecho	0.360750	Débil (+)	0.391136	Débil (+)
video 2 flexion_codo_derecho	0.359048	Débil (+)	0.407054	Moderada (+)
video 18 flexion_codo_derecho	0.355815	Débil (+)	0.381653	Débil (+)
video 8 rotacion_hombro_derecho	0.352952	Débil (+)	0.394447	Débil (+)
video 7 rotacion_hombro_derecho	0.297609	Débil (+)	0.325440	Débil (+)
video 15 rotacion_hombro_derecho	0.293977	Débil (+)	0.331229	Débil (+)
video 6 rotacion_hombro_derecho	0.289036	Débil (+)	0.311105	Débil (+)
video 19 rotacion_hombro_derecho	0.286340	Débil (+)	0.327737	Débil (+)
video 14 flexion_codo_derecho	0.265058	Débil (+)	0.297656	Débil (+)
video 17 rotacion_hombro_derecho	0.248864	Débil (+)	0.277420	Débil (+)
video 7 flexion_codo_derecho	0.245773	Débil (+)	0.283101	Débil (+)
video 5 flexion_codo_derecho	0.241914	Débil (+)	0.279029	Débil (+)
video 10 rotacion_hombro_derecho	0.239661	Débil (+)	0.273752	Débil (+)
video 2 rotacion_hombro_derecho	0.223634	Débil (+)	0.270393	Débil (+)
video 11 flexion_codo_derecho	0.217985	Débil (+)	0.264854	Débil (+)
video 16 rotacion_hombro_derecho	0.214838	Débil (+)	0.295109	Débil (+)
video 20 flexion_codo_derecho	0.207996	Débil (+)	0.233956	Débil (+)
video 11 flexion_muneca_derecha	0.189024	Muy Débil (+)	0.198423	Muy Débil (+)
video 3 flexion_hombro_derecho	0.187331	Muy Débil (+)	0.392156	Débil (+)
video 5 rotacion_hombro_derecho	0.181245	Muy Débil (+)	0.197618	Muy Débil (+)
video 8 flexion_codo_derecho	0.151391	Muy Débil (+)	0.174189	Muy Débil (+)
video 11 rotacion_hombro_derecho	0.148858	Muy Débil (+)	0.194694	Muy Débil (+)
video 16 flexion_muneca_derecha	0.133015	Muy Débil (+)	0.136773	Muy Débil (+)

Continúa en la página siguiente

Tabla 13 – Continuación de la página anterior

Video-segmento corporal	Concordancia Lin	Categoría Concordancia	Correlación	Categoría Correlación
video 3 abduccion_hombro_derecho	0.128615	Muy Débil (+)	0.289109	Débil (+)
video 9 flexion_muneca_derecha	0.109939	Muy Débil (+)	0.125179	Muy Débil (+)
video 1 rotacion_hombro_derecho	0.104715	Muy Débil (+)	0.173526	Muy Débil (+)
video 14 flexion_muneca_derecha	0.103353	Muy Débil (+)	0.107468	Muy Débil (+)
video 19 flexion_muneca_derecha	0.101047	Muy Débil (+)	0.106186	Muy Débil (+)
video 17 flexion_muneca_derecha	0.072665	Muy Débil (+)	0.073482	Muy Débil (+)
video 20 flexion_muneca_derecha	0.072326	Muy Débil (+)	0.073336	Muy Débil (+)
video 14 rotacion_hombro_derecho	0.072119	Muy Débil (+)	0.093025	Muy Débil (+)
video 2 flexion_muneca_derecha	0.069982	Muy Débil (+)	0.071057	Muy Débil (+)
video 3 rotacion_hombro_derecho	0.061967	Muy Débil (+)	0.092257	Muy Débil (+)
video 7 flexion_muneca_derecha	0.061393	Muy Débil (+)	0.063250	Muy Débil (+)
video 4 flexion_muneca_derecha	0.051884	Muy Débil (+)	0.052103	Muy Débil (+)
video 8 flexion_muneca_derecha	0.046892	Muy Débil (+)	0.047812	Muy Débil (+)
video 13 flexion_muneca_derecha	0.044449	Muy Débil (+)	0.048277	Muy Débil (+)
video 13 rotacion_hombro_derecho	0.043247	Muy Débil (+)	0.051554	Muy Débil (+)
video 12 flexion_muneca_derecha	0.039833	Muy Débil (+)	0.042102	Muy Débil (+)
video 6 flexion_muneca_derecha	0.038082	Muy Débil (+)	0.038732	Muy Débil (+)
video 5 flexion_muneca_derecha	0.026206	Muy Débil (+)	0.026952	Muy Débil (+)
video 1 flexion_muneca_derecha	0.015110	Muy Débil (+)	0.015397	Muy Débil (+)
video 3 flexion_muneca_derecha	0.012008	Muy Débil (+)	0.019292	Muy Débil (+)
video 12 rotacion_hombro_derecho	0.009674	Muy Débil (+)	0.012127	Muy Débil (+)
video 10 flexion_muneca_derecha	0.008956	Muy Débil (+)	0.009264	Muy Débil (+)
video 3 flexion_codo_derecho	0.003472	Muy Débil (+)	0.007740	Muy Débil (+)
video 20 rotacion_hombro_derecho	0.002978	Muy Débil (+)	0.004052	Muy Débil (+)
video 18 rotacion_hombro_derecho	-0.001639	Muy Débil (-)	-0.002350	Muy Débil (-)
video 15 flexion_muneca_derecha	-0.008842	Muy Débil (-)	-0.008865	Muy Débil (-)
video 18 flexion_muneca_derecha	-0.097448	Muy Débil (-)	-0.102489	Muy Débil (-)

Tabla 14: Concordancia y Correlación entre Ángulos de Dataset y Mediapipe (Lado Izquierdo)

Video-segmento corporal	Concordancia Lin	Categoría Concordancia	Correlación r-Pearson	Categoría Correlación
video 15 abduccion_hombro_izquierdo	0.872181	Alta (+)	0.872713	Alta (+)
video 5 flexion_hombro_izquierdo	0.867618	Alta (+)	0.872348	Alta (+)
video 4 flexion_hombro_izquierdo	0.866095	Alta (+)	0.867432	Alta (+)
video 5 abduccion_hombro_izquierdo	0.853597	Alta (+)	0.860655	Alta (+)
video 10 abduccion_hombro_izquierdo	0.846255	Alta (+)	0.849505	Alta (+)
video 15 flexion_hombro_izquierdo	0.844586	Alta (+)	0.847723	Alta (+)
video 10 flexion_hombro_izquierdo	0.843985	Alta (+)	0.846264	Alta (+)
video 13 flexion_hombro_izquierdo	0.830756	Alta (+)	0.832483	Alta (+)

Continúa en la página siguiente

Tabla 14 – Continuación de la página anterior

Video-segmento corporal	Concordancia Lin	Categoría Concordancia	Correlación	Categoría Correlación
video 6 flexion_hombro_izquierdo	0.830329	Alta (+)	0.831837	Alta (+)
video 19 abduccion_hombro_izquierdo	0.823158	Alta (+)	0.826687	Alta (+)
video 4 abduccion_hombro_izquierdo	0.803010	Alta (+)	0.810573	Alta (+)
video 13 abduccion_hombro_izquierdo	0.800486	Alta (+)	0.806747	Alta (+)
video 2 flexion_hombro_izquierdo	0.795437	Moderada (+)	0.800781	Alta (+)
video 7 abduccion_hombro_izquierdo	0.782394	Moderada (+)	0.784034	Moderada (+)
video 7 flexion_hombro_izquierdo	0.781294	Moderada (+)	0.786415	Moderada (+)
video 6 abduccion_hombro_izquierdo	0.760431	Moderada (+)	0.783385	Moderada (+)
video 17 abduccion_hombro_izquierdo	0.756842	Moderada (+)	0.758485	Moderada (+)
video 19 flexion_hombro_izquierdo	0.749472	Moderada (+)	0.753352	Moderada (+)
video 2 abduccion_hombro_izquierdo	0.748932	Moderada (+)	0.762311	Moderada (+)
video 18 abduccion_hombro_izquierdo	0.727160	Moderada (+)	0.745757	Moderada (+)
video 11 flexion_hombro_izquierdo	0.724454	Moderada (+)	0.733209	Moderada (+)
video 17 flexion_hombro_izquierdo	0.716807	Moderada (+)	0.716879	Moderada (+)
video 1 abduccion_hombro_izquierdo	0.716263	Moderada (+)	0.726904	Moderada (+)
video 18 flexion_hombro_izquierdo	0.711299	Moderada (+)	0.740505	Moderada (+)
video 11 abduccion_hombro_izquierdo	0.703590	Moderada (+)	0.710700	Moderada (+)
video 9 abduccion_hombro_izquierdo	0.695041	Moderada (+)	0.723486	Moderada (+)
video 8 abduccion_hombro_izquierdo	0.690477	Moderada (+)	0.691996	Moderada (+)
video 9 flexion_hombro_izquierdo	0.689027	Moderada (+)	0.718394	Moderada (+)
video 12 flexion_hombro_izquierdo	0.677377	Moderada (+)	0.715727	Moderada (+)
video 12 abduccion_hombro_izquierdo	0.677153	Moderada (+)	0.686232	Moderada (+)
video 1 flexion_hombro_izquierdo	0.666836	Moderada (+)	0.682114	Moderada (+)
video 8 flexion_hombro_izquierdo	0.663472	Moderada (+)	0.670196	Moderada (+)
video 14 abduccion_hombro_izquierdo	0.660821	Moderada (+)	0.664622	Moderada (+)
video 14 flexion_hombro_izquierdo	0.648499	Moderada (+)	0.673783	Moderada (+)
video 16 flexion_hombro_izquierdo	0.590519	Moderada (+)	0.591471	Moderada (+)
video 8 flexion_codo_izquierdo	0.589114	Moderada (+)	0.615207	Moderada (+)
video 6 flexion_codo_izquierdo	0.581326	Moderada (+)	0.597731	Moderada (+)
video 5 flexion_codo_izquierdo	0.578885	Moderada (+)	0.625373	Moderada (+)
video 12 flexion_codo_izquierdo	0.568963	Moderada (+)	0.584803	Moderada (+)
video 16 abduccion_hombro_izquierdo	0.551574	Moderada (+)	0.560526	Moderada (+)
video 4 rotacion_hombro_izquierdo	0.543715	Moderada (+)	0.585808	Moderada (+)
video 13 flexion_codo_izquierdo	0.528455	Moderada (+)	0.555465	Moderada (+)
video 17 flexion_codo_izquierdo	0.509406	Moderada (+)	0.544703	Moderada (+)
video 19 flexion_codo_izquierdo	0.507383	Moderada (+)	0.526234	Moderada (+)
video 2 flexion_codo_izquierdo	0.502208	Moderada (+)	0.536817	Moderada (+)
video 15 flexion_codo_izquierdo	0.496141	Moderada (+)	0.530704	Moderada (+)
video 4 flexion_codo_izquierdo	0.491231	Moderada (+)	0.535188	Moderada (+)
video 18 flexion_codo_izquierdo	0.490070	Moderada (+)	0.509139	Moderada (+)

Continúa en la página siguiente

Tabla 14 – Continuación de la página anterior

Video-segmento corporal	Concordancia Lin	Categoría Concordancia	Correlación	Categoría Correlación
video 20 flexion_hombro_izquierdo	0.430875	Moderada (+)	0.437774	Moderada (+)
video 10 flexion_codo_izquierdo	0.419645	Moderada (+)	0.450288	Moderada (+)
video 7 flexion_codo_izquierdo	0.381777	Débil (+)	0.412336	Moderada (+)
video 14 flexion_codo_izquierdo	0.371045	Débil (+)	0.396512	Débil (+)
video 13 rotacion_hombro_izquierdo	0.364374	Débil (+)	0.379658	Débil (+)
video 16 flexion_codo_izquierdo	0.353687	Débil (+)	0.377834	Débil (+)
video 1 flexion_codo_izquierdo	0.334878	Débil (+)	0.373193	Débil (+)
video 20 abduccion_hombro_izquierdo	0.322909	Débil (+)	0.346900	Débil (+)
video 9 flexion_codo_izquierdo	0.320423	Débil (+)	0.346884	Débil (+)
video 9 rotacion_hombro_izquierdo	0.320075	Débil (+)	0.326700	Débil (+)
video 19 rotacion_hombro_izquierdo	0.307533	Débil (+)	0.315836	Débil (+)
video 5 rotacion_hombro_izquierdo	0.254961	Débil (+)	0.262633	Débil (+)
video 11 flexion_codo_izquierdo	0.251665	Débil (+)	0.275447	Débil (+)
video 7 rotacion_hombro_izquierdo	0.193124	Muy Débil (+)	0.197181	Muy Débil (+)
video 6 rotacion_hombro_izquierdo	0.189173	Muy Débil (+)	0.193854	Muy Débil (+)
video 15 rotacion_hombro_izquierdo	0.187250	Muy Débil (+)	0.200535	Débil (+)
video 18 rotacion_hombro_izquierdo	0.164200	Muy Débil (+)	0.205039	Débil (+)
video 8 rotacion_hombro_izquierdo	0.139197	Muy Débil (+)	0.188005	Muy Débil (+)
video 20 flexion_codo_izquierdo	0.127730	Muy Débil (+)	0.131956	Muy Débil (+)
video 11 flexion_muneca_izquierda	0.126683	Muy Débil (+)	0.148789	Muy Débil (+)
video 11 rotacion_hombro_izquierdo	0.125744	Muy Débil (+)	0.173959	Muy Débil (+)
video 12 rotacion_hombro_izquierdo	0.114469	Muy Débil (+)	0.125484	Muy Débil (+)
video 3 flexion_hombro_izquierdo	0.106776	Muy Débil (+)	0.163756	Muy Débil (+)
video 10 rotacion_hombro_izquierdo	0.101240	Muy Débil (+)	0.107011	Muy Débil (+)
video 17 rotacion_hombro_izquierdo	0.099814	Muy Débil (+)	0.129777	Muy Débil (+)
video 14 rotacion_hombro_izquierdo	0.089015	Muy Débil (+)	0.115920	Muy Débil (+)
video 3 rotacion_hombro_izquierdo	0.056347	Muy Débil (+)	0.096860	Muy Débil (+)
video 16 flexion_muneca_izquierda	0.051479	Muy Débil (+)	0.053595	Muy Débil (+)
video 1 rotacion_hombro_izquierdo	0.038538	Muy Débil (+)	0.045842	Muy Débil (+)
video 16 rotacion_hombro_izquierdo	0.036807	Muy Débil (+)	0.045796	Muy Débil (+)
video 17 flexion_muneca_izquierda	0.032368	Muy Débil (+)	0.035606	Muy Débil (+)
video 3 abduccion_hombro_izquierdo	0.029926	Muy Débil (+)	0.040787	Muy Débil (+)
video 20 rotacion_hombro_izquierdo	0.023214	Muy Débil (+)	0.033367	Muy Débil (+)
video 2 rotacion_hombro_izquierdo	0.005938	Muy Débil (+)	0.007439	Muy Débil (+)
video 4 flexion_muneca_izquierda	-0.017916	Muy Débil (-)	-0.021871	Muy Débil (-)
video 3 flexion_codo_izquierdo	-0.019315	Muy Débil (-)	-0.040985	Muy Débil (-)
video 1 flexion_muneca_izquierda	-0.023088	Muy Débil (-)	-0.023757	Muy Débil (-)
video 14 flexion_muneca_izquierda	-0.028746	Muy Débil (-)	-0.030366	Muy Débil (-)
video 20 flexion_muneca_izquierda	-0.037740	Muy Débil (-)	-0.038928	Muy Débil (-)
video 8 flexion_muneca_izquierda	-0.048877	Muy Débil (-)	-0.050354	Muy Débil (-)

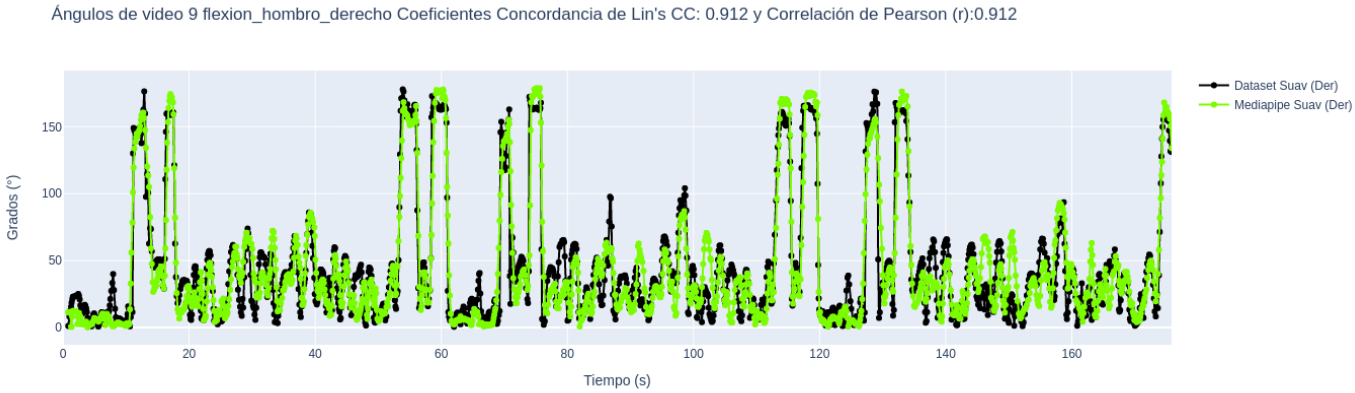
Continúa en la página siguiente

Tabla 14 – Continuación de la página anterior

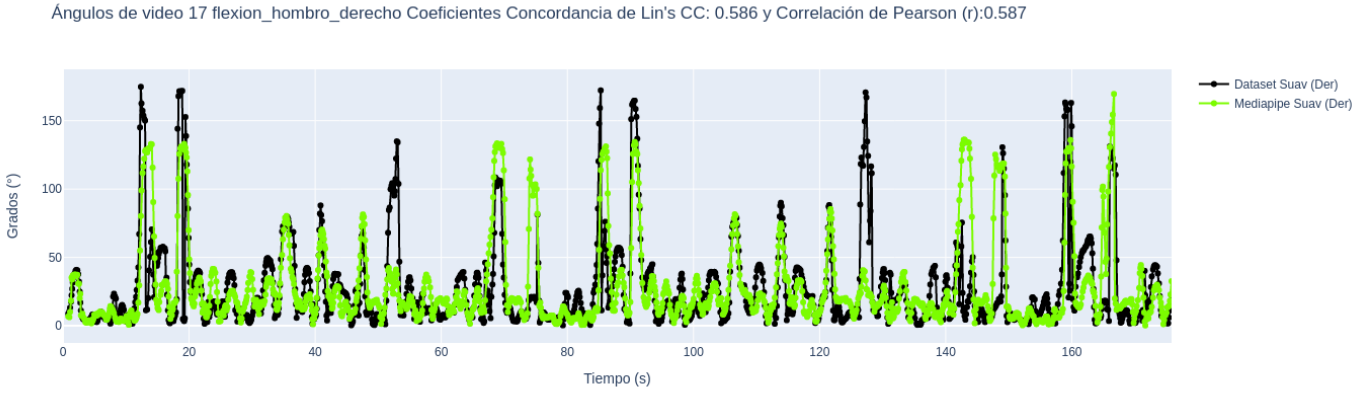
Video-segmento corporal	Concordancia Lin	Categoría Concordancia	Correlación	Categoría Correlación
video 19 flexion_muneca_izquierda	-0.050058	Muy Débil (-)	-0.051219	Muy Débil (-)
video 6 flexion_muneca_izquierda	-0.057132	Muy Débil (-)	-0.061797	Muy Débil (-)
video 3 flexion_muneca_izquierda	-0.066425	Muy Débil (-)	-0.085671	Muy Débil (-)
video 9 flexion_muneca_izquierda	-0.088236	Muy Débil (-)	-0.090560	Muy Débil (-)
video 18 flexion_muneca_izquierda	-0.089889	Muy Débil (-)	-0.097094	Muy Débil (-)
video 13 flexion_muneca_izquierda	-0.092094	Muy Débil (-)	-0.096371	Muy Débil (-)
video 2 flexion_muneca_izquierda	-0.098243	Muy Débil (-)	-0.102600	Muy Débil (-)
video 12 flexion_muneca_izquierda	-0.113939	Muy Débil (-)	-0.118946	Muy Débil (-)
video 7 flexion_muneca_izquierda	-0.147163	Muy Débil (-)	-0.155000	Muy Débil (-)
video 10 flexion_muneca_izquierda	-0.155131	Muy Débil (-)	-0.157414	Muy Débil (-)
video 15 flexion_muneca_izquierda	-0.157617	Muy Débil (-)	-0.171696	Muy Débil (-)
video 5 flexion_muneca_izquierda	-0.230288	Débil (-)	-0.248745	Débil (-)

11.7.2. Anexo: Gráficos de las curvas del movimiento, ejemplos en base a la concordancia y correlación

De manera complementaria, y con la finalidad de presentar gráficos del comportamiento de las curvas analizadas, en las Figuras 29 a 33 muestran un ejemplo de la parte alta de la tabla, presentada en 11.7 Anexo: Detalle de resultados de concordancia y correlación entre ángulos de Dataset y Mediapipe, y un ejemplo de la parte baja de la tabla referente a cada tipo de movimiento, los que se presentan en el siguiente orden de acuerdo al coeficiente de concordancia: flexión de hombro, abducción de hombro, flexión de codo, rotación de hombro y flexión de muñeca. Una característica general que surge luego de visualizar todas figuras a continuación, es que en la medida que el coeficiente de concordancia se va reduciendo, aumenta tanto la cantidad y como la magnitud de algunos peaks que están presentes en el $Dataset_{Suavizado}$ y que no siempre están presentes en las curvas de $Mediapipe_{Suavizado}$.

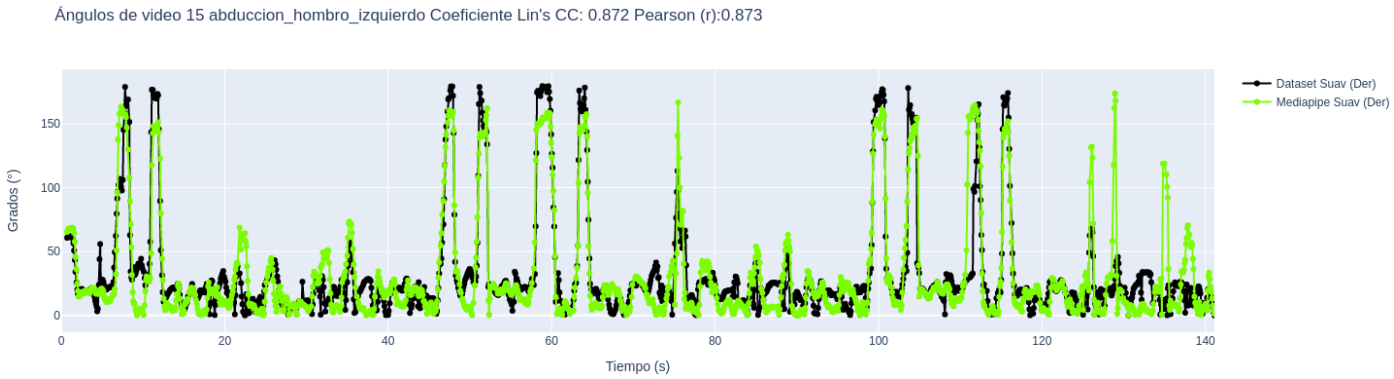


(a)

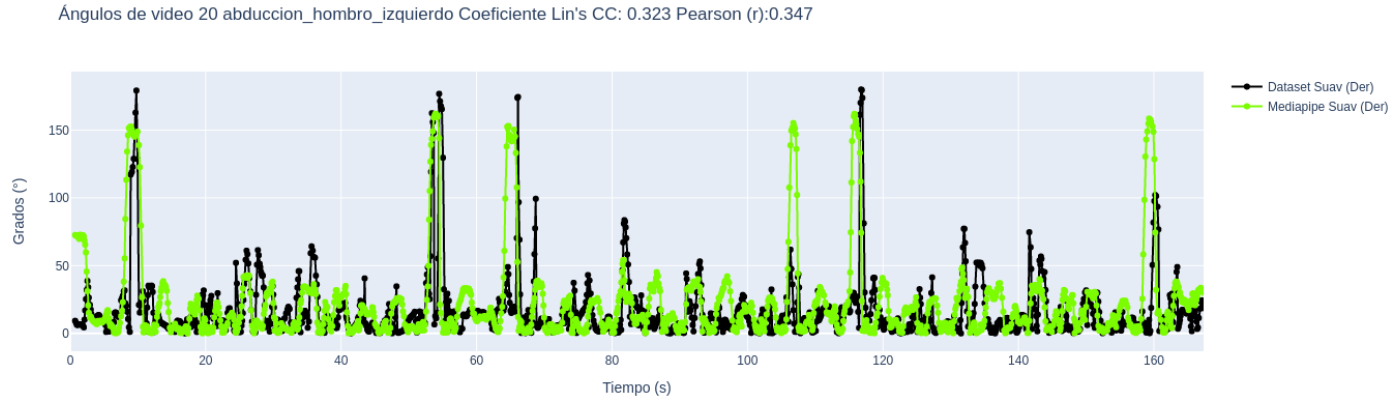


(b)

Figura 29: Curvas de flexión hombro derecho de concordancia y correlación alta (+) 29a y moderada (+) 29b entre Dataset suavizado y Mediapipe suavizado.

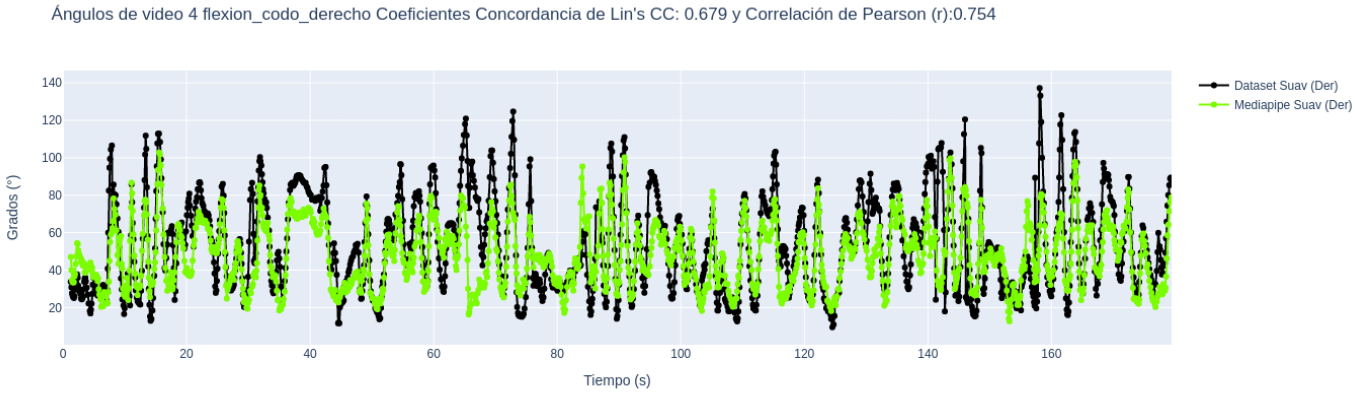


(a)

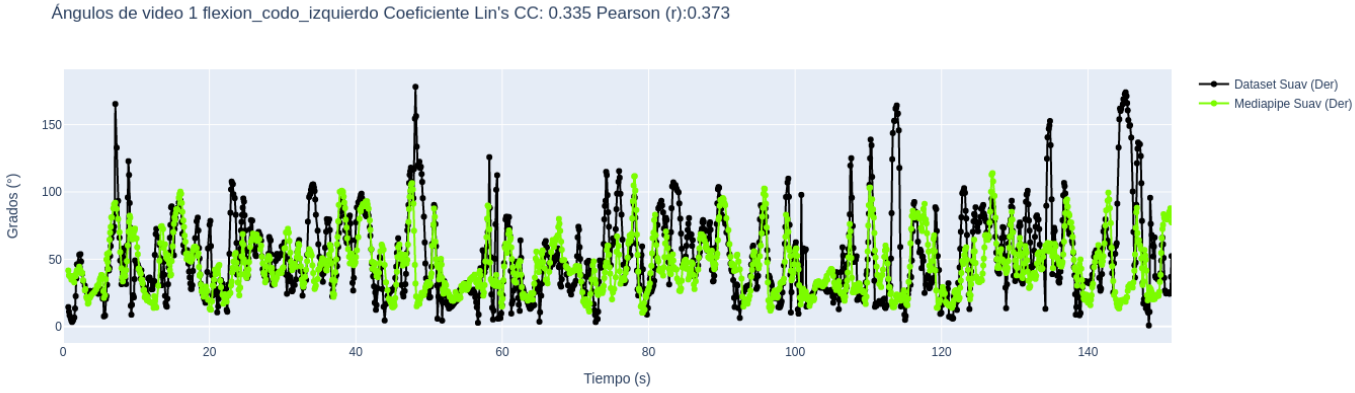


(b)

Figura 30: Curvas de abducción hombro izquierdo de concordancia y correlación alta (+) 30a y débil (+) 30bentre Dataset suavizado y Mediapipe suavizado.

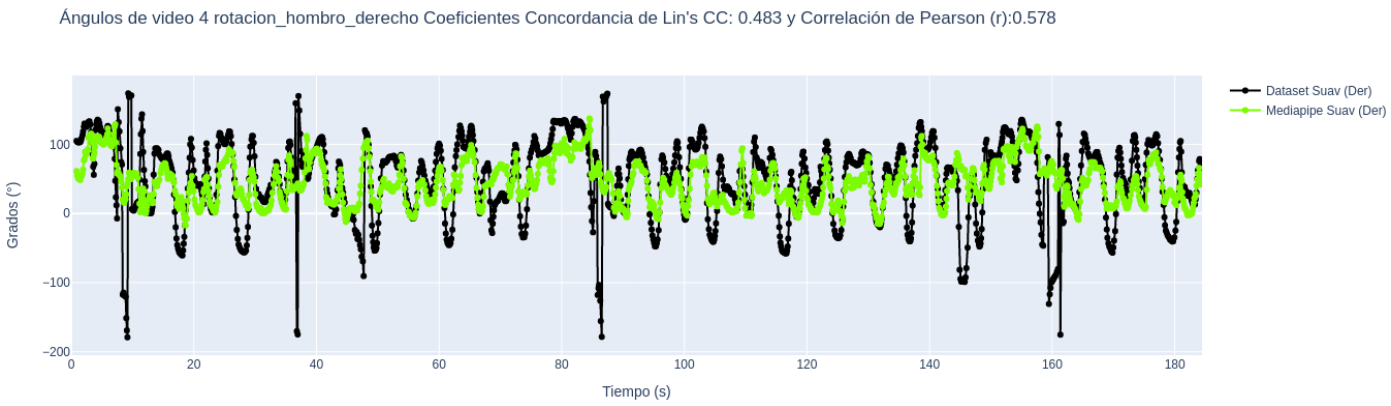


(a)

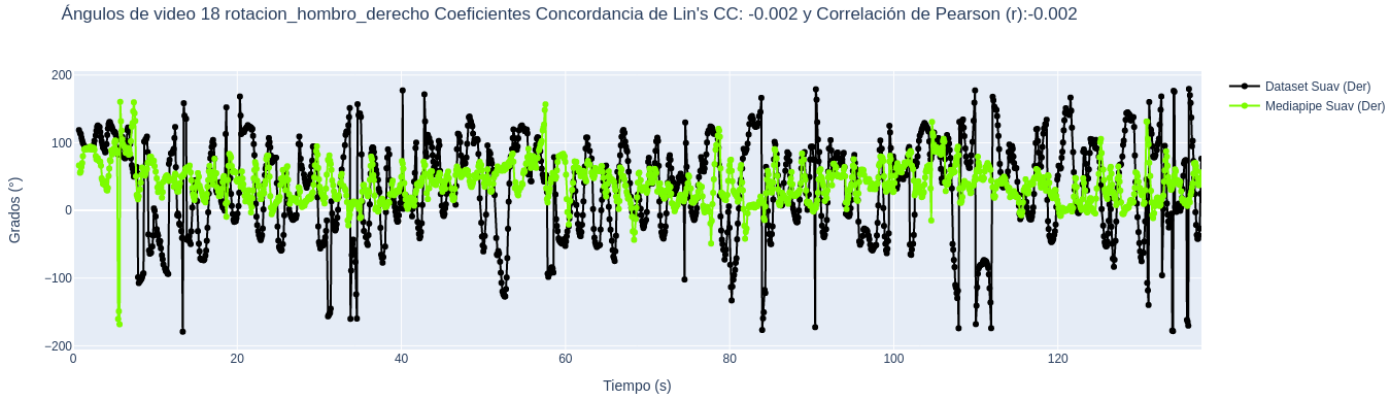


(b)

Figura 31: Curvas de flexión de codo derecho de concordancia y correlación moderada (+) y débil (+) entre Dataset suavizado y Mediapipe suavizado.

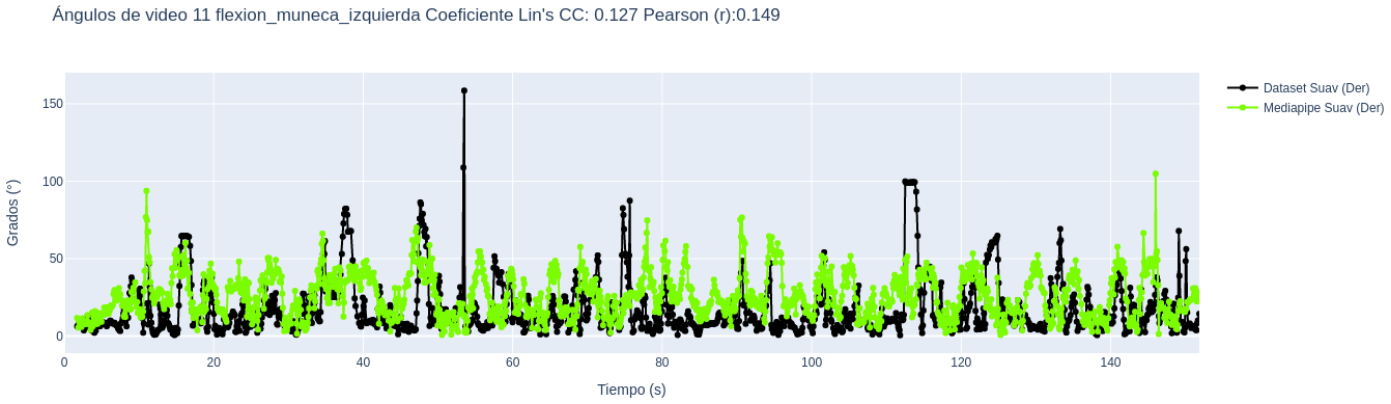


(a)

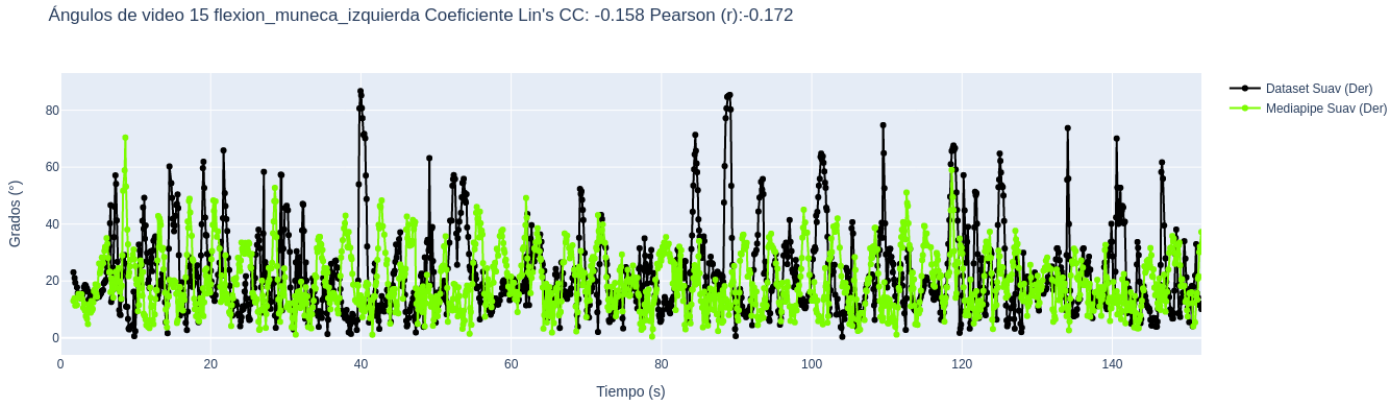


(b)

Figura 32: Curvas de rotación hombro derecho de concordancia y correlación moderada (+) y muy débil (-) entre Dataset suavizado y Mediapipe suavizado.



(a)



(b)

Figura 33: Curvas de muñeca izquierda de concordancia muy débil (+) y muy débil (-) entre Dataset suavizado y Mediapipe suavizado.

11.7.3. Anexo: Orden decreciente de los videos según el el valor promedio de concordancia de Lin

Se presenta la Tabla 15, que ordena de manera decreciente los videos según el el valor promedio de concordancia de Lin de todo el conjunto de movimientos.

video	Extremidad izquierda Concordancia Lin Promedio $\pm$ desv.std	Extremidad izquierda Clasificación de Concordancia Lin	Extremidad derecha Concordancia Lin Promedio $\pm$ desv.std	Extremidad derecha Clasificación de Concordancia Lin.
video 4	0.537 ± 0.350	Moderada (+)	0.517 ± 0.344	Moderada (+)
video 13	0.486 ± 0.377	Moderada (+)	0.347 ± 0.309	Débil (+)
video 19	0.467 ± 0.354	Moderada (+)	0.349 ± 0.218	Débil (+)
video 5	0.465 ± 0.462	Moderada (+)	0.350 ± 0.338	Débil (+)
video 6	0.461 ± 0.382	Moderada (+)	0.469 ± 0.376	Moderada (+)
video 15	0.449 ± 0.440	Moderada (+)	0.404 ± 0.319	Moderada (+)
video 17	0.423 ± 0.340	Moderada (+)	0.357 ± 0.221	Débil (+)
video 10	0.411 ± 0.445	Moderada (+)	0.351 ± 0.296	Débil (+)
video 8	0.407 ± 0.339	Moderada (+)	0.218 ± 0.139	Débil (+)
video 18	0.401 ± 0.356	Moderada (+)	0.329 ± 0.400	Débil (+)
video 7	0.398 ± 0.398	Débil (+)	0.341 ± 0.262	Débil (+)
video 2	0.391 ± 0.416	Débil (+)	0.307 ± 0.223	Débil (+)
video 9	0.387 ± 0.324	Débil (+)	0.534 ± 0.343	Moderada (+)
video 11	0.386 ± 0.303	Débil (+)	0.325 ± 0.265	Débil (+)
video 12	0.385 ± 0.363	Débil (+)	0.355 ± 0.319	Débil (+)
video 14	0.348 ± 0.315	Débil (+)	0.338 ± 0.310	Débil (+)
video 1	0.347 ± 0.343	Débil (+)	0.379 ± 0.344	Débil (+)
video 16	0.317 ± 0.265	Débil (+)	0.387 ± 0.260	Débil (+)
video 20	0.173 ± 0.199	Muy Débil (+)	0.256 ± 0.262	Débil (+)
video 3	0.021 ± 0.067	Muy Débil (+)	0.073 ± 0.078	Muy Débil (+)

Tabla 15: Tabla de la concordancia de los videos según el promedio de todos los movimientos, ordenados de manera decreciente.

La Tabla 15 muestra que la extremidad izquierda presenta 10 videos donde la concordancia resultante del análisis de todos los movimientos es Moderada (+), 8 videos se clasifican en concordancia Débil (+) y 2 videos Muy Débil (+). En el caso de la extremidad derecha sólo 4 videos poseen una concordancia Moderada (+), 15 en categoría Débil (+) y 1 en Muy Débil (+)

11.8. Anexo: Detalle de resultados conteo automático de «acciones técnicas dinámicas»

11.8.1. Anexo: Análisis de normalidad en las pruebas combinación de umbral

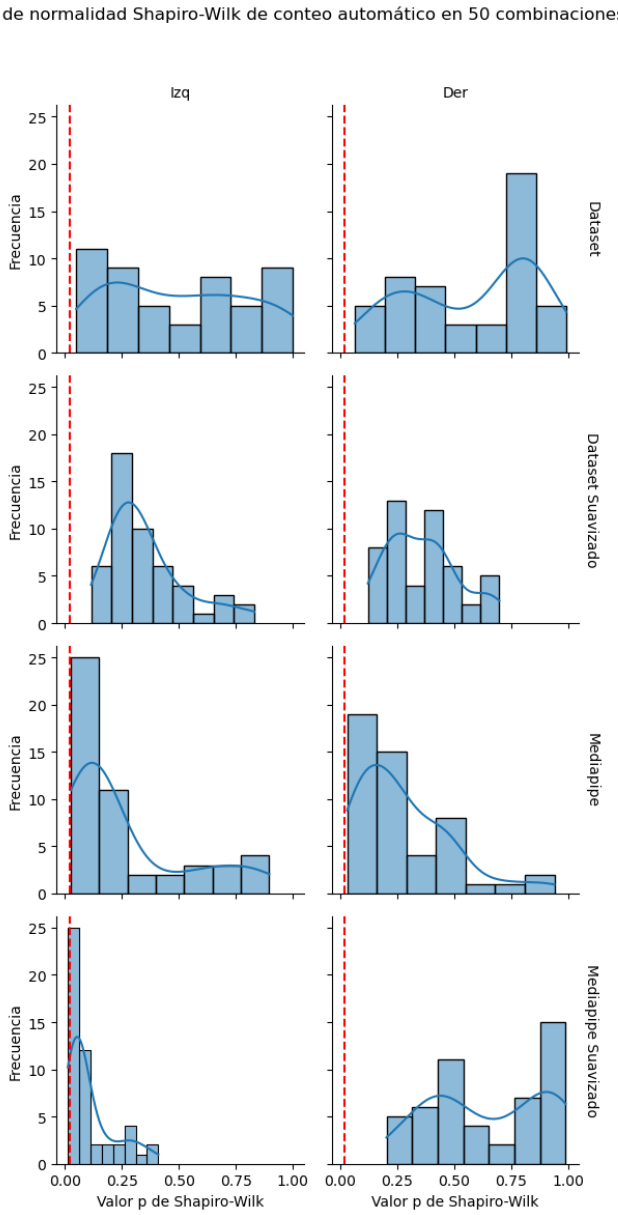


Figura 34: Histograma prueba de normalidad, con un $alpha = 0.05$, a los resultados de conteos automáticos en 50 pruebas de umbral en cada grupo de datos, incluyendo cada extremidad.
Este histograma representa los resultados de la prueba de normalidad Shapiro-Wilk aplicada a los conteos automáticos obtenidos en 50 pruebas de combinación de umbrales por cada extremidad en cada grupo de datos. La prueba se realizó utilizando un nivel de significancia (alfa) de 0.05 para evaluar si los datos se ajustan a una distribución normal.

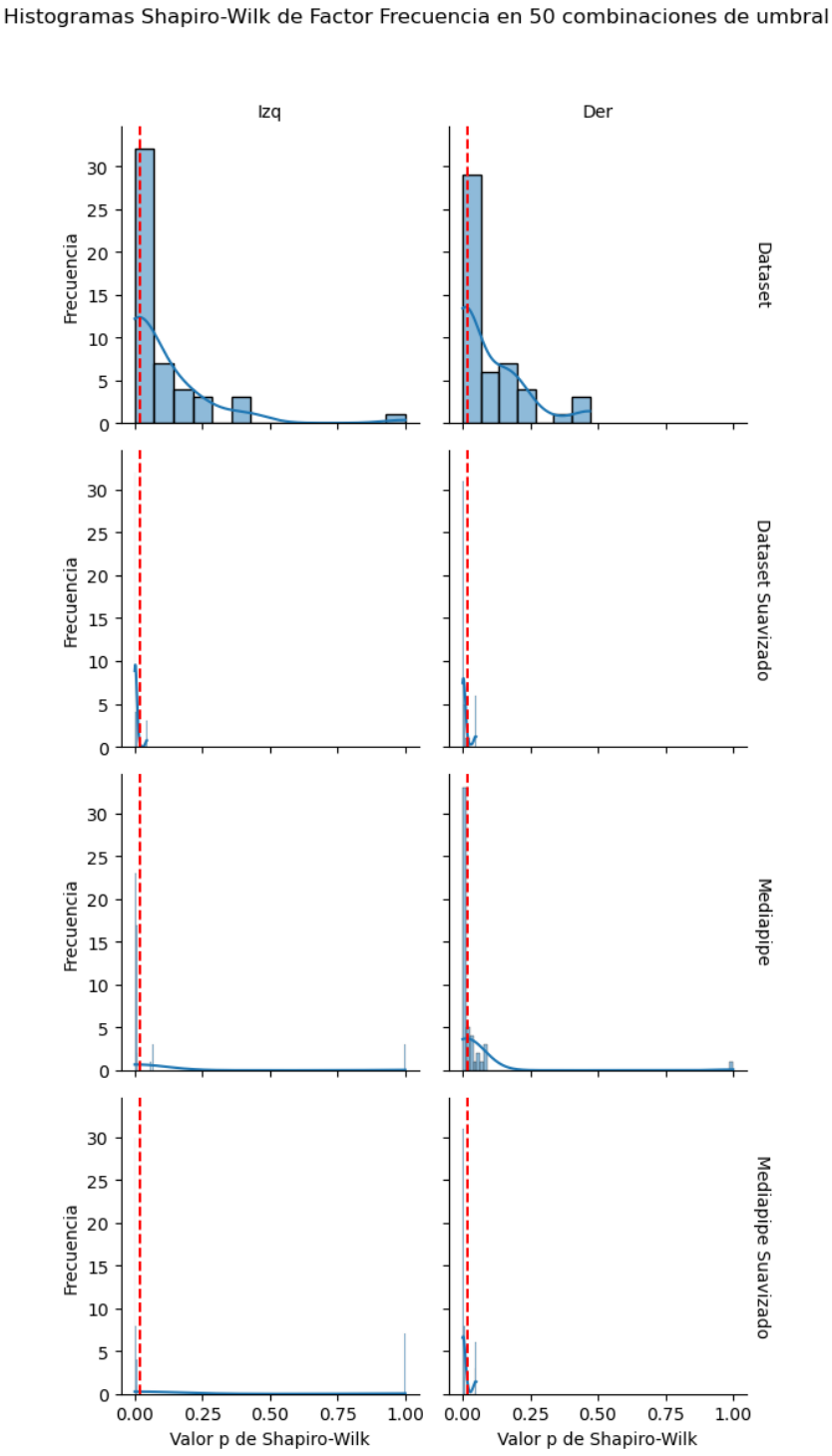


Figura 35: Histograma prueba de normalidad, con un $alpha = 0.05$, a los resultados del FF como resultado de los conteos automáticos en 50 pruebas de umbral en cada grupo de datos, incluyendo cada extremidad.

Este histograma representa los resultados de la prueba de normalidad Shapiro-Wilk aplicada al FF resultante de los conteos automáticos obtenidos en 50 pruebas de combinación de umbrales por cada extremidad en cada grupo de datos. La prueba se realizó utilizando un nivel de significancia (alfa) de 0.05 para evaluar si los datos se ajustan a una distribución normal.

11.8.2. Anexo: Combinación de umbrales que ofrecen el menor error respecto al conteo humano de consenso

Respecto al error en los conteos automáticos «$Error\ RMSE\ conteo$» comparado al «conteo humano de consenso», partiendo por el grupo de datos con el menor error y ordenados de manera incremental, para la extremidad derecha son: el grupo de datos de $Mediapipe$, usando la combinación de $UT = 0.6\ syUA = 25°$ ofreció el menor error entre todos los grupos de datos para el «$Error\ RMSE\ conteo = 5.96$», en los 19 videos, comparado al conteo humano de consenso. Luego, los resultados de $Dataset\_Suavizado$, con la combinación de $UT = 1\ s$ y $UA = 15°$ ofreció su menor error para el «$Error\ RMSE\ conteo = 5.52$». A continuación, el conjunto de datos de $Mediapipe\_Suavizado$, con la combinación de $UT = 0.7\ s$ y $UA = 15°$ ofreció su menor error para el «$Error\ RMSE\ conteo = 5.787$». Finalmente, el grupo $Dataset$, con la combinación de $UT = 0.6\ s$ y $UA = 25°$ ofreció el menor error para el «$Error\ RMSE\ conteo = 5.884$». Lo señalado puede ser observado en la Tabla 16a.

En el caso de la asignación del FF «$Error\ RMSE\ frecuencia$», el grupo de datos de $Mediapipe$, usando la combinación de $UT = 0.5\ s$ y $UA = 25$ ofreció el menor error entre todos los grupos de datos para el «$Error\ RMSE\ frecuencia = 0.548$», en los 19 videos, comparado al al FF asognado por el «conteo humano de consenso». Luego, los resultados de $Dataset\_Suavizado$, tuvo más de 1 combinación con el menor error: $UT = 0.9\ s$ (dos veces) y $UA = 5°$ y $10°$ ofreció el error para el «$Error\ RMSE\ frecuencia = 0.627$» en ambos casos. A continuación, el conjunto de datos de $Mediapipe\_Suavizado$, con la combinación de $UT = 0.8\ s$ y $UA = 10°$ ofreció su menor error para el «$Error\ RMSE\ frecuencia = 0.627$» también. Finalmente, el grupo $Dataset$, con la combinación de $UT = 0.7\ s$ y $UA = 25°$ ofreció el menor error para el «$Error\ RMSE\ frecuencia = 5.884$». Lo señalado puede ser observado en la Tabla 16a.

Grupo	Extremidad	umbral_temporal	umbral_amplitud	RMSE_conteo	RMSE_ffrec	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe	Der	0.600	25.000	5.096	0.756	0.564	Moderada (+)	0.334	Débil (+)	18.450	inf	1.000	0.720
Dataset Suavizado	Der	1.000	15.000	5.520	0.770	0.468	Moderada (+)	0.330	Débil (+)	20.248	inf	0.977	0.465
Mediapipe Suavizado	Der	0.700	15.000	5.787	0.805	0.461	Moderada (+)	0.245	Débil (+)	22.205	inf	0.793	1.000
Dataset	Der	0.800	25.000	5.884	0.865	0.359	Débil (+)	0.101	Muy Débil (+)	21.085	inf	0.953	0.728

(a) Basada en el error de los conteos $RMSE_conteos$

Grupo	Extremidad	umbral_temporal	umbral_amplitud	RMSE_conteo	RMSE_ffrec	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe	Der	0.500	25.000	5.357	0.548	0.580	Moderada (+)	0.689	Moderada (+)	22.302	inf	0.220	0.375
Dataset Suavizado	Der	0.900	5.000	6.844	0.627	0.392	Débil (+)	0.593	Moderada (+)	29.934	inf	0.021	0.135
Dataset Suavizado	Der	0.900	10.000	6.724	0.627	0.396	Débil (+)	0.593	Moderada (+)	29.378	inf	0.023	0.135
Mediapipe Suavizado	Der	0.800	10.000	6.676	0.627	0.392	Débil (+)	0.593	Moderada (+)	28.161	inf	0.062	0.135
Dataset	Der	0.700	25.000	6.505	0.747	0.356	Débil (+)	0.308	Débil (+)	26.675	inf	0.054	0.340

(b) Basada en el error del FF $Error\ RMSE\ frecuencia$

Tabla 16: Combinaciones de UT y UA con menor error RMSE respecto al «conteo humano de consenso» para $Dataset, Dataset\_Suavizado, Mediapipe, Mediapipe\_Suavizado$
La Tabla 16a presenta las combinaciones de umbral que resultaron con el menor error RMSE comparado con el conteo humano de consenso para cada grupo de datos, considerando los datos de ambas extremidades, siendo en todos superior la extremidad derecha. El menor error RMSE fue obtenido por Mediapipe (sin suavizar), tanto para el conteo automático de ATD y del FF, seguido de $Dataset\_Suavizado$, $Mediapipe\_Suavizado$ y $Dataset$, con el mayor errorr RMSE.

11.8.3. Anexo: Resultados de las distintas combinaciones de Umbral Temporal y Umbral de Amplitud

Tabla 17: Resultados de las distintas combinaciones de UT y UA. Ordenados de manera decreciente en base al coeficiente de concordancia de Lin de los conteos de ATD y del FF.

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe	Der	0.5	25.0	5.357 ± 2.524	0.55 ± 0.32	0.58	Moderada (+)	0.671	Moderada (+)	22.302 ± 16.743	0.05 ± 0.037	0.22	0.113
Mediapipe	Der	0.6	25.0	5.096 ± 2.749	0.669 ± 0.472	0.564	Moderada (+)	0.374	Débil (+)	18.45 ± 13.268	0.053 ± 0.054	1.0	0.663
Mediapipe	Der	0.4	25.0	6.514 ± 3.022	0.752 ± 0.447	0.511	Moderada (+)	0.555	Moderada (+)	27.953 ± 20.072	0.067 ± 0.051	0.033	0.015
Dataset Suavizado	Der	0.8	20.0	5.529 ± 3.319	0.716 ± 0.513	0.487	Moderada (+)	0.299	Débil (+)	18.874 ± 15.441	0.056 ± 0.059	0.953	0.816
Dataset Suavizado	Der	0.9	15.0	5.669 ± 2.713	0.679 ± 0.459	0.469	Moderada (+)	0.356	Débil (+)	22.866 ± 18.058	0.056 ± 0.052	0.397	0.379
Dataset Suavizado	Der	1.0	15.0	5.52 ± 2.964	0.669 ± 0.472	0.468	Moderada (+)	0.381	Débil (+)	20.248 ± 15.574	0.053 ± 0.054	0.977	0.913
Dataset Suavizado	Der	0.7	20.0	5.735 ± 3.003	0.688 ± 0.499	0.466	Moderada (+)	0.353	Débil (+)	21.642 ± 16.534	0.053 ± 0.057	0.559	0.498
Mediapipe	Der	0.1	25.0	7.324 ± 3.458	0.819 ± 0.408	0.465	Moderada (+)	0.573	Moderada (+)	31.298 ± 22.074	0.079 ± 0.047	0.008	0.011
Mediapipe	Der	0.2	25.0	7.324 ± 3.458	0.819 ± 0.408	0.465	Moderada (+)	0.573	Moderada (+)	31.298 ± 22.074	0.079 ± 0.047	0.008	0.011
Mediapipe	Der	0.3	25.0	7.324 ± 3.458	0.819 ± 0.408	0.465	Moderada (+)	0.573	Moderada (+)	31.298 ± 22.074	0.079 ± 0.047	0.008	0.011
Mediapipe Suavizado	Der	0.6	15.0	5.861 ± 2.851	0.725 ± 0.499	0.463	Moderada (+)	0.273	Débil (+)	23.067 ± 16.578	0.058 ± 0.057	0.521	0.317
Mediapipe Suavizado	Der	0.7	15.0	5.787 ± 2.846	0.752 ± 0.51	0.461	Moderada (+)	0.237	Débil (+)	22.205 ± 15.485	0.061 ± 0.058	0.793	0.424
Mediapipe Suavizado	Der	0.5	15.0	5.992 ± 2.751	0.649 ± 0.38	0.46	Moderada (+)	0.498	Moderada (+)	24.236 ± 16.936	0.058 ± 0.043	0.335	0.189
Mediapipe Suavizado	Der	0.1	20.0	5.95 ± 3.574	0.716 ± 0.513	0.456	Moderada (+)	0.338	Débil (+)	18.823 ± 12.636	0.056 ± 0.059	0.397	0.638
Mediapipe Suavizado	Der	0.2	20.0	5.95 ± 3.574	0.716 ± 0.513	0.456	Moderada (+)	0.338	Débil (+)	18.823 ± 12.636	0.056 ± 0.059	0.397	0.638
Mediapipe Suavizado	Der	0.3	20.0	5.95 ± 3.574	0.716 ± 0.513	0.456	Moderada (+)	0.338	Débil (+)	18.823 ± 12.636	0.056 ± 0.059	0.397	0.638
Mediapipe	Der	0.7	20.0	5.693 ± 2.823	0.679 ± 0.459	0.454	Moderada (+)	0.349	Débil (+)	23.103 ± 18.599	0.056 ± 0.052	0.35	0.229
Dataset Suavizado	Der	0.8	15.0	6.076 ± 2.742	0.761 ± 0.494	0.453	Moderada (+)	0.23	Débil (+)	25.74 ± 20.473	0.064 ± 0.056	0.096	0.069
Dataset Suavizado	Der	0.9	20.0	5.763 ± 3.671	0.725 ± 0.549	0.451	Moderada (+)	0.305	Débil (+)	18.02 ± 14.303	0.053 ± 0.063	0.483	0.79
Mediapipe Suavizado	Der	0.4	15.0	6.111 ± 2.893	0.649 ± 0.38	0.449	Moderada (+)	0.498	Moderada (+)	24.762 ± 17.962	0.058 ± 0.043	0.255	0.189
Mediapipe	Der	0.6	20.0	6.502 ± 2.836	0.659 ± 0.359	0.446	Moderada (+)	0.545	Moderada (+)	28.625 ± 21.906	0.061 ± 0.041	0.023	0.038
Mediapipe Suavizado	Der	0.8	15.0	5.797 ± 3.011	0.734 ± 0.484	0.445	Moderada (+)	0.255	Débil (+)	21.256 ± 14.87	0.061 ± 0.055	0.953	0.577
Mediapipe Suavizado	Der	0.4	20.0	6.023 ± 3.673	0.716 ± 0.513	0.443	Moderada (+)	0.338	Débil (+)	18.779 ± 12.807	0.056 ± 0.059	0.381	0.638
Mediapipe	Der	0.7	25.0	5.898 ± 3.765	0.769 ± 0.626	0.441	Moderada (+)	0.305	Débil (+)	17.645 ± 12.967	0.05 ± 0.071	0.267	0.091

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe Suavizado	Der	0.9	15.0	5.821 ± 3.347	0.679 ± 0.51	0.44	Moderada (+)	0.378	Débil (+)	19.921 ± 14.898	0.05 ± 0.058	0.661	0.95
Mediapipe Suavizado	Der	0.1	15.0	6.198 ± 2.928	0.679 ± 0.394	0.439	Moderada (+)	0.457	Moderada (+)	25.191 ± 18.333	0.061 ± 0.045	0.231	0.151
Mediapipe Suavizado	Der	0.2	15.0	6.198 ± 2.928	0.679 ± 0.394	0.439	Moderada (+)	0.457	Moderada (+)	25.191 ± 18.333	0.061 ± 0.045	0.231	0.151
Mediapipe Suavizado	Der	0.3	15.0	6.198 ± 2.928	0.679 ± 0.394	0.439	Moderada (+)	0.457	Moderada (+)	25.191 ± 18.333	0.061 ± 0.045	0.231	0.151
Dataset Suavizado	Der	0.6	20.0	6.114 ± 2.921	0.716 ± 0.456	0.436	Moderada (+)	0.292	Débil (+)	24.525 ± 18.844	0.061 ± 0.052	0.243	0.149
Mediapipe	Der	0.8	20.0	5.6 ± 3.163	0.734 ± 0.583	0.436	Moderada (+)	0.244	Débil (+)	19.801 ± 15.148	0.05 ± 0.066	0.884	1.0
Mediapipe Suavizado	Der	0.5	20.0	6.127 ± 3.797	0.761 ± 0.549	0.433	Moderada (+)	0.277	Débil (+)	18.712 ± 13.008	0.058 ± 0.063	0.307	0.309
Dataset Suavizado	Der	1.0	10.0	5.917 ± 2.789	0.698 ± 0.487	0.43	Moderada (+)	0.329	Débil (+)	24.918 ± 20.905	0.056 ± 0.056	0.161	0.149
Mediapipe Suavizado	Der	1.0	10.0	5.868 ± 2.603	0.716 ± 0.456	0.425	Moderada (+)	0.283	Débil (+)	23.816 ± 17.087	0.061 ± 0.052	0.502	0.379
Dataset Suavizado	Der	0.5	25.0	5.995 ± 3.402	0.743 ± 0.525	0.425	Moderada (+)	0.246	Débil (+)	20.873 ± 15.411	0.058 ± 0.06	0.907	0.498
Mediapipe	Izq	0.1	25.0	9.251 ± 4.692	1.1 ± 0.708	0.423	Moderada (+)	0.445	Moderada (+)	44.233 ± 34.975	0.094 ± 0.081	0.07	0.105
Mediapipe	Izq	0.2	25.0	9.251 ± 4.692	1.1 ± 0.708	0.423	Moderada (+)	0.445	Moderada (+)	44.233 ± 34.975	0.094 ± 0.081	0.07	0.105
Mediapipe	Izq	0.3	25.0	9.251 ± 4.692	1.1 ± 0.708	0.423	Moderada (+)	0.445	Moderada (+)	44.233 ± 34.975	0.094 ± 0.081	0.07	0.105
Mediapipe	Izq	0.4	25.0	8.842 ± 4.308	1.158 ± 0.848	0.422	Moderada (+)	0.344	Débil (+)	41.962 ± 31.977	0.088 ± 0.097	0.122	0.224
Mediapipe Suavizado	Der	0.6	20.0	6.291 ± 3.951	0.743 ± 0.573	0.419	Moderada (+)	0.338	Débil (+)	18.757 ± 12.752	0.053 ± 0.065	0.161	0.145
Mediapipe Suavizado	Izq	0.4	15.0	8.228 ± 3.921	1.187 ± 0.862	0.419	Moderada (+)	0.291	Débil (+)	34.985 ± 23.62	0.091 ± 0.098	0.293	0.663
Dataset Suavizado	Der	1.0	5.0	6.026 ± 2.815	0.725 ± 0.499	0.419	Moderada (+)	0.283	Débil (+)	25.615 ± 21.609	0.058 ± 0.057	0.108	0.118
Mediapipe Suavizado	Der	0.9	10.0	6.19 ± 2.966	0.761 ± 0.494	0.419	Moderada (+)	0.22	Débil (+)	25.46 ± 19.754	0.064 ± 0.056	0.161	0.214
Mediapipe Suavizado	Der	1.0	5.0	5.975 ± 2.681	0.743 ± 0.466	0.415	Moderada (+)	0.237	Débil (+)	24.466 ± 18.007	0.064 ± 0.053	0.381	0.317
Mediapipe Suavizado	Izq	0.1	15.0	8.3 ± 3.945	1.187 ± 0.862	0.414	Moderada (+)	0.291	Débil (+)	35.652 ± 24.997	0.091 ± 0.098	0.28	0.663
Mediapipe Suavizado	Izq	0.2	15.0	8.3 ± 3.945	1.187 ± 0.862	0.414	Moderada (+)	0.291	Débil (+)	35.652 ± 24.997	0.091 ± 0.098	0.28	0.663
Mediapipe Suavizado	Izq	0.3	15.0	8.3 ± 3.945	1.187 ± 0.862	0.414	Moderada (+)	0.291	Débil (+)	35.652 ± 24.997	0.091 ± 0.098	0.28	0.663
Dataset Suavizado	Der	0.7	25.0	6.131 ± 3.84	0.795 ± 0.638	0.412	Moderada (+)	0.184	Muy Débil (+)	18.959 ± 13.207	0.053 ± 0.073	0.397	0.348
Mediapipe Suavizado	Izq	0.5	15.0	8.219 ± 3.857	1.17 ± 0.863	0.408	Moderada (+)	0.314	Débil (+)	34.471 ± 22.003	0.088 ± 0.099	0.293	0.674
Dataset Suavizado	Der	0.5	20.0	6.529 ± 3.048	0.803 ± 0.46	0.405	Moderada (+)	0.232	Débil (+)	26.835 ± 20.538	0.073 ± 0.053	0.102	0.079
Dataset Suavizado	Der	0.1	25.0	6.278 ± 3.307	0.698 ± 0.487	0.404	Moderada (+)	0.344	Débil (+)	22.961 ± 16.626	0.056 ± 0.056	0.64	0.424
Dataset Suavizado	Der	0.2	25.0	6.278 ± 3.307	0.698 ± 0.487	0.404	Moderada (+)	0.344	Débil (+)	22.961 ± 16.626	0.056 ± 0.056	0.64	0.424

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Dataset Suavizado	Der	0.3	25.0	6.278 ± 3.307	0.698 ± 0.487	0.404	Moderada (+)	0.344	Débil (+)	22.961 ± 16.626	0.056 ± 0.056	0.64	0.424
Mediapipe Suavizado	Der	0.9	5.0	6.375 ± 3.017	0.786 ± 0.502	0.403	Moderada (+)	0.177	Muy Débil (+)	26.51 ± 20.536	0.067 ± 0.057	0.122	0.172
Dataset Suavizado	Der	0.6	25.0	6.104 ± 3.699	0.769 ± 0.535	0.401	Moderada (+)	0.219	Débil (+)	19.991 ± 14.744	0.061 ± 0.061	0.748	0.877
Dataset Suavizado	Der	0.7	15.0	6.867 ± 3.207	0.778 ± 0.454	0.398	Débil (+)	0.477	Moderada (+)	29.422 ± 23.441	0.07 ± 0.052	0.025	0.019
Mediapipe Suavizado	Der	0.7	20.0	6.514 ± 4.072	0.819 ± 0.649	0.397	Débil (+)	0.188	Muy Débil (+)	19.336 ± 12.775	0.056 ± 0.074	0.108	0.111
Dataset Suavizado	Der	0.9	10.0	6.724 ± 3.068	0.669 ± 0.413	0.396	Débil (+)	0.536	Moderada (+)	29.378 ± 24.184	0.058 ± 0.047	0.023	0.027
Mediapipe Suavizado	Der	1.0	15.0	6.115 ± 3.652	0.761 ± 0.549	0.395	Débil (+)	0.244	Débil (+)	19.758 ± 14.229	0.058 ± 0.063	0.381	0.549
Dataset Suavizado	Der	0.8	25.0	6.361 ± 4.097	0.811 ± 0.658	0.395	Débil (+)	0.22	Débil (+)	18.607 ± 12.453	0.053 ± 0.075	0.189	0.068
Dataset Suavizado	Der	0.4	25.0	6.239 ± 3.409	0.743 ± 0.525	0.394	Débil (+)	0.246	Débil (+)	22.39 ± 16.551	0.058 ± 0.06	0.683	0.498
Dataset Suavizado	Der	0.9	5.0	6.844 ± 3.149	0.669 ± 0.413	0.392	Débil (+)	0.536	Moderada (+)	29.934 ± 24.548	0.058 ± 0.047	0.021	0.027
Mediapipe Suavizado	Der	0.8	10.0	6.676 ± 3.126	0.669 ± 0.413	0.392	Débil (+)	0.536	Moderada (+)	28.161 ± 21.537	0.058 ± 0.047	0.062	0.027
Dataset Suavizado	Der	1.0	20.0	6.285 ± 4.265	0.795 ± 0.638	0.39	Débil (+)	0.219	Débil (+)	17.593 ± 13.861	0.053 ± 0.073	0.161	0.171
Mediapipe	Izq	0.5	25.0	8.448 ± 3.659	1.118 ± 0.765	0.388	Débil (+)	0.343	Débil (+)	38.45 ± 25.04	0.091 ± 0.087	0.17	0.408
Mediapipe Suavizado	Izq	0.6	15.0	8.308 ± 3.911	1.267 ± 0.947	0.383	Débil (+)	0.201	Débil (+)	33.914 ± 20.177	0.094 ± 0.108	0.335	0.696
Mediapipe Suavizado	Der	0.8	5.0	6.87 ± 3.211	0.743 ± 0.466	0.382	Débil (+)	0.479	Moderada (+)	29.188 ± 22.366	0.064 ± 0.053	0.044	0.023
Mediapipe Suavizado	Der	0.7	10.0	7.08 ± 3.349	0.843 ± 0.558	0.378	Débil (+)	0.386	Débil (+)	30.091 ± 22.773	0.07 ± 0.064	0.025	0.019
Dataset Suavizado	Der	0.1	20.0	7.129 ± 3.403	0.91 ± 0.497	0.373	Débil (+)	0.278	Débil (+)	29.57 ± 22.816	0.085 ± 0.057	0.036	0.027
Dataset Suavizado	Der	0.2	20.0	7.129 ± 3.403	0.91 ± 0.497	0.373	Débil (+)	0.278	Débil (+)	29.57 ± 22.816	0.085 ± 0.057	0.036	0.027
Dataset Suavizado	Der	0.3	20.0	7.129 ± 3.403	0.91 ± 0.497	0.373	Débil (+)	0.278	Débil (+)	29.57 ± 22.816	0.085 ± 0.057	0.036	0.027
Dataset Suavizado	Der	0.4	20.0	6.988 ± 3.308	0.851 ± 0.468	0.372	Débil (+)	0.312	Débil (+)	28.986 ± 22.396	0.079 ± 0.053	0.047	0.032
Mediapipe Suavizado	Der	0.7	5.0	7.249 ± 3.481	0.932 ± 0.633	0.371	Débil (+)	0.308	Débil (+)	30.823 ± 23.551	0.076 ± 0.072	0.021	0.016
Mediapipe Suavizado	Izq	0.4	20.0	8.561 ± 5.136	1.343 ± 1.066	0.371	Débil (+)	0.169	Muy Débil (+)	27.318 ± 15.89	0.091 ± 0.122	0.726	0.694
Mediapipe Suavizado	Der	0.6	10.0	7.477 ± 3.608	0.953 ± 0.635	0.368	Débil (+)	0.282	Débil (+)	31.815 ± 23.943	0.079 ± 0.072	0.014	0.011
Mediapipe Suavizado	Izq	0.1	20.0	8.575 ± 5.069	1.343 ± 1.066	0.366	Débil (+)	0.169	Muy Débil (+)	27.988 ± 16.919	0.091 ± 0.122	0.683	0.694
Mediapipe Suavizado	Izq	0.2	20.0	8.575 ± 5.069	1.343 ± 1.066	0.366	Débil (+)	0.169	Muy Débil (+)	27.988 ± 16.919	0.091 ± 0.122	0.683	0.694
Mediapipe Suavizado	Izq	0.3	20.0	8.575 ± 5.069	1.343 ± 1.066	0.366	Débil (+)	0.169	Muy Débil (+)	27.988 ± 16.919	0.091 ± 0.122	0.683	0.694
Mediapipe Suavizado	Der	0.6	5.0	7.684 ± 3.822	0.953 ± 0.635	0.361	Débil (+)	0.282	Débil (+)	32.67 ± 25.133	0.079 ± 0.072	0.01	0.011

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe Suavizado	Der	0.8	20.0	6.834 ± 4.366	0.819 ± 0.649	0.361	Débil (+)	0.22	Débil (+)	19.712 ± 13.545	0.056 ± 0.074	0.058	0.038
Mediapipe Suavizado	Izq	0.7	15.0	8.379 ± 4.075	1.333 ± 1.011	0.36	Débil (+)	0.137	Muy Débil (+)	32.61 ± 17.991	0.096 ± 0.115	0.365	0.862
Dataset	Der	0.8	25.0	5.884 ± 3.381	0.769 ± 0.585	0.359	Débil (+)	0.128	Muy Débil (+)	21.085 ± 17.867	0.056 ± 0.067	0.953	0.691
Dataset	Der	0.7	25.0	6.505 ± 3.42	0.716 ± 0.456	0.356	Débil (+)	0.309	Débil (+)	26.675 ± 23.501	0.061 ± 0.052	0.054	0.091
Mediapipe	Der	0.8	25.0	6.896 ± 4.544	0.835 ± 0.669	0.355	Débil (+)	0.221	Débil (+)	19.036 ± 13.508	0.056 ± 0.076	0.029	0.009
Mediapipe	Izq	0.6	25.0	8.384 ± 3.815	1.308 ± 1.001	0.353	Débil (+)	0.166	Muy Débil (+)	34.245 ± 17.942	0.094 ± 0.114	0.293	0.85
Mediapipe	Der	0.5	20.0	8.486 ± 4.11	1.039 ± 0.608	0.352	Débil (+)	0.421	Moderada (+)	36.771 ± 27.404	0.094 ± 0.069	0.001	0.002
Mediapipe Suavizado	Izq	0.5	20.0	8.719 ± 5.347	1.405 ± 1.125	0.349	Débil (+)	0.131	Muy Débil (+)	26.983 ± 15.614	0.094 ± 0.128	0.77	0.299
Dataset Suavizado	Der	0.8	10.0	7.739 ± 3.862	0.866 ± 0.495	0.348	Débil (+)	0.403	Moderada (+)	33.472 ± 27.659	0.079 ± 0.057	0.004	0.004
Dataset Suavizado	Der	0.9	25.0	6.897 ± 4.472	0.819 ± 0.649	0.348	Débil (+)	0.22	Débil (+)	19.496 ± 12.691	0.056 ± 0.074	0.058	0.038
Dataset Suavizado	Der	0.8	5.0	7.902 ± 4.028	0.925 ± 0.522	0.346	Débil (+)	0.368	Débil (+)	34.008 ± 28.168	0.085 ± 0.06	0.002	0.003
Mediapipe Suavizado	Izq	0.8	15.0	8.497 ± 4.468	1.318 ± 1.014	0.344	Débil (+)	0.16	Muy Débil (+)	30.967 ± 17.037	0.094 ± 0.116	0.465	0.874
Dataset Suavizado	Der	0.6	15.0	7.724 ± 3.843	0.925 ± 0.592	0.343	Débil (+)	0.372	Débil (+)	32.95 ± 26.809	0.079 ± 0.068	0.007	0.009
Mediapipe Suavizado	Der	0.5	10.0	7.968 ± 4.004	1.106 ± 0.685	0.342	Débil (+)	0.213	Débil (+)	33.702 ± 25.741	0.096 ± 0.078	0.009	0.006
Mediapipe Suavizado	Der	0.5	5.0	8.197 ± 4.215	1.082 ± 0.711	0.339	Débil (+)	0.333	Débil (+)	34.604 ± 26.68	0.091 ± 0.081	0.006	0.006
Mediapipe Suavizado	Izq	0.5	10.0	9.65 ± 4.623	1.23 ± 0.815	0.335	Débil (+)	0.228	Débil (+)	45.858 ± 35.488	0.102 ± 0.093	0.085	0.225
Mediapipe Suavizado	Izq	0.4	10.0	9.874 ± 4.814	1.209 ± 0.84	0.334	Débil (+)	0.278	Débil (+)	47.052 ± 37.013	0.096 ± 0.096	0.085	0.204
Mediapipe Suavizado	Der	0.4	10.0	8.279 ± 4.293	1.082 ± 0.711	0.332	Débil (+)	0.333	Débil (+)	34.877 ± 26.982	0.091 ± 0.081	0.005	0.006
Mediapipe Suavizado	Der	0.4	25.0	7.851 ± 4.824	0.843 ± 0.658	0.331	Débil (+)	0.222	Débil (+)	22.48 ± 13.886	0.058 ± 0.075	0.004	0.004
Mediapipe Suavizado	Der	0.5	25.0	7.961 ± 4.845	0.843 ± 0.658	0.331	Débil (+)	0.222	Débil (+)	22.945 ± 14.049	0.058 ± 0.075	0.002	0.004
Mediapipe Suavizado	Der	0.9	20.0	7.24 ± 4.69	0.851 ± 0.647	0.33	Débil (+)	0.192	Muy Débil (+)	20.373 ± 14.31	0.061 ± 0.074	0.031	0.009
Mediapipe Suavizado	Der	0.1	25.0	7.843 ± 4.786	0.843 ± 0.658	0.329	Débil (+)	0.222	Débil (+)	22.617 ± 13.669	0.058 ± 0.075	0.005	0.004
Mediapipe Suavizado	Der	0.2	25.0	7.843 ± 4.786	0.843 ± 0.658	0.329	Débil (+)	0.222	Débil (+)	22.617 ± 13.669	0.058 ± 0.075	0.005	0.004
Mediapipe Suavizado	Der	0.3	25.0	7.843 ± 4.786	0.843 ± 0.658	0.329	Débil (+)	0.222	Débil (+)	22.617 ± 13.669	0.058 ± 0.075	0.005	0.004
Mediapipe Suavizado	Izq	0.6	10.0	9.417 ± 4.462	1.23 ± 0.815	0.327	Débil (+)	0.228	Débil (+)	44.375 ± 33.755	0.102 ± 0.093	0.108	0.225
Mediapipe Suavizado	Izq	0.1	10.0	10.047 ± 5.013	1.02 ± 0.676	0.326	Débil (+)	0.517	Moderada (+)	47.891 ± 38.668	0.085 ± 0.077	0.075	0.165
Mediapipe Suavizado	Izq	0.2	10.0	10.047 ± 5.013	1.02 ± 0.676	0.326	Débil (+)	0.517	Moderada (+)	47.891 ± 38.668	0.085 ± 0.077	0.075	0.165

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe Suavizado	Izq	0.3	10.0	10.047 ± 5.013	1.02 ± 0.676	0.326	Débil (+)	0.517	Moderada (+)	47.891 ± 38.668	0.085 ± 0.077	0.075	0.165
Mediapipe Suavizado	Izq	0.5	5.0	9.857 ± 4.767	1.23 ± 0.815	0.323	Débil (+)	0.228	Débil (+)	47.164 ± 36.901	0.102 ± 0.093	0.085	0.225
Mediapipe Suavizado	Der	0.1	5.0	8.683 ± 4.589	1.088 ± 0.689	0.32	Débil (+)	0.329	Débil (+)	36.381 ± 28.276	0.094 ± 0.079	0.003	0.004
Mediapipe Suavizado	Der	0.2	5.0	8.683 ± 4.589	1.088 ± 0.689	0.32	Débil (+)	0.329	Débil (+)	36.381 ± 28.276	0.094 ± 0.079	0.003	0.004
Mediapipe Suavizado	Der	0.3	5.0	8.683 ± 4.589	1.088 ± 0.689	0.32	Débil (+)	0.329	Débil (+)	36.381 ± 28.276	0.094 ± 0.079	0.003	0.004
Mediapipe Suavizado	Der	0.1	10.0	8.479 ± 4.449	1.082 ± 0.711	0.318	Débil (+)	0.333	Débil (+)	35.546 ± 27.633	0.091 ± 0.081	0.005	0.006
Mediapipe Suavizado	Der	0.2	10.0	8.479 ± 4.449	1.082 ± 0.711	0.318	Débil (+)	0.333	Débil (+)	35.546 ± 27.633	0.091 ± 0.081	0.005	0.006
Mediapipe Suavizado	Der	0.3	10.0	8.479 ± 4.449	1.082 ± 0.711	0.318	Débil (+)	0.333	Débil (+)	35.546 ± 27.633	0.091 ± 0.081	0.005	0.006
Mediapipe Suavizado	Der	0.4	5.0	8.574 ± 4.568	1.082 ± 0.711	0.318	Débil (+)	0.333	Débil (+)	35.88 ± 28.204	0.091 ± 0.081	0.003	0.006
Mediapipe Suavizado	Izq	0.9	15.0	8.715 ± 4.976	1.333 ± 1.093	0.318	Débil (+)	0.206	Débil (+)	29.34 ± 16.137	0.085 ± 0.125	0.559	0.416
Mediapipe Suavizado	Izq	0.6	20.0	9.012 ± 5.712	1.4 ± 1.138	0.317	Débil (+)	0.149	Muy Débil (+)	26.636 ± 15.744	0.091 ± 0.13	0.815	0.202
Mediapipe	Izq	0.6	20.0	9.167 ± 4.463	1.209 ± 0.782	0.316	Débil (+)	0.242	Débil (+)	42.749 ± 32.516	0.102 ± 0.089	0.102	0.279
Mediapipe Suavizado	Izq	0.6	5.0	9.566 ± 4.581	1.23 ± 0.815	0.316	Débil (+)	0.228	Débil (+)	45.315 ± 34.852	0.102 ± 0.093	0.096	0.225
Mediapipe Suavizado	Izq	0.4	5.0	10.14 ± 4.995	1.094 ± 0.729	0.315	Débil (+)	0.426	Moderada (+)	48.489 ± 38.239	0.091 ± 0.083	0.062	0.184
Mediapipe	Izq	0.4	20.0	11.482 ± 5.998	1.262 ± 0.734	0.315	Débil (+)	0.41	Moderada (+)	56.451 ± 45.363	0.114 ± 0.084	0.01	0.017
Mediapipe Suavizado	Izq	0.8	5.0	8.917 ± 3.881	1.181 ± 0.828	0.315	Débil (+)	0.28	Débil (+)	40.997 ± 28.552	0.094 ± 0.095	0.161	0.517
Mediapipe	Der	0.9	20.0	6.479 ± 4.257	0.795 ± 0.638	0.315	Débil (+)	0.251	Débil (+)	18.933 ± 15.114	0.053 ± 0.073	0.243	0.068
Mediapipe Suavizado	Izq	0.7	10.0	9.134 ± 4.292	1.198 ± 0.825	0.315	Débil (+)	0.241	Débil (+)	42.068 ± 31.166	0.096 ± 0.094	0.136	0.389
Mediapipe Suavizado	Izq	0.8	10.0	8.856 ± 3.877	1.192 ± 0.844	0.314	Débil (+)	0.272	Débil (+)	40.184 ± 27.558	0.094 ± 0.096	0.189	0.537
Mediapipe Suavizado	Izq	0.9	10.0	8.656 ± 3.726	1.203 ± 0.859	0.314	Débil (+)	0.265	Débil (+)	38.038 ± 23.92	0.094 ± 0.098	0.209	0.557
Mediapipe Suavizado	Izq	0.9	5.0	8.704 ± 3.653	1.203 ± 0.859	0.313	Débil (+)	0.265	Débil (+)	38.912 ± 24.775	0.094 ± 0.098	0.209	0.557
Mediapipe Suavizado	Izq	0.1	5.0	10.281 ± 5.168	1.164 ± 0.775	0.311	Débil (+)	0.405	Moderada (+)	49.212 ± 39.772	0.096 ± 0.089	0.058	0.131
Mediapipe Suavizado	Izq	0.2	5.0	10.281 ± 5.168	1.164 ± 0.775	0.311	Débil (+)	0.405	Moderada (+)	49.212 ± 39.772	0.096 ± 0.089	0.058	0.131
Mediapipe Suavizado	Izq	0.3	5.0	10.281 ± 5.168	1.164 ± 0.775	0.311	Débil (+)	0.405	Moderada (+)	49.212 ± 39.772	0.096 ± 0.089	0.058	0.131
Mediapipe Suavizado	Der	0.6	25.0	8.252 ± 4.845	0.843 ± 0.658	0.311	Débil (+)	0.222	Débil (+)	24.365 ± 13.596	0.058 ± 0.075	0.001	0.004
Mediapipe	Izq	0.5	20.0	10.314 ± 5.312	1.088 ± 0.689	0.309	Débil (+)	0.456	Moderada (+)	49.971 ± 39.953	0.094 ± 0.079	0.044	0.042
Dataset Suavizado	Izq	1.0	10.0	8.88 ± 3.693	1.187 ± 0.748	0.308	Débil (+)	0.239	Débil (+)	41.988 ± 30.388	0.102 ± 0.085	0.144	0.247

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe Suavizado	Izq	0.7	5.0	9.262 ± 4.343	1.214 ± 0.82	0.307	Débil (+)	0.242	Débil (+)	43.146 ± 32.248	0.099 ± 0.094	0.129	0.348
Dataset	Der	0.6	25.0	8.43 ± 4.331	0.939 ± 0.547	0.306	Débil (+)	0.344	Débil (+)	36.096 ± 30.084	0.085 ± 0.062	0.001	0.001
Mediapipe	Izq	0.1	20.0	12.256 ± 6.677	1.381 ± 0.792	0.305	Débil (+)	0.416	Moderada (+)	59.559 ± 48.575	0.126 ± 0.09	0.005	0.008
Mediapipe	Izq	0.2	20.0	12.256 ± 6.677	1.381 ± 0.792	0.305	Débil (+)	0.416	Moderada (+)	59.559 ± 48.575	0.126 ± 0.09	0.005	0.008
Mediapipe	Izq	0.3	20.0	12.256 ± 6.677	1.381 ± 0.792	0.305	Débil (+)	0.416	Moderada (+)	59.559 ± 48.575	0.126 ± 0.09	0.005	0.008
Dataset Suavizado	Izq	0.9	15.0	8.768 ± 3.466	1.225 ± 0.836	0.305	Débil (+)	0.221	Débil (+)	40.318 ± 27.847	0.099 ± 0.095	0.17	0.433
Mediapipe Suavizado	Izq	1.0	5.0	8.641 ± 3.713	1.277 ± 0.912	0.305	Débil (+)	0.177	Muy Débil (+)	36.613 ± 21.013	0.099 ± 0.104	0.28	0.575
Dataset Suavizado	Der	0.5	15.0	8.551 ± 4.254	0.953 ± 0.571	0.302	Débil (+)	0.342	Débil (+)	36.46 ± 28.668	0.085 ± 0.065	0.002	0.004
Dataset Suavizado	Izq	1.0	15.0	8.634 ± 3.679	1.262 ± 0.915	0.3	Débil (+)	0.2	Débil (+)	36.767 ± 22.893	0.096 ± 0.104	0.267	0.584
Mediapipe	Der	0.8	15.0	6.854 ± 3.437	0.734 ± 0.484	0.299	Débil (+)	0.371	Débil (+)	28.847 ± 24.754	0.061 ± 0.055	0.05	0.044
Mediapipe	Der	0.4	20.0	10.451 ± 5.066	1.581 ± 0.792	0.297	Débil (+)	0.256	Débil (+)	44.569 ± 31.785	0.152 ± 0.09	0.0	0.0
Dataset Suavizado	Der	1.0	25.0	7.602 ± 4.822	0.819 ± 0.649	0.297	Débil (+)	0.251	Débil (+)	21.42 ± 13.148	0.056 ± 0.074	0.01	0.009
Mediapipe Suavizado	Izq	1.0	10.0	8.669 ± 3.807	1.343 ± 0.977	0.296	Débil (+)	0.095	Muy Débil (+)	36.16 ± 20.324	0.102 ± 0.112	0.293	0.584
Dataset Suavizado	Izq	0.9	10.0	9.471 ± 4.395	1.106 ± 0.613	0.295	Débil (+)	0.345	Débil (+)	46.051 ± 36.769	0.102 ± 0.07	0.08	0.19
Dataset Suavizado	Der	0.7	10.0	8.811 ± 4.6	0.993 ± 0.567	0.294	Débil (+)	0.399	Débil (+)	37.696 ± 30.865	0.091 ± 0.065	0.001	0.001
Mediapipe Suavizado	Der	0.7	25.0	8.526 ± 4.889	0.843 ± 0.658	0.293	Débil (+)	0.222	Débil (+)	25.561 ± 13.534	0.058 ± 0.075	0.001	0.004
Mediapipe Suavizado	Izq	0.4	25.0	9.795 ± 6.862	1.543 ± 1.276	0.293	Débil (+)	0.024	Muy Débil (+)	24.057 ± 17.436	0.096 ± 0.146	0.43	0.014
Mediapipe	Izq	0.7	25.0	8.863 ± 5.047	1.313 ± 1.068	0.288	Débil (+)	0.229	Débil (+)	30.187 ± 15.77	0.085 ± 0.122	0.559	0.416
Mediapipe Suavizado	Der	1.0	20.0	7.766 ± 4.762	0.843 ± 0.658	0.288	Débil (+)	0.222	Débil (+)	22.761 ± 13.709	0.058 ± 0.075	0.003	0.004
Mediapipe Suavizado	Izq	0.1	25.0	9.789 ± 6.777	1.543 ± 1.276	0.287	Débil (+)	0.024	Muy Débil (+)	24.664 ± 17.469	0.096 ± 0.146	0.465	0.014
Mediapipe Suavizado	Izq	0.2	25.0	9.789 ± 6.777	1.543 ± 1.276	0.287	Débil (+)	0.024	Muy Débil (+)	24.664 ± 17.469	0.096 ± 0.146	0.465	0.014
Mediapipe Suavizado	Izq	0.3	25.0	9.789 ± 6.777	1.543 ± 1.276	0.287	Débil (+)	0.024	Muy Débil (+)	24.664 ± 17.469	0.096 ± 0.146	0.465	0.014
Mediapipe	Izq	0.7	20.0	8.76 ± 3.9	1.298 ± 0.94	0.286	Débil (+)	0.133	Muy Débil (+)	38.278 ± 24.484	0.099 ± 0.107	0.209	0.449
Mediapipe Suavizado	Izq	0.7	20.0	9.341 ± 6.189	1.433 ± 1.159	0.286	Débil (+)	0.122	Muy Débil (+)	25.686 ± 16.384	0.094 ± 0.132	0.93	0.125
Dataset Suavizado	Izq	0.9	20.0	8.768 ± 4.208	1.272 ± 0.93	0.285	Débil (+)	0.214	Débil (+)	34.172 ± 20.297	0.096 ± 0.106	0.365	0.862
Mediapipe Suavizado	Izq	0.5	25.0	9.952 ± 7.029	1.543 ± 1.276	0.285	Débil (+)	0.024	Muy Débil (+)	23.939 ± 17.56	0.096 ± 0.146	0.321	0.014
Dataset Suavizado	Der	0.7	5.0	9.081 ± 4.758	1.203 ± 0.742	0.284	Débil (+)	0.255	Débil (+)	38.822 ± 31.554	0.105 ± 0.085	0.001	0.0

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Dataset Suavizado	Izq	1.0	5.0	9.118 ± 3.868	1.219 ± 0.734	0.284	Débil (+)	0.19	Muy Débil (+)	43.392 ± 31.9	0.108 ± 0.084	0.122	0.236
Dataset	Izq	1.0	10.0	8.652 ± 3.864	1.246 ± 0.867	0.284	Débil (+)	0.189	Muy Débil (+)	37.908 ± 25.387	0.099 ± 0.099	0.17	0.353
Dataset	Der	0.8	20.0	6.54 ± 3.263	0.725 ± 0.437	0.283	Débil (+)	0.265	Débil (+)	26.834 ± 22.813	0.064 ± 0.05	0.153	0.186
Dataset Suavizado	Izq	0.8	20.0	8.815 ± 3.823	1.257 ± 0.933	0.28	Débil (+)	0.22	Débil (+)	37.386 ± 24.232	0.094 ± 0.106	0.307	0.718
Dataset Suavizado	Izq	0.7	20.0	8.924 ± 3.719	1.293 ± 0.907	0.279	Débil (+)	0.154	Muy Débil (+)	39.86 ± 27.686	0.102 ± 0.104	0.243	0.566
Dataset	Der	0.9	25.0	6.562 ± 4.458	0.819 ± 0.649	0.278	Débil (+)	0.188	Muy Débil (+)	18.615 ± 16.338	0.056 ± 0.074	0.243	0.111
Mediapipe Suavizado	Izq	1.0	15.0	9.131 ± 5.623	1.433 ± 1.159	0.276	Débil (+)	0.122	Muy Débil (+)	28.05 ± 16.287	0.094 ± 0.132	0.683	0.125
Dataset Suavizado	Izq	0.8	15.0	9.234 ± 3.842	1.1 ± 0.64	0.274	Débil (+)	0.374	Débil (+)	44.064 ± 32.981	0.099 ± 0.073	0.129	0.287
Mediapipe	Der	0.1	20.0	11.532 ± 5.45	1.658 ± 0.897	0.274	Débil (+)	0.23	Débil (+)	48.791 ± 33.351	0.155 ± 0.102	0.0	0.0
Mediapipe	Der	0.2	20.0	11.532 ± 5.45	1.658 ± 0.897	0.274	Débil (+)	0.23	Débil (+)	48.791 ± 33.351	0.155 ± 0.102	0.0	0.0
Mediapipe	Der	0.3	20.0	11.532 ± 5.45	1.658 ± 0.897	0.274	Débil (+)	0.23	Débil (+)	48.791 ± 33.351	0.155 ± 0.102	0.0	0.0
Mediapipe	Der	0.9	15.0	6.304 ± 3.405	0.786 ± 0.559	0.273	Débil (+)	0.113	Muy Débil (+)	24.144 ± 20.305	0.061 ± 0.064	0.599	0.522
Mediapipe	Der	0.7	15.0	8.197 ± 4.163	0.946 ± 0.521	0.272	Débil (+)	0.277	Débil (+)	35.401 ± 29.523	0.088 ± 0.059	0.002	0.002
Dataset Suavizado	Izq	1.0	20.0	8.984 ± 5.063	1.386 ± 1.101	0.272	Débil (+)	0.154	Muy Débil (+)	30.872 ± 19.149	0.094 ± 0.126	0.54	0.299
Dataset	Izq	0.8	25.0	8.675 ± 3.896	1.323 ± 0.998	0.272	Débil (+)	0.139	Muy Débil (+)	36.089 ± 21.186	0.096 ± 0.114	0.231	0.728
Dataset Suavizado	Der	0.4	15.0	9.246 ± 4.792	1.058 ± 0.673	0.271	Débil (+)	0.37	Débil (+)	39.071 ± 30.901	0.091 ± 0.077	0.001	0.003
Dataset	Der	0.9	15.0	6.377 ± 3.082	0.725 ± 0.437	0.271	Débil (+)	0.256	Débil (+)	26.283 ± 22.326	0.064 ± 0.05	0.243	0.1
Dataset Suavizado	Der	0.1	15.0	9.478 ± 4.942	1.203 ± 0.742	0.27	Débil (+)	0.296	Débil (+)	39.93 ± 31.289	0.105 ± 0.085	0.001	0.001
Dataset Suavizado	Der	0.2	15.0	9.478 ± 4.942	1.203 ± 0.742	0.27	Débil (+)	0.296	Débil (+)	39.93 ± 31.289	0.105 ± 0.085	0.001	0.001
Dataset Suavizado	Der	0.3	15.0	9.478 ± 4.942	1.203 ± 0.742	0.27	Débil (+)	0.296	Débil (+)	39.93 ± 31.289	0.105 ± 0.085	0.001	0.001
Dataset Suavizado	Izq	0.8	10.0	10.232 ± 5.081	1.13 ± 0.654	0.269	Débil (+)	0.347	Débil (+)	50.657 ± 42.24	0.102 ± 0.075	0.058	0.105
Dataset	Izq	0.9	10.0	9.458 ± 3.873	1.136 ± 0.699	0.266	Débil (+)	0.268	Débil (+)	47.169 ± 35.382	0.099 ± 0.08	0.102	0.119
Dataset	Izq	1.0	5.0	8.824 ± 3.899	1.262 ± 0.862	0.266	Débil (+)	0.147	Muy Débil (+)	39.734 ± 27.757	0.102 ± 0.098	0.161	0.257
Mediapipe Suavizado	Der	0.8	25.0	8.936 ± 4.942	0.896 ± 0.705	0.265	Débil (+)	0.137	Muy Débil (+)	27.388 ± 13.336	0.061 ± 0.08	0.0	0.001
Dataset	Der	0.9	20.0	6.22 ± 3.563	0.769 ± 0.585	0.265	Débil (+)	0.093	Muy Débil (+)	21.952 ± 17.617	0.056 ± 0.067	0.907	0.626
Dataset	Izq	0.8	20.0	8.968 ± 3.707	1.209 ± 0.782	0.264	Débil (+)	0.206	Débil (+)	42.022 ± 28.762	0.102 ± 0.089	0.144	0.19
Dataset	Izq	1.0	15.0	8.734 ± 4.409	1.343 ± 1.066	0.264	Débil (+)	0.169	Muy Débil (+)	33.869 ± 19.96	0.091 ± 0.122	0.267	0.694

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe Suavizado	Izq	0.8	20.0	9.694 ± 6.596	1.513 ± 1.22	0.264	Débil (+)	0.048	Muy Débil (+)	25.166 ± 17.045	0.099 ± 0.139	0.726	0.034
Dataset	Izq	0.9	15.0	8.883 ± 3.877	1.17 ± 0.754	0.263	Débil (+)	0.247	Débil (+)	40.689 ± 28.904	0.099 ± 0.086	0.122	0.181
Dataset Suavizado	Izq	0.9	5.0	9.788 ± 4.535	1.112 ± 0.583	0.262	Débil (+)	0.326	Débil (+)	47.914 ± 38.161	0.105 ± 0.066	0.08	0.137
Mediapipe	Izq	0.7	15.0	9.985 ± 4.921	1.241 ± 0.769	0.262	Débil (+)	0.195	Muy Débil (+)	48.734 ± 39.161	0.108 ± 0.088	0.066	0.069
Dataset Suavizado	Izq	0.6	20.0	9.186 ± 3.897	1.313 ± 0.881	0.26	Débil (+)	0.109	Muy Débil (+)	42.253 ± 31.266	0.108 ± 0.101	0.189	0.441
Dataset Suavizado	Izq	0.7	15.0	9.749 ± 4.521	1.106 ± 0.613	0.259	Débil (+)	0.345	Débil (+)	47.219 ± 38.444	0.102 ± 0.07	0.08	0.19
Mediapipe Suavizado	Izq	0.6	25.0	10.27 ± 7.259	1.573 ± 1.293	0.257	Débil (+)	0.0	Muy Débil (+)	24.453 ± 17.982	0.099 ± 0.148	0.22	0.004
Dataset	Izq	0.8	15.0	9.905 ± 4.456	1.198 ± 0.697	0.255	Débil (+)	0.197	Muy Débil (+)	49.659 ± 38.232	0.108 ± 0.08	0.07	0.088
Mediapipe	Izq	0.6	15.0	11.464 ± 5.863	1.251 ± 0.786	0.253	Débil (+)	0.361	Débil (+)	57.163 ± 46.245	0.108 ± 0.09	0.019	0.023
Dataset	Izq	0.9	20.0	8.752 ± 4.282	1.313 ± 1.029	0.252	Débil (+)	0.179	Muy Débil (+)	34.928 ± 20.893	0.091 ± 0.117	0.243	1.0
Dataset Suavizado	Der	0.6	10.0	10.067 ± 5.267	1.328 ± 0.809	0.244	Débil (+)	0.175	Muy Débil (+)	42.895 ± 33.947	0.117 ± 0.092	0.0	0.0
Mediapipe Suavizado	Der	0.9	25.0	9.406 ± 5.047	0.896 ± 0.705	0.242	Débil (+)	0.137	Muy Débil (+)	29.329 ± 13.66	0.061 ± 0.08	0.0	0.001
Dataset Suavizado	Izq	0.6	15.0	10.286 ± 4.966	1.147 ± 0.647	0.24	Débil (+)	0.317	Débil (+)	50.432 ± 42.181	0.105 ± 0.074	0.062	0.165
Mediapipe	Izq	0.8	15.0	9.194 ± 4.036	1.235 ± 0.852	0.239	Débil (+)	0.197	Muy Débil (+)	42.617 ± 30.397	0.099 ± 0.097	0.136	0.341
Dataset Suavizado	Izq	0.7	10.0	11.112 ± 5.871	1.1 ± 0.559	0.238	Débil (+)	0.48	Moderada (+)	55.243 ± 47.2	0.105 ± 0.064	0.033	0.033
Dataset	Izq	0.9	5.0	9.869 ± 4.082	1.164 ± 0.712	0.238	Débil (+)	0.248	Débil (+)	49.81 ± 37.654	0.102 ± 0.081	0.085	0.096
Dataset Suavizado	Izq	0.8	5.0	10.615 ± 5.261	1.136 ± 0.626	0.237	Débil (+)	0.329	Débil (+)	52.777 ± 43.749	0.105 ± 0.071	0.058	0.073
Dataset	Izq	0.7	25.0	9.254 ± 3.862	1.225 ± 0.776	0.237	Débil (+)	0.186	Muy Débil (+)	43.709 ± 31.181	0.105 ± 0.089	0.122	0.242
Dataset Suavizado	Izq	0.5	20.0	9.57 ± 4.201	1.246 ± 0.809	0.237	Débil (+)	0.181	Muy Débil (+)	44.804 ± 34.53	0.105 ± 0.092	0.136	0.398
Dataset	Der	0.8	15.0	7.79 ± 4.054	0.967 ± 0.593	0.237	Débil (+)	0.113	Muy Débil (+)	33.404 ± 29.897	0.085 ± 0.068	0.004	0.002
Dataset	Der	0.7	20.0	8.369 ± 4.135	1.039 ± 0.675	0.237	Débil (+)	0.069	Muy Débil (+)	36.082 ± 29.802	0.088 ± 0.077	0.002	0.001
Mediapipe	Der	0.9	25.0	8.374 ± 4.986	0.925 ± 0.742	0.234	Débil (+)	0.08	Muy Débil (+)	24.655 ± 12.452	0.061 ± 0.085	0.001	0.001
Mediapipe Suavizado	Izq	0.9	20.0	10.116 ± 7.125	1.573 ± 1.293	0.234	Débil (+)	0.0	Muy Débil (+)	24.496 ± 18.433	0.099 ± 0.148	0.43	0.004
Dataset Suavizado	Der	0.6	5.0	10.363 ± 5.43	1.46 ± 0.889	0.233	Débil (+)	0.161	Muy Débil (+)	44.123 ± 34.665	0.129 ± 0.102	0.0	0.0
Dataset	Izq	0.8	10.0	11.226 ± 5.443	1.039 ± 0.528	0.23	Débil (+)	0.482	Moderada (+)	57.494 ± 45.956	0.099 ± 0.06	0.031	0.04
Mediapipe	Der	0.6	15.0	10.174 ± 5.251	1.386 ± 0.762	0.229	Débil (+)	0.211	Débil (+)	43.734 ± 34.955	0.129 ± 0.087	0.0	0.0
Mediapipe	Izq	0.5	15.0	13.587 ± 7.052	1.5 ± 0.849	0.226	Débil (+)	0.366	Débil (+)	67.923 ± 53.831	0.137 ± 0.097	0.001	0.004

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe	Izq	0.8	20.0	9.005 ± 4.716	1.323 ± 1.041	0.226	Débil (+)	0.193	Muy Débil (+)	33.815 ± 18.953	0.091 ± 0.119	0.307	0.694
Mediapipe	Izq	0.4	15.0	15.486 ± 7.942	1.926 ± 1.175	0.224	Débil (+)	0.336	Débil (+)	76.807 ± 58.68	0.17 ± 0.134	0.0	0.001
Dataset	Izq	0.9	25.0	9.22 ± 5.564	1.419 ± 1.161	0.224	Débil (+)	0.126	Muy Débil (+)	29.952 ± 18.524	0.091 ± 0.133	0.559	0.202
Dataset	Der	0.9	10.0	7.217 ± 3.393	0.896 ± 0.469	0.224	Débil (+)	0.025	Muy Débil (+)	31.246 ± 27.077	0.085 ± 0.054	0.031	0.016
Mediapipe Suavizado	Izq	0.7	25.0	10.743 ± 7.608	1.573 ± 1.293	0.224	Débil (+)	0.0	Muy Débil (+)	25.178 ± 18.932	0.099 ± 0.148	0.108	0.004
Dataset Suavizado	Izq	0.4	20.0	9.9 ± 4.711	1.251 ± 0.786	0.22	Débil (+)	0.192	Muy Débil (+)	46.216 ± 37.911	0.108 ± 0.09	0.122	0.338
Mediapipe Suavizado	Der	1.0	25.0	9.909 ± 5.065	0.896 ± 0.705	0.218	Débil (+)	0.137	Muy Débil (+)	31.642 ± 13.337	0.061 ± 0.08	0.0	0.001
Mediapipe	Izq	0.1	15.0	16.566 ± 8.075	2.146 ± 1.185	0.217	Débil (+)	0.307	Débil (+)	81.962 ± 60.499	0.199 ± 0.135	0.0	0.0
Mediapipe	Izq	0.2	15.0	16.566 ± 8.075	2.146 ± 1.185	0.217	Débil (+)	0.307	Débil (+)	81.962 ± 60.499	0.199 ± 0.135	0.0	0.0
Mediapipe	Izq	0.3	15.0	16.566 ± 8.075	2.146 ± 1.185	0.217	Débil (+)	0.307	Débil (+)	81.962 ± 60.499	0.199 ± 0.135	0.0	0.0
Dataset	Izq	1.0	20.0	9.357 ± 5.776	1.464 ± 1.216	0.216	Débil (+)	0.096	Muy Débil (+)	29.363 ± 17.601	0.091 ± 0.139	0.64	0.07
Dataset Suavizado	Izq	0.6	10.0	11.928 ± 6.645	1.214 ± 0.688	0.215	Débil (+)	0.415	Moderada (+)	59.055 ± 51.91	0.111 ± 0.079	0.014	0.02
Dataset Suavizado	Der	0.5	10.0	11.108 ± 5.651	1.646 ± 0.915	0.214	Débil (+)	0.165	Muy Débil (+)	47.366 ± 35.637	0.152 ± 0.104	0.0	0.0
Dataset Suavizado	Izq	0.9	25.0	9.622 ± 5.926	1.522 ± 1.191	0.214	Débil (+)	0.009	Muy Débil (+)	29.72 ± 19.426	0.105 ± 0.136	0.884	0.125
Dataset Suavizado	Izq	0.7	5.0	11.454 ± 6.092	1.17 ± 0.607	0.213	Débil (+)	0.428	Moderada (+)	56.945 ± 48.648	0.111 ± 0.069	0.031	0.028
Dataset Suavizado	Izq	0.7	25.0	9.297 ± 4.804	1.338 ± 0.994	0.212	Débil (+)	0.134	Muy Débil (+)	34.562 ± 21.703	0.099 ± 0.114	0.43	0.874
Mediapipe	Der	0.9	10.0	7.296 ± 3.867	0.835 ± 0.514	0.212	Débil (+)	0.095	Muy Débil (+)	30.301 ± 27.277	0.073 ± 0.059	0.025	0.01
Dataset	Der	0.5	25.0	11.615 ± 5.565	1.564 ± 0.846	0.209	Débil (+)	0.196	Muy Débil (+)	50.136 ± 37.119	0.146 ± 0.097	0.0	0.0
Dataset Suavizado	Izq	0.8	25.0	9.444 ± 5.33	1.423 ± 1.107	0.209	Débil (+)	0.107	Muy Débil (+)	32.278 ± 20.277	0.099 ± 0.126	0.62	0.299
Dataset Suavizado	Izq	0.6	25.0	9.282 ± 4.471	1.323 ± 0.998	0.208	Débil (+)	0.139	Muy Débil (+)	36.694 ± 23.859	0.096 ± 0.114	0.381	0.728
Mediapipe	Izq	0.8	25.0	9.801 ± 6.336	1.577 ± 1.28	0.208	Débil (+)	-0.019	Muy Débil (-)	27.924 ± 15.698	0.102 ± 0.146	0.953	0.014
Dataset Suavizado	Izq	0.5	15.0	10.957 ± 5.592	1.277 ± 0.724	0.207	Débil (+)	0.205	Débil (+)	53.737 ± 46.043	0.117 ± 0.083	0.047	0.133
Mediapipe Suavizado	Izq	1.0	20.0	10.593 ± 7.543	1.573 ± 1.293	0.207	Débil (+)	0.0	Muy Débil (+)	24.759 ± 19.389	0.099 ± 0.148	0.22	0.004
Dataset Suavizado	Izq	0.1	20.0	10.098 ± 4.963	1.136 ± 0.699	0.206	Débil (+)	0.353	Débil (+)	47.17 ± 39.93	0.099 ± 0.08	0.108	0.3
Dataset Suavizado	Izq	0.2	20.0	10.098 ± 4.963	1.136 ± 0.699	0.206	Débil (+)	0.353	Débil (+)	47.17 ± 39.93	0.099 ± 0.08	0.108	0.3
Dataset Suavizado	Izq	0.3	20.0	10.098 ± 4.963	1.136 ± 0.699	0.206	Débil (+)	0.353	Débil (+)	47.17 ± 39.93	0.099 ± 0.08	0.108	0.3
Mediapipe	Izq	0.8	10.0	10.605 ± 5.297	1.147 ± 0.647	0.206	Débil (+)	0.332	Débil (+)	52.488 ± 42.474	0.105 ± 0.074	0.062	0.058

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Dataset Suavizado	Der	0.5	5.0	11.45 ± 5.884	1.662 ± 0.862	0.205	Débil (+)	0.173	Muy Débil (+)	48.642 ± 36.546	0.158 ± 0.098	0.0	0.0
Dataset	Izq	0.8	5.0	11.852 ± 5.901	1.153 ± 0.525	0.203	Débil (+)	0.396	Débil (+)	60.685 ± 48.751	0.114 ± 0.06	0.023	0.028
Mediapipe	Der	1.0	20.0	7.736 ± 5.226	0.925 ± 0.742	0.203	Débil (+)	0.08	Muy Débil (+)	20.919 ± 16.057	0.061 ± 0.085	0.023	0.001
Mediapipe	Der	1.0	15.0	6.644 ± 4.11	0.803 ± 0.628	0.199	Muy Débil (+)	0.107	Muy Débil (+)	21.716 ± 18.437	0.056 ± 0.072	0.483	0.751
Mediapipe	Der	0.8	10.0	8.912 ± 4.539	1.118 ± 0.634	0.197	Muy Débil (+)	0.07	Muy Débil (+)	38.402 ± 32.192	0.102 ± 0.072	0.001	0.001
Mediapipe	Izq	0.7	10.0	12.324 ± 6.17	1.272 ± 0.752	0.195	Muy Débil (+)	0.328	Débil (+)	62.391 ± 50.117	0.114 ± 0.086	0.012	0.019
Dataset	Der	0.6	20.0	11.075 ± 5.098	1.53 ± 0.782	0.195	Muy Débil (+)	0.203	Débil (+)	48.516 ± 35.968	0.146 ± 0.089	0.0	0.0
Dataset Suavizado	Izq	1.0	25.0	10.068 ± 6.563	1.509 ± 1.234	0.194	Muy Débil (+)	0.067	Muy Débil (+)	28.285 ± 18.481	0.096 ± 0.141	0.704	0.014
Mediapipe Suavizado	Izq	0.8	25.0	11.283 ± 7.922	1.573 ± 1.293	0.193	Muy Débil (+)	0.0	Muy Débil (+)	26.748 ± 19.369	0.099 ± 0.148	0.038	0.004
Mediapipe	Izq	0.9	15.0	9.144 ± 4.47	1.323 ± 0.998	0.19	Muy Débil (+)	0.139	Muy Débil (+)	37.054 ± 23.244	0.096 ± 0.114	0.243	0.728
Dataset Suavizado	Der	0.4	10.0	11.973 ± 6.083	1.74 ± 0.924	0.189	Muy Débil (+)	0.146	Muy Débil (+)	50.803 ± 37.526	0.164 ± 0.106	0.0	0.0
Dataset Suavizado	Izq	0.5	25.0	9.436 ± 4.296	1.352 ± 0.99	0.189	Muy Débil (+)	0.093	Muy Débil (+)	38.966 ± 25.766	0.102 ± 0.113	0.321	0.707
Dataset	Der	0.9	5.0	7.616 ± 3.637	0.896 ± 0.469	0.189	Muy Débil (+)	0.025	Muy Débil (+)	33.208 ± 29.312	0.085 ± 0.054	0.01	0.016
Dataset	Izq	1.0	25.0	10.172 ± 6.916	1.573 ± 1.293	0.189	Muy Débil (+)	0.0	Muy Débil (+)	26.901 ± 18.541	0.099 ± 0.148	0.683	0.004
Dataset Suavizado	Izq	0.4	25.0	9.518 ± 4.312	1.303 ± 0.921	0.188	Muy Débil (+)	0.152	Muy Débil (+)	40.242 ± 28.063	0.102 ± 0.105	0.293	0.685
Dataset Suavizado	Izq	0.6	5.0	12.331 ± 6.88	1.298 ± 0.68	0.187	Muy Débil (+)	0.364	Débil (+)	61.092 ± 53.45	0.123 ± 0.078	0.014	0.013
Dataset	Der	1.0	15.0	6.609 ± 3.912	0.835 ± 0.626	0.187	Muy Débil (+)	0.035	Muy Débil (+)	22.476 ± 18.3	0.061 ± 0.071	0.683	0.751
Dataset	Der	1.0	10.0	6.583 ± 3.29	0.811 ± 0.568	0.186	Muy Débil (+)	0.044	Muy Débil (+)	25.613 ± 20.606	0.064 ± 0.065	0.64	0.603
Mediapipe	Der	0.5	15.0	13.01 ± 5.791	1.899 ± 0.877	0.185	Muy Débil (+)	0.172	Muy Débil (+)	56.484 ± 38.25	0.187 ± 0.1	0.0	0.0
Dataset Suavizado	Der	0.1	10.0	12.321 ± 6.133	1.784 ± 0.924	0.185	Muy Débil (+)	0.126	Muy Débil (+)	52.348 ± 37.881	0.17 ± 0.106	0.0	0.0
Dataset Suavizado	Der	0.2	10.0	12.321 ± 6.133	1.784 ± 0.924	0.185	Muy Débil (+)	0.126	Muy Débil (+)	52.348 ± 37.881	0.17 ± 0.106	0.0	0.0
Dataset Suavizado	Der	0.3	10.0	12.321 ± 6.133	1.784 ± 0.924	0.185	Muy Débil (+)	0.126	Muy Débil (+)	52.348 ± 37.881	0.17 ± 0.106	0.0	0.0
Mediapipe	Izq	0.9	10.0	9.605 ± 4.08	1.262 ± 0.734	0.185	Muy Débil (+)	0.113	Muy Débil (+)	45.548 ± 32.487	0.114 ± 0.084	0.115	0.168
Mediapipe	Der	1.0	10.0	6.659 ± 3.727	0.795 ± 0.544	0.185	Muy Débil (+)	0.106	Muy Débil (+)	25.303 ± 22.472	0.064 ± 0.062	0.483	0.447
Mediapipe Suavizado	Izq	0.9	25.0	11.597 ± 8.124	1.573 ± 1.293	0.185	Muy Débil (+)	0.0	Muy Débil (+)	27.525 ± 20.143	0.099 ± 0.148	0.018	0.004
Dataset	Der	1.0	20.0	7.133 ± 5.057	0.903 ± 0.734	0.184	Muy Débil (+)	0.033	Muy Débil (+)	19.102 ± 18.13	0.058 ± 0.084	0.153	0.068
Dataset Suavizado	Izq	0.1	15.0	11.708 ± 6.356	1.376 ± 0.82	0.183	Muy Débil (+)	0.272	Débil (+)	57.024 ± 50.754	0.123 ± 0.094	0.031	0.038

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Dataset Suavizado	Izq	0.2	15.0	11.708 ± 6.356	1.376 ± 0.82	0.183	Muy Débil (+)	0.272	Débil (+)	57.024 ± 50.754	0.123 ± 0.094	0.031	0.038
Dataset Suavizado	Izq	0.3	15.0	11.708 ± 6.356	1.376 ± 0.82	0.183	Muy Débil (+)	0.272	Débil (+)	57.024 ± 50.754	0.123 ± 0.094	0.031	0.038
Dataset Suavizado	Izq	0.4	15.0	11.561 ± 6.152	1.282 ± 0.693	0.183	Muy Débil (+)	0.234	Débil (+)	56.395 ± 49.389	0.12 ± 0.079	0.033	0.073
Dataset	Izq	0.7	20.0	10.508 ± 4.725	1.235 ± 0.647	0.183	Muy Débil (+)	0.11	Muy Débil (+)	52.616 ± 40.112	0.117 ± 0.074	0.07	0.092
Dataset Suavizado	Izq	0.5	10.0	12.846 ± 7.28	1.376 ± 0.82	0.182	Muy Débil (+)	0.342	Débil (+)	63.342 ± 55.747	0.123 ± 0.094	0.006	0.009
Dataset	Der	0.7	15.0	10.589 ± 5.009	1.46 ± 0.633	0.182	Muy Débil (+)	0.159	Muy Débil (+)	46.55 ± 36.263	0.146 ± 0.072	0.0	0.0
Dataset	Izq	0.6	25.0	10.848 ± 5.308	1.136 ± 0.538	0.181	Muy Débil (+)	0.333	Débil (+)	53.945 ± 43.878	0.111 ± 0.061	0.062	0.092
Mediapipe	Der	0.9	5.0	7.792 ± 4.023	0.819 ± 0.488	0.181	Muy Débil (+)	0.219	Débil (+)	32.808 ± 28.573	0.073 ± 0.056	0.008	0.003
Dataset Suavizado	Der	0.4	5.0	12.34 ± 6.224	1.784 ± 0.924	0.18	Muy Débil (+)	0.126	Muy Débil (+)	52.393 ± 38.407	0.17 ± 0.106	0.0	0.0
Mediapipe	Izq	0.8	5.0	11.155 ± 5.677	1.164 ± 0.638	0.179	Muy Débil (+)	0.309	Débil (+)	55.552 ± 45.71	0.108 ± 0.073	0.05	0.055
Dataset	Der	1.0	25.0	8.081 ± 5.461	0.967 ± 0.793	0.179	Muy Débil (+)	-0.004	Muy Débil (-)	21.696 ± 16.421	0.061 ± 0.091	0.005	0.001
Dataset	Der	1.0	5.0	6.67 ± 3.248	0.786 ± 0.559	0.178	Muy Débil (+)	0.067	Muy Débil (+)	26.644 ± 22.105	0.061 ± 0.064	0.43	0.289
Dataset Suavizado	Der	0.1	5.0	12.661 ± 6.292	1.799 ± 0.862	0.177	Muy Débil (+)	0.18	Muy Débil (+)	53.729 ± 38.651	0.175 ± 0.098	0.0	0.0
Dataset Suavizado	Der	0.2	5.0	12.661 ± 6.292	1.799 ± 0.862	0.177	Muy Débil (+)	0.18	Muy Débil (+)	53.729 ± 38.651	0.175 ± 0.098	0.0	0.0
Dataset Suavizado	Der	0.3	5.0	12.661 ± 6.292	1.799 ± 0.862	0.177	Muy Débil (+)	0.18	Muy Débil (+)	53.729 ± 38.651	0.175 ± 0.098	0.0	0.0
Mediapipe	Izq	0.6	10.0	14.563 ± 7.308	1.732 ± 0.819	0.176	Muy Débil (+)	0.248	Débil (+)	73.717 ± 56.974	0.17 ± 0.093	0.001	0.001
Mediapipe	Der	1.0	5.0	6.825 ± 3.747	0.819 ± 0.552	0.176	Muy Débil (+)	0.062	Muy Débil (+)	26.605 ± 23.732	0.067 ± 0.063	0.267	0.379
Mediapipe	Izq	0.9	5.0	9.928 ± 4.477	1.187 ± 0.678	0.171	Muy Débil (+)	0.209	Débil (+)	47.654 ± 36.143	0.108 ± 0.077	0.09	0.157
Mediapipe	Der	0.7	10.0	10.965 ± 5.06	1.564 ± 0.758	0.171	Muy Débil (+)	0.07	Muy Débil (+)	48.158 ± 36.264	0.152 ± 0.087	0.0	0.0
Mediapipe	Der	1.0	25.0	9.754 ± 5.355	0.973 ± 0.782	0.171	Muy Débil (+)	0.0	Muy Débil (+)	30.036 ± 12.949	0.064 ± 0.089	0.0	0.0
Dataset Suavizado	Izq	0.4	10.0	13.514 ± 7.72	1.487 ± 0.863	0.169	Muy Débil (+)	0.313	Débil (+)	66.411 ± 58.559	0.135 ± 0.099	0.003	0.006
Dataset	Der	0.8	10.0	9.393 ± 4.767	1.136 ± 0.699	0.169	Muy Débil (+)	0.167	Muy Débil (+)	40.933 ± 34.99	0.099 ± 0.08	0.001	0.0
Mediapipe	Izq	0.7	5.0	13.085 ± 6.545	1.496 ± 0.711	0.168	Muy Débil (+)	0.263	Débil (+)	66.507 ± 53.205	0.146 ± 0.081	0.006	0.007
Dataset Suavizado	Izq	0.1	25.0	9.681 ± 4.469	1.303 ± 0.921	0.168	Muy Débil (+)	0.152	Muy Débil (+)	41.101 ± 29.979	0.102 ± 0.105	0.293	0.685
Dataset Suavizado	Izq	0.2	25.0	9.681 ± 4.469	1.303 ± 0.921	0.168	Muy Débil (+)	0.152	Muy Débil (+)	41.101 ± 29.979	0.102 ± 0.105	0.293	0.685
Dataset Suavizado	Izq	0.3	25.0	9.681 ± 4.469	1.303 ± 0.921	0.168	Muy Débil (+)	0.152	Muy Débil (+)	41.101 ± 29.979	0.102 ± 0.105	0.293	0.685
Mediapipe	Der	0.8	5.0	9.602 ± 4.676	1.251 ± 0.634	0.167	Muy Débil (+)	0.1	Muy Débil (+)	41.786 ± 33.762	0.12 ± 0.072	0.0	0.001

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe Suavizado	Izq	1.0	25.0	12.093 ± 8.33	1.573 ± 1.293	0.163	Muy Débil (+)	0.0	Muy Débil (+)	29.477 ± 20.187	0.099 ± 0.148	0.005	0.004
Dataset	Izq	0.7	15.0	12.325 ± 6.397	1.288 ± 0.742	0.162	Muy Débil (+)	0.309	Débil (+)	62.533 ± 51.163	0.117 ± 0.085	0.019	0.018
Dataset Suavizado	Izq	0.5	5.0	13.265 ± 7.489	1.482 ± 0.892	0.16	Muy Débil (+)	0.278	Débil (+)	65.611 ± 57.271	0.132 ± 0.102	0.004	0.007
Dataset	Der	0.4	25.0	15.08 ± 6.34	2.368 ± 1.083	0.16	Muy Débil (+)	0.137	Muy Débil (+)	64.971 ± 41.35	0.234 ± 0.124	0.0	0.0
Mediapipe	Der	0.4	15.0	15.458 ± 6.153	2.466 ± 1.092	0.157	Muy Débil (+)	0.132	Muy Débil (+)	67.082 ± 41.12	0.246 ± 0.125	0.0	0.0
Dataset Suavizado	Izq	0.1	10.0	13.798 ± 7.931	1.585 ± 0.922	0.156	Muy Débil (+)	0.257	Débil (+)	67.758 ± 60.347	0.143 ± 0.105	0.003	0.005
Dataset Suavizado	Izq	0.2	10.0	13.798 ± 7.931	1.585 ± 0.922	0.156	Muy Débil (+)	0.257	Débil (+)	67.758 ± 60.347	0.143 ± 0.105	0.003	0.005
Dataset Suavizado	Izq	0.3	10.0	13.798 ± 7.931	1.585 ± 0.922	0.156	Muy Débil (+)	0.257	Débil (+)	67.758 ± 60.347	0.143 ± 0.105	0.003	0.005
Mediapipe	Izq	0.6	5.0	15.452 ± 7.908	1.933 ± 1.037	0.156	Muy Débil (+)	0.199	Muy Débil (+)	78.045 ± 60.597	0.181 ± 0.118	0.0	0.0
Mediapipe	Izq	0.4	10.0	19.681 ± 8.347	2.856 ± 1.428	0.151	Muy Débil (+)	0.16	Muy Débil (+)	99.232 ± 67.586	0.275 ± 0.163	0.0	0.0
Mediapipe	Der	0.6	10.0	13.6 ± 5.884	1.993 ± 0.878	0.151	Muy Débil (+)	0.126	Muy Débil (+)	59.668 ± 40.232	0.199 ± 0.1	0.0	0.0
Mediapipe	Izq	0.5	10.0	17.315 ± 8.383	2.259 ± 1.146	0.15	Muy Débil (+)	0.149	Muy Débil (+)	87.333 ± 64.404	0.216 ± 0.131	0.0	0.0
Mediapipe	Izq	0.9	20.0	9.822 ± 5.833	1.547 ± 1.263	0.149	Muy Débil (+)	0.005	Muy Débil (+)	31.288 ± 15.216	0.099 ± 0.144	0.704	0.034
Dataset Suavizado	Izq	0.4	5.0	13.948 ± 7.99	1.5 ± 0.849	0.148	Muy Débil (+)	0.299	Débil (+)	68.585 ± 60.227	0.137 ± 0.097	0.002	0.006
Mediapipe	Der	0.1	15.0	16.814 ± 6.394	2.729 ± 1.152	0.148	Muy Débil (+)	0.095	Muy Débil (+)	72.664 ± 42.297	0.275 ± 0.132	0.0	0.0
Mediapipe	Der	0.2	15.0	16.814 ± 6.394	2.729 ± 1.152	0.148	Muy Débil (+)	0.095	Muy Débil (+)	72.664 ± 42.297	0.275 ± 0.132	0.0	0.0
Mediapipe	Der	0.3	15.0	16.814 ± 6.394	2.729 ± 1.152	0.148	Muy Débil (+)	0.095	Muy Débil (+)	72.664 ± 42.297	0.275 ± 0.132	0.0	0.0
Dataset	Izq	0.7	10.0	14.26 ± 7.503	1.451 ± 0.714	0.147	Muy Débil (+)	0.331	Débil (+)	72.868 ± 58.757	0.14 ± 0.081	0.001	0.004
Dataset	Der	0.1	25.0	17.074 ± 6.839	2.838 ± 1.184	0.147	Muy Débil (+)	0.113	Muy Débil (+)	72.952 ± 43.396	0.287 ± 0.135	0.0	0.0
Dataset	Der	0.2	25.0	17.074 ± 6.839	2.838 ± 1.184	0.147	Muy Débil (+)	0.113	Muy Débil (+)	72.952 ± 43.396	0.287 ± 0.135	0.0	0.0
Dataset	Der	0.3	25.0	17.074 ± 6.839	2.838 ± 1.184	0.147	Muy Débil (+)	0.113	Muy Débil (+)	72.952 ± 43.396	0.287 ± 0.135	0.0	0.0
Mediapipe	Der	0.7	5.0	11.81 ± 5.325	1.694 ± 0.835	0.147	Muy Débil (+)	0.055	Muy Débil (+)	51.923 ± 37.992	0.164 ± 0.095	0.0	0.0
Mediapipe	Izq	0.1	10.0	20.954 ± 8.561	2.865 ± 1.352	0.146	Muy Débil (+)	0.186	Muy Débil (+)	105.096 ± 69.89	0.281 ± 0.154	0.0	0.0
Mediapipe	Izq	0.2	10.0	20.954 ± 8.561	2.865 ± 1.352	0.146	Muy Débil (+)	0.186	Muy Débil (+)	105.096 ± 69.89	0.281 ± 0.154	0.0	0.0
Mediapipe	Izq	0.3	10.0	20.954 ± 8.561	2.865 ± 1.352	0.146	Muy Débil (+)	0.186	Muy Débil (+)	105.096 ± 69.89	0.281 ± 0.154	0.0	0.0
Dataset	Der	0.8	5.0	10.025 ± 5.127	1.225 ± 0.707	0.145	Muy Débil (+)	0.138	Muy Débil (+)	43.788 ± 37.649	0.111 ± 0.081	0.0	0.0
Dataset	Der	0.6	15.0	14.032 ± 5.626	2.09 ± 0.882	0.145	Muy Débil (+)	0.103	Muy Débil (+)	62.363 ± 41.118	0.211 ± 0.101	0.0	0.0

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Mediapipe	Izq	1.0	10.0	9.372 ± 4.294	1.333 ± 0.963	0.143	Muy Débil (+)	0.096	Muy Débil (+)	39.877 ± 25.9	0.102 ± 0.11	0.199	0.47
Mediapipe	Izq	0.9	25.0	10.986 ± 7.691	1.573 ± 1.293	0.141	Muy Débil (+)	0.0	Muy Débil (+)	26.62 ± 18.768	0.099 ± 0.148	0.307	0.004
Mediapipe	Izq	0.4	5.0	20.804 ± 8.404	2.865 ± 1.352	0.14	Muy Débil (+)	0.186	Muy Débil (+)	105.09 ± 69.782	0.281 ± 0.154	0.0	0.0
Mediapipe	Izq	0.5	5.0	18.285 ± 8.532	2.373 ± 1.191	0.139	Muy Débil (+)	0.218	Débil (+)	92.474 ± 66.67	0.228 ± 0.136	0.0	0.0
Mediapipe	Izq	1.0	15.0	9.62 ± 5.281	1.46 ± 1.154	0.137	Muy Débil (+)	0.061	Muy Débil (+)	33.827 ± 18.322	0.099 ± 0.132	0.381	0.299
Mediapipe	Izq	0.1	5.0	22.085 ± 8.681	3.293 ± 1.468	0.135	Muy Débil (+)	0.154	Muy Débil (+)	110.869 ± 72.324	0.327 ± 0.168	0.0	0.0
Mediapipe	Izq	0.2	5.0	22.085 ± 8.681	3.293 ± 1.468	0.135	Muy Débil (+)	0.154	Muy Débil (+)	110.869 ± 72.324	0.327 ± 0.168	0.0	0.0
Mediapipe	Izq	0.3	5.0	22.085 ± 8.681	3.293 ± 1.468	0.135	Muy Débil (+)	0.154	Muy Débil (+)	110.869 ± 72.324	0.327 ± 0.168	0.0	0.0
Dataset	Der	0.5	20.0	14.928 ± 6.013	2.395 ± 0.922	0.135	Muy Débil (+)	0.104	Muy Débil (+)	65.69 ± 42.285	0.246 ± 0.105	0.0	0.0
Mediapipe	Der	0.6	5.0	14.546 ± 6.04	2.2 ± 0.918	0.134	Muy Débil (+)	0.064	Muy Débil (+)	63.914 ± 41.46	0.222 ± 0.105	0.0	0.0
Dataset Suavizado	Izq	0.1	5.0	14.244 ± 8.169	1.598 ± 0.906	0.132	Muy Débil (+)	0.243	Débil (+)	70.051 ± 61.812	0.146 ± 0.103	0.002	0.005
Dataset Suavizado	Izq	0.2	5.0	14.244 ± 8.169	1.598 ± 0.906	0.132	Muy Débil (+)	0.243	Débil (+)	70.051 ± 61.812	0.146 ± 0.103	0.002	0.005
Dataset Suavizado	Izq	0.3	5.0	14.244 ± 8.169	1.598 ± 0.906	0.132	Muy Débil (+)	0.243	Débil (+)	70.051 ± 61.812	0.146 ± 0.103	0.002	0.005
Mediapipe	Izq	1.0	5.0	9.499 ± 4.216	1.333 ± 0.963	0.132	Muy Débil (+)	0.096	Muy Débil (+)	41.815 ± 28.29	0.102 ± 0.11	0.189	0.47
Dataset	Der	0.7	10.0	12.693 ± 5.387	1.85 ± 0.871	0.131	Muy Débil (+)	0.089	Muy Débil (+)	56.503 ± 40.034	0.181 ± 0.099	0.0	0.0
Mediapipe	Der	0.5	10.0	16.889 ± 5.949	2.753 ± 0.963	0.127	Muy Débil (+)	0.103	Muy Débil (+)	74.437 ± 42.735	0.287 ± 0.11	0.0	0.0
Dataset	Izq	0.7	5.0	15.1 ± 7.93	1.762 ± 0.881	0.124	Muy Débil (+)	0.201	Débil (+)	77.155 ± 61.553	0.17 ± 0.101	0.001	0.001
Dataset	Izq	0.6	20.0	12.956 ± 7.013	1.367 ± 0.726	0.124	Muy Débil (+)	0.178	Muy Débil (+)	65.11 ± 54.43	0.129 ± 0.083	0.01	0.026
Dataset	Der	0.7	5.0	13.455 ± 5.596	1.864 ± 0.798	0.117	Muy Débil (+)	0.098	Muy Débil (+)	60.078 ± 42.118	0.187 ± 0.091	0.0	0.0
Dataset	Izq	0.5	25.0	13.566 ± 7.644	1.423 ± 0.897	0.113	Muy Débil (+)	0.277	Débil (+)	67.228 ± 58.064	0.123 ± 0.102	0.008	0.01
Dataset	Der	0.4	20.0	18.73 ± 6.551	3.129 ± 1.308	0.112	Muy Débil (+)	0.078	Muy Débil (+)	81.886 ± 46.037	0.316 ± 0.149	0.0	0.0
Dataset	Der	0.6	10.0	16.453 ± 5.705	2.606 ± 0.963	0.112	Muy Débil (+)	0.072	Muy Débil (+)	73.61 ± 44.061	0.269 ± 0.11	0.0	0.0
Mediapipe	Der	0.5	5.0	17.958 ± 6.214	2.875 ± 1.029	0.111	Muy Débil (+)	0.088	Muy Débil (+)	79.076 ± 44.374	0.298 ± 0.117	0.0	0.0
Mediapipe	Der	0.4	10.0	19.698 ± 6.45	3.285 ± 1.213	0.109	Muy Débil (+)	0.081	Muy Débil (+)	86.233 ± 46.034	0.339 ± 0.138	0.0	0.0
Mediapipe	Izq	1.0	25.0	12.071 ± 8.537	1.573 ± 1.293	0.107	Muy Débil (+)	0.0	Muy Débil (+)	28.108 ± 21.387	0.099 ± 0.148	0.029	0.004
Dataset	Der	0.5	15.0	18.339 ± 5.914	3.095 ± 1.095	0.106	Muy Débil (+)	0.079	Muy Débil (+)	81.706 ± 46.358	0.322 ± 0.125	0.0	0.0
Dataset	Der	0.1	20.0	20.927 ± 7.198	3.502 ± 1.398	0.106	Muy Débil (+)	0.07	Muy Débil (+)	90.417 ± 48.222	0.357 ± 0.16	0.0	0.0

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Dataset	Der	0.2	20.0	20.927 ± 7.198	3.502 ± 1.398	0.106	Muy Débil (+)	0.07	Muy Débil (+)	90.417 ± 48.222	0.357 ± 0.16	0.0	0.0
Dataset	Der	0.3	20.0	20.927 ± 7.198	3.502 ± 1.398	0.106	Muy Débil (+)	0.07	Muy Débil (+)	90.417 ± 48.222	0.357 ± 0.16	0.0	0.0
Mediapipe	Der	0.1	10.0	21.174 ± 6.731	3.706 ± 1.292	0.105	Muy Débil (+)	0.061	Muy Débil (+)	92.178 ± 47.159	0.386 ± 0.148	0.0	0.0
Mediapipe	Der	0.2	10.0	21.174 ± 6.731	3.706 ± 1.292	0.105	Muy Débil (+)	0.061	Muy Débil (+)	92.178 ± 47.159	0.386 ± 0.148	0.0	0.0
Mediapipe	Der	0.3	10.0	21.174 ± 6.731	3.706 ± 1.292	0.105	Muy Débil (+)	0.061	Muy Débil (+)	92.178 ± 47.159	0.386 ± 0.148	0.0	0.0
Dataset	Izq	0.6	15.0	15.503 ± 8.255	1.878 ± 0.93	0.104	Muy Débil (+)	0.144	Muy Débil (+)	78.961 ± 64.13	0.181 ± 0.106	0.0	0.001
Mediapipe	Izq	1.0	20.0	10.788 ± 7.348	1.573 ± 1.293	0.104	Muy Débil (+)	0.0	Muy Débil (+)	28.06 ± 18.679	0.099 ± 0.148	0.559	0.004
Dataset	Der	0.6	5.0	17.392 ± 5.781	2.81 ± 0.985	0.101	Muy Débil (+)	0.076	Muy Débil (+)	77.992 ± 45.969	0.292 ± 0.112	0.0	0.0
Mediapipe	Der	0.4	5.0	20.944 ± 6.673	3.598 ± 1.265	0.099	Muy Débil (+)	0.062	Muy Débil (+)	91.568 ± 47.377	0.374 ± 0.144	0.0	0.0
Dataset	Izq	0.6	10.0	18.005 ± 8.821	2.294 ± 1.124	0.095	Muy Débil (+)	0.177	Muy Débil (+)	92.488 ± 70.475	0.222 ± 0.128	0.0	0.0
Mediapipe	Der	0.1	5.0	22.377 ± 7.026	3.77 ± 1.333	0.095	Muy Débil (+)	0.055	Muy Débil (+)	97.275 ± 48.794	0.392 ± 0.152	0.0	0.0
Mediapipe	Der	0.2	5.0	22.377 ± 7.026	3.77 ± 1.333	0.095	Muy Débil (+)	0.055	Muy Débil (+)	97.275 ± 48.794	0.392 ± 0.152	0.0	0.0
Mediapipe	Der	0.3	5.0	22.377 ± 7.026	3.77 ± 1.333	0.095	Muy Débil (+)	0.055	Muy Débil (+)	97.275 ± 48.794	0.392 ± 0.152	0.0	0.0
Dataset	Izq	0.4	25.0	16.476 ± 8.98	2.081 ± 1.185	0.089	Muy Débil (+)	0.094	Muy Débil (+)	81.724 ± 67.795	0.19 ± 0.135	0.0	0.001
Dataset	Der	0.4	15.0	22.471 ± 6.568	3.742 ± 1.091	0.089	Muy Débil (+)	0.062	Muy Débil (+)	98.944 ± 50.147	0.398 ± 0.125	0.0	0.0
Dataset	Der	0.1	15.0	24.954 ± 7.37	4.261 ± 1.317	0.085	Muy Débil (+)	0.063	Muy Débil (+)	108.45 ± 52.394	0.45 ± 0.15	0.0	0.0
Dataset	Der	0.2	15.0	24.954 ± 7.37	4.261 ± 1.317	0.085	Muy Débil (+)	0.063	Muy Débil (+)	108.45 ± 52.394	0.45 ± 0.15	0.0	0.0
Dataset	Der	0.3	15.0	24.954 ± 7.37	4.261 ± 1.317	0.085	Muy Débil (+)	0.063	Muy Débil (+)	108.45 ± 52.394	0.45 ± 0.15	0.0	0.0
Dataset	Der	0.5	10.0	21.149 ± 5.971	3.569 ± 1.016	0.085	Muy Débil (+)	0.055	Muy Débil (+)	94.425 ± 49.471	0.38 ± 0.116	0.0	0.0
Dataset	Izq	0.5	20.0	16.281 ± 8.637	1.867 ± 0.954	0.081	Muy Débil (+)	0.155	Muy Débil (+)	82.324 ± 66.523	0.178 ± 0.109	0.0	0.001
Dataset	Izq	0.6	5.0	18.958 ± 8.992	2.513 ± 1.196	0.08	Muy Débil (+)	0.107	Muy Débil (+)	97.606 ± 72.854	0.246 ± 0.136	0.0	0.0
Dataset	Izq	0.1	25.0	18.266 ± 9.871	2.463 ± 1.321	0.08	Muy Débil (+)	0.05	Muy Débil (+)	90.151 ± 73.988	0.231 ± 0.151	0.0	0.0
Dataset	Izq	0.2	25.0	18.266 ± 9.871	2.463 ± 1.321	0.08	Muy Débil (+)	0.05	Muy Débil (+)	90.151 ± 73.988	0.231 ± 0.151	0.0	0.0
Dataset	Izq	0.3	25.0	18.266 ± 9.871	2.463 ± 1.321	0.08	Muy Débil (+)	0.05	Muy Débil (+)	90.151 ± 73.988	0.231 ± 0.151	0.0	0.0
Dataset	Der	0.5	5.0	22.116 ± 5.992	3.72 ± 1.016	0.078	Muy Débil (+)	0.045	Muy Débil (+)	98.914 ± 51.204	0.398 ± 0.116	0.0	0.0
Dataset	Der	0.4	10.0	25.558 ± 6.646	4.454 ± 1.291	0.075	Muy Débil (+)	0.052	Muy Débil (+)	112.666 ± 53.04	0.474 ± 0.147	0.0	0.0
Dataset	Izq	0.5	15.0	19.513 ± 9.282	2.606 ± 1.291	0.073	Muy Débil (+)	0.114	Muy Débil (+)	100.008 ± 74.757	0.251 ± 0.147	0.0	0.0

Continúa en la página siguiente

Tabla 17 – Continuación de la página anterior

Grupo	Extremidad	umbral_temporal	umbral_amplitud	Error RMSE conteo	Error RMSE frecuencia	Lin_conteo	ClasLin_conteo	Lin_frec	ClasLin_frec	ErrorRel_conteo	ErrorRel_ffrec	pValue_conteo	pValue_ffrec
Dataset	Der	0.1	10.0	28.083 ± 7.571	5.052 ± 1.608	0.072	Muy Débil (+)	0.037	Muy Débil (+)	122.251 ± 55.522	0.532 ± 0.184	0.0	0.0
Dataset	Der	0.2	10.0	28.083 ± 7.571	5.052 ± 1.608	0.072	Muy Débil (+)	0.037	Muy Débil (+)	122.251 ± 55.522	0.532 ± 0.184	0.0	0.0
Dataset	Der	0.3	10.0	28.083 ± 7.571	5.052 ± 1.608	0.072	Muy Débil (+)	0.037	Muy Débil (+)	122.251 ± 55.522	0.532 ± 0.184	0.0	0.0
Dataset	Izq	0.5	10.0	22.491 ± 9.083	3.253 ± 1.257	0.07	Muy Débil (+)	0.089	Muy Débil (+)	116.265 ± 79.882	0.333 ± 0.143	0.0	0.0
Dataset	Der	0.4	5.0	26.676 ± 6.701	4.588 ± 1.228	0.07	Muy Débil (+)	0.05	Muy Débil (+)	117.696 ± 54.807	0.491 ± 0.14	0.0	0.0
Dataset	Der	0.1	5.0	29.273 ± 7.728	5.161 ± 1.468	0.067	Muy Débil (+)	0.048	Muy Débil (+)	127.538 ± 57.445	0.55 ± 0.168	0.0	0.0
Dataset	Der	0.2	5.0	29.273 ± 7.728	5.161 ± 1.468	0.067	Muy Débil (+)	0.048	Muy Débil (+)	127.538 ± 57.445	0.55 ± 0.168	0.0	0.0
Dataset	Der	0.3	5.0	29.273 ± 7.728	5.161 ± 1.468	0.067	Muy Débil (+)	0.048	Muy Débil (+)	127.538 ± 57.445	0.55 ± 0.168	0.0	0.0
Dataset	Izq	0.4	20.0	19.725 ± 9.997	2.685 ± 1.445	0.064	Muy Débil (+)	0.108	Muy Débil (+)	99.118 ± 76.487	0.251 ± 0.165	0.0	0.0
Dataset	Izq	0.4	15.0	23.433 ± 9.845	3.464 ± 1.424	0.059	Muy Débil (+)	0.089	Muy Débil (+)	119.668 ± 83.285	0.351 ± 0.163	0.0	0.0
Dataset	Izq	0.4	10.0	26.852 ± 9.649	4.072 ± 1.348	0.058	Muy Débil (+)	0.084	Muy Débil (+)	137.643 ± 88.601	0.427 ± 0.154	0.0	0.0
Dataset	Izq	0.5	5.0	23.593 ± 9.312	3.38 ± 1.334	0.058	Muy Débil (+)	0.069	Muy Débil (+)	122.01 ± 82.725	0.345 ± 0.152	0.0	0.0
Dataset	Izq	0.1	20.0	21.665 ± 10.513	3.137 ± 1.624	0.057	Muy Débil (+)	0.062	Muy Débil (+)	108.626 ± 81.957	0.298 ± 0.185	0.0	0.0
Dataset	Izq	0.2	20.0	21.665 ± 10.513	3.137 ± 1.624	0.057	Muy Débil (+)	0.062	Muy Débil (+)	108.626 ± 81.957	0.298 ± 0.185	0.0	0.0
Dataset	Izq	0.3	20.0	21.665 ± 10.513	3.137 ± 1.624	0.057	Muy Débil (+)	0.062	Muy Débil (+)	108.626 ± 81.957	0.298 ± 0.185	0.0	0.0
Dataset	Izq	0.1	15.0	25.639 ± 10.371	3.934 ± 1.632	0.053	Muy Débil (+)	0.051	Muy Débil (+)	130.254 ± 89.113	0.398 ± 0.186	0.0	0.0
Dataset	Izq	0.2	15.0	25.639 ± 10.371	3.934 ± 1.632	0.053	Muy Débil (+)	0.051	Muy Débil (+)	130.254 ± 89.113	0.398 ± 0.186	0.0	0.0
Dataset	Izq	0.3	15.0	25.639 ± 10.371	3.934 ± 1.632	0.053	Muy Débil (+)	0.051	Muy Débil (+)	130.254 ± 89.113	0.398 ± 0.186	0.0	0.0
Dataset	Izq	0.1	10.0	29.201 ± 10.267	4.611 ± 1.624	0.051	Muy Débil (+)	0.035	Muy Débil (+)	148.727 ± 94.632	0.48 ± 0.185	0.0	0.0
Dataset	Izq	0.2	10.0	29.201 ± 10.267	4.611 ± 1.624	0.051	Muy Débil (+)	0.035	Muy Débil (+)	148.727 ± 94.632	0.48 ± 0.185	0.0	0.0
Dataset	Izq	0.3	10.0	29.201 ± 10.267	4.611 ± 1.624	0.051	Muy Débil (+)	0.035	Muy Débil (+)	148.727 ± 94.632	0.48 ± 0.185	0.0	0.0
Dataset	Izq	0.4	5.0	28.054 ± 9.891	4.431 ± 1.531	0.049	Muy Débil (+)	0.047	Muy Débil (+)	143.871 ± 91.607	0.462 ± 0.175	0.0	0.0
Dataset	Izq	0.1	5.0	30.4 ± 10.474	4.845 ± 1.728	0.045	Muy Débil (+)	0.038	Muy Débil (+)	154.874 ± 97.22	0.503 ± 0.197	0.0	0.0
Dataset	Izq	0.2	5.0	30.4 ± 10.474	4.845 ± 1.728	0.045	Muy Débil (+)	0.038	Muy Débil (+)	154.874 ± 97.22	0.503 ± 0.197	0.0	0.0
Dataset	Izq	0.3	5.0	30.4 ± 10.474	4.845 ± 1.728	0.045	Muy Débil (+)	0.038	Muy Débil (+)	154.874 ± 97.22	0.503 ± 0.197	0.0	0.0

11.8.4. Anexo: Características de los videos, personas y estrategia de movimiento en relación al conteo de «acciones técnicas dinámicas»

N° de Video	N° Acciones Técnicas Consenso	N° Acciones Técnicas Mediapipe Suavizado	Error - Diferencia	Ropa	Género / Estatura	Color de piel / Etnia	Fps Video	Estrategia de Movimiento
video 12	27	26.1	0.9	Sin gorro, polerón negro desabrochado y polera azul, antebrazos cubiertos, buen contraste	Hombre estatura media	Tez clara, pelo corto y barba negra	11	Giros: Tipo 1 Hombros: Tipo 1 Agachado: Tipo 1 Perspectiva: Tipo 1
video 14	28.3	29.3	1.0	polerón verde, antebrazos cubiertos, buen contraste,	Mujer, estatura media-alta	Tez clara, pelo suelto hasta los hombros y barba negra	12	Giros: Tipo 1 Hombros: Tipo 1 Agachado: Tipo 1 Perspectiva: Tipo 1
video 3	18.3	19.7	1.9	Sin gorro, con lentes, chaqueta de cuero, antebrazos descubiertos, contraste regular	Hombre Alto	Tez morena, no caucásico	8	Giros: Tipo 2 Hombros: Tipo 2 Agachado: Tipo 2 Perspectiva: Tipo 1
video 17	17.7	19.2	-1.5	Sin gorro, poleron gris, antebrazos cubiertos, buen contraste	Hombre Alto	Tez clara, pelo corto, sin barba, oriental	8	Giros: Tipo 1 Hombros: Tipo 1 Agachado: Tipo 1 Perspectiva: Tipo 1
video 18	26	24.1	1.9	Pelo largo, suelto, camisa negra con cuadros blancos, antebrazos descubiertos, contraste regular	mujer baja	tez clara, oriental	8	Giros: Tipo 1 Hombros: Tipo 1 Agachado: Tipo 1 Perspectiva: Tipo 1
video 15	26.3	28.7	-2.4	Con gorro, chaqueta negra cerrada, antebrazos cubiertos, mal contraste	Hombre alto	Tez oscura no caucásico	12	Giros: Tipo 1 Hombros: Tipo 1 Agachado: Tipo 1 Perspectiva: Tipo 1
video 16	24.3	27.8	-3.5	Suéter o polerón azul oscuro, antebrazos cubiertos , contraste regular	Hombre bajo	Tez morena no caucásico	12	Giros: Tipo 1 Hombros: Tipo 1 Agachado: Tipo 2 Perspectiva: Tipo 1
video 10	22.3	26.8	-4.5	Con gorro, polerón negro, antebrazos cubiertos , contraste regular	Hombre alto	Tez clara caucásico	12	Giros: Tipo 2 Hombros: Tipo 2 Agachado: Tipo 3 Perspectiva: Tipo 1
video 5	20	24.7	-4.7	Sin gorro, camisa de cuadros, antebrazos descubiertos buen contraste	Hombre Alto	Tez clara, oriental	11	Giros: Tipo 2 Hombros: Tipo 3 Agachado: Tipo 2 Perspectiva: Tipo 1
video 6	36.6	31.3	5	Pelo largo, suelto, suéter negro, antebrazos cubiertos, contraste regular	mujer baja	tez clara, oriental	8	Giros: Tipo 3 Hombros: Tipo 3 Agachado: Tipo 1 Perspectiva: Tipo 1

Tabla 18: Descripción de videos de menor error el conteo automático «Acciones Técnicas Dinámicas» para Mediapipe Suavizado

Donde, la estrategia de movimiento:
a) Giros: Tipo 1) En bloque, sin rotación de tronco, pies acompañan la dirección del movimiento. Tipo 2) Con rotación de tronco, poco movimiento de pies o no acompañan el movimiento. Tipo 3) Alterna tipo 1 y tipo 2.
b) Movimiento en hombros: Tipo 1) Movimientos monoplanares tendientes a la Flexión. No combina de manera categórica la abducción y rotaciones de hombro. Tipo 2) Movimientos combinados de flexión, abducción y rotaciones de hombro. Tipo 3) Alterna tipo 1 y tipo 2.
c) Agachado: Tipo 1) Predomina el movimiento caderas y rodillas. Tipo 2) Predomina el movimiento en flexión de tronco. Tipo 3) Alterna tipo 1 y tipo 2.
d) Perspectiva: Tipo 1) Vista frontal a la cámara. Tipo 2) Vista trasera a la cámara. Tipo 3) Alterna tipo 1 y tipo 2.

N° de Video	N° Acciones Técnicas Consenso	N° Acciones Técnicas Mediapipe Suavizado	Error (Consenso - Mediapipe S.)	Ropa	Sexo, Estatura	Color de piel / Etnia	Fps Video	Estrategia de Movimiento
video 2	19	24.2	-5.2	Sin gorro, polerón blango, antebrazos cubiertos buen contraste	Hombre media	Tez clara, Caucásico	10	Giros: Tipo 3 Hombros: Tipo 2 Agachado: Tipo 3 Perspectiva: Tipo 3
video 4	31	24.5	6.5	Sin gorro, polerón gris, antebrazos descubiertos, buen contraste	Hombre, Alto	Tez clara, caucásico	11	Giros: Tipo 2 Hombros: Tipo 2 Agachado: Tipo 1 Perspectiva: Tipo 1
video 19	10.7	17.4	-6.7	Sin gorro, suéter verde con contraste, antebrazos cubiertos buen contraste	Hombre, Alto	Tez clara, Caucásico	8	Giros: Tipo 3 Hombros: Tipo 3 Agachado: Tipo 1 Perspectiva: Tipo 1
video 7	22.3	29.7	-7.4	Pelo rubio corto, polera verde, antebrazos descubiertos, buen contraste	Hombre, alta	Tez clara, caucásico	12	Giros: Tipo 3 Hombros: Tipo 3 Agachado: Tipo 1 Perspectiva: Tipo 1
video 1	30.3	22.8	7.5	Pelo negro largo hombros, suelto, polera blanca, antebrazos descubiertos, buen contraste	Mujer, estatura baja	Tez clara, oriental, lentes ópticos	8	Giros: Tipo 3 Hombros: Tipo 2 Agachado: Tipo 3 Perspectiva: Tipo 3
video 11	20	27.6	-7.6	Pelo negro largo hombros, suelto, polera blanca, antebrazos descubiertos, buen contraste	Mujer, estatura baja	Tez clara, oriental	12	Giros: Tipo 2 Hombros: Tipo 2 Agachado: Tipo 1 Perspectiva: Tipo 3
video 9	41	33.3	7.7	pelo corto, lentes, polerón negro con rayas blancas hacia los brazos, un antebrazo cubierto y el otro descubierto, buen contraste	Hombre, media-baja	Tez clara, oriental	12	Giros: Tipo 2 Hombros: Tipo 2 Agachado: Tipo 1 Perspectiva: Tipo 1
video 20	28.7	20.9	7.8	Sin gorro, chaqueta negra suelta, antebrazos cubiertos mal contraste	Hombre bajo	Tez clara, oriental	8	Giros: Tipo 1 Hombros: Tipo 1 Agachado: Tipo 1 Perspectiva: Tipo 1
video 8 ‡	17	26	-9.3	polerón gris oscuro, antebrazos cubiertos , contraste regular	Hombre alto	Tez clara oriental	12	Giros: Tipo 1 Hombros: Tipo 1 Agachado: Tipo 1 Perspectiva: Tipo 1
video 13	20.3	30.2	-9.9	Gorro celeste, polerón azul, antebrazos cubiertos buen contraste	Hombre Alto	Tez clara, Caucásico	12	Giros: Tipo 2 Hombros: Tipo 2 Agachado: Tipo 2 Perspectiva: Tipo 1

Tabla 19: Descripción de videos de mayor error el conteo automático «Acciones Técnicas Dinámicas» para Mediapipe Suavizado

Donde, la estrategia de movimiento:
a) Giros: Tipo 1) En bloque, sin rotación de tronco, pies acompañan la dirección del movimiento. Tipo 2) Con rotación de tronco, poco movimiento de pies o no acompañan el movimiento. Tipo 3) Alterna tipo 1 y tipo 2.
b) Movimiento en hombros: Tipo 1) Movimientos monoplanares tendientes a la Flexión. No combina de manera categórica la abducción y rotaciones de hombro. Tipo 2) Movimientos combinados de flexión, abducción y rotaciones de hombro. Tipo 3) Alterna tipo 1 y tipo 2.
c) Agachado: Tipo 1) Predomina el movimiento caderas y rodillas. Tipo 2) Predomina el movimiento en flexión de tronco. Tipo 3) Alterna tipo 1 y tipo 2.
d) Perspectiva: Tipo 1) Vista frontal a la cámara. Tipo 2) Vista trasera a la cámara. Tipo 3) Alterna tipo 1 y tipo 2.

‡ agrega un movimiento de flexión de codos en exceso, posterior a depositar cada caja o cada barra.