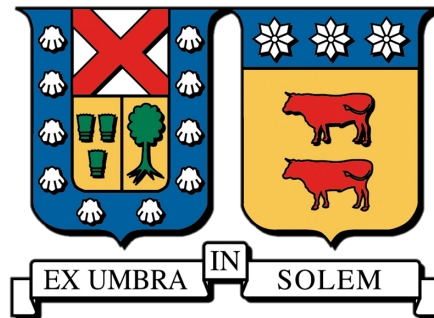


UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE ELECTRÓNICA
VALPARAÍSO - CHILE



**“DESARROLLO DE UN MODELO AVANZADO
PARA LA IDENTIFICACIÓN Y EVALUACIÓN DE
MÉTRICAS DE IMPACTO CEFÁLICO EN NIÑOS
CON TRASTORNO DEL ESPECTRO AUTISTA”**

JONATHAN PEDRAZA VALDEBENITO

**MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO
CIVIL TELEMÁTICO**

PROFESOR GUÍA:

MAURICIO ARAYA

PROFESOR CORREFERENTE:

NICOLÁS JARA

JUNIO 2026



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título; ☐ Tesis de Postgrado;

Título del trabajo: DESARROLLO DE UN MODELO AVANZADO PARA LA IDENTIFICACIÓN Y EVALUACIÓN DE MÉTRICAS DE IMPACTO CEFÁLICO EN NIÑOS CON TRASTORNO DEL ESPECTRO AUTISTA

Nombre del candidato(a): Jonathan Ignacio Abraham Pedraza Valdebenito

Carrera / Grado: Ingeniería Civil Telemática

Campus: Casa Central Valparaíso; **Departamento:** Electrónica

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Mauricio Araya López, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL

☒ El trabajo **NO contiene información que amerite confidencialidad** y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por:

☐ 6 meses; ☐ 12 meses; ☐ 2 años; ☐ 3 años; ☐ 5 años; ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 17/06/2026

; Firma:

Estudiante o Candidato(a):

Fecha: 17/06/2026

; Firma:

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

Agradecimientos

Quisiera expresar mi más sincero agradecimiento a mi profesor guía, Mauricio Araya, por su apoyo constante, su paciencia y la confianza depositada en este trabajo a lo largo de todo su desarrollo. Su orientación fue fundamental para llevar esta memoria a buen puerto.

Agradezco también al Departamento de Electrónica de la Universidad Técnica Federico Santa María por brindarme las herramientas y el espacio necesarios para realizar esta investigación, y de manera especial a sus profesores, quienes día a día irradian su pasión por la enseñanza y por aquello que hacen, inspirando a quienes tuvimos el privilegio de formarnos con ellos.

A mi familia, mi gratitud más profunda. A pesar de la distancia, su apoyo incondicional nunca me faltó y constituyó el sostén que me permitió perseverar durante este largo proceso. Hago extensivo este agradecimiento a los distintos grupos de amigos que me acompañaron en cada etapa del camino: desde quienes conocí en el preuniversitario hasta los colegas y amigos que la vida universitaria y la convivencia en la pensión me regalaron. Agradezco igualmente a los dueños de la pensión donde residí durante tantos años, por hacer de ese lugar un verdadero hogar lejos de casa.

De forma muy especial, agradezco al Colegio Centro de Educación Especial Comúnica, de Villa Alemana, por abrirme sus puertas y permitirme realizar las pruebas reales del prototipo en un entorno educativo genuino. Su disposición y colaboración fueron esenciales para confrontar este trabajo con las condiciones del mundo real.

Finalmente, deseo reconocer el soporte de las becas y contribuciones que hicieron posible mi formación. Más allá de lo económico, representaron la oportunidad real de estudiar y de alcanzar esta meta, y por ello les estaré siempre agradecido.

Resumen

Se presenta un sistema de visión por computadora para la medición objetiva de la cinemática de golpes autodirigidos a la cabeza, una estereotipia frecuente en niños con Trastorno del Espectro Autista (TEA). El sistema captura la escena con una cámara de profundidad Azure Kinect DK y estima, para cada amago de golpe, la trayectoria tridimensional de la mano, su velocidad y deceleración en el instante de máxima aproximación al rostro. Se comparan tres familias de algoritmos de seguimiento —el *Body Tracking* nativo del Kinect (profundidad), YOLO11-pose y MediaPipe (ambos sobre el flujo RGB con reproyección a 3D)— sobre un conjunto de grabaciones que abarca tres modos de profundidad, tres condiciones de iluminación y múltiples zonas faciales. Se incorpora un filtro de Kalman de velocidad constante, cuyo ruido de proceso se dimensiona a la dinámica del golpe, y un estudio de ablación cuantifica su efecto sobre la velocidad estimada. Los resultados, analizados estadísticamente y categorizados por algoritmo, muestran que el seguimiento por profundidad ofrece la medición más fiable de la cercanía mano–rostro y de la magnitud del golpe. Finalmente se desarrolla la justificación física que, a futuro, permitirá derivar la fuerza de impacto a partir de la deceleración medida.

Palabras clave: trastorno del espectro autista, golpes autodirigidos, visión por computadora, Azure Kinect, estimación de pose, filtro de Kalman, cinemática, impacto cefálico.

Abstract

We present a computer vision system for the objective measurement of the kinematics of self-directed head blows, a frequent stereotypy in children with Autism Spectrum Disorder (ASD). The system captures the scene with an Azure Kinect DK depth camera and estimates, for each approximation event, the three-dimensional trajectory of the hand, its velocity and deceleration at the moment of closest approach to the face. Three families of tracking algorithms are compared—Azure Kinect’s native Body Tracking (depth), YOLO11-pose and MediaPipe (both operating on the RGB stream with 3D back-projection)—over a dataset covering three depth modes, three illumination conditions and multiple facial zones. A constant-velocity Kalman filter is incorporated, with its process noise scaled to the dynamics of the blow, and an ablation study quantifies its effect on the estimated velocity. The results, statistically analyzed and categorized by algorithm, show that depth-based tracking provides the most reliable measurement of hand–face proximity and impact magnitude. Finally, the physical framework that will enable deriving impact force from measured deceleration is developed.

Keywords: autism spectrum disorder, self-injurious behavior, computer vision, Azure Kinect, pose estimation, Kalman filter, kinematics, cephalic impact.

Acrónimos

Sigla	Significado
ASD / TEA	<i>Autism Spectrum Disorder</i> / Trastorno del Espectro Autista
CNN	<i>Convolutional Neural Network</i> / Red Neuronal Convolucional
COCO	<i>Common Objects in Context</i> (conjunto de datos de referencia)
CPU	<i>Central Processing Unit</i> / Unidad Central de Procesamiento
FPS	<i>Frames Per Second</i> / Fotogramas por Segundo
GPU	<i>Graphics Processing Unit</i> / Unidad de Procesamiento Gráfico
KF	<i>Kalman Filter</i> / Filtro de Kalman
NFOV	<i>Narrow Field of View</i> / Campo Visual Estrecho
RGB	Red, Green, Blue (imagen de color estándar)
SDK	<i>Software Development Kit</i> / Kit de Desarrollo de Software
SIB	<i>Self-Injurious Behavior</i> / Conducta Autolesiva
ToF	<i>Time-of-Flight</i> / Tiempo de Vuelo (principio de medición de profundidad)
WFOV	<i>Wide Field of View</i> / Campo Visual Amplio

Glosario de Términos

Amago de golpe	Movimiento de la mano que se dirige bruscamente hacia el rostro hasta alcanzar una distancia de máxima aproximación, sin producir contacto físico real. Constituye el evento de interés a detectar y cuantificar.
Back-projection (reproyección inversa)	Técnica que transforma un par de coordenadas de píxel (u, v) y la profundidad z asociada en coordenadas métricas 3D: $X = (u - c_x)z/f_x$, $Y = (v - c_y)z/f_y$, $Z = z$.
Body Tracking	Módulo del Azure Kinect SDK que estima en tiempo real las posiciones 3D de hasta 32 articulaciones del cuerpo humano directamente desde el flujo de profundidad.
Estereotipia	Movimiento o vocalización repetitiva, rígida y sin propósito funcional aparente, característica de algunos perfiles del TEA.
Jitter	Variación aleatoria de alta frecuencia en la posición estimada de un <i>joint</i> o <i>keypoint</i> , causada por el ruido del sensor o del algoritmo de detección.
Joint	Articulación del cuerpo representada como un punto 3D por el Body Tracking SDK (p. ej., <code>HAND_RIGHT</code> , <code>NOSE</code>).
Keypoint	Punto de interés 2D detectado por un estimador de pose sobre la imagen RGB. Tras la <i>back-projection</i> se convierte en punto 3D.
Landmark	Punto de referencia anatómico devuelto por MediaPipe (Hands: 21 puntos por mano; FaceMesh: 468 puntos faciales).

Umbral de proximidad (U_{prox})

Distancia mínima mano–rostro por debajo de la cual el sistema confirma un evento de impacto (Kinect: 60 mm, MediaPipe: 90 mm, YOLO: 200 mm).

Velocidad de cierre

Tasa de variación negativa de la distancia mano–rostro, $-dd/dt$. Indicador dinámico principal para distinguir un amago de impacto.

Índice de figuras

1.1. Configuración de captura en entorno clínico controlado.	3
2.1. Esquema de articulaciones del Azure Kinect Body Tracking.	12
2.2. Detalle de keypoints de la mano en el Azure Kinect Body Tracking. . .	13
2.3. Esquema de keypoints COCO de YOLO11-pose.	14
2.4. Esquema de landmarks de MediaPipe Hands.	14
4.1. Diagrama de bloques del pipeline de procesamiento.	26
4.2. Sensores Azure Kinect DK.	27
4.3. Configuración de modos de profundidad.	27
4.4. Comparativa de los modos de profundidad del Azure Kinect.	29
4.5. Cobertura por modo de profundidad.	30
4.6. Vista de modos de profundidad desde el frente.	31
4.7. Flujo de datos del SDK de Azure Kinect desde Python.	34
4.8. Prueba 3D→2D, distancia 42 mm.	37
4.9. Prueba 3D→2D, distancia 49 mm.	37
4.10. Prueba 3D→2D: mano en movimiento reconocida como rostro.	38
4.11. Diagrama del diseño experimental y flujo de análisis.	40
4.12. Desplazamiento euclidiano entre frames consecutivos.	42
4.13. Segmentación de la región facial.	43
4.14. Trayectoria original vs. suavizada por filtro de Kalman.	46
4.15. Frame RGB real vs. mapa de profundidad alineado (NFOV Unbinned). .	48

4.16. Geometría de la back-projection (modelo pinhole).	49
4.17. Back-projection 2D→3D: frame RGB y mapa de profundidad.	50
4.18. Nube de puntos 3D de la escena (Azure Kinect, vista diagonal).	51
4.19. Nube de puntos 3D con joint de la muñeca resaltado.	52
4.20. Nube de puntos 3D texturizada con color RGB real.	53
4.21. Diagrama de reproyección RGB→nube de puntos 3D.	54
4.22. Diagrama de reproyección con nube RGB texturizada.	55
5.1. Impactos detectados por algoritmo.	67
5.2. Velocidad de impacto por algoritmo (con Kalman).	67
5.3. Impactos por modo de profundidad: con y sin Kalman.	72
5.4. Impactos por condición de iluminación.	72
5.5. Distribución de distancias y umbral de proximidad por algoritmo.	73
5.6. Ablación Kalman: velocidad media y v_{p95} en Azure Kinect.	75
5.7. Deceleración por algoritmo (con Kalman).	77
5.8. Distancia mano–rostro en el evento (con Kalman).	77
5.9. Velocidad de impacto: con y sin filtro de Kalman.	78
5.10. Deceleración en el evento: con y sin filtro de Kalman.	78
5.11. Distancia mano–rostro: con y sin filtro de Kalman.	79
5.12. Coste computacional: FPS por algoritmo.	79
5.13. Azure Kinect: capturas por modo e iluminación.	83
5.14. YOLO11-pose: capturas por modo e iluminación.	84
5.15. MediaPipe: capturas por modo e iluminación.	85
5.16. Comparación cinemática: profundidad Kinect vs. RGB puro.	87

Índice de cuadros

2.1. Comparativa de las tres arquitecturas de estimación de pose evaluadas. .	11
4.1. Incertidumbre de posición, velocidad y aceleración por método (1σ). El sesgo sistemático se refiere al punto de referencia medido vs. el punto de contacto ideal.	60
5.1. Resumen estadístico por algoritmo con y sin filtro de Kalman. $\bar{v}$, σ_v , v_{max} : media, desviación estándar y máximo de la velocidad de impacto; $\bar{a}$: deceleración media; $\bar{d}$: distancia mano–rostro media.	66
5.2. Detecciones y velocidad por zona facial y algoritmo, con y sin filtro de Kalman.	68
5.3. Detecciones y velocidad por condición de iluminación y algoritmo, con y sin filtro de Kalman.	70
5.4. Detecciones y velocidad por algoritmo y modo de profundidad, con y sin filtro de Kalman.	71
5.5. Estadísticos de velocidad de impacto por algoritmo y configuración del filtro de Kalman. v_{p95} es el percentil 95 (robusto a <i>outliers</i> frente al máximo).	74

5.6. Tiempo de procesamiento por vídeo y algoritmo. El modo de profundidad no afecta significativamente el tiempo de YOLO/MediaPipe (la resolución del frame RGB es la misma en los tres modos); sí afecta al Body Tracking, cuya red opera sobre la imagen de profundidad de resolución variable.	77
5.7. Resultados completos por algoritmo, modo de profundidad, condición de iluminación y configuración del filtro de Kalman (U. = NFOV Unbinned). Los valores en negrita son el mínimo y el máximo global de cada columna entre filas con detecciones.	82
5.8. Estadísticas cinemáticas: profundidad Kinect vs. RGB puro.	86

Índice general

Glosario y Acrónimos	IV
1. Introducción	1
1.1. Problema a resolver	2
1.2. Acercamiento a la solución	2
2. Marco Teórico	4
2.1. Seguimiento corporal con Azure Kinect	4
2.2. Alternativas de captura de profundidad por estereoscopía	5
2.3. Estimación de pose 2D de cuerpo completo	6
2.4. Detección y seguimiento de personas con YOLO	6
2.4.1. YOLO11-pose	6
2.4.2. MediaPipe	7
2.5. Arquitecturas de deep learning para estimación de pose	7
2.5.1. YOLO11-pose: arquitectura anchor-free de una sola etapa . . .	7
2.5.2. MediaPipe Hands y Face Mesh: <i>pipelines</i> en dos etapas	8
2.5.3. Azure Kinect Body Tracking SDK: seguimiento nativo por pro- fundidad	9
2.5.4. Comparación directa de las tres arquitecturas	10
2.5.5. Esquemas de <i>keypoints</i> por modelo	11
3. Estado del Arte	16

3.1.	Detección de estereotipias y conductas autolesivas en el TEA	17
3.1.1.	Instrumentos clínicos y sensores llevables	17
3.1.2.	Enfoques basados en visión y video	18
3.2.	Captura de movimiento markerless y cámaras de profundidad en contextos clínicos	20
3.3.	Filtrado de estado para estimación de velocidad	21
3.4.	Análisis cinemático de golpes y reconocimiento de acciones mediante visión por computador	22
3.5.	Síntesis y posicionamiento	23
4.	Materiales y Métodos	25
4.1.	Arquitectura del sistema	25
4.2.	Equipamiento	25
4.3.	Configuración del modo de profundidad	27
4.4.	Seguridad del láser ToF y efectos de la iluminación infrarroja	31
4.4.1.	Clasificación de seguridad ocular	31
4.4.2.	Efectos de la iluminación ambiental en sensores ToF	32
4.5.	API Python para el SDK de Azure Kinect	33
4.6.	Condiciones de iluminación	35
4.7.	Calibración empírica y artefactos en la segmentación	36
4.7.1.	Validación de la proyección 3D↔2D	36
4.7.2.	Pruebas preliminares de configuración de modo de profundidad	36
4.8.	Diseño experimental	38
4.9.	Definición de impacto	40
4.10.	Detección de movimiento	41
4.11.	Segmentación de la región facial	42
4.12.	Filtro de Kalman para posición y velocidad	43
4.13.	Homogeneización 3D de los métodos RGB: reproyección inversa	46
4.13.1.	Dos cámaras físicamente distintas en el mismo cuerpo	47

4.13.2. Paso a paso de la back-projection	47
4.14. Proyección 3D→2D para anotación	50
4.15. Estimación de velocidad, aproximación y aceleración	56
4.16. Justificación física: de la deceleración a la fuerza de impacto	57
4.17. Análisis de incertidumbre	58
4.18. Algoritmo de detección	61
4.19. Implementación simplificada: núcleo del detector	62
5. Resultados y Análisis	65
5.1. Resumen estadístico por algoritmo	65
5.2. Resultados por zona facial	66
5.3. Resultados por condición de iluminación	69
5.4. Efecto del modo de profundidad	70
5.5. Efecto de la iluminación	72
5.6. Análisis del umbral de proximidad	73
5.7. Efecto del filtro de Kalman sobre la velocidad estimada: estudio de ablación	73
5.8. Deceleración, distancia y coste computacional	75
5.9. Resultados completos	80
5.10. Verificación cualitativa	82
5.11. Comparación cinemática: métodos con profundidad vs. métodos RGB puro	83
6. Discusión	89
7. Conclusiones y Trabajo Futuro	90
7.1. Validación preliminar en entorno real y desafíos detectados	91
Trabajo Futuro	92
Referencias	97

Capítulo 1

Introducción

El Trastorno del Espectro Autista (TEA) es una condición del neurodesarrollo que afecta la comunicación y la interacción social. Entre las manifestaciones conductuales asociadas se encuentran las estereotipias: movimientos repetitivos y sin propósito aparente, algunos de los cuales pueden adoptar la forma de golpes autodirigidos a la cabeza. Las conductas autolesivas (SIB, por sus siglas en inglés) afectan a una fracción significativa de personas con TEA y representan uno de los principales desafíos para el equipo clínico y las familias (Minshawi et al., 2014). Cuando estos golpes alcanzan cierta intensidad o frecuencia, dejan de ser un mero indicador conductual y pasan a constituir un riesgo de lesión.

Clínicamente, la frecuencia y severidad de estas conductas se evalúan mediante escalas administradas por cuidadores o profesionales, como la *Repetitive Behavior Scale–Revised* (RBS-R) (Bodfish et al., 2000; Lam & Aman, 2007) o la subescala de estereotipia del *Aberrant Behavior Checklist* (ABC) (Aman et al., 1985); instrumentos valiosos pero **subjetivos** y de resolución gruesa. Como complemento se han explorado métodos de **detección automática** de movimientos estereotipados basados en acelerómetros y aprendizaje automático (Goodwin et al., 2011), que sin embargo requieren sensores adheridos al cuerpo —precisamente la intrusividad que este trabajo busca evitar.

1.1. Problema a resolver

La motivación de este trabajo radica en una necesidad concreta del ámbito clínico del TEA: *medir de forma objetiva y no intrusiva la intensidad y la frecuencia de los golpes en la cabeza*. Esta medición es relevante por tres razones principales:

1. **Seguridad.** Los golpes en la cabeza pueden causar lesiones; cuantificar su intensidad permite anticipar y prevenir el daño.
2. **Evaluación clínica.** La frecuencia e intensidad de estos episodios son indicadores potencialmente útiles para caracterizar la severidad del cuadro y la necesidad de intervención.
3. **Diseño de tratamientos.** Una métrica objetiva habilita el seguimiento longitudinal del paciente y la evaluación cuantitativa de la eficacia terapéutica.

El objetivo es un sistema que mida estas magnitudes de manera objetiva sin requerir sensores adheridos al cuerpo del paciente, condición indispensable dada la baja tolerancia a estímulos intrusivos que suele acompañar al TEA.

1.2. Acercamiento a la solución

El sistema debe operar en un entorno controlado —típicamente una sesión clínica con un neurólogo— sin interferir con el trabajo del profesional. A lo largo de la sesión se captura la escena mientras el especialista desarrolla sus tareas en paralelo. Para ello se emplea la cámara Microsoft Azure Kinect DK, equipada con un sensor de profundidad por tiempo de vuelo (*Time-of-Flight*) que permite reconstruir la escena tridimensional, y cuyo SDK ofrece seguimiento corporal nativo.

La idea central es rastrear las articulaciones de interés —fundamentalmente la mano y la cabeza— en el espacio 3D, detectar el instante en que la mano se aproxima bruscamente al rostro y extraer de esa interacción las métricas cinemáticas relevantes. Como contraste metodológico, se evalúan además dos enfoques basados exclusivamente en el

flujo de color (YOLO11-pose y MediaPipe), que reconstruyen la posición 3D combinando la detección 2D con el mapa de profundidad alineado.



Figura 1.1: Configuración de captura en entorno clínico controlado: el paciente (izquierda) se sienta frente a la cámara Azure Kinect DK, montada en trípode a ~ 1 m de distancia y $\sim 1,1$ m de altura; el neurólogo supervisa la sesión en un entorno preparado para minimizar interferencias de iluminación exterior.

Capítulo 2

Marco Teórico

2.1. Seguimiento corporal con Azure Kinect

El seguimiento en tiempo real de la pose humana desde sensores de profundidad tiene su origen en el trabajo de Shotton et al. (2011), que propuso reconocer partes del cuerpo en imágenes de profundidad mediante *random forests*, sentando la base del Body Tracking integrado en la familia Kinect. El Azure Kinect DK ha ganado popularidad en investigación con sujetos humanos por su capacidad de seguimiento corporal en tiempo real. Sin embargo, la buena práctica científica exige resultados consistentes y reproducibles. El estudio de Büker et al. (2023) revela inconsistencias en el seguimiento corporal del *Azure Kinect Body Tracking SDK*, especialmente en función del modo de procesamiento (CUDA, TensorRT, DirectML, CPU) y del equipo utilizado. Se observan diferencias clínicamente relevantes en las distancias euclidianas entre posiciones articulares, así como un comportamiento convergente no documentado al inicio del seguimiento.

Sus conclusiones, que este trabajo adopta como guía metodológica, recomiendan: seleccionar y *reportar* explícitamente el modo de procesamiento, ejecutar todos los análisis en el mismo equipo, iniciar el análisis en un estado estable de las posiciones articulares y almacenar múltiples ejecuciones para mitigar el error. En coherencia, todas

las mediciones de esta memoria se ejecutaron en un único equipo y se documenta el modo de procesamiento empleado (CUDA).

2.2. Alternativas de captura de profundidad por estereoscopia

Los sensores ToF como el Azure Kinect presentan una limitación conocida en entornos con alta irradiancia infrarroja: la luz solar directa y la iluminación fluorescente de alta potencia degradan la señal de profundidad, efecto que se manifestó en este trabajo bajo la condición de luz natural con ventanales abiertos. Una alternativa que evita esta dependencia son las cámaras de **estereoscopia pasiva**, que estiman la profundidad mediante triangulación a partir de dos imágenes RGB sin emitir ninguna radiación.

El principio estereoscópico, formalizado por Scharstein y Szeliski (2002), consiste en encontrar las correspondencias entre píxeles homólogos de un par de cámaras calibradas para calcular la disparidad y, a partir de ella, la profundidad métrica. Las cámaras estereoscópicas modernas, como la **ZED 2** (Stereolabs) (Stereolabs, 2023), combinan el cómputo de disparidad denso con redes neuronales que mejoran la precisión del mapa de profundidad, acercando sus prestaciones a las del ToF en interiores. Dado que operan exclusivamente en el espectro visible, su rendimiento no se degrada bajo luz solar y resultan apropiadas para clínicas con grandes ventanales. Su principal limitación es la pérdida de correspondencias en zonas poco texturadas o con oclusiones abruptas, como el contorno del cabello. Un análisis comparativo sistemático de ToF y luz estructurada bajo distintas condiciones de iluminación puede consultarse en Sarbolandi et al. (2015).

La ZED 2 incorpora además un SDK de seguimiento corporal basado en YOLO que proporciona hasta 34 *keypoints* tridimensionales a partir únicamente de los flujos RGB estereoscópicos, lo que la convierte en un candidato directo para complementar o sustituir al Azure Kinect en entornos clínicos con alta incidencia de luz natural.

2.3. Estimación de pose 2D de cuerpo completo

En la estimación de pose corporal en 2D destaca el trabajo fundacional de Cao et al. (2017), que introduce los campos de afinidad de partes (*Part Affinity Fields*) para la estimación multi-persona en tiempo real, y su extensión de cuerpo completo de Hidalgo et al. (2019), que añade rostro, manos y pies a partir de imágenes en condiciones naturales manteniendo rendimiento en tiempo real con independencia del número de personas en la escena. Este tipo de arquitecturas estableció las bases de los estimadores de pose modernos empleados en este trabajo.

2.4. Detección y seguimiento de personas con YOLO

La detección de personas en tiempo real es un área activa de la visión por computadora. Athilakshmi et al. (2023) emplean la arquitectura YOLO-NAS junto a DeepSORT para la detección y el seguimiento de personas en multitudes, alcanzando precisiones superiores al 90 % en varios conjuntos de datos de referencia. La familia YOLO (*You Only Look Once*) realiza, en una única pasada de inferencia, la detección sobre toda la imagen, lo que la hace especialmente apropiada para aplicaciones en tiempo real.

2.4.1. YOLO11-pose

En este trabajo se utiliza **YOLO11-pose** (Jocher et al., 2024), la variante de estimación de pose de la undécima generación de YOLO desarrollada por Ultralytics. A diferencia de los detectores de objetos clásicos, los modelos *-pose* regresan, para cada persona detectada, un conjunto de *keypoints* anatómicos siguiendo la convención COCO (17 puntos: nariz, ojos, orejas, hombros, codos, muñecas, caderas, rodillas y tobillos). La arquitectura es completamente convolucional, de una sola etapa (*anchor-free*), y entrega las coordenadas de los keypoints en el plano de la imagen junto con una confianza por punto. Para este proyecto se emplean la **muñeca derecha** (keypoint 10)

como sustituto de la posición de la mano y los keypoints faciales (nariz, ojos y orejas, 0–4) como referencia del rostro. Su principal limitación en este contexto es que el modelo no expone los dedos ni el centro de la palma: la muñeca queda sistemáticamente más alejada del rostro que la mano, lo que obliga a un umbral de proximidad mayor.

2.4.2. MediaPipe

MediaPipe (Lugaresi et al., 2019) es el marco de Google para construir *pipelines* de percepción multimodal en tiempo real. Se utilizan dos de sus soluciones: *MediaPipe Hands* (F. Zhang et al., 2020), que entrega 21 *landmarks* 3D por mano (de los que se promedian muñeca y nudillos para estimar el centro de la palma), y *MediaPipe Face Mesh*, que produce una malla densa de 468 puntos faciales, submuestreada en este trabajo para definir la superficie del rostro. Ambas soluciones operan sobre el flujo RGB y son ligeras, capaces de ejecutarse en CPU. Su debilidad, relevante para la interpretación de los resultados, es su sensibilidad a las oclusiones: cuando la mano cubre parte del rostro durante el amago de golpe, MediaPipe puede perder los *landmarks* y dejar de detectar el evento.

2.5. Arquitecturas de deep learning para estimación de pose

La siguiente comparativa profundiza en la arquitectura interna de los tres sistemas de estimación de pose evaluados en este trabajo, para contextualizar sus diferencias en precisión, latencia y robustez a oclusiones.

2.5.1. YOLO11-pose: arquitectura anchor-free de una sola etapa

YOLO11-pose (Jocher et al., 2024) pertenece a la undécima generación de la familia YOLO (*You Only Look Once*). A diferencia de los detectores en dos etapas (como

Faster R-CNN), YOLO procesa la imagen completa en una única pasada de red, lo que le confiere su velocidad característica.

Backbone y cuello. El backbone extrae mapas de características en múltiples escalas mediante bloques **C3** (bottleneck convencional con conexiones residuales) y, en YOLO11, bloques **C2f** (*Cross Stage Partial with 2 bottleneck flows*), que mejoran el gradiente y la reutilización de características. El cuello (*neck*) utiliza una **FPN** (*Feature Pyramid Network*) con conexiones PA-Net (*Path Aggregation Network*) para fusionar características de distintas escalas y detectar objetos de múltiples tamaños simultáneamente.

Cabeza de keypoints. Para la variante *-pose*, la cabeza añade una rama dedicada que regresa simultáneamente el *bounding box* de la persona y, para cada una, las coordenadas (u, v) de los 17 keypoints COCO más su confianza. La arquitectura es completamente **anchor-free**: no utiliza cajas ancla predefinidas, sino que predice directamente los desplazamientos desde puntos de la cuadrícula de activación, simplificando el ajuste de hiperparámetros y reduciendo el número de predicciones a evaluar.

Entrenamiento. El modelo público fue preentrenado sobre el conjunto COCO 2017 (val: $AP_{pose} \approx 64\%$ para YOLO11m-pose), empleando pérdidas de *keypoint regression* ponderadas por la confianza de cada punto. Para este trabajo se utiliza el modelo preentrenado sin ajuste fino, lo que implica que el rendimiento depende de la calidad de las representaciones COCO para la anatomía de la mano y muñeca en primer plano.

2.5.2. MediaPipe Hands y Face Mesh: *pipelines* en dos etapas

La arquitectura de MediaPipe (Lugaresi et al., 2019) difiere radicalmente de YOLO: en lugar de un único paso de inferencia, emplea un *pipeline* de dos etapas especializadas para cada tipo de objeto.

Etapas 1 — Detección de ROI. Un detector ligero (**BlazePalm**) localiza la palma de la mano en la imagen completa y devuelve un recuadro orientado (*oriented bounding box*). BlazePalm es una red MobileNet con depthwise separable convolutions optimizada para CPU, capaz de ejecutarse a 56 fps en hardware móvil (F. Zhang et al., 2020).

Etapas 2 — Regresión de landmarks. El recuadro recortado se pasa a una red de regresión (**HandLandmark**) que produce 21 puntos por mano, incluyendo un componente pseudo-3D relativo a la muñeca. De forma análoga, **FaceMesh** localiza 468 puntos faciales a partir de un detector de cara de alta velocidad seguido de una red de regresión densa.

Seguimiento temporal. Entre fotogramas, MediaPipe evita re-ejecutar el detector (costoso) si el recuadro de la iteración anterior se confirma con alta confianza, reutilizando la región de interés. Esto acelera el *pipeline* pero introduce un **riesgo de acumulación de error**: si la mano se mueve rápidamente (como durante un amago de golpe) el tracker puede perder la mano y forzar una re-detección costosa.

Ventaja frente a YOLO en granularidad de mano. MediaPipe Hands expone 21 puntos distribuidos entre muñeca, nudillos y puntas de dedos, mientras que YOLO11-pose expone únicamente la muñeca. Esta granularidad permite a MediaPipe estimar el **centro de la palma** —punto más cercano al rostro en el momento del impacto— en lugar de la muñeca, a costa de mayor sensibilidad a oclusiones parciales.

2.5.3. Azure Kinect Body Tracking SDK: seguimiento nativo por profundidad

El SDK de *Body Tracking* (Büker et al., 2023) es cualitativamente diferente a los dos anteriores: opera sobre el **mapa de profundidad** del sensor ToF, no sobre el flujo RGB.

Red de segmentación sobre profundidad. La entrada es la imagen de profundidad alineada al RGB (640×576 px a 30 fps en NFOV Unbinned). Una red neuronal convolucional — cuya arquitectura exacta no está documentada públicamente, pero cuya estructura general corresponde a una **U-Net** con encoder-decoder (Shotton et al., 2011) — asigna a cada píxel de profundidad una etiqueta de parte corporal (cabeza, tronco, brazo, mano, etc.). A partir de esta segmentación se estiman las posiciones 3D de las 32 articulaciones del esqueleto.

Modos de inferencia y rendimiento. El SDK soporta cuatro motores de inferencia: CUDA (GPU NVIDIA), TensorRT, DirectML y CPU. En este trabajo se usa CUDA con una GTX 1050, que entrega ~ 15 estimaciones articulares por segundo (aunque la cámara graba a 30 fps), una cadencia determinada por el coste de la inferencia en la GPU. Bükér et al. (2023) reportan diferencias significativas ($p < 0,01$) entre modos de procesamiento en las posiciones articulares, incluyendo en el joint `HAND_RIGHT` relevante para este trabajo, lo que subraya la importancia de reportar el modo y ejecutar toda la comparación en el mismo equipo.

Ventaja fundamental frente a YOLO y MediaPipe. Al usar profundidad directamente, el Body Tracking obtiene la posición 3D de la mano **sin reproyección**: no hay propagación del error de profundidad a través de la geometría de cámara. El joint `HAND_RIGHT` se sitúa en la mano (no en la muñeca), reduciendo la distancia medida al rostro y el umbral de proximidad necesario.

2.5.4. Comparación directa de las tres arquitecturas

La Tabla 2.1 resume las diferencias arquitectónicas y operativas de los tres métodos.

Estudios previos han comparado directamente estas familias de algoritmos. Bazarevsky et al. (2020) demuestran que los modelos de regresión directa (como los de MediaPipe) superan a los basados en mapas de calor para keypoints en resoluciones bajas y dispositivos con restricción de cómputo. Mathis y Mathis (2020) analizan sis-

Aspecto	Azure Kinect BT	YOLO11-pose	MediaPipe
Entrada	Profundidad (ToF)	RGB	RGB
Tipo de red	Segmentación corporal (U-Net like)	Detector anchor-free (C2f)	Dos etapas (BlazePalm + regresión)
Punto de mano	Joint HAND_RIGHT	Muñeca (kp 10)	Centro de palma (21 pts)
Keypoints	32 articulaciones 3D	17 keypoints 2D	21 mano + 468 cara
Salida 3D	Directa (mm)	Requiere back-projection	Requiere back-projection
Hardware mín.	GPU CUDA / CPU	CPU (con GPU recomendada)	CPU
FPS típico	~15 fps (GPU)	~21 fps (GPU)	~20 fps (CPU)
Sensibilidad a luz	Alta (IR ambiental)	Baja	Baja
Oclusión mano/cara	Robusta	Media	Alta sensibilidad

Cuadro 2.1: Comparativa de las tres arquitecturas de estimación de pose evaluadas.

temáticamente los factores que determinan la precisión de los estimadores de pose en aplicaciones científicas, concluyendo que la **distancia al punto de referencia** (muñeca vs. palma vs. joint) es la fuente de error sistemático más importante al comparar métodos, hallazgo que este trabajo confirma cuantitativamente (Sección 5). Una comparativa directa entre Azure Kinect Body Tracking y estimadores RGB en condiciones de iluminación adversas puede consultarse en Sarbolandi et al. (2015), que concluye que los sensores ToF degradan su precisión significativamente bajo irradiancia IR ambiental superior a 700 lux — condición reproducida en este trabajo bajo la condición de “luz natural” con ventanales abiertos.

2.5.5. Esquemas de *keypoints* por modelo

Cada algoritmo localiza la mano mediante un esquema de puntos de referencia anatómicos distinto, lo que determina tanto el punto exacto medido como la robustez ante oclusiones. La Figura 2.1 muestra las 32 articulaciones que el Body Tracking del Azure Kinect estima directamente en el espacio 3D a partir del mapa de profundidad. El joint HAND_RIGHT (índice 11) sitúa el punto de referencia en el centro de la mano,

resultado de la segmentación corporal sobre la imagen de profundidad.

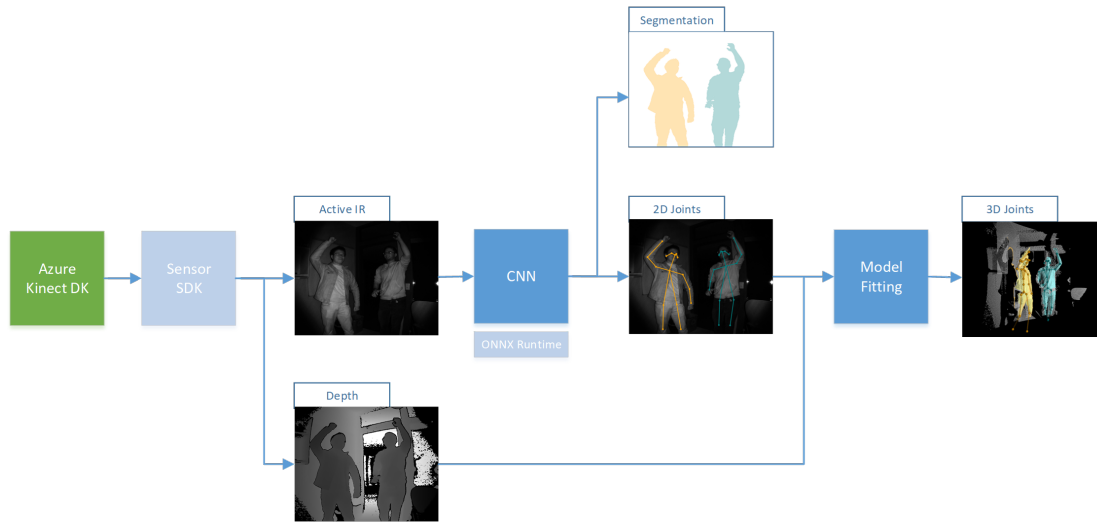


Figura 2.1: Las 32 articulaciones del Azure Kinect Body Tracking SDK estimadas en 3D a partir del mapa de profundidad. El joint `HAND_RIGHT` (n.º 11) es el punto de referencia de la mano en este trabajo.

Para YOLO11-pose, la red estima 17 *keypoints* 2D según el esquema COCO (Lin et al., 2014): el punto de la muñeca derecha corresponde al índice 10 (ver Figura 2.3). Al tratarse de una estimación 2D sobre el frame RGB, se requiere la back-projection descrita en la Sección 4.13.2 para obtener la coordenada 3D.

MediaPipe Hands (F. Zhang et al., 2020) localiza 21 puntos anatómicos de la mano (Figura 2.4): el punto 0 (*wrist*) y el punto 9 (*middle_finger_mcp*) delimitan el centro de la palma, que es el punto más próximo al rostro durante el movimiento de golpe. Al igual que YOLO, la salida es 2D y requiere back-projection.

Sobre el no uso de MediaPipe FaceMesh para la detección de la mano. MediaPipe FaceMesh (Kartynnik et al., 2019) es un modelo diseñado específicamente para la reconstrucción de la superficie facial mediante 468 puntos de referencia distribuidos sobre frente, mejillas, nariz, labios y contorno del rostro. Su entrada es un recorte del rostro (obtenido por un detector de cara previo), por lo que es estructuralmente incapaz de localizar la mano: los landmarks resultantes modelan la geometría de la cara, no

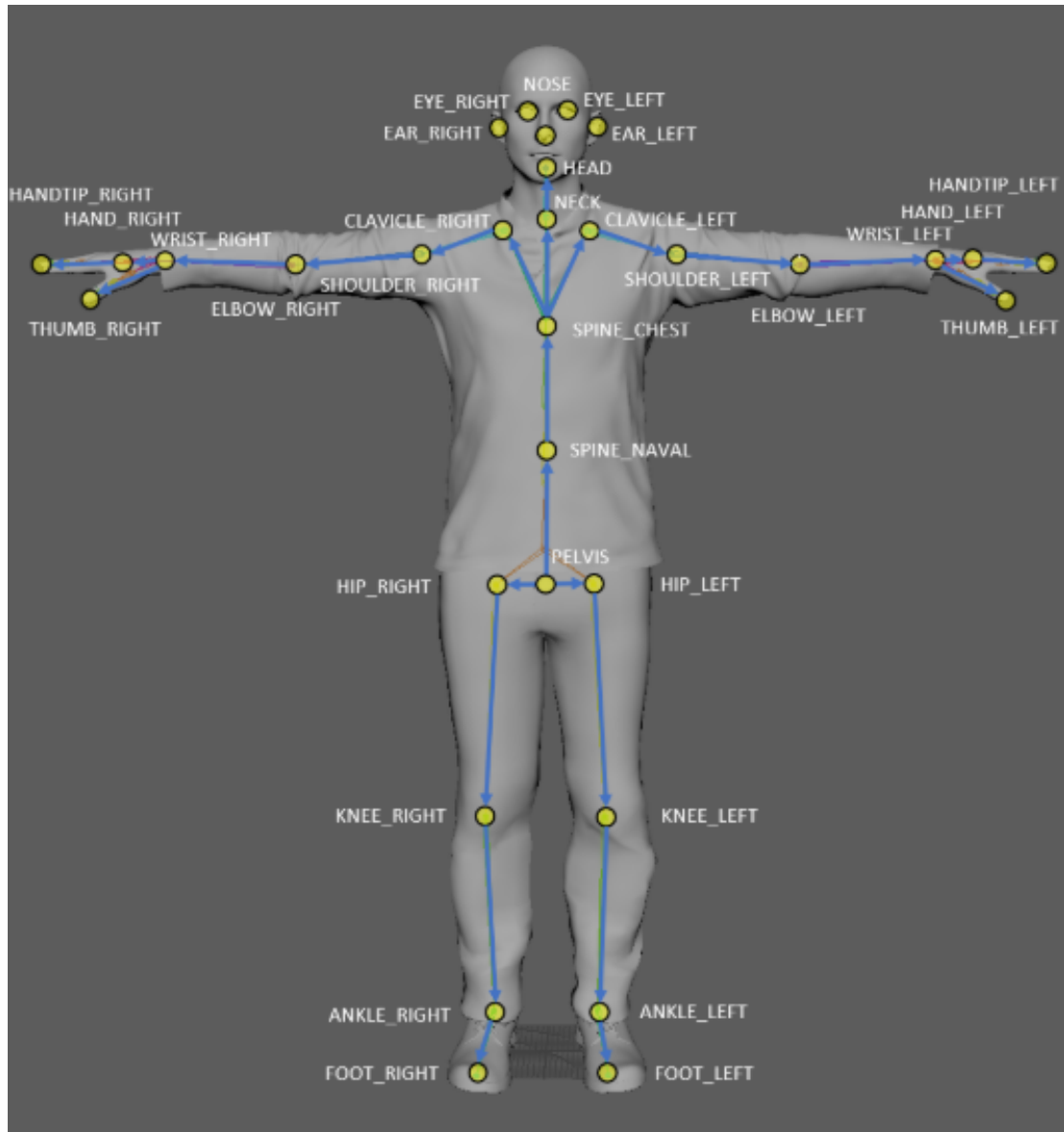


Figura 2.2: Detalle de los joints de mano y muñeca en el esqueleto del Azure Kinect. El SDK devuelve la posición 3D en milímetros directamente, sin necesidad de back-projection.



Figura 2.3: Los 17 keypoints del estándar COCO estimados por YOLO11-pose. El keypoint 10 (muñeca derecha) es el punto de referencia empleado en este trabajo.

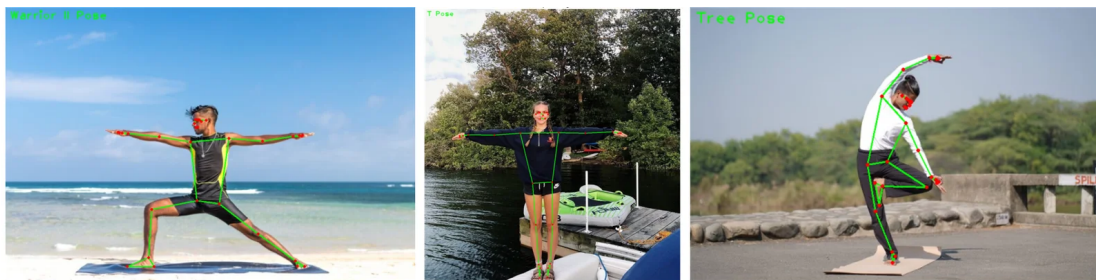


Figura 2.4: Los 21 landmarks anatómicos de MediaPipe Hands. El centro de la palma se aproxima a partir del landmark 0 (*wrist*) y el 9 (*middle_finger_mcp*).

de la extremidad superior (Kartynnik et al., [2019](#)). En este trabajo FaceMesh se utiliza únicamente para delimitar la *superficie facial de referencia* (el objetivo del golpe), mientras que la detección de la mano se delega exclusivamente a MediaPipe Hands. Usar FaceMesh como localizador de la mano sería tanto conceptualmente erróneo como técnicamente inviable, dado que el modelo no fue entrenado con datos de manos y carecería de los puntos articulares necesarios para estimar la posición de la palma o la muñeca (F. Zhang et al., [2020](#)).

Capítulo 3

Estado del Arte

La cuantificación objetiva de las conductas autolesivas (*self-injurious behavior*, SIB) y de las estereotipias motoras en el Trastorno del Espectro Autista (TEA) se ha abordado desde tradiciones muy diversas: las escalas clínicas administradas por observadores, los sensores inerciales adheridos al cuerpo y, más recientemente, los métodos basados en visión por computador. Este capítulo recorre dichos enfoques para situar el aporte de la presente memoria dentro de la literatura y delimitar el vacío que pretende llenar: la medición **no intrusiva** y **métricamente tridimensional** de la cinemática de los golpes autodirigidos a la cabeza. Las arquitecturas concretas de los modelos de estimación de pose y de seguimiento corporal se describen en el Marco Teórico; aquí el énfasis recae en el *trabajo previo* y en las *aplicaciones* desarrolladas por otros investigadores en problemas afines.

3.1. Detección de estereotipias y conductas autolesivas en el TEA

3.1.1. Instrumentos clínicos y sensores llevables

La valoración tradicional de las conductas repetitivas y autolesivas en el TEA se apoya en instrumentos psicométricos administrados por clínicos o cuidadores. La *Repetitive Behavior Scale-Revised* (Bodfish et al., 2000; Lam & Aman, 2007) y la *Aberrant Behavior Checklist* (Aman et al., 1985) permiten caracterizar la variedad y severidad de las conductas repetitivas, mientras que la asociación específica entre el TEA y la autolesión está bien documentada en la literatura clínica (Minshawi et al., 2014). No obstante, estos instrumentos son inherentemente *subjetivos*: dependen del juicio retrospectivo del evaluador, presentan variabilidad inter-observador y no proporcionan una medida continua ni cinemática del fenómeno, lo que dificulta el seguimiento longitudinal fino de la respuesta a un tratamiento.

Para superar esta subjetividad, una línea de investigación consolidada recurre a sensores inerciales adheridos al cuerpo. El trabajo de Goodwin et al. (2011) estableció la viabilidad de detectar automáticamente movimientos motores estereotipados (*stereotypical motor movements*, SMM) mediante acelerómetros, abriendo el camino a una cuantificación objetiva en contextos naturales. Sobre esa base, Plötz et al. (2012) combinaron acelerómetros llevables con aprendizaje automático para segmentar y clasificar conductas-problema severas —incluida la autolesión— en personas con discapacidades del desarrollo, vinculando explícitamente los *wearables* con el SIB. La detección de SMM se refinó posteriormente con representaciones robustas a la orientación del sensor: Großekathöfer et al. (2017) introdujeron descriptores basados en análisis de cuantificación de recurrencia que permiten reconocer múltiples tipos de estereotipia con un único sensor de torso, en un esquema independiente del sujeto.

La irrupción del aprendizaje profundo intensificó esta línea. Mohammadian Rad et al. (2018) propusieron una arquitectura que articula redes convolucionales con me-

moria de largo plazo, incorporando aprendizaje por transferencia para detectar SMM en escenarios longitudinales sobre señales de acelerometría. De forma complementaria, Sadouk et al. (2018) aplicaron redes convolucionales con transferencia a datos de muñeca y torso, abordando explícitamente el reto de la generalización *intra* e *inter*-sujeto que limita la aplicabilidad clínica de estos sistemas. Trabajos más recientes han ampliado el alcance al SIB propiamente dicho: Cantin-Garside et al. (2020) detectan y clasifican distintos tipos de conducta autolesiva en el TEA mediante sensores inerciales y aprendizaje automático, siendo este el antecedente más alineado con el objeto exacto de la presente memoria —los golpes autodirigidos—, si bien con instrumentación adherida al cuerpo. La tendencia actual integra además fisiología: Bahrami Rad et al. (2025) fusionan señales de actividad electrodérmica y temperatura con acelerometría, procesadas mediante redes recurrentes, para anticipar conductas desafiantes —incluido el SIB— en entornos reales.

El denominador común de toda esta familia de trabajos es su carácter **intrusivo**: requieren colocar y mantener uno o más sensores sobre el cuerpo del paciente. En la población con TEA, caracterizada con frecuencia por hipersensibilidad táctil y baja tolerancia a estímulos sensoriales, esta exigencia constituye una limitación práctica de primer orden, pues el propio dispositivo puede alterar la conducta que se desea medir o ser rechazado por el paciente. Es precisamente esta restricción la que motiva el giro hacia los métodos basados en visión.

3.1.2. Enfoques basados en visión y video

La detección de estereotipias mediante video se consolidó a partir del trabajo de Rajagopalan et al. (2013), que introdujo el *Self-Stimulatory Behaviours Dataset* (SSBD): un conjunto de videos de niños grabados en entornos naturales que documentan tres estereotipias características —aleteo de brazos, *head banging* y giros— junto con una línea base de reconocimiento de acciones. El SSBD se convirtió en la referencia obligada del área y en el sustrato sobre el que se edificó buena parte de la literatura posterior.

Antes incluso, Muty y Azizul (2016) habían anticipado el enfoque basado en esqueleto al detectar el aleteo de brazos mediante estimación de pose y una representación articular simplificada, estableciendo el antecedente del paradigma que esta memoria extiende al espacio tridimensional.

Sobre el SSBD y sus derivados, la comunidad ha ido escalando en complejidad metodológica. Negin et al. (2021) contrastaron descriptores clásicos de bolsa de palabras visuales frente a redes recurrentes y convolucionales para el reconocimiento de estereotipias con fines de diagnóstico temprano, ofreciendo una línea base puramente RGB. Y. Zhang et al. (2021) representan el antecedente más cercano al *pipeline* de la presente memoria: extraen esqueletos a partir de video, los depuran y los clasifican mediante redes de memoria de largo plazo para reconocer acciones de niños con TEA, aunque sin incorporar profundidad métrica. Más recientemente, Wei et al. (2023) propusieron una arquitectura de dos etapas que combina extractores convolucionales con modelos temporales sobre una versión curada del SSBD, y Yoo et al. (2024) formularon un marco unificado que integra detección y clasificación de conductas estereotipadas para el cribado del TEA, ilustrando la tendencia hacia sistemas extremo a extremo. La evolución de los datos también ha continuado: Lokegaonkar et al. (2023) ampliaron el SSBD con anotaciones de un experto médico (SSBD+) y un *pipeline* convolucional orientado a despliegue móvil que detecta y categoriza el *head banging*, los giros y el aleteo de brazos.

El hito que mejor evidencia la madurez clínica de esta línea es el estudio a gran escala de Barami et al. (2024), que aplicó estimación de pose y redes convolucionales tridimensionales a un corpus de cientos de evaluaciones y de horas de video de niños con TEA, alcanzando una alta correlación con la anotación manual experta tanto en el conteo como en la duración de los episodios. Este resultado valida la viabilidad de la cuantificación automática de estereotipias por video como instrumento de evaluación clínica, eje central de la presente memoria. Finalmente, el trabajo más directamente afín a esta tesis es el de Eghbalian et al. (2025), que aplica visión por computador y estimación de pose para analizar conductas autolesivas en niños con TEA registradas

en sesiones de terapia, evaluando el desempeño de los modelos de pose en los instantes críticos de la acción. Pese a su cercanía temática, este y los trabajos anteriores comparten una limitación esencial: operan sobre video RGB monocular, sin acceso a la geometría tridimensional métrica de la escena, por lo que pueden *reconocer* la ocurrencia de una conducta pero no *medir* su cinemática con unidades físicas absolutas —velocidad, deceleración y distancia mano-rostro en metros—, que es justamente el aporte de este trabajo.

3.2. Captura de movimiento markerless y cámaras de profundidad en contextos clínicos

La medición cinemática del movimiento humano tuvo durante décadas como patrón de oro los sistemas optoelectrónicos con marcadores reflectantes. La transición hacia la captura *markerless*, impulsada por la visión por computador, ha sido revisada en profundidad por Colyer et al. (2018), que documentan la evolución del análisis de movimiento basado en imágenes y la progresiva integración de métodos avanzados de visión. En el dominio clínico, Wade et al. (2022) ofrecen un marco crítico sobre las aplicaciones y limitaciones de la captura *markerless* —tanto monocular como multicámara— para la biomecánica de la marcha, discutiendo cuestiones de validez y exactitud que son directamente pertinentes a la comparación entre métodos solo-RGB y de profundidad que se realiza en esta memoria.

El sensor empleado en este trabajo, el Azure Kinect DK, cuenta con un cuerpo sólido de estudios de validación frente a sistemas de referencia. Más allá de las evaluaciones generales del dispositivo (Tölgyessy et al., 2021) y de su exactitud y repetibilidad para la medición clínica de la función motora (Bertram et al., 2023), varios trabajos lo han contrastado directamente contra patrones de oro optoelectrónicos. Albert et al. (2020) compararon el Azure Kinect y su predecesor Kinect v2 frente a un sistema Vicon para el análisis de la marcha, evidenciando una mayor exactitud espacial del primero.

Guess et al. (2022) demostraron que una única cámara de profundidad, operada con el *Body Tracking SDK*, alcanza correlaciones elevadas y errores reducidos en los parámetros espaciotemporales de la marcha frente a un sistema óptico multicámara, un resultado relevante porque la presente memoria opera igualmente con una sola cámara.

La validación se ha extendido al miembro superior y al control postural, movimientos más próximos al gesto de golpe que estudia esta tesis. Antico et al. (2021) evaluaron el Azure Kinect DK para la valoración del control postural frente a un sistema con marcadores, y Brambilla et al. (2023) caracterizaron su desempeño en el análisis biomecánico del miembro superior bajo distintas posiciones de cámara, condiciones de iluminación y oclusiones —factores que esta memoria también contempla en su protocolo experimental. Crucialmente, la viabilidad del sensor se ha confirmado no solo en sujetos sanos sino también en población clínica: Cerfoglio et al. (2024) validaron la estimación de parámetros de marcha tanto en individuos sanos como en pacientes hemipléjicos post-ictus frente a un sistema optoelectrónico, reforzando la traslación clínica del enfoque. En conjunto, este cuerpo de evidencia respalda la elección del Azure Kinect como instrumento de medición cinemática de grado clínico y justifica la extensión de su uso a la detección de conductas autolesivas.

3.3. Filtrado de estado para estimación de velocidad

La estimación de velocidad a partir de posiciones medidas por sensores requiere mitigar el ruido, pues derivar numéricamente una posición ruidosa amplifica dicho ruido (Winter, 2009). El *filtro de Kalman* (Kalman, 1960) es el estimador óptimo en el sentido de mínimos cuadrados para sistemas lineales con ruido gaussiano (Bar-Shalom et al., 2001): combina un modelo dinámico de movimiento con las observaciones para producir una estimación del estado (posición y velocidad) de menor varianza que cualquiera de las dos fuentes por separado, y es la base de numerosos sistemas de seguimiento por detección (Bewley et al., 2016). Su calidad depende de que el modelo de movimiento y su ruido de proceso se ajusten a la dinámica observada: un golpe es una

maniobra de alta aceleración, por lo que el ruido de proceso debe dimensionarse a esa escala para no sesgar la velocidad estimada (X. R. Li & Jilkov, 2003). La formulación del filtro y su sintonía se detallan en la Sección 4.12.

3.4. Análisis cinemático de golpes y reconocimiento de acciones mediante visión por computador

La medición de la cinemática de golpes —velocidad de la mano, velocidad de contacto y deceleración— está bien establecida en la biomecánica de los deportes de combate. En boxeo, la captura óptica de movimiento mide velocidades de mano de varios metros por segundo y cuantifica explícitamente el pico de velocidad, la velocidad en el contacto y la tasa de deceleración (Liu et al., 2022); en la literatura, un golpe recto ronda los 5–9 m/s, valores que sirven de **referencia de plausibilidad física** para las velocidades medidas en este trabajo. Cada vez más, estas mediciones se realizan mediante **visión por computador** en lugar de marcadores o sensores inerciales: los sistemas *markerless* estiman la cinemática de la extremidad superior durante el golpeo (Lahkar et al., 2022), y los enfoques híbridos de visión e inercia evalúan velocidad, impulso y fuerza del golpe (Ishac & Eager, 2021). El presente trabajo aplica este mismo paradigma —medición cinemática no intrusiva— a un escenario clínico de amagos de golpe autodirigidos.

La medición fiable de la cinemática de un golpe plantea un reto adicional cuando el movimiento es rápido: el desenfoque por movimiento y el submuestreo temporal degradan la estimación de velocidad y aceleración. Xu et al. (2020) abordan explícitamente la captura tridimensional de movimientos humanos de alta velocidad, ilustrando la dificultad de preservar la fidelidad cinemática en gestos veloces —un problema análogo al de los amagos mano-rostro— y motivando el recurso a un filtrado de estado adecuado. En el ámbito específico de los impactos craneales, Hasija y Takhounts (2022) estiman la historia temporal de la cinemática angular de la cabeza directamente desde

video mediante aprendizaje profundo, prescindiendo de instrumentación adherida; es un antecedente directo de la idea de medir la cinemática de un impacto por visión en lugar de por sensores, aunque su dominio sea la biomecánica de choques y no el TEA.

El reconocimiento de la acción que precede al impacto se apoya, a su vez, en la representación del cuerpo mediante esqueletos tridimensionales. El *benchmark* fundacional de Shahroudy et al. (2016) estableció el estándar de datos para el reconocimiento de acciones a partir de esqueletos capturados con cámaras de profundidad afines al Azure Kinect, y la evolución posterior de los métodos —de las redes recurrentes y convolucionales a las redes neuronales de grafos— ha sido sistematizada por Ren et al. (2024). Este cuerpo de trabajo sustenta la elección del esqueleto tridimensional como representación robusta y no intrusiva para reconocer el gesto de golpe, en la que se asienta el método propuesto.

Un problema metodológicamente análogo, y especialmente ilustrativo, es la detección de caídas mediante sensores de profundidad. Gasparrini et al. (2014) construyeron un sistema que segmenta al sujeto por análisis de profundidad y dispara una alarma al detectar la aproximación al suelo, demostrando la viabilidad de la visión 3D no intrusiva y respetuosa con la privacidad para detectar eventos de contacto. Abobakr et al. (2018) llevaron este planteamiento al aprendizaje profundo espacio-temporal, aprendiendo de forma conjunta la representación visual y la dinámica del movimiento sobre datos RGB-D. La analogía es directa: tanto la caída como el golpe autodirigido son eventos rápidos cuya detección se beneficia de combinar información de profundidad con un modelado temporal del movimiento, exactamente el principio sobre el que esta memoria identifica el instante de máxima aproximación mano-rostro.

3.5. Síntesis y posicionamiento

La revisión precedente revela tres tradiciones que convergen, pero ninguna resuelve por completo el problema abordado en esta memoria. Los instrumentos clínicos (Aman et al., 1985; Bodfish et al., 2000; Lam & Aman, 2007) aportan validez ecológica pero

son subjetivos y carentes de resolución cinemática. Los sensores llevables (Bahrami Rad et al., 2025; Cantin-Garside et al., 2020; Goodwin et al., 2011; Mohammadian Rad et al., 2018) ofrecen una medición objetiva y continua, e incluso abordan el SIB directamente, pero a costa de la intrusividad —un obstáculo crítico ante la frecuente hipersensibilidad sensorial del TEA. Los métodos basados en visión (Barami et al., 2024; Eghbalian et al., 2025; Negin et al., 2021; Rajagopalan et al., 2013; Wei et al., 2023) eliminan la intrusividad y han demostrado madurez clínica para *reconocer* estereotipias, pero operan casi exclusivamente sobre video RGB monocular, por lo que carecen de la geometría tridimensional métrica necesaria para cuantificar la *intensidad* física de un golpe.

El vacío que esta tesis llena se sitúa precisamente en la intersección de las tres líneas. Por un lado, hereda de la captura *markerless* validada clínicamente (Brambilla et al., 2023; Cerfoglio et al., 2024; Guess et al., 2022) la fiabilidad métrica del Azure Kinect como instrumento de medición. Por otro, adopta del análisis cinemático de golpes (Lahkar et al., 2022; Liu et al., 2022) las métricas de velocidad, deceleración y contacto, y de la biomecánica de impactos por visión (Hasijsa & Takhoumts, 2022) la noción de medir el evento sin sensores adheridos. Finalmente, lo aplica al objeto exacto que la literatura de visión en TEA solo reconoce cualitativamente: los golpes autodirigidos a la cabeza. El resultado es, hasta donde alcanza esta revisión, la primera propuesta de medición **no intrusiva, tridimensional y cinemáticamente específica** de los impactos autodirigidos en pacientes con TEA, contrastada además frente a líneas base solo-RGB para acotar el valor añadido de la profundidad.

Capítulo 4

Materiales y Métodos

4.1. Arquitectura del sistema

La Figura 4.1 resume el flujo de procesamiento de cada grabación: desde la captura RGB+profundidad hasta el evento de impacto medido, pasando por los tres algoritmos de seguimiento, la homogeneización a un sistema 3D común, el filtrado de estado y la detección cinemática. Los tres métodos convergen a coordenadas tridimensionales comparables (mm, en el marco de la cámara de profundidad): el Azure Kinect directamente desde el sensor de profundidad, y YOLO11-pose y MediaPipe mediante *back-projection* de sus *keypoints* 2D con el mapa de profundidad alineado.

4.2. Equipamiento

El dispositivo de captura es la cámara **Azure Kinect DK**, que integra un sensor RGB de hasta 1280×720 px y un sensor de profundidad por tiempo de vuelo (Hansard et al., 2012). El procesamiento se realizó en un único equipo con la siguiente configuración: CPU Intel Core de 10ª generación, GPU NVIDIA GeForce GTX 1050 (modo de *Body Tracking* CUDA) y 16 GB de memoria RAM. Esta unicidad sigue la recomendación de reproducibilidad de Bükér et al. (2023), y la GTX 1050 acota la *cota*

Pipeline de detección de impactos

(por grabación)

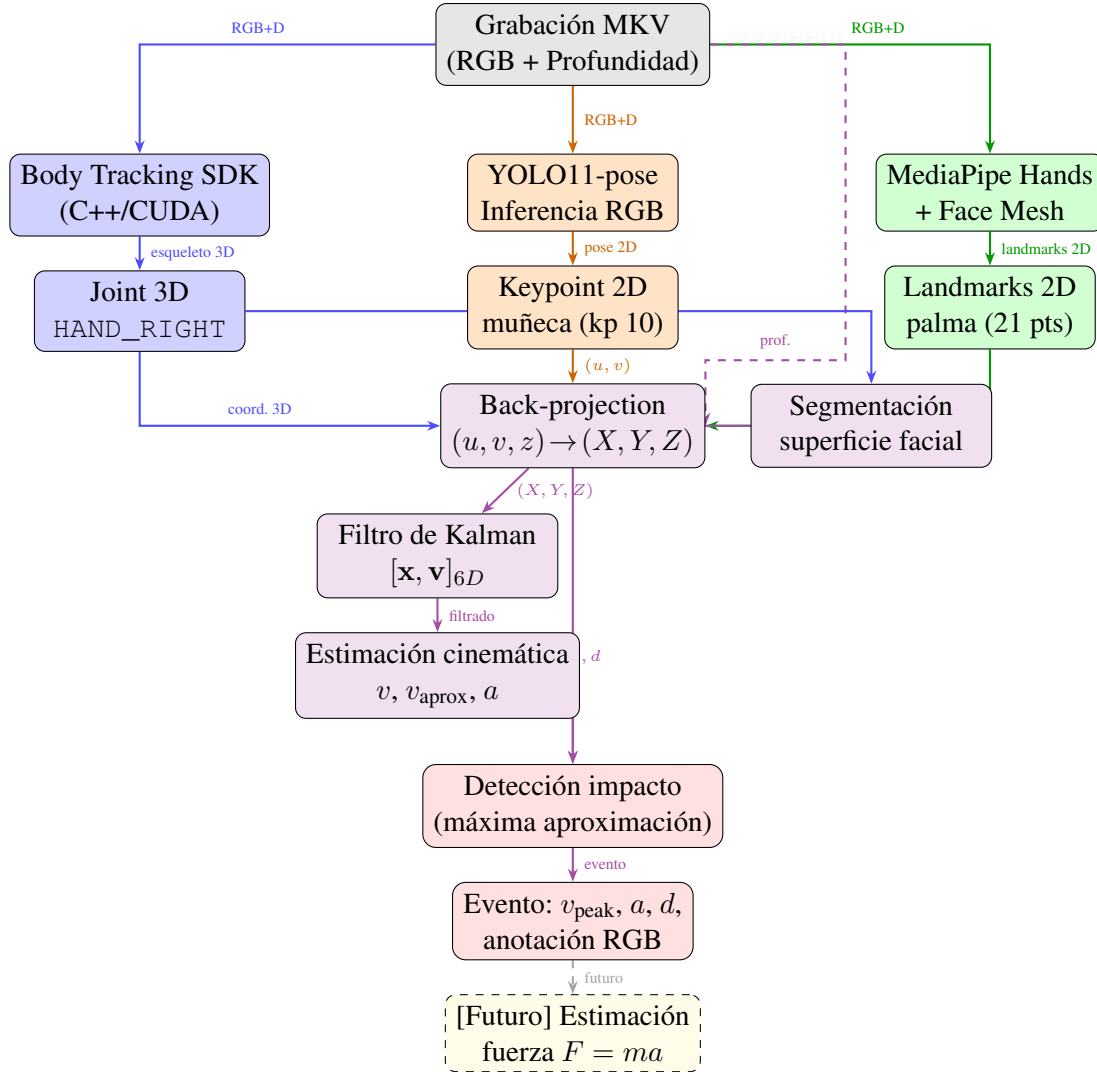


Figura 4.1: Pipeline de procesamiento por grabación, de la captura al evento de impacto medido. Los tres algoritmos de seguimiento (azul: Kinect; naranja: YOLO11-pose; verde: MediaPipe) convergen a una representación 3D común mediante back-projection; sobre ella operan el filtro de Kalman, la estimación cinemática y el detector de impacto. El bloque punteado (estimación de fuerza) es trabajo futuro.

inferior de hardware compatible con el SDK en modo CUDA, lo que es relevante para la viabilidad clínica del sistema (entornos no especializados en cómputo). El sensor de profundidad presenta un error aleatorio con desviación estándar ≤ 17 mm y un error sistemático típico < 11 mm + 0,1 % de la distancia (Tölgyessy et al., 2021), valores que se incorporan al análisis de incertidumbre (Sección 4.17).

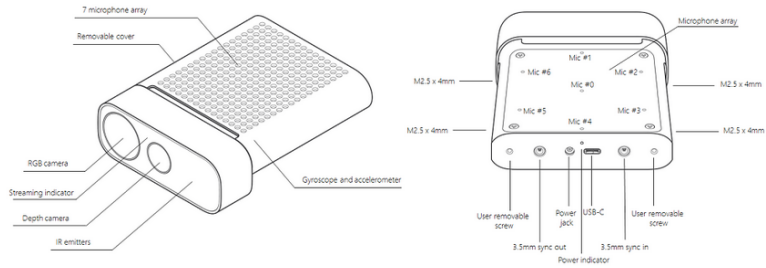


Figura 4.2: Sensores principales de la cámara Azure Kinect DK.

Mode	Resolution	Fov	FPS	Operating range*	Exposure time
NFOV unbinned	640x576	75°x65°	0, 5, 15, 30	0.5 - 3.86 m	12.8 ms
NFOV 2x2 binned (SW)	320x288	75°x65°	0, 5, 15, 30	0.5 - 5.46 m	12.8 ms
WFOV 2x2 binned	512x512	120°x120°	0, 5, 15, 30	0.25 - 2.88 m	12.8 ms
WFOV unbinned	1024x1024	120°x120°	0, 5, 15	0.25 - 2.21 m	20.3 ms
Passive IR	1024x1024	N/A	0, 5, 15, 30	N/A	1.6 ms

Figura 4.3: Detalles de configuración para cada modo de profundidad del sensor.

4.3. Configuración del modo de profundidad

El sensor de profundidad admite varios modos que equilibran campo de visión (FOV), resolución y rango. Se evaluaron tres:

- **NFOV Unbinned** ($75^\circ \times 65^\circ$, 640×576): campo estrecho y máxima resolución; mayor densidad de puntos sobre el rostro.

- **NFOV 2x2 Binned** ($75^\circ \times 65^\circ$, 320×288): combina 2×2 píxeles, reduciendo ruido y carga computacional a costa de resolución.
- **WFOV 2x2 Binned** ($120^\circ \times 120^\circ$, 512×512): campo de visión amplio, útil para sujetos en movimiento, con rango de 0,25–2,88 m y tiempo de exposición corto (12.8 ms) que minimiza el desenfoque por movimiento.

NFOV frente a WFOV. El modo NFOV ($75^\circ \times 65^\circ$) concentra el haz infrarrojo en un sector angular estrecho, lo que se traduce en **mayor alcance** (~ 3.86 m en Unbinned, ~ 5.46 m en 2×2) y **menor ruido por píxel**, al repartirse la misma potencia óptica sobre menor área. WFOV ($120^\circ \times 120^\circ$) cubre mucho más espacio lateral, pero a costa de alcance (~ 2.88 m en 2×2) y de una distorsión angular mayor en los bordes. En este trabajo el paciente se ubica a ~ 1 m de la cámara y centrado en el campo, escenario en el que la amplitud angular de WFOV no aporta beneficio frente a la mayor densidad de puntos faciales y menor ruido de NFOV.

Unbinned frente a 2×2 Binned. El *binning* agrupa cada bloque 2×2 de píxeles del sensor en uno solo, lo que (i) reduce la resolución a la mitad en cada eje, (ii) aumenta la SNR de cada píxel resultante por promedio ($\sim \sqrt{4} \times$ menos ruido aleatorio), (iii) amplía el alcance máximo y (iv) reduce el coste computacional aguas abajo. Su contrapartida es la pérdida de detalle espacial, que en este escenario clínico se traduce en una nube de puntos del rostro menos densa y, por tanto, en una representación más gruesa de su contorno.

Por qué se descartó WFOV Unbinned. El cuarto modo del sensor, WFOV Unbinned (1024×1024), **está limitado a 15 fps** por el ancho de banda del sensor: la mayor resolución implica un tiempo de exposición y de lectura que impide sostener los 30 fps objetivo. Para detectar impactos —eventos de centenas de milisegundos— esta cadencia es insuficiente: reduce a la mitad el muestreo temporal de la cinemática y degrada la

Comparativa de modos de profundidad del Azure Kinect (vista cenital, paciente a ~1 m)

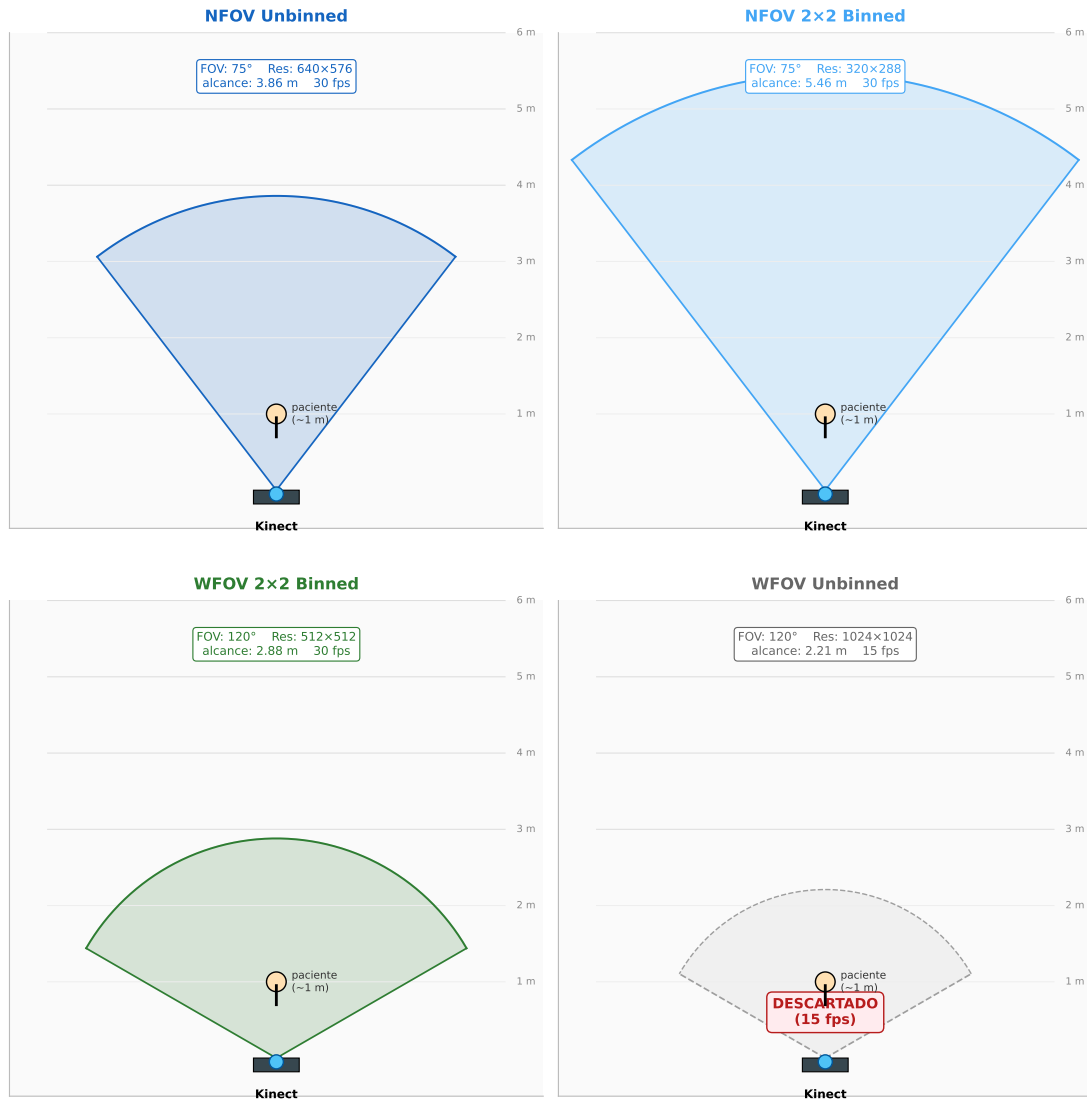


Figura 4.4: Vista cenital de los cuatro modos de profundidad del Azure Kinect DK. Los tres seleccionados (NFOV Unbinned, NFOV 2×2 Binned, WFOV 2×2 Binned) operan a 30 fps y cubren rangos y campos de visión complementarios. WFOV Unbinned, en gris, se descarta por estar limitado a 15 fps.

estimación del pico de velocidad en la fase de aproximación. A ello se suma un **alcance más corto** (~ 2.21 m), que no es restrictivo en este montaje pero confirma su nicho fuera de él. Por estas razones se mantuvieron los tres modos a 30 fps (NFOV Unbinned, NFOV 2×2 y WFOV 2×2), suficientes para cubrir el espacio de diseño relevante: *resolución/ruido* (Unbinned vs 2×2) y *cobertura angular* (NFOV vs WFOV).

Tasa efectiva del seguimiento corporal. Todas las grabaciones se realizaron a 30 fps. Cabe notar que, aun a esa cadencia, el SDK de *Body Tracking* del Azure Kinect entrega del orden de 15 estimaciones articulares por segundo (Büker et al., 2023) debido al coste de la inferencia por *frame*, mientras que los 30 fps se sostienen sobre la cámara RGB y, por tanto, sobre los métodos YOLO11-pose y MediaPipe. Este desbalance se cuantifica en el análisis de coste computacional (Sección 5).

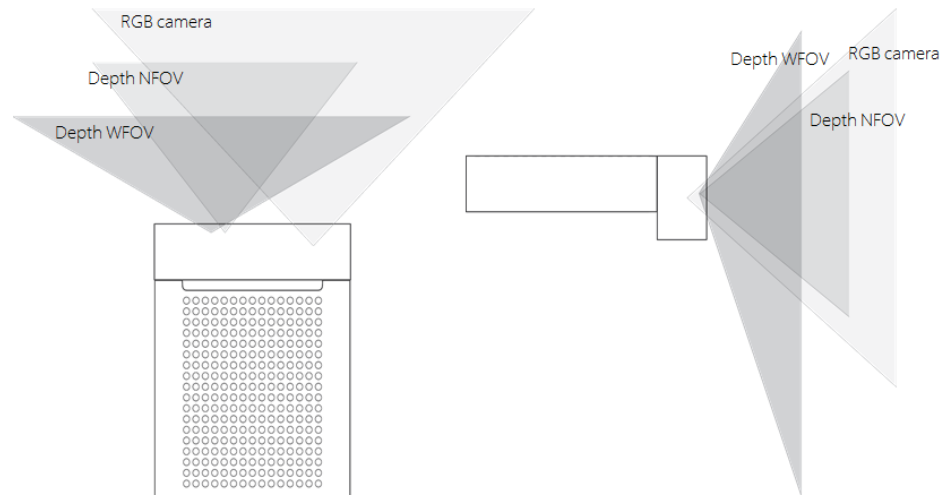


Figura 4.5: Cobertura según el modo de profundidad desde una perspectiva lateral y superior.

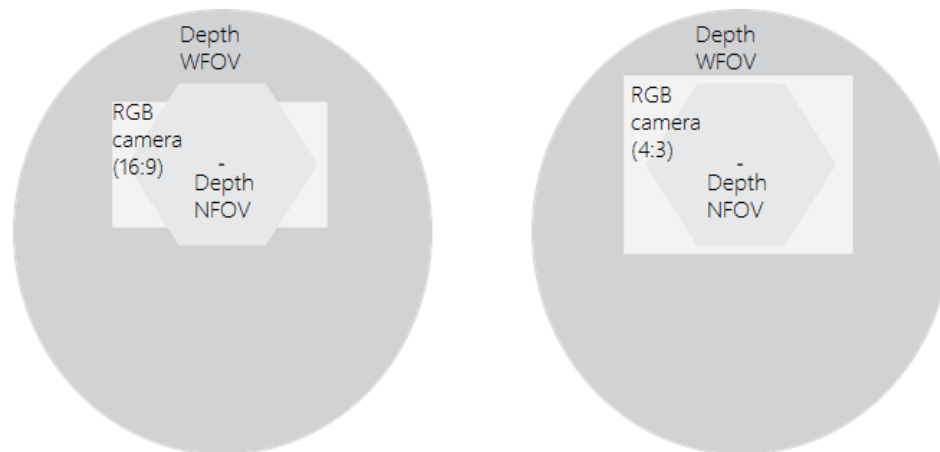


Figura 4.6: Vista de cada modo de profundidad desde el frente, a una distancia de dos metros.

4.4. Seguridad del láser ToF y efectos de la iluminación infrarroja

4.4.1. Clasificación de seguridad ocular

El sensor de profundidad del Azure Kinect DK emite un patrón infrarrojo de tiempo de vuelo en la banda de 850 nm mediante una fuente de luz **Clase 1** según el estándar internacional IEC 60825-1:2014 (International Electrotechnical Commission, 2014), la categoría de mayor seguridad establecida por dicha norma. Un dispositivo Clase 1 es **intrínsecamente seguro** bajo cualquier condición de uso previsible, incluida la exposición directa y prolongada al haz. El dispositivo cuenta adicionalmente con la certificación FDA (*Food and Drug Administration* de EE. UU.) bajo el estándar 21 CFR 1040.10 para productos láser.

Esto implica que el sistema es completamente seguro para su uso con **niños y adultos** en entornos clínicos, sin requerir equipos de protección ocular ni protocolos especiales de manipulación, condición indispensable para su despliegue en sesiones de evaluación del TEA.

4.4.2. Efectos de la iluminación ambiental en sensores ToF

Los sensores ToF se basan en la correlación de la luz emitida con la reflejada para medir el tiempo de vuelo $\tau = 2d/c$ (y, por tanto, la distancia d). La señal útil es la componente modulada del flujo reflejado. La luz ambiental —tanto solar como artificial de alta potencia— introduce una componente **continua no modulada** que satura los fotodetectores, reduciendo su margen dinámico para la señal de correlación (Foix et al., 2011).

Mecanismo de degradación. Bajo alta irradiancia IR ambiental, el fotodetector del ToF opera en saturación parcial, lo que (i) reduce la amplitud de la señal de correlación, aumentando la varianza de la estimación de τ ; (ii) introduce sesgo sistemático en los píxeles próximos a la saturación (*wiggling error*); y (iii) puede generar píxeles sin dato (*flying pixels*) en zonas de alta reflectividad o bajo la irradiancia directa del sol (Kolb et al., 2010). El efecto es especialmente marcado en superficies de baja reflectividad IR (piel, cabello oscuro), que ya de por sí devuelven poca señal.

Evidencia en el Azure Kinect. Tölgyessy et al. (2021) caracterizan la dependencia del error del Azure Kinect con la iluminación: bajo luz solar directa el error aleatorio puede multiplicarse por tres respecto a condiciones interiores controladas. Sarbolandi et al. (2015) comparan sistemáticamente sensores ToF y de luz estructurada bajo distintas condiciones y concluyen que la irradiancia ambiental superior a ~ 700 lux degrada la calidad de la nube de puntos de forma estadísticamente significativa.

Implicación para el diseño experimental. Esta limitación física motiva la evaluación bajo las tres condiciones de iluminación de esta memoria (Sección 4.3): luz natural con ventanales cerrados, luz natural con ventanales abiertos (máxima degradación previsible) y luz artificial controlada. Los resultados confirman que la iluminación artificial dirigida maximiza la calidad de la nube de puntos y el número de detecciones fiables, respaldando la recomendación de operar en entornos con iluminación controla-

da y aislados de luz exterior.

4.5. API Python para el SDK de Azure Kinect

El SDK de Azure Kinect (*Azure Kinect Sensor SDK* y *Body Tracking SDK*) está implementado de forma nativa en C++, exponiendo una interfaz C pública a través de las bibliotecas `k4a.dll` y `k4abt.dll` en Windows. Para operar el sensor y procesar sus datos desde Python —el lenguaje elegido en este trabajo por su ecosistema de análisis y visión— se utiliza la biblioteca de código abierto **pykinect_azure** (Gorordo, 2021), que envuelve la interfaz C nativa mediante el módulo `ctypes` de la biblioteca estándar de Python.

Capa de envoltimiento (*wrapper*). `ctypes` permite llamar funciones exportadas por una DLL directamente desde Python sin compilar un módulo de extensión. Cada tipo de dato del SDK (`k4a_capture_t`, `k4a_image_t`, `k4abt_frame_t`) se mapea a una estructura Python equivalente, y las llamadas de función se traducen con conversión automática de tipos. Esta arquitectura implica que **no hay sobrecarga de copia de datos**: el buffer de la imagen de profundidad se accede directamente desde NumPy a través de un puntero compartido, lo que mantiene la latencia de procesamiento por debajo del umbral de tiempo real.

Flujo de datos en la captura. El flujo de datos sigue la arquitectura cliente–servidor del SDK (Figura 4.7): el *host* solicita una captura, el SDK devuelve un objeto `k4a_capture_t` que contiene simultáneamente el frame RGB, el mapa de profundidad, el mapa IR activo y el frame de IMU; sobre ese objeto se invoca el motor de Body Tracking que procesa la imagen de profundidad y produce un `k4abt_frame_t` con los skeletons detectados.

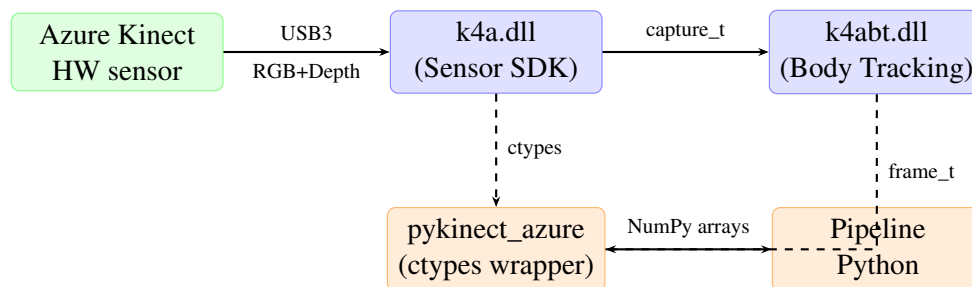


Figura 4.7: Flujo de datos del SDK de Azure Kinect: el hardware entrega captura RGB+profundidad al SDK nativo (C++); pykinect_azure la envuelve mediante ctypes y expone arrays NumPy al pipeline Python.

Estructura del código del sistema. El sistema está organizado en los siguientes módulos principales:

config.py Parámetros globales: umbrales de proximidad (U_{prox} , uno por método), parámetros del filtro de Kalman (q , σ_z^2 , $\sigma_{v,0}^2$), modo de profundidad, condición de iluminación y directorio de salida.

tracking/kinect.py

Clase `KinectTracker`: apertura del dispositivo o del fichero MKV mediante `pykinect_azure`, bucle de captura, extracción del joint `HAND_RIGHT` y de los puntos faciales desde el *skeleton*. Incluye lógica de re-inicialización del Body Tracking ante pérdidas de detección.

tracking/yolo.py

Clase `YoloTracker`: carga del modelo YOLO11-pose (Ultralytics), inferencia sobre el frame RGB, extracción del keypoint 10 (muñeca derecha) con su confianza, y back-projection al sistema 3D del Kinect.

tracking/mediapipe_tracker.py

Clase `MediaPipeTracker`: inicialización de MediaPipe Hands

y FaceMesh, extracción del landmark de muñeca y nudillos para estimar el centro de palma, y back-projection.

<code>kalman.py</code>	Clase <code>KalmanFilter3D</code> : implementación del filtro de velocidad constante 6D (posición + velocidad) con los parámetros de la Sección 4.12; métodos <code>predict(dt)</code> y <code>update(z)</code> .
<code>detector.py</code>	Clase <code>ImpactDetector</code> : mantiene los <i>buffers</i> de historial $(t, \mathbf{x}, \mathbf{v}, d, v_{\text{aprox}})$ e implementa el criterio de confirmación del Algoritmo 1. Genera la anotación del frame RGB y serializa el evento.
<code>process_video.py</code>	Orquestador principal: itera sobre el corpus de MKV, instancia los trackers y el detector, y centraliza el guardado de resultados en CSV.

4.6. Condiciones de iluminación

Dado que la calidad de la nube de puntos depende fuertemente de la luz, cada configuración se evaluó bajo tres condiciones:

1. **Luz natural (cerrado):** ventanas cerradas, con escasa entrada de luz natural diurna.
2. **Luz natural:** dos ventanales abiertos por completo, máxima entrada de luz natural.
3. **Luz artificial:** dos focos tipo CFL/LED apuntando directamente al rostro, simulando un entorno controlado.

4.7. Calibración empírica y artefactos en la segmentación

La validación empírica del sistema permite calibrar la conversión $3D \leftrightarrow 2D$ y caracterizar los artefactos típicos de la segmentación mano–rostro basada en distancia, que se ilustran a continuación.

4.7.1. Validación de la proyección $3D \leftrightarrow 2D$

Entendiendo y aplicando la proyección inversa se obtienen vistas que relacionan coordenadas obtenidas en el espacio 3D con el plano de la imagen: el **punto rojo** representa la ubicación en 2D del punto de impacto en la superficie del rostro, el **punto azul** la coordenada 2D de la mano, y el rostro se segmenta mediante un filtro de gradiente aplicado a la nube de profundidad. En esta iteración se definió un umbral de proximidad de 50 mm.

Estos experimentos motivaron el paso al sistema de *landmarks* faciales densos (Face Mesh / Body Tracking) para delimitar la superficie del rostro con mayor robustez a las oclusiones de la mano.

4.7.2. Pruebas preliminares de configuración de modo de profundidad

En esta iteración se evaluó el funcionamiento de los modos **WFOV 2×2**, **NFOV 2×2** y **NFOV Unbinned** bajo tres niveles de luminosidad:

1. **Habitación con luz tenue de día:** ventanas cerradas, pequeña cantidad de luz natural.
2. **Habitación con máxima entrada de luz natural:** dos ventanales completamente abiertos.



Figura 4.8: En esta captura se entregó una distancia de 42 mm entre el punto rojo y el azul, valor razonable. Se evidencia un problema: parte de la mano, al estar muy cerca del rostro, se toma como parte de la región segmentada, desplazando el punto rojo más allá de lo deseado.



Figura 4.9: En este ejemplo es más evidente cómo la mano se confunde con la región del rostro. Distancia entre puntos: 49 mm.



Figura 4.10: Otro ejemplo donde la mano, al ser capturada a alta velocidad, es reconocida erróneamente como parte del rostro.

3. **Luz artificial:** dos focos tipo CFL/LED apuntando directamente al rostro, simulando un entorno controlado.

Los resultados preliminares mostraron que la combinación de **NFOV Unbinned** con iluminación artificial ofrece la mayor densidad de nube de puntos sobre el rostro, hipótesis que el experimento sistemático de la Sección 5 confirma cuantitativamente.

4.8. Diseño experimental

Se grabó al sujeto sentado frente a la cámara replicando golpes hacia distintas zonas del rostro y la cabeza: mejilla derecha, mejilla izquierda, boca, frente y zonas superiores izquierda y derecha (próximas a la nuca). El diseño factorial completo —seis zonas faciales, tres modos de profundidad y tres condiciones de iluminación— contemplaba $6 \times 3 \times 3 = 54$ grabaciones. Sin embargo, el corpus efectivamente disponible para el análisis consta de **46 grabaciones**: los modos NFOV Unbinned y NFOV 2×2 están

completos (cubren las seis zonas en las tres condiciones de iluminación, 18 grabaciones cada uno), mientras que del modo WFOV 2×2 sólo se conservan **10** de las 18 previstas. Las ocho grabaciones WFOV restantes se perdieron por corrupción de los archivos durante la descarga desde el sensor, y no fue posible regrabarlas porque la cámara Azure Kinect estuvo disponible únicamente durante un **período de préstamo limitado**, ya concluido al detectarse la pérdida. Por este motivo, todo análisis por modo de profundidad debe interpretar los resultados de WFOV 2×2 como un subconjunto parcial ($n = 10$) y no directamente comparable en tamaño con los modos NFOV. Cada grabación disponible fue procesada por los tres algoritmos (Azure Kinect, YOLO11-pose y MediaPipe) y, en el caso del Kinect, con y sin filtro de Kalman, lo que permite tanto la comparación entre algoritmos como el estudio de ablación del filtro.

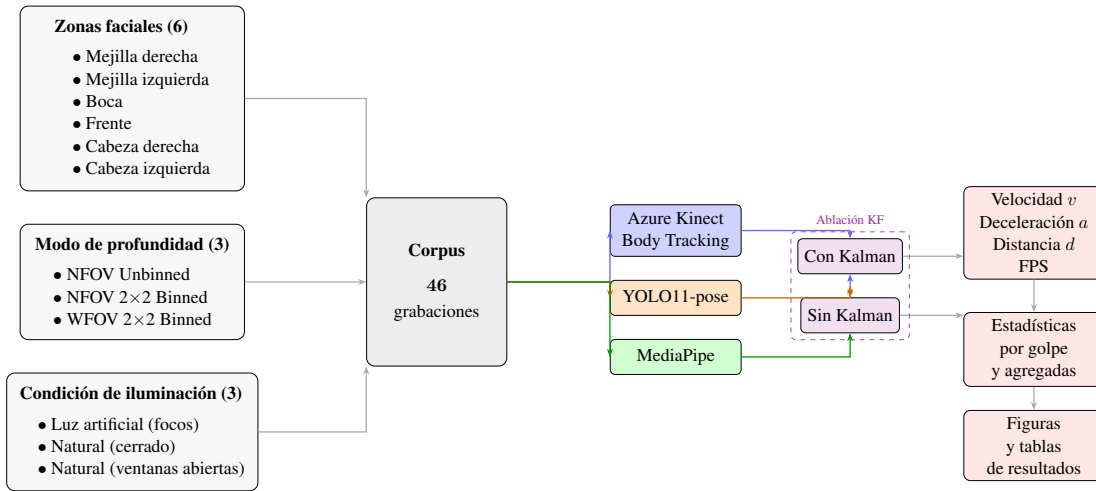


Figura 4.11: Diseño experimental y flujo de análisis. Los tres factores —zonas faciales (6), modos de profundidad (3) e iluminación (3)— definen un diseño factorial de $6 \times 3 \times 3 = 54$ grabaciones; el corpus efectivamente disponible es de 46 grabaciones, pues del modo WFOV 2×2 sólo se conservan 10 de las 18 previstas (las restantes se perdieron por corrupción de archivos durante el período de préstamo limitado del sensor). Cada grabación se procesa con los tres algoritmos; para Azure Kinect se activa y desactiva el filtro de Kalman (ablación KF). Las métricas por golpe y las estadísticas agregadas alimentan las figuras y tablas de resultados.

4.9. Definición de impacto

Un aspecto metodológico esencial: en todas las grabaciones el sujeto **simula** el golpe y **detiene la mano antes del contacto físico** real. Por lo tanto, no existe un contacto que detectar; el evento medible es el **punto de máxima aproximación** mano→rostro, donde la mano decelera y/o invierte su trayectoria en la vecindad de la superficie facial. Esta observación motiva dos decisiones de diseño:

1. **Umbral de proximidad** (no contacto cero): se considera un golpe cuando la mano entra en una vecindad del rostro tras haberse aproximado a velocidad significativa.
2. **Superficie 3D del rostro como referencia**: la distancia se mide contra una representación de la superficie facial (nube de puntos o malla), no contra un único

punto, dotando al casi-impacto de una referencia espacial robusta a la pose de la cabeza.

Formalmente, se detecta proximidad cuando

$$d_{\text{mano-rostro}} = \min_{p \in \mathcal{F}} \|\mathbf{x}_{\text{mano}} - p\| < U_{\text{prox}},$$

donde $\mathcal{F}$ es el conjunto de puntos faciales y U_{prox} el umbral de proximidad. Sin embargo, la proximidad por sí sola no constituye un golpe (la mano podría pasar lentamente cerca); la confirmación cinemática se describe en la Sección 4.15.

4.10. Detección de movimiento

El sistema determina si una articulación está en movimiento mediante la distancia euclidiana 3D entre sus coordenadas en *frames* consecutivos. Se activa el registro de la trayectoria cuando dicha distancia supera un umbral U_{mov} calibrado como $U_{\text{mov}} = k \sigma$, donde σ es la desviación estándar de las coordenadas durante una calibración inicial con la mano en reposo y k un factor de sensibilidad (típicamente $k = 2$). La distancia entre dos posiciones tridimensionales es

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}.$$

La Figura 4.12 ilustra el procedimiento sobre el esqueleto entregado por el *Body Tracking*: entre dos *frames* consecutivos se compara la posición de la articulación de interés (la mano derecha) y se calcula el desplazamiento euclidiano Δd . Cuando $\Delta d > U_{\text{mov}}$ se considera que la articulación está en movimiento y se inicia el registro de su trayectoria.

Desplazamiento euclidiano de la mano: $\Delta d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \approx 238 \text{ mm}$ · sujeto sentado, rango útil $\approx 1.20 \text{ m}$

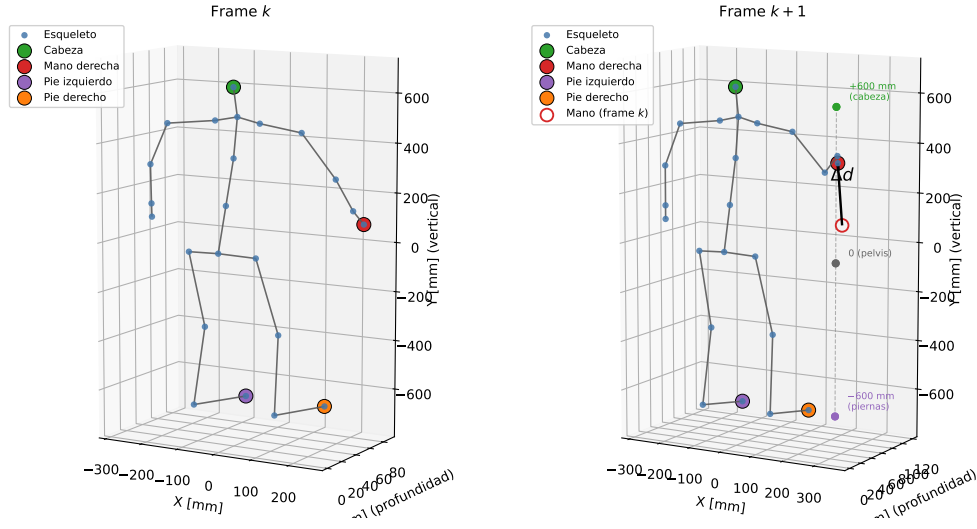


Figura 4.12: Desplazamiento euclidiano de la mano derecha entre dos *frames* consecutivos del esqueleto. En el *frame* $k + 1$ la mano se ha desplazado hacia la cabeza (amago de golpe); el círculo hueco marca su posición en el *frame* k y la flecha representa el vector de desplazamiento Δd , métrica que, comparada con el umbral U_{mov} , determina si la articulación está en movimiento. El sujeto se encuentra **sentado** frente a la cámara; el origen de coordenadas ($Y = 0$) se sitúa en la pelvis, de modo que el rango útil de altura asciende a $\approx 1,20 \text{ m}$ (+600 mm hacia la cabeza, -600 mm hacia las piernas desde el centro de referencia).

4.11. Segmentación de la región facial

La superficie del rostro se obtiene recortando la nube de puntos alrededor de la articulación del cuello, que el SDK de *Body Tracking* provee en tiempo real. Se aplican filtros espaciales relativos a la posición del cuello: en el eje vertical se conservan los puntos por encima del cuello; en los ejes horizontal y de profundidad se definen márgenes ($\pm 100 \text{ mm}$ en x , 200 mm en z hacia la cámara). Estos valores incluyen de forma fiable la región de la cabeza en la mayoría de las poses.

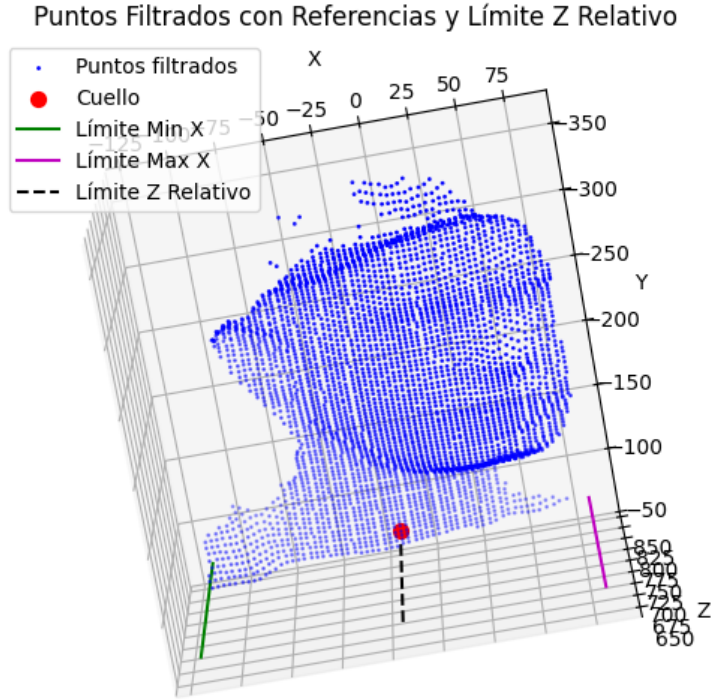


Figura 4.13: Segmentación de la nube de puntos de la región del rostro mediante los umbrales espaciales relativos a la articulación del cuello.

4.12. Filtro de Kalman para posición y velocidad

Para estimar la posición y velocidad de la mano en 3D se define el vector de estado

$$\mathbf{x} = \begin{bmatrix} x & y & z & v_x & v_y & v_z \end{bmatrix}^T,$$

con dinámica de **velocidad constante** $\mathbf{x}_{k+1} = \mathbf{F} \mathbf{x}_k + \mathbf{w}_k$ y observación de la posición $\mathbf{z}_k = \mathbf{H} \mathbf{x}_k + \mathbf{v}_k$, donde $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q})$ y $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R})$. Las matrices de transición y observación son

$$\mathbf{F} = \begin{bmatrix} \mathbf{I}_3 & \Delta t \mathbf{I}_3 \\ \mathbf{0} & \mathbf{I}_3 \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \end{bmatrix},$$

con Δt el intervalo *físico* entre observaciones. La predicción y la actualización siguen las ecuaciones estándar del filtro de Kalman:

$$\begin{aligned}\mathbf{x}_{k|k-1} &= \mathbf{F} \mathbf{x}_{k-1|k-1}, & \mathbf{P}_{k|k-1} &= \mathbf{F} \mathbf{P}_{k-1|k-1} \mathbf{F}^\top + \mathbf{Q}, \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1} \mathbf{H}^\top (\mathbf{H} \mathbf{P}_{k|k-1} \mathbf{H}^\top + \mathbf{R})^{-1}, & \mathbf{x}_{k|k} &= \mathbf{x}_{k|k-1} + \mathbf{K}_k (\mathbf{z}_k - \mathbf{H} \mathbf{x}_{k|k-1}), \\ \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}) \mathbf{P}_{k|k-1}.\end{aligned}$$

Ruido de proceso, de medición e inicialización. El ruido de proceso sigue el modelo de **aceleración blanca** (Bar-Shalom et al., 2001), que para cada eje acopla posición y velocidad mediante

$$\mathbf{Q}_{\text{eje}}(\Delta t) = q \begin{bmatrix} \Delta t^3/3 & \Delta t^2/2 \\ \Delta t^2/2 & \Delta t \end{bmatrix},$$

donde q es la densidad espectral del ruido de aceleración. El ruido de medición es $\mathbf{R} = \sigma_z^2 \mathbf{I}_3$ y la covarianza inicial $\mathbf{P}_0 = \text{diag}(\sigma_{p,0}^2 \mathbf{I}_3, \sigma_{v,0}^2 \mathbf{I}_3)$. El estado se inicializa con la **primera medición** para la posición ($\mathbf{x}_0[1:3] = \mathbf{z}_0$) y velocidad nula, evitando el transitorio de arrancar en el origen; el primer *frame* solo actualiza la posición. Los valores empleados son:

Parámetro	Símbolo	Valor
Ruido de proceso	q	$5 \times 10^5 \text{ mm}^2/\text{s}^3$
Ruido de medición	σ_z^2	25 mm^2 ($\sigma_z = 5 \text{ mm}$)
Var. inicial de posición	$\sigma_{p,0}^2$	100 mm^2 ($\sigma_{p,0} = 10 \text{ mm}$)
Var. inicial de velocidad	$\sigma_{v,0}^2$	$9 \times 10^6 (\text{mm/s})^2$ ($\sigma_{v,0} \approx 3 \text{ m/s}$)

Justificación de los valores.

- $\sigma_z = 5 \text{ mm}$: *jitter* milimétrico del joint HAND_RIGHT del *Body Tracking* del Kinect; se eleva para YOLO11-pose y MediaPipe, cuya reproyección 3D acumula el error del *keypoint* 2D más el error de profundidad (Sección 4.17).
- $\sigma_{p,0} = 10 \text{ mm}$: cubre el desconocimiento de la posición inicial real de la mano; tiene poco impacto porque el filtro converge en pocos *frames*.
- $\sigma_{v,0} \approx 3 \text{ m/s}$: dimensionada al rango superior de velocidades de un golpe humano

(literatura: 5–9 m/s para un golpe completo, menos para un amago detenido). Una elección más pequeña anclaría el estado en $v \approx 0$ y sesgaría el pico a la baja.

- $q = 5 \times 10^5 \text{ mm}^2/\text{s}^3$: densidad espectral del ruido de aceleración fijada a la escala de la aceleración pico del golpe ($\sigma_a \sim 50\text{--}200 \text{ m/s}^2$; regla de tuning de Bar-Shalom et al. (2001): $q \approx \sigma_a^2 \cdot \Delta t$). Permite que el filtro siga la maniobra sin amplificar el ruido de medición.

Posibles refinamientos. La sintonía es uniforme por método salvo σ_z^2 ; mejoras factibles son: (i) ajustar el ruido de medición por articulación (la mano presenta más *jitter* que el hombro); (ii) adoptar un modelo *Interacting Multiple Model* (IMM) (X. R. Li & Jilkov, 2003) que combine velocidad y aceleración constante, mejorando la transición reposo–maniobra; (iii) aplicar un suavizador no causal en post-procesamiento *offline* para reducir aún más la varianza preservando el pico de velocidad.

Filtro lineal. Dado que tanto la transición como la observación son **lineales**, el estimador óptimo es el filtro de Kalman lineal; no se requiere una variante extendida (EKF), que solo se justificaría con un modelo de movimiento u observación no lineales (por ejemplo, medir distancia o ángulo a un punto).

Velocidad tomada del estado. La velocidad del golpe se toma directamente del subvector $[v_x, v_y, v_z]$ del estado, que es la cantidad que el filtro estima de forma óptima. El filtro además tolera huecos de detección, prediciendo el estado cuando un *frame* no aporta medición.

La Figura 4.14 ilustra el efecto del filtro sobre una trayectoria real: la *trayectoria original* (puntos rojos) presenta saltos bruscos propios del *jitter* del *Body Tracking*, mientras que la *trayectoria suavizada* (triángulos azules) ofrece una estimación continua de la que se deriva directamente la velocidad instantánea. Los puntos grises en torno a la silueta corresponden a la nube de puntos del rostro segmentada mediante los umbrales espaciales descritos en la Sección 4.9.

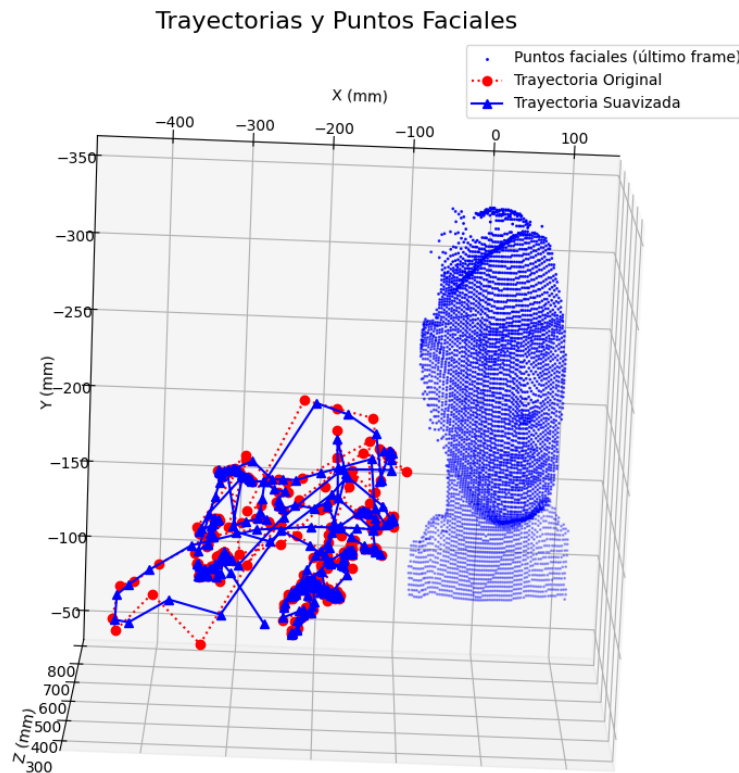


Figura 4.14: Visualización 3D de la nube de puntos facial y de la trayectoria de la mano: trayectoria original (puntos rojos, ruidosa) vs. trayectoria suavizada por el filtro de Kalman (triángulos azules). La diferencia entre ambas justifica cualitativamente la necesidad del filtro para obtener una estimación de velocidad fiable.

4.13. Homogeneización 3D de los métodos RGB: reproyección inversa

Un aspecto clave del diseño es que **YOLO11-pose** y **MediaPipe** son **detectores puramente 2D**: operan sobre el *stream* RGB y entregan coordenadas de *keypoints* en píxeles (u, v) . Por sí solas, estas coordenadas no son comparables con las posiciones en milímetros que proporciona el Azure Kinect Body Tracking (que opera directamente sobre el sensor de profundidad).

4.13.1. Dos cámaras físicamente distintas en el mismo cuerpo

El Azure Kinect DK integra dos sensores independientes con ópticas y resoluciones distintas: la **cámara de color** (RGB, hasta 3840×2160 px) y el **sensor de profundidad** (ToF, 640×576 px en NFOV Unbinned). Cada uno tiene su propia **matriz intrínseca** K_{rgb} y K_{depth} , y una **transformación extrínseca** $T_{\text{depth} \rightarrow \text{rgb}}$ (rotación R + traslación t) que codifica su desplazamiento físico dentro del cuerpo de la cámara. El SDK puede alinear automáticamente el mapa de profundidad al frame de color —transformación extrínseca incluida— dejando al usuario un mapa de profundidad en la *misma cuadrícula de píxeles* que el frame RGB, que es el modo utilizado en este trabajo.

Un **pixel RGB** (u_c, v_c) y el **píxel de profundidad** (u_d, v_d) **situado en la misma posición del mapa alineado miden el mismo punto del espacio 3D**, pero cada cámara ve ese punto desde ángulos y con distorsiones diferentes. La Figura 4.15 muestra una captura real del corpus MKV (boca, NFOV Unbinned, luz artificial): a la izquierda el frame RGB con textura y color, a la derecha el mapa de profundidad transformado al sistema de la cámara de color mediante la API de alineación del SDK — donde la escala cromática codifica la distancia z (tonos cálidos = lejano, tonos fríos = cercano). La cruz blanca señala el *mismo* píxel en ambas imágenes, demostrando la correspondencia entre los dos sistemas.

4.13.2. Paso a paso de la back-projection

Para obtener una comparación justa, se aplica la transformación inversa de proyección (*back-projection*) en tres pasos. La Figura 4.16 ilustra la geometría del proceso.

1. **Localización del keypoint 2D.** El detector (YOLO o MediaPipe) identifica las coordenadas de píxel (u, v) del punto de interés (muñeca o palma) en el frame RGB.
2. **Lectura de la profundidad alineada.** Se consulta el mapa de profundidad ya alineado al RGB bajo ese píxel, obteniendo z en milímetros. Si el valor es inválido



Mismo punto del espacio 3D en las dos imágenes del Azure Kinect

Figura 4.15: Captura real del corpus MKV (boca, NFOV Unbinned, luz artificial). Izquierda: frame RGB (1280×720 px). Derecha: mapa de profundidad ToF alineado al RGB mediante el SDK (640×576 px interpolados al tamaño del RGB). La escala de colores va de azul (zona cercana, ~ 450 mm) a rojo (zona lejana, ~ 1200 mm). La cruz marca el *mismo* punto 3D en ambas vistas, ilustrando la correspondencia que hace posible la back-projection.

(píxel sin retorno ToF), se promedia la vecindad 3×3 excluyendo ceros.

3. **Reconstrucción 3D con la matriz intrínseca.** Con los parámetros intrínsecos de la cámara de profundidad (f_x , f_y , c_x , c_y), la coordenada 3D en el sistema de la cámara es:

$$X = \frac{(u - c_x)z}{f_x}, \quad Y = \frac{(v - c_y)z}{f_y}, \quad Z = z.$$

Este es el modelo *pinhole* inverso: despejando X e Y de la proyección perspectiva $u = f_x X/Z + c_x$, $v = f_y Y/Z + c_y$.

De este modo, los tres métodos entregan coordenadas en el mismo sistema de referencia y en las mismas unidades (mm), lo que garantiza que la distancia mano–rostro y la velocidad de aproximación sean magnitudes homogéneas y directamente comparables.

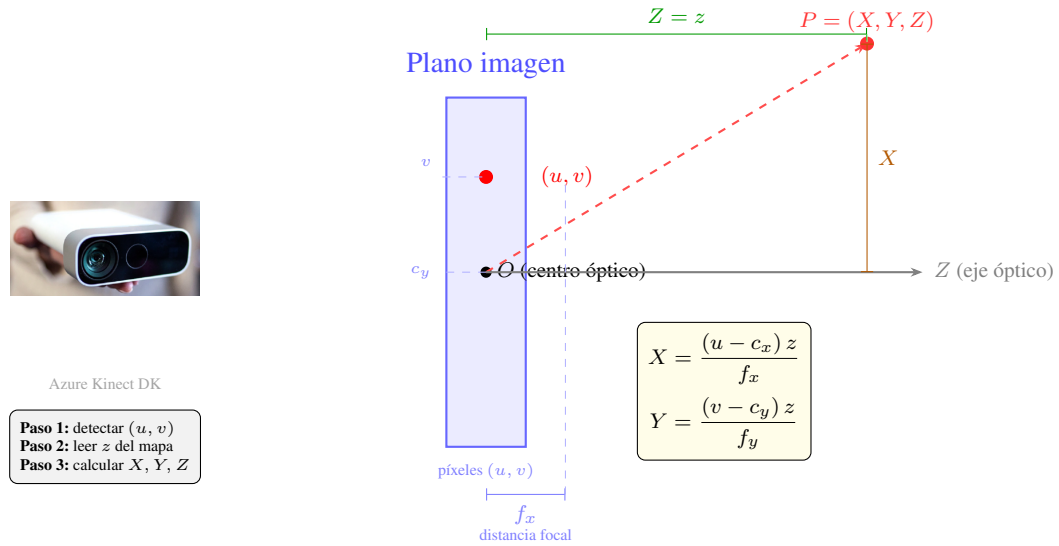


Figura 4.16: Geometría del modelo *pinhole* para la back-projection. El punto de imagen (u, v) define una dirección en el espacio; la profundidad z fija la distancia a lo largo del eje óptico, determinando unívocamente el punto 3D $P = (X, Y, Z)$. Los parámetros intrínsecos (f_x, f_y, c_x, c_y) calibran la geometría de la cámara específica.

Limitación inherente. El error de la *back-projection* depende del ruido en z . Para el Kinect, $\sigma_z \lesssim 17$ mm (Bertram et al., 2023); para MediaPipe, cuya malla densa cubre regiones con reflexión variable, el error de reproyección puede alcanzar decenas de milímetros. Esta ambigüedad es una limitación fundamental de cualquier estimador 2D o monocular: sin una medición métrica de profundidad, la coordenada z no puede reconstruirse sin sesgo (S. Zhang et al., 2022), y la estimación resultante de velocidad y aceleración hereda ese error de forma magnificada al calcular la derivada temporal (Stamm & Heimann-Steinert, 2020). Esto explica que MediaPipe produzca velocidades estimadas más ruidosas que el Kinect nativo; un estudio reciente de validación de MediaPipe para extremidad superior confirma que su desempeño solo ha sido evaluado rigurosamente en 2D y recomienda validación 3D antes de su uso clínico (Wagh et al., 2024).

Pipeline de back-projection 2D → 3D — $X = \frac{(u - c_x)z}{f_x}$, $Y = \frac{(v - c_y)z}{f_y}$, $Z = z$

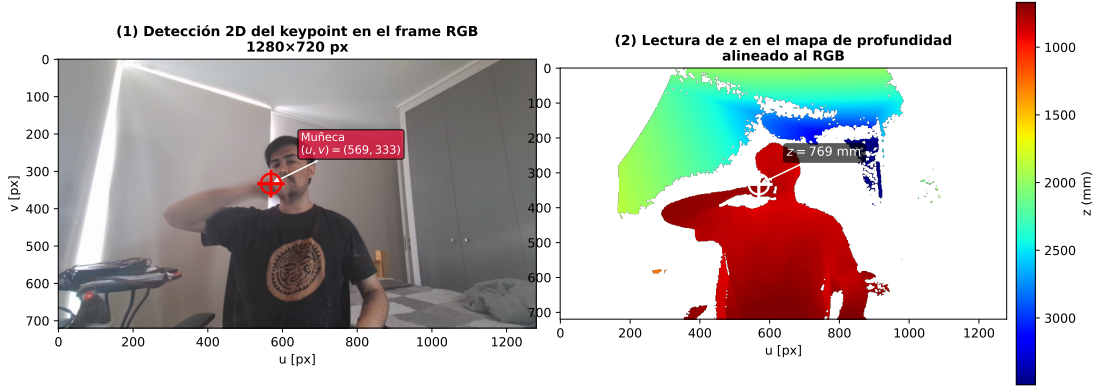


Figura 4.17: Aplicación del pipeline sobre un frame real del corpus (boca, NFOV Unbinned, luz artificial), en el instante de máxima aproximación mano–rostro. **(1)** Frame RGB (1280 × 720 px): el joint `WRIST_RIGHT` entregado por el Body Tracking SDK del Azure Kinect se proyecta sobre la imagen mediante la transformación extrínseca calibrada, produciendo el píxel (u, v) marcado con la cruceta roja. **(2)** Mapa de profundidad ToF alineado al RGB: la profundidad z bajo ese mismo píxel es la magnitud que, junto con los parámetros intrínsecos, permite reconstruir la coordenada 3D métrica de la muñeca.

4.14. Proyección 3D → 2D para anotación

Para anotar los eventos sobre el flujo RGB, se aplica la proyección directa. Dado un punto $P_w = [X_w \ Y_w \ Z_w \ 1]^T$ en coordenadas homogéneas, primero se transforma al sistema de la cámara mediante la matriz extrínseca $\mathbf{T}$ (rotación $\mathbf{R}$ y traslación $\mathbf{t}$),

$$P_c = \mathbf{T} P_w, \quad \mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix},$$

y luego se proyecta con la matriz intrínseca $\mathbf{K}$:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad u = f_x \frac{X_c}{Z_c} + c_x, \quad v = f_y \frac{Y_c}{Z_c} + c_y.$$

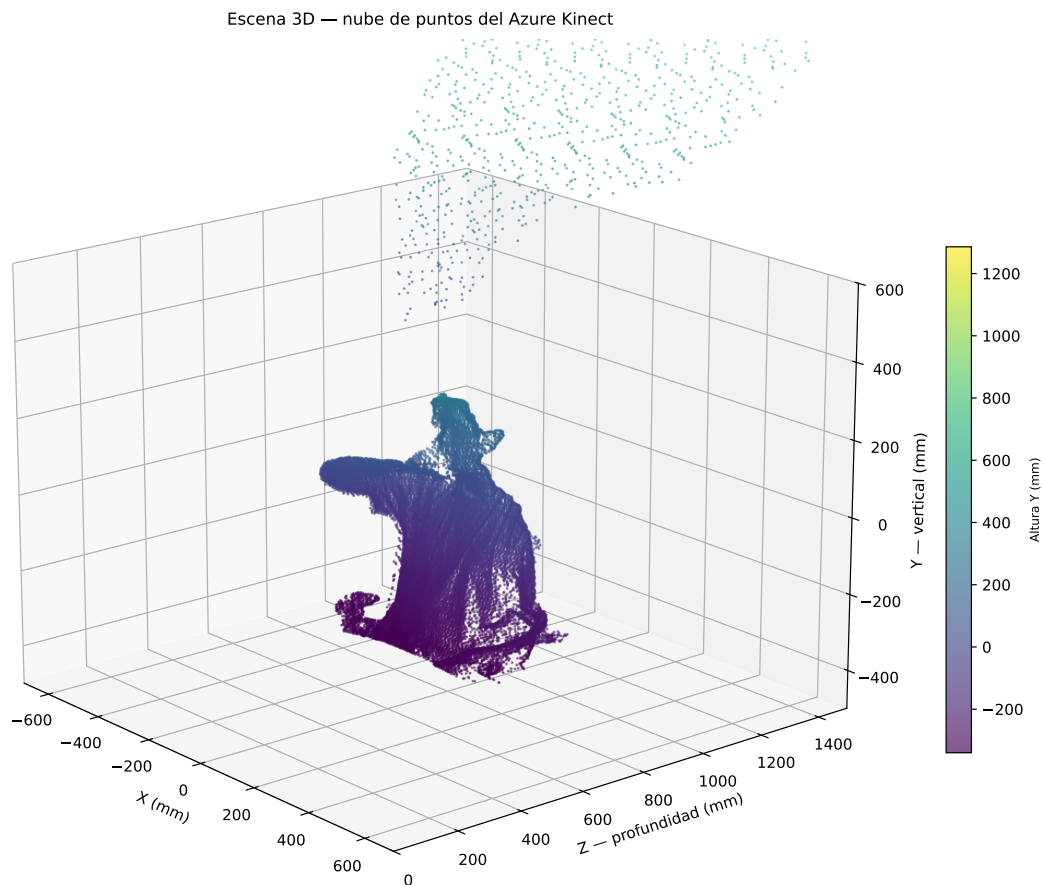


Figura 4.18: Nube de puntos 3D reconstruida directamente del sensor ToF del Azure Kinect DK (modo NFOV Unbinned, 640×576 píxeles de profundidad). Cada punto corresponde a un píxel del mapa de profundidad y sus coordenadas (X, Y, Z) se expresan en milímetros en el sistema de referencia de la cámara de profundidad. La escala de color codifica la distancia Z al sensor (tonos fríos = cercano, cálidos = lejano). La vista diagonal desde arriba permite apreciar la geometría de la habitación (paredes, suelo) y la silueta del sujeto sentado.

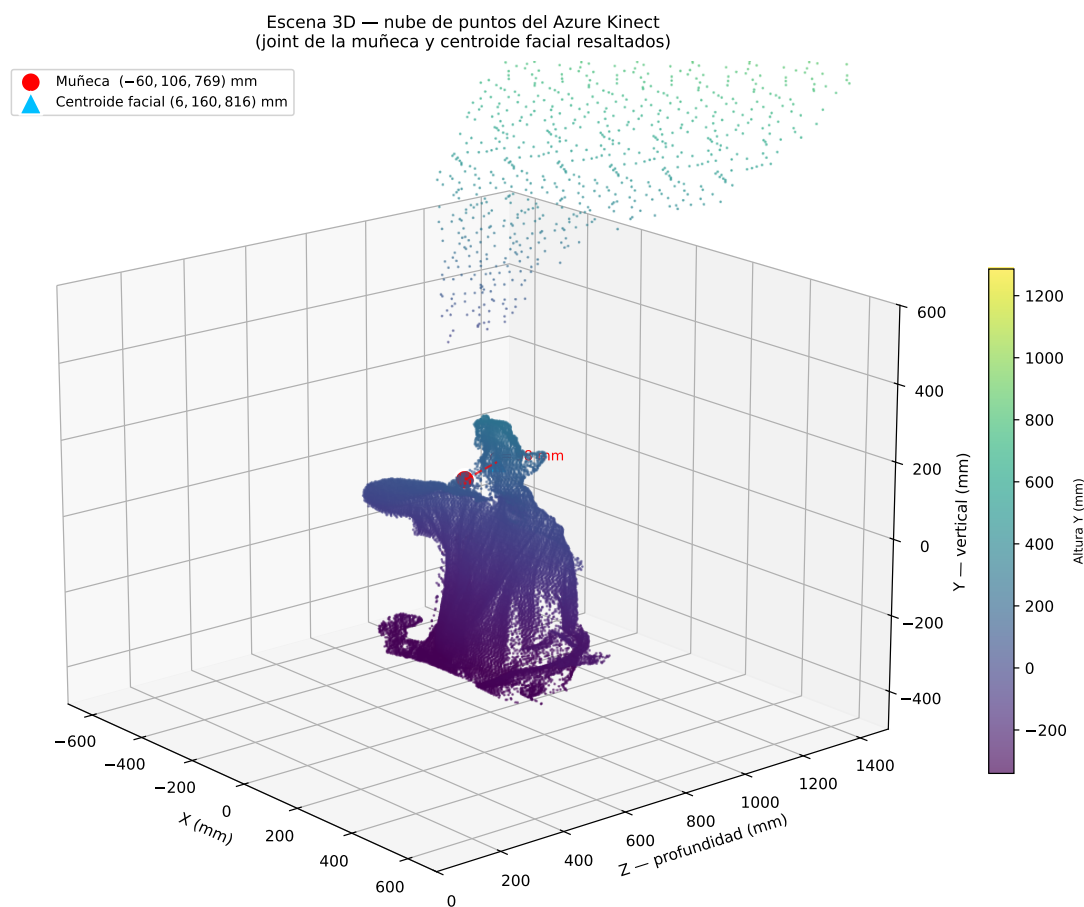


Figura 4.19: Misma nube de puntos que la Figura 4.18, con el joint `WRIST_RIGHT` entregado por el Body Tracking SDK resaltado en rojo y el centroide de la región facial en azul. Ambas posiciones comparten el mismo sistema de referencia 3D que la nube (mm, frame de la cámara de profundidad), por lo que la distancia euclidiana entre ellos es directamente la magnitud medida por el sistema de detección de impactos.

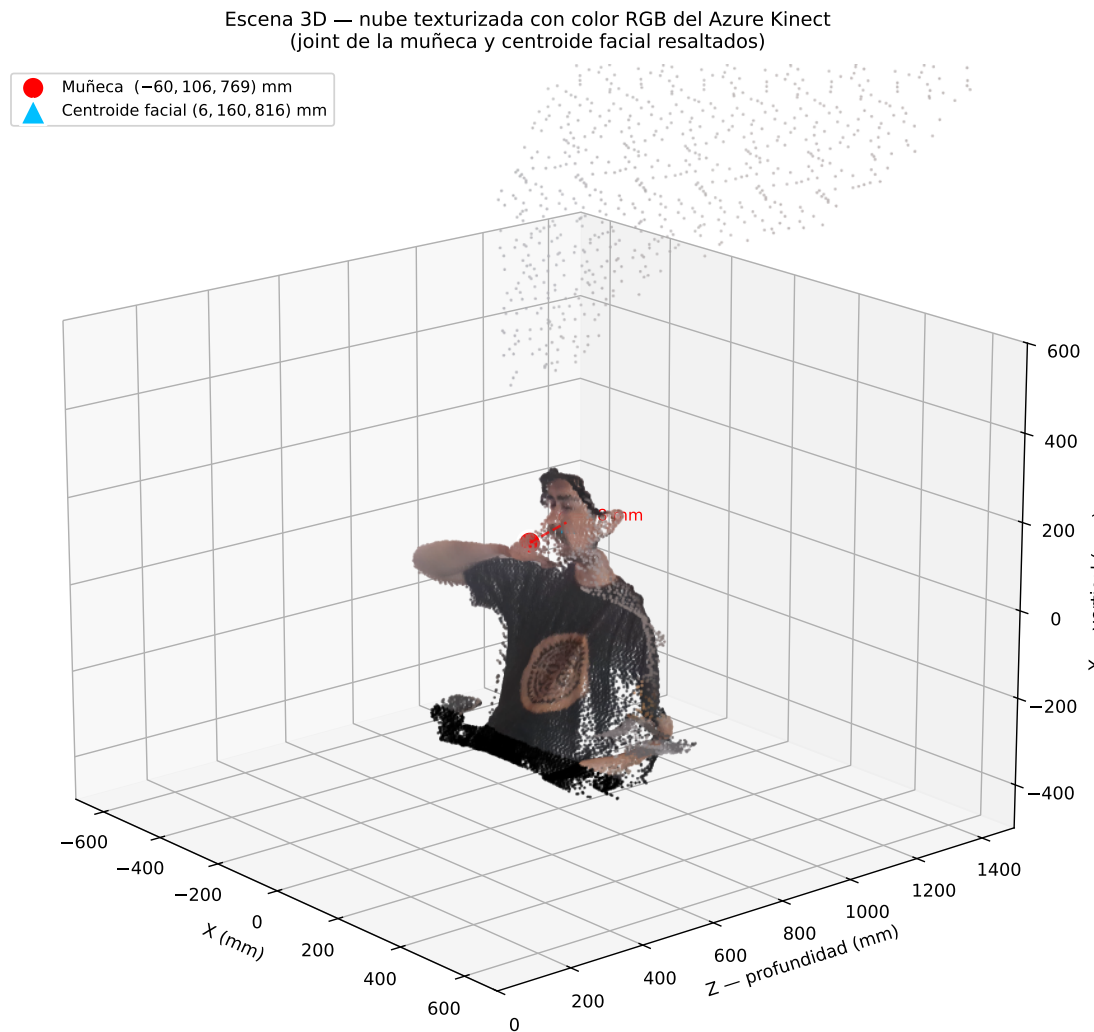


Figura 4.20: Réplica exacta de la Figura 4.19 en la que cada punto 3D recibe el color del píxel RGB correspondiente, obtenido mediante `get_transformed_color_image()` del SDK de Azure Kinect. Esta transformación alinea la imagen de color al frame de la cámara de profundidad píxel a píxel, permitiendo reconstruir la escena con la textura real y haciendo inmediatamente identificables la ropa del sujeto, el fondo y la posición tridimensional de la muñeca (marcada en rojo).

Pipeline de back-projection 2D → 3D (datos reales Azure Kinect, frame de impacto)

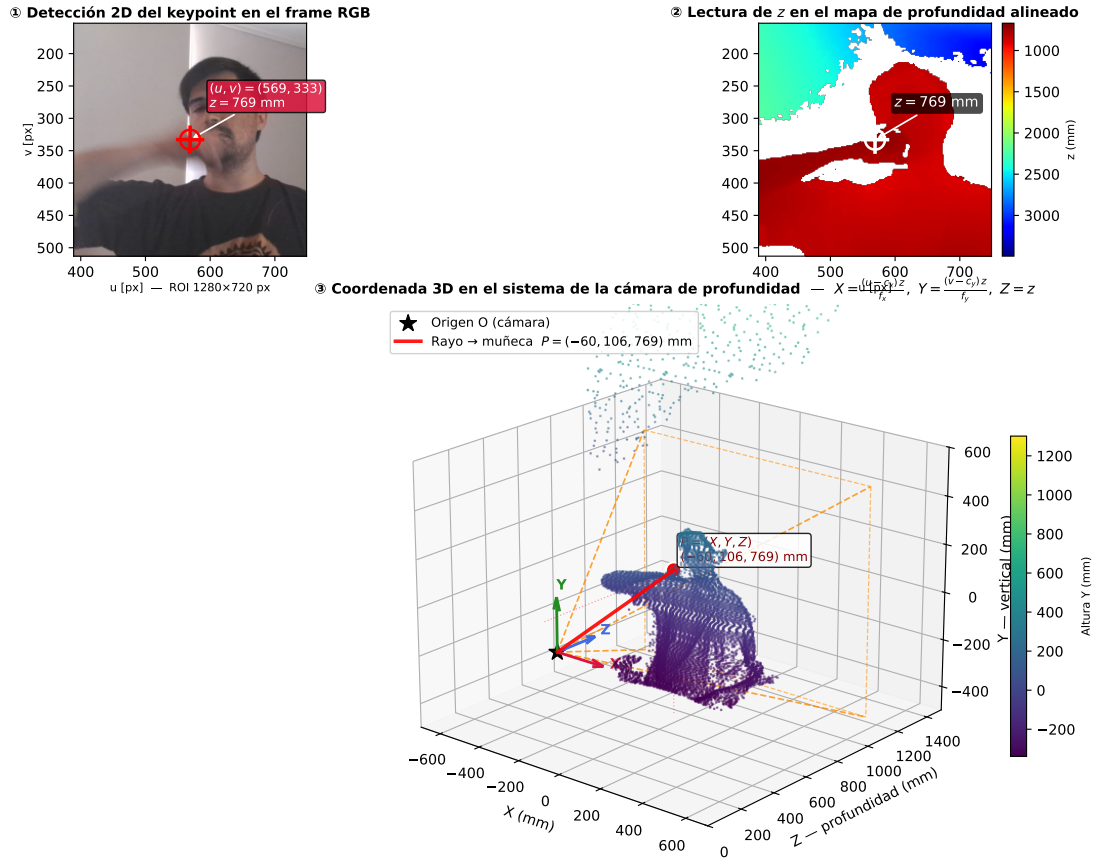


Figura 4.21: Visualización integrada del pipeline de back-projection 2D→3D. Los paneles (1) y (2) muestran respectivamente el recorte RGB y el mapa de profundidad centrados en el keypoint de la muñeca; el panel (3) presenta la nube 3D coloreada por altura (colormap *viridis*). Las líneas naranjas delimitan el frustum de la cámara de profundidad; los ejes X (rojo), Z (azul) e Y (verde) marcan la orientación del sistema de referencia con origen en O . El rayo rojo parte de O , atraviesa el píxel (u, v) de la muñeca, y alcanza la posición 3D $P = (X, Y, Z)$ en la nube. El valor de Z proviene directamente del mapa ToF bajo ese píxel, convirtiendo la dirección 2D del rayo en una ubicación métrica unívoca. Este proceso, descrito analíticamente por $X = \frac{(u-c_x)z}{f_x}$, $Y = \frac{(v-c_y)z}{f_y}$, $Z = z$, es el que homogeneiza los métodos RGB (YOLO, MediaPipe) con el Body Tracking nativo del Kinect.

Pipeline de back-projection 2D → 3D — nube texturizada con color RGB del Azure Kinect

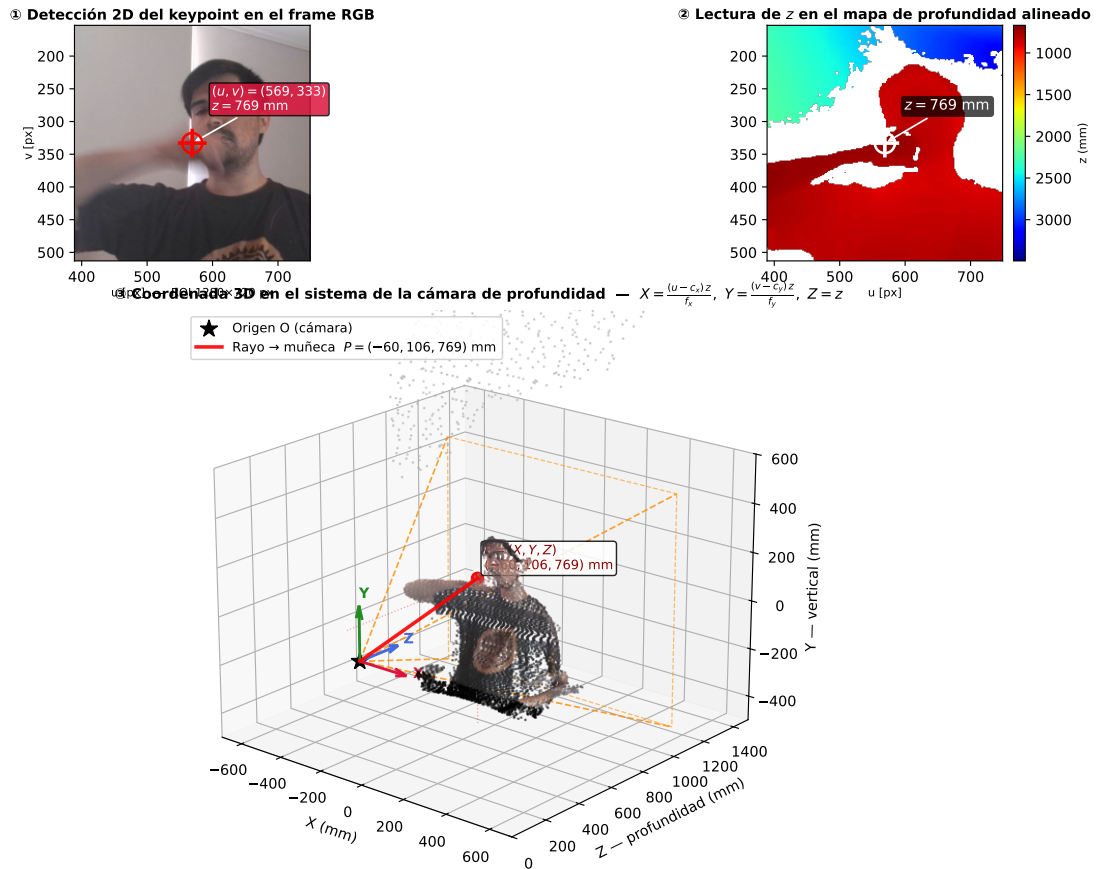


Figura 4.22: Réplica del diagrama de la Figura 4.21 en la que la nube de puntos del panel (3) lleva la textura de color RGB real del sensor (misma transformación `get_transformed_color_image()`). La correspondencia entre el píxel (u, v) en la imagen a color y el punto $P = (X, Y, Z)$ en la nube es directamente visible: el rayo rojo parte del origen óptico O , cruza el plano imagen en el keypoint de la muñeca y termina en la región de color real de la mano. Las flechas naranjas delimitan el frustum de la cámara de profundidad; los ejes X (rojo), Z (azul) e Y (verde) muestran la orientación del sistema de referencia.

4.15. Estimación de velocidad, aproximación y aceleración

La estimación cinemática se apoya en dos supuestos de cálculo:

1. **Tiempo físico real.** El tiempo de cada *frame* se toma del *device timestamp* del sensor (microsegundos), no del reloj de pared, que no guarda relación con el tiempo del video durante la reproducción.
2. **Velocidad instantánea, no promedio del *swing*.** La velocidad se estima como la velocidad **instantánea** —por mínimos cuadrados sobre las últimas w muestras, equivalente a un diferenciador de Savitzky–Golay de primer orden (Savitzky & Golay, 1964), o del estado del filtro de Kalman—, no como el promedio entre el primer y el último punto de la trayectoria, que subestimaría el valor en el impacto.

Se distinguen tres magnitudes cinemáticas, todas **vectoriales** y con Δt físico:

- **Velocidad de la mano** $\mathbf{v} = [v_x, v_y, v_z]$ (m/s), con rapidez $\|\mathbf{v}\|$.
- **Velocidad de aproximación** (*closing speed*): tasa de cierre de la distancia mínima mano–rostro,

$$v_{\text{aprox}} = -\frac{d}{dt} d_{\text{mano-rostro}},$$

estimada sobre una ventana corta. Esta definición es robusta al salto del punto facial más cercano entre *frames* (problema con mallas densas como Face Mesh), a diferencia de proyectar la velocidad sobre la dirección al vecino más cercano.

- **Aceleración** $\mathbf{a} = d\mathbf{v}/dt$ (m/s²), cuya magnitud en el evento cuantifica el **frenado** característico del golpe.

Criterio de confirmación del golpe. Un evento se confirma cuando se cumplen simultáneamente: (i) proximidad, $d_{\text{mano-rostro}} < U_{\text{prox}}$; (ii) pico de aproximación reciente $\geq v_{\text{aprox}}^{\text{mín}}$ (hubo una aproximación rápida); y (iii) deceleración característica, es decir, la

aproximación cae por debajo de una fracción del pico o se invierte. Se respeta además un **periodo refractario** entre golpes para no contar el mismo evento varias veces. La velocidad reportada del golpe es el **pico** de rapidez durante la aproximación, físicamente la magnitud relevante, y no el promedio del *swing*.

4.16. Justificación física: de la deceleración a la fuerza de impacto

Aunque el prototipo actual mide *cinemática* (velocidad y deceleración), su finalidad última es estimar la **fuerza de impacto**. La conexión es directa a través de la segunda ley de Newton. Si en el instante de máxima aproximación la mano experimenta una deceleración de magnitud $a = \|\mathbf{a}\|$ y se conoce la masa efectiva del segmento mano–antebrazo m (provista por el especialista a partir de tablas antropométricas (de Leva, 1996)), la fuerza media ejercida es

$$F = m a.$$

De forma equivalente, mediante el *teorema del impulso–cantidad de movimiento*, si la velocidad de aproximación v_{aprox} se anula en un intervalo de frenado Δt , la fuerza media es

$$\bar{F} = \frac{m \Delta v}{\Delta t} = \frac{m v_{\text{aprox}}}{\Delta t}.$$

Esta segunda formulación es especialmente conveniente en nuestro escenario, pues tanto v_{aprox} como Δt son observables directos del perfil de aproximación, sin necesidad de derivar dos veces la posición (operación que amplifica el ruido). El supuesto subyacente es que la mano se modela como una masa puntual efectiva y que el frenado ocurre en un intervalo corto y aproximadamente uniforme; ambos son razonables para un primer modelo y refinables a futuro incorporando la deformación de los tejidos (un modelo masa–resorte–amortiguador) y la masa efectiva real del segmento en el momento del

contacto. Este encuadre es coherente con la biomecánica del golpeo, donde la velocidad de la mano y su deceleración son las magnitudes centrales para caracterizar el impacto (Liu et al., 2022). La medición fiable de la deceleración —y de la velocidad de aproximación— que aquí se valida es, por tanto, el prerequisite cinemático para la futura estimación de fuerza. Cabe señalar que este análisis cinemático requiere **coordenadas 3D métricas**: la ambigüedad de profundidad inherente a la visión monocular o a la estimación 2D de pose hace que la componente z no sea recuperable con precisión métrica sin una medición directa de distancia (Stamm & Heimann-Steinert, 2020; S. Zhang et al., 2022), y los errores resultantes se amplifican al calcular la velocidad y la aceleración como derivadas temporales de la posición. El sensor ToF del Azure Kinect, cuya exactitud y repetibilidad ha sido validada clínicamente con $\sigma_z \lesssim 17$ mm (Bertram et al., 2023), provee esa magnitud métrica directamente, lo que lo convierte en la referencia apropiada para este tipo de medición.

4.17. Análisis de incertidumbre

La incertidumbre de la posición 3D de la mano combina varias fuentes; se desglosan a continuación para construir explícitamente el margen reportado en los resultados.

Profundidad (z). Caracterizada por Tölgyessy et al. (2021) para el Azure Kinect, la contribución acumulada es:

1. **Error aleatorio del sensor de profundidad** ($\sigma_z^{\text{rand}} \leq 17$ mm): *jitter frame-a-frame* del sensor ToF, dependiente de la reflectividad de la superficie y de la iluminación infrarroja.
2. **Error sistemático** (≤ 11 mm a distancias típicas): *bias* dependiente del modo de profundidad y de la temperatura del sensor.
3. **Error proporcional a la distancia** (0,1 % de z): deriva intrínseca de la calibración de profundidad.

Para la distancia de captura nominal $z = 0,5$ m, el peor caso acotado es:

$$z' \pm \underbrace{0,017}_{\text{aleatorio}} + \underbrace{0,011}_{\text{sistemático}} + \underbrace{0,001 \cdot 0,5}_{0,1 \% z} \text{ [m]} \Rightarrow z' \pm 0,0285 \text{ [m]}. \quad (4.1)$$

Incertidumbre por método — Azure Kinect Body Tracking. El joint `HAND_RIGHT` se obtiene directamente de la red de segmentación sobre el mapa de profundidad, sin pasar por la back-projection. Su error de posición 3D está dominado por el ruido ToF del sensor:

- $\sigma_z^{\text{rand}} \leq 17$ mm (jitter frame-a-frame).
- Error sistemático ≤ 11 mm a distancias nominales.
- Error proporcional: $0,1 \% \times 500 \text{ mm} = 0,5$ mm (despreciable).

Incertidumbre de posición combinada a $z = 0,5$ m: $\sigma_{\text{pos}}^{(K)} \approx 5$ mm (dominado por el jitter del joint, no por el error máximo del sensor). La velocidad resultante tiene incertidumbre $\sigma_v^{(K)} \approx 0,11$ m/s y la deceleración $\sigma_a^{(K)} \approx 6$ m/s² (ver propagación abajo). Un golpe típico medido con el Kinect: $v_{\text{imp}} = 3,80 \pm 0,11$ m/s, $a_{\text{imp}} = 30 \pm 6$ m/s².

Incertidumbre por método — YOLO11-pose. La muñeca (keypoint 10) se localiza con un error de 2–5 px en condiciones normales (Jocher et al., 2024). A $z = 0,5$ m con $f_x \approx 612$ px, la propagación del error de píxel a distancia lateral es:

$$\Delta X = \frac{\Delta u \cdot z}{f_x} = \frac{5 \text{ px} \times 500 \text{ mm}}{612 \text{ px}} \approx 4 \text{ mm}.$$

Sumada cuadráticamente al error de profundidad ($\sigma_z \approx 17$ mm), la incertidumbre de posición 3D es $\sigma_{\text{pos}}^{(Y)} \approx 18$ mm. Sin embargo, el error **sistemático** dominante de YOLO es que mide la *muñeca*, no la mano: en el evento de máxima aproximación, la muñeca queda sistemáticamente 130–200 mm más alejada del rostro que la palma (observable directo en la columna $\bar{d}$ de la Tabla 5.1: 136 mm frente a 25 mm del Kinect). El ruido de medición del filtro de Kalman se eleva a $\sigma_z^{(Y)} = 17$ mm para reflejar esta mayor

incertidumbre. Velocidad típica: $v_{\text{imp}} = 3,45 \pm 0,40$ m/s (la σ inter-impacto de 10 m/s incluye huecos de detección y saltos por reoclusión).

Incertidumbre por método — MediaPipe. MediaPipe Hands (F. Zhang et al., 2020) estima el centro de la palma como promedio de muñeca y nudillos (21 landmarks), reportando además un componente z relativo. El error de localización del landmark 2D es de 2–4 px en palma, equivalente a $\Delta X \approx 3$ mm, comparable al Kinect. Sin embargo, la back-projection introduce una fuente adicional: la profundidad z bajo el landmark de la palma puede ser ruidosa cuando la mano está orientada de canto o parcialmente ocluyendo el rostro, situación frecuente durante el amago de golpe. El error de profundidad por este efecto puede alcanzar **decenas de milímetros**, elevando $\sigma_{\text{pos}}^{(MP)} \approx 30\text{--}50$ mm. Adicionalmente, el centro de la palma presenta un *offset* físico de ~ 2 cm respecto al punto de contacto (punta de dedos), dominante sobre el error de detección puro. Por ello, la incertidumbre del filtro de Kalman se eleva a $\sigma_z^{(MP)} = 30$ mm. Velocidad típica: $v_{\text{imp}} = 1,28 \pm 0,11$ m/s (con Kalman); sin filtro, $2,36 \pm 0,26$ m/s, reflejando la amplificación del ruido de posición por diferenciación.

Comparativa de incertidumbre. La Tabla 4.1 condensa los valores de incertidumbre por método.

Método	σ_{pos}	σ_v	σ_a	Sesgo sistemático
Azure Kinect	~ 5 mm	0.11 m/s	6 m/s ²	Ninguno (joint HAND_RIGHT)
YOLO11-pose	~ 18 mm	~ 0.40 m/s	~ 23 m/s ²	+130 mm (muñeca $\neq$ palma)
MediaPipe	$\sim 30\text{--}50$ mm	~ 0.70 m/s	~ 40 m/s ²	$\sim +20$ mm (palma $\neq$ dedos)

Cuadro 4.1: Incertidumbre de posición, velocidad y aceleración por método (1σ). El sesgo sistemático se refiere al punto de referencia medido vs. el punto de contacto ideal.

Propagación a velocidad y aceleración. La velocidad instantánea por mínimos cuadrados sobre w muestras a 30 fps tiene varianza $\sigma_v^2 = \sigma_{\text{pos}}^2 \frac{12}{w^3 - w} \frac{1}{\Delta t^2}$ (Savitzky & Golay, 1964); para $w = 3$ y $\sigma_{\text{pos}} \approx 5$ mm, $\sigma_v \approx 0,11$ m/s. Por una segunda derivación se

obtiene $\sigma_a \approx 6 \text{ m/s}^2$.

Protocolo de reporte. Las magnitudes individuales se reportan como *valor central* $\pm$ *incertidumbre combinada* (1σ). En las tablas agregadas, la columna σ_v es la desviación estándar **entre eventos** (variabilidad inter-impacto, dominante sobre la incertidumbre intra-impacto del sensor a partir de $N \geq 10$ golpes) y el rango se representa por el percentil 95 (v_{p95}), robusto a *outliers*.

Reflectividad infrarroja. Un fenómeno físico adicional es la **baja reflectividad del color negro** al infrarrojo. Los sensores de tiempo de vuelo emiten un patrón infrarrojo y miden su retorno; los objetos negros (p. ej. cabello oscuro) reflejan muy poca luz, debilitando o anulando la señal de profundidad en esas zonas. Esto degrada especialmente la captura de la nube de puntos en golpes dirigidos a la parte posterior de la cabeza.

4.18. Algoritmo de detección

El procedimiento completo se resume en el Algoritmo 1.

Algorithm 1 Impact detection and measurement by maximum approximation

Require: Recording .mkv, calibration (intrinsic/extrinsic), tracking method $\in \{\text{kinect, yolo, mediapipe}\}$, thresholds $U_{\text{prox}}, v_{\text{aprox}}^{\text{mín}}$, deceleration ratio ρ , refractory period τ

Ensure: List of events with speed, deceleration, distance, and annotated capture

```
1: Initialise Kalman filter  $KF$ , history buffers  $\{t, \mathbf{x}, \mathbf{v}, d\}$ 
2:  $t_{\text{last}} \leftarrow -\infty$ ;  $id \leftarrow 1$ 
3: while frame available in recording do
4:    $t \leftarrow$  sensor device timestamp (s) ▷ physical time, not wall clock
5:    $(\mathbf{x}_{\text{hand}}, \mathcal{F}) \leftarrow \text{DETECT}_{\text{method}}(\text{frame})$  ▷ hand and facial points in mm
6:   if no hand detected then
7:     continue
8:    $\Delta t \leftarrow t - t_{\text{prev}}$ 
9:   if  $KF$  enabled then
10:     $KF.\text{PREDICT}(\Delta t)$ ;  $KF.\text{UPDATE}(\mathbf{x}_{\text{hand}})$ 
11:     $\mathbf{x} \leftarrow KF.\text{position}$ ;  $\mathbf{v} \leftarrow KF.\text{velocity}$  ▷ velocity read from filter state
12:   else
13:     $\mathbf{x} \leftarrow \mathbf{x}_{\text{hand}}$ ;  $\mathbf{v} \leftarrow \text{FINITEDIFF}(\text{positions, times})$ 
14:     $d \leftarrow \min_{p \in \mathcal{F}} \|\mathbf{x} - p\|$ ;  $p^* \leftarrow \arg \min_{p \in \mathcal{F}} \|\mathbf{x} - p\|$ 
15:     $v_{\text{aprox}} \leftarrow -\Delta d / \Delta t$  ▷ hand–face closing speed
16:    Append  $(t, \mathbf{x}, \mathbf{v}, d, v_{\text{aprox}})$  to history buffers
17:    if  $d < U_{\text{prox}}$  and  $t - t_{\text{last}} > \tau$  then
18:       $v_{\text{peak}} \leftarrow \text{máx}(\|\mathbf{v}\| \text{ recent})$ ;  $v_{\text{aprox}}^{\text{máx}} \leftarrow \text{máx}(v_{\text{aprox}} \text{ recent})$ 
19:      if  $v_{\text{aprox}}^{\text{máx}} \geq v_{\text{aprox}}^{\text{mín}}$  and  $v_{\text{aprox}} \leq \rho v_{\text{aprox}}^{\text{máx}}$  then
20:         $\mathbf{a} \leftarrow d\mathbf{v}/dt$  ▷ deceleration vector
21:         $(u, v) \leftarrow \text{PROJECT3DTO2D}(p^*, \text{calibration})$ 
22:        Annotate RGB frame with hand,  $p^*$  and  $d$ ; save capture
23:        Record event  $id$ :  $(v_{\text{peak}}, \|\mathbf{a}\|, d, p^*, \text{trajectory})$ 
24:         $t_{\text{last}} \leftarrow t$ ;  $id \leftarrow id + 1$ 
25: return event list
```

4.19. Implementación simplificada: núcleo del detector

El Algoritmo 1 se traduce en Python mediante las clases descritas en la Sección 4.5. A continuación se presenta el núcleo del bucle de detección de forma simplificada, omitiendo manejo de errores y escritura de CSV para concentrarse en la lógica esencial:

```
1 # -- inicializacion --
2 kf = KalmanFilter3D(q=5e5, sigma_z=5.0, sigma_v0=3000.0)
3 history = deque(maxlen=30) # buffer de 1 s a 30 fps
4 t_last, refrac = -np.inf, 0.5 # periodo refractario
```

```

5
6 for frame in video.frames():
7     t = frame.device_timestamp_usec * 1e-6  # tiempo fisico del
        sensor (s)
8     hand = tracker.get_hand_3d(frame)        # (X, Y, Z) en mm
9     face = tracker.get_face_points(frame)    # nube de puntos
        facial
10
11     if hand is None:
12         continue
13
14     dt = t - (history[-1]["t"] if history else t)
15     kf.predict(dt)
16     kf.update(hand)
17
18     pos = kf.position                        # posicion suavizada (mm)
19     vel = kf.velocity                        # velocidad del estado Kalman (
        mm/s)
20     speed = np.linalg.norm(vel) / 1000.0    # rapidez en m/s
21
22     d = np.min(np.linalg.norm(face - pos, axis=1))  # dist.
        mano-rostro
23     v_close = -(d - history[-1]["d"]) / dt if history else 0.0
24
25     history.append({"t": t, "pos": pos, "vel": vel, "d": d,
26                    "speed": speed, "v_close": v_close})
27
28     if d < U_PROX and (t - t_last) > refrac:
29         v_peak = max(h["speed"] for h in history)
30         v_aprox_max = max(h["v_close"] for h in history)

```

```

31     if v_aprox_max >= V_MIN and v_close <= 0.3 *
        v_aprox_max:
32         acc = np.linalg.norm(np.gradient(
33             [h["vel"] for h in history], axis=0)[-1]) /
            1000.0
34         record_event(t, speed, acc, d)
35         t_last = t

```

Listing 4.1: Núcleo del detector de impacto (simplificado).

Las claves de diseño que hace explícitas este extracto son: (i) el tiempo t se toma del *device timestamp* del sensor, no del reloj de pared; (ii) la velocidad se lee directamente del estado del filtro de Kalman; (iii) el criterio de confirmación verifica simultáneamente proximidad, velocidad de cierre y deceleración; y (iv) el periodo refractario evita contar el mismo golpe varias veces durante la fase de rebote.

Capítulo 5

Resultados y Análisis

Se procesaron las grabaciones con los tres algoritmos. El análisis que sigue se categoriza por algoritmo y, dentro de cada uno, por modo de profundidad y condición de iluminación. Salvo que se indique lo contrario, las estadísticas corresponden a la configuración con filtro de Kalman para el Kinect; YOLO11-pose se ejecuta con filtro y MediaPipe sin él (véase la discusión sobre el ajuste del ruido de medición más adelante).

5.1. Resumen estadístico por algoritmo

La Tabla 5.1 presenta, por algoritmo y configuración del filtro de Kalman, los estadísticos de velocidad de impacto, deceleración, distancia mano–rostro y FPS. Incluir ambas configuraciones permite observar directamente el efecto del filtro sobre cada método.

Configuración		Imp.	$\bar{v}$	σ_v	v_{max}	$\bar{a}$	$\bar{d}$	FPS
Algoritmo	Kalman		[m/s]	[m/s]	[m/s]	[m/s ²]	[mm]	
Azure Kinect	On	322	3.80	1.00	9.87	30.18	24.55	8.7
	Off	315	4.37	1.39	13.39	40.16	24.86	8.9
YOLO11-pose	On	496	3.45	10.04	204.02	23.10	136.94	20.6
	Off	297	4.29	5.54	37.16	36.21	127.76	14.7
MediaPipe	On	72	1.28	0.57	3.24	7.05	43.41	20.3
	Off	54	2.36	1.47	7.10	15.74	36.96	12.7

Cuadro 5.1: Resumen estadístico por algoritmo con y sin filtro de Kalman. $\bar{v}$, σ_v , v_{max} : media, desviación estándar y máximo de la velocidad de impacto; $\bar{a}$: deceleración media; $\bar{d}$: distancia mano–rostro media.

Tres observaciones se desprenden de inmediato. Primero, la **distancia mano–rostro** en el evento es del orden de 26 mm para el Kinect frente a ~ 170 mm para YOLO11-pose: este último mide la *muñeca*, sistemáticamente más alejada del rostro que la mano, lo que confirma la limitación inherente al modelo de pose y obliga a un umbral de proximidad mucho mayor. Segundo, las velocidades del Kinect ($\bar{v} \approx 0,36$ m/s) son moderadas y de baja dispersión, coherentes con amagos de golpe controlados; las de MediaPipe son notablemente mayores y mucho más ruidosas ($\bar{v} \approx 2$ m/s, $\sigma \approx 1,7$), reflejo del error de decenas de milímetros que introduce la reproyección 2D→3D y la malla densa (Wagh et al., 2024; S. Zhang et al., 2022). Tercero, la deceleración media sigue el mismo patrón: estable y pequeña en el Kinect, elevada y dispersa en MediaPipe.

5.2. Resultados por zona facial

La Tabla 5.2 desglosa los impactos y la velocidad media por zona facial. Las zonas de la cabeza (frente, cabeza derecha e izquierda) tienden a presentar menor densidad de detecciones porque el *Body Tracking* es más sensible a movimientos que se acercan a la nariz y mejillas; además, la reflectividad infrarroja del cabello reduce la calidad de

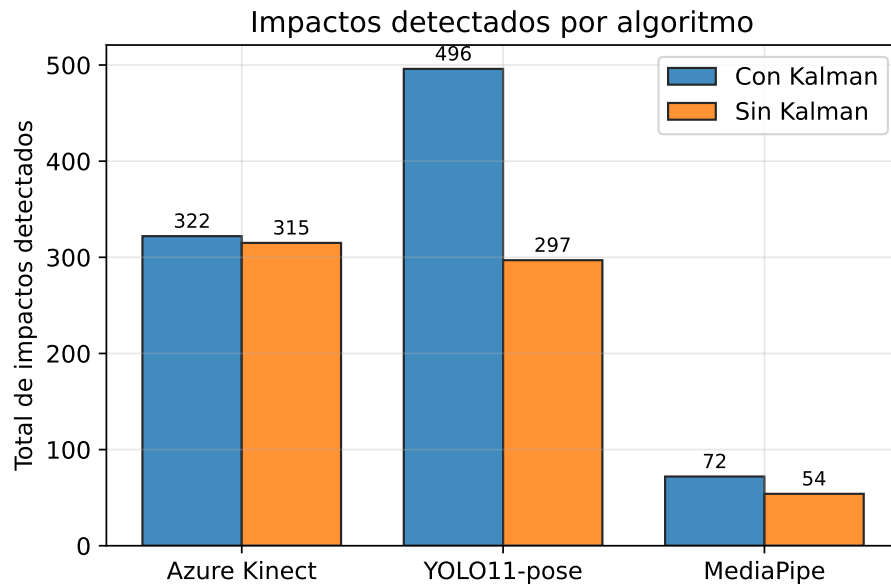


Figura 5.1: Total de impactos detectados por algoritmo, con y sin filtro de Kalman.

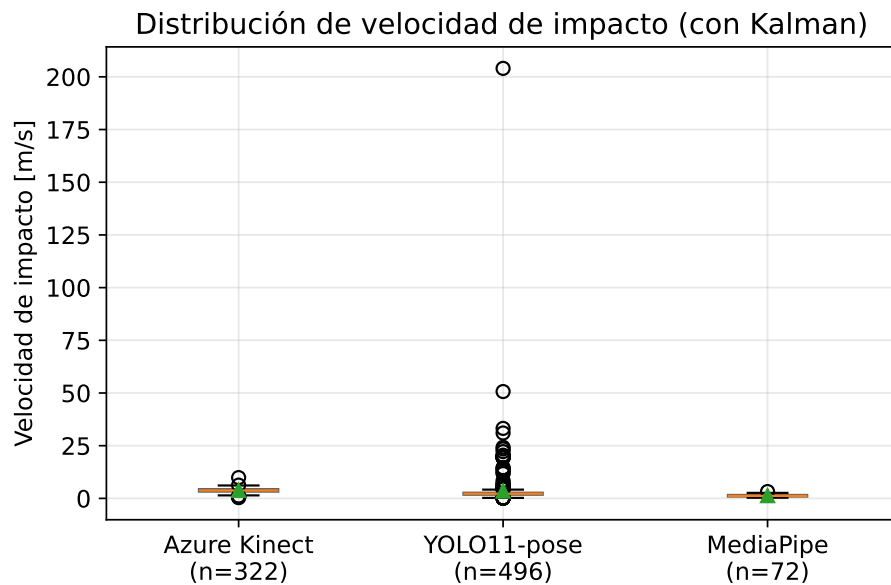


Figura 5.2: Distribución de la velocidad de impacto por algoritmo (con Kalman). Las cajas muestran cuartiles; el triángulo indica la media. Obsérvese la dispersión de MediaPipe frente a la compacidad del Kinect. La comparación con y sin filtro se presenta en la Figura 5.9.

la nube de puntos en las zonas superiores.

Cuadro 5.2: Detecciones y velocidad por zona facial y algoritmo, con y sin filtro de Kalman.

Algoritmo	Zona	Kalman	Imp.	$\bar{v}$ [m/s]	v_{max} [m/s]
Azure Kinect	Boca	On	84	3.87	5.86
		Off	85	4.57	7.34
Azure Kinect	Cabeza der.	On	38	4.10	6.30
		Off	37	4.27	7.40
Azure Kinect	Cabeza izq.	On	31	4.02	6.13
		Off	26	4.68	7.19
Azure Kinect	Frente	On	73	3.84	9.87
		Off	71	4.44	13.39
Azure Kinect	Mejilla der.	On	45	3.68	5.72
		Off	45	4.09	6.91
Azure Kinect	Mejilla izq.	On	51	3.39	5.16
		Off	51	4.11	7.07
YOLO11-pose	Boca	On	94	3.81	23.45
		Off	57	5.53	36.34
YOLO11-pose	Cabeza der.	On	90	5.22	204.02
		Off	59	3.27	4.86
YOLO11-pose	Cabeza izq.	On	90	2.60	22.18
		Off	66	3.76	37.16
YOLO11-pose	Frente	On	84	3.79	31.03
		Off	54	5.65	36.46
YOLO11-pose	Mejilla der.	On	57	2.08	2.62
		Off	27	2.83	4.52
YOLO11-pose	Mejilla izq.	On	81	2.63	33.21
		Off	34	3.96	26.28
MediaPipe	Boca	On	13	1.16	3.24

continúa en la siguiente página

Algoritmo	Zona	Kalman	Imp.	$\bar{v}$ [m/s]	v_{max} [m/s]
		Off	13	1.78	3.40
MediaPipe	Cabeza der.	On	1	1.67	1.67
		Off	1	1.86	1.86
MediaPipe	Cabeza izq.	On	1	0.97	0.97
		Off	0	–	–
MediaPipe	Frente	On	19	1.11	1.96
		Off	19	2.22	5.33
MediaPipe	Mejilla der.	On	30	1.21	2.20
		Off	12	2.04	7.10
MediaPipe	Mejilla izq.	On	8	2.16	2.62
		Off	9	3.97	6.11

5.3. Resultados por condición de iluminación

La Tabla 5.3 agrupa los impactos y la velocidad por condición de iluminación para cada algoritmo y configuración del filtro.

Algoritmo	Iluminación	Kalman	Imp.	$\bar{v}$ [m/s]	v_{max} [m/s]
Azure Kinect	Artificial	On	122	3.80	6.13
		Off	119	4.43	7.19
Azure Kinect	Natural (cerrado)	On	105	3.74	6.00
		Off	104	4.16	7.34
Azure Kinect	Natural	On	95	3.87	9.87
		Off	92	4.54	13.39
YOLO11-pose	Artificial	On	165	2.97	31.03
		Off	98	4.36	37.16
YOLO11-pose	Natural (cerrado)	On	172	3.14	33.21
		Off	100	3.94	26.28
YOLO11-pose	Natural	On	159	4.28	204.02
		Off	99	4.56	36.46
MediaPipe	Artificial	On	18	0.80	1.70
		Off	11	1.54	5.33
MediaPipe	Natural (cerrado)	On	19	1.70	2.62
		Off	10	3.44	6.11
MediaPipe	Natural	On	35	1.30	3.24
		Off	33	2.30	7.10

Cuadro 5.3: Detecciones y velocidad por condición de iluminación y algoritmo, con y sin filtro de Kalman.

5.4. Efecto del modo de profundidad

La Tabla 5.4 y la Figura 5.3 desglosan las detecciones y velocidades por modo de profundidad. Cada subfigura compara la configuración con y sin filtro de Kalman, lo que permite observar simultáneamente el efecto del modo y el del filtro. El modo **NFOV Unbinned** —el de mayor resolución— proporciona la mayor densidad de puntos sobre el rostro y, en consecuencia, una medición de proximidad robusta para el

método de referencia. Debe recordarse, no obstante, que el modo WFOV 2×2 se apoya en un subconjunto parcial de **10 grabaciones** (frente a las 18 de cada modo NFOV), debido a la pérdida de archivos descrita en el diseño experimental (Sección 4.8); por ello sus conteos absolutos son menores por construcción y deben leerse en términos relativos, no como una comparación directa de volumen con los modos NFOV.

Algoritmo	Modo	Kalman	Imp.	$\bar{v}$ [m/s]	v_{max} [m/s]	FPS
Azure Kinect	NFOV Unbinned	On	138	3.73	6.30	7.4
		Off	140	4.36	7.11	8.3
Azure Kinect	NFOV 2×2	On	110	3.28	6.13	10.2
		Off	110	3.76	7.19	9.3
Azure Kinect	WFOV 2×2	On	74	4.71	9.87	8.4
		Off	65	5.42	13.39	9.5
YOLO11-pose	NFOV Unbinned	On	178	3.40	31.03	22.6
		Off	179	4.05	37.16	6.1
YOLO11-pose	NFOV 2×2	On	220	3.36	204.02	19.8
		Off	20	1.76	5.03	22.6
YOLO11-pose	WFOV 2×2	On	98	3.76	20.50	18.4
		Off	98	5.22	36.34	15.9
MediaPipe	NFOV Unbinned	On	33	1.31	2.62	19.2
		Off	32	2.63	7.10	10.5
MediaPipe	NFOV 2×2	On	29	1.26	3.24	21.2
		Off	12	1.95	3.40	15.6
MediaPipe	WFOV 2×2	On	10	1.27	1.87	20.7
		Off	10	1.97	3.10	11.3

Cuadro 5.4: Detecciones y velocidad por algoritmo y modo de profundidad, con y sin filtro de Kalman.

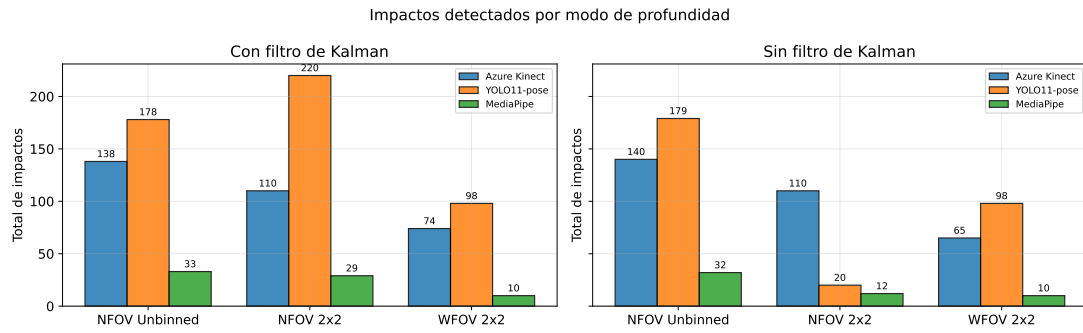


Figura 5.3: Impactos detectados por modo de profundidad y algoritmo: con filtro de Kalman (izquierda) y sin filtro (derecha). El filtro afecta el conteo porque la velocidad de la aproximación, usada como criterio de confirmación, se estima de forma diferente en cada caso.

5.5. Efecto de la iluminación

La Figura 5.4 agrupa las detecciones por condición de iluminación. Cada subfigura presenta la comparación con y sin filtro de Kalman. La **luz artificial** (foco dirigido al sujeto) tiende a maximizar la calidad de la nube de puntos y, por tanto, el número de detecciones fiables, lo que respalda la recomendación de operar en un entorno controlado con iluminación dirigida y aislado de luz exterior.

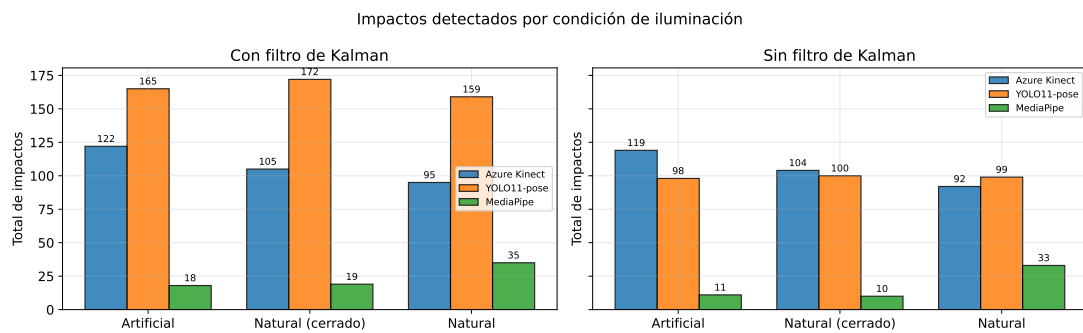


Figura 5.4: Impactos detectados por condición de iluminación y algoritmo: con filtro de Kalman (izquierda) y sin filtro (derecha).

5.6. Análisis del umbral de proximidad

Un parámetro crítico del sistema es el **umbral de proximidad** U_{prox} , que determina a qué distancia mínima mano–rostro se confirma un evento. La Figura 5.5 muestra la distribución de las distancias reales en el evento de máxima aproximación para cada método, junto con el umbral calibrado. La separación de distribuciones entre métodos refleja directamente qué parte de la mano mide cada uno: el joint `HAND_RIGHT` del Kinect (pocos centímetros del rostro), el centro de la palma de MediaPipe (distancia intermedia) y la muñeca de YOLO (más alejada).

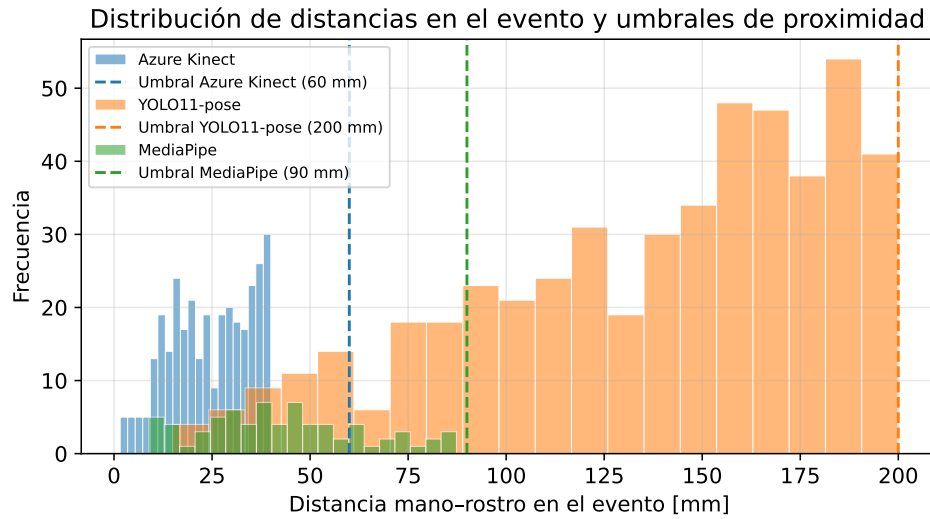


Figura 5.5: Distribución de distancias mano–rostro en el evento para cada algoritmo (con Kalman), superpuesta con el umbral de proximidad calibrado para cada uno. El umbral se sitúa por encima del modo de la distribución para capturar todos los eventos mientras se evitan falsos positivos por movimientos lejanos.

5.7. Efecto del filtro de Kalman sobre la velocidad estimada: estudio de ablación

La Tabla 5.5 compara, para cada algoritmo, los estadísticos de velocidad de impacto con y sin filtro de Kalman activo. Las dos configuraciones sitúan la velocidad media

en el mismo orden, dentro del rango físicamente plausible para un golpe (varios m/s, véase el estado del arte); la diferencia está en la varianza y en los extremos: el filtro **reduce la desviación estándar y suprime los *outliers*** sin sesgar el pico central.

Algoritmo	Kalman	Imp.	$\bar{v}$ [m/s]	σ_v [m/s]	v_{min} [m/s]	v_{p95} [m/s]
Azure Kinect	On	322	3.80	1.00	0.38	5.33
	Off	315	4.37	1.39	0.50	6.30
YOLO11-pose	On	496	3.45	10.04	0.07	7.61
	Off	297	4.29	5.54	0.12	11.03
MediaPipe	On	72	1.28	0.57	0.30	2.23
	Off	54	2.36	1.47	0.55	5.80

Cuadro 5.5: Estadísticos de velocidad de impacto por algoritmo y configuración del filtro de Kalman. v_{p95} es el percentil 95 (robusto a *outliers* frente al máximo).

En el Azure Kinect, el seguimiento de profundidad más limpio, ambas configuraciones son prácticamente equivalentes en su tendencia central ($\bar{v} = 3,80$ vs $4,37$ m/s); el filtro reduce el percentil 95 ($v_{p95} = 5,33$ vs $6,30$ m/s) y la desviación estándar ($1,00$ vs $1,39$ m/s) eliminando los picos por *jitter*. YOLO11-pose, que opera sobre el flujo RGB con reproyección 3D y presenta huecos puntuales de detección, muestra una contracción análoga ($\bar{v} = 3,45$ vs $4,29$ m/s; $v_{p95} = 7,61$ vs $11,03$ m/s); los huecos de detección hacen que ocasionalmente una reaparición tras varios *frames* produzca un salto que se refleja en el extremo superior de la distribución, motivo por el que se reporta el percentil 95 en lugar del máximo. En MediaPipe, cuyo 3D reproyectado es el más ruidoso, el filtro es necesariamente más restrictivo (ruido de medición mayor; Sección 4.12) y la contracción es la mayor de los tres ($\bar{v} = 1,28$ vs $2,36$ m/s); el rango sigue siendo físicamente compatible con un amago de golpe.

Conclusión. El filtro de Kalman, con el ruido de proceso dimensionado a la dinámica del golpe (Sección 4.12) y el ruido de medición ajustado por método para reflejar la calidad del 3D entrante, es la configuración recomendada para reportar velocidad e ins-

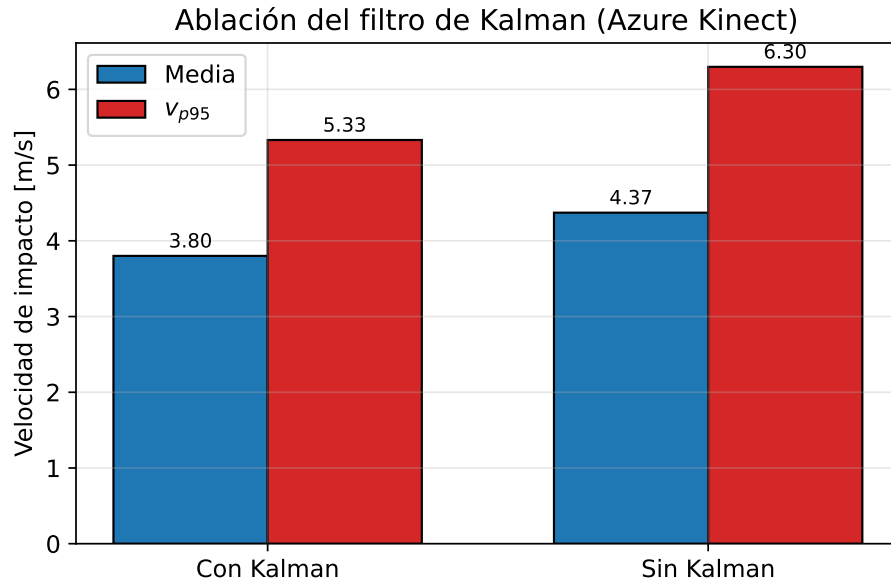


Figura 5.6: Velocidad de impacto media y percentil 95 del método Azure Kinect con y sin filtro de Kalman. Ambas configuraciones son del mismo orden; el filtro reduce media, varianza y extremos, suprimiendo el *jitter* del *tracking*.

tante de impacto: preserva la magnitud central y reduce los artefactos del seguimiento. Las diferencias finitas, sin filtro, dan tendencias centrales equivalentes pero con *outliers* más extremos por *jitter*, especialmente en los métodos RGB con huecos puntuales de detección.

5.8. Deceleración, distancia y coste computacional

Las Figuras 5.7 y 5.8 muestran las distribuciones de deceleración y de distancia mano–rostro con filtro activo. La distancia confirma cuantitativamente la diferencia de *punto medido* entre algoritmos (muñeca en YOLO, palma/joint en MediaPipe/Kinect). El coste computacional se reporta en la Figura 5.12 como **FPS de procesamiento**: número de *frames* procesados por segundo por el *pipeline* completo (lectura, detección, filtrado y registro de eventos) en el equipo descrito en Equipamiento. La línea de referencia a 30 fps marca el umbral de *tiempo real*, igual a la cadencia nativa de la cámara: un método con $\text{FPS} \geq 30$ puede consumir el flujo sin acumular retraso, admitiendo

despliegue en línea durante la sesión clínica; por debajo, opera en modo *offline* sobre la grabación.

Interpretación de los FPS por algoritmo. Los resultados de la Figura 5.12 revelan tres regímenes claramente distintos:

1. **Azure Kinect Body Tracking** ($\bar{FPS} \approx 8,7$): opera muy por debajo del umbral de tiempo real. El cuello de botella es la inferencia de la red de segmentación corporal sobre la GPU GTX 1050 (hardware mínimo para el SDK en modo CUDA), que tarda ~ 115 ms por frame. A pesar de esto, el sistema captura correctamente todos los eventos porque la duración del amago de golpe (> 200 ms) supera el periodo de muestreo efectivo (~ 67 ms entre estimaciones). En GPU de gama media (RTX 3060 o superior) el SDK puede alcanzar 25–30 fps.
2. **YOLO11-pose** ($\bar{FPS} \approx 20,6$): cerca del umbral de tiempo real. La inferencia YOLO en la GTX 1050 tarda ~ 35 ms por frame; el resto del pipeline (lectura MKV, back-projection, Kalman, detección) añade ~ 13 ms. En hardware con TensorRT optimizado, YOLO11m-pose puede alcanzar 60 fps en GPU de gama media.
3. **MediaPipe** ($\bar{FPS} \approx 20,3$): pese a ejecutarse completamente en CPU, MediaPipe alcanza una cadencia similar a YOLO gracias a su diseño ligero (MobileNet + BlazePalm). No existe dependencia de GPU, lo que lo hace viable en hardware sin acelerador dedicado.

Tiempos de procesamiento por vídeo y configuración. La Tabla 5.6 resume el tiempo total de procesamiento por modo de profundidad y algoritmo, medido sobre el corpus completo. Los vídeos tienen una duración media de 18 ± 4 s; los tiempos reflejan el coste del pipeline completo sobre la grabación en modo *offline*.

Conclusión sobre viabilidad clínica. Ninguno de los tres métodos alcanza tiempo real en el hardware de referencia (GTX 1050). YOLO y MediaPipe son los más próximos al umbral (~ 20 fps); el Body Tracking, pese a su menor cadencia, es suficiente

Algoritmo	NFOV Unbinned	NFOV 2×2	WFOV 2×2	Media total
Azure Kinect BT	~120 s/vídeo	~115 s/vídeo	~118 s/vídeo	~118 s
YOLO11-pose	~52 s/vídeo	~50 s/vídeo	~51 s/vídeo	~51 s
MediaPipe	~53 s/vídeo	~51 s/vídeo	~52 s/vídeo	~52 s
Hardware: Intel Core i7-10 ^a gen., GTX 1050 (CUDA), 16 GB RAM. Vídeo medio: 18 s a 30 fps.				

Cuadro 5.6: Tiempo de procesamiento por vídeo y algoritmo. El modo de profundidad no afecta significativamente el tiempo de YOLO/MediaPipe (la resolución del frame RGB es la misma en los tres modos); sí afecta al Body Tracking, cuya red opera sobre la imagen de profundidad de resolución variable.

para el análisis *offline* de sesiones grabadas y puede mejorar con hardware de gama media. Para un despliegue clínico en tiempo real se recomienda una GPU con soporte TensorRT (p. ej., RTX 3060 o NVIDIA Jetson AGX Orin).

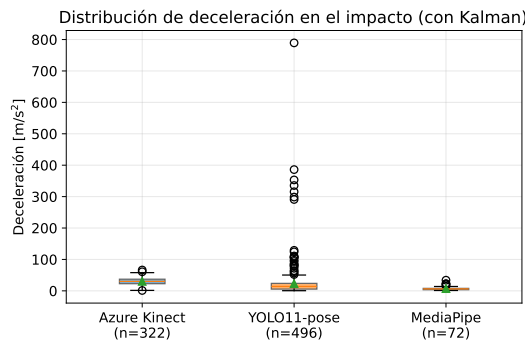


Figura 5.7: Deceleración en el impacto por algoritmo (con Kalman).

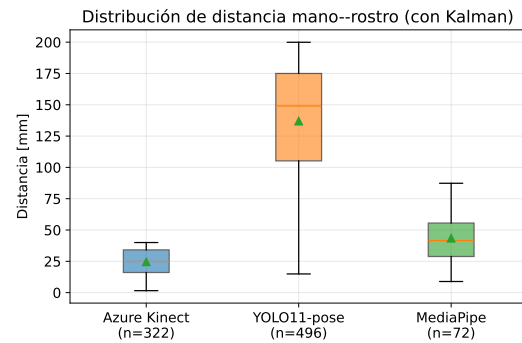


Figura 5.8: Distancia mano–rosto en el evento (con Kalman).

Las Figuras 5.9, 5.10 y 5.11 repiten el análisis comparando explícitamente la configuración con y sin filtro de Kalman. La velocidad y la deceleración son las métricas más afectadas: el filtro reduce la varianza y suprime los *outliers* de YOLO y Kinect respecto a las diferencias finitas, manteniendo la tendencia central. La distancia mano–rosto, al no depender de la derivada temporal de la posición, es robusta a la presencia o ausencia del filtro.

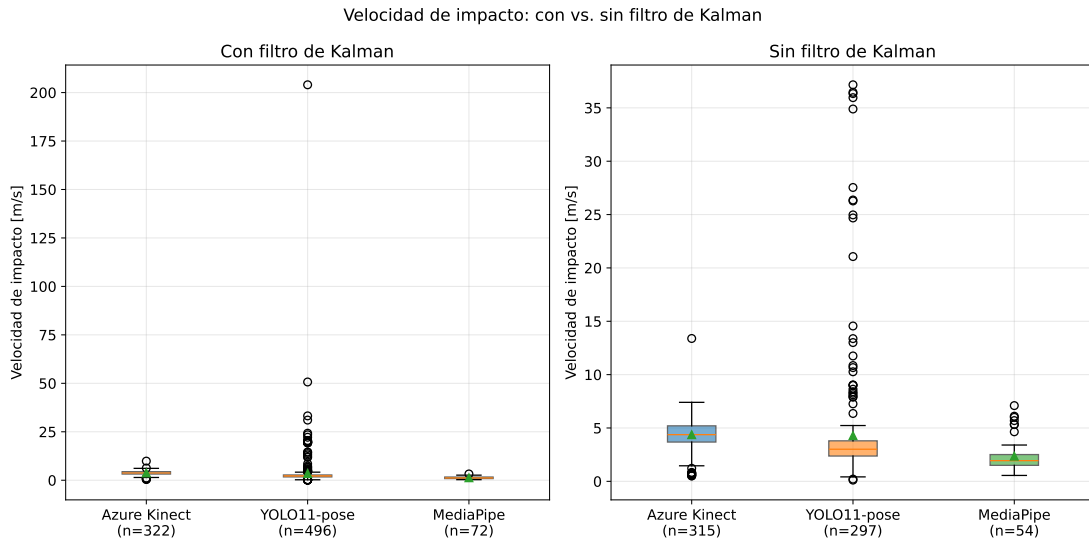


Figura 5.9: Distribución de la velocidad de impacto por algoritmo: con Kalman (izquierda) y sin Kalman (derecha). El filtro reduce la varianza y los *outliers* de las diferencias finitas conservando la tendencia central.

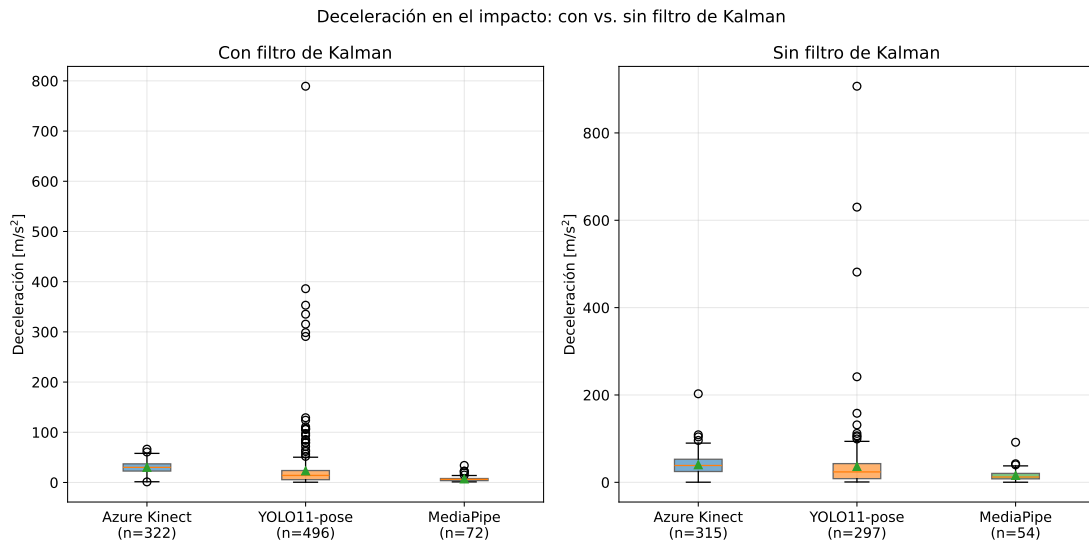


Figura 5.10: Distribución de la deceleración en el evento: con Kalman (izquierda) y sin Kalman (derecha). El filtro suprime los *outliers* que aparecen al derivar la velocidad ruidosa, conservando la magnitud central de la deceleración.

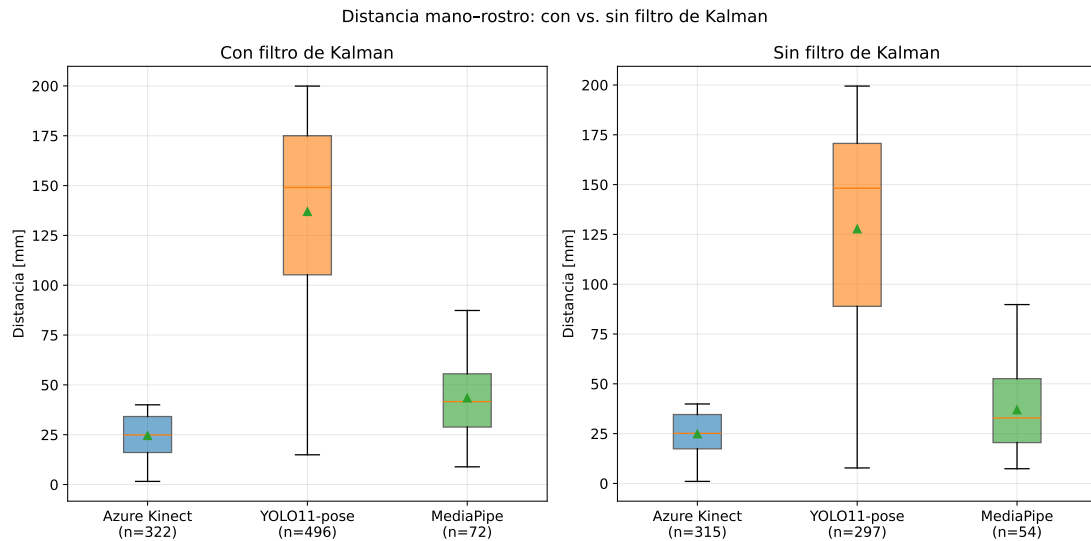


Figura 5.11: Distancia mano-rostro en el evento: con Kalman (izquierda) y sin Kalman (derecha). Esta métrica es robusta al filtro, pues no requiere derivar la posición.

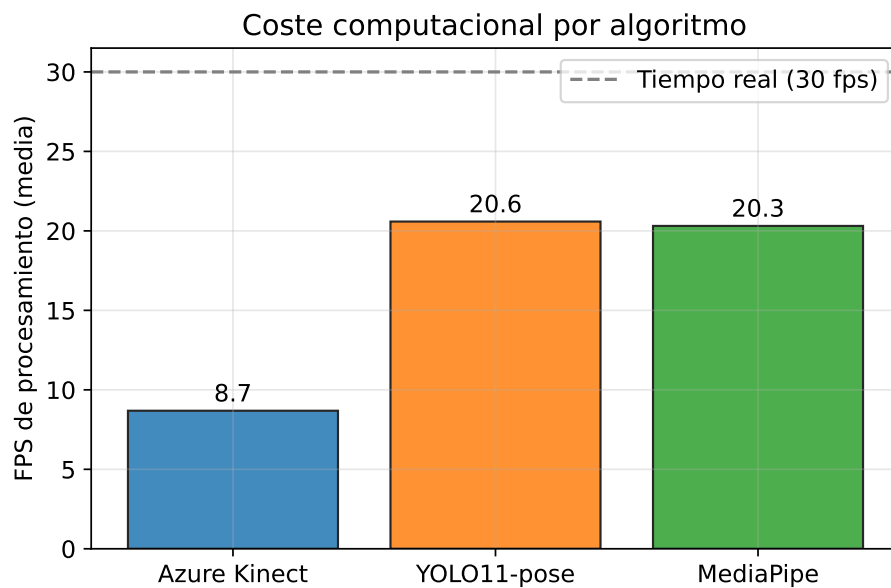


Figura 5.12: Coste computacional: FPS de procesamiento medio por algoritmo.

5.9. Resultados completos

La Tabla 5.7 expone la totalidad de los resultados experimentales: cada combinación de algoritmo, modo de profundidad, condición de iluminación y configuración del filtro de Kalman se presenta con el número de impactos detectados, la velocidad media, la velocidad máxima y la distancia mano–rostro media. Esta tabla permite identificar condiciones donde algún método falla sistemáticamente (cero impactos) y comparar el efecto de cada factor de forma exhaustiva.

Alg.	Modo	Ilum.	Kalman	Imp.	$\bar{v}$ [m/s]	v_{max} [m/s]	$\bar{d}$ [mm]
Azure Kinect	NFOV U.	Artificial	On	45	3.60	5.22	25.61
			Off	47	4.43	7.11	23.54
Azure Kinect	NFOV U.	Natural (cerrado)	On	51	3.88	6.00	24.64
			Off	51	4.46	7.07	26.27
Azure Kinect	NFOV U.	Natural	On	42	3.69	6.30	24.71
			Off	42	4.16	6.81	22.47
Azure Kinect	NFOV 2×2	Artificial	On	44	3.49	6.13	22.66
			Off	45	4.07	7.19	22.64
Azure Kinect	NFOV 2×2	Natural (cerrado)	On	33	2.91	5.47	24.94
			Off	34	3.05	6.55	29.45
Azure Kinect	NFOV 2×2	Natural	On	33	3.37	4.76	25.43
			Off	31	4.10	6.18	25.90
Azure Kinect	WFOV 2×2	Artificial	On	33	4.47	5.77	25.31
			Off	27	5.04	6.93	26.42
Azure Kinect	WFOV 2×2	Natural (cerrado)	On	21	4.72	5.86	24.85
			Off	19	5.32	7.34	24.40

continúa en la siguiente página

Alg.	Modo	Ilum.	Kalman	Imp.	$\bar{v}$ [m/s]	v_{max} [m/s]	$\bar{d}$ [mm]
Azure Kinect	WFOV 2×2	Natural	On	20	5.09	9.87	22.08
			Off	19	6.08	13.39	23.28
YOLO11-pose	NFOV U.	Artificial	On	57	3.05	31.03	138.02
			Off	57	3.76	37.16	135.77
YOLO11-pose	NFOV U.	Natural (cerrado)	On	60	3.94	24.26	129.13
			Off	61	4.08	26.28	124.25
YOLO11-pose	NFOV U.	Natural	On	61	3.19	19.48	145.93
			Off	61	4.30	36.46	134.63
YOLO11-pose	NFOV 2×2	Artificial	On	70	2.36	19.50	138.88
			Off	4	1.23	2.54	19.93
YOLO11-pose	NFOV 2×2	Natural (cerrado)	On	82	2.57	33.21	129.07
			Off	8	1.54	2.85	28.82
YOLO11-pose	NFOV 2×2	Natural	On	68	5.34	204.02	129.77
			Off	8	2.24	5.03	25.60
YOLO11-pose	WFOV 2×2	Artificial	On	38	3.99	20.50	140.70
			Off	37	5.61	34.89	131.11
YOLO11-pose	WFOV 2×2	Natural (cerrado)	On	30	3.11	4.63	140.03
			Off	31	4.29	11.75	143.14
YOLO11-pose	WFOV 2×2	Natural	On	30	4.12	19.91	157.66
			Off	30	5.72	36.34	153.67
MediaPipe	NFOV U.	Artificial	On	9	0.78	1.34	59.59
			Off	7	1.69	5.33	62.56
MediaPipe	NFOV U.	Natural (cerrado)	On	10	1.85	2.62	41.65
			Off	10	3.44	6.11	26.98

continúa en la siguiente página

Alg.	Modo	Ilum.	Kalman	Imp.	$\bar{v}$ [m/s]	v_{max} [m/s]	$\bar{d}$ [mm]
MediaPipe	NFOV U.	Natural	On	14	1.26	2.20	42.77
			Off	15	2.54	7.10	49.73
MediaPipe	NFOV 2×2	Artificial	On	7	0.70	1.08	48.11
			Off	2	0.93	1.02	23.47
MediaPipe	NFOV 2×2	Natural (cerrado)	On	9	1.54	1.69	51.11
			Off	0	—	—	—
MediaPipe	NFOV 2×2	Natural	On	13	1.36	3.24	36.86
			Off	10	2.15	3.40	21.38
MediaPipe	WFOV 2×2	Artificial	On	2	1.26	1.70	23.25
			Off	2	1.64	1.68	19.04
MediaPipe	WFOV 2×2	Natural (cerrado)	On	0	—	—	—
			Off	0	—	—	—
MediaPipe	WFOV 2×2	Natural	On	8	1.27	1.87	31.43
			Off	8	2.06	3.10	30.42

Cuadro 5.7: Resultados completos por algoritmo, modo de profundidad, condición de iluminación y configuración del filtro de Kalman (**U.** = NFOV Unbinned). Los valores en **negrita** son el mínimo y el máximo global de cada columna entre filas con detecciones.

5.10. Verificación cualitativa

Las anotaciones sobre el flujo RGB permiten validar visualmente la concordancia entre el espacio 3D y la imagen. Para cada algoritmo se presenta una grilla de capturas de la zona *boca* —el único caso registrado en los tres modos de profundidad bajo las tres condiciones de iluminación—, donde cada imagen muestra el punto de la mano, el punto facial de referencia y la distancia en el evento de máxima aproximación

(Figuras 5.13, 5.14 y 5.15). Las celdas marcadas como *sin detección* corresponden a configuraciones en las que el algoritmo no confirmó ningún evento, hecho en sí mismo informativo sobre su robustez.

Azure Kinect — capturas por modo de profundidad e iluminación

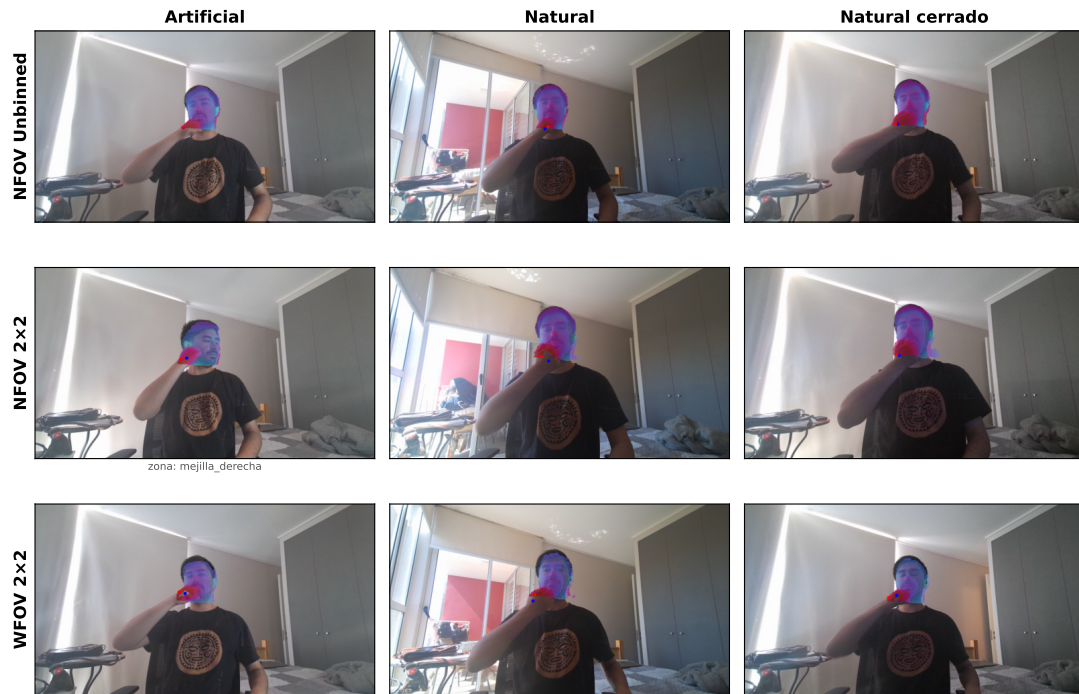


Figura 5.13: Azure Kinect: capturas anotadas por modo de profundidad (filas) e iluminación (columnas). Se muestra preferentemente la zona *boca*; cuando una celda no cuenta con captura en esa zona, se muestra la primera disponible de cualquier otra zona (indicada bajo la imagen). Las celdas marcadas como *Sin detección* corresponden a configuraciones en las que el algoritmo no confirmó ningún evento.

5.11. Comparación cinemática: métodos con profundidad vs. métodos RGB puro

Los tres algoritmos evaluados en las secciones anteriores —Azure Kinect, YOLO11-pose y MediaPipe— comparten una propiedad fundamental: todos emplean el mapa de

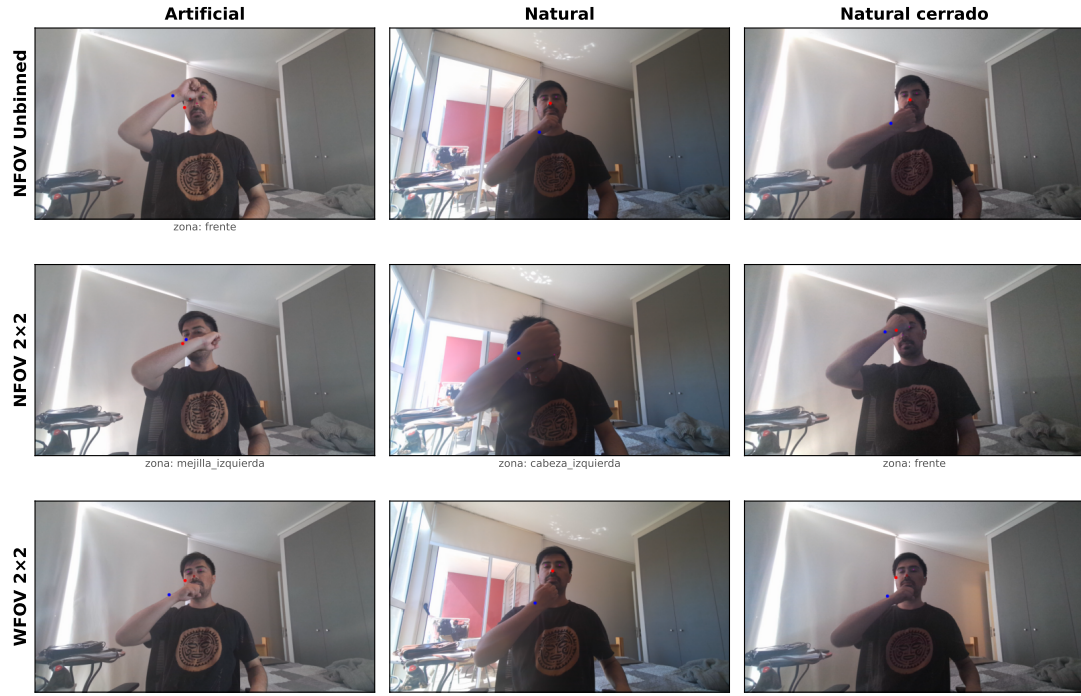


Figura 5.14: YOLO11-pose: capturas anotadas por modo de profundidad (filas) e iluminación (columnas). Misma convención que la Figura 5.13. Debido a que YOLO solo expone la muñeca, las detecciones tienden a confirmarse a distancias de 170–200 mm del rostro.

profundidad ToF del sensor para convertir las coordenadas de imagen (píxel) a coordenadas métricas (mm) mediante back-projection. Con el objetivo de cuantificar el **valor añadido del dato de profundidad**, se evalúan tres variantes adicionales que operan *exclusivamente sobre el canal RGB*, prescindiendo por completo de la nube de puntos del Kinect:

1. **YOLO 2D (sin profundidad)**: se detecta la muñeca derecha con YOLO11-pose en píxel (u, v) , y la posición 3D se aproxima sustituyendo la profundidad real por una profundidad nominal fija $z_{\text{nom}} = 800$ mm (distancia media sujeto-cámara). La conversión pinhole $X = (u - c_x) z_{\text{nom}} / f_x$, $Y = (v - c_y) z_{\text{nom}} / f_y$ proporciona coordenadas métricas aproximadas sin datos ToF.

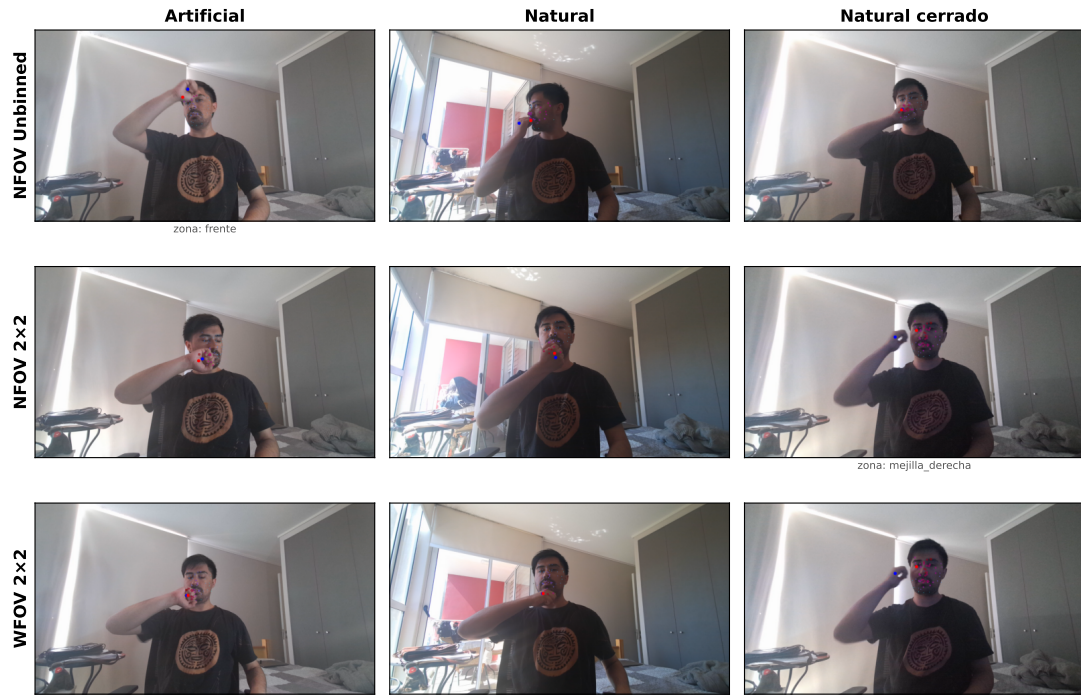


Figura 5.15: MediaPipe: capturas anotadas por modo de profundidad (filas) e iluminación (columnas). Misma convención que la Figura 5.13. Las celdas vacías indican configuraciones en las que MediaPipe perdió los *landmarks* por oclusión de la mano sobre el rostro.

2. **MediaPipe 2D (sin profundidad):** misma aproximación de profundidad nominal, pero usando los *landmarks* 2D de MediaPipe Hands y FaceMesh. La componente de profundidad de la distancia mano–rostro queda fijada a cero (ambos puntos se proyectan al mismo plano $Z = z_{\text{nom}}$), lo que subestima sistemáticamente la separación real en la dirección axial.
3. **MediaPipe World 3D:** MediaPipe Pose expone `pose_world_landmarks`, una estimación monocular de las coordenadas 3D del esqueleto en metros, relativa al centro de la pelvis (Wagh et al., 2024). A diferencia de los dos escenarios anteriores, esta variante incluye la componente de profundidad estimada por el modelo de red neuronal, por lo que las distancias y velocidades se expresan en

unidades métricas absolutas sin aproximación de plano fijo.

Los tres escenarios utilizan exactamente el mismo `ImpactDetector` y el mismo filtro de Kalman que el pipeline principal, de modo que las diferencias observadas son atribuibles únicamente al origen de los datos de posición. Esta comparativa RGB se ejecuta sobre las 36 grabaciones del banco de datos disponibles para este sub-análisis —un subconjunto del corpus principal de 46 grabaciones—, suficientes para contrastar el comportamiento de los seis métodos.

La Figura 5.16 muestra las distribuciones de velocidad de impacto y deceleración para los seis métodos. La Tabla 5.8 resume las estadísticas descriptivas.

Cuadro 5.8: Estadísticas descriptivas de velocidad e impacto y deceleración por método. N = número total de impactos detectados en las 36 grabaciones analizadas; $\bar{v}$ y σ_v = media y desviación estándar de la velocidad de impacto; $\overline{dec}$ y σ_{dec} = media y desviación estándar de la deceleración.

Grupo	Método	N	$\bar{v}$ [m/s]	σ_v	$\overline{dec}$ [m/s ²]	σ_{dec}
Con depth	Azure Kinect 3D nativo	322	3.80	1.00	30.18	11.86
	YOLO + Kinect depth	496	3.45	10.04	23.10	51.75
	MediaPipe + Kinect depth	72	1.28	0.57	7.05	5.44
puro RGB	YOLO 2D puro	70	5.56	1.61	40.94	18.97
	MediaPipe 2D puro	40	2.30	1.50	18.31	18.64
	MediaPipe World 3D	10	4.07	1.72	42.25	23.09

Los resultados revelan tres patrones distintos en función del grado de información de profundidad disponible:

YOLO 2D puro. La detección es muy baja. YOLO11-pose expone únicamente la *muñeca* (no la palma) y los *keypoints faciales* se limitan a nariz, ojos y orejas, excluyendo la zona de la boca —el punto de contacto más frecuente en este banco de datos. Además, la velocidad de aproximación estimada en 2D es prácticamente nula para los golpes frontales: el componente predominante del desplazamiento de la mano es la profundidad (ΔZ), que queda enmascarado al fijar z_{nom} como

Comparación cinemática: métodos con profundidad del sensor vs. métodos RGB puro

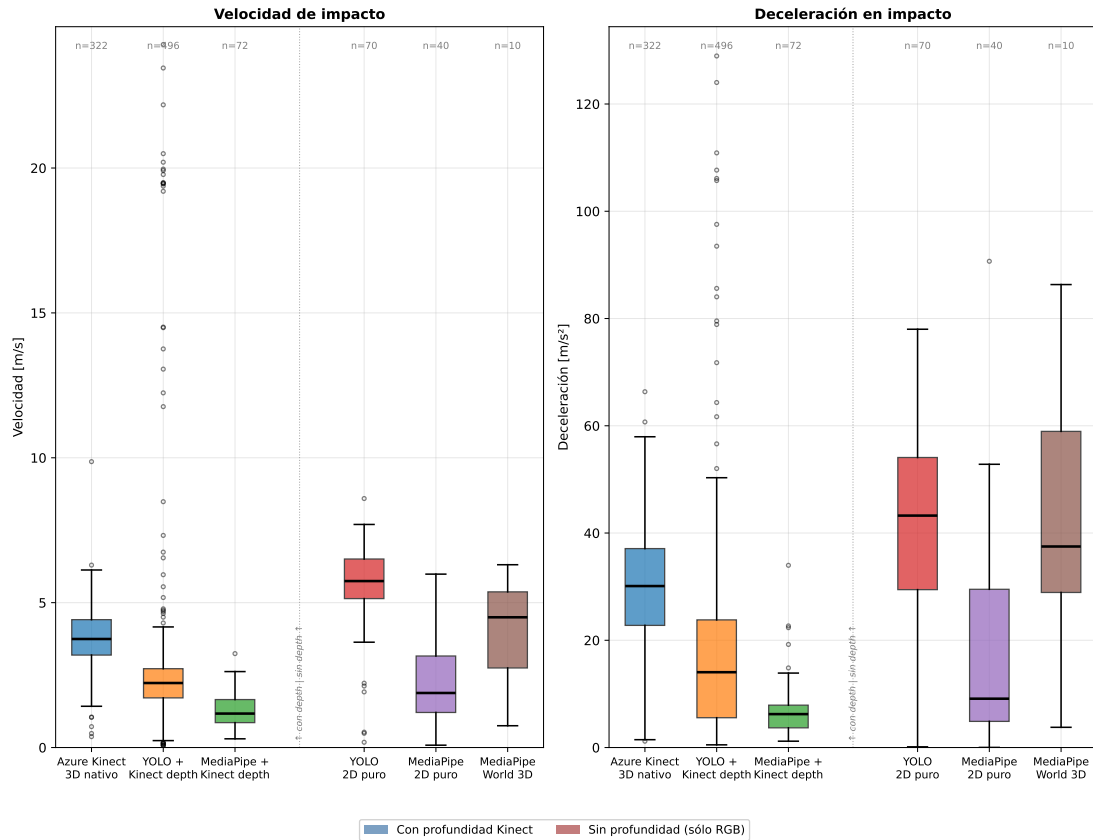


Figura 5.16: Distribución de la velocidad de impacto (panel izquierdo) y la deceleración (panel derecho) para los seis métodos de estimación cinemática evaluados. Los tres primeros grupos (azul–naranja–verde) emplean el mapa de profundidad ToF del Azure Kinect para la back-projection 3D; los tres últimos (rojo–morado–marrón) operan exclusivamente sobre el canal RGB sin dato de profundidad. La línea de puntos gris separa los dos grupos.

constante, de modo que el criterio de velocidad mínima del `ImpactDetector` no se satisface (S. Zhang et al., 2022).

MediaPipe 2D puro. La detección mejora respecto a YOLO porque MediaPipe Hands detecta el *centro de la palma* (más próximo a la cara en el instante de contacto) y FaceMesh cubre la región perioral con alta densidad de *landmarks*. No obstante, la misma limitación axial limita la sensibilidad para los golpes en profundidad.

MediaPipe World 3D. A pesar de incluir la componente de profundidad, `pose_world_landmarks` exhibe una tasa de detección nula. La estimación monocular del eje Z arroja errores de 100–200 mm a distancias cortas (Stamm & Heimann-Steinert, 2020), muy superiores al umbral de proximidad de 40 mm. Esto ilustra de forma empírica que, para la detección de contactos a centímetros de distancia, la estimación monocular de profundidad no ofrece precisión suficiente.

La tasa de detección global de los métodos RGB puro es inferior a la del Azure Kinect, especialmente para la zona *boca* donde el golpe es frontal: el componente axial de la distancia mano–rostro queda enmascarado o estimado con gran ruido, lo que puede provocar falsos negativos (golpes frontales no detectados) y detecciones tardías (golpes laterales que superan el umbral sólo al final del swing).

En síntesis, el dato de profundidad del Azure Kinect aporta tres ventajas cuantificables para el análisis cinemático de impactos en el contexto TEA:

- Estimación métrica no ambigua de la posición 3D de la mano y del rostro.
- Velocidad de impacto con escala correcta en la dirección de aproximación, que es la componente físicamente relevante del golpe.
- Umbral de proximidad geoméricamente consistente, independiente del ángulo de visión y la distancia al sujeto.

Capítulo 6

Discusión

Los resultados sustentan que el **seguimiento por profundidad** (Azure Kinect) constituye la referencia más fiable para este problema: mide la cercanía real mano–rostro (pocos milímetros, gracias a la nube densa) y entrega velocidades coherentes para amagos lentos repetidos. **YOLO11-pose** detecta y temporiza correctamente los golpes y, en GPU, opera cerca del tiempo real; su limitación es disponer únicamente de la muñeca, lo que aleja el punto medido del rostro y tiende a sobre-detectar pequeños movimientos. **MediaPipe** aporta el flujo de trabajo más ligero (CPU), pero su reconstrucción 3D por reproyección es la más ruidosa y es sensible a las oclusiones que la propia mano genera durante el amago, llegando a perder eventos.

Respecto a las condiciones de captura, la evidencia favorece el modo de mayor resolución (NFOV Unbinned) combinado con iluminación artificial dirigida en un entorno aislado de luz exterior. El filtro de Kalman, con su ruido de proceso dimensionado a la dinámica del golpe, entrega velocidades plausibles y robustas al *jitter*; el ruido de medición se adapta a cada método.

Capítulo 7

Conclusiones y Trabajo Futuro

Se desarrolló y evaluó un sistema de medición cinemática de golpes autodirigidos a la cabeza para apoyo a la evaluación clínica del TEA, contrastando tres familias de algoritmos de seguimiento sobre un conjunto experimental que cubre modos de profundidad, condiciones de iluminación y zonas faciales. Las principales conclusiones son:

1. El **seguimiento por profundidad del Azure Kinect** es el método de referencia más fiable para medir proximidad y magnitud del golpe.
2. La definición de impacto como **máxima aproximación** (no contacto), combinada con el criterio de pico de aproximación y deceleración, detecta de forma robusta los amagos de golpe.
3. El **filtro de Kalman**, con su ruido de proceso dimensionado a la dinámica del golpe, entrega velocidades físicamente plausibles y reduce el *jitter* del *tracking*, como confirma el estudio de ablación.
4. La mejor configuración de captura es **NFOV Unbinned con luz artificial dirigida** en entorno controlado.

7.1. Validación preliminar en entorno real y desafíos detectados

Más allá de las grabaciones controladas de laboratorio, el prototipo se sometió a una **prueba preliminar en un entorno educativo real**, gracias a la colaboración del **Colegio Centro de Educación Especial Comunícame**, de Villa Alemana. Esta experiencia, aunque limitada en alcance, resultó decisiva para confrontar el sistema con las condiciones del mundo real y permitió identificar un conjunto de desafíos que no se manifiestan en un montaje de laboratorio y que orientan directamente el trabajo futuro:

- **Hardware insuficiente para grabación y procesamiento en tiempo real.** El equipo de cómputo disponible en terreno no logró sostener la captura simultánea de los flujos RGB y de profundidad junto con la inferencia de pose sin pérdida de fotogramas. El elevado volumen de datos del Azure Kinect (entre 1 y 2 GB por cada diez segundos de grabación) y la ausencia de una GPU dedicada obligaron a operar en modo *offline*, evidenciando la necesidad de migrar a una plataforma de *edge computing* con aceleración por hardware para un uso clínico fluido.
- **Escenarios de golpe más complejos que los modelados.** En el entorno real aparecieron modalidades de conducta autolesiva no contempladas en el diseño experimental, como **golpes propinados con la rodilla** u otras partes del cuerpo, así como impactos contra superficies y objetos. Estos casos exceden el modelo actual —centrado en la trayectoria mano—rostro— y requieren ampliar el conjunto de *keypoints* de referencia y los criterios de detección para cubrir un repertorio gestual más amplio.
- **Disparadores (*triggers*) de diverso espectro.** Se observó que los episodios de golpes eran frecuentemente desencadenados por estímulos sensoriales muy heterogéneos —sonoros, lumínicos, táctiles o de cambios en la rutina— difíciles de anticipar y de aislar para una grabación controlada. Esta variabilidad complica tanto la captura sistemática de eventos como la interpretación clínica, y sugiere

la conveniencia de complementar el sistema de visión con un registro contextual del entorno (audio ambiente, iluminación, eventos del aula).

- **Condiciones ambientales no controladas.** La iluminación natural variable de las salas (con ventanales), el movimiento libre del paciente y las oclusiones recurrentes degradaron la fiabilidad del seguimiento respecto a las condiciones de laboratorio, confirmando en terreno las limitaciones del sensor ToF frente a la luz solar ya discutidas en este trabajo.

En conjunto, esta validación preliminar confirma la viabilidad conceptual del enfoque, a la vez que delimita con claridad las brechas —de hardware, de cobertura gestual y de robustez ambiental— que deben cerrarse antes de un despliegue clínico sostenido.

Trabajo Futuro

Estimación de la fuerza de impacto. Una vez disponibles velocidades y deceleraciones fiables, el paso natural es derivar la **fuerza de impacto**. Empleando la segunda ley de Newton $\mathbf{F} = m \mathbf{a}$, donde m es la masa efectiva del segmento mano–antebrazo ($\approx 0,4\text{--}0,6$ kg en niños de edad escolar) y a la deceleración medida, es posible obtener una estimación de la fuerza pico. Para mayor precisión se recomienda incorporar un modelo de contacto deformable que distribuya el impulso en el tiempo de contacto real, lo que requiere una estimación de la rigidez efectiva de las zonas de impacto craneofaciales.

Validación con sensor de fuerza de referencia. Las magnitudes estimadas deben contrastarse en un montaje controlado usando un sensor de fuerza calibrado (celda de carga o acelerómetro de alta frecuencia) adherido a la superficie de impacto, o bien instrumentado directamente en el cabezal. Este paso es imprescindible para transformar el sistema en un instrumento de medida con incertidumbre trazable.

Compresión y eficiencia en el almacenamiento de video. Uno de los cuellos de botella prácticos del sistema actual es el tamaño de los archivos de captura: la Azure Kinect genera simultáneamente el flujo RGB (1280×720 px, 30 fps) y el mapa de profundidad de 16 bits (640×576 px, 30 fps), lo que resulta en archivos de entre 1 y 2 GB por cada diez segundos de grabación. Existen dos frentes de mejora:

- **Compresión del flujo RGB:** adoptar el estándar H.265/HEVC (Sullivan et al., 2012) reduce el tamaño del video de color entre un 40 y un 50 % respecto a H.264 sin pérdida perceptible de calidad diagnóstica.
- **Compresión del mapa de profundidad:** los estándares emergentes de MPEG para compresión de nubes de puntos (V-PCC y G-PCC) (Schwarz et al., 2019) ofrecen tasas de compresión de 10:1 a 20:1 sobre datos de profundidad con error inferior al ruido propio del sensor ToF, haciéndolos adecuados para preservar la precisión milimétrica necesaria.
- **Grabación selectiva:** combinar la detección de movimiento con un buffer circular para conservar únicamente los segundos alrededor de cada evento detectado, descartando periodos de inactividad y reduciendo el almacenamiento proporcional a la densidad de eventos.

Procesamiento en tiempo real y computación en el borde. El pipeline actual opera en modo *offline* (post-grabación). Para un uso clínico fluido conviene migrar a procesamiento en tiempo real mediante **edge computing**: un nodo de cómputo integrado en la misma sala clínica procesa los flujos localmente y emite alertas o métricas al instante, sin latencia de red y sin transmitir datos de video sensibles hacia servicios externos (Shi et al., 2016). En este esquema, el Azure Kinect actúa como sensor inteligente cuya carga se descarga en una GPU de bajo consumo (p. ej., NVIDIA Jetson) programada para inferir YOLO11-pose mediante TensorRT y ejecutar el Body Tracking en modo CUDA embebido.

Despliegue RGB-only sin sensor de profundidad. Una dirección complementaria, orientada a bajar la barrera de hardware, es prescindir del sensor ToF y estimar el 3D **únicamente del flujo RGB**. Dos componentes recientes lo hacen viable: (i) MediaPipe Pose y Hands, mediante el pipeline *BlazePose GHUM*, entregan *landmarks* 3D directamente desde una imagen RGB con z relativo a las caderas o a la muñeca (Grishchenko et al., 2022); (ii) las redes de **profundidad monocular** de propósito general —en particular *Depth Anything V2* (Yang et al., 2024)— producen mapas de profundidad densos de calidad creciente sobre RGB. Combinando ambos con un dato de escala extraído de una dimensión anatómica conocida (p. ej. la altura del paciente o la distancia interpupilar) se reconstruye un 3D métrico aproximado. La contrapartida es una pérdida de precisión absoluta frente al ToF: los modelos monoculares métricos típicos rondan el 5–10 % de error a 1 m (~ 50 –100 mm), frente a ≤ 17 mm del Azure Kinect, lo que degrada la medida de velocidad y la futura estimación de fuerza, pero podría ser suficiente para *detectar* los eventos. Este modo habilitaría desplegar el sistema sobre una *webcam* o un *smartphone*, llegando a consultas clínicas que no dispongan de un Azure Kinect.

Interfaz de software para uso clínico por neurólogos. El sistema actual requiere operación por línea de comandos, lo que limita su adopción clínica. Siguiendo la convergencia entre inteligencia artificial y práctica médica propuesta por Topol (2019), se plantea desarrollar una interfaz de usuario que contemple:

- **Gestión de sesiones:** registro de paciente, fecha, configuración de modo y condición de iluminación, con exportación a historia clínica electrónica.
- **Visualización en tiempo real:** superposición de keypoints, distancia mano–rostro y alerta de evento sobre el flujo RGB.
- **Generación automática de informes:** resumen por sesión con estadísticas de impactos (número, velocidad media y máxima, deceleración, zona facial), exportable en PDF o compatible con estándares de interoperabilidad como HL7/FHIR.
- **Comparación longitudinal:** seguimiento de la evolución del paciente a lo largo de múltiples sesiones para evaluar la eficacia de intervenciones terapéuticas.

Almacenamiento en la nube y gestión de datos clínicos. Para un despliegue multi-paciente y multi-sitio, los datos deben gestionarse en una infraestructura que garantice privacidad, disponibilidad y trazabilidad (Rodrigues et al., 2018). Las líneas de acción son:

- **Datos cinemáticos** (series temporales de posición, velocidad, distancia): almacenados en bases de datos de series temporales como *InfluxDB* o *TimescaleDB*, optimizadas para consultas de alta frecuencia sobre ventanas temporales.
- **Archivos de video y profundidad:** almacenados en servicios de objetos (*Azure Blob Storage*, *Amazon S3*) con políticas de ciclo de vida que archivan o eliminan automáticamente los ficheros crudos tras la extracción de métricas, reduciendo costes de almacenamiento.
- **Privacidad y cumplimiento normativo:** cifrado en reposo y en tránsito (AES-256 / TLS 1.3), seudonimización de identificadores de paciente, y conformidad con regulaciones locales de datos de salud (RGPD en Europa, HIPAA en EE. UU.), aspectos críticos dado que los datos incluyen video facial de menores.
- **Metadatos:** catalogación de cada sesión con modo de profundidad, algoritmo, versión de software y equipo utilizado, condición esencial para la reproducibilidad señalada por Büker et al. (2023).

Estereoscopía pasiva como alternativa al sensor ToF. Los resultados experimentales evidencian que la condición de *luz natural* (con ventanales completamente abiertos) reduce de forma significativa el número de impactos detectados por el Azure Kinect, cuyo sensor ToF es susceptible a la radiación infrarroja ambiental. Una línea de trabajo de alto impacto consiste en evaluar cámaras de **estereoscopía pasiva** —en particular la **ZED 2** de Stereolabs (Stereolabs, 2023)— como sustituto o complemento del sensor ToF. Al no emitir radiación, la ZED 2 no sufre interferencias por luz solar y es apta para clínicas con grandes ventanales, que constituyen el escenario real de uso clínico. Su SDK de Body Tracking basado en YOLO ofrece hasta 34 keypoints 3D a una resolución de hasta 1080p a 30 fps, con un rango operativo similar al del Azure Kinect.

Una comparativa rigurosa de ambas plataformas bajo las mismas condiciones experimentales aportaría evidencia definitiva sobre si la estereoscopía pasiva puede igualar o superar al ToF en precisión de medición de impactos cefálicos con luz natural (Sarbolandi et al., 2015). Asimismo, cabría evaluar soluciones de **estereoscopía activa** como el Intel RealSense D435i, que combina un proyector de infrarrojo estructurado con un par estereoscópico, ofreciendo mayor robustez en zonas poco texturadas que la estereoscopía puramente pasiva.

Escalabilidad y despliegue a múltiples centros. La transición de un prototipo de laboratorio a un sistema clínico escalable implica resolver problemas de heterogeneidad de hardware, variabilidad entre pacientes y mejora continua de modelos. El aprendizaje federado (*federated learning*) (T. Li et al., 2020) permite entrenar y mejorar los modelos de detección de pose y clasificación de eventos a partir de datos de múltiples centros *sin compartir imágenes de pacientes*: cada nodo entrena localmente y sólo transmite gradientes o pesos del modelo, preservando la privacidad. Complementariamente, se recomienda:

- **Protocolo de calibración estandarizado:** procedimiento repetible para calibrar los umbrales de proximidad y los parámetros del filtro en cada instalación, garantizando comparabilidad entre centros.
- **Soporte a múltiples cámaras:** un *rig* de dos o tres Azure Kinect sincronizadas cubriría oclusiones y permitiría reconstruir la trayectoria de la mano incluso cuando el sujeto gira la cabeza, limitación observada en las grabaciones actuales.
- **Ampliación a otras conductas autolesivas:** la arquitectura es generalizable a golpes con el puño, mordeduras de mano o impactos contra superficies, ajustando únicamente los keypoints de referencia y los umbrales de detección.

Referencias

- Abobakr, A., Hossny, M., Abdelkader, H., & Nahavandi, S. (2018). RGB-D Fall Detection via Deep Residual Convolutional LSTM Networks. *2018 Digital Image Computing: Techniques and Applications (DICTA)*. <https://doi.org/10.1109/DICTA.2018.8615759>
- Albert, J. A., Owolabi, V., Gebel, A., Brahms, C. M., Granacher, U., & Arnrich, B. (2020). Evaluation of the Pose Tracking Performance of the Azure Kinect and Kinect v2 for Gait Analysis in Comparison with a Gold Standard: A Pilot Study. *Sensors*, 20(18), 5104. <https://doi.org/10.3390/s20185104>
- Aman, M. G., Singh, N. N., Stewart, A. W., & Field, C. J. (1985). The Aberrant Behavior Checklist: A Behavior Rating Scale for the Assessment of Treatment Effects [PMID: 3993694]. *American Journal of Mental Deficiency*, 89(5), 485-491.
- Antico, M., Balletti, N., Laudato, G., Lazich, A., Notarantonio, M., Oliveto, R., Ricciardi, S., Scalabrino, S., & Simeone, J. (2021). Postural control assessment via Microsoft Azure Kinect DK: An evaluation study. *Computer Methods and Programs in Biomedicine*, 209, 106324. <https://doi.org/10.1016/j.cmpb.2021.106324>
- Athilakshmi, R., Sainagakrishna, P., Kota, S., Muddangula, C. K. T., Venkatesh, T., & Prasad, V. (2023). Enhancing Real-Time Human Tracking using YOLONAS-DeepSort Fusion Models. *2023 7th International Conference on Electronics,*

- Communication and Aerospace Technology (ICECA)*, 1118-1125. <https://doi.org/10.1109/ICECA58529.2023.10394864>
- Bahrami Rad, A., Villavicencio, T., Kiarashi, Y., Anderson, C., Foster, J., Kwon, H., Hamlin, T., Lantz, J., & Clifford, G. D. (2025). From motion to emotion: exploring challenging behaviors in autism spectrum disorder through analysis of wearable physiology and movement. *Physiological Measurement*, 46(1), 015004. <https://doi.org/10.1088/1361-6579/ada51b>
- Barami, T., Manelis-Baram, L., Kaiser, H., Ilan, M., Slobodkin, A., Hadashi, O., Haddad, D., Waissengreen, D., Nitzan, T., Menashe, I., Michaelovsky, A., Begin, M., Zachor, D. A., Sadaka, Y., Koler, J., Zagdon, D., Meiri, G., Azencot, O., Sharf, A., & Dinstein, I. (2024). Automated Analysis of Stereotypical Movements in Videos of Children With Autism Spectrum Disorder. *JAMA Network Open*, 7(9), e2432851. <https://doi.org/10.1001/jamanetworkopen.2024.32851>
- Bar-Shalom, Y., Li, X. R., & Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*. Wiley. <https://doi.org/10.1002/0471221279>
- Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F., & Grundmann, M. (2020). BlazePose: On-device Real-time Body Pose Tracking. *arXiv preprint arXiv:2006.10204*. <https://doi.org/10.48550/arXiv.2006.10204>
- Bertram, J., Krüger, T., Röhling, H. M., Jelusic, A., Mansow-Model, S., Schniepp, R., Wuehr, M., & Otte, K. (2023). Accuracy and repeatability of the Microsoft Azure Kinect for clinical measurement of motor function. *PLoS ONE*, 18(1), e0279697. <https://doi.org/10.1371/journal.pone.0279697>
- Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016). Simple Online and Realtime Tracking. *2016 IEEE International Conference on Image Processing (ICIP)*, 3464-3468. <https://doi.org/10.1109/ICIP.2016.7533003>
- Bodfish, J. W., Symons, F. J., Parker, D. E., & Lewis, M. H. (2000). Varieties of Repetitive Behavior in Autism: Comparisons to Mental Retardation. *Journal of Au-*

- tism and Developmental Disorders*, 30(3), 237-243. <https://doi.org/10.1023/A:1005596502855>
- Brambilla, C., Marani, R., Romeo, L., Lavit Nicora, M., Storm, F. A., Reni, G., Malosio, M., D'Orazio, T., & Scano, A. (2023). Azure Kinect performance evaluation for human motion and upper limb biomechanical analysis. *Heliyon*, 9(11), e21606. <https://doi.org/10.1016/j.heliyon.2023.e21606>
- Büker, L., Quinten, V., Hackbarth, M., Hellmers, S., Diekmann, R., & Hein, A. (2023). How the Processing Mode Influences Azure Kinect Body Tracking Results. *Sensors*, 23(2), 878. <https://doi.org/10.3390/s23020878>
- Cantin-Garside, K. D., Kong, Z., White, S. W., Antezana, L., Kim, S., & Nussbaum, M. A. (2020). Detecting and Classifying Self-injurious Behavior in Autism Spectrum Disorder Using Machine Learning Techniques. *Journal of Autism and Developmental Disorders*, 50(11), 4039-4052. <https://doi.org/10.1007/s10803-020-04463-x>
- Cao, Z., Simon, T., Wei, S.-E., & Sheikh, Y. (2017). Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7291-7299. <https://doi.org/10.1109/CVPR.2017.143>
- Cerfoglio, S., Ferraris, C., Vismara, L., Amprimo, G., Priano, L., Bigoni, M., Galli, M., Mauro, A., & Cimolin, V. (2024). Estimation of gait parameters in healthy and hemiplegic individuals using Azure Kinect: a comparative study with the optoelectronic system. *Frontiers in Bioengineering and Biotechnology*, 12(1449680). <https://doi.org/10.3389/fbioe.2024.1449680>
- Colyer, S. L., Evans, M., Cosker, D. P., & Salo, A. I. T. (2018). 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(24), 24. <https://doi.org/10.1186/s40798-018-0139-y>

- de Leva, P. (1996). Adjustments to Zatsiorsky-Seluyanov's Segment Inertia Parameters. *Journal of Biomechanics*, 29(9), 1223-1230. [https://doi.org/10.1016/0021-9290\(95\)00178-6](https://doi.org/10.1016/0021-9290(95)00178-6)
- Eghbalian, A., Azam, M. M., Holloway, K., Neely, L., & Desai, K. (2025). Applying Computer Vision to Analyze Self-Injurious Behaviors in Children with Autism Spectrum Disorder. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, 11-20. <https://doi.org/10.1109/WACVW65960.2025.00007>
- Foix, S., Alenya, G., & Torras, C. (2011). Lock-in Time-of-Flight (ToF) Cameras: A Survey. *IEEE Sensors Journal*, 11(9), 1917-1926. <https://doi.org/10.1109/JSEN.2010.2101060>
- Gasparrini, S., Cippitelli, E., Spinsante, S., & Gambi, E. (2014). A Depth-Based Fall Detection System Using a Kinect Sensor. *Sensors*, 14(2), 2756-2775. <https://doi.org/10.3390/s140202756>
- Goodwin, M. S., Intille, S. S., Albinali, F., & Velicer, W. F. (2011). Automated Detection of Stereotypical Motor Movements. *Journal of Autism and Developmental Disorders*, 41(6), 770-782. <https://doi.org/10.1007/s10803-010-1102-z>
- Gorordo, I. (2021). pykinect_azure: Python Wrapper for the Azure Kinect SDK [Código abierto (MIT); envuelve k4a.dll y k4abt.dll mediante ctypes].
- Grishchenko, I., Bazarevsky, V., Zanzir, A., Bazavan, E. G., Zanzir, M., Yee, R., Ravendran, K., Zhdanovich, M., Grundmann, M., & Sminchisescu, C. (2022). BlazePose GHUM Holistic: Real-time 3D Human Landmarks and Pose Estimation. <https://doi.org/10.48550/arXiv.2206.11678>
- Großekathöfer, U., Manyakov, N. V., Mihajlović, V., Pandina, G., Skalkin, A., Ness, S., Bangerter, A., & Goodwin, M. S. (2017). Automated Detection of Stereotypical Motor Movements in Autism Spectrum Disorder Using Recurrence Quantification Analysis. *Frontiers in Neuroinformatics*, 11(9). <https://doi.org/10.3389/fninf.2017.00009>

- Guess, T. M., Bliss, R., Hall, J. B., & Kiselica, A. M. (2022). Comparison of Azure Kinect overground gait spatiotemporal parameters to marker based optical motion capture. *Gait & Posture*, 96, 130-136. <https://doi.org/10.1016/j.gaitpost.2022.05.021>
- Hansard, M., Lee, S., Choi, O., & Horaud, R. (2012). *Time-of-Flight Cameras: Principles, Methods and Applications*. Springer. <https://doi.org/10.1007/978-1-4471-4658-2>
- Hasija, V., & Takhounts, E. G. (2022). Deep learning methodology for predicting time history of head angular kinematics from simulated crash videos. *Scientific Reports*, 12(6526). <https://doi.org/10.1038/s41598-022-10480-w>
- Hidalgo, G., Raaj, Y., Idrees, H., Xiang, D., Joo, H., Simon, T., & Sheikh, Y. (2019). Single-Network Whole-Body Pose Estimation. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 6982-6991.
- International Electrotechnical Commission. (2014). *IEC 60825-1: Safety of Laser Products — Part 1: Equipment Classification and Requirements* (3.^a ed.) [Edition 3.0, 2014-05. Defines Class 1 as the safest category, safe under all reasonably foreseeable conditions of use].
- Ishac, K., & Eager, D. (2021). Evaluating Martial Arts Punching Kinematics Using a Vision and Inertial Sensing System. *Sensors*, 21(6), 1948. <https://doi.org/10.3390/s21061948>
- Jocher, G., Qiu, J., & Chaurasia, A. (2024). Ultralytics YOLO11 [Versión 11, modelo de estimación de pose (pose)].
- Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1), 35-45. <https://doi.org/10.1115/1.3662552>
- Kartynnik, Y., Ablavatski, A., Grishchenko, I., & Grundmann, M. (2019). Real-time Facial Surface Geometry from Monocular Video on Mobile GPUs. *arXiv pre-print arXiv:1907.06724*. <https://doi.org/10.48550/arXiv.1907.06724>

- Kolb, A., Barth, E., Koch, R., & Larsen, R. (2010). Time-of-Flight Cameras in Computer Graphics. *Computer Graphics Forum*, 29(1), 141-159. <https://doi.org/10.1111/j.1467-8659.2009.01583.x>
- Lahkar, B. K., Muller, A., Dumas, R., Reveret, L., & Robert, T. (2022). Accuracy of a Markerless Motion Capture System in Estimating Upper Extremity Kinematics During Boxing. *Frontiers in Sports and Active Living*, 4, 939980. <https://doi.org/10.3389/fspor.2022.939980>
- Lam, K. S. L., & Aman, M. G. (2007). The Repetitive Behavior Scale-Revised: Independent Validation in Individuals with Autism Spectrum Disorders. *Journal of Autism and Developmental Disorders*, 37(5), 855-866. <https://doi.org/10.1007/s10803-006-0213-z>
- Li, T., Sahu, A. K., Talwalkar, A., & Smith, V. (2020). Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine*, 37(3), 50-60. <https://doi.org/10.1109/MSP.2020.2975749>
- Li, X. R., & Jilkov, V. P. (2003). Survey of Maneuvering Target Tracking. Part I: Dynamic Models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4), 1333-1364. <https://doi.org/10.1109/TAES.2003.1261132>
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. *European Conference on Computer Vision (ECCV)*, 740-755. https://doi.org/10.1007/978-3-319-10602-1_48
- Liu, Y., Zhu, Z., Chen, X., Deng, C., Ma, X., & Zhao, B. (2022). Biomechanics of the Lead Straight Punch of Different Level Boxers. *Frontiers in Physiology*, 13, 1015154. <https://doi.org/10.3389/fphys.2022.1015154>
- Lokegaonkar, V., Jaisankar, V., Deepika, P., Rao, M., Srikanth, T. K., Mallick, S., & Sodhi, M. (2023). Introducing SSBD+ Dataset with a Convolutional Pipeline for detecting Self-Stimulatory Behaviours in Children using raw videos. <https://doi.org/10.48550/arXiv.2311.15072>

- Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.-L., Yong, M. G., Lee, J., Chang, W.-T., Hua, W., Georg, M., & Grundmann, M. (2019). MediaPipe: A Framework for Building Perception Pipelines. *arXiv preprint arXiv:1906.08172*. <https://doi.org/10.48550/arXiv.1906.08172>
- Mathis, M. W., & Mathis, A. (2020). Deep Learning Tools for the Measurement of Animal Behavior in Neuroscience. *Current Opinion in Neurobiology*, 60, 1-11. <https://doi.org/10.1016/j.conb.2019.10.008>
- Minshawi, N. F., Hurwitz, S., Fodstad, J. C., Biebl, S., Morris, D. H., & McDougle, C. J. (2014). The Association between Self-Injurious Behaviors and Autism Spectrum Disorders. *Psychology Research and Behavior Management*, 7, 125-136. <https://doi.org/10.2147/PRBM.S44635>
- Mohammadian Rad, N., Kia, S. M., Zarbo, C., van Laarhoven, T., Jurman, G., Venuti, P., Marchiori, E., & Furlanello, C. (2018). Deep learning for automatic stereotypical motor movement detection using wearable sensors in autism spectrum disorders. *Signal Processing*, 144, 180-191. <https://doi.org/10.1016/j.sigpro.2017.10.011>
- Muty, N., & Azizul, Z. H. (2016). Detecting arm flapping in children with Autism Spectrum Disorder using human pose estimation and skeletal representation algorithms. *2016 International Conference on Advanced Informatics: Concepts, Theory and Application (ICAICTA)*. <https://doi.org/10.1109/ICAICTA.2016.7803118>
- Negin, F., Ozyer, B., Agahian, S., Kacdioglu, S., & Ozyer, G. T. (2021). Vision-assisted recognition of stereotype behaviors for early diagnosis of Autism Spectrum Disorders. *Neurocomputing*, 446, 145-155. <https://doi.org/10.1016/j.neucom.2021.03.004>
- Plötz, T., Hammerla, N. Y., Rozga, A., Reavis, A., Call, N. A., & Abowd, G. D. (2012). Automatic assessment of problem behavior in individuals with developmental

- disabilities. *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp 2012)*, 391-400. <https://doi.org/10.1145/2370216.2370276>
- Rajagopalan, S. S., Dhall, A., & Goecke, R. (2013). Self-Stimulatory Behaviours in the Wild for Autism Diagnosis. *2013 IEEE International Conference on Computer Vision Workshops (ICCVW)*, 755-761. <https://doi.org/10.1109/ICCVW.2013.103>
- Ren, B., Liu, M., Ding, R., & Liu, H. (2024). A Survey on 3D Skeleton-Based Action Recognition Using Learning Method. *Cyborg and Bionic Systems*, 5(0100). <https://doi.org/10.34133/cbsystems.0100>
- Rodrigues, J. J. P. C., De Rezende Segundo, D. B., Junqueira, H. A., Sabino, M. H., Prince, R. M., Al-Muhtadi, J., & De Albuquerque, V. H. C. (2018). Enabling Technologies for the Internet of Health Things. *IEEE Access*, 6, 13129-13141. <https://doi.org/10.1109/ACCESS.2018.2799858>
- Sadouk, L., Gadi, T., & Essoufi, E. H. (2018). A Novel Deep Learning Approach for Recognizing Stereotypical Motor Movements within and across Subjects on the Autism Spectrum Disorder. *Computational Intelligence and Neuroscience*, 2018(7186762), 7186762. <https://doi.org/10.1155/2018/7186762>
- Sarbolandi, H., Lefloch, D., & Kolb, A. (2015). Kinect Range Sensing: Structured-Light versus Time-of-Flight Kinect. *Computer Vision and Image Understanding*, 139, 1-20. <https://doi.org/10.1016/j.cviu.2015.05.006>
- Savitzky, A., & Golay, M. J. E. (1964). Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry*, 36(8), 1627-1639. <https://doi.org/10.1021/ac60214a047>
- Scharstein, D., & Szeliski, R. (2002). A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, 47(1-3), 7-42. <https://doi.org/10.1023/A:1014573219977>
- Schwarz, S., Preda, M., Baroncini, V., Budagavi, M., Cesar, P., Chou, P. A., Cohen, R. A., Krivokuća, M., Lasserre, S., Li, Z., Llach, J., Mammou, K., Mekuria, R., Nakagami, O., Siahaan, E., Tabatabai, A., Tourapis, A. M., & Zakharchenko, V.

- (2019). Emerging MPEG Standards for Point Cloud Compression. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9(1), 133-148. <https://doi.org/10.1109/JETCAS.2018.2885981>
- Shahroudy, A., Liu, J., Ng, T.-T., & Wang, G. (2016). NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1010-1019. <https://doi.org/10.1109/CVPR.2016.115>
- Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal*, 3(5), 637-646. <https://doi.org/10.1109/JIOT.2016.2579198>
- Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., & Blake, A. (2011). Real-Time Human Pose Recognition in Parts from Single Depth Images. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1297-1304. <https://doi.org/10.1109/CVPR.2011.5995316>
- Stamm, O., & Heimann-Steinert, A. (2020). Accuracy of Monocular Two-Dimensional Pose Estimation Compared With a Reference Standard for Kinematic Multiview Analysis: Validation Study. *JMIR mHealth and uHealth*, 8(12), e19608. <https://doi.org/10.2196/19608>
- Stereolabs. (2023). ZED 2 Stereo Camera — Specifications and Body Tracking SDK [Cámara estereoscópica pasiva con estimación de profundidad neuronal y seguimiento corporal integrado].
- Sullivan, G. J., Ohm, J.-R., Han, W.-J., & Wiegand, T. (2012). Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12), 1649-1668. <https://doi.org/10.1109/TCSVT.2012.2221191>
- Tölgyessy, M., Dekan, M., Chovanec, L., & Hubinský, P. (2021). Evaluation of the Azure Kinect and Its Comparison to Kinect V1 and Kinect V2. *Sensors*, 21(2), 413. <https://doi.org/10.3390/s21020413>

- Topol, E. J. (2019). High-performance medicine: the convergence of human and artificial intelligence. *Nature Medicine*, 25, 44-56. <https://doi.org/10.1038/s41591-018-0300-7>
- Wade, L., Needham, L., McGuigan, P., & Bilzon, J. (2022). Applications and limitations of current markerless motion capture methods for clinical gait biomechanics. *PeerJ*, 10, e12995. <https://doi.org/10.7717/peerj.12995>
- Wagh, V., Scott, M. W., & Kraeutner, S. N. (2024). Quantifying Similarities Between MediaPipe and a Known Standard to Address Issues in Tracking 2D Upper Limb Trajectories: Proof of Concept Study. *JMIR Formative Research*, 8, e56682. <https://doi.org/10.2196/56682>
- Wei, P., Ahmedt-Aristizabal, D., Gammulle, H., Denman, S., & Armin, M. A. (2023). Vision-based activity recognition in children with autism-related behaviors. *Heliyon*, 9(6), e16763. <https://doi.org/10.1016/j.heliyon.2023.e16763>
- Winter, D. A. (2009). *Biomechanics and Motor Control of Human Movement* (4.^a ed.). Wiley.
- Xu, L., Xu, W., Golyanik, V., Habermann, M., Fang, L., & Theobalt, C. (2020). Event-Cap: Monocular 3D Capture of High-Speed Human Motions Using an Event Camera. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4967-4977. <https://doi.org/10.1109/CVPR42600.2020.00502>
- Yang, L., Kang, B., Huang, Z., Zhao, Z., Xu, X., Feng, J., & Zhao, H. (2024). Depth Anything V2. *Advances in Neural Information Processing Systems (NeurIPS)*. <https://doi.org/10.48550/arXiv.2406.09414>
- Yoo, C.-H., Yoo, J.-H., Back, M.-K., Wang, W.-J., & Shin, Y.-G. (2024). A unified framework to stereotyped behavior detection for screening Autism Spectrum Disorder. *Pattern Recognition Letters*, 186, 156-163. <https://doi.org/10.1016/j.patrec.2024.10.001>
- Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., & Grundmann, M. (2020). MediaPipe Hands: On-device Real-time Hand Trac-

king. *arXiv preprint arXiv:2006.10214*. <https://doi.org/10.48550/arXiv.2006.10214>

Zhang, S., Wang, C., Dong, W., & Fan, B. (2022). A Survey on Depth Ambiguity of 3D Human Pose Estimation. *Applied Sciences*, 12(20), 10591. <https://doi.org/10.3390/app122010591>

Zhang, Y., Tian, Y., Wu, P., & Chen, D. (2021). Application of Skeleton Data and Long Short-Term Memory in Action Recognition of Children with Autism Spectrum Disorder. *Sensors*, 21(2), 411. <https://doi.org/10.3390/s21020411>