Reconocimiento automático de gestos de manos para control de interfaces 3D mediante cámara RGB y deep learning

Juan Carlos Mora
juancarlosmora@correo.ugr.es

Alejandro López
alejandrold@correo.ugr.es

Pablo Martín
pablomarsol10@correo.ugr.es

Abstract

La interacción persona-ordenador basada en gestos constituye una alternativa natural a los periféricos tradicionales en entornos interactivos tridimensionales. Sin embargo, el reconocimiento robusto de gestos mediante Visión por Computador plantea importantes desafíos relacionados con la variabilidad natural de la mano, la iluminación y la complejidad del fondo de la escena.

En este trabajo se presenta un sistema de reconocimiento automático de gestos de mano utilizando únicamente una cámara RGB convencional y técnicas de Deep Learning. El sistema propuesto sigue una arquitectura en dos etapas: en primer lugar, se localiza la región de interés correspondiente a la mano y, posteriormente, se clasifica el gesto mediante redes neuronales convolucionales entrenadas mediante transfer learning.

Se realiza un estudio comparativo entre diferentes arquitecturas modernas (ResNet, MobileNet, EfficientNet y ConvNeXt), analizando no solo la precisión de clasificación, sino también métricas de eficiencia computacional en aplicaciones en tiempo real. Además, se evalúa el impacto de la segmentación previa de la mano frente al uso de imágenes completas, demostrando experimentalmente la importancia de la etapa de preprocesamiento de datos para lograr un equilibrio adecuado entre robustez y rendimiento. Los resultados obtenidos muestran que el sistema es capaz de reconocer gestos de forma fiable en escenarios realistas incluso en situaciones complicadas, constituyendo una base sólida para el control de interfaces 3D mediante gestos naturales.

1. Introduction

La interacción persona-ordenador (Human-Computer Interaction, HCI) ha evolucionado en las últimas décadas hacia metodologías cada vez más naturales e intuitivas, donde el usuario puede comunicarse con el sistema sin necesidad de dispositivos físicos tradicionales como ratón, teclado o dispositivos específicos. Así, el reconocimiento de gestos manuales se ha consolidado como una de las líneas de investigación más activas de la actualidad.

No obstante, el reconocimiento automático de gestos mediante visión por computador presenta importantes desafíos cuando se trabaja con cámaras RGB convencionales. La diferencia morfológica de la mano entre distintos usuarios, las oclusiones entre dedos, los cambios de iluminación y la presencia de fondos complejos son algunos de los principales factores que dificultan la extracción de características mediante técnicas clásicas de procesamiento de imagen.

Tradicionalmente, se ha recurrido a dispositivos físicos de profundidad o sensores dedicados, como Leap Motion, que facilitan la segmentación precisa de la mano. Sin embargo, este tipo de soluciones reducen la accesibilidad y limitan su adopción en dispositivos de uso general. Por este motivo, en este trabajo se propone un sistema al alcance de casi cualquier usuario, basado exclusivamente en una cámara RGB estándar, presente en la mayoría de ordenadores personales, y apostando por técnicas de Deep Learning capaces de aprender representaciones robustas directamente a partir de la captación de la cámara.

El objetivo principal de este proyecto es desarrollar un sistema de reconocimiento automático de gestos de mano que permita controlar interfaces 3D de forma natural y eficiente. Para ello, se plantea una arquitectura modular en dos etapas. En la primera, se aborda la localización de la mano en la imagen, con el fin de aislar la región de interés (ROI) y reducir la influencia del fondo. En la segunda, se realiza la clasificación del gesto utilizando redes neuronales convolucionales entrenadas mediante transfer learning a partir de modelos preentrenados en ImageNet.

A lo largo del trabajo se realiza un estudio experimental que compara desde técnicas basadas como Template Matching hasta distintas arquitecturas de Deep Learning bajo un protocolo controlado, analizando tanto métricas de rendimiento (accuracy y F1-score) como aspectos prácticos relacionados con la eficiencia computacional y la viabilidad en tiempo real. Asimismo, se evalúa el impacto de prescindir de la etapa de preprocesamiento de los datos (recorte

previo de la ROI), evidenciando empíricamente la importancia de dicha fase para lograr sistemas robustos en entornos reales. Los resultados obtenidos sientan las bases para el desarrollo de interfaces gestuales accesibles, precisas y escalables utilizando hardware estándar.

2. Background

Tradicionalmente, el problema de la interacción persona-ordenador se ha abordado mediante técnicas de procesamiento de imagen clásicas, basadas en la extracción manual de características geométricas, contornos, descriptores de forma o análisis de movimiento. Sin embargo, estos enfoques presentan limitaciones significativas cuando se enfrentan a entornos no controlados, donde la variabilidad visual es elevada.

Uno de los principales retos en el reconocimiento de gestos es el problema de la *invarianza*. Un mismo gesto puede presentar múltiples variaciones debidas a diferencias anatómicas entre usuarios, cambios en la orientación de la mano, oclusiones entre dedos, distintas condiciones de iluminación y fondos complejos. Las técnicas clásicas suelen requerir segmentaciones precisas o escenarios muy controlados, lo que dificulta su aplicación en sistemas interactivos reales.

El auge del Deep Learning ha supuesto un cambio de paradigma en el campo de la visión por computador. Las redes neuronales convolucionales (CNNs) han demostrado una capacidad superior para aprender representaciones jerárquicas directamente a partir de los datos de entrenamiento, eliminando la necesidad de diseñar manualmente descriptores específicos. Estas redes son capaces de capturar patrones locales de bajo nivel, como bordes o texturas, así como características de alto nivel relacionadas con la forma global del objeto, lo que resulta especialmente adecuado para el reconocimiento de gestos.

En este contexto, el *transfer learning* se ha consolidado como una estrategia fundamental para entrenar modelos cuando el número de muestras disponibles es limitado o cuando se desea reducir el coste computacional del entrenamiento. Esta técnica consiste en reutilizar modelos ya entrenados en grandes conjuntos de datos genéricos, como ImageNet, y adaptarlos a una tarea específica mediante un proceso de ajuste fino (*fine-tuning*). El uso de modelos preentrenados nos permite acelerar la convergencia y mejorar la capacidad de generalización, especialmente en tareas de clasificación visual.

Por otro lado, en sistemas de reconocimiento de gestos en tiempo real resulta de vital importancia reducir la complejidad de la escena de entrada. La localización previa de la mano y la extracción de una región de interés (ROI) permiten eliminar gran parte del ruido visual asociado al fondo, facilitando el aprendizaje del modelo y reduciendo la dimensionalidad de entrada. Este enfoque modular, que separa la detección de la mano de la clasificación del gesto, ha demostrado ser muy efectivo.

Finalmente, la eficiencia computacional es también un aspecto determinante en aplicaciones interactivas. Más allá de la precisión de clasificación, es necesario considerar métricas como la latencia y el número de fotogramas por segundo (FPS), especialmente cuando el sistema debe ejecutarse en dispositivos no muy potentes con recursos limitados. Por ello, debemos de tener en cuenta tanto el rendimiento predictivo como la viabilidad práctica en escenarios de uso real.

3. Related Works

El reconocimiento automático de gestos de mano es una línea de investigación bien establecida y con múltiples soluciones ya propuestas, tanto en el ámbito académico como en aplicaciones prácticas. Todos estos trabajos se agrupan principalmente en enfoques clásicos y enfoques basados en aprendizaje profundo.

Los métodos clásicos de reconocimiento de gestos suelen estar basados en la extracción manual de características geométricas o de forma, combinadas con clasificadores tradicionales como SVM, k-NN o modelos basados en reglas. Estos métodos funcionan en escenarios controlados, pero cuando hay variaciones de iluminación, oclusiones o fondos complejos, se quedan muy limitados.

Con la llegada del Deep Learning, las redes neuronales convolucionales (CNN) se han convertido en el enfoque principal en reconocimiento de gestos. En trabajos como el de Eid y Schwenker (2023), se demuestra que las CNN pueden lograr altos niveles de precisión mediante arquitecturas profundas entrenadas desde cero o mediante transfer learning [5]. Por su parte, Zhan et al. también proponen modelos de CNN con varias capas convolucionales y uso de augmentations para mejorar la generalización, alcanzando precisiones superiores al 0.98 en conjuntos de datos específicos [2].

Además de las CNN, se han explorado arquitecturas que incorporan información temporal o secuencial usando redes recurrentes, como LSTM o GRU, cuando los gestos se capturan en secuencias de vídeo. Por ejemplo, Obaid et al. (2020) combinan CNN con LSTM para reconocer gestos en vídeo, aprovechando la capacidad de las redes recurrentes para modelar dependencias temporales entre frames [6]. Sin embargo, estos enfoques suelen requerir mayor capacidad computacional y más datos etiquetados.

En cuanto a sensores, muchos trabajos han utilizado dispositivos de profundidad o electromiografía para obtener información adicional que facilita la segmentación de la mano. Por ejemplo, Anfredo Atzori, Matteo Cognolato y Henning Müller han propuesto reconocimiento de gestos utilizando datos de electromiografía combinados con redes profundas, mostrando la efectividad de estas señales

biológicas para tareas de clasificación [3]. Sin embargo, estos enfoques dependen de hardware adicional que no está disponible en cámaras RGB estándar, lo que limita su aplicabilidad general.

La disponibilidad de grandes conjuntos de datos también ha hecho que este campo se explote mucho más. El dataset IPN Hand propone un gran volumen de muestras de gestos en vídeo con variación de iluminación y fondo para evaluar modelos en escenarios cercanos al uso real [4]. De forma similar, HaGRID (el utilizado en este proyecto) es uno de los datasets más extensos para gestos estáticos en RGB, con cientos de miles de imágenes con la suficiente variabilidad, lo que lo convierte en uno de los datasets más valiosos para entrenar modelos de Deep Learning [1].

En otros trabajos, se ha explorado el diseño de arquitecturas avanzadas como Vision Transformers (ViT) aplicados a reconocimiento de gestos para capturar relaciones globales en la imagen sin recurrir solo a operaciones convolucionales [7]. Estas soluciones están empezando a mostrar resultados competitivos frente a las CNN tradicionales, especialmente cuando se dispone de grandes cantidades de datos.

Finalmente, un aspecto clave es la importancia de la localización o segmentación de la mano antes de la clasificación. Algunos trabajos demuestran que separar explícitamente la mano del fondo incrementa significativamente la precisión de los modelos y reduce el coste computacional, mejorando la viabilidad de los sistemas en tiempo real.

En este trabajo se parte de estas líneas previas y se propone un análisis comparativo de varias arquitecturas bajo un protocolo experimental controlado, evaluando tanto métricas de clasificación como métricas de eficiencia computacional, así como el impacto de la segmentación previa de la mano respecto al aprendizaje directamente sobre imágenes completas.

4. Methods

En esta sección se describe detalladamente la metodología seguida para el desarrollo del sistema de reconocimiento de gestos de mano. El objetivo principal es evaluar distintas arquitecturas de *Deep Learning* en un escenario realista de interacción humano-computador, analizando tanto la precisión como la viabilidad computacional del sistema para su uso en tiempo real.

4.1. Visión general del pipeline

El sistema propuesto se distribuye en las siguientes fases:

1. Preprocesamiento de los datos y generación del conjunto de entrenamiento.
2. Localización y recorte de la región de interés (ROI) correspondiente a la mano.
3. Clasificación del gesto mediante un modelo de *Deep Learning* entrenado con imágenes RGB.
4. Evaluación del rendimiento del sistema tanto en precisión como en eficiencia computacional.

Este diseño permite analizar de forma aislada el impacto de cada componente del sistema, especialmente el recorte previo de la mano frente al uso de imágenes completas.

4.2. Enfoque clásico: Template Matching

Antes de adoptar un enfoque basado en *Deep Learning*, se realizó un experimento utilizando técnicas clásicas de visión por computador, concretamente *template matching*, con el objetivo de evaluar su viabilidad como método base para el reconocimiento de gestos estáticos.

El método de *template matching* consiste en comparar una imagen de entrada con un conjunto de plantillas previamente definidas, midiendo la similitud mediante métricas como la correlación normalizada. Para cada gesto se construyó un conjunto reducido de plantillas representativas, obtenidas a partir de imágenes preprocesadas y alineadas manualmente.

Las imágenes fueron convertidas a escala de grises y normalizadas en tamaño para reducir la variabilidad introducida por cambios de escala y condiciones de iluminación. La imagen de entrada se comparaba con todas las plantillas disponibles, asignando la clase correspondiente a la plantilla con mayor similitud.

No obstante, los resultados obtenidos evidenciaron limitaciones. El rendimiento del sistema dependía en gran medida de la similitud exacta entre la plantilla y la imagen de entrada, mostrando una sensibilidad a variaciones de rotación, iluminación, fisionomía de la mano y posición relativa respecto a la cámara.

Además, el método no escala adecuadamente cuando se incrementa el número de gestos, y su robustez se ve afectada en entornos desconocidos. Estas limitaciones hacen que el enfoque basado en *template matching* resulte poco adecuado para escenarios de interacción natural en tiempo real. A partir de estas observaciones, se decidió adoptar un enfoque basado en *Deep Learning*, capaz de aprender automáticamente representaciones discriminativas.

4.3. Dataset y preprocesamiento

Para el entrenamiento y evaluación de los modelos se ha utilizado el *dataset* **HaGRID (HAnd Gesture Recognition Image Dataset)**, uno de los conjuntos de datos más extensos y variados para reconocimiento de gestos estáticos con imágenes RGB.

HaGRID contiene cientos de miles de imágenes capturadas en entornos no controlados, con variaciones significativas de fondo, iluminación y morfología de la mano.

Cada imagen incluye anotaciones en formato JSON que especifican las coordenadas de la caja delimitadora (*bounding box*) de la mano.

Con el objetivo de reducir la complejidad del problema y facilitar el entrenamiento, se seleccionaron ocho gestos representativos: *call, fist, like, ok, palm, peace, rock* y *stop*. Para cada clase se extrajo un subconjunto de imágenes.

4.4. Recorte de la región de interés (ROI)

Una de las decisiones metodológicas fundamentales del proyecto es la evaluación del impacto del recorte previo de la región de la mano sobre el rendimiento del sistema.

Utilizando las anotaciones proporcionadas por HaGRID, se realizó un recorte de la región correspondiente a la mano en cada imagen original. A partir de las coordenadas normalizadas de la *bounding box*, se extrajo la región de interés y se añadió un margen de seguridad del 20% (*bounding box*).

Las imágenes resultantes fueron redimensionadas a una resolución fija de 224×224 píxeles, compatible con las arquitecturas preentrenadas utilizadas. Tras este proceso, la etiqueta de cada imagen se asignó mediante la estructura de carpetas, eliminando la necesidad de archivos de anotaciones adicionales.

4.5. Entrenamiento con imágenes completas

Se evaluó el entrenamiento de los modelos utilizando las imágenes completas sin ningún tipo de segmentación previa. En este caso, las imágenes originales fueron redimensionadas directamente a 224×224 píxeles, conservando todo el fondo y el ruido presente en la escena.

Este experimento permite analizar hasta qué punto distintas arquitecturas son capaces de localizar implícitamente la mano dentro de la imagen y aprender características discriminativas sin una etapa explícita de detección. Este enfoque representa el escenario de mayor complejidad debido al ruido (iluminación, distancia y fisionomía).

4.6. Data Augmentation

Con el objetivo de mejorar la capacidad de generalización de los modelos (y considerando necesaria dicha mejora para este tipo de problema), se introdujeron técnicas de *data augmentation* durante el entrenamiento. Estas transformaciones simulan variaciones realistas que pueden aparecer durante el uso del sistema, como rotaciones leves de la muñeca, cambios de iluminación, pequeñas deformaciones de perspectiva o variaciones en la distancia a la cámara.

Las transformaciones se aplicaron únicamente al conjunto de entrenamiento. Esta estrategia reduce el riesgo de sobreajuste (*overfitting*) y mejora la robustez del sistema en escenarios reales frente a variaciones de fondo, iluminación o escala.

En conjunto, esta metodología permite evaluar el impacto de cada decisión de diseño, desde el preprocesamiento hasta la selección de la arquitectura final, sentando las bases para un sistema de reconocimiento de gestos eficiente y apto para su integración en interfaces 3D en tiempo real.

5. Experiments

En esta sección detallamos los experimentos que se han llevado a cabo para validar nuestra propuesta. El objetivo principal ha sido encontrar el equilibrio óptimo entre precisión y velocidad de inferencia, para reducir la latencia a la hora de detectar gestos con la cámara.

Todos los experimentos han sido implementados utilizando el framework FastAI (sobre PyTorch) para la clasificación y librerías específicas Ultralytics YOLO, MediaPipe) para la detección.

5.1. Dataset y Preprocesamiento

Para el entrenamiento y validación de los modelos, hemos utilizado el dataset HaGRID (HAnd Gestures for Human-Computer Interaction). Dado el enorme tamaño del dataset original (cientos de miles de imágenes de alta resolución), hemos realizado un submuestreo para adaptarlo a los recursos computacionales disponibles.

- **Clases seleccionadas:** Hemos filtrado 8 gestos fundamentales para la interacción: *Call, Fist, Like, Ok, Palm, Peace, Rock, Stop*.
- **Volumen de datos:** Se han seleccionado aproximadamente 3.000 imágenes por clase, manteniendo un 20
- **Preprocesamiento:** Todas las imágenes han sido redimensionadas a 224×224 píxeles. Se ha aplicado normalización basada en las estadísticas de ImageNet.

5.2. Experimento 1: Template Matching

Antes de aplicar Deep Learning, evaluamos la capacidad de las técnicas clásicas de visión para resolver el problema de localización de la región de interés sin necesidad de entrenamiento. Implementamos un algoritmo de Template Matching basado en el coeficiente de Correlación Cruzada Normalizada (NCC):

$$R(x,y) = \frac{\sum_{x',y'}(T(x',y') \cdot I(x+x',y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}} \quad (1)$$

Generamos manualmente una ”base de conocimiento” con 24 plantillas (3 variaciones morfológicas por cada una de las 8 clases) y sometimos al sistema a pruebas de detección.

5.2.1 Resultados y Discusión del Experimento 1

Los resultados evidenciaron las severas limitaciones de este enfoque para entornos con tantas posibilidades:

1. **Ambigüedad Morfológica:** El algoritmo presentó falsas correlaciones positivas debido a similitudes en los contornos. Se documentaron errores críticos donde la clase *Rock* fue detectada como *Call* con una confianza del 66%, y *Stop* confundida con *Peace* (71%). Esto se debe a que la NCC prioriza la coincidencia de patrones de píxeles brutos sin entender la geometría del gesto.

2. **Falta de Invarianza a la Escala:** Al utilizar plantillas de tamaño fijo, el sistema falló cuando la distancia del usuario a la cámara variaba, obteniendo puntuaciones de confianza insuficientes.

3. **Ruido de Fondo:** Las plantillas incluían inevitablemente texturas del fondo original (ropa, rostro). El algoritmo buscaba esa combinación específica de "mano + fondo", fallando al cambiar el entorno.

Conclusión del Exp. 1: La visión clásica carece de la invarianza espacial y semántica necesaria. Se confirma la necesidad de utilizar Redes Neuronales Convolucionales (CNNs) capaces de aprender características jerárquicas robustas.

5.3. Experimento 2: Clasificación en Condiciones Ideales (ROI Recortada)

En esta fase, asumimos un escenario ideal donde la mano ya ha sido localizada y recortada. El objetivo es encontrar la arquitectura que ofrezca el mejor balance entre capacidad de discriminación y eficiencia computacional. Para ello, utilizamos técnicas de *Transfer Learning*, partiendo de modelos pre-entrenados en ImageNet y ajustándolos (*Fine-Tuning*) a las 8 clases de nuestro dataset HaGRID.

5.3.1 Fase A: Sin Data Augmentation

En una primera iteración, entrenamos cinco arquitecturas: ResNet18, ResNet152, MobileNetV3 (Large), EfficientNet B0 y ConvNeXt Tiny. El entrenamiento se realizó redimensionando las imágenes a 224×224 píxeles sin aplicar ninguna técnica de data augmentation, sirviendo esto como línea base de rendimiento.

Como se observa en la tabla, los modelos más pesados (ResNet152 y ConvNeXt) alcanzan un rendimiento casi perfecto ($> 99\%$). Sin embargo, ResNet18 se mantiene muy cerca con un 98.57%, demostrando que una arquitectura ligera es suficiente para distinguir estos gestos si la imagen de entrada es limpia.

Modelo	Accuracy	F1-Score	Valid Loss
ResNet152	99.16%	0.991	0.030
ConvNeXt Tiny	99.16%	0.991	0.030
ResNet18	98.57%	0.985	0.050
EfficientNet B0	97.68%	0.976	0.100
MobileNetV3	96.79%	0.967	0.099

Table 1. Resultados de clasificación en condiciones ideales sin regularización.

5.3.2 Fase B: Robustez con Data Augmentation

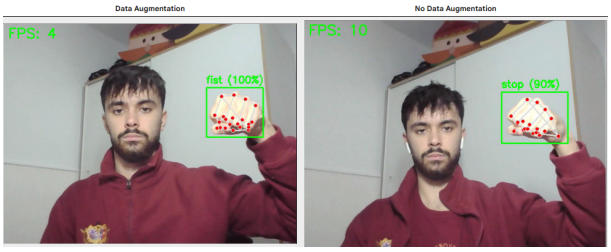
Para dotar al sistema de invarianza frente a la variabilidad del mundo real (cambios de iluminación, rotaciones de muñeca, distancia a la cámara), repetimos el entrenamiento aplicando un Data Augmentation agresivo. Las transformaciones aplicadas durante el entrenamiento fueron:

- Rotación: Aleatoria hasta $\pm 15°$.
- Zoom: Aleatorio hasta $1.1x$.
- Iluminación: Variación de brillo/contraste ($p = 0.75$).
- Perspectiva: Deformación del 20%.
- Flip Horizontal: Para invarianza de mano izq./der.

Modelo	Acc. (Sin)	Acc. (Con Aug)	Dif.
ResNet152	99.16%	99.26%	+0.10%
ResNet18	98.57%	99.14%	+0.57%
ConvNeXt Tiny	99.16%	99.12%	-0.04%
EfficientNet B0	97.68%	97.10%	-0.58%
MobileNetV3	96.79%	95.60%	-1.19%

Table 2. Impacto del Data Augmentation en la precisión.

Mientras que modelos muy ligeros como MobileNetV3 sufrieron una caída de rendimiento (-1.19%) al no tener suficiente capacidad para modelar la variabilidad extra, la ResNet18 mejoró un poco (+0.57%), alcanzando un 99.14% de precisión. Esto confirma que, para ResNet18, el aumento de datos actúa como un regularizador efectivo, permitiéndole generalizar mejor sin saturar su capacidad.



5.3.3 Selección del mejor modelo en este experimento

Aunque los modelos entrenados sin data augmentation alcanzaron métricas de precisión muy buenas, estos resultados resultan insuficientes para garantizar la eficacia del sistema en un entorno real. Dado que nuestra interfaz debe operar con una entrada de video en tiempo real, sujeta a movimientos involuntarios, cambios de distancia y rotaciones de muñeca, descartamos los modelos de la Fase A en favor de los entrenados en la Fase B. La regularización introducida por el data augmentation es necesaria, pues asegura que la red no memorice patrones de píxeles específicos, proporcionando la robustez necesaria para nuestro problema. Para la decisión final, no solo consideramos la precisión, sino el factor crítico de la latencia tanto en entornos acelerados (GPU) como en entornos limitados (CPU). Realizamos una prueba de estrés midiendo la velocidad de inferencia promedio en 1000 imágenes.

Modelo	FPS (GPU)	Lat. GPU (ms)	FPS (CPU)	Lat. CPU (ms)
MobileNetV3	479.53	2.09	**97.45**	**10.26**
ResNet18	**527.10**	**1.90**	72.61	13.77
EfficientNet B0	461.76	2.17	66.93	14.94
ConvNeXt Tiny	380.18	2.63	31.31	31.93
ResNet152	388.09	2.58	13.97	71.59

Table 3. Comparativa de rendimiento en GPU vs CPU.

Dados esta información podemos sacar las siguientes conclusiones :

1. **El cuello de botella de la CPU:** Mientras que en GPU todos los modelos superan holgadamente los 300 FPS, en CPU la historia cambia drásticamente. Modelos pesados como ResNet152 caen a 13 FPS, que sumado al proceso de detectar la mano y recortarla y procesarla en vivo, haría que los FPS bajen aún más, lo que rompería la experiencia de usuario en ordenadores sin tarjeta gráfica dedicada,.

2. **Compromiso Final:** Aunque MobileNetV3 es el líder indiscutible en CPU (97 FPS), su precisión con Data Augmentation cayó al 95.6%. Por contra, ResNet18 mantiene una precisión muy alta(99.14%) y una velocidad en CPU de 72 FPS, más que suficiente para nuestro problema.

Por tanto, seleccionamos ResNet18 como el clasificador final. Es la opción más versátil, ofrece la máxima velocidad posible si el usuario tiene GPU, y una fluidez decente si solo dispone de CPU, sin sacrificar la precisión.

.

5.4. Experimento 3: Impacto del Ruido de Fondo

Para verificar si era posible prescindir de la etapa de detección, entrenamos los mismos modelos utilizando las imágenes completas capturadas por la cámara(sin recortar la mano), donde el gesto ocupa solo una pequeña fracción de la escena.

Modelo	Acc. (Recorte)	Acc. (Completa)
ConvNeXt Tiny	99.12%	96.25% (↓ 3%)
ResNet152	99.26%	96.12% (↓ 3%)
ResNet18	9.14%	83.21% (↓ 16%)
MobileNetV3	95.60%	63.46% (↓ 32%)
EfficientNet B0	97.10%	48.06% (↓ 49%)

Table 4. Degradación del rendimiento al introducir fondo complejo.

Los resultados muestran que arquitecturas modernas y pesadas como ConvNeXt lograron filtrar el fondo, manteniendo un 96% de precisión. Los modelos ligeros, en cambio, colapsaron. ResNet18 cayó al 83% y MobileNet al 63%, volviéndose inútiles para la aplicación. Esto valida nuestra arquitectura final: dado que estamos limitados a CPU, no podemos usar ConvNeXt (demasiado lento, 13 FPS). Debemos usar ResNet18 por su velocidad, pero es obligatorio incluir una etapa previa de detección para ”limpiar” la imagen antes de clasificarla.

Como se puede observar en esta imagen comparativa, es necesario recortar la región de interés antes de entrenar el modelo, ya que si no aprendería patrones básicos como el que podemos observar aquí: Como en todas las imágenes del dataset del gesto ”call” aparece la mano pegada a la cara de la persona, si se aleja la mano de la cara se detectará otro gesto, lo que es incorrecto.

5.5. Experimento 4: Detector Propio (YOLOv8)

Finalmente, para sustituir la dependencia de librerías externas cerradas como MediaPipe, entrenamos un detector de objetos propio basado en YOLOv8 Nano. Transformamos el problema de detección de 8 clases en un problema de detección de clase única, ya que solo nos interesa que sustituya a mediapipe como detector de la mano.

5.5.1 Preparación y Transformación del Dataset

Se utilizó el dataset **HaGRID** como fuente principal, seleccionando las 8 clases representativas. El protocolo de preparación incluyó:

- **Balanceo de Datos:** Se extrajeron 3000 imágenes para entrenamiento y 500 para validación por cada clase.

- **Normalización YOLO:** Las coordenadas originales de HaGRID fueron convertidas al formato requerido por el modelo: $[x_{centro}, y_{centro}, ancho, alto]$ normalizados entre 0 y 1.

- **Estructura de Directorios:** Se organizó el dataset en carpetas segregadas de `images` y `labels` para entrenamiento y validación, junto con un archivo de configuración `data.yaml` que define la ruta del dataset y la clase (*hand*).

5.5.2 Entrenamiento y Resultados

El modelo fue entrenado utilizando la librería `ultralytics` durante 15 épocas . Los registros de validación muestran un rendimiento excepcional:

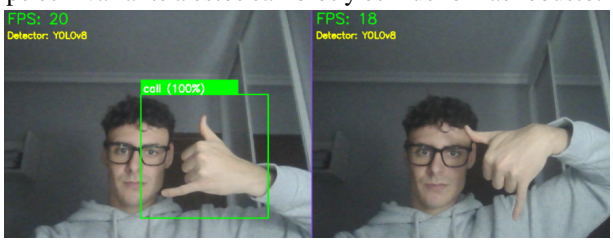
- **Precisión Media (mAP50):** El detector alcanzó un valor de **0.995**, lo que garantiza una localización casi perfecta de la mano antes de pasar el recorte al clasificador.

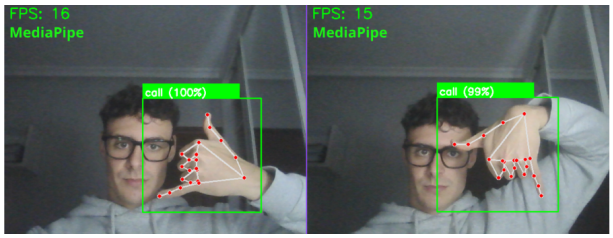
- **Benchmarking de Latencia:** En pruebas sobre CPU, YOLOv8 demostró una latencia de **21.69 ms** por frame, frente a los **26.90 ms** de MediaPipe, representando una mejora en el rendimiento computacional.

5.5.3 Conclusiones del Módulo

Tras la comparativa, se determinó que YOLOv8 es superior en términos de personalización y velocidad teórica. Sin embargo, MediaPipe mantiene una mayor robustez ante rotaciones gestuales bruscas debido a su modelo de refinamiento de 21 puntos. A pesar de que la alta precisión de localización obtenida con YOLOv8 permite una segmentación limpia de la Región de Interés (ROI), facilitando que el clasificador *ResNet18* reciba entradas estables y centradas, se prioriza robustez frente a latencia, por lo que el detector escogido para problemas HCI es **MediaPipe**.

En la imagen de abajo se puede observar que cuando el gesto sufre una traslación/rotación, el detector propio sufre y directamente no detecta la ROI, mientras que el MediaPipe es invariante a estos cambios y es mucho más robusto.





6. Conclusions

Una vez desarrollado y validado un sistema de reconocimiento automático de gestos de mano orientado al control de interfaces 3D, utilizando exclusivamente *hardware* estándar y técnicas de *Deep Learning*, se presentan las principales conclusiones derivadas de la experimentación:

- **Eficacia del *Deep Learning*:** Se ha demostrado que las técnicas de *Deep Learning* superan ampliamente a los métodos clásicos como el *Template Matching*, el cual carece de la invarianza semántica necesaria para operar en entornos no controlados con variaciones de escala, iluminación y rotación.

- **Arquitectura en Dos Etapas:** La investigación confirma que un *pipeline* modular que separa la localización (ROI) de la clasificación es esencial para el uso de modelos ligeros. Mientras que modelos pesados pueden filtrar el fondo, arquitecturas eficientes como **ResNet18** requieren una etapa previa de segmentación para evitar la degradación del rendimiento ante ruido.

- **Equilibrio Precisión-Latencia:** Tras comparar múltiples arquitecturas, se concluye que **ResNet18** ofrece el mejor equilibrio para aplicaciones HCI en tiempo real, logrando una precisión del 99.14% con una latencia en CPU de tan solo 13.77 ms, superando a modelos más pesados o excesivamente livianos.

- **Impacto de la Regularización:** El uso de *Data Augmentation* ha resultado ser un factor crítico para dotar al sistema de robustez frente a rotaciones axiales y cambios lumínicos, actuando como un regularizador que mejora la capacidad de generalización del clasificador.

- **Selección del Detector:** Aunque el detector propio basado en **YOLOv8 Nano** demostró una precisión (mAP50=0.995) y velocidad (21.69 ms) superiores en condiciones ideales, la robustez de **MediaPipe** ante rotaciones gestuales bruscas lo consolida como la opción preferente.

6.1. Trabajos Futuros

A partir de los resultados obtenidos, se plantean las siguientes líneas de mejora y expansión del sistema:

1. **Reconocimiento Dinámico:** Integrar capas de memoria temporal, como redes *LSTM* o *GRU*, para permitir el reconocimiento de gestos dinámicos en secuencias de vídeo.

2. **Optimización del Detector Propio:** Mejorar el entrenamiento del modelo YOLOv8 mediante la inclusión de aumentos específicos de rotación y oclusión para alcanzar la estabilidad de rastreo de MediaPipe manteniendo su ventaja en latencia, obviando el *Data Augmentation* interno que contiene.

3. **Exploración de Modelos Transformer:** Evaluar el rendimiento de arquitecturas *Vision Transformer* (ViT) para capturar relaciones globales en la fisionomía de la mano que las redes convolucionales podrían omitir.

4. **Ampliación del Diccionario Gestual:** Incrementar el número de clases y la complejidad de los gestos permitidos para habilitar un control más granular en entornos 3D complejos.

References

[1] Hagrid (hand gesture recognition image dataset). 2022. Dataset de gestos ampliamente usado para entrenamiento de Deep Learning. 3

[2] Hand gesture recognition using deep cnns and data augmentation. 2024. Adapted from available literature summary; example techniques used. 2

[3] Manfredo Atzori, Matteo Cognolato, and Henning Müller. Deep learning-based hand gesture recognition using electromyography signals. 2019. 3

[4] Gibran Benitez-Garcia et al. Ipn hand: A video dataset and benchmark for real-time continuous hand gesture recognition. 2020. 3

[5] Ahmed Eid and Friedhelm Schwenker. Visual static hand gesture recognition using convolutional neural network. *Algorithms*, 2023. 2

[6] Falah Obaid et al. Hand gesture recognition in video sequences using deep convolutional and recurrent neural networks. In *Applied Computer Systems*, 2020. 2

[7] Vasyl Teslyuk et al. Investigating hand gesture recognition models: 2d cnns vs visual transformers. 2025. Comparative study 2D-CNN vs ViT for gesture recognition. 3