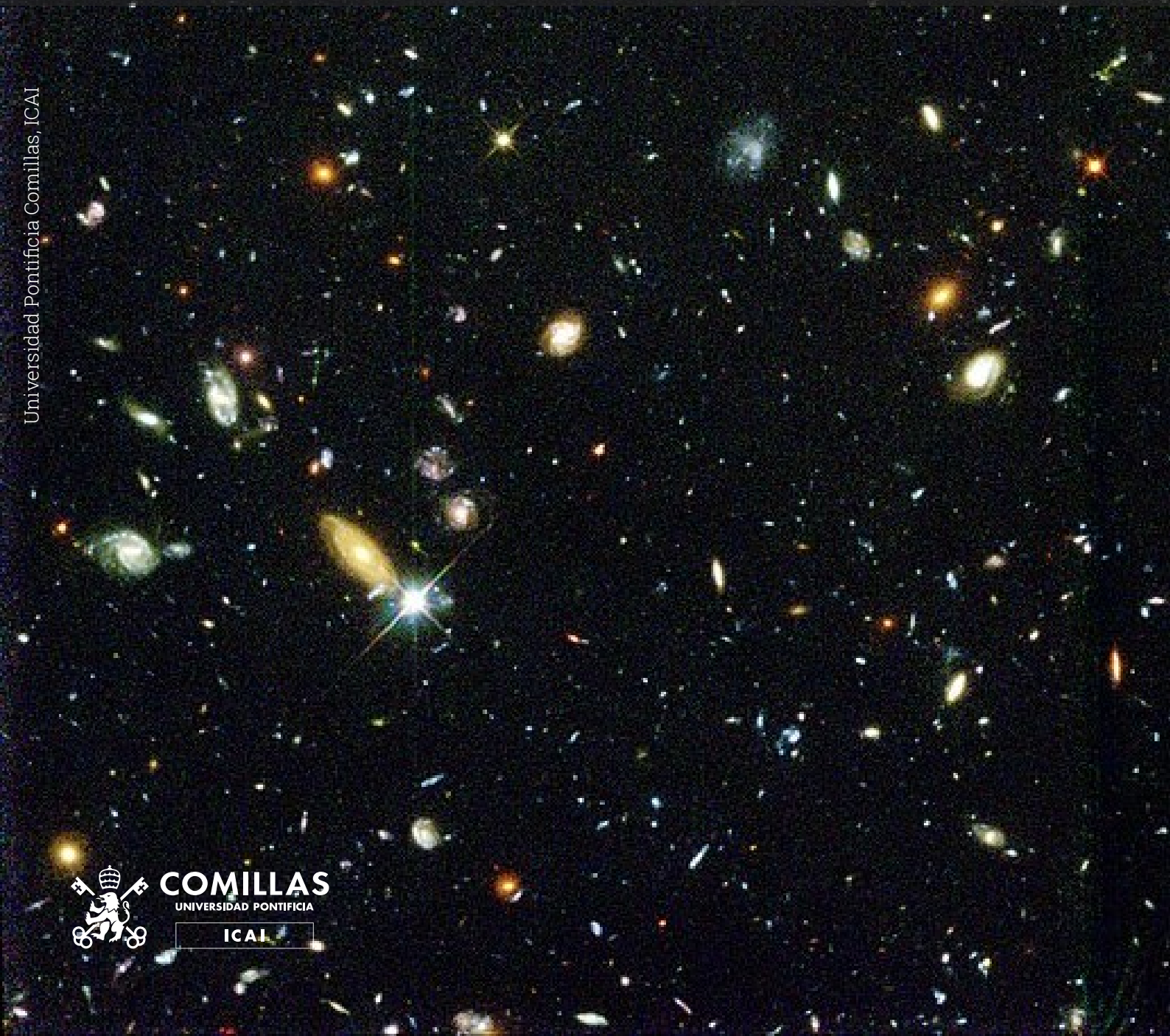
Proyecto final

FitVision
Plataforma interactiva de ejercicio

Visión por ordenador I

Ana Ling Gil González
Leyre Fontaneda Fernández



Universidad Pontificia Comillas, ICAI

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

Proyecto final

FitVision
Plataforma interactiva de ejercicio

by

Ana Ling Gil González
Leyre Fontaneda Fernández

3º Ingeniería Matemática

Curso 2025/26

Profesor:	**Email**
Lionel Güitta	lglopez@icai.comillas.edu
Daniel Pinilla	dpinilla@icai.comillas.edu
Luis Arias	learias@icai.comillas.edu
José María Bengochea	jmbengochea@icai.comillas.edu
Mario Triviño	mtrivino@comillas.edu
Erik Velasco	evelasco@icai.comillas.edu

Cover: Hubble Deep Field: Thousands of galaxies captured in a single view, showcasing the vastness of the universe.

Style: TU Delft Report Style, with modifications by Daan Zwaneveld

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

Contents

1

Introduction

El proyecto final tiene como objetivo desarrollar un sistema de visión por ordenador centrado en la detección y registro de ejercicios físicos mediante el uso de una cámara. La aplicación principal consiste en una especie de plataforma de "retos" de ejercicio, en la que los usuarios deben realizar diferentes ejercicios frente a la cámara, los cuales son detectados y registrados automáticamente.

Para garantizar la seguridad y el control de acceso al sistema, se implementará un bloque de seguridad basado en un código de color, que se reconocerá mostrando diferentes prendas de ropa ante la cámara. Esta secuencia de colores actuará como un mecanismo de identificación, permitiendo el acceso únicamente a usuarios autorizados.

El objetivo principal es crear una experiencia interactiva que monitorice ejercicios en forma de reto, para crear un ambiente de juego y competitividad. Esto abre la posibilidad de aplicaciones variadas, como un modelo de entrenamiento personal completamente automatizado o retos deportivos que puedan compartirse entre amigos y difundirse en redes sociales.

Aquí se puede visitar el GitHub del proyecto.

Materiales

Hardware

El sistema utiliza como entrada de datos la cámara integrada del portátil en el que se ejecuta el proyecto. Esta cámara nos permite capturar vídeo en tiempo real para la detección de patrones de seguridad y el seguimiento de los ejercicios físicos.

Software

El proyecto se desarrolla en Python, utilizando principalmente las siguientes librerías:

1. OpenCV: para todas las operaciones relacionadas con la visión por ordenador clásica, como captura de vídeo, preprocesado de imagen y detección de colores.
2. MediaPipe: para la detección y seguimiento de la postura del usuario, permitiendo reconocer los diferentes ejercicios realizados.

El resto de librerías utilizadas se encuentran detalladas en el archivo requirements.txt del repositorio, además de las librerias ya instaladas en el entorno proporcionado al comenzar los laboratorios, fue necesario instalar las librerias Mediapipe y Pygame.

Uso de IA

Se ha utilizado ChatGPT como asistente para la creación de comentarios explicativos en el código y la mejora de la documentación, con el fin de facilitar la comprensión y mantenimiento del proyecto. Además, se ha empleado para el uso de botones y la lógica de eventos con la librería pygame.

2

Metodología

En esta sección se describe la calibración de la cámara, la arquitectura del sistema, los algoritmos de detección de ejercicios y colores, y la secuencia completa de procesamiento de imagen.

2.1. Calibración de la cámara

La calibración tiene como objetivo estimar los parámetros de la cámara y corregir sus distorsiones, asegurando una correcta relación entre el espacio tridimensional y su proyección en la imagen 2D. En este proyecto, la calibración se realizó utilizando 20 imágenes de un tablero de ajedrez, empleando OpenCV para la detección automática de esquinas y el cálculo de los parámetros mediante correspondencias 3D-2D. Todo el proceso se implementó en el notebook `camera_calibration.ipynb`, obteniéndose la matriz intrínseca, los coeficientes de distorsión y los parámetros extrínsecos, con un error RMS de 1.34 píxeles, considerado adecuado para aplicaciones de visión por ordenador.

Matriz intrínseca

La matriz intrínseca de la cámara resultante es:

$$K = \begin{bmatrix} 968.79 & 0 & 630.74 \\ 0 & 966.90 & 376.66 \\ 0 & 0 & 1 \end{bmatrix}$$

Coeficientes de distorsión

Los coeficientes de distorsión calculados son:

$$\text{Distorsión} = \begin{bmatrix} 0.0491 & 0.2203 & -0.0042 & -0.0003 & -1.6333 \end{bmatrix}$$

2.2. Diagrama de bloques del sistema

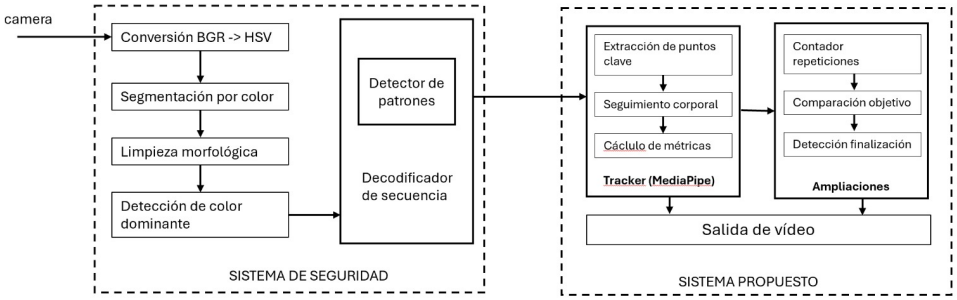


Figure 2.1: Diagrama de bloques del sistema propuesto

2.3. Sistema de seguridad

El bloque de seguridad del sistema permite controlar el acceso mediante la detección de una secuencia de ropa de diferentes colores mostrada frente a la cámara. Cada color funciona como un elemento de la contraseña visual, y la combinación correcta permite continuar con la ejecución del sistema. El patrón elegido puede ser modificado por el usuario, quien deberá capturar imágenes de los objetos que desea utilizar en el código de seguridad y seleccionar los rangos de color HSV adecuados para cada objeto.

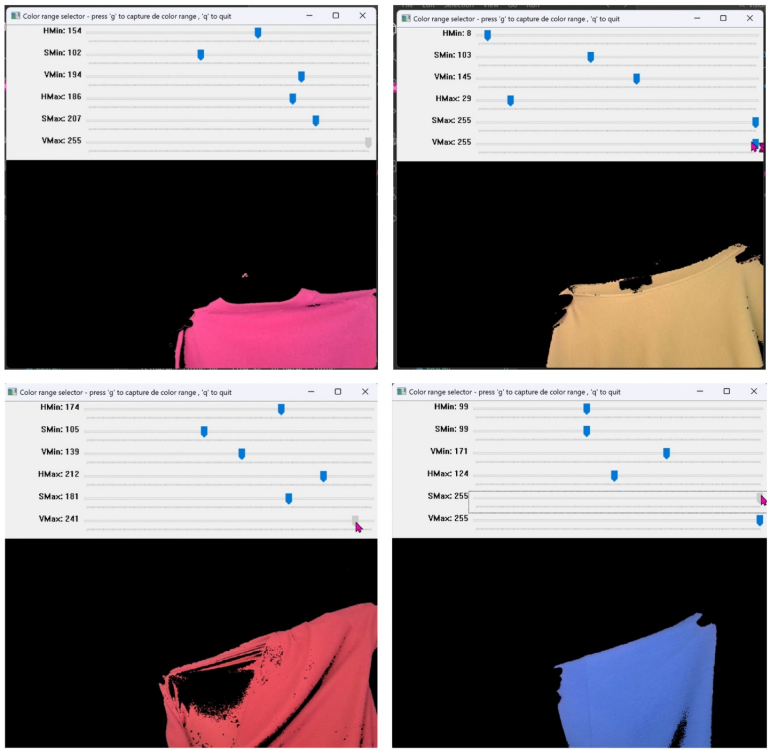


Figure 2.2: Selección de rangos de color

Detección de patrones

1. Conversión a HSV: Cada frame capturado se transforma de BGR a HSV para separar la información de color de la intensidad.
2. Segmentación por color: Para cada color definido como parte del patrón, se genera una máscara binaria.

3. Filtrado morfológico: Se aplican operaciones morfológicas de apertura y cerrado para eliminar ruido (eliminan pixeles aislados y rellenan zonas de la prenda que no se esten detectando correctamente respectivamente).
4. Detección por área: Se calcula el número de píxeles detectados por color y el de mayor superficie detectada se compara con un umbral. Si se supera, el color se considera presente.

Extracción de información

Una vez detectado un color, el sistema lo integra en la secuencia introducida por el usuario, almacenando el orden en que aparecen los elementos visuales. La validación se realiza comparando esta secuencia con la secuencia correcta previamente definida.

1. La primera detección inicia un temporizador de 30s; si se supera sin completar la secuencia, esta se reinicia.
2. Cuando la secuencia completa coincide con la esperada, se permite el acceso; si no, se reinicia y se puede reintentar.
3. Detecciones consecutivas del mismo color se ignoran para evitar duplicados.

El sistema está pensado para ser ligero, interactivo y ejecutable en tiempo real, con tolerancia a pequeñas variaciones en la escena y a errores temporales.

2.4. Sistema propuesto: tracker, ampliaciones y salida de vídeo

Una vez que se muestra la secuencia correcta como código de verificación, accedemos al sistema propuesto. Éste consiste en un programa que detecta el ejericico físico realizado a través de la cámara. Primero, podemos personalizar la rutina de ejercicio seleccionando el número de repeticiones de los siguientes ejericios: sentadilla, flexión y jumping jacks (saltos). Para ello, se ha diseñado una ventana con pygame que permite ver de manera visual el diseño de la rutina. Una vez se han establecido los objetivos, se procede a la realización de los ejericios.

Tracker

Para la detección de movimiento empleamos MediaPipe, que es un framework de visión por ordenador y machine learning desarrollado por Google. Consideramos MediaPipe en este proyecto porque incluye sistemas de tracking ya desarrollados específicamente para detectar movimiento y poses. Nos permitió detectar las poses correctamente y nos facilitó el cálculo de ángulos corporales. El flujo empleado fue el siguiente:

1. OpenCV captura cada frame de la cámara, y se envía a MediaPipe Pose.
2. MediaPipe devuelve los landmarks corporales, a partir de los cuales se calculan ángulos entre articulaciones.
3. Los ángulos se comparan con umbrales definidos para detectar cada ejercicio, y cuando se completa un repetición válida, se incrementa el contador.

Ampliaciones

Una vez que el sistema detecta y valida correctamente un movimiento con MediaPipe, se inicia el control del entrenamiento. Con cada ejecución válida del ejercicio , el contador de repeticiones se actualiza, incrementando en una unidad el valor asociado al ejercicio realizado. Luego, el programa realiza una comparación continua entre el número de repeticiones realizadas y las establecidas anteriormente. Cuando el contador llega al objetivo, el sistema llega a la fase de detección de finalización. Cuando se llega se considera que el ejercicio ha sido completado correctamente.

Salida de vídeo

En la ventana que muestra lo que captura la cámara, en la esquina superior izquierda se muestra el estado de la rutina. Para cada tipo de ejercicio, aparece el número de repeticiones realizadas, y el número de repeticiones objetivo. A medida que se realizan los diferentes ejercicios, el contador del ejericico correspondiente se actualiza automáticamente. Finalmente cuando se completa la rutina establecida, aparece un mensaje, y se muestra la persona detectada por MediaPipe en un fondo rosa.

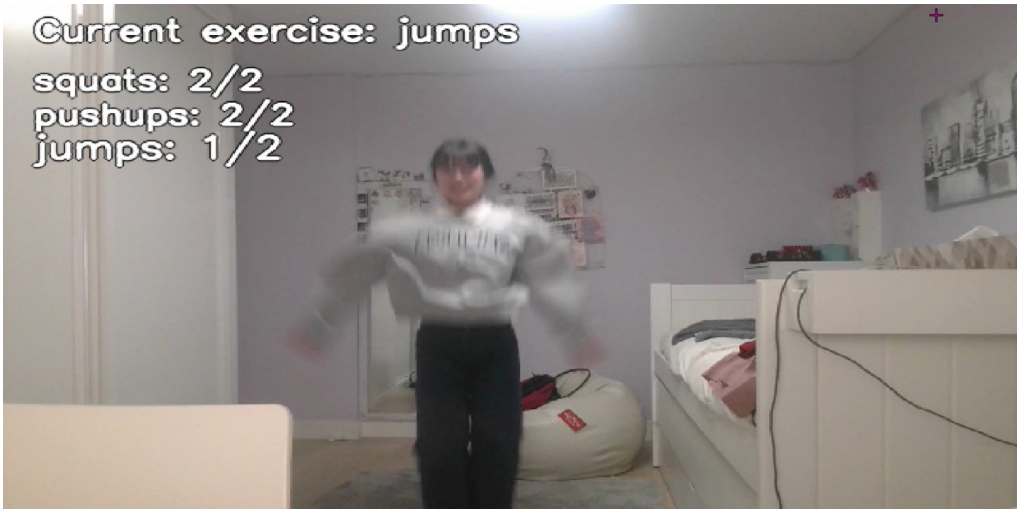


Figure 2.3: Tracker de movimiento en ejecución

3

Conclusion

3.1. Resultados

El sistema desarrollado detecta correctamente en tiempo real el patrón de seguridad basado en color. Esta detección sirve como base para la integración de las funcionalidades de seguimiento de ejercicio físico mediante visión por ordenador. Gracias al uso combinado de OpenCV y MediaPipe, el sistema puede analizar el movimiento corporal del usuario, identificar posturas y contar repeticiones de distintos ejercicios automáticamente. Los resultados obtenidos confirman que la aplicación responde de manera fluida, proporciona información visual clara al usuario y cumple con los objetivos planteados.

3.2. Futuros desarrollos

Como líneas de mejora futura, se plantea ampliar el número de ejercicios que el sistema es capaz de detectar, incorporando movimientos más difíciles, complicados y combinados. También se podría aumentar la robustez del sistema frente a condiciones de iluminación variables. Además, la inclusión de feedback por audio permitiría mejorar significativamente la experiencia de usuario. Otras posibles ampliaciones incluyen la personalización de rutinas, el almacenamiento histórico del rendimiento del usuario y la integración con interfaces web o móviles para una visualización más completa y detallada de los resultados.