
Universidad De Alicante

ESCUELA POLITÉCNICA SUPERIOR

Just Dance Now



*Grado en Ingeniería Robótica
Visión por computador*

Autores: Alejandro Climent Peñalver, Laura Navarro Perez, David Moreno Diaz & Erica
Tébar Máximo
23 de diciembre de 2022

Índice

1. Introducción	1
2. Objetivos	1
3. Funcionamiento del juego	1
4. Estado del arte	1
4.1. Captación y matching de movimientos	2
4.2. Seguimiento de los movimientos	9
4.3. Interfaz gráfica	11
5. Experimentación	13
5.1. Testeo del programa principal	15
6. Conclusiones	17

1. Introducción

Para este proyecto nos hemos inspirado en el famoso videojuego de consola JustDance. La idea principal es conseguir que el usuario, tras elegir una de las canciones propuestas y un personaje, pueda desafiar a sí mismo bailando, para ello se compararán los movimientos de este usuario con el baile original para su posterior evaluación.

2. Objetivos

El objetivo de este proyecto es recrear el funcionamiento del juego Just Dance dándole nuestro propio toque personal e incorporando otras funcionalidades que nos parecieron interesantes.

Por otro lado, se desea divertir al usuario retándole a los diferentes desafíos propuestos.

3. Funcionamiento del juego

En este juego, el usuario irá recibiendo una puntuación en función de la correcta posición de los movimientos y la velocidad con la que se realicen.

En la pantalla se mostrará un modelo a seguir, el jugador tendrá que imitarlo.

La dinámica del juego consiste en copiar el movimiento que realiza el avatar mostrado por pantalla, cuanto más rápido sea el jugador imitando movimientos de forma correcta más puntuación obtendrá.

El programa finalizará cuando el jugador haya hecho todos los movimientos, hayan pasado 3 minutos (como un watchdog) o se interrumpa el programa presionando la tecla de 'esc', se mostrará una estadística con la puntuación por cada posición en base al tiempo óptimo y al que haya tardado en realizar una posición, esta puntuación será nula si ha pasado el tiempo máximo de 3 minutos. También se dispone de una interfaz gráfica para mejorar la estética de juego, como si de un videojuego real se tratara, esta se detallará en un apartado posterior.

4. Estado del arte

Nuestro just dance consistirá en parodiar los movimientos del famoso baile de Rasputin. Para minimizar el tamaño del problema, solo se han implementado los primeros 83 segundos del baile, que constan de 17 movimiento siguiendo una secuencia predefinida.

Además, solo se tendrán en cuenta los movimientos de las extremidades superiores, es decir, los hombros, codos y muñecas, esta restricción no solo se ha propuesto para simplificar el problema, también porque el personaje del baile real apenas mueve las otras partes del cuerpo, por lo que el movimiento se considerará correcto tanto en la imagen 12a como en la 12b.



Imagen 1: Posición 6

En cuanto a la implementación de la solución a este problema, se puede dividir en varias etapas:

- Captación y matching de movimientos
 - Seguimiento de los movimientos
 - Interfaz gráfica

4.1. Captación y matching de movimientos

Para la detección y tracking de las partes del cuerpo se emplea MediaPipe, se trata de una librería de machine learning ya entrenada que es capaz de detectar y traquear hasta 543 puntos del cuerpo. En una fase inicial nos encargamos de testear esta librería, para ello se crea un programa que obtiene imágenes de un vídeo cada determinados frames (para que las imágenes no sean demasiado similares) y se aplica MediaPipe. Usamos 8 vídeos con diversas características:

- Variando número de personas.

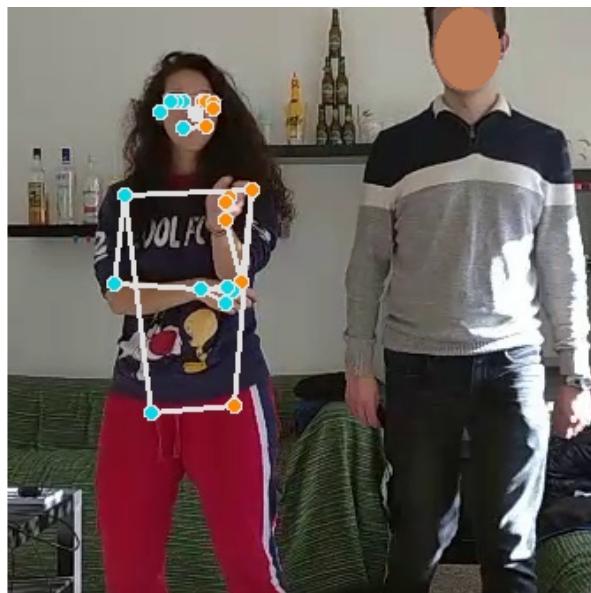
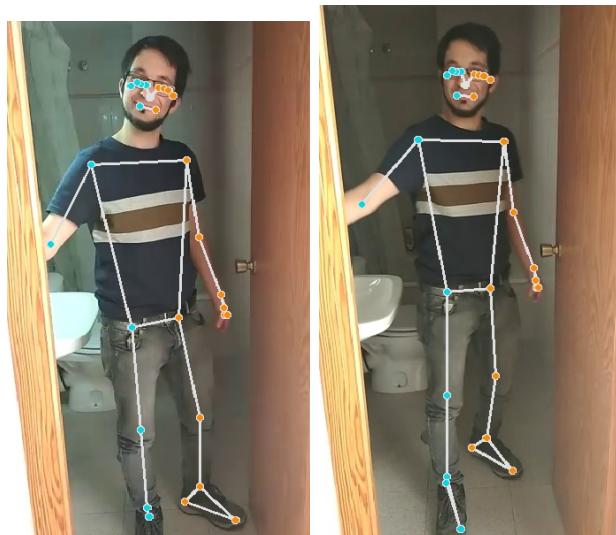


Imagen 2: Varias personas en una imagen

- Distinta luminosidad.



(a) Mayor luz

(b) Menor luz

- Distinta nitidez.



Imagen 4: Borroso

- Múltiples proporciones físicas (altura, distancia hombro-muñeca...).

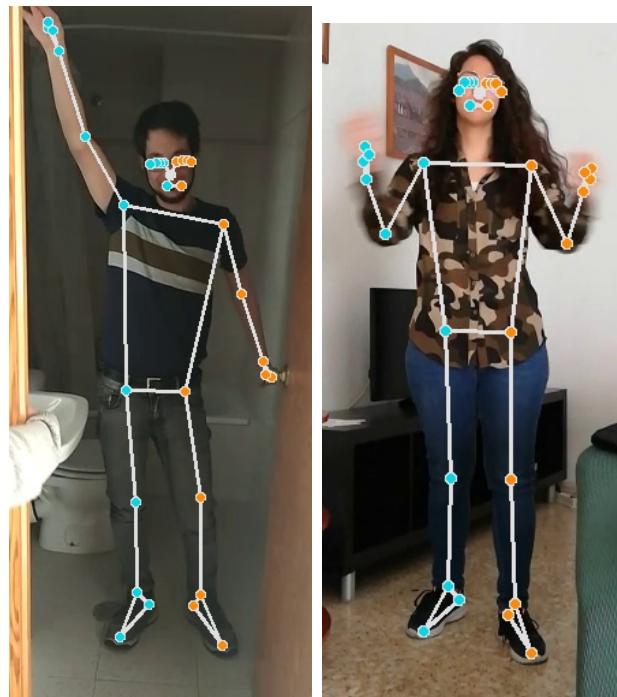


Imagen 5: Distintas personas

- Oclusiones.



Imagen 6: Borroso

- Fondos del mismo color que la ropa del jugador.



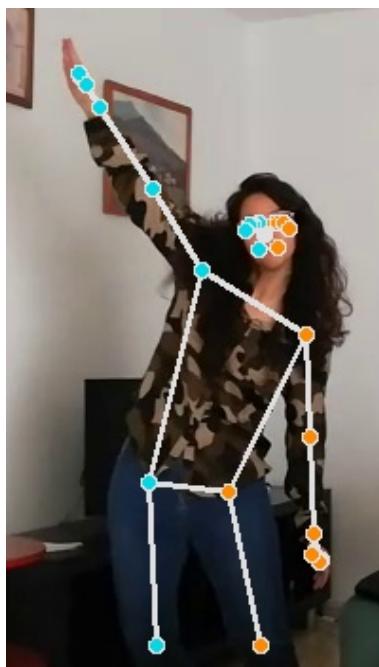
Imagen 7: Fondos con varios colores

- Rotados.



Imagen 8: boca abajo

- Distintas distancias.



(a) Más cerca



(b) Más lejos

- Trasnformaciones físicas (de espalda, de lado...).

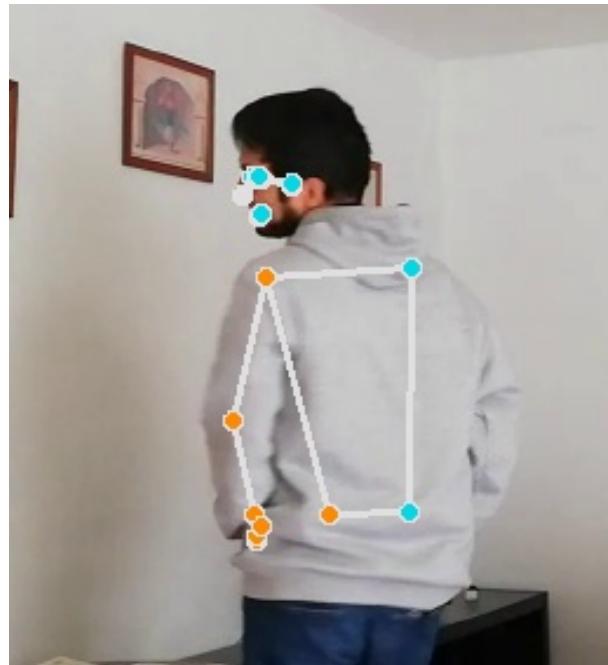


Imagen 10: De espaldas

Obtuvimos un porcentaje de acierto del 83,84 %, pero nos dimos cuenta que la mayoría de los fallos ocurrían en determinadas situaciones:

- Oclusión total de un punto.
- Aparición de varias personas.
- Fondos del mismo color que la ropa del jugador.

Empleando únicamente imágenes 'correctas' se obtiene un porcentaje de aciertos del 93,75 %. Este porcentaje es similar al que aparece en la página oficial de la librería:

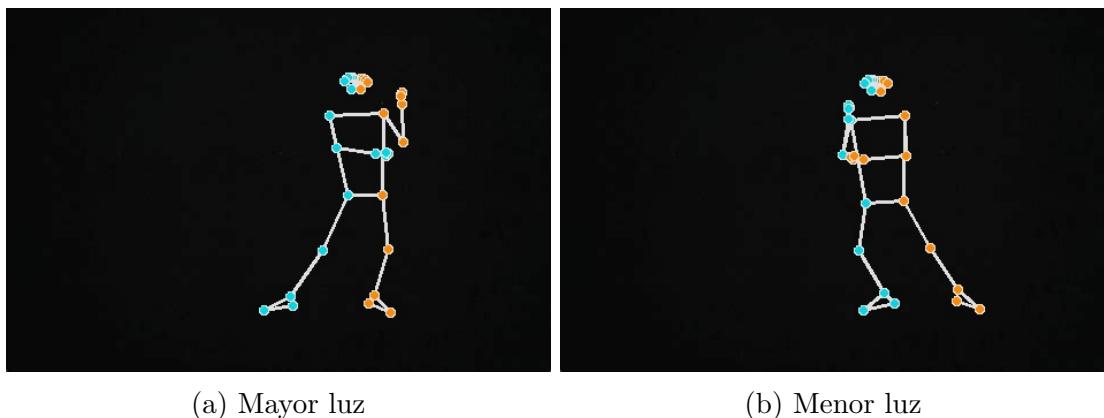
Method	Yoga mAP	Yoga PCK@.2	Dance mAP	Dance PCK@.2
BlazePose GHUM Heavy	68.1	96.4	73.0	97.2
BlazePose GHUM Full	62.6	95.5	67.4	96.3
BlazePose GHUM Lite	45.0	90.2	53.6	92.5

Imagen 11: Tabla comparativa mostrando el porcentaje de aciertos

NOTA: Se emplea el método 'BlazePose GHUM 3D lite'.

Este porcentaje de error de las imágenes correctas se tendrá en cuenta a la hora de calcular la puntuación, y se supondrá que todos los frames capturados durante el juego son 'correctos', es decir, sin occlusiones, ni fondos del mismo color que la ropa del jugador y con un solo jugador.

Una vez testeada la librería se procede a capturar los 17 movimientos que contiene el baile mediante fotografías (que serán empleadas como modelo), algunos de estos movimientos simplemente se han dado la vuelta, puesto que las posiciones son idénticas pero con efecto espejo:



(a) Mayor luz

(b) Menor luz

Para tener en cuenta que las características físicas del jugador no son idénticas a las del modelo (brazos más largos, mayor separación entre hombros...), en vez de comparar la distancia entre los puntos del modelo y los del jugador, dividimos las imágenes en 4 tipos:

- Dos manos arriba
- Dos manos abajo
- Mano derecha arriba; mano izquierda abajo
- Mano izquierda arriba; mano derecha abajo

Entendiéndose por arriba si todos los puntos de los brazos son superiores al punto medio de los hombros y viceversa. Una vez hecha la subdivisión se crea un sistema de cuadrículas con tres sistemas de referencia (codo derecho, punto intermedio entre hombros y codo izquierdo) y 7 puntos, (muñecas, codos, hombro y su punto intermedio), en función de la cuadrícula y los puntos se pueden determinar el movimiento realizado:

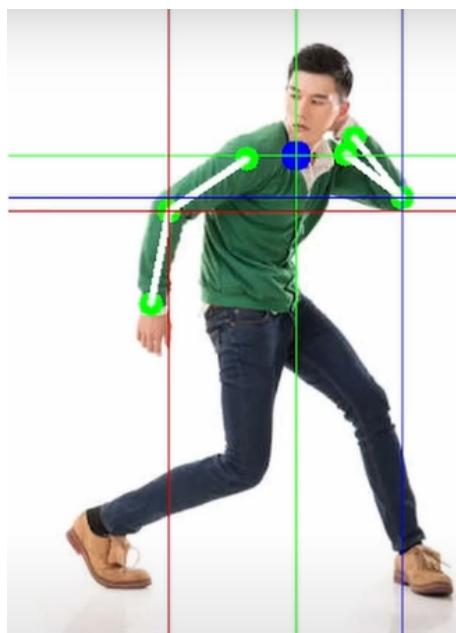


Imagen 13: Detección por cuadrícula

La asignación de un movimiento en función de sus puntos en la cuadrícula se ha implementado en base a las imágenes 'modelo' y se ha [comprobado que el método es válido](#) para cualquier tipo de persona y situación independiente del modelo, con un determinado error.

Así somos capaces de detectar y etiquetar movimientos.

4.2. Seguimiento de los movimientos

Llegados a este punto, en el que somos capaces de detectar a qué movimiento de baile se corresponde una posición, es necesario que el jugador vea en pantalla el tipo de movimiento que debe hacer.

Para ello, se ha usado un programa de modelado y animación 3D (Blender v3.4) y un avatar obtenido mediante 'Ready player me', se trata de una plataforma basada en IA de avatares multijuego para el meta-universo, se ha elegido esta plataforma porque proporciona avatares bastante similares a las personas reales y además riggeados (dotados de 'huesos' para poder moverse). Los pasos realizados son los siguientes:

- 1.- Registro y verificación de la plataforma.
- 2.- Subir fotografías de las caras cuyos avatares se desean recrear.
- 3.- Retocar el resultado para hacerlo más fiel a la realidad

Just Dance Now

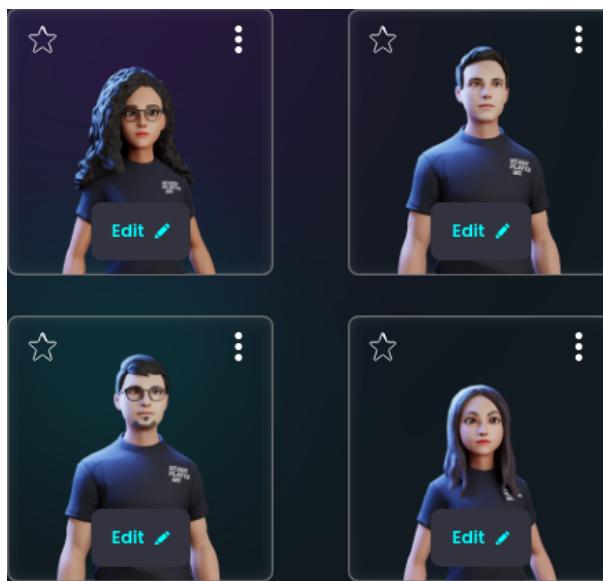


Imagen 14: Avatares

- 4.- Tras descargar los avatares en formato .glb, deben ser importadas a blender (formato .blend).
- 5.- Desde blender se modifican las posiciones del esqueleto del personaje para que siga la secuencia de baile.



Imagen 15: Ajuste de posiciones del avatar

- 6.- Finalmente, se renderiza un vídeo con los tiempos especificados, este vídeo será el que se muestre al jugador.

Aunque, una vez cargado el video con open cv se muestra un retardo en el mismo, por ello se ha aumentado la velocidad en la que se reproduce el vídeo.

La segunda parte consiste en que, para cada posición, el programa contará el tiempo que el jugador está realizando la posición correcta ($t_{correctx}$), como se puede observar en la siguiente imagen: Donde t_{errorx} es el tiempo que el jugador no está realizando el

movimiento correcto. La puntuación se calcula de la siguiente manera:

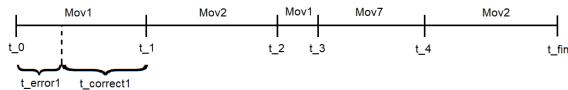


Imagen 16: Asignación de puntuación en base al tiempo

$$Puntuacion_{movx} = \frac{t_{correctx}}{t_x} (2 - aciertos_mediapipe)(2 - aciertos_poses)$$

$$Puntuacion_{total} = \frac{\sum_{x=0}^x t_{correctx}}{t_{fin}}$$

Siendo aciertos_mediapipe=0,9375, aciertos_poses 0,8856 (obtenido del apartado 5.1), t_x el tiempo del movimiento x y t_fin el sumatorio de t_x.

Cuando termine el juego mostrá por pantalla una estadística de puntuaciones para cada movimiento junto con la puntuación total, un ranking y, si la puntuación está dentro del ranking, permitirá incluir el nombre en él.

4.3. Interfaz gráfica

Para darle verosimilitud al juego se ha creado una interfaz gráfica, esta sale al principio y al final del juego y está compuesta de:

- Un menú principal: Permite pasar a la pantalla de configuración, iniciar el juego o salir de él.



Imagen 17: Menú principal

- Elección de personaje: Permite seleccionar entre los cuatro componenetes el trabajo, aunque solo se ha implementado uno de ellos (Erika).



Imagen 18: Elección de personaje

- Elección de canción: Permite seleccionar entre dos canciones: Rasputin y Never Gonna Give You Up, pero al pulsar Never Gonna Give You Up se muestra un mensaje de no disponible, puesto que no se ha implementado. Adicionalmente, al pasar el ratón por la pantalla suena una cación u otra.

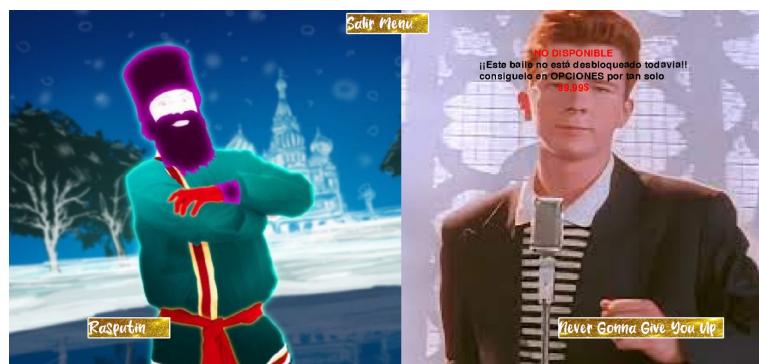


Imagen 19: Elección de canción

- Ajustes: Esta pantalla está compuesta por tres botones: ayuda (activa la opción de ayuda del juego), volumen, comprar 'DLC' (DLC es la canción no disponible) e instrucciones (que indica al jugador cómo funciona el juego).



Imagen 20: Ajustes

Además, mientras se muestran las distintas pantallas suena la canción de 'Boogie Wonderland'.

Para su creación se ha empleado la librería pygame.

5. Experimentación

Destacar que para este proyecto se han empleado varias bibliotecas, para evitar cualquier tipo de fallo, se listan a continuación:

- opencv
- mediapipe
- playsound
- time
- pygame

Tras haber implementado las anteriores secciones en python, se ha testeado su funcionamiento, he aquí un [ejemplo](#) del mismo. Los programas realizados son los siguientes:

- menu.py: Interfaz gráfica; menu para elegir la configuración del juego.
- justdance.py: Programa principal que captura la imagen de la cámara del ordenador y calcula la puntuación.
- take_img_from_video.py: Obtener determinados frames de vídeos.

- test_mediapipe.py: Test para la librería Mediapipe, aplica mediapipe a todas las imágenes de una carpeta.
- take_img.py: Realizar una foto cuando se pulse 'Q'.
- Test_pose.py: Test para la captación de movimientos.
- change_name.py: Asignar el nombre correspondiente a las imágenes de test de manera automática.
- inicializa_ranking.py: Inicializador del fichero xml del ranking

Como material adicional se tiene:

- 170 fotografías para testear la librería Mediapipe, guardados en la carpeta frames_mediapipe.
- 201 fotografías para testear la captación de movimientos, guardados en la carpeta Muestras.
- Un archivo (rasputin.wav) de audio, que es el que suena cuando el jugador empieza a bailar.
- Tres archivos de audio para el menú (boogie.wav, rasputin.wav, rick.wav).
- 5 imágenes para ponerlas de fondo en la interfaz (rasputin.jpg, rick.jpeg, brillo.jpeg, jugadores.jpg e img1.jpg).
- Un archivo .ttf, que determina un tipo de letra para el menú.
- Un video (jd_375.wav) que es el que se muestra al jugador
- 8 vídeos de los que sacamos los fotogramas.
- Fichero xml (config.xml) en el que se guarda la variable que se pasa de un programa a otro, este fichero se autogenera al ejecutar el menu.
- Fichero xml (ranking.xml) que guarda las puntuaciones más altas.
- PowerPoint de la presentación (presentacion.pptx)

Junto con este informe se encuentran los programas del menú, justdance, la presentación, un archivo de texto que muestra los pasos a seguir para su ejecución y los archivos necesarios para que funcione el juego (jd_375.wav, rasputin.jpg, rick.jpeg, brillo.jpeg, jugadores.jpg, img1.jpg, boogie.wav, rasputin.wav, rick.mp3 y el archivo .ttf). El resto de archivos podrán descargarse desde el depósito de [GitHub](#).

Para comunicar los programas se emplea el comando 'os.system("python3 justdance.py")' desde el menú, este comando ejecuta el programa principal, que se llama justdance.py. Además es necesario pasar variables de un programa a otro, para ello se emplea un fichero xml, con el comando 'cv2.FileStorage('nombre_xml.xml', cv2.FileStorage_WRITE)' se crea un fichero xml, en el que se escribe la variable que se desea guardar: 'write('etiqueta', variable)', cuando se ejecuta el segundo programa se debe leer el fichero con: cv.FileStorage('nombre_xml.xml', cv.FileStorage_READ) y se asigna la variable con el comando: 'getNode('etiqueta')', después de ser usado se debe cerrar el fichero 'fs.release()' tanto en el programa del menú como en el principal. En algunos casos, al emplear la función 'getNode('etiqueta')' es necesario añadir el tipo de dato, por ejemplo: getNode('etiqueta').real().

5.1. Testeo del programa principal

Para el testeo del programa principal lo que se ha hecho ha sido realizar diferentes muestras de cada pose por separado para su posterior proceso y análisis. En total se han utilizado 201 muestras, las cuales son imágenes.

En primer lugar, se ha utilizado un programa auxiliar que nos ha permitido hacer las capturas de las fotos rápidamente con la cámara del ordenador y nos la ha guardado en una carpeta para su posterior etiquetado. Una vez hecha todas las fotos se ha utilizado un segundo programa auxiliar que ha estiquetado de forma semiautomática dichas imágenes. El formato escogido es el siguiente: Pose00_Muestra1 (Los primeros dos dígitos representan la pose que se ha imitado y el tercero el número de la muestra).

La idea general de este testeo es comprobar como de preciso es nuestro programa principal haciéndole pasar por él imágenes de poses, las cuales hemos considerado que son correctas y que se podrían dar perfectamente durante el juego.

Para ello, cabe destacar que para el programa del test se ha modificado ligeramente el programa principal, haciendo que compare fotos específicas que se le han pasado en lugar de un video. También recalcar que se han modificado algunos puntos, ya que el programa principal, al utilizar la cámara, se volteó la imagen, cosa que al pasar las imágenes tal

cual no ocurre.

En general lo que hace el programa de testeo es leer todas las imágenes de una carpeta una por una, se apunta el número de pose deseado (El primer y segundo dígito) y compara. Si la pose que ha detectado el programa coincide con la deseada (la que pensabamos que era), nos la apuntamos como que ha sido detectada correctamente y en caso negativo lo contrario.

A continuación, se muestra por pantalla esta información en tres columnas, la primera representa la pose que debería ser esa muestra, la segunda representa la pose que ha detectado el programa y por último el nombre de esa muestra por si queremos saber de cual se trataba:

```
La posicion deseada es: 0. Posicion detectada es:0 Nombre: Pose00_Muestra37.png
La posicion deseada es: 11. Posicion detectada es:12 Nombre: Pose11_Muestra143.png
La posicion deseada es: 13. Posicion detectada es:13 Nombre: Pose13_Muestra175.png
La posicion deseada es: 13. Posicion detectada es:13 Nombre: Pose13_Muestra173.png
La posicion deseada es: 5. Posicion detectada es:5 Nombre: Pose05_Muestra6.png
La posicion deseada es: 12. Posicion detectada es:12 Nombre: Pose12_Muestra13.png
La posicion deseada es: 10. Posicion detectada es:10 Nombre: Pose10_Muestra137.png
La posicion deseada es: 11. Posicion detectada es:11 Nombre: Pose11_Muestra151.png
La posicion deseada es: 13. Posicion detectada es:13 Nombre: Pose13_Muestra18.jpeg
La posicion deseada es: 10. Posicion detectada es:10 Nombre: Pose10_Muestra128.png
La posicion deseada es: 12. Posicion detectada es:12 Nombre: Pose12_Muestra161.png
La posicion deseada es: 15. Posicion detectada es:15 Nombre: Pose15_Muestra192.png
La posicion deseada es: 5. Posicion detectada es:5 Nombre: Pose05_Muestra82.png
```

Imagen 21: Salida por pantalla de los tests

Por último, se muestra por pantalla un pequeño resumen de los resultados obtenidos:

```
POSE 0: 91.667 % de aciertos
POSE 1: 87.5 % de aciertos
POSE 2: 100.0 % de aciertos
POSE 3: 84.615 % de aciertos
POSE 4: 75.0 % de aciertos
POSE 5: 80.0 % de aciertos
POSE 6: 60.0 % de aciertos
POSE 7: 84.615 % de aciertos
POSE 8: 84.615 % de aciertos
POSE 9: 100.0 % de aciertos
POSE 10: 100.0 % de aciertos
POSE 11: 75.0 % de aciertos
POSE 12: 91.667 % de aciertos
POSE 13: 100.0 % de aciertos
POSE 14: 90.909 % de aciertos
POSE 15: 100.0 % de aciertos
POSE 16: 100.0 % de aciertos

-----
Porcentaje de aciertos Total= 88.56403400521049 %

-----
Poses no detectadas: 3
Poses mal detectadas: 21
Pose mejor detectada: 100.0 %
Pose peor detectada: 60.0 %
Muestras Totales: 201
```

Imagen 22: Resumen de los tests

Adicionalmente, se ha realizado una tabla comparativa de manera que pueda realizarse una comparación más visual:

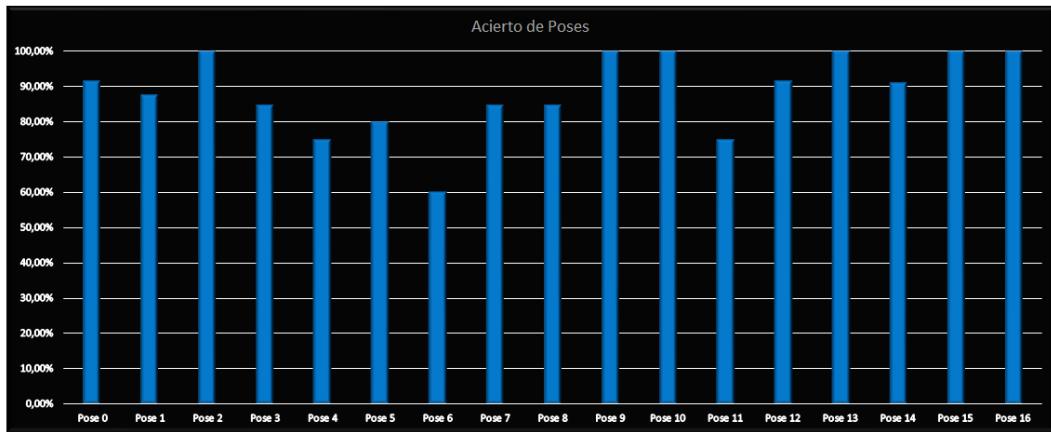


Imagen 23: Comparativa de aciertos por cada pose

Este porcentaje también se tendrá en cuenta a la hora de calcular la puntuación.

6. Conclusiones

Este trabajo no solo nos ha permitido un acercamiento a la librería open cv de python, también se ha aprendido el uso de otras librería útiles y más atípicas como pygame, playsound y mediapipe. Además, se han empleado ideas idénticas o similares a las aprendidas en las clases de práctica de la asignatura (interacción con la cámara, espacios de colores, persistencia con ficheros xml...), por otra parte, se han aprendido características propias del reconocimiento de posición mediante la visión por computador.

Como posibles mejoras al proyecto se han concluido:

- Detención del juego cuando no se detecte bien al jugador.
- Detección más ajustada de movimientos.
- Implementación de más movimientos, bailes y personajes.
- Posibilidad de modo multijugador.

Referencias

- [1] *MediaPipe in python.* en. 1. URL: https://google.github.io/mediapipe/getting_started/python.html.
- [2] *Mediapipe.* en. 2. URL: <https://pypi.org/project/mediapipe/>.
- [3] 3. URL: <https://www.pygame.org/news>.
- [4] *Playsound.* en. 4. URL: <https://pypi.org/project/playsound/>.