Universidad Zaragoza
1542

Trabajo Fin de Grado

Visión por computador para monitorización de terapias con neurotecnología en casa

Computer vision for monitoring neurotechnology therapies at home

Autor

Juan Eizaguerri Serrano

Director

Luis Montesano del Campo

Grado en Ingeniería Informática

Escuela de Ingeniería y Arquitectura

2023-2024

Agradecimientos

Quiero expresar mi más sincero agradecimiento a mi tutor, Luis Montesano, por brindarme la oportunidad de trabajar en este proyecto. También a Carlos Escolano, por prestarme orientación, apoyo y conocimientos que han sido invaluables en el desarrollo de este trabajo. Gracias también a todo el equipo de Bitbrain por recibirme abiertamente, facilitar mi proceso de aprendizaje y sobre todo por su paciencia y disposición para aguantar mis experimentos.

Por último, no puedo dejar de mencionar a mis amigos, familia y pareja por el apoyo que me han brindado en todo momento a lo largo de este proyecto y toda mi etapa universitaria. Este logro no habría sido posible sin su aliento y comprensión.

Resumen

En los últimos años, los avances en neurotecnología han permitido desarrollar sistemas de medida de la actividad cerebral portables y usables por no expertos. Se abre por tanto la oportunidad de realizar intervenciones de neurotecnología en el entorno doméstico, tales como Elevvo, cuyo objetivo es la mejora cognitiva. Su realización en casa permitiría reducir costes, favorecer la comodidad de los sujetos, y permitir intervenciones más largas. Esto sin embargo trae consigo retos en cuanto al aseguramiento de la correcta realización de las sesiones sin supervisión de un experto.

Como posible solución a estos problemas surge este trabajo. El objetivo principal del estudio es desarrollar y evaluar un sistema de visión por computador capaz de supervisar sesiones de terapia de neurotecnología en un entorno doméstico no controlado. Para ello, utilizando técnicas de detección de rostros, detección de puntos faciales y estimación de mirada se ha desarrollado una serie de algoritmos para monitorizar el movimiento de los usuarios, determinar si están hablando o tienen los ojos cerrados y hacer una estimación general de hacia dónde miran.

El proyecto se centra en varios componentes clave: el estudio del arte de las tecnologías relacionadas, el desarrollo e implementación de algoritmos de monitorización mediante visión por computador, el diseño y ejecución de un protocolo de recogida de datos, y su utilización para evaluar los algoritmos desarrollados. Por último se muestra la viabilidad del sistema propuesto mediante el desarrollo e integración de un sistema de monitorización en tiempo real completo en la plataforma de neurotecnología de Bitbrain.

Se discute la precisión de los distintos algoritmos utilizados y se proponen posibles mejoras en la eficacia del sistema. Los resultados obtenidos muestran que la visión por computador es una opción viable para identificar con precisión patrones de movimiento relevantes para la monitorización de terapias con neurotecnología.

Tabla de contenidos

Lista de figuras

Lista de tablas

Capítulo 1

Introducción

1.1. Neurotecnología para la mejora cognitiva

Elevvo es una tecnología de mejora cognitiva desarrollada por Bitbrain, empresa en la que se desarrolla este TFG. Busca como objetivo la mejora cognitiva mediante la modificación de patrones de actividad cerebral relacionados con el rendimiento cognitivo. En concreto, Elevvo implementa un sistema de interfaz cerebro-computador en bucle cerrado en el cual:

1. Se registra la actividad eléctrica cerebral de una persona (electroencefalografía o EEG).
2. Se aplican algoritmos para la decodificación en tiempo real de los patrones cerebrales de interés.
3. Se muestran al usuario (en una pantalla de ordenador) los niveles de dichos patrones cerebrales en forma de un cuadrado que cambia de color con estos niveles.

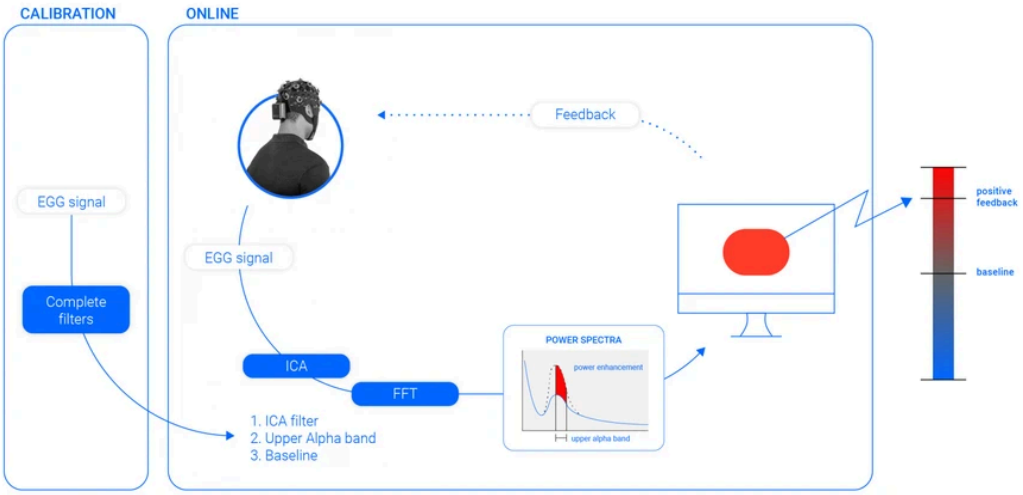
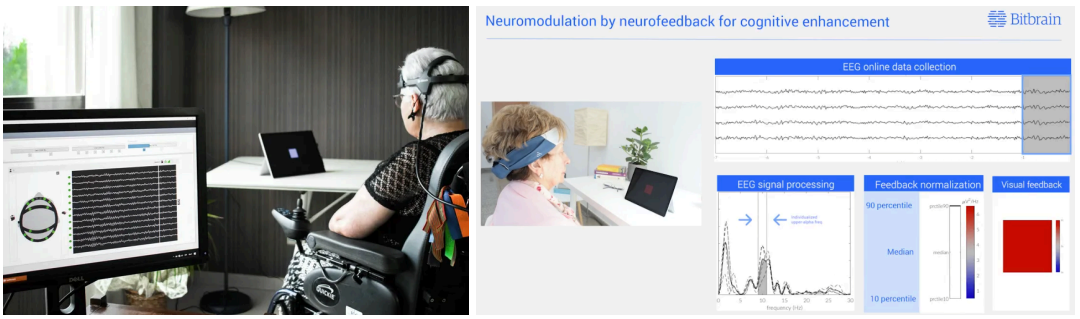


Figura 1.1: Funcionamiento en bucle cerrado de Elevvo [90].

De esta forma, la persona puede modificar (hasta cierto punto) su propia actividad cerebral hacia patrones que están relacionados de forma positiva con el rendimiento cognitivo. Este tipo de procedimientos ha mostrado su potencial para la mejora de memoria de trabajo, velocidad de procesamiento y atención sostenida [94].

1.2. Del laboratorio a casa

Este tipo de sesiones se realizan habitualmente en el contexto de investigación en laboratorio, en el que el usuario es asistido por un operador experto que se encarga tanto de la colocación del equipamiento como de guiar a la persona en la correcta realización de la sesión.



(a) (b)

Figura 1.2: Realización y monitorización de Elevvo [90, 92].

Recientemente surgió la oportunidad en Bitbrain de llevar esta neurotecnología a la casa de los participantes. Esto presenta ventajas para los usuarios como la posibilidad de la realización de más sesiones, la reducción de costes de desplazamiento a los laboratorios y la comodidad de realizar las terapias desde casa. Sin embargo, este cambio de enfoque provoca también la problemática de guiar a la persona en la correcta realización de la intervención sin asistencia de un tercero.

Como solución a este problema surge la propuesta de utilizar tecnologías de visión por computador para desarrollar un sistema que a través de la cámara del dispositivo en el que se realizan las sesiones monitorice distintos aspectos del comportamiento de los usuarios.

1.3. Alcance del proyecto

El principal objetivo de este proyecto es el diseño y desarrollo de un sistema de visión por computador para la monitorización de los usuarios durante terapias de neurotecnología en casa. Para ello, se hace una valoración de los aspectos que puede ser beneficioso verificar para asegurar una correcta realización de las sesiones, los métodos de monitorización necesarios para hacerlo y las tecnologías sobre las que se apoyan dichos métodos.

Así pues, los objetivos específicos planteados para este proyecto son los siguientes:

- Identificación de los requisitos del sistema a desarrollar, elección de las tecnologías principales e implementación de los algoritmos de monitorización que las utilizan (capítulo 2).

- Planificación y ejecución de una metodología de recogida de datos y su utilización para la evaluación *offline* de los distintos algoritmos desarrollados (capítulo 3).
- Desarrollo e integración en la plataforma de neurotecnología de un sistema de monitorización en tiempo real (capítulo 4).
- Conclusiones y discusión de posibles mejoras futuras (capítulo 5).

Capítulo 2

Visión por computador para monitorización

El objetivo de este capítulo es en primer lugar reconocer las funcionalidades deseables del sistema de monitorización a desarrollar y las limitaciones existentes a la hora de su desarrollo. El segundo objetivo es el estudio del estado del arte de las tecnologías relacionadas con los bloques básicos sobre las que se construye dicho sistema. Por último, se explica el funcionamiento de los distintos algoritmos de monitorización implementados. Se realiza una implementación en Python de cada uno de estos algoritmos para su posterior evaluación.

2.1. Requisitos

Aunque Elevvo cuenta con herramientas para filtrar artefactos en la señal EEG, es conveniente minimizar movimientos corporales y faciales durante las sesiones. Además, diversas tareas requieren mantener los ojos cerrados (tarea de calibración) o mirar a un punto fijo de la pantalla (ejercicios de neurofeedback).

Para asegurar el cumplimiento de estas condiciones se propone un sistema capaz de detectar la presencia del usuario y estimar los movimientos que realiza tanto de traslación como de rotación, los movimientos de la boca, la apertura de los ojos y a dónde está mirando. Este sistema se construye sobre tres tecnologías principales: La detección de rostros, la estimación de puntos faciales y la estimación de la mirada.

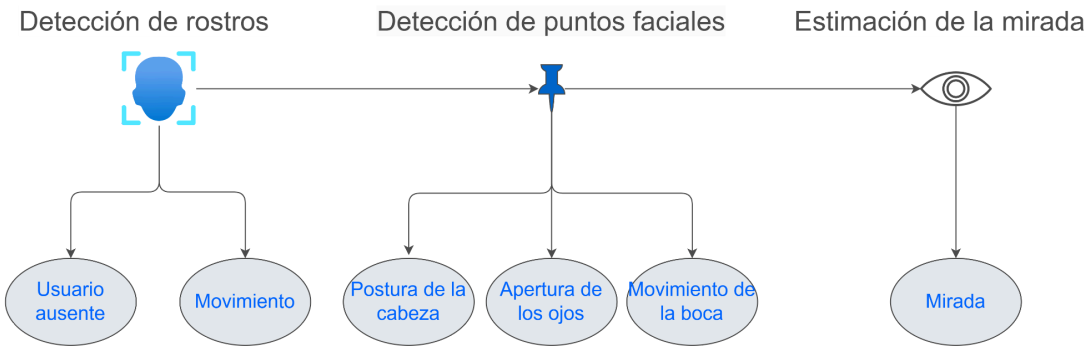


Figura 2.1: Tecnologías y detecciones del sistema propuesto.

Dadas las necesidades explicadas y el contexto en el que se desarrolla este trabajo aparecen varias características de funcionamiento deseadas que influyen en las decisiones tomadas a lo largo del proyecto.

En primer lugar, para proporcionar información útil durante las sesiones, el sistema de monitorización debe funcionar en tiempo real en el dispositivo Microsoft Surface Go 3 en el que se llevan a cabo, que cuenta con un procesador Intel Core i3-10100Y y 8 GB de memoria RAM.

El sistema planteado funcionará en segundo plano durante las sesiones, por lo que no es permisible que su utilización de recursos entre en conflicto con el funcionamiento prioritario de la plataforma de neurotecnología.

Adicionalmente, en la medida de lo posible se quiere limitar las tecnologías utilizadas a aquellas de software libre.

Por último, se debe asegurar el correcto funcionamiento del sistema desarrollado para la variedad de usuarios más amplia posible, y dado que va a ser utilizado en un entorno doméstico, se comprobará también su rendimiento bajo distintas condiciones de iluminación no ideales.

2.2. Bloques básicos

2.2.1. Detección de rostros

El primer paso en la monitorización de los usuarios es la detección de rostros. Esta técnica consiste en encontrar los rostros que existan en la imagen, y si los hay, buscar el rectángulo que los delimita.

La detección de rostros es una de las técnicas fundamentales de la interacción persona-ordenador mediante la visión por computador [33, 61, 62, 63], por lo tanto, existe una gran cantidad de estudios y métodos propuestos para la resolución de este problema. Con la limitación de funcionamiento en tiempo real en mente, se van a evaluar distintos detectores que se pueden agrupar en dos grupos principales, la detección mediante algoritmos “tradicionales” de visión por computador y mediante redes neuronales. Erik Hjelmås propone en [33] una agrupación de las técnicas de detección de rostros en dos grandes grupos que coinciden con los mencionados anteriormente, métodos basados en características o *feature-based* [67], y basados en imagen o *image-based* [66].

- **Detección mediante algoritmos “tradicionales” de visión por computador:**
    - **Haar Cascade**: Es un método de detección propuesto por Paul Viola y Michael Jones en 2001 [44]. Durante la fase de entrenamiento se calculan las características de Haar de las imágenes y se seleccionan las mejores mediante AdaBoost [46]. Utiliza clasificadores en cascada para acelerar el proceso de clasificación descartando regiones no faciales. Se utilizará la implementación del algoritmo de la librería OpenCV.

- **Histogram of Oriented Gradients** (HOG): El histograma de gradientes orientados es un descriptor de características que consiste en la división de la imagen en bloques de tamaño fijo. Para cada uno se calcula la dirección y magnitud del gradiente y se crea un histograma del mismo. Cada uno de los valores del vector del histograma se considera una característica. El vector de características de la imagen es la concatenación de los vectores de los histogramas de los bloques que la conforman [47, 68]. Habitualmente se utilizan en conjunto con algoritmos de aprendizaje automático como máquinas de vectores de soporte (SVM) [48, 49] para entrenar sistemas de detección y clasificación, como es el caso de la detección de rostros. Para la evaluación del algoritmo se utilizará la implementación del detector frontal de rostros de la librería Dlib.

- **Detección mediante redes neuronales:**

  - **MTCNN**: Es un modelo de detección facial propuesto por Kaipeng Zhang en [50]. Utiliza redes neuronales convolucionales multitarea en cascada para detectar rostros. El proceso de detección está dividido en 3 etapas llevadas a cabo por tres redes neuronales de complejidad creciente dedicadas a generar posibles candidatos, refinar los resultados y generar el resultado final, respectivamente.

  - **DNN**: Existen multitud de modelos de redes neuronales profundas pre entrenadas capaces de detectar rostros en imágenes [52, 65]. Se va a utilizar el modelo *res10_300x300_ssd_iter_140000* ofrecido por OpenCV, que utiliza una red neuronal residual o *ResNet* [51] que utiliza *Single Shot MultiBox Detector* [54] para la detección de rostros.

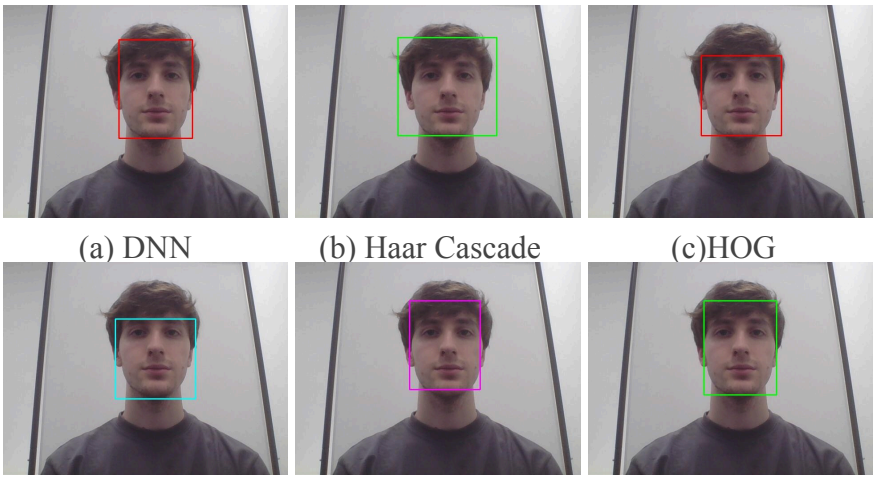
  - **MediaPipe**: Es un framework de código abierto desarrollado por Google dirigido al desarrollo de pipelines que utilizan algoritmos de aprendizaje automático, especialmente para procesamiento de vídeo y audio [69]. Adicionalmente, MediaPipe ofrece modelos y soluciones ya preparadas para ser utilizadas fuera de su framework. Una de ellas es la detección de rostros que será evaluada en esta sección.

  - **Yunet**: Es un detector de rostros mediante redes neuronales centrado en la eficiencia, busca encontrar un equilibrio entre el número de parámetros y capas de la red neuronal utilizada (y por tanto su tiempo de ejecución) y la precisión de los resultados, para ser capaz de ser utilizado en sistemas empotrados [55]. Se utilizará el modelo ONNX [70] *face_detection_yunet_2023mar.onnx* ofrecido en el GitHub ligado a la publicación.

En el caso de uso específico en el que se van a aplicar estas soluciones el usuario se encuentra en frente de la cámara a una distancia cercana. Por lo tanto, algunos de los

aspectos que habitualmente se tienen en cuenta para la evaluación de estos métodos, como el funcionamiento ante caras parcialmente ocluidas o de distintos tamaños en la imagen [22, 23], carecen de importancia significativa en este caso. Sí tiene importancia el funcionamiento de los modelos ante distintas condiciones de iluminación, y con usuarios que llevan puesta una banda y que puedan llevar gafas.

En primer lugar se prueban todos los algoritmos y se comprueba la calidad de sus detecciones en condiciones ideales. En la Figura 2.2 se observa que todos los modelos hacen una detección similar, con la única diferencia de la forma del rectángulo estimado de la cara.



(a) DNN (b) Haar Cascade (c)HOG

(d) MediaPipe (e) MTCNN (f) Yunet

Figura 2.2: Detección facial frontal mediante diferentes modelos de detección.

Dado que uno de los objetivos del proyecto es estimar la rotación de la cabeza del usuario, es importante que la detección de rostros funcione en un rango amplio de ángulos de rotación. Haciendo una comparación cualitativa de las detecciones en distintos rangos de rotación e iluminación (ver Figura 2.3) se aprecia que los detectores DNN y MediaPipe son los más consistentes bajo las distintas condiciones probadas. Haar Cascade falla al rotar la cabeza independientemente de la iluminación. Los resultados de HOG también son considerablemente malos, fallando en casi todos los casos en los que la cabeza está girada, y cuando funciona muestra un error notable, como en el caso a 45º y buena iluminación. MTCNN funciona independientemente del nivel de luz, pero deja de detectar rostros entre los 45 y 90º. Finalmente, Yunet presenta un buen desempeño a excepción de en los casos límite probados de iluminación y giro.

Figura 2.3: Modelos de detección facial ante distintas condiciones de posición e iluminación.

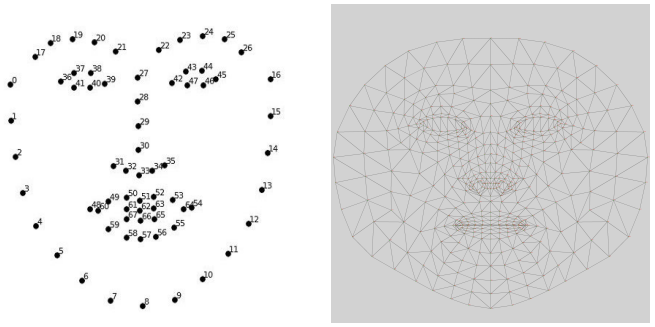
2.2.2. Detección de puntos faciales

Al igual que la detección de rostros, la detección de puntos faciales o *landmarks* es uno de los pilares fundamentales relacionados con el análisis de características faciales mediante la visión por computador. Consiste en localizar puntos específicos de los rostros que aparecen en una imágen. La detección de puntos faciales tiene múltiples usos en distintos ámbitos, como por ejemplo segmentación de las facciones de la cara, reconocimiento facial, detección de expresiones faciales o estimación de la postura de la cabeza [31, 83, 84].

Se va a utilizar la detección de puntos faciales como base para varios de los sistemas de monitorización del usuario desarrollados en la sección 2.3. La calidad de los resultados obtenidos dependerá de la precisión de los puntos 2D y en el caso de la estimación de postura, de la adecuación del modelo 3D correspondiente al objeto real.

Se van a evaluar dos de las técnicas más populares para este problema, ambas basadas en el uso de redes neuronales y técnicas de regresión. En primer lugar, el predictor de 68 puntos faciales ofrecido por la librería Dlib, que es una implementación de [9], y en segundo lugar el predictor de puntos faciales de la solución Mediapipe de

Google, este devuelve una predicción de 468 puntos faciales en 2D y una estimación de sus correspondientes posiciones en el espacio tridimensional [29].



(a) Dlib (b) MediaPipe

Figura 2.4: Topología de *landmarks* de los modelos utilizados.

Un aspecto importante de estos sistemas es el rango máximo de posiciones en el que son capaces de alinear los *landmarks* al rostro de la imágen. El modelo de Dlib está entrenado sobre el dataset 300-W [17, 18], que contiene rostros en posiciones de entre -30º y 30º, siendo más del 85% entre -15º y 15º, por lo que no se espera que el alineamiento sea bueno una vez sobrepasado este límite. Sobre los datos de entrenamiento del modelo de MediaPipe, tan sólo se revela que contiene 30K imágenes tomadas con teléfonos móviles. Comprobando cualitativamente el funcionamiento de ambos detectores (ver Figura 2.5), ambos modelos hacen un buen trabajo ante rostros con una rotación pequeña. Al contrario que Dlib, MediaPipe sí llega a estimar una buena alineación a 45º, pero tiene un error muy grande en el caso extremo de 90º.

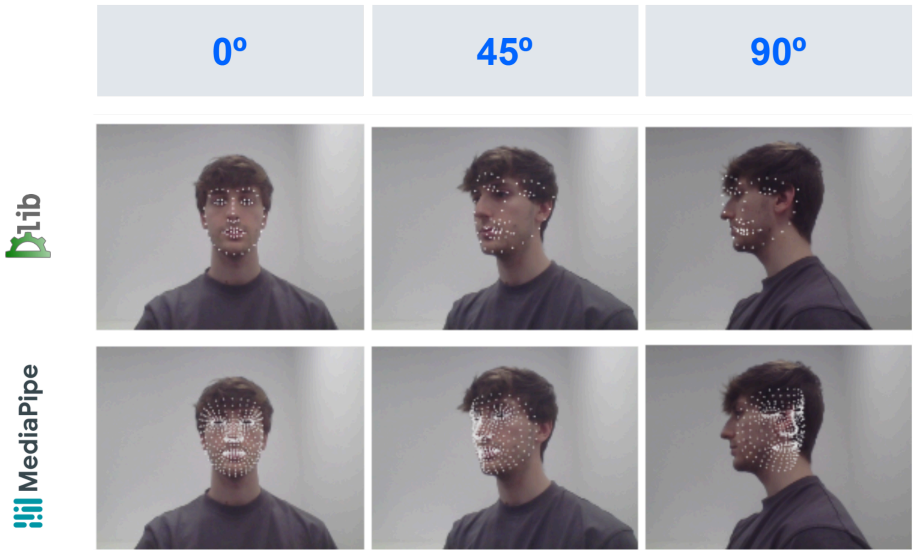


Figura 2.5: Estimación de puntos faciales ante distintos ángulos de rotación de los modelos MediaPipe y Dlib (junto a detector facial Yunet).

Las redes neuronales de ambos modelos utilizan como entrada la imágen RGB recortada del rectángulo de la cara por lo que deben utilizarse en conjunto con un detector de rostros. El predictor de la librería Dlib toma estos valores como argumento por lo que puede ser ejecutado con el detector que se crea conveniente, mientras tanto, MediaPipe no permite esta opción, y utiliza el detector de la librería. Como se ha visto en la sección anterior, este detector tiene un gran rendimiento tanto en calidad de resultados como en coste de ejecución por lo que no supone un problema significativo.

2.2.3. Estimación de la mirada

A la hora de determinar el nivel de implicación de un usuario en un ejercicio propuesto, uno de los métodos más efectivos es saber hacia dónde está mirando. Habitualmente se utilizan para esta tarea técnicas que hacen uso de cámaras de profundidad (habitualmente mediante proyectores y sensores de infrarrojos), pero también existe una amplia variedad de técnicas basadas en imagen RGB que pretenden resolver el problema de la estimación de mirada con un coste computacional bajo y un hardware menos especializado [38, 72, 73] mientras se mantiene un error cercano al obtenido mediante cámaras RGBD [10, 39, 74]. Algunos de estos métodos, pese a utilizar una sóla cámara RGB, requieren otros componentes hardware para funcionar. Por ejemplo, en [38] se utilizan luces en posiciones específicas para reconocer su reflejo en el ojo y corregir las estimaciones calculadas. Este tipo de soluciones no son factibles para el caso de uso en el que se va a utilizar el sistema implementado.

Muchos de los algoritmos actuales calculan el vector de la mirada como la composición del vector del ojo y el de la postura de la cabeza. El sistema de monitorización completo considerará incorrectas las posiciones con la cabeza girada, por lo que conocer la mirada del usuario en este caso se considera innecesario. Por lo tanto, se puede simplificar el problema asumiendo que la cabeza siempre tiene una rotación hacia la pantalla.

Se llama vector del ojo a aquel que conecta el centro del iris a cierta referencia. Por ejemplo, en [38] se utiliza el extremo interno como referencia.

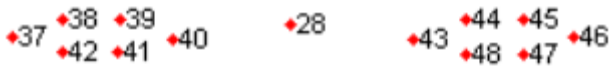
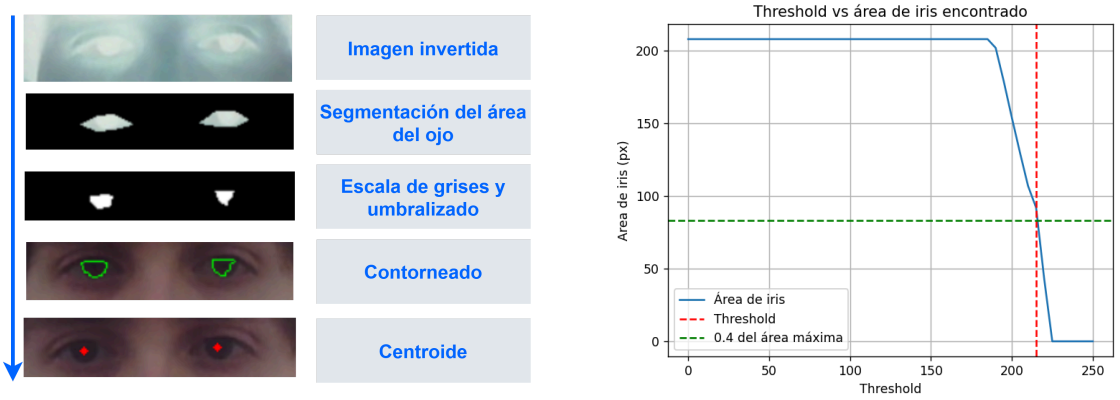


Figura 2.6: *Landmarks* de la región de los ojos del modelo de Dlib.

En el sistema implementado se cuenta con los *landmarks* de los ojos del usuario (ver Figura 2.6), lo que facilita el proceso de obtención del vector de ojo. Al conocer los puntos que delimitan el ojo se puede recortar el fragmento sobre el que realizar las operaciones, además, se puede obtener el rectángulo que delimita los puntos de los ojos a partir de estos puntos. El centro de este rectángulo será la referencia para el cálculo del vector de ojo e.

Como se observa en la Figura 2.7 (a), el cálculo del centro del iris se realiza en varios pasos. En primer lugar se utilizan los *landmarks* de los ojos para crear una máscara que se aplica a la imagen invertida, después se aplica un umbralizado o *thresholding* binario que separa el iris del resto del ojo, junto con una operación de cerramiento utilizando un kernel de tamaño 3x3 para eliminar posibles irregularidades. Se busca el contorno más grande, que coincide con el iris y de él se extraen sus momentos *M* para calcular el centroide *c* del iris.



(a) (b)

Figura 2.7: (a): Esquema de los pasos para el cálculo del centroide del iris. (b): Cálculo del threshold óptimo.

$$c_x = \frac{M_{10}}{M_{00}};\ c_y = \frac{M_{01}}{M_{00}}\ ;\ e = \left(\frac{c_x - rect_{left}}{rect_{width}}, \frac{c_y - rect_{top}}{rect_{height}}\right)\text{g}_x$$

No es posible encontrar un umbral que funcione en todas las ocasiones debido a las posibles diferencias entre los usuarios y los entornos en los que se pruebe el sistema. En primer lugar, el color de ojos del usuario afecta en la luminosidad de los píxeles correspondientes al iris en la imagen al paso a escala de grises. Los ojos claros presentan un menor contraste entre el iris y la esclerótica, mientras que en ojos oscuros es más sencillo encontrar un umbral claro que separa los dos colores. Además, las condiciones de iluminación del entorno también afectan al color del iris que proyecta la imagen. Si se escoge un umbral demasiado bajo se seleccionará todo el área del ojo en lugar de únicamente el iris, por lo que calcular el centroide de esta área no aportará información sobre la mirada. Por el contrario, si el umbral es demasiado alto, se corre el riesgo de una desaparición parcial o total del iris en el área seleccionada (Ver Figura 2.8).

(a) Bajo (b) Bueno (c) Alto

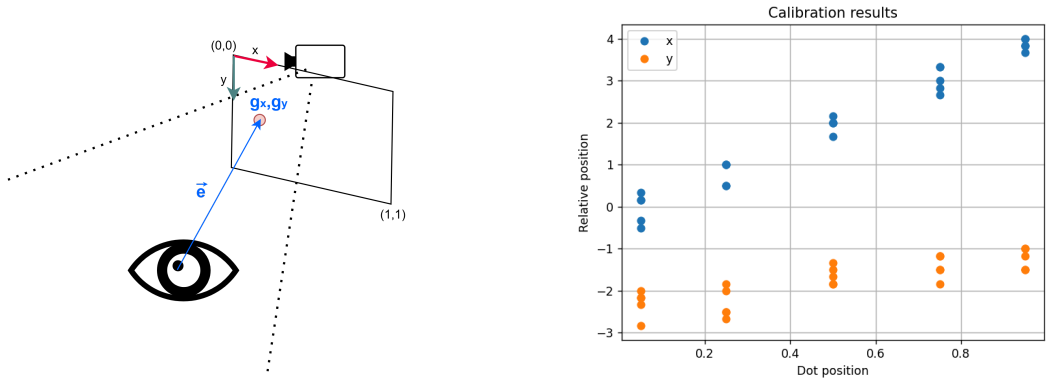
Figura 2.8: Ejemplo de segmentación del iris utilizando distintos umbrales.

Con el objetivo de resolver el problema recién planteado, se ha implementado un método de umbralizado adaptativo basado en el análisis del área del ojo en función del umbral. Este área se calcula como la cantidad de píxeles blancos de la imagen tras aplicar el umbralizado. El área del ojo pasa de su máximo a su mínimo en un rango de umbrales pequeño, el problema consiste en encontrar el umbral correspondiente dentro de ese rango. De un estudio de Carlos Eduardo Palhares sobre el análisis de los ratios de los rasgos faciales [41], se puede extraer que el iris supone aproximadamente un 40% de la superficie visible del ojo. Por lo tanto, se puede calcular el 40% del área máxima del ojo encontrada, y buscar el umbral que resulte en el área más cercana a ese valor (ver Figura 2.7 (b)).

Si la iluminación de la escena no es uniforme, es posible que la luminancia de los píxeles de los dos ojos sea completamente distinta. Esto es especialmente notable cuando el usuario se ve iluminado por una luz lateral. Debido a esto, se calcula y aplica un umbral independiente para cada ojo.

Alternativamente a este cálculo, el modelo de estimación de puntos faciales de MediaPipe tiene la posibilidad de estimar la posición del iris mediante el uso de redes neuronales [58].

Una vez calculado el vector de ojo, la estimación de la mirada consiste en el mapeo de los valores del vector de ojo a coordenadas de pantalla. Para llevar a cabo esta tarea se realiza en primer lugar una calibración donde se pide al usuario que mire a ciertos puntos en pantalla de los que se conocen las coordenadas de pantalla y se calcula y registra el vector de ojo para cada uno. Una vez terminado el proceso de calibración, para cada frame se realiza una interpolación lineal del vector de ojo sobre los calculados durante la calibración para obtener las coordenadas de pantalla. La Figura 2.9 (b) muestra la relación entre la posición del punto al que mira el usuario y el vector de ojo para cada punto del proceso de calibración. Se puede ver que la variación es significativamente mayor en el eje x que en el eje y, lo que puede significar una mayor posibilidad de error al calcular el valor interpolado en el eje vertical.



(a) (b)

Figura 2.9: (a): Diagrama del mapeo del vector de ojo a coordenadas de pantalla. (b): Ejemplo de calibración de mirada.

En la estimación de mirada, se necesita una unidad de medida para la salida independiente del tamaño y resolución de la pantalla que se está empleando. La solución común a este problema es el uso de un sistema de coordenadas de pantalla normalizado entre 0 y 1, en el que el punto (0,0) corresponde a la esquina superior izquierda de la pantalla y el (1,1) a la inferior derecha (Ver Figura 2.9 (a)).

2.3. Algoritmos de monitorización

Una vez seleccionadas las tecnologías básicas se puede utilizar su salida para desarrollar distintos sistemas de monitorización que ofrecen información relevante sobre la correcta realización de las tareas que se proponen al usuario.

A excepción de la estimación de la postura (ver sección 2.3.2), estos sistemas tan sólo utilizan operaciones básicas sobre la salida de los algoritmos para generar algunas medidas derivadas por lo que su coste computacional puede ser despreciado.

2.3.1. Movimiento de la cabeza

Durante las terapias, es deseable que los pacientes adopten una postura cómoda al inicio y permanezcan tranquilos y sin moverse durante el desarrollo de la misma. Para detectar cuándo se mueve el usuario a lo largo del tiempo se utiliza la información proporcionada por la detección de rostros.

El usuario puede realizar movimientos de traslación en los tres ejes de coordenadas. Para detectar los movimientos arriba, abajo, izquierda y derecha basta por comprobar la distancia euclídea del centro del rectángulo de la cara *c* devuelto por el detector facial al de la posición inicial. Esta medida no es efectiva para detectar los movimientos hacia delante y atrás. En su lugar, se utiliza el área *a* del rectángulo. Al ser una medida compuesta fruto del producto de la altura y anchura del rectángulo, su crecimiento es cuadrático con respecto al movimiento, por lo que se utiliza la raíz cuadrada del área, que además tiene la ventaja de utilizar las mismas unidades de medida que para detectar el resto de movimientos. Esto permite agregar las dos medidas en una sóla dando lugar a la variable de movimiento *m* Para asegurar robustez ante distintas resoluciones de imágen se utilizan coordenadas normalizadas entre 0 y 1 donde (0, 0) es la esquina superior izquierda y (1, 1) la esquina inferior derecha.

$$d_c = \sqrt{\left(c_{x\,(0)} - c_x\right)^2 + \left(c_{y\,(0)} - c_y\right)^2}; \quad d_a = \sqrt{\left|a_0 - a\right|}; \quad m = d_c + d_a$$

La Figura 2.10 muestra la salida del sistema en una sesión en la que se realizan series de movimientos cada vez mayores: a) Movimientos de cabeza en el sitio, b) Movimientos laterales; y c) Salir de la imágen de la cámara.

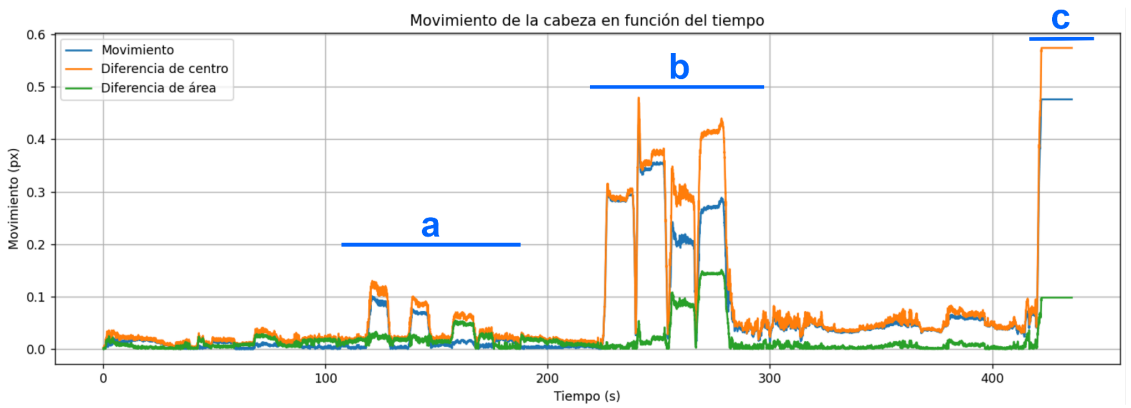


Figura 2.10: Ejemplo de medidas de movimiento durante una grabación en la que se realizan series de movimientos.

2.3.2. Estimación de postura

Existen múltiples métodos para obtener la rotación de la cabeza a partir de los puntos faciales. Algunas técnicas tradicionales utilizan información general sobre las proporciones y la geometría de la cabeza para estimar su plano y por tanto su vector normal. [3] propone utilizar el plano del triángulo isósceles formado por los ojos y la boca proyectado en el plano de imágen. Sin embargo, la mayoría de los métodos actuales utilizan el método de ajuste de modelo o *model fitting* para abordar este problema.

La estimación de los ángulos de rotación y flexión de la cabeza del usuario mediante *model fitting* es un caso del problema de estimación de postura (PnP), en el que dadas las coordenadas de n puntos 3D de un objeto modelo y su proyección en el plano 2D de la imagen (n≥3) se trata de encontrar la posición y rotación de dicho objeto respecto al modelo (ver Figura 2.11). Existen multitud de enfoques para la resolución de este problema [11, 12, 25, 26, 28].

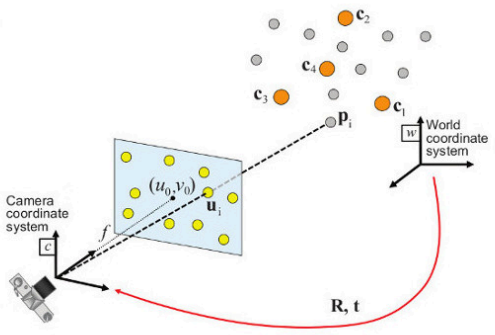
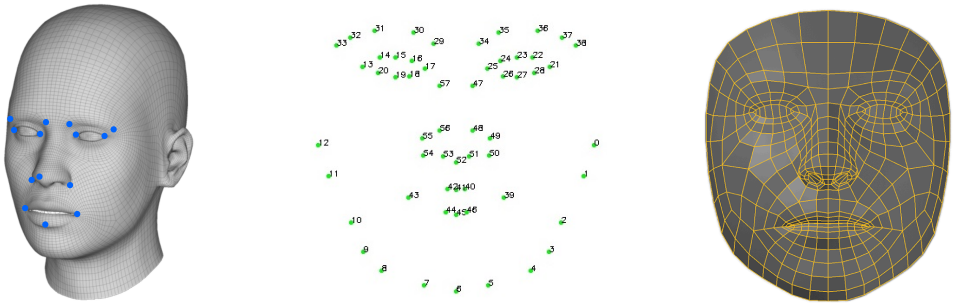


Figura 2.11: Diagrama del problema de estimación de postura (PnP) [91].

Se ha utilizado la implementación de OpenCV del método *Infinitesimal Plane-based Pose Estimation* (IPPE) por su velocidad de cálculo frente a otros métodos como EPnP, RPnP o IPnP al utilizar un número reducido de puntos [58]. Según las pruebas realizadas en [25], utilizando una configuración de puntos cuasiplanar, como es el caso del modelo de un rostro, a medida que se aumenta el número de puntos disminuye el error de rotación, aunque esta mejora es cada vez más pequeña conforme se añaden más puntos. Utilizando tan sólo 5 puntos el ángulo de error medio es menor a 1º para la mayoría de métodos.

Los puntos característicos de la cara se obtienen mediante los métodos explicados anteriormente en el apartado de detección de puntos faciales (sección 2.2.2). El siguiente paso es encontrar un modelo 3D que contenga vértices que encajen con el modelo de *landmarks* utilizado. Para el modelo Dlib se utiliza el modelo de la cabeza humana obtenida mediante muestreo láser de múltiples individuos en [30], cuyos vértices coinciden en gran medida con los del modelo de *landmarking* (ver Figura 2.12 (b)). Por su parte, MediaPipe incluye un modelo canónico con las coordenadas 3D de 468 puntos faciales (ver Figura 2.12 (c)).



(a) (b) (c)

Figura 2.12: (a): Puntos utilizados para el PnP. (b): Modelo 3D utilizado para los *landmarks* de Dlib. (c): Modelo 3D de MediaPipe.

Se utilizan con ambos modelos los mismos puntos como referencia para resolver el problema PnP que se pueden ver en la Figura 2.12 (a). Son los equivalentes a los extremos de los ojos y las cejas, la base de la nariz a ambos lados, la punta de la nariz, y los extremos izquierdo, derecho e inferior de la boca.

Para calcular la proyección de un punto de la imágen en las coordenadas 2D, además de las matrices de rotación y traslación es necesario conocer la matriz de cámara con sus parámetros intrínsecos. Estos parámetros, por lo tanto, son necesarios para la ejecución de los algoritmos de resolución del problema PnP, por lo que han tenido que ser encontrados mediante una calibración de la cámara que se puede ver en el Anexo 2.

El uso de modelos de detección de *landmarks* a nivel de frame da lugar a inconsistencias notables en la posición de los mismos a lo largo del tiempo debido a diferencias en la imagen entre frames provocadas por cambios en variables como la

postura, la iluminación o el ruido del sensor [31], lo que produce un ruido considerable en la postura estimada. El algoritmo de MediaPipe tiene en cuenta este problema e incorpora un filtro de ruido de baja latencia basado en el 1 Euro filter [29, 75]. Se puede observar que la salida de la postura obtenida con los *landmarks* de MediaPipe es visiblemente menos ruidosa que la obtenida con Dlib (ver Figura 2.13). Utilizando la misma idea, se aplica un filtro de ruido de media móvil a la salida del estimador de postura con Dlib.

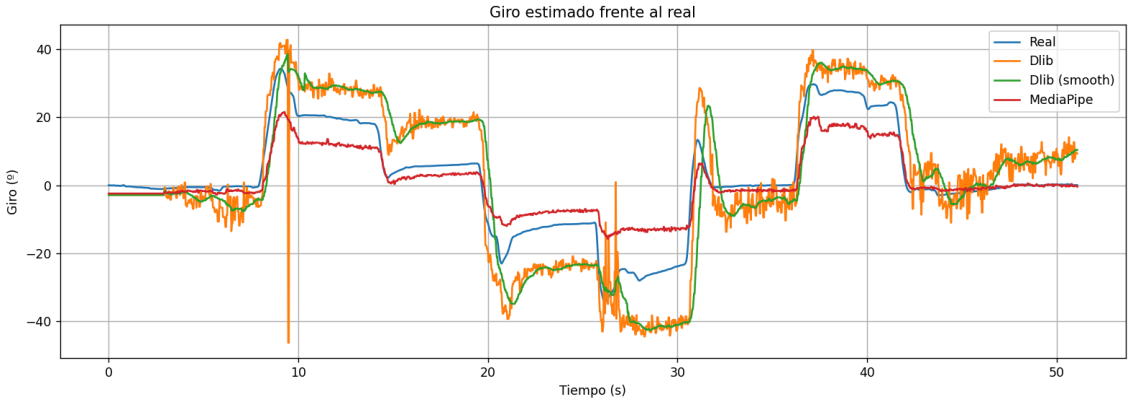


Figura 2.13: Resultado de estimación de rotación (eje z) de una grabación mediante los modelos de *landmarking* de Dlib y MediaPipe frente a la rotación real calculada mediante IMU (ver sección 3.1.4).

Analizando el ejemplo de la Figura 2.13 se observa que el sistema de estimación de postura implementado con Dlib tiende a sobreestimar el ángulo de rotación de la cabeza, mientras que el sistema que hace uso del estimador de *landmarks* de MediaPipe tiene más tendencia a subestimar dicho giro.

2.3.3. Movimiento de la boca

Según la Encuesta Continua de Hogares del Instituto Nacional de Estadística (2020), el tamaño medio del hogar en España es de 2.5 personas [76]. Debido a esto, es habitual que los usuarios sufran distracciones durante las terapias manteniendo conversaciones con las personas con las que conviven. Por lo tanto, uno de los objetivos del sistema de monitorización por visión por computador desarrollado es la detección del habla del usuario. Adicionalmente, la monitorización del movimiento de la boca también permite reconocer bostezos, uno de los principales indicadores utilizados para medir el nivel de somnolencia de las personas [77, 78].

Para realizar esta detección se va a utilizar como base la apertura de la boca. Utilizando los puntos característicos extraídos del rostro del usuario, se calcula la distancia euclídea en el plano de imagen entre el labio superior e inferior. Con el objetivo de ser invariante a tamaño de la cara y distancia a la cámara se normaliza esta distancia por el alto total de la cara h. En otros trabajos se utiliza el ancho de la boca o la altura del labio como factor de normalización en lugar de la altura de la cara. Sin

embargo, se ha decidido utilizar esta medida por ser independiente del ángulo de rotación del rostro y no sólo del tamaño del mismo.

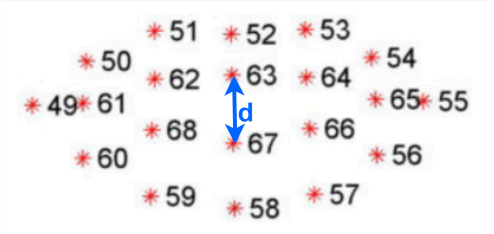


Figura 2.14: Puntos utilizados para el cálculo de la apertura de la boca.

Otros trabajos previos utilizan esta medida para devolver un resultado positivo cuando supera un umbral especificado, sin embargo, este método puede producir falsos positivos ante movimientos lentos de la boca, además, aunque este método es efectivo para detectar bostezos, tiene una probabilidad error mayor al detectar los movimientos de boca de un usuario hablando que pueden ser mucho más pequeños que los de un bostezo. En su lugar, el detector utilizará la información de la imagen de varios frames a lo largo del tiempo para detectar la velocidad de movimiento de la boca.

$$d = \frac{\sqrt{(x_2^2 - x_1^2) - (y_2^2 - y_1^2)}}{h} \qquad (\%h)$$

$$x_1, y_1 = landmark_{63};\ x_2, y_2 = landmark_{67}$$

Durante el habla, los distintos fonemas utilizados requieren una apertura de la boca distinta, incluyendo algunos, como las consonantes bilabiales sonoros (*B*, *V*), requieren que los labios se toquen para pronunciarlos. Debido a esto, este sistema de detección no puede procesar un resultado frame a frame, sino que deberá tener en cuenta la velocidad de movimiento de la boca *m* para determinar si el usuario está hablando o no. Para ello se calcula la diferencia entre la apertura de la boca en el frame actual y el anterior y se divide por la variación de tiempo entre la toma de las imágenes, dando lugar a la variación de la apertura de la boca en relación al alto de la cara por unidad de tiempo (segundos).

$$m_n = \frac{d_n - d_{n-1}}{t_n - t_{n-1}} \qquad (\%h\ /\ s)$$

La medida resultante toma valor positivo cuando la boca se está abriendo y negativo cuando se está cerrando. Para el problema que se pretende resolver, no interesa saber esto en cada momento, tan sólo si se está moviendo o no, por lo que se utiliza el valor absoluto de la medida. Además, el movimiento de la boca durante el habla no es uniforme, sino que ocurre a golpes con cada sílaba, lo que produce que la medida calculada tenga una gran cantidad de picos. Este problema se ha solucionado utilizando una media móvil de un tamaño determinado *w*, que puede ser el número de medidas

tomadas en los últimos s segundos. La velocidad de elocución es de aproximadamente 5.83 sílabas por segundo en Español y 4 en inglés [14, 15, 16]. Además, como se ha comentado anteriormente, distintos fonemas requieren aperturas diferentes de la boca, por lo que el tamaño de ventana elegido debe ser por lo menos suficientemente grande para suavizar varias sílabas completas. En base a las pruebas realizadas, se ha determinado que un tamaño de ventana de entre 1 y 2 segundos es suficiente para suavizar estas diferencias entre fonemas sin llegar a perder información relevante.

$$m_{n\,(smooth)} = \frac{1}{w}\sum_{i=0}^{w} m_{n-i} \mid (w = |I| \; tq. \; (t_n - t_i < s) \; \forall \, i \in I) \;\; (\%h_{face} \,/\, s)$$

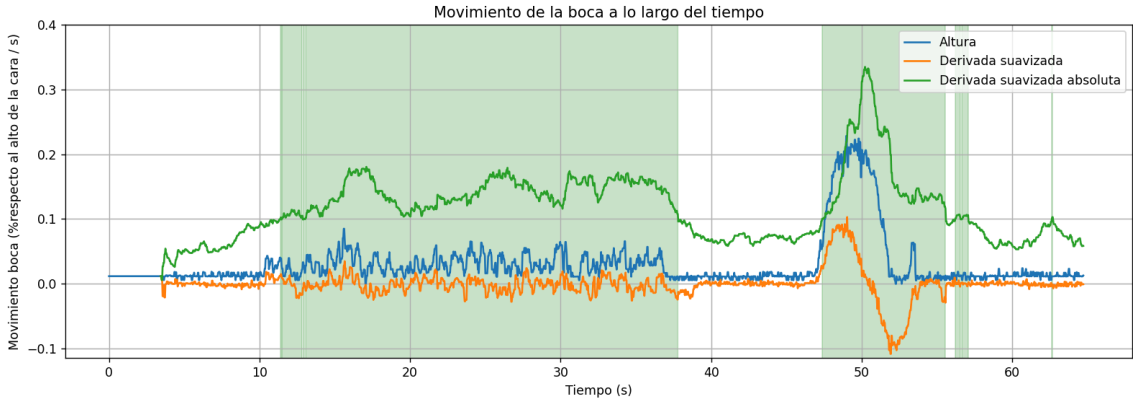


Figura 2.15: Ejemplo de movimiento de la boca a lo largo del tiempo en una ejecución.

Como se puede observar en la Figura 2.15, las medidas calculadas, especialmente el movimiento de boca absoluto suavizado, reflejan claramente los fragmentos de vídeo en los que el usuario está hablando (segundos 12-38), así como los bostezos (segundos 46-55).

2.3.4. Apertura de los ojos

La detección de ojos abiertos es uno de los métodos más utilizados para la monitorización de somnolencia mediante visión por computador, especialmente utilizado en sistemas de monitorización de conductores [6, 34, 81].

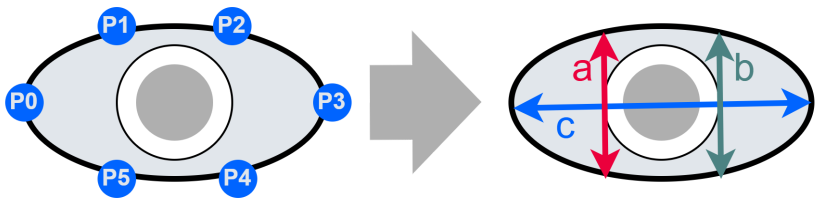


Figura 2.16: Diagrama de los puntos del ojo utilizados para calcular el EAR.

La medida por excelencia para la detección de ojos abiertos es la relación de aspecto de ojo o *Eye Aspect Ratio* (EAR), que denota la relación entre la altura y anchura del ojo [34]. A mayor EAR más abierto está el ojo. Utilizando las técnicas de detección de rostros y *landmarks* desarrolladas en la sección 2.2, es sencillo obtener las medidas de los ojos del usuario en un frame y calcular el EAR. Por último, se utiliza una media móvil para suavizar el resultado y eliminar cualquier pico en el EAR calculado producto de inconsistencias de la detección de *landmarks*.

$$a = ||P1 - P5||;\ b = ||P2 - P4||;\ c = ||P0 - P3||$$

$$EAR = \frac{a+b}{2c}$$

$$EAR_{AVG} = \frac{EAR_{left} + EAR_{right}}{2}$$

$$EAR_{n\,(smooth)} = \frac{1}{w}\sum_{i=0}^{w} EAR_{AVG\,n-i} \mid (w = |I|\ tq.\ (t_n - t_i < s)\ \forall\, i \in I)$$

El ojo se considera cerrado cuando el EAR está por debajo de cierto umbral establecido. Es importante tener en cuenta que las dimensiones del ojo varían en función de distintas personas y etnias, por lo que este umbral debe ser escogido con precaución.

2.3.5. Comprobación de mirada fuera de pantalla

Por último, se quiere tener en cuenta la posibilidad de que el usuario no se esté moviendo ni hablando, mantenga una buena postura y tenga los ojos abiertos, pero no esté mirando a la pantalla del dispositivo. Es el detector menos sofisticado de los descritos en este trabajo, pues su funcionalidad se limita a la comprobación de que los datos de la estimación de la mirada se encuentren dentro del límite establecido.

En primera instancia se podría pensar en comprobar que las coordenadas de mirada calculadas por el estimador descrito en la sección 2.2.3 se encuentren entre 0 y 1 en ambos ejes. Sin embargo, dado que el cálculo final de dichas coordenadas se realiza mediante una interpolación lineal con los datos de la calibración, la salida siempre se encuentra dentro de los límites de la pantalla. En su lugar, se utiliza el vector de ojo para realizar esta comprobación.

Durante la calibración de la mirada se almacenan los valores máximos y mínimos medidos del vector de ojo en ambos ejes de la pantalla. Después, este sistema de monitorización se encarga de comprobar que el vector de ojo calculado se encuentre dentro de estos límites.

Capítulo 3

Evaluación de los algoritmos

El siguiente paso a la implementación de los algoritmos es la evaluación de los mismos. Este capítulo aborda la metodología seguida para la recogida de un conjunto de datos sobre el que evaluar los algoritmos, la explicación de las herramientas utilizadas para esta tarea y el planteamiento de las pruebas a que se van a realizar.

3.1. Metodología

3.1.1. Protocolo y grabación

Con el fin de evaluar las distintas implementaciones de las partes que componen el sistema monitorización, es necesario un conjunto de pruebas con datos conocidos. Para ello se ha diseñado un protocolo de grabación en el que se pide a los usuarios que realicen ciertas tareas. Particularmente, el objetivo del protocolo es producir datos que permitan probar los siguientes aspectos, poniendo especial atención a los casos límite.

- Existencia o no de rostros en la imágen.
- Movimientos de giros de cabeza y traslación del usuario.
- Usuarios hablando.
- Usuarios con los ojos abiertos y cerrados.
- Usuarios mirando hacia distintos sitios dentro y fuera de la pantalla del dispositivo.

(a) Equipo Ikon de Bitbrain

(b) Tobii Pro Nano

(c)Microsoft Surface Go 3

Figura 3.1: Hardware principal utilizado para las grabaciones.

La grabación se realiza utilizando el equipo Ikon de Bitbrain, un dispositivo de EEG en forma de banda que cuenta con 5 sensores secos textiles. Esto es con el fin de recoger datos de la unidad de medición inercial (IMU) incorporada en el equipo de EEG. También se recogen datos del *eye tracker* Tobii Pro Nano durante la grabación. Las señales de ambos dispositivos se utilizarán como *ground truth* en el proceso de evaluación de los algoritmos. Adicionalmente a estas medidas, también se almacena la

señal de los 5 canales de EEG del equipo, que puede resultar útil en líneas de trabajo futuras (ver Tabla 3.1).

Dispositivo de captura	Medida
IMU	Ángulos pitch, yaw y roll de la cabeza (º/s)
	Aceleración en ejes x y z (m/s^2)
	Fuerza magnética en ejes x y z (mT)
Eye tracker	Diámetro de la pupila (mm)
	Posición (x, y, z) del iris
	Coordenadas (x,y) en pantalla de la mirada
	Detección de la pupila (bool)
EEG	Actividad cerebral, 5 canales.

Tabla 3.1: Señales de la plataforma almacenadas durante la recogida de datos.

Para leer y registrar la información de los dispositivos conectados, ha sido necesario implementar un módulo en C++ integrado en la plataforma de neurotecnología. Dicho módulo se encarga de grabar en un vídeo en formato *.avi* a una frecuencia lo más cercana posible a 30 FPS mientras registra el *timestamp* de cada frame grabado en un fichero *.csv*. Además el programa también se encarga de mostrar en pantalla la imágen del usuario, o las pantallas que específicas de las tareas 1a, 1b, 4c y 5a del protocolo.

Antes de comenzar la grabación se hace una introducción al usuario de las tareas que se van a realizar y se le prepara para la grabación colocando el equipo de EEG y calibrando el *eye tracker* (desarrollado en el Anexo 1). Tras realizar las calibraciones necesarias comienza la grabación, tomando medidas de las unidades de adquisición a sus respectivas frecuencias de muestreo. En concreto, se pide realizar las siguientes tareas:

1) **Tareas para la comprobación de la mirada** (100s). El primer paso está dirigido a la calibración de los algoritmos de estimación de mirada evaluados, los otros dos tienen como objetivo comprobar su rendimiento recién calibrados, sin que el usuario se haya movido todavía.
    a) Realizar la calibración de ojos siguiendo las instrucciones de pantalla (30s). Aparecerán puntos en pantalla uno a uno y de manera ordenada, situados en las posiciones 10%, 30%, 50%, 70% y 90% del ancho de pantalla en el eje x, y en las posiciones 25%, 50% y 75% del alto de la pantalla en el eje y. Para cada punto, se da un segundo para que el usuario fije su mirada en dicho punto y después se toman 5 fotografías a lo largo de otro segundo. Esto servirá como información para la calibración de los distintos sistemas de estimación de mirada evaluados.

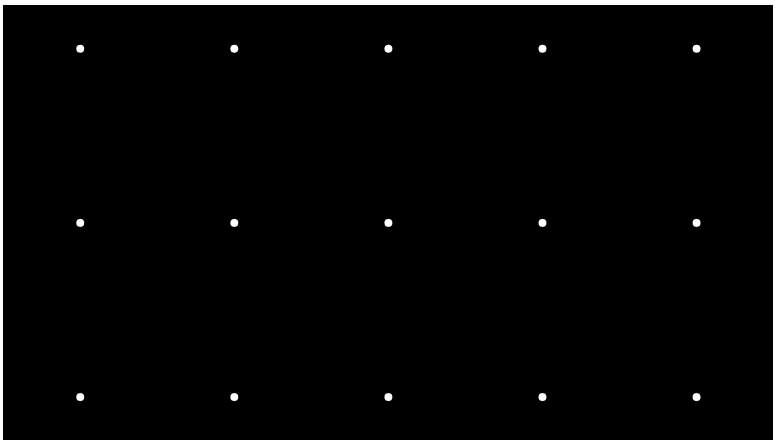


Figura 3.2: Puntos de calibración de mirada.

b) Repetir el proceso del punto anterior (30s). Es en este paso en el que comienza el registro de información en el vídeo y los ficheros de medidas. Se repite el proceso del apartado anterior en el que el usuario tiene que mirar a los puntos uno a uno, pero no se toman fotografías. En su lugar, la imagen se incluye en el vídeo grabado.
c) Mirar fuera de la pantalla (40s). Sin mover la cabeza, mirar a la izquierda, derecha, arriba y abajo de la pantalla, en ese orden, durante 10 segundos en cada posición. El usuario es libre de mirar a distintos puntos siempre que sean en la dirección requerida y fuera de la pantalla.

2) **Tareas para la comprobación de ojos abiertos** (20 s).
   a) Sin pestañear, mantener la posición mirando a la pantalla (10 s).
   b) Cerrar los ojos manteniendo la posición (10 s).
3) **Tareas para la evaluación de la estimación de postura** (80 s).
   a) Sin girar el resto del cuerpo, girar la cabeza lentamente hacia la posición pedida. Una vez alcanzado el límite, mantener la posición hasta que pase el tiempo propuesto.

Posición	L	C	R	C	U	C	D	C
Tiempo (s)	10	10	10	10	10	10	10	10

Tabla 3.2: Tiempos y giros de la tarea 3a del protocolo. Los símbolos L, R, U, D, y C se corresponden con las posiciones izquierda, derecha, arriba, abajo, y centro, respectivamente.

4) **Tareas para la evaluación de la detección de rostros y movimiento del usuario** (110 s).
   a) Permanecer en la posición central (30s).
   b) Sin dejar de mirar hacia la pantalla, Moverse a las posiciones deseadas y permanecer en esa posición durante el tiempo de la tarea (40s).

Posición	L	R	F	B
Tiempo (s)	10	10	10	10

Tabla 3.3: Tiempos y movimientos de la tarea 4b del protocolo. Los símbolos L, R, F y B se corresponden con las posiciones izquierda, derecha, adelante y atrás, respectivamente.

c) Hablar manteniendo la posición central (30s). Para este paso, se mostrará al usuario un texto que leer hasta que termine el tiempo.
d) Salir del cuadro de imagen de la cámara (10s).

5) **Tareas para la comprobación de la mirada con usuario movido** (70 s). En este paso se van a repetir las tareas del paso 2 (exceptuando la de calibración) con el objetivo de evaluar si los sistemas de estimación de mirada pierden eficacia tras la acumulación de movimientos del usuario desde el momento de la calibración.

Antes de comenzar cada tarea se dará el tiempo necesario para la explicación de la misma antes de comenzar su realización. Con el fin de agilizar la grabación y facilitar el posterior etiquetado de datos, se ha preparado una tabla que se puede seguir y rellenar durante la grabación, junto a algunas notas que recordar a los sujetos antes de comenzar para facilitar el entendimiento de los ejercicios (Ver Anexo 1).

La señal del EEG se registra a 256 Hz, mientras que la de la IMU y el *eye tracker* lo hace a 32 Hz. Este registro se hace en un tipo de fichero propietario *.bbt* que deberá ser decodificado mediante la herramienta proporcionada por la empresa para su posterior análisis.

Dado que la recogida de los fotogramas no se hace a una frecuencia perfectamente constante, el vídeo generado presenta alteraciones en las que la imagen avanza a una velocidad ligeramente distinta a la real. Para facilitar el análisis visual y el etiquetado de las partes del vídeo se ha diseñado e implementado en Python un algoritmo que utiliza el vídeo y el fichero de *timestamps* como entrada, y dado un framerate objetivo duplica o elimina frames para que el paso del tiempo del vídeo se corresponda con el verdadero.

3.1.2. Entorno de grabación

Las grabaciones se realizan en un entorno controlado sin distracciones y preparado para facilitar la recogida de datos sin errores.

El sujeto se sienta en una silla con ruedas que facilitará el movimiento en las tareas de desplazamiento, frente a una mesa con una Surface apoyada sobre ella, desde la que se realiza la grabación. Durante el proceso de experimentación, a excepción de las fases de comprobación de la mirada y del habla, donde se muestra una pantalla específica en pantalla, el sujeto verá la imagen de la cámara, a modo de espejo. El dispositivo tiene conectado teclado, ratón y *eye tracker*. Sobre la mesa, también se encuentran la banda de EEG y un paquete de toallitas húmedas, necesario para la colocación de la misma.

Figura 3.3: Entorno de grabación del conjunto de pruebas.

Se quiere probar el funcionamiento de los sistemas evaluados tanto en condiciones ideales de iluminación como en el caso realista en el que el sujeto realiza el tratamiento en el salón de su casa. En la condición ideal, se utiliza una luz blanca uniforme que proyecta una luz blanca sobre el usuario de unos 1500 lux. Varios estudios sugieren que los salones tienen habitualmente una iluminancia de entre 150 y 300 lx, de colores cálidos de unos 2700 K [35, 36]. Para la grabación en condiciones de luz de salón se utiliza una lámpara de intensidad y temperatura regulable para iluminar la habitación con una luz cálida de aproximadamente 150 lx. Esto se hace mediante una lámpara regulable colocada lateralmente al usuario para que su rostro no esté uniformemente iluminado.

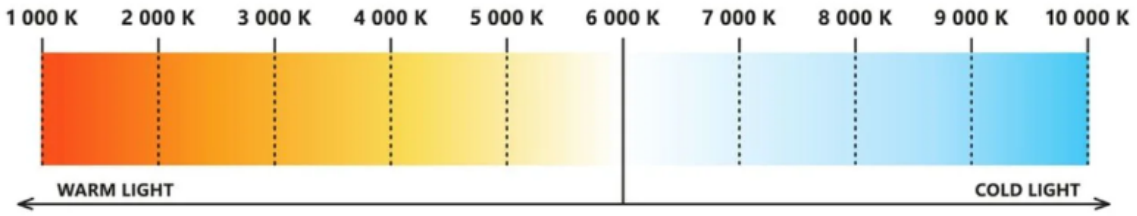


Figura 3.4: Escala Kelvin de la temperatura del color de la luz.

Las medidas de la iluminancia se han realizado con un luxómetro calibrado con error ± 15%. Adicionalmente, se han comprobado estas mediciones con las obtenidas mediante el sensor de un teléfono móvil, obteniendo resultados casi idénticos como se expresa en [37].

Se realiza una grabación del protocolo para cada usuario por cada condición de iluminación. Con el objetivo de asegurar que la diferencia entre los resultados obtenidos de las distintas grabaciones de un mismo usuario se deben al cambio de las condiciones y no de otros factores como el aprendizaje del protocolo, se realiza un balanceo del órden de forma que cada sujeto realiza la primera grabación en las condiciones de luz en las que el último sujeto realizó la segunda. Como resultado, el 50% de los sujetos

realizan primero la grabación en condiciones ideales y después la de condiciones de salón, y el otro 50% procede en orden inverso.

3.1.3. Datos recogidos

El conjunto de datos recogido está formado por los datos de 12 sujetos distintos sometidos a 2 sesiones cada uno bajo distintas condiciones de iluminación. Estos datos son la grabación del vídeo en formato *.avi* a resolución 640x480 píxeles, los *timestamps* de cada *frame* en un fichero *.csv*, la salida recogida de los sensores de la plataforma (*eye tracking*, IMU, EEG) en un fichero *.bbt* y un directorio con las 75 imágenes tomadas durante la tarea 1a del protocolo (sección 3.1.1). Por último, a cada grabación le acompaña un fichero *.json* de elaboración manual que contiene etiquetas de los fragmentos de tiempo en los que se realizan las distintas tareas del protocolo. La descripción de estas etiquetas están disponibles en el Anexo 5.

En la Tabla 3.4 se observa la duración, frames y fotogramas por segundo (FPS) medios de cada una de las grabaciones. Sumando todas las grabaciones, se cuenta con un total de casi 300K imágenes.

Sujeto	Nivel de luz	Duración (s)	Frames	FPS medios
P01	Alta	436	12809	29,38
	Baja	436	12772	29,29
P02	Alta	453	13294	29,35
	Baja	418	12261	29,33
P03	Alta	413	12110	29,32
	Baja	441	12913	29,28
P04	Alta	437	12859	29,43
	Baja	388	11413	29,41
P05	Alta	423	12420	29,36
	Baja	422	12403	29,39
P06	Alta	416	12221	29,38
	Baja	421	12378	29,40
P07	Alta	422	12409	29,41
	Baja	408	12010	29,44
P08	Alta	422	12397	29,38
	Baja	415	11894	28,66
P09	Alta	418	12281	29,38
	Baja	423	12451	29,43
P10	Alta	423	12444	29,42
	Baja	423	12028	28,43
P11	Alta	412	12137	29,46
	Baja	411	10731	26,11
P12	Alta	399	11743	29,43
	Baja	408	10686	26,19

Tabla 3.4: Resumen de los datos recogidos.

La duración media de las grabaciones es de 420 segundos con una desviación estándar de 13.78 s, una variabilidad baja dada por el seguimiento del protocolo. Además, cada una de las tareas tiene una duración muy similar en las distintas grabaciones. Esta uniformidad en el conjunto de datos facilita en gran medida su uso. No parece que haya habido ningún problema de rendimiento del dispositivo de grabación durante la recogida de datos, a excepción de pequeñas variaciones, la mayoría de los vídeos tienen una frecuencia de grabación media de entre 28 y 29 FPS.

Durante la recogida de datos se ha mantenido en mente la posibilidad de que el conjunto de datos generado pueda ser utilizado en estudios futuros, de ahí que se decidiese almacenar la señal EEG de la banda y se haya pedido la realización de ciertas tareas que no forman parte del protocolo, como quitarse la banda, en algunas de las grabaciones.

Con el objetivo de comprobar la variedad de la muestra, para cada uno de los sujetos se ha apuntado su edad, sexo, tipo de piel según la escala Fitzpatrick [93] y otras características, en especial si el usuario lleva gafas o no, que puedan ser relevantes para este u otros estudios. La Tabla 3.5 muestra la información general de la variedad de la muestra en cuanto a sexo y gafas. Por otro lado, todos los sujetos de la muestra tienen entre 24 y 49 años (media 35 y desviación estándar 9.14), y están entre los tipos 1 y 4 en cuanto a tipo de piel. La información desglosada por sujeto está disponible en el Anexo 3.

Según un estudio del Consejo Europeo de Óptica y Optometría, aproximadamente un 55% de la población española utiliza gafas en su día a día [71]. Por lo tanto, la muestra recogida no es perfectamente representativa de la población en este aspecto pero es suficientemente grande como para realizar evaluaciones específicas para este grupo.

Característica		Muestra	Porcentaje (%)
Sexo	Hombre	6	50
	Mujer	6	50
Gafas	Sí	4	33.3
	No	8	66.6

Tabla 3.5: Características de los sujetos del conjunto de datos.

3.1.4. Medidas *ground truth*

Para realizar evaluaciones cuantitativas de los algoritmos se necesita saber en cada momento el resultado ideal esperado que dichos algoritmos tratan de calcular, es decir, una verdad fundamental o *ground truth*. Este *ground truth* puede ser un dato escalar para calcular métricas de error, o un booleano para dar lugar a métricas de confusión. En

esta sección se va a profundizar en las fuentes de *ground truths* que se utilizarán durante las evaluaciones.

En primer lugar, adicionalmente a los datos obtenidos de los dispositivos utilizados durante la grabación, se utiliza información etiquetada manualmente como *ground truth* para las evaluaciones (ver Anexo 5).

Integración de la rotación absoluta

Para la evaluación de la estimación de postura se va a utilizar la información de la IMU para obtener la rotación real del usuario. Para probar la precisión de los sensores se prueba en una grabación controlada en la que se realizan cuatro giros consecutivos a 90, 180, 270 y 360 grados desde el origen (ver Figura 3.5).

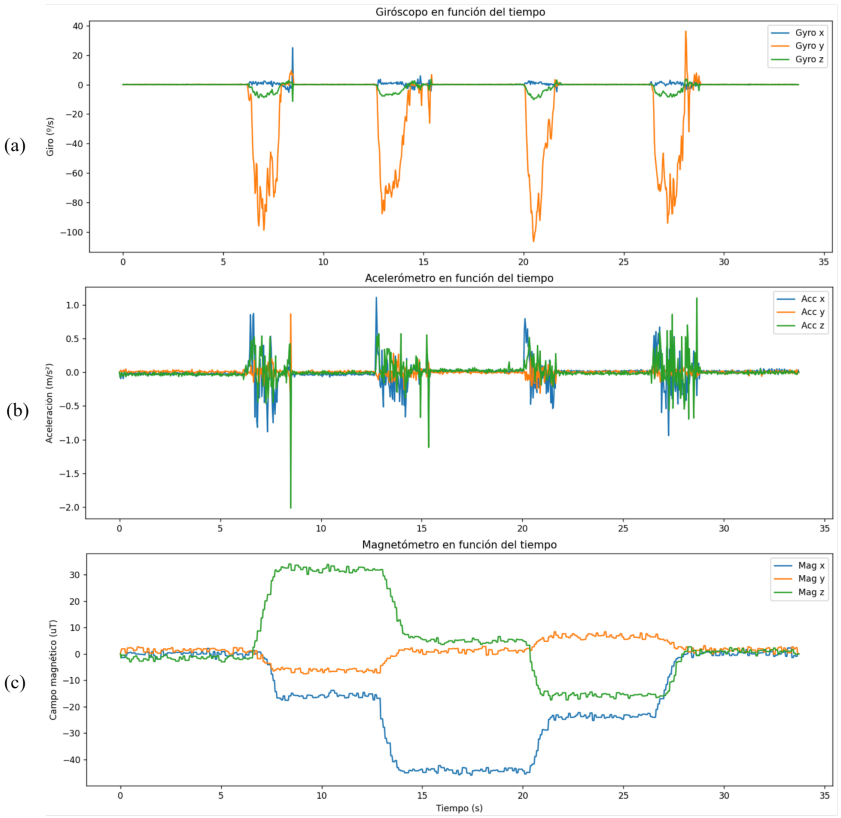
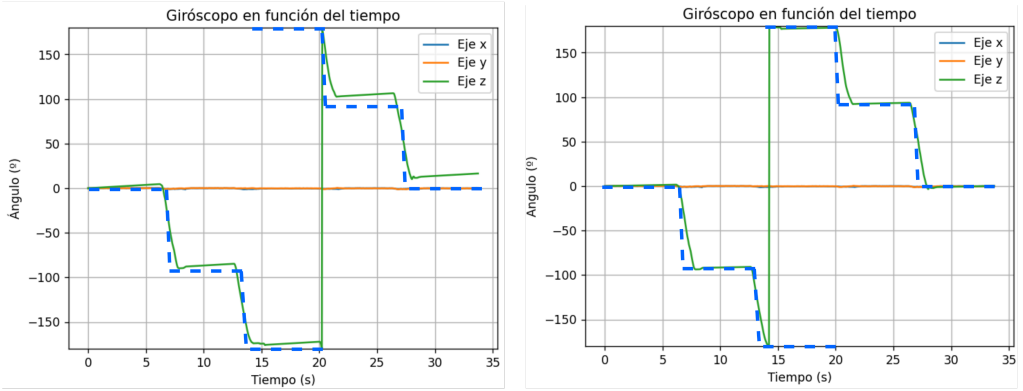


Figura 3.5: Sensores de la IMU durante una grabación de cuatro giros de 90º.

En los resultados de la prueba realizada se observa que tanto el giróscopo como el acelerómetro devuelven medidas de variación en lugar de absolutas. Por lo tanto, para obtener la rotación total en cada momento es necesario integrar las medidas del giróscopo. Esta integración tiene el problema de generar un error acumulativo causado por el error de medida de los sensores. Para resolver este problema se ha utilizado un filtro de Madgwick [7, 8] que utiliza la información del acelerómetro para medir la gravedad de la tierra y utilizar este vector de aceleración como referencia absoluta en la orientación.

Cabe destacar que este filtro tiene también la posibilidad de utilizar el campo magnético de la Tierra como una segunda referencia, sin embargo, la salida del magnetómetro está sujeta a distorsiones causadas por el entorno cercano que podría empeorar los resultados si no se hace una calibración anterior. Esta calibración es un proceso lento y debería hacerse en un entorno controlado con un campo magnético constante para cada grabación [85, 86, 87], lo que dificultaría en gran medida la recogida de datos. Por este motivo se ha decidido no utilizar las medidas del magnetómetro en el filtro de Madgwick. Debido a esto, el eje de rotación horizontal (izquierda-derecha) de la cabeza es paralelo a la única referencia utilizada, por lo que sigue existiendo un error acumulativo en este eje de rotación incluso después de aplicar el filtro. Utilizando la rotación inicial como origen y una escala entre -180º y 180º se obtiene el resultado de la Figura 3.6 (a). Para eliminar el error, dados dos puntos en el tiempo conocidos que tienen la misma rotación se calcula la función de la recta que modela la deriva y se resta al resultado.



(a) (b)

Figura 3.6: Integración de la rotación de la banda. (a): Sin corregir deriva en eje z, y (b): corrigiendo deriva. La línea discontinua muestra la rotación real.

En la Figura 3.6 (b) se observa que la aproximación propuesta soluciona el problema de la deriva. Los intervalos de tiempo en el que dispositivo no se ha movido quedan correctamente reflejados en la gráfica y la rotación absoluta calculada encaja con la real con un error menor a 2º en todo momento.

Valoración de las señales del *eye tracker* como posibles medidas de *ground truth*

Durante las grabaciones, se registran del *eye tracker* varias señales que pueden ser utilizadas como *ground truth* para determinar si el ojo está realmente abierto o no. En primer lugar, se dispone de la información del EEG. Como se expone en [42, 43], el EEG muestra diferentes características con los ojos cerrados y ojos abiertos. Sin embargo, se descarta esta opción por la dificultad y el coste de interpretación de los datos, especialmente cuando se dispone también de la información grabada por el *eye tracker*. Entre sus salidas, existen varias que pueden resultar útiles en este caso: El diámetro medido de la pupila (mm) y el código de validez (bool), independientes para cada ojo.

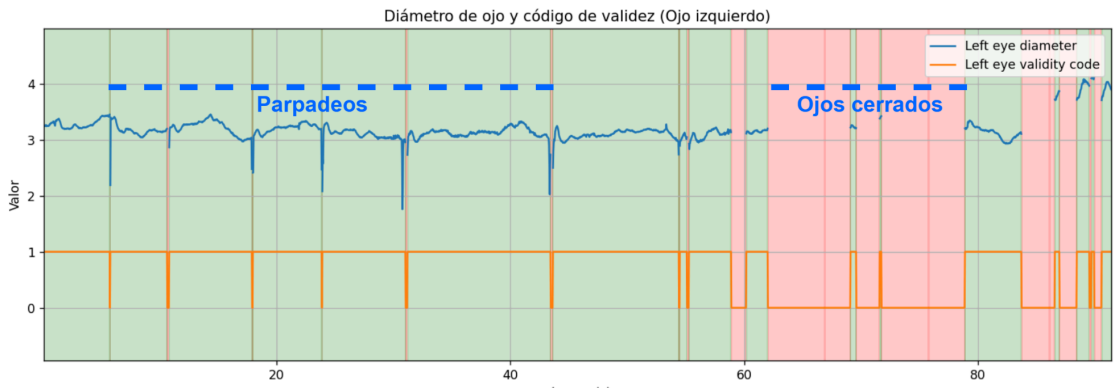


Figura 3.7: Ejemplo de medidas del *eye tracker* durante una ejecución.

Como se observa en la Figura 3.7, ambas medidas aportan una información similar sobre la apertura del ojo, con la ligera diferencia de que el código de validez es 0 durante parpadeos y periodos con los ojos cerrados, mientras que el diámetro del ojo sólo es nulo en estos periodos mantenidos y no durante los parpadeos (aunque sí se puede ver una disminución del diámetro considerable). Dadas las similitudes entre las dos medidas se ha decidido utilizar el código de validez como medida *ground truth* para la evaluación de los algoritmos de detección de apertura de ojos por tener valores booleanos útiles para el cálculo de métricas de confusión.

Por otro lado, el *eye tracker* también proporciona la mirada del usuario en coordenadas normales de pantalla, información que puede ser utilizada como *ground truth* para la estimación de la mirada.

3.1.5. Métodos de evaluación

Los vídeos almacenados durante la grabación pueden ser procesados de forma *offline* de manera que para cada frame se calculen una serie de medidas mediante las técnicas de visión por computador que se quieren evaluar. Dichas medidas quedan registradas en el fichero *.csv* junto al *timestamp* del frame correspondiente.

Los datos del fichero *.bbt* tienen una frecuencia constante, pero los del fichero de medidas no siempre cumplirán dicha condición. Debido a esto, el primer paso en el proceso de evaluación es el emparejamiento de datos de la plataforma de neurotecnología con los de los algoritmos. Para esto, se ha implementado en Python una herramienta que convierte toda la información a frecuencia de muestreo constante emparejando la información de los dos ficheros de medidas.

A partir de este fichero se pueden realizar análisis cuantitativos sobre la calidad de los resultados de las técnicas utilizadas para generar las medidas.

Se pueden dividir las métricas que se van a realizar en dos grandes grupos:

- Para las medidas booleanas se utilizan métricas de confusión, calculando los *True Positives* (TP), *True Negatives* (TN), *False Positives* (FP) y *False Negatives* (FN). A partir de esas medidas se calculan métricas como la precisión (P), la exhaustividad o *recall* (R), el *F-score* balanceado (F_1) y la exactitud o *accuracy* (ACC). La precisión denota qué proporción de los valores predichos como positivos lo son realmente, mientras que el *recall* expresa cuántos de los valores realmente positivos se han clasificado correctamente. La medida F_1 combina estas dos medidas mediante una media armónica. El *accuracy* muestra la proporción de datos etiquetados correctamente.

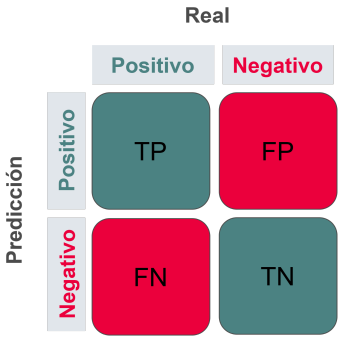


Figura 3.8: Matriz de confusión booleana.

$$P = \frac{TP}{TP+FP};\ R = \frac{TP}{TP+FN};\ F_1 = 2\frac{P \cdot R}{P+R};\ ACC = \frac{TP+TN}{TP+FP+TN+FN}$$

- Para las medidas de valor numérico en las que se cuenta con un *ground truth* escalar como los ángulos de rotación de la cabeza o la mirada, se utilizarán métricas de error. Generalmente, se van a utilizar las medidas *Mean Average Error* (MAE), *Mean Squared Error* (MSE) y *Root Mean Square Error* (RMSE). MAE es la medida más básica, que muestra simplemente el error medio, aunque tiene el problema de que errores positivos y negativos puedan cancelarse entre sí. MSE es la media de los cuadrados de los errores, que penaliza errores grandes, pero puede ser difícil de interpretar. Por último, RMSE tiene las mismas ventajas que el MSE, pero es más interpretable al estar en las mismas unidades que la variable de salida.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right);\ MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2;\ RMSE = \sqrt{MSE}$$

Adicionalmente, se hará uso de las gráficas que se crean necesarias para una mejor visualización de los resultados.

El análisis de rendimiento en tiempo de ejecución se realiza midiendo y almacenando en memoria para cada frame el tiempo de procesamiento de los distintos algoritmos utilizados durante las pruebas realizadas.. Al final de la ejecución se calcula

el tiempo medio de los algoritmos como la media aritmética de las medidas tomadas durante el procesamiento de los distintos vídeos.

Una vez definido el conjunto de datos se diseñan las pruebas a las que se somete cada uno de los algoritmos. En el Anexo 6 se muestran las pruebas propuestas para cada algoritmo, junto con los datos que se utilizan como *ground truth* y la parte de las grabaciones que se utiliza para cada una de ellas.

Algunos de los tests descritos se realizan a modo de análisis de rendimiento que permita realizar una selección informada de las tecnologías utilizadas en la integración final. Otros sin embargo se utilizan con el objetivo de encontrar la configuración óptima de los parámetros de los algoritmos.

Otra porción de las pruebas consiste en separar y comparar los resultados de los algoritmos en función del nivel de iluminación de la grabación para comprobar el impacto de esta variable.

Por último, la utilización de accesorios puede afectar a la capacidad de los detectores de rostros de generar resultados [56, 57], por lo que es importante asegurarse de que los algoritmos de monitorización funcionan correctamente con usuarios con gafas. Para estas pruebas se separan los sujetos 4, 6, 8 y 11 del resto de la muestra para evaluar los resultados.

3.2. Resultados

Este apartado recoge los resultados y el análisis de las pruebas propuestas para probar los distintos algoritmos desarrollados en el capítulo 2. Los análisis de tiempo de ejecución de esta sección se han realizado en un sistema con un procesador Intel Core i7-10710U y 32 GB de RAM.

3.2.1. Detección de rostros

Utilizando la salida booleana de los algoritmos de detección de rostros junto con la información etiquetada del conjunto de datos (Tabla 3.6), se calculan métricas de confusión de los distintos modelos evaluados. Se analiza también el *accuracy* en función de las variables de iluminación y gafas (Tabla 3.7).

Nótese que el detector MTCNN ha quedado excluido de las evaluaciones de esta sección debido a su lenta velocidad de ejecución, de aproximadamente 1178 ms por frame.

Uno de los posibles criterios por los que se podría decidir utilizar un modelo u otro es su tiempo de ejecución, especialmente importante en sistemas en tiempo real como el que se está tratando de desarrollar en este trabajo. La Tabla 3.8 muestra los tiempos de ejecución medios de los distintos detectores al evaluarlos con el dataset.

	DNN	Haar	HOG	MediaPipe	Yunet
P	1.00	1.00	1.00	1.00	1.00
R	0.99	0.67	0.89	0.99	0.99
F_1	0.99	0.80	0.99	0.97	0.99
ACC	0.99	0.67	0.89	0.99	0.97

Tabla 3.6: Métricas de confusión de los modelos de detección facial.

		Accuracy (ACC)				
		DNN	Haar	HOG	MediaPipe	Yunet
Iluminación (lx)	150	0.99	0.60	0.87	0.99	0.97
	2000	0.98	0.76	0.92	0.99	0.98
Gafas?	Sin gafas	0.99	0.73	0.91	0.98	0.98
	Con gafas	0.98	0.57	0.86	1.00	0.95
Total		0.99	0.67	0.89	0.99	0.97

Tabla 3.7: Métricas de confusión de los modelos de detección facial en función de variables de iluminación y gafas.

Modelo	DNN	Haar	HOG	MediaPipe	Yunet
T_{frame} (ms)	39.9	11.5	46.3	3.5	28.5

Tabla 3.8: Tiempo medio de ejecución de los modelos de detección facial.

Se observa que los detectores MediaPipe, DNN y Yunet reportan los mejores resultados con *accuracies* de entre 97 y 99%. Haar Cascade y HOG mantienen parecen tener más falsos negativos que el resto de detectores dando lugar a un *recall* inferior al del resto de modelos.

Además, los modelos basados en redes neuronales apenas sufren ninguna disminución en su eficacia ante iluminación baja o usuarios con gafas, mientras que los métodos tradicionales (Haar Cascade y HOG) muestran un empeoramiento considerable de sus resultados.

Sobre el tiempo de ejecución, todos los detectores evaluados tienen la velocidad suficiente para formar parte de un sistema en tiempo real, permitiendo más de 20 detecciones por segundo, siendo el detector de MediaPipe casi 10 veces más rápido que la mayoría de los demás modelos con un tiempo de ejecución menor a 4 ms. Aunque *Haar Cascade* también es muy rápido, la mala calidad de sus resultados lo descarta como una posible opción. Entre las opciones con mejores resultados, MediaPipe y Yunet son sin duda las dos mejores.

3.2.2. Detección de puntos faciales

Los modelos de detección de puntos explicados en la sección 2.2.2 generan para cada frame una serie de coordenadas correspondientes a ciertos puntos faciales. En dicha sección se ha hecho una valoración cualitativa de la calidad de las estimaciones de puntos en función de la rotación de la cabeza.

Para llevar a cabo una valoración más en profundidad se toman para cada grabación 4 imágenes extraídas de la tarea 3 del protocolo a aproximadamente 0º, 30º, 45º y *max*º de rotación de la cabeza, siendo *max* el ángulo máximo de giro de la cabeza del sujeto. Un estudio realizado sobre 97 sujetos en [13] sugiere que este valor se encuentra en torno a los 73 ± 7º. Cada imágen es procesada por ambos algoritmos de detección y se valora subjetivamente si los puntos estimados coinciden con los reales. La Tabla 3.9 muestra el resumen de los resultados de dicha evaluación.

		0º	30º	45º	*max*	Total
Dlib	Alta	0,92	1,00	0,67	0,00	0,65
	Baja	1,00	1,00	0,00	0,00	0,50
	Total	0,96	1,00	0,33	0,00	0,57
MediaPipe	Alta	0,92	0,75	0.50	0,25	0,60
	Baja	0,75	0,83	0,58	0,25	0,63
	Total	0.87	0,79	0,54	0,25	0,61

Tabla 3.9: Precisión de los modelos de detección de puntos faciales en función de la rotación de la cara.

En general MediaPipe tiene una mejor precisión que Dlib en la estimación de los puntos. Esto es especialmente notable alrededor de los 45° de rotación. Ninguno de los dos modelos es capaz de realizar detecciones consistentemente en el ángulo máximo de rotación.

Dado que el sistema que va a utilizar la detección de puntos faciales debe ser ejecutado en tiempo real, también es importante conocer el tiempo de ejecución de las tecnologías utilizadas.

	Yunet + Dlib	MediaPipe	MediaPipe + Dlib
T detección	28.5	3.50	3.50
T *landmarking*	4.44	14.6	4.44
T total	32.94	18.1	7.94

Tabla 3.10: Tiempo de ejecución medio de los detectores de puntos faciales.

El tiempo medio de procesamiento del modelo de Dlib es de 4.44 ms en y el de MediaPipe 18.1 ms. Hay que tener en cuenta que como se ha comentado anteriormente, el estimador de Dlib necesita realizar una detección facial previa, por lo que en la

evaluación realizada el tiempo completo de procesamiento de cada frame es de 32.94 ms. Sin embargo, la ejecución de la parte específica al *landmarking* es significativamente más rápida utilizando Dlib que MediaPipe, incluso restando a esta el coste de la detección facial. Por lo tanto, la opción más rápida encontrada es la de utilizar el detector facial de MediaPipe junto con el detector de puntos faciales de Dlib, con un tiempo total menor a 8 ms (ver Tabla 3.10).

3.2.3. Estimación de la mirada

En este apartado se comprueban las diferencias entre los métodos *ad hoc* y MediaPipe para el cálculo del centroide del ojo, seguido de evaluación de la estimación de la mirada. Las evaluaciones de este apartado se llevan a cabo utilizando las partes del conjunto de datos correspondientes a las tareas 1 y 5 del protocolo.

En la sección 2.2.3 se han explicado dos posibles métodos de la estimación del centroide del iris. De acuerdo a las pruebas realizadas, la distancia euclídea media entre el centroide calculado mediante el algoritmo *ad hoc* y MediaPipe es de tan sólo 1.65 píxeles, una diferencia que no resultaría en cambios notables en la estimación final de la mirada, por lo tanto, se utilizará únicamente el método *ad hoc* en las evaluaciones de la estimación de la mirada de esta sección.

Pasando a la evaluación de las coordenadas de la mirada devueltas por el algoritmo de estimación, al no haber incluido el vector de posición de la cabeza en el cálculo de la estimación de la mirada, el sistema es potencialmente susceptible a los movimientos que realice el usuario tras la calibración. Se evalúa en qué medida afectan los movimientos realizados a lo largo de una sesión comparando los resultados de los datos recogidos en las tareas 2 y 5 del protocolo de grabación. Como se explica en la sección 3.1.1, estas tareas son la comprobación de la mirada dentro y fuera de la pantalla recién realizada la calibración (tarea 2) y tras unos minutos realizando distintos ejercicios (tarea 5).

Figura 3.9: Oclusión del iris por reflejos en las gafas del usuario.

En último lugar, se estudia la eficacia del detector bajo distintas circunstancias que pueden dificultar el cálculo de la posición del iris. La primera de estas condiciones es la iluminación baja. En una imágen mal iluminada todos los colores de la imágen se

acercan al negro por lo que la umbralización del iris tiene que ser precisa para que la estimación de la mirada sea correcta.

Se estudia también el impacto de que el usuario lleve gafas, las cuales pueden producir reflejos que ocluyen parcialmente el iris dificultando el cálculo de la posición del iris y con esto la estimación de la mirada (ver Figura 3.9). Los resultados de ambas evaluaciones se encuentran en la Tabla 3.11.

		Eje x		Eje y	
		MAE	RMSE	MAE	RMSE
Tarea	Tarea 1	0.23	0.30	0.31	0.40
	Tarea 5	0.24	0.32	0.31	0.41
Iluminación (lx)	150	0.23	0.31	0.31	0.41
	2000	0.24	0.31	0.29	0.39
Gafas?	Sin gafas	0.18	0.23	0.29	0.39
	Con gafas	0.34	0.41	0.33	0.43
Total		0.23	0.31	0.30	0.40

Tabla 3.11: Error medio de la estimación de mirada frente al *eye tracker* en los ejes x e y de la pantalla. Resultados en función de la tarea, iluminación y gafas.

Atendiendo al MAE de la Tabla 3.11, se ve que el error medio en la estimación de la mirada es del 23% del ancho de la pantalla y 30% en altura. Esto significa que las coordenadas estimadas se pueden considerar una aproximación general de la zona de la pantalla a la que está mirando el usuario pero no un punto preciso.

De los resultados obtenidos en función de las distintas variables se pueden extraer varias conclusiones. En primer lugar, se aprecia un deterioro en la calidad de las estimaciones de la mirada en el eje x tras la acumulación de movimientos durante la grabación, aunque los resultados en el eje y permanecen similares. Además, el sistema es robusto ante diferencias en la iluminación de la escena, aunque la utilización de gafas sí influye negativamente en la precisión del sistema.

El estimador de puntos faciales de MediaPipe, que incluye la detección del iris, tiene un tiempo medio de ejecución de 18.7 ms. Mientras tanto, el cálculo del centroide mediante el método *ad hoc* se realiza en un tiempo medio de 4.1 ms, que sumado al tiempo de la estimación de puntos faciales de Dlib resulta en un tiempo de tan sólo 8.54 ms al que se se le suma el coste de la detección de rostros. En caso de utilizar la detección de MediaPipe, el coste teórico de la ejecución de las tres tecnologías principales que componen el sistema de monitorización sería menor a 12 ms.

El cálculo del vector de ojo requiere un tiempo de ejecución medio de 4.3 ms, de los cuales 4.1 se dedican al cálculo del centro del iris. La estimación de mirada mediante la interpolación de dicho vector con los datos de calibración tiene un coste de tan sólo 0.1 ms.

3.2.4. Movimiento de la cabeza

El algoritmo desarrollado en la sección 2.3.1 calcula una medida de movimiento *m* que determina la distancia del usuario desde la posición inicial a la actual.

Para determinar si el usuario se ha movido demasiado desde el inicio del ejercicio se somete la *m* a un umbral que debe ser predeterminado. Utilizando la información etiquetada de las grabaciones donde se indica si el usuario está o no movido, se pone a prueba el sistema utilizando distintos umbrales y calculando las métricas de confusión visibles en la Tabla 3.12. Para evaluar con datos lo más balanceados posibles, se utiliza la información de la tarea 4 del protocolo (ver sección 3.1.1) que consta de aproximadamente 30 segundos en buena posición y 40 segundos en mala posición por cada grabación.

Threshold	0.00	0.5	**0.10**	**0.15**	0.20	0.25	0.30	0.35	0.40	0.45
P	0.44	0.77	**0.96**	**0.98**	0.99	1.00	1.00	1.00	1.00	1.00
R	1.00	0.97	**0.94**	**0.92**	0.89	0.87	0.54	0.41	0.20	0.01
F_1	0.61	0.86	**0.95**	**0.95**	0.94	0.93	0.70	0.58	0.34	0.02
ACC	0.44	0.86	**0.96**	**0.96**	0.95	0.94	0.80	0.74	0.65	0.57

Tabla 3.12: Métricas de confusión de la detección de movimiento con distintos umbrales.

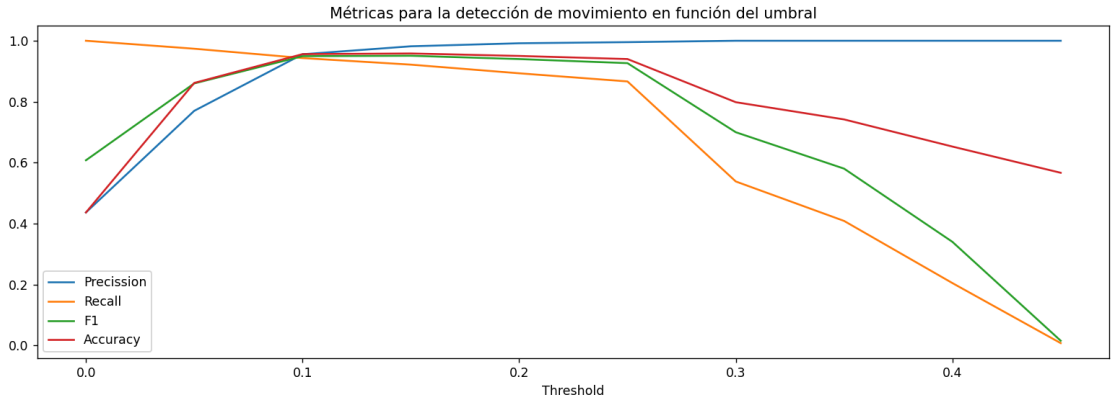


Figura 3.10: Métricas de confusión de la detección de movimiento con distintos umbrales.

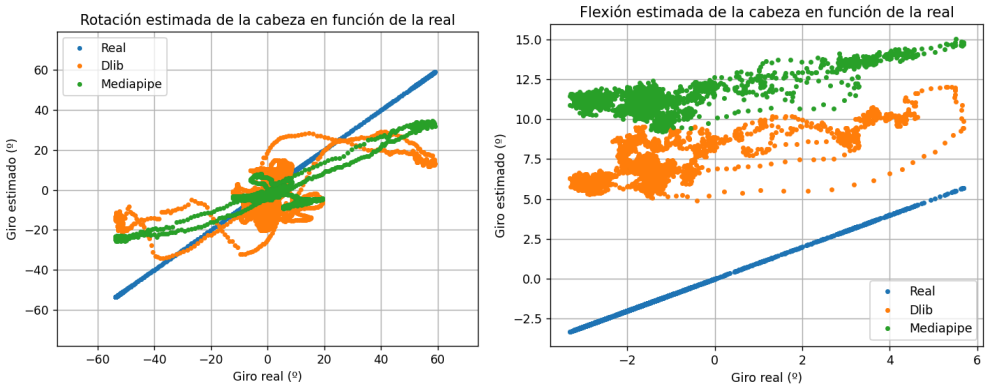
De la Tabla 3.12 y la Figura 3.10 se puede extraer que existe un rango amplio de umbrales entre 0.10 y 0.25 que pueden ser utilizados manteniendo un buen *accuracy*. Dentro de este rango se puede utilizar uno más alto o más bajo dependiendo del balance deseado entre precisión y *recall*. Teniendo en cuenta ambas métricas mediante la F_1, se puede decir que los umbrales entre 0.10 y 0.15 dan lugar a los mejores resultados, acertando el 96% de las ocasiones.

3.2.5. Estimación de postura

Las evaluaciones realizadas en esta sección están destinadas principalmente a comprobar el error del ángulo de rotación de la cabeza calculado por el algoritmo de estimación de mirada utilizando los dos modelos de detección de puntos faciales evaluados en este trabajo. Las evaluaciones se realizan utilizando los datos correspondientes a la tarea 3 del conjunto de datos grabado.

		MAE	RMSE
Rotación	Dlib	8.53	9.88
	MediaPipe	3.96	4.06
Flexión	Dlib	6.43	7.16
	MediaPipe	10.52	10.92

Tabla 3.13: Métricas de error de la estimación de postura.



(a) Rotación (b) Flexión

Figura 3.11: Estimaciones de posición de la cabeza frente a la real de una grabación.

Analizando la Tabla 3.13 se observa que el modelo MediaPipe tiene un error medio menor en el eje de movimientos de rotación mientras que el de Dlib tiene más precisión ante movimientos de flexión. El MAE y el RMSE toman valores muy similares, lo que significa que no hay residuos con errores muy grandes. Generalmente, ambos modelos permiten generar resultados suficientemente fiables para estimar la dirección general de la cabeza.

La Figura 3.11 muestra las estimaciones de la postura de la cabeza en relación a la real. El resultado obtenido concuerda con la observación de la sección 2.2.2 respecto a los ángulos límite de estimación de los modelos.

El estimador de Dlib tiene su límite alrededor de los 30º de rotación. También es pertinente comentar que este estimador tiende a sobreestimar el giro para ángulos pequeños y subestimar el de ángulos grandes. Por el contrario, la distribución de las predicciones utilizando el estimador de MediaPipe mantienen una distribución mucho más lineal y alcanzan un límite mayor, aunque siempre subestimando ligeramente la

rotación. Por la uniformidad de los datos, este error podría modelarse y ser corregido multiplicando por un factor. En cuanto al ángulo de flexión, ambos métodos generan resultados muy similares con un pequeña diferencia fija entre los dos, probablemente generada por las diferencias entre los modelos 3D utilizados para resolver el problema PnP.

La salida de los ángulos de giro de la cabeza puede ser umbralizada para decidir si la posición se considera inadecuada o no. Las gráficas de la Figura 3.12 muestran las métricas de confusión en función del umbral utilizado para los dos modelos de estimación de puntos faciales utilizados.

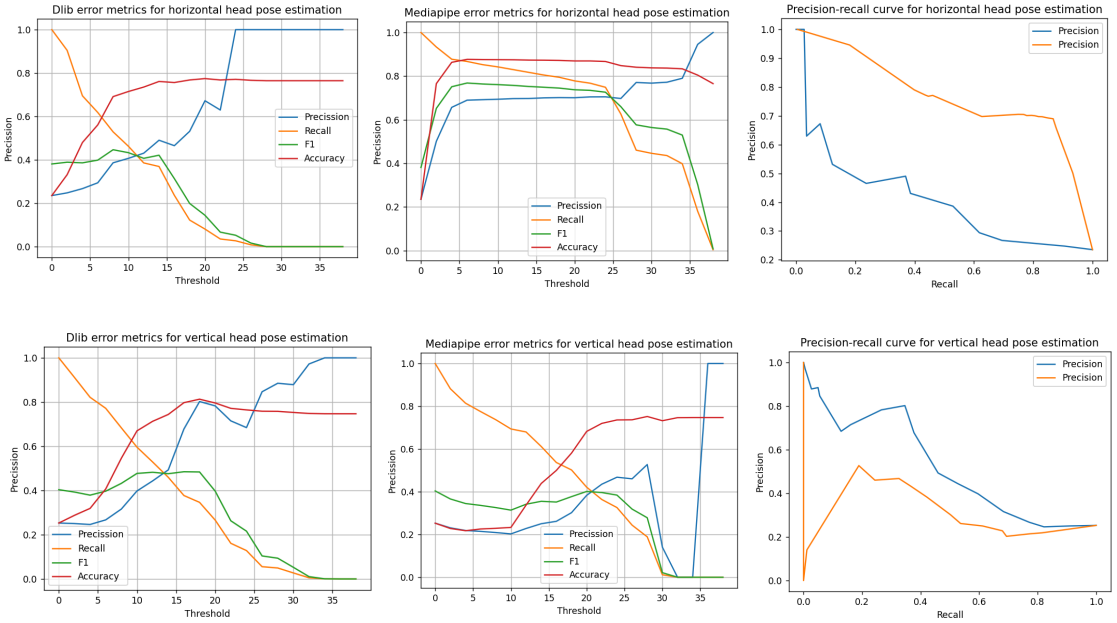


Figura 3.12: Métricas de estimación de postura en función del umbral en umbrales entre 0º y 40º.

Centrando el foco en el eje de rotación, el sistema que utiliza el estimador de Dlib requiere elegir entre una mayor precisión o un mayor *recall*. Por el contrario, utilizando el estimador de MediaPipe existe un amplio umbral en el que ambas métricas mantienen un valor considerablemente bueno. En el eje de flexión tan sólo se consigue un F_1 máximo de 0.4 para ambos modelos, unos resultados mejorables en comparación con los del eje de rotación.

El tiempo de resolución medio del algoritmo PnP con los puntos 14 puntos utilizados es de 1.4 ms.

3.2.6. Movimiento de la boca

En la sección 2.3.3 se han descrito los cálculos del algoritmo implementado para medir la velocidad de movimiento de la boca. A esta salida se le puede aplicar el umbralizado que se crea necesario para determinar si el usuario está hablando o no.

Probando distintos valores de umbral entre 0 y 0.45 en incrementos de 0.05 se obtiene la información de la Tabla 3.14 y la Figura 3.13.

Threshold	0.00	0.5	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45
P	0.48	0.48	0.8	**1.00**	1.00	1.00	1.00	1.00	1.00	1.00
R	1.00	1.00	1.00	**1.00**	0.62	0.05	0.00	0.00	0.00	0.00
F_1	0.65	0.65	0.89	**1.00**	0.76	0.10	0.00	0.00	0.00	0.00
ACC	0.48	0.48	0.88	**1.00**	0.81	0.54	0.52	0.52	0.52	0.52

Tabla 3.14: Métricas de confusión de detección de habla y bostezos en función del umbral.

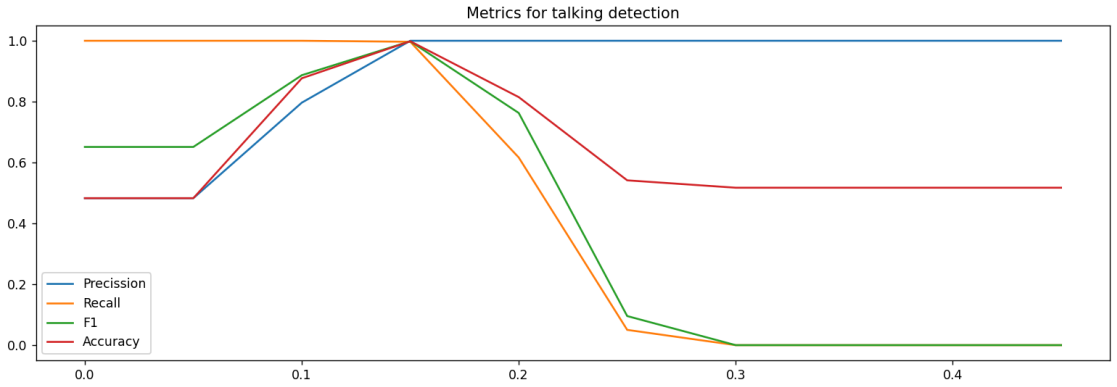


Figura 3.13: Métricas de confusión de detección de habla y bostezos en función del umbral.

En la Figura 3.13 se visualiza gráficamente esta información, donde se puede ver que a medida que aumenta el umbral se realizan menos precisiones pero con mayor confianza, por lo que la precisión del sistema aumenta y el recall disminuye, sin embargo, en el umbral seleccionado se mantiene un valor alto de ambas dando resultados muy acertados.

Utilizando un umbral de 0.15 se obtiene un accuracy cercano a 1 para los datos con los que se ha evaluado el algoritmo.

3.2.7. Apertura de ojos

El EAR calculado como salida del algoritmo de detección de la apertura de los ojos desarrollado en la sección 3.2.7 puede ser sometido a un umbralizado para determinar si el ojo está abierto o no. Para evaluar este algoritmo se utiliza código de validez de ojos del *eye tracker* como *ground truth* usando el fragmento de las grabaciones correspondiente a la tarea 2 del protocolo que consta de una partición balanceada de 10 segundos de ojos abiertos y 10 segundos de ojos cerrados.

En primer lugar se busca encontrar el umbral óptimo para el algoritmo probando distintos valores entre 0.15 y 0.45 en incrementos de 0.05, dando lugar a la información

de la Tabla 3.15 y la Figura 3.14. Una vez seleccionado este valor, se estudia el impacto que tiene que el usuario lleve gafas en el funcionamiento de este algoritmo separando el conjunto de datos según este criterio.

Threshold	0.15	0.20	**0.25**	0.3	0.35	0.4	0.45
P	0.53	0.57	**0.74**	0.85	0.98	1.00	1.00
R	1.00	1.00	**0.94**	0.53	0.20	0.00	0.00
F_1	0.69	0.73	**0.82**	0.65	0.33	0.00	0.00
ACC	0.55	0.62	**0.80**	0.71	0.59	0.49	0.49

Tabla 3.15: Métricas de confusión de detección de ojos abiertos o cerrados. Los valores booleanos positivo y negativo se corresponden con ojos cerrados y ojos abiertos, respectivamente.

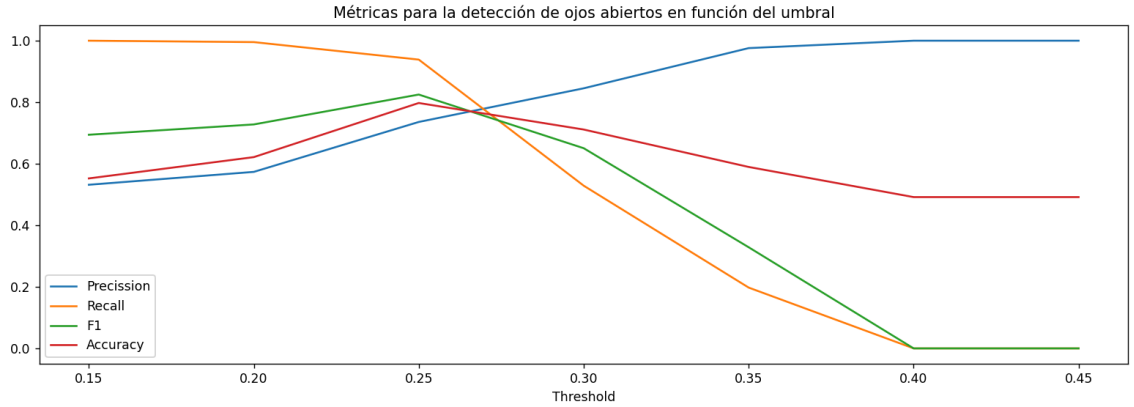


Figura 3.14: Métricas de confusión de detección de apertura de ojos en función del umbral.

	Sin gafas	**Con gafas**
P	0.70	0.81
R	0.98	0.87
F_1	0.82	0.84
ACC	0.78	0.83

Tabla 3.16: Métricas de confusión de detección de ojos abiertos o cerrados para usuarios con y sin gafas. Umbral 0.25.

El rango de umbrales cercano a 0.25 es el de mayor *accuracy* y F_1. En este rango se obtiene un *recall* perfecto manteniendo una precisión del 74%.

Los resultados para usuarios con gafas son incluso mejores que para aquellos con gafas. Esto se da probablemente por el desbalance de los datos. Conforme crece el número de usuarios aumenta la probabilidad de que el umbral no se acomode

perfectamente a las dimensiones de los ojos de todos ellos. Pese a eso, sin necesitar adecuar los parámetros del algoritmo a cada usuario específico se consiguen resultados que cumplen con los objetivos del proyecto, detectando casi la totalidad de ocasiones en las que el usuario cierra los ojos a cambio de algunos falsos positivos.

3.2.8. Comprobación de mirada fuera de pantalla

De forma similar a la evaluación de la estimación de la mirada (sección 3.2.3), se evalúan las métricas de confusión del detector de mirada fuera de pantalla recién calibrado el sistema y unos minutos después (tareas 1 y 5), en las distintas condiciones de iluminación grabadas y para usuarios con y sin gafas.

		P	R	F_1	ACC
Tarea	Tarea 1	0.76	0.44	0.56	0.67
	Tarea 5	0.80	0.32	0.46	0.64
Iluminancia (lx)	150	0.74	0.37	0.50	0.64
	2000	0.82	0.39	0.53	0.66
Gafas?	Sin gafas	0.77	0.40	0.52	0.64
	Con gafas	0.79	0.35	0.49	0.67
Total		0.78	0.38	0.46	0.65

Tabla 3.17: Métricas de confusión de detección de mirada fuera de la pantalla en función de la tarea, iluminancia y gafas.

En general, aunque la precisión del sistema es buena, el *recall* obtenido tiene un amplio rango de mejora. Toma un valor del 44% recién calibrado (tarea 1), que desciende hasta el 32% cuando el usuario se ha movido (tarea 5). Una buena iluminación mejora levemente los resultados aunque no es estrictamente necesaria y no se han encontrado diferencias significativas de rendimiento entre usuarios con y sin gafas.

Capítulo 4

Integración en la plataforma de neurotecnología

A lo largo de este trabajo se han desarrollado algoritmos de monitorización independientes que han sido evaluados por separado de forma *offline*. Como forma de comprobar la viabilidad y el funcionamiento combinado de todos ellos se desarrolla un sistema en C++ que los integra en la plataforma de neurotecnología de Bitbrain, discutiendo los algoritmos seleccionados y comprobando el funcionamiento del sistema en una Surface Go 3.

4.1. Diseño y desarrollo del sistema en C++

De acuerdo con los resultados obtenidos durante la evaluación *offline* de los algoritmos, se ha hecho una valoración con Bitbrain de las tecnologías escogidas para formar parte del sistema final.

En primer lugar, aunque el *framework* de MediaPipe está disponible en C++, no se ofrece una API de soluciones equivalente a la de Python utilizada a lo largo de este trabajo, por lo que un *port* no sería sencillo. Por lo tanto, aunque se ha podido ver su buen rendimiento, se ha descartado esta opción como tecnología utilizada para la detección de rostros y puntos faciales.

El sistema final incorpora el modelo de detección de rostros Yunet, detección de puntos faciales mediante Dlib y el cálculo del centro del ojo mediante el método *ad hoc* para la estimación de la mirada (ver sección 2,2). Los algoritmos de monitorización se configuran con los mejores umbrales encontrados durante la evaluación (ver sección 3.2).

Dado que todos los sistemas de monitorización desarrollados independientemente utilizan y comparten las mismas tres tecnologías base, se combinan todas ellas en un único sistema de monitorización. El bucle de ejecución principal puede separarse en tres etapas (ver Figura 4.1):

1. La primera es la detección de rostros en la imágen. Si no hay, ninguno de los detectores es capaz de generar información por lo que no se procede a la siguiente fase.
2. En caso de encontrar un rostro, se lanza el detector de movimiento y se ejecuta la detección de puntos faciales, cuyo resultado es utilizado para efectuar la estimación de la postura, el movimiento de la boca y la apertura de los ojos.

3. Por último, si el usuario no está movido ni girado y tiene los ojos abiertos, se realiza la estimación de la mirada y la consiguiente comprobación de mirada dentro de los límites de la pantalla.

Cuando cualquiera de los algoritmos detecta un comportamiento incorrecto en el 50% de las últimas w comprobaciones realizadas genera un evento en la plataforma con el código correspondiente al problema detectado. A mayor w más fiables son los eventos generados porque se sabe que perduran en el tiempo, aunque también aumenta el retraso de envío de los eventos.

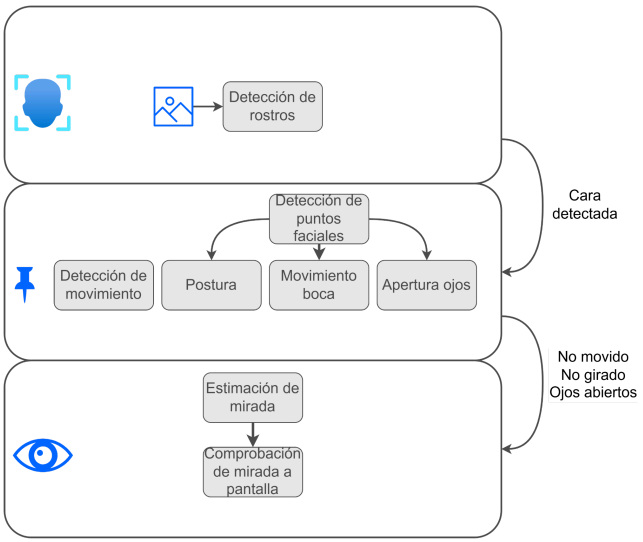


Figura 4.1: Bucle de ejecución del sistema de monitorización completo.

4.2. Integración en la plataforma de Bitbrain

La plataforma de desarrollo es un sistema multiproceso implementado en C++ basado en el patrón publicación/suscripción en el que los distintos procesos se comunican entre sí utilizando sockets TCP/IP (ver Figura 4.1).

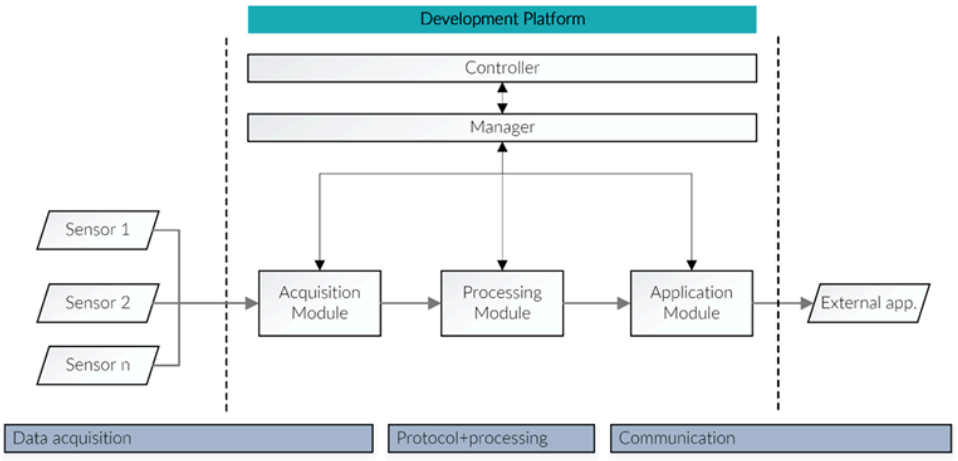


Figura 4.2: Visión general de la arquitectura de la plataforma.

La plataforma utiliza un bucle de funcionamiento a frecuencia constante en el que el módulo de adquisición toma las señales de los sensores, el módulo de procesamiento opera con ellas generando señales y eventos que son utilizados por el módulo de aplicación para proporcionar información al usuario o a una aplicación externa.

La plataforma sigue una estructura jerárquica en la que cada módulo se compone de una o más unidades que se ejecutan secuencialmente utilizando los señales y eventos de las unidades anteriores (ver Figura 4.3).

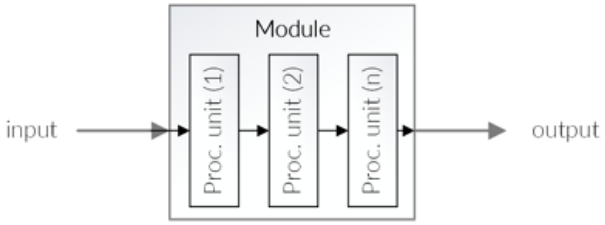


Figura 4.3: Visión general de un módulo compuesto de varias unidades.

La integración del sistema de monitorización en la plataforma de neurotecnología consiste en el desarrollo de una unidad que toma la información de la cámara como si se tratara de cualquier otro sensor y la utiliza para generar eventos relevantes que pueden utilizar las capas de más alto nivel. Para cumplir con los objetivos de rendimiento propuestos, esta unidad ejecuta el sistema de monitorización en un proceso concurrente con una frecuencia de ejecución independiente a la del resto de la plataforma que puede ser configurada.

4.3. Demostración del funcionamiento

El objetivo de esta sección es determinar qué información producida por la unidad desarrollada es relevante para asegurar la correcta realización de cada una de las fases de una sesión y verificar por medio de un pequeño demostrador que se puede alertar al usuario cuando no está realizando correctamente la tarea.

A lo largo del capítulo 3 se ha demostrado la posibilidad de detectar ciertos comportamientos del usuario mediante técnicas de visión por computador. Algunos de dichos comportamientos coinciden con las condiciones de realización de varias etapas de las sesiones de Elevvo. En cada una de estas sesiones se pide al usuario que realice una serie de ejercicios repartidos entre las fases de calibración y ejecución. Las tareas de calibración sirven al sistema para aprender y adaptarse a los patrones de actividad de la persona (su frecuencia alfa individualizada) y el nivel de alfa en el momento de realización de la sesión. Estas tareas son dos:

1. Estado de reposo con ojos cerrados.
2. Tarea de conteo con ojos abiertos, donde la persona tiene que contar mentalmente los cambios de color-saturación de un cuadrado que aparece en la pantalla.

Después de estas dos tareas de calibración, se realizan los ensayos de neurofeedback. En estos ensayos las personas ven el mismo cuadrado de la tarea de conteo (tarea 2 de calibración), pero ahora el color de ese cuadrado cambia en tiempo real con el nivel de su ritmo alfa. De esta forma la persona puede buscar estrategias mentales para conseguir que el cuadrado se ponga de color rojo, lo que significa que está modificando su ritmo alfa, aumentándolo, lo que está relacionado en la literatura con rendimiento cognitivo.

Figura 4.4: Realización de la tarea de neurofeedback.

A efectos prácticos, se pueden dividir los ejercicios descritos en dos grandes grupos: aquellos en los que el usuario debe permanecer relajado con los ojos cerrados, y en los que debe mirar al cuadrado de la pantalla.

- **Durante los ejercicios de ojos cerrados:** Se comprueba que los ojos están efectivamente cerrados. Si a lo largo de los últimos segundos prevalecen las detecciones de ojos abiertos sobre cerrados la tarea se advierte al usuario y se reinicia la tarea.
- **Durante los ejercicios mirando al cuadrado de la pantalla:** Se comprueba que los ojos estén abiertos, mirando al centro de la pantalla y sin girar la cara. De forma similar al ejercicio de ojos cerrados, si se incumple cualquiera de estas condiciones de manera continuada a lo largo de los últimos segundos la tarea se considera inválida, advirtiendo al usuario y reiniciando la tarea.

Independientemente del tipo de tarea, si se detecta una actividad alta de movimientos de la cabeza o de la boca, también se genera una advertencia y se reinicia la tarea.

A modo de ejemplo, la Figura 4.5 muestra las señales del EEG y la IMU del dispositivo durante una sesión. Se puede observar que tras un movimiento (visible en la IMU) el sistema de visión genera un evento con código 1, que corresponde a “usuario movido”. Tras otro movimiento en el que el usuario vuelve a la posición inicial el sistema de visión envía otro evento, esta vez con código 0 que indica que todas las comprobaciones son correctas.

Adicionalmente, la unidad ofrece un modo de depuración en el que se muestra la imágen captada por la cámara junto a información adicional calculada por los distintos algoritmos del sistema (ver Figura 4.6).

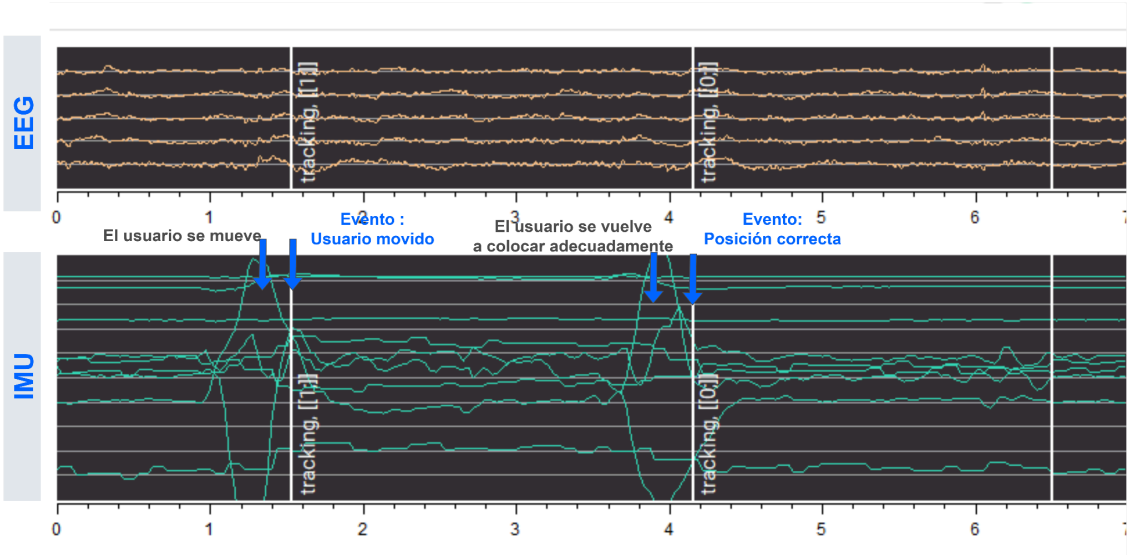


Figura 4.5: Ejemplo de los eventos generados por la unidad de monitorización ante movimiento del usuario.

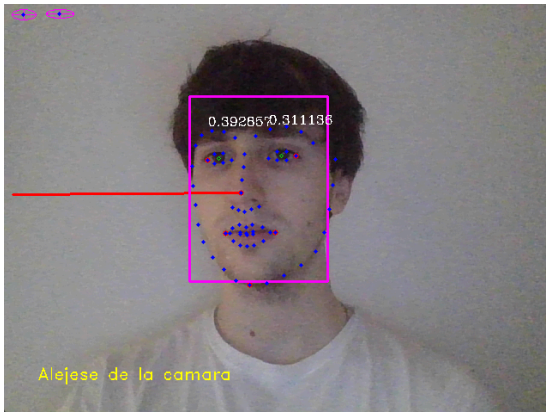


Figura 4.6: Imágen de la unidad de monitorización en modo de depuración.

La unidad puede ser configurada para funcionar a una frecuencia de muestreo determinada de forma que se pueda reducir su consumo de recursos si se considera necesario. El tiempo medio de frame medido en la Surface Go 3 es de 64.34 ms, por lo que se puede alcanzar una frecuencia de hasta 16 fotogramas por segundo.

Capítulo 5
Conclusiones

El resultado del trabajo desarrollado es generalmente satisfactorio de acuerdo a los objetivos previamente establecidos.

Se ha desarrollado un conjunto de algoritmos para monitorizar el movimiento y giro de la cara, el movimiento de la boca, la apertura de la boca y la mirada de los usuarios, construidos sobre los bloques básicos de detección facial, detección de puntos faciales y estimación de mirada.

En cuanto a los bloques básicos, queda más que comprobado el rendimiento y la fiabilidad de los detectores faciales actuales y se ha observado la variedad de posibles aplicaciones de las tecnologías de estimación de puntos faciales. Sobre la estimación de la mirada, sorprende la calidad de los resultados del método *ad hoc* para el cálculo del centroide del iris mediante técnicas tradicionales de visión por computador frente a una solución basada en redes neuronales, con una diferencia media entre ambos resultados menor a 2 píxeles. Sin embargo, la evaluación del algoritmo de estimación de la mirada muestra que tan sólo es suficiente para realizar aproximaciones generales y está lejos de ser una sustitución de los *eye tracker* con hardware especializado.

Se ha diseñado y llevado a cabo un método experimental para la recogida de datos implementando las herramientas necesarias para llevar a cabo esta tarea. Los datos recogidos de 12 usuarios distintos han permitido llevar a cabo una evaluación *offline* de los algoritmos implementados.

Configurados correctamente, la mayoría de los algoritmos de monitorización desarrollados presentan un grado de precisión que cumple los objetivos del proyecto, detectando con un 95% de *accuracy* la presencia de los usuarios y su movimiento, así como cuándo están hablando. La detección de ojos abiertos y de cara girada también alcanza un grado de exactitud cercano al 80%. La comprobación de mirada fuera de pantalla tiene una buena precisión, aunque un *recall* inferior al 50% producido por una alta cantidad de falsos negativos.

Por último, se ha realizado una demostración del funcionamiento de todos los algoritmos de monitorización integrados en un sólo sistema funcionando en conjunto con la plataforma de neurotecnología de Bitbrain, probando que el sistema puede ser ejecutado en tiempo real en dispositivos con pocos recursos computacionales.

5.1. Líneas de trabajo futuras

Analizando los resultados obtenidos durante la evaluación del sistema desarrollado aparecen varias posibles líneas de trabajo. En primer lugar, tras la evaluación *offline* de

los algoritmos realizada en este trabajo, sería interesante una segunda evaluación del sistema completo que se ha integrado en la plataforma de neurotecnología realizando las grabaciones en casa de la población objetivo.

En cuanto a posibles mejoras, pese a que los modelos de estimación de puntos faciales utilizados presentan resultados razonablemente buenos, su ángulo máximo de detección de entre 30º y 45º resulta un factor que limita su funcionalidad. Existen datasets 2D y 3D recientes con rangos de posturas más amplios que podrían ser utilizados para el entrenamiento de un modelo propio de estimación de *landmarking*.

Adicionalmente, aprovechando el hecho de que los usuarios llevan puesta la tecnología de EEG durante las sesiones, se plantea la posibilidad de contrastar la información de la actividad cerebral con la visual para hacer estimaciones más precisas. Esto podría ser evaluado con el mismo dataset recogido para la evaluación del sistema desarrollado en este trabajo.

Por último, aunque el sistema de monitorización desarrollado hace un trabajo destacable en el seguimiento del comportamiento, la comprobación de la correcta colocación de la banda sigue siendo una tarea completamente a criterio del usuario cuando no está acompañado un experto. Por ello sería de gran interés el desarrollo de un sistema (o ampliación del desarrollado en este trabajo) para la detección de la banda y la comprobación de su correcta colocación.

Bibliografía

[1] X. P. Burgos-Artizzu, P. Perona, and P. Dollár, 'Robust face landmark estimation under occlusion', in Proceedings of the IEEE international conference on computer vision, 2013, pp. 1513–1520.

[2] S. R. H. Langton, H. Honeyman, and E. Tessler, 'The influence of head contour and nose angle on the perception of eye-gaze direction', Perception & psychophysics, vol. 66, pp. 752–771, 2004.

[3] A. Nikolaidis and I. Pitas, 'Facial feature extraction and pose determination', Pattern Recognition, vol. 33, no. 11, pp. 1783–1791, 2000.

[4] E. Murphy-Chutorian and M. M. Trivedi, 'Head pose estimation in computer vision: A survey', IEEE transactions on pattern analysis and machine intelligence, vol. 31, no. 4, pp. 607–626, 2008.

[5] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, 'Appearance-based gaze estimation in the wild', in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 4511–4520.

[6] J. D. Fuletra and D. Bosamiya, 'A survey on drivers drowsiness detection techniques', International Journal on Recent and Innovation Trends in Computing and Communication, vol. 1, no. 11, pp. 816–819, 2013.

[7] S. O. H. Madgwick, A. J. L. Harrison, and R. Vaidyanathan, 'Estimation of IMU and MARG orientation using a gradient descent algorithm', in 2011 IEEE international conference on rehabilitation robotics, 2011, pp. 1–7.

[8] S. Madgwick and Others, 'An efficient orientation filter for inertial and inertial/magnetic sensor arrays', Report x-io and University of Bristol (UK), vol. 25, pp. 113–118, 2010.

[9] V. Kazemi and J. Sullivan, 'One millisecond face alignment with an ensemble of regression trees', in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 1867–1874.

[10] T. H. J. Anthonijsz, 'Are high-end and webcam eye trackers comparable?', 2021.

[11] E. Marchand, H. Uchiyama, and F. Spindler, 'Pose estimation for augmented reality: a hands-on survey', IEEE transactions on visualization and computer graphics, vol. 22, no. 12, pp. 2633–2651, 2015.

[12] M. A. Fischler and R. C. Bolles, 'Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography', Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

[13] C. Niewiadomski, R.-J. Bianco, S. Afquir, M. Evin, and P.-J. Arnoux, 'Experimental assessment of cervical ranges of motion and compensatory strategies', Chiropractic & manual therapies, vol. 27, pp. 1–9, 2019.

[14] S. Schwab, 'Las variables temporales en el español de Costa Rica y de España: un estudio comparativo', Revista de Filología y Lingüística de la Universidad de Costa Rica, vol. 41, no. 1, pp. 127–139, 2015.

[15] A. Cruttenden, Gimson's pronunciation of English. Routledge, 2014, pp. 54–55.

[16] J. Trouvain, Tempo variation in speech production: Implications for speech synthesis. Univ. Institut für Phonetik, 2004.

[17] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, '300 faces in-the-wild challenge: The first facial landmark localization challenge', in Proceedings of the IEEE international conference on computer vision workshops, 2013, pp. 397–403.

[18] C. Sagonas, E. Antonakos, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, '300 faces in-the-wild challenge: Database and results', Image and vision computing, vol. 47, pp. 3–18, 2016.

[19] J. Phipps-Nelson, J. R. Redman, D.-J. Dijk, and S. M. W. Rajaratnam, 'Daytime exposure to bright light, as compared to dim light, decreases sleepiness and improves psychomotor vigilance performance', Sleep, vol. 26, no. 6, pp. 695–700, 2003.

[20] L. M. Huiberts, K. C. Smolders, and Y. A. W. de Kort, 'Non-image forming effects of illuminance level: exploring parallel effects on physiological arousal and task performance', Physiology & behavior, vol. 164, pp. 129–139, 2016.

[21] P. Sharma and R. B. Reilly, 'A colour face image database for benchmarking of automatic face detection algorithms', in Proceedings EC-VIP-MC 2003. 4th EURASIP Conference focused on Video/Image Processing and Multimedia Communications (IEEE Cat. No. 03EX667), 2003, vol. 1, pp. 423–428.

[22] S. Yang, P. Luo, C.-C. Loy, and X. Tang, 'Wider face: A face detection benchmark', in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 5525–5533.

[23] J. Deng et al., 'The menpo benchmark for multi-pose 2d and 3d facial landmark localisation and tracking', International Journal of Computer Vision, vol. 127, pp. 599–624, 2019.

[24] R. Hartley and A. Zisserman, Multiple view geometry in computer vision. Cambridge university press, 2003.

[25] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi, 'Revisiting the pnp problem: A fast, general and optimal solution', in Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 2344–2351.

[26] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li, 'Face alignment across large poses: A 3d solution', in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 146–155.

[27] T. Collins and A. Bartoli, 'Infinitesimal plane-based pose estimation', International journal of computer vision, vol. 109, no. 3, pp. 252–286, 2014.

[28] V. Lepetit, F. Moreno-Noguer, and P. Fua, 'EP n P: An accurate O (n) solution to the P n P problem', International journal of computer vision, vol. 81, pp. 155–166, 2009.

[29] Y. Kartynnik, A. Ablavatski, I. Grishchenko, and M. Grundmann, 'Real-time facial surface geometry from monocular video on mobile GPUs', arXiv preprint arXiv:1907. 06724, 2019.

[30] P. Martins and J. Batista, 'Accurate single view model-based head pose estimation', in 2008 8th IEEE International Conference on Automatic Face & Gesture Recognition, 2008, pp. 1–6.

[31] A. Bulat and G. Tzimiropoulos, 'How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks)', in Proceedings of the IEEE international conference on computer vision, 2017, pp. 1021–1030.

[32] M. D. Grossberg and S. K. Nayar, 'What is the space of camera response functions?', in 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings., 2003, vol. 2, p. II–602.

[33] E. Hjelmås and B. K. Low, 'Face detection: A survey', Computer vision and image understanding, vol. 83, no. 3, pp. 236–274, 2001.

[34] C. Dewi, R.-C. Chen, C.-W. Chang, S.-H. Wu, X. Jiang, and H. Yu, 'Eye aspect ratio for real-time drowsiness detection to improve driver safety', Electronics, vol. 11, no. 19, p. 3183, 2022.

[35] W.-S. Sun, C.-L. Tien, J.-W. Pan, T.-H. Yang, C.-H. Tsuei, and Y.-H. Huang, 'Simulation and comparison of the lighting efficiency for household illumination with LEDs and fluorescent lamps', Journal of the Optical Society of Korea, vol. 17, no. 5, pp. 376–383, 2013.

[36] J. Mardaljevic, M. Andersen, N. Roy, and J. Christoffersen, 'Daylighting metrics for residential buildings', Proceedings of the 27th Session of the CIE, 2011.

[37] J.-M. Gutierrez-Martinez, A. Castillo-Martinez, J.-A. Medina-Merodio, J. Aguado-Delgado, and J.-J. Martinez-Herraiz, 'Smartphones as a light measurement tool: Case of study', Applied Sciences, vol. 7, no. 6, p. 616, 2017.

[38] Y.-M. Cheung and Q. Peng, 'Eye gaze tracking with a web camera in a desktop environment', IEEE Transactions on Human-Machine Systems, vol. 45, no. 4, pp. 419–430, 2015.

[39] X. Xiong, Z. Liu, Q. Cai, and Z. Zhang, 'Eye gaze tracking using an RGBD camera: a comparison with a RGB solution', in Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication, 2014, pp. 1113–1121.

[40] S. Suzuki and Others, 'Topological structural analysis of digitized binary images by border following', Computer vision, graphics, and image processing, vol. 30, no. 1, pp. 32–46, 1985.

[41] C. E. P. Machado et al., 'A new approach for the analysis of facial growth and age estimation: Iris ratio', PloS one, vol. 12, no. 7, p. e0180330, 2017.

[42] R. J. Barry, A. R. Clarke, S. J. Johnstone, C. A. Magee, and J. A. Rushby, 'EEG differences between eyes-closed and eyes-open resting conditions', Clinical neurophysiology, vol. 118, no. 12, pp. 2765–2773, 2007.

[43] L. Li, 'The differences among eyes-closed, eyes-open and attention states: an EEG study', in 2010 6th international conference on wireless communications networking and mobile computing (WiCOM), 2010, pp. 1–4.

[44] P. Viola and M. Jones, 'Rapid object detection using a boosted cascade of simple features', in Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001, 2001, vol. 1, p. I–I.

[45] R. Lienhart, A. Kuranov, and V. Pisarevsky, 'Empirical analysis of detection cascades of boosted classifiers for rapid object detection', in Pattern Recognition: 25th DAGM Symposium, Magdeburg, Germany, September 10-12, 2003. Proceedings 25, 2003, pp. 297–304.

[46] Y. Freund and R. E. Schapire, 'A desicion-theoretic generalization of on-line learning and an application to boosting', in European conference on computational learning theory, 1995, pp. 23–37.

[47] N. Dalal and B. Triggs, 'Histograms of oriented gradients for human detection', in 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), 2005, vol. 1, pp. 886–893.

[48] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, 'Support vector machines', IEEE Intelligent Systems and their applications, vol. 13, no. 4, pp. 18–28, 1998.

[49] I. Steinwart and A. Christmann, Support vector machines. Springer Science & Business Media, 2008.

[50] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, 'Joint face detection and alignment using multitask cascaded convolutional networks', IEEE signal processing letters, vol. 23, no. 10, pp. 1499–1503, 2016.

[51] K. He, X. Zhang, S. Ren, and J. Sun, 'Deep residual learning for image recognition', in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[52] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, 'A survey of the recent architectures of deep convolutional neural networks', Artificial intelligence review, vol. 53, pp. 5455–5516, 2020.

[53] J. Gu et al., 'Recent advances in convolutional neural networks', Pattern recognition, vol. 77, pp. 354–377, 2018.

[54] W. Liu et al., 'Ssd: Single shot multibox detector', in Computer Vision--ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11--14, 2016, Proceedings, Part I 14, 2016, pp. 21–37.

[55] W. Wu, H. Peng, and S. Yu, 'Yunet: A tiny millisecond-level face detector', Machine Intelligence Research, vol. 20, no. 5, pp. 656–665, 2023.

[56] J. Wang, Y. Yuan, and G. Yu, 'Face attention network: An effective face detector for the occluded faces', arXiv preprint arXiv:1711. 07246, 2017.

[57] R. Min, A. Hadid, and J.-L. Dugelay, 'Improving the recognition of faces occluded by facial accessories', in 2011 IEEE International Conference on Automatic Face & Gesture Recognition (FG), 2011, pp. 442–447.

[58] A. Ablavatski, A. Vakunov, I. Grishchenko, K. Raveendran, and M. Zhdanovich, 'Real-time pupil tracking from monocular video for digital puppetry', arXiv preprint arXiv:2006. 11341, 2020.

[59] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, 'Multi-pie', Image and vision computing, vol. 28, no. 5, pp. 807–813, 2010.

[60] Y. Yanga, X.-J. Wu, and J. Kittler, 'Landmark weighting for 3dmm shape fitting', arXiv preprint arXiv:1808. 05399, 2018.

[61] A. Kumar, A. Kaur, and M. Kumar, 'Face detection techniques: a review', Artificial Intelligence Review, vol. 52, pp. 927–948, 2019.

[62] C. Zhang and Z. Zhang, 'A survey of recent advances in face detection', 2010.

[63] R.-L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, 'Face detection in color images', IEEE transactions on pattern analysis and machine intelligence, vol. 24, no. 5, pp. 696–706, 2002.

[64] V. Jain and E. Learned-Miller, 'Fddb: A benchmark for face detection in unconstrained settings', UMass Amherst technical report, 2010.

[65] H. A. Rowley, S. Baluja, and T. Kanade, 'Neural network-based face detection', IEEE Transactions on pattern analysis and machine intelligence, vol. 20, no. 1, pp. 23–38, 1998.

[66] F. Ahmad, A. Najam, and Z. Ahmed, 'Image-based face detection and recognition:" state of the art"', arXiv preprint arXiv:1302. 6379, 2013.

[67] K. C. Yow and R. Cipolla, 'Feature-based human face detection', Image and vision computing, vol. 15, no. 9, pp. 713–735, 1997.

[68] O. Déniz, G. Bueno, J. Salido, and F. De la Torre, 'Face recognition using histograms of oriented gradients', Pattern recognition letters, vol. 32, no. 12, pp. 1598–1603, 2011.

[69] C. Lugaresi et al., 'Mediapipe: A framework for building perception pipelines', arXiv preprint arXiv:1906. 08172, 2019.

[70] 'GitHub - onnx/onnx: Open standard for machine learning interoperability --- github.com'. [Online]. Available: https://github.com/onnx/onnx. [Accessed: 16-Jun-2024].

[71] E. C. of Optometry and Optics, 'Blue Book', 2020. [Online]. Available: https://ecoo.info/ecoo-blue-book/. [Accessed: 16-Jun-2024].

[72] A. Papoutsaki, 'Scalable webcam eye tracking by learning from user interactions', in Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems, 2015, pp. 219–222.

[73] A. Papoutsaki, J. Laskey, and J. Huang, 'Searchgazer: Webcam eye tracking for remote studies of web search', in Proceedings of the 2017 conference on conference human information interaction and retrieval, 2017, pp. 17–26.

[74] K. Wisiecka et al., 'Comparison of webcam and remote eye tracking', in 2022 Symposium on eye tracking research and applications, 2022, pp. 1–7.

[75] G. Casiez, N. Roussel, and D. Vogel, '1€ filter: a simple speed-based low-pass filter for noisy input in interactive systems', in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2012, pp. 2527–2530.

[76] I. N. de Estadística, 'Encuesta Continua de Hogares', 2020. [Online]. Available: https://www.ine.es/prensa/ech_2020.pdf. [Accessed: 16-Jun-2024].

[77] S. Abtahi, B. Hariri, and S. Shirmohammadi, 'Driver drowsiness monitoring based on yawning detection', in 2011 IEEE International Instrumentation and Measurement Technology Conference, 2011, pp. 1–4.

[78] Z. Jie, M. Mahmoud, Q. Stafford-Fraser, P. Robinson, E. Dias, and L. Skrypchuk, 'Analysis of yawning behaviour in spontaneous expressions of drowsy drivers', in 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), 2018, pp. 571–576.

[79] F. Hasan and A. Kashevnik, 'State-of-the-art analysis of modern drowsiness detection algorithms based on computer vision', in 2021 29th Conference of Open Innovations Association (FRUCT), 2021, pp. 141–149.

[80] H. Garg, 'Drowsiness detection of a driver using conventional computer vision application', in 2020 international conference on power electronics & IoT applications in renewable energy and its control (PARC), 2020, pp. 50–53.

[81] W. Zhang, B. Cheng, and Y. Lin, 'Driver drowsiness recognition based on computer vision technology', Tsinghua Science and Technology, vol. 17, no. 3, pp. 354–362, 2012.

[82] S. Baul, M. R. Rana, and F. B. Alam, 'A Real-time Light-weight Computer Vision Application for Driver's Drowsiness Detection'.

[83] Y. Wu and Q. Ji, 'Facial landmark detection: A literature survey', International Journal of Computer Vision, vol. 127, no. 2, pp. 115–142, 2019.

[84] K. Khabarlak and L. Koriashkina, 'Fast facial landmark detection and applications: A survey', arXiv preprint arXiv:2101. 10808, 2021.

[85] M. Kok, J. D. Hol, T. B. Schön, F. Gustafsson, and H. Luinge, 'Calibration of a magnetometer in combination with inertial sensors', in 2012 15th International Conference on Information Fusion, 2012, pp. 787–793.

[86] M. Kok and T. B. Schön, 'Magnetometer calibration using inertial sensors', IEEE Sensors Journal, vol. 16, no. 14, pp. 5679–5689, 2016.

[87] J. M. G. Merayo, P. Brauer, F. Primdahl, J. R. Petersen, and O. V. Nielsen, 'Scalar calibration of vector magnetometers', Measurement science and technology, vol. 11, no. 2, p. 120, 2000.

[88] Z. Zhang, 'A flexible new technique for camera calibration', IEEE Transactions on pattern analysis and machine intelligence, vol. 22, no. 11, pp. 1330–1334, 2000.

[89] R. Tsai, 'A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses', IEEE Journal on Robotics and Automation, vol. 3, no. 4, pp. 323–344, 1987.

[90] The Bitbrain. Team, 'Modern BCI-based Neurofeedback for Cognitive Enhancement|Bitbrain---bitbrain.com'. [Online]. Available: https://www.bitbrain.com/blog/neurofeedback.

[91] 'OpenCV: Perspective-n-Point (PnP) pose computation --- docs.opencv.org'. [Online]. Available: https://docs.opencv.org/4.x/d5/d1f/calib3d_solvePnP.html.

[92] 'Cognitive Enhancement Lab | Bitbrain --- bitbrain.com'. [Online]. Available: https://www.bitbrain.com/neurotechnology-products/complete-solutions/cognitive-enhancement-lab.

[93] T. B. Fitzpatrick, 'The validity and practicality of sun-reactive skin types I through VI', Archives of dermatology, vol. 124, no. 6, pp. 869–871, 1988.

[94] J. H. Gruzelier, 'EEG-neurofeedback for optimising performance. I: A review of cognitive and affective outcome in healthy participants', Neuroscience & Biobehavioral Reviews, vol. 44, pp. 124–141, 2014.

[95] 'i·bug - resources - 300 Faces In-the-Wild Challenge (300-W), ICCV 2013 --- ibug.doc.ic.ac.uk'. [Online]. Available: https://ibug.doc.ic.ac.uk/resources/300-W/.

[96] 'SPDX License List | Software Package Data Exchange (SPDX) --- spdx.org'. [Online]. Available: https://spdx.org/licenses/.

Anexos

Anexo 1. Seguimiento del protocolo de grabación

Sujeto		Edad	
Condición de luz		Sexo	
Tipo de piel (Escala fitzpatrick)			
Otras características			

Tarea	T (s)	Tini (s)	Comentarios
Calibración mirada	30		
Comprobación mirada en pantalla	30		
Comprobación mirada fuera de pantalla: L, R, U, D.	40		
Mirar al frente, ojos abiertos	10		
Ojos cerrados	10		
Giro izquierda	10		
Giro centro	10		
Giro derecha	10		
Giro centro	10		
Giro arriba	10		
Giro centro	10		
Giro abajo	10		
Giro centro	10		
Posición central	30		
Desplazamiento a la izquierda	10		
Desplazamiento a la derecha	10		
Desplazamiento adelante	10		
Desplazamiento atrás	10		
Hablar	30		
Comprobación mirada en pantalla	30		
Comprobación mirada fuera de pantalla: L, R, U, D	40		
Quitar la banda	10		
Salir del cuadro de imagen	10		

Tabla A1.1: Tabla de seguimiento del protocolo de grabación.

Comentarios antes de comenzar la sesión

Para evitar confusiones durante la grabación del protocolo se hacen algunos apuntes y aclaraciones al sujeto antes de comenzar:

- Puesta en contexto general del estudio que se está realizando, las tareas que se van a realizar y el tiempo total de la grabación.
- Diferencias entre los verbos mirar, mover y girar.
    - Mirar: Quieto, moviendo los ojos.
    - Mover: Mirando a la pantalla, moverse en la dirección indicada sin salir del cuadro de imagen.. Se puede utilizar la silla con ruedas para realizar los movimientos.
    - Girar: Rotar el cuello sin mover el cuerpo. Los giros se realizan lentamente y hasta el límite (cómodo) de movimiento. Una vez alcanzado el límite, permanecer en esa hasta ser notificados.
- Se pueden hacer preguntas si hay alguna duda durante la realización de las tareas pero se pide no hablar en la medida de lo posible.
- Aviso de que la primera comprobación de ojos se realiza nada más empezar la grabación.

Pasos previos a la grabación.

1. En primer lugar se muestra la cámara para que adopten una posición cómoda y adecuada, centrando el rostro en la pantalla.
2. Se pide que sigan los pasos necesarios para la colocación de la banda.
    a. Retirar el pelo de la frente y las orejas.
    b. Limpiar las zonas de la piel que entrarán en contacto con los sensores de la banda con una toallita húmeda que se proporciona.
    c. Colocar la banda correctamente, centrando en el eje vertical la marca central de la banda con la nariz, con los sensores en contacto con la piel, apoyada sobre las orejas.
3. Calibración del *eye tracker*. Realizar el proceso de calibración, consistentes en seguir con la mirada una serie de puntos que aparecen uno a uno en el centro y esquinas de la pantalla, y después comprobar la calidad de la calibración. Si la calibración es mala, mover ligeramente al sujeto y repetir este paso.
4. Comprobación de la señal de EEG. Una vez conectada la banda, se comprueba la calidad de las señales de los 5 canales de la banda, si la señal es mala, corregir o recolocar completamente la banda.
5. Comenzar la grabación mirando al centro de la pantalla. La posición inicial se toma como referencia para el cálculo del giro absoluto mediante la integración de las señales de la IMU.

Anexo 2. Calibración de la cámara

La calibración de cámara es un proceso fundamental en el ámbito de la visión por computador cuyo objetivo es determinar los parámetros internos o intrínsecos y los externos o extrínsecos de la cámara. Estos parámetros permiten corregir distorsiones de cámara así como la reconstrucción de escenas 3D a partir de las imágenes 2D captadas.

- Los parámetros intrínsecos describen las propiedades de la cámara, como la distancia focal, el centro óptico y los coeficientes de distorsión radial y tangencial.
- Los parámetros extrínsecos definen la posición y orientación de la cámara en el sistema de coordenadas del mundo real, habitualmente utilizando el tablero de calibración como origen de coordenadas.

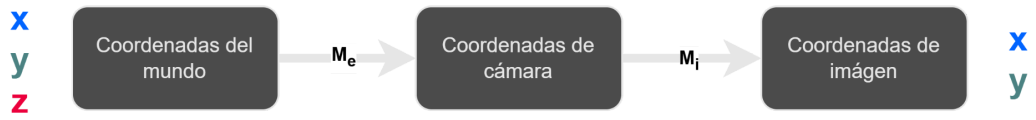


Figura A2.1: Transformación de coordenadas del mundo a coordenadas de imágen.

Los parámetros extrínsecos se definen en forma de matrices rotación *R* y traslación *T*. Se pueden combinar en una sóla matriz extrínseca $M_e=[R|T]$. Los extrínsecos son el vector de cuatro componentes de distorsiones radial y tangencial en los dos ejes, y la matriz de la cámara conteniendo el centro óptico *c* y la distancia focal *f* en una misma matriz M_i.

$$M_e = \begin{bmatrix} R_{3x3} & T_{3x1} \\ 0_{1x3} & 1 \end{bmatrix} \qquad M_i = \begin{bmatrix} f_x & 0 & c_x \\ 0 & fy & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

Habitualmente, la calibración se realiza reconociendo en imágenes tomadas por la cámara patrones conocidos en forma de tablero, siendo los más conocidos los métodos de Tsai [89] y Zhang [88]. Se toman 15 imágenes del patrón de la Figura A2.2 con la cámara interior de la Surface, a la misma resolución que se toman en el sistema final.

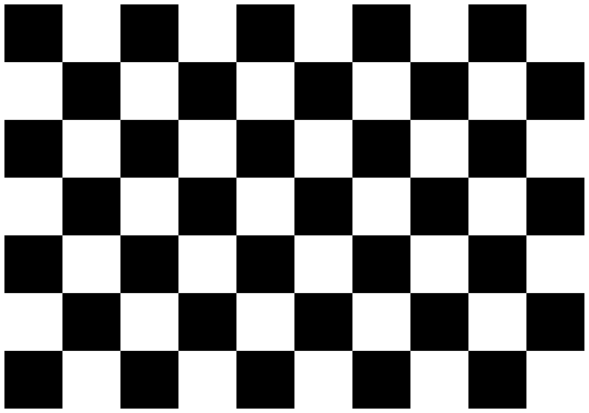


Figura A2.2: Patrón de calibración utilizado.

Dado que todos los dispositivos utilizados incorporan la misma webcam, se asume que comparten los mismos parámetros ya que el pequeño rango de error individual no resulta crítico en las aplicaciones en los que se están usando los parámetros de la cámara,

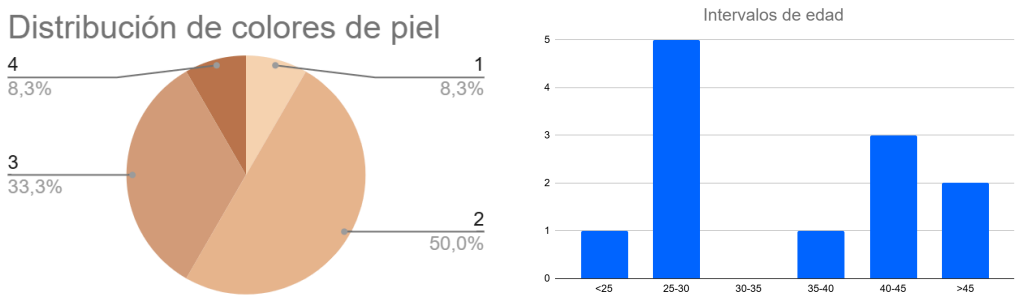
Anexo 3. Variedad de la muestra de sujetos

Este anexo está dedicado al estudio de la muestra de sujetos participantes en el conjunto de datos recogido en este trabajo. Todos los sujetos utilizados para las grabaciones han sido voluntarios de Bitbrain.

En la tabla A3.1 se muestra la desglosada de cada uno de los sujetos.

Sujeto	P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12
Edad	39	43	24	46	29	25	26	28	28	41	42	49
Sexo (H/M)	H	M	M	H	M	H	M	H	M	M	H	H
Tipo de piel (1-7)	2	2	3	2	4	3	2	2	1	2	3	3
Gafas (S/N)	N	N	N	S	N	S	N	S	N	N	S	N

Tabla A3.1: Perfil de los sujetos grabados.



(a) Color de piel (b) Edad

Figura A3.1: Distribución de color de piel y edad de los sujetos grabados.

De los 12 sujetos utilizados, 4 de ellos llevaban gafas durante la grabación. En especial, el sujeto 4 utiliza gafas con filtro de azules que provocan reflejos más notables. Todos los sujetos están entre el valor 1 y el 4 en la escala Fitzpatrick, faltando muestra de la mitad más oscura de la escala, no es una muestra suficiente para hacer estudios significativos. Algo similar ocurre con el rango de edades de la muestra, todos los sujetos tienen entre 24 y 49 años. En cuanto a sexo, la muestra está perfectamente balanceada entre hombres y mujeres, con 6 sujetos en cada grupo.

Anexo 4. Herramientas utilizadas

La implementación de los distintos algoritmos desarrollados Python 3.11, utilizando como librerías principales OpenCV 4.9 para las tareas de visión por computador, NumPy 1.26, Pandas 2.2.1 para las operaciones y el manejo de datos y matplotlib 3.8.3 para la visualización de resultados.

Se ha utilizado Python también para las herramientas auxiliares desarrolladas como postprocesado de los vídeos para ponerlos a tiempo real, la sincronización de las medidas de los algoritmos con las de la plataforma y la evaluación de los algoritmos.

El programa de grabación está implementado en C++ como un módulo integrado en la plataforma de neurotecnología, al igual que el sistema final de monitorización.

Se ha utilizado un repositorio de Gitlab para el control de versiones y la compartición de código con Bitbrain.

Uno de los requisitos definidos en la sección 2.1 es el de limitar las tecnologías utilizadas a aquellas de software libre en la medida de lo posible. En la Tabla A4.1 se muestra un resumen de las librerías y modelos utilizados junto con las licencias a las que están sujetas.

Tecnología	SPDX ID
OpenCV	Apache-2.0
Numpy	BSD-3-Clause
Pandas	BSD-3-Clause
Matplotlib	PSF-2.0
Dlib	BSL-1.0
MediaPipe	Apache-2.0
MTCNN	MIT
AHRS	Apache-2.0
face_detecion_yunet_2023mar.onnx	Apache-2.0
res10_300x300_ssd_iter_140000	MIT
shape_predictor_68_face_landmarks.dat	El dataset 300-W en el que está entrenado prohíbe uso comercial [95].

Tabla A4.1: Licencias de las tecnologías utilizadas.

Todos las las tecnologías a excepción del modelo utilizado para detección de puntos faciales son software libre que permiten uso comercial. Más información de cada una en [96].

Anexo 5. Etiquetado de datos

En la evaluación de los algoritmos de monitorización es necesario conocer los fragmentos de tiempo en los que se sabe con certeza que el usuario está realizando cada tarea. Para esto se revisan los vídeos mientras se etiqueta manualmente cada una de las tareas del protocolo, apuntando su tiempo de inicio y final, la etiqueta y su valor en un fichero *.json* que acompañará a los datos grabados.

En ocasiones, además de la tarea, se apunta la subtarea que se está realizando u otra información relevante. En la Tabla A5.1 se muestran las etiquetas aplicadas a cada vídeo.

Etiqueta	Valores
task	1, 2, 3, 4, 5
sub_task	1_points_screen, 1_look_away, 4_center, 4_moving, 4_talking, 5_points_screen, 5_look_away, out_of_screen
bad_horizontal	0, 1
bad_vertical	0,1
in_screen	0,1
talking	0,1

Tabla A5.1: Etiquetas aplicadas a las grabaciones. Nota: Aunque la información de las etiquetas *in_screen* y *talking* es redundante se mantiene por motivos de comodidad durante las evaluaciones.

Anexo 6. Evaluaciones realizadas

A modo de resumen, la Tabla A6.1 muestra una lista de las pruebas realizadas para la evaluación de los algoritmos implementados, las razones de su realización y los datos utilizados para ello (*ground truth* y tareas del protocolo).

Descripción	Motivación	*Ground truth*	Tareas
Detección facial			
Métricas de confusión de los modelos utilizados	Selección del modelo	Etiquetado	*
Accuracy en función de la iluminación.		Etiquetado	
Accuracy en función de accesorios.		Etiquetado	
Tiempo de ejecución medio.		-	
Detección de puntos faciales			
Precisión de la estimación en función del ángulo de rotación.	Selección del modelo.	-	-
Tiempo de ejecución medio.		-	*
Estimación de mirada			
Diferencia entre centro del iris calculado por método ad-hoc y MediaPipe.	Selección del modelo.	Eye tracker	1, 5
Métricas de error tareas 1 y 5.	Comprobar efecto del movimiento en la estimación	Eye tracker	
Métricas de error en función de la iluminación..	Comprobar robustez a oclusiones	Eye tracker	
Métricas de error en función de accesorios.		Eye tracker	
Tiempo de ejecución medio.	Selección del modelo.	-	
Detección de movimiento			
Métricas de confusión en función del threshold.	Selección del threshold.	Etiquetado	4
Estimación de postura			
Métricas de error de los modelos Dlib y MediaPipe.	Selección de modelo.	IMU	3
Rotación estimada frente a real.	Comprobar ángulos límite de ajuste de los modelos.	IMU	
Métricas de confusión en función del threshold.	Selección del threshold.	Etiquetado	
Tiempo de ejecución medio.		-	
Movimiento de boca			
Métricas de confusión en función del threshold.	Selección del threshold.	Etiquetado	4
Apertura de ojos			
Métricas de confusión en función del threshold.	Selección del threshold.	Etiquetado	2
Métricas de confusión en función de accesorios.	Comprobar robustez a oclusiones.		
Comprobación de mirada fuera de pantalla			
Métricas de confusión tareas 1 y 5.	Comprobar el efecto del movimiento en la estimación.	Etiquetado	1, 5

Métricas de confusión en función de la iluminación.	Comprobar robustez a oclusiones.	Etiquetado	
Métricas de confusión en función de accesorios.		Etiquetado	

Tabla A6.1: Resumen de las evaluaciones realizadas sobre el conjunto de datos adquirido.