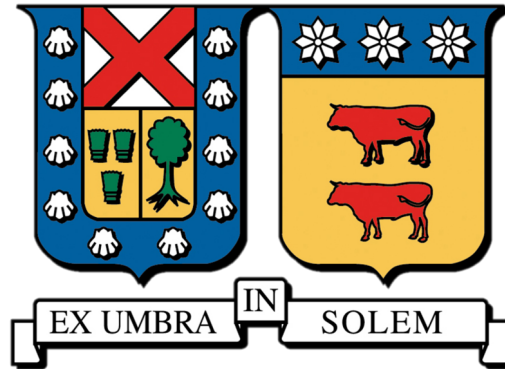


UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE ELECTRÓNICA
VALPARAÍSO - CHILE



“Modular design and development of heart rate recognition and attention level, to enhance emotion recognition in remote oncology consultations”

Sebastián Israel Araneda Soto

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL
TELEMÁTICO

PROFESOR GUÍA:

Nicolás Torres Rudloff

PROFESOR CORREFERENTE:

Marcos Zúñiga Barraza

Enero - 2023

Agradecimientos

Me gustaría agradecer a mi profesor guía, el Sr. Nicolás Torres por su gran ayuda y apoyo en este proyecto. Por haberme dado la oportunidad de comprender nuevas áreas de las tecnologías, y su máxima ayuda para guiarme en este documento.

También quiero agradecer a Javiera Barrientos, médica cirujana que ha estado presente durante toda mi etapa universitaria, quien ha sido mi ánimo para seguir adelante en momentos difíciles, a quien fue mi compañera en este equipo que hemos formado al iniciar nuestras carreras profesionales, un equipo para impulsarnos mutuamente en este difícil viaje. Menciono también a Marcela, quien me ayudó a revisar el documento, y por ayudarme con su profesionalismo y afecto durante la última etapa de este escrito.

Por último, Carlos, Camila y Catalina que supieron estar presentes en los últimos momentos de mi carrera y acompañarme a pesar de la distancia.

Este documento está dedicado al área de investigación de la salud, para motivar su trabajo por crear nuevos métodos y técnicas para dejar atrás nuestras dañinas costumbres, y optar por el máximo provecho de las tecnologías.

Resumen Ejecutivo

Si bien las tecnologías de la información se encuentran en constante crecimiento, durante los años 2019 y 2020 se han vivido avances explosivamente crecientes. La razón se debe a la necesidad que tuvo la sociedad a enfrentarse a un ambiente remoto y la disminución del contacto presencial entre personas, un distanciamiento que dio lugar a un cambio que ha afectado en diferentes aspectos. Por consiguiente, el área de la salud y sus pacientes no se han vistos exentos de este fenómeno.

Los efectos del distanciamiento afectan directamente en los resultados de una consulta, debido a que el factor de contacto físico y no verbal entre paciente y personal de la salud está ausente, lo que interfiere con la buena relación de inter pares. Particularmente, esto ocurre también en consultas oncológicas, que es el foco de solución al que apunta este trabajo.

El presente proyecto pretende desarrollar un software informático, que permite disipar el problema del distanciamiento entre médico-paciente en consultas oncológicas, gracias a la implementación de la detección de emociones basado en computer vision. El software se ejecuta en tiempo real mientras la teleconsulta está en proceso, utilizando solo la imagen que la cámara del paciente emite hacia el servidor. El proyecto propone que este software sea potenciado gracias a tres módulos independientes: Detección de seis emociones básicas (tristeza, ira, felicidad, miedo, angustia e indiferencia), detección de ritmo cardiaco y detección del nivel de atención del paciente.

Por otra parte, este documento indaga en el desarrollo e implementación de los módulos de ritmo cardiaco, utilizando técnicas clásicas en el estado del arte de fotoplethysmografía remota; y el nivel de atención, basándose en una expresión matemática que modela una cuantificación de la atención de una persona, cuyos factores serían la rotación cefálica en tres ejes (yaw, pitch y roll), bostezos, parpadeos y vista fija.

Palabras Clave: Teleconsulta, software, computer vision, fotoplethysmografía remota

Abstract

While information technologies are constantly growing, during the years 2019 and 2020 there have been explosively increasing advances. The reason is due to the need that society had to face a remote environment and the decrease in face-to-face contact between people, a distancing that led to a change that, for different types of people, has affected in different ways. Consequently, the health sector and its patients have not been exempt from this phenomenon.

The effects of distancing directly affect the results of a consultation because the factor of physical and non-verbal contact between patient and health staff is absent, which interferes with the good relationship of peers. Particularly, this also happens in oncological consultations, which is the focus of this study.

This project aims to develop a computer software that allows to resolve the problem of distance between doctor and patient in oncology consultations, thanks to the implementation of emotion detection based on computer vision. The software runs in real time while the teleconsultation is in progress, using only the image that the patient's camera emits to the server. The project proposes that this software be enhanced by three independent modules: six basic emotions detection (sadness, anger, happiness, fear, anguish and indifference), heart rate detection and patient's level of focus detection.

On the other hand, this paper investigates the development and implementation of heart rhythm modules, using classic techniques in the state of the art of remote photoplethysmography; and the level of attention, based on a mathematical expression that models a quantification of a person's attention, whose factors would be the head rotation in three axes (yaw, pitch and roll), yawning, blinking and fixed vision.

Keywords: Teleconsultation, software, computer vision, remote photoplethysmography

Glosario

Computer Vision Traducido al español como “Visión por computador”, es la disciplina que busca producir información numérica o simbólica, por medio de métodos para adquirir, procesar, analizar y comprender imágenes del mundo real.

fotopletismografía Técnica óptica ampliamente utilizada en clínica para la monitorización periférica de la frecuencia cardíaca, respiración y presión sanguínea.

frame Anglosajismo de “fotograma”. Imagen extraída de un video.

frames Anglosajismo en plural de “fotogramas”. Conjunto de imágenes contiguas provenientes de un video.

framework En ingeniería de software, se refiere a un conjunto de herramientas que ayudan al desarrollador para construir aplicaciones y sistemas digitales.

Ground Truth Información que se sabe que es real o verdadera, útil para verificar o evaluar los resultados de un algoritmo con respecto a la realidad.

landmark Atributos esenciales de un rostro humano que nos permiten distinguir entre diferentes rostros, en forma de puntos ubicados en lugares claves.

pipeline Es una cadena de elementos agrupados de manera que la salida de uno sea la entrada del siguiente.

pletismografía Técnica biomédica para obtener los cambios de volumen y presión en distintas partes del cuerpo. Se utiliza para medir parámetros orientados a problemas pulmonares y cardiovasculares.

Python Lenguaje de programación de alto nivel, destinado al uso de propósito general.

red neuronal convolucional Tipo de red neuronal. Las neuronas artificiales corresponden a campos receptivos que se asimilan a las neuronas de la corteza visual de un cerebro humano.

reutilización de código Técnicas que garantizan que una parte o la totalidad de un programa informático existente se pueda emplear en la construcción de otro programa.

SSH Se refiere a Secure Shell, un protocolo de comunicación remota entre un servidor y un cliente, por medio de un canal en el que toda la información está cifrada.

teleconsulta Referencia a una consulta médica remota. Interacción virtual entre paciente y personal de la salud, mediante el uso de plataformas telemáticas.

telepresencia Interacción virtual entre personas, a travez de la tecnología.

webcam Anglocismo referente a una cámara web. Cámara de vídeo miniaturizada que se puede conectar a un ordenador para grabar imágenes o emitirlas en directo a través de Internet.

Siglas

API Application Programming Interfaces.

AtMod Módulo de nivel de atención.

BPM Beats Per Minute.

BVP Blood Volume Pulse.

CarMod Módulo de ritmo cardíaco.

CV Visión por Computador.

EMod Módulo de emociones.

FALP Fundación Arturo López Pérez.

HR Heart rate.

IPP Primitivas de Rendimiento Integradas.

MAE Mean Absolute Error.

ML Machine Learning.

OMS Organización Mundial de la Salud.

OpenCV Open Source Computer Vision Library.

POS Plane-Orthogonal-to-Skin.

PPG Photoplethysmography.

PyVHR Python framework for Virtual Heart Rate.

RF Requisitos funcionales.

RNF Requisitos no funcionales.

rPPG Remote Photoplethysmography.

TDAH Trastorno por Déficit de Atención e Hiperactividad.

TICs Tecnologías de Información y Comunicación.

Tabla de Contenido

1	Introducción	1
1.1	Problema	1
1.2	Acercamiento a la solución	1
1.3	Objetivos	2
1.3.1	Objetivo General	2
1.3.2	Objetivos Específicos	3
1.3.3	Herramientas utilizadas	3
1.4	Estructura del Documento	4
2	Marco Teórico	5
2.1	¿Qué es una teleconsulta?	5
2.1.1	Requisitos de una teleconsulta	5
2.2	Relación médico-paciente	6
2.2.1	Empoderamiento del paciente	6
2.2.2	Oncología y emocionalidad	7
2.3	Rítmico cardíaco	7
2.4	Fotopleitismografía	7
2.4.1	Fotopleitismografía Remota	8
2.5	Visión por computador	8
3	Estado del Arte	11
3.1	Fotopleitismografía remota	11
3.2	Detección de atención	15
3.3	Posibles Datasets	15
4	Diseño de la Arquitectura del sistema	18
4.1	Requisitos funcionales	18
4.2	Requisitos no funcionales	19

ÍNDICE DE FIGURAS

4.3	Diagrama de Contexto	20
4.4	Diagrama de Arquitectura	21
5	Desarrollo de los módulos	22
5.1	Entorno de desarrollo.	22
5.2	Módulo: Detección de Ritmo Cardíaco	23
5.2.1	Paso 1: Captura de imagen y requisitos de código.	24
5.2.2	Paso 2: Extracción de rostro	26
5.2.3	Paso 3: Procesamiento de señal RGB	30
5.2.4	Paso 4: Métodos de rPPG, obtención de BVP	31
5.2.5	Paso 5: Estimación de BPM	34
5.2.6	¿Qué método rPPG utilizar?	35
5.2.7	Rendimiento	38
5.3	Módulo: Detección de Nivel de Atención	42
5.3.1	Métricas de cuantificación atencional	42
5.3.2	Expresión matemática propuesta	47
6	Resultados	50
6.1	CarMod	50
6.2	AtMod	53
7	Verificación y Validación	55
8	Conclusiones	57

Referencias

Índice de figuras

2.1	Esquema CV	9
4.1	Diagrama de contexto.	20

ÍNDICE DE TABLAS

4.2	Diagrama de arquitectura.	21
5.1	Comparación de dos métodos implementados en pyVHR para la extrac- ción facial.	27
5.2	Landmarks posicionados y enumerados de FaceMesh.	28
5.3	Arquitectura del banco de pruebas.	36
5.4	MAE vs actividad. Un gráfico por cada método.	39
5.5	MAE general por actividad.	40
5.6	Comparativa de métodos en tiempo real	40
5.7	Landmarks posicionados y enumerados de FaceMesh.	44
5.8	Líneas que proyectan las distancias entre los labios.	45
5.9	Líneas que proyectan las distancias entre los párpados, por cada ojo. . .	46
5.10	Imagen de ejemplo de OCAT.	47
6.1	Carmod y emociones	51
6.2	Carmod y emociones, calculando BPM.	53
6.3	AtMod visible e invisible a	53
6.4	AtMod visible e invisible b	54
6.5	AtMod visible e invisible c	54

Índice de tablas

1	MAE por actividad, de cada método rPPG.	50
2	MAE por cada persona y actividad.	52

1. Introducción

En la actualidad existen diversas herramientas de software útiles para reuniones virtuales. Las consultas de telemedicina oncológica no se quedan exentas de su uso, facilitando esta actividad para las personas que no pueden asistir presencialmente a la sesión. Como es de esperar, una consulta médica posee muchos factores que no se encuentran presentes dentro de la telemedicina, siendo uno de estos la interpretación de las emociones del paciente. Con este fin, esta investigación se centra en la utilización de algoritmos de machine learning para detectar seis emociones principales en el paciente; alegría, enfado, miedo, tristeza, sorpresa e indiferencia, además de determinar el nivel de atención del paciente. Considerando lo anterior, la detección de ritmo cardíaco es otro factor para potenciar la interpretación emocional. De esta forma, se busca mejorar la relación paciente-médico, generando un entendimiento emocional más integral hacia el paciente, generando un entendimiento como individuo.

1.1. Problema

Los profesionales de la salud, médicos oncológicos en particular, no tienen un contacto presencial con el paciente en las consultas telemáticas, por lo que, interpretar sus emociones se hace una tarea complicada por estar limitada al uso de una cámara y un micrófono, y por esto se debe hacer un esfuerzo mayor para comprender y empatizar con el paciente, para solventar y llevar de mejor manera la teleconsulta se plantea el sistema en tiempo real de detección de emociones del paciente.

1.2. Acercamiento a la solución

Considerando que el paciente tiene en su poder durante la consulta, una cámara para su captura de video, el acercamiento a la solución está enfocado en computer vision: un sistema que señala, en el computador del médico, el porcentaje de cada emoción que está sintiendo el paciente en tiempo real, además, de su nivel de atención y su ritmo

cardíaco.

Existen diversos trabajos previos relacionados a la biometría que responden a algunas de las necesidades del problema. Unos ejemplos de ello son:

- La pletismografía es un método que podría ser utilizado para lograr parte de los objetivos, ya que podría obtener información acerca de los cambios de ritmo cardiaco en base a los colores faciales.
- Reconocimiento de expresiones faciales para emociones neutrales, alegría, tristeza y enojo: Emotion Detection y Deepface.
- Reconocimiento de entonación vocal para la detección de estado anímico. Los modelos Speech-Motion-Analyzer y Multimodal-Motion-Recognition son modelos entrenados en idioma inglés.
- Para la medición de pulso cardiaco, existen modelos que requieren solo el uso de la imagen facial. Los proyectos estudiados son Heartbeat y WebCam-Pulse-Detector. Heartbeat para teléfonos móviles Android utiliza el principio de un saturómetro, es decir, el flash del celular sobre el dedo contrasta los colores para detectar variaciones con la cámara.
- Nivel de atención. Los modelos explorados son OTAC, basado en pestañeos y bostezos, y Driver-State-Detection, el cual se basa en EAR (apertura de los ojos), distanciamiento central de la pupila, y tiempo de estado con ojos cerrados.

1.3. Objetivos

1.3.1. Objetivo General

Desarrollar una herramienta informática para el área oncológica de la Fundación Arturo López Perez (FALP), que se utilice en un ambiente de consulta oncológica remota, para potenciar la relación emocional entre paciente y profesional de la salud,

mediante la detección de emociones en base a la información obtenida desde una Webcam.

1.3.2. Objetivos Específicos

- **Precisar los requerimientos para la solución.**

Definir objetivos de estudio previo para comprender en profundidad el desafío y fijar las herramientas necesarias para su solución.

- **Reconocer seis emociones principales.**

Detectar las seis emociones principales en el paciente: tristeza, ira, felicidad, miedo, angustia e indiferencia.

- **Monitorizar cambios sanguíneos faciales, mediante webcam.**

Identificar los cambios en el flujo sanguíneo mediante la detección de colores en el rostro del paciente utilizando técnicas de pletismografía. El resultado debe ser una medición en BPM.

- **Medir el nivel de atención del paciente.**

Definir un sistema para detectar qué tan atento está el paciente en la teleconsulta.

- **Crear una API que ejecute los módulos durante la teleconsulta.**

Integrar los sistemas de reconocimiento anteriores en una API, de modo que se pueda acceder a ella durante la consulta.

1.3.3. Herramientas utilizadas

Dada la naturaleza modular del proyecto, existen herramientas que han sido útiles para los tres módulos, las que llamaremos "herramientas generales". Por otra parte, existen herramientas específicas por cada módulo. Vale mencionar que este documento redacta el desarrollo en detalle de dos módulos: Detección de ritmo cardíaco (CarMod),

y detección de nivel de atención (AtMod). Los dos módulos restantes son mencionados, pero no corresponden al desarrollo de este documento.

- Herramientas generales.

Para este trabajo se consideran herramientas tipo de software y de hardware.

1.4. Estructura del Documento

La estructura del documento es la siguiente: en la Sección 2 se hizo un acercamiento de los conceptos teleconsulta y la relación médico-paciente, para comprender el desarrollo del proyecto, además de conceptos técnicos referentes a las tecnologías, como son la fotoplethysmografía, señales cardíacas y Computer Vision.

La siguiente sección, Sección 3, se presenta una revisión de artículos y estudios, en donde se encuentran soluciones y acercamientos relacionados al software desarrollado.

En la Sección 4 se estructura el modelo del sistema, mostrando en ella los diagramas de contexto y arquitectura, especificados por los requisitos funcionales y no funcionales propuestos por el equipo de desarrollo.

El desarrollo de la solución ha sido explicitado en la Sección 5, el cual posee dos subsecciones: desarrollo del módulo de ritmo cardíaco, y el desarrollo del módulo de nivel de atención.

Por último, las validaciones y resultados de pruebas realizadas del sistema ya desarrollado han sido verbalizados en la Sección 7.

2. Marco Teórico

2.1. ¿Qué es una teleconsulta?

Este concepto se ha hecho popular desde la declaración de pandemia en el 2020 producto de la propagación del virus SARS-COV-2 a nivel mundial, el cual causa la enfermedad conocida como coronavirus 2019 (COVID-19). La situación ha hecho que las Tecnologías de Información y Comunicación (TICs) tomen un rumbo de potencialización hacia las modalidades remotas, ya que estas existen desde hace varios años, por lo que el concepto de telemedicina no es algo nuevo en términos tecnológicos. Pero, ¿qué es la telemedicina? La telemedicina es una prestación de servicios médicos empleada de forma remota [18], lo que incluye la ejecución de consultas remotas médicas, que se conoce como teleconsulta. Esta modalidad consiste en que el paciente y el personal de la salud interactúan por medio de tecnologías de información remota, con el fin de proporcionar asesoramiento de diagnóstico o terapéutico [19].

Hoy en día se utilizan las teleconsultas en diversas áreas de la salud para el trato con sus pacientes y, a pesar que se ha demostrado su efectividad, se distancia considerablemente de las consultas médicas tradicionales, o presenciales.

2.1.1. Requisitos de una teleconsulta

Para llevar a cabo una teleconsulta, es indispensable contar con los siguientes requisitos:

- Contar con dispositivos de comunicación.

Por ejemplo, un computador, tablet o smartphone.

- Conocimientos básicos sobre el uso de tecnologías.

Las personas involucradas deben conocer y saber manejar dispositivos electrónicos (e.g. Computador o Smartphone). También conocer acerca de sus limitaciones, lo que involucra tasa de transferencia de bits (bitrate), capacidad de procesa-

miento de su unidad de procesamiento de cómputo (CPU).

- Cumplir con requisitos de acceso a internet.

Se recomiendan conexiones estables de acceso a internet, como ADSL, fibra óptica o 4G al menos, con una tasa de transferencia mínima de 1Mbit, o 125kBps.

- Contar con software acorde a la consulta.

Existen muchas herramientas de software que permiten la telepresencia. Algunos ejemplos son Wbex, Skype, Zoom, Teams, Face Time, entre otras. Es importante que los interlocutores tengan un acuerdo de cuál plataforma utilizar.

2.2. Relación médico-paciente

Una de las bases de la ética médica contemporánea es la relación médico-paciente. Juega un papel realmente fundamental para la práctica médica de alta calidad en el diagnóstico y en el tratamiento de enfermedades. El personal médico debería ser un profesional con vocación de servir, con actitud compasiva y espíritu de ayuda hacia la persona que está enferma. Bajo este principio, estudia una compleja carrera científica para conocer el funcionamiento del cuerpo humano, aprendiendo de potenciales patologías al que se puede enfrentar y sus tratamientos.

Por otra parte, con el curso de la historia médica, esta carrera ha tenido una desnaturalización en la relación médico-paciente, lo que ha llevado a la creación de un término que ha cobrado importancia relevante: “El empoderamiento del paciente”.

2.2.1. Empoderamiento del paciente

La Organización Mundial de la Salud (OMS) lo define como el “proceso mediante el cual las personas adquieren un mayor control sobre las decisiones y acciones que afectan su salud, para ello, los individuos y las comunidades necesitan desarrollar habilidades, tener acceso a la información y recursos, y la oportunidad de participar e influir en los factores que afectan su salud y bienestar” [20].

Dicho lo anterior, se especifica el motivo del desarrollo del proyecto descrito en este documento, donde se establece una forma de potenciar la relación médico-paciente en consultas oncológicas remotas.

2.2.2. Oncología y emocionalidad

FALP solicitó al equipo de desarrollo una solución para la detección de emociones específicamente para consultas oncológicas remotas. Considerando lo descrito anteriormente con respecto a la responsabilidad que tiene el personal de la salud con el paciente, éste debe ser capaz de manejar la situación en la consulta a la hora de entregar un diagnóstico al paciente. Considerar que una noticia de cáncer es razón suficiente para alterar o perturbar la emocionalidad de la mayoría de las personas [21], lo que implica un efecto a la hora de sobrellevar el futuro tratamiento.

2.3. Ritmo cardíaco

La detección de emociones está relacionado con aspectos que pueden potenciar la interpretación de emociones para el médico. El ritmo cardíaco del paciente tiene directa relación con la expresión de algunas emociones [22]. Es útil a la hora de cuantificar la cantidad de veces que el corazón late, lo que implica movimientos cíclicos y regulares (para la mayoría de pacientes sanos), para mantener el flujo de sangre en el resto del cuerpo.

2.4. Fotopletismografía

Los avances tecnológicos han otorgado al área de la salud algunas herramientas para agilizar ciertos procesos semiológicos, como son la toma de presión y el ritmo cardíaco como tal. Estas métricas pueden ser capturadas tradicionalmente, pero las técnicas de pletismografía ayudan a obtener estos datos utilizando la reflexión de luz en la piel. La pletismografía funciona con el siguiente principio:

Es posible determinar el volumen del gas interno del tórax. Según la Ley de Boyle, el producto entre la presión y el volumen en un ambiente cerrado, es siempre constante. Esto significa que cualquier variación de V o P en $P \cdot V = k$ su producto k siempre es el mismo, tanto antes como después de causar el cambio. Esto llevó a la invención del pletismógrafo a presión que ayudó a obtener la variación de presión del cuerpo, midiendo los cambios de gas intratorácico.

Si bien el principio de la pletismografía difiere físicamente de la fotopletismografía, fue una creación importante para impulsar el siguiente método para medir la presión sanguínea, en base a la emisión y reflexión de luz sobre la piel.

2.4.1. Fotopletismografía Remota

El principio de la rPPG es similar al PPG, con la diferencia clave de que su objetivo es obtener el resultado de la variación volumen de pulso sanguíneo de forma remota. Para esto, se basa en la emisión de señales de color facial, capturadas por una cámara de video. Los sistemas informáticos pueden obtener estas variaciones de color en el formato RGB para ser interpretados como una variación de presión sanguínea.

2.5. Visión por computador

El equipo ha decidido desarrollar el software para el reconocimiento de emociones en base a herramientas existentes de Visión por Computador. También conocido como Visión Artificial, se trata de una disciplina que, en base a procesos de extracción de imágenes con cámaras, busca producir información tratable y entendible por un computador, en formato de números y símbolos. Una analogía que permite comprender lo que es la Visión por Computador (CV) es el funcionamiento en la transmisión de señales entre el ojo humano y el cerebro, ya que las imágenes captadas deben ser procesadas en el cerebro para que pueda comprender el mundo en el que está inmerso. La adquisición de imágenes no sólo son producto del uso de cámaras, ya que pueden

CAPÍTULO 2 : MARCO TEÓRICO

ser obtenidas por varios dispositivos que generen secuencia de imágenes, e incluso datos multidimensionales desde un escaner médico. La figura 2.1 muestra un esquema de visión por computador.

En términos de programación, Open Source Computer Vision Library (OpenCV) es un software dedicado a este campo. Se trata de una biblioteca que provee una infraestructura para aplicaciones de CV, y acelerar la percepción computacional en el campo de Machine Learning (ML). Para términos del proyecto, se considera OpenCV como base de desarrollo por el equipo. Es importante considerar, que, por motivos de rendimiento, OpenCV está desarrollado principalmente en C++, pero hoy en día existen conectores para ser trabajado en Python, Java y Matlab/Octave; así como también cuenta con la detección de Primitivas de Rendimiento Integradas (IPP) para optimizar recursos en el hardware que soporte esta tecnología.

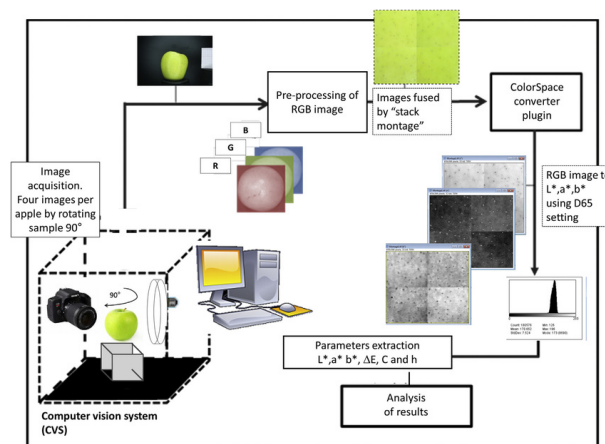


Figura 2.1: Esquema de Visión por Computador.

Python puede trabajar sobre OpenCV, y se pueden encontrar otras bibliotecas basadas en OpenCV que están basadas en el mismo lenguaje, como lo es Keras y PyTorchCV:

- **PyTorchCV** está basado en el framework PyTorch, siendo una colección de modelos de clasificación, segmentación, detección y estimación de posición corporal con imágenes.

- **Keras** es una biblioteca de redes neuronales de alto nivel, capaz de funcionar sobre PyTorchCV, cuyo enfoque es agilizar la experimentación para desarrolladores. Ofrece soporte tanto para redes neuronales convolucionales y redes recurrentes.

Estas bibliotecas han sido útiles para lograr el software final, ya que, reiteradamente, fue necesario realizar pruebas de rendimiento utilizando diversos modelos y métodos para el módulo de detección de ritmo cardíaco.

3. Estado del Arte

3.1. Fotopletismografía remota

1. PPG Heart Beat for Cognitive Fatigue Prediction

El estándar de oro para los neurocientíficos que estudian la fatiga laboral, es un electroencefalograma (EEG), pero es difícil obtener buenas mediciones de EEG fuera de un entorno clínico. Recientemente han surgido más investigaciones que usan mediciones de electrocardiograma (ECG o EKG), particularmente usando la Variabilidad del Ritmo Cardíaco (HRV) como factor. Las buenas mediciones de ECG tampoco son prácticas para las personas que se mueven en su entorno de trabajo [3, 2].

Esta aproximación de un sistema PPG nos indica que se pueden obtener datos de variabilidad del ritmo cardíaco. Esta técnica en un entorno remoto (rPPG) se puede aplicar para la detección de ritmo cardíaco en sistemas basados en el uso de la cámara web.

2. Heart Beat

Esta es una implementación simple de rPPG, una forma de medir la frecuencia cardíaca sin contacto con la piel. Utiliza una grabación de vídeo o transmisión en vivo de la cara para analizar cambios sutiles en el color de la piel.

Funciona de la siguiente manera:

- La cara es detectada y rastreada continuamente.
- La serie de señales se obtiene determinando el color facial en cada fotograma.
- La frecuencia cardíaca se estima mediante análisis de frecuencia y filtrado de la serie. [4]

3. PyVHR

Dada la aparición de variados métodos basados en programación dirigida por datos, basados en modelos y estadísticas en las últimas dos décadas, se ha logrado mejorar la estimación de señal del pulso de volumen sanguíneo (BVP) para ser interpretado

como BPM (latidos por minuto). El presente framework PyVHR, contiene conductos de varias etapas que cubren el proceso de extracción y análisis de fluctuaciones cardiacas. Está destinado a uso de estudio y aplicaciones prácticas donde el uso de sensores no es una opción viable.

Dado que los métodos implementados en el framework ya han sido estudiados en diversas oportunidades por la comunidad científica, se establece PyVHR como piedra angular del trabajo relacionado a la detección de ritmo cardiaco. La implementación de los métodos de PyVHR se han basado en los siguientes estudios: [5]

a) *Green*: Ambiente experimental, las señales de fotopletismografía fueron medidas a más de un metro de la fuente, utilizando luz ambiental y una cámara de nivel consumidor. Latidos y respiración pudieron ser cuantificados hasta varios armónicos. El canal verde (green) se caracteriza como el más fuerte en la señal de pletismografía, correspondiente al peak de absorción por la hemoglobina.

- Verde: peak de absorción por la hemoglobina.
- Rojo y azul: otra información pletismográfica.

La fotopletismografía con luz ambiental puede ser útil para fines médicos, como la caracterización de lesiones vasculares en la piel (marcas de nacimiento) y captura remota de signos vitales (ritmos cardiacos y respiratorios).[6]

b) *CHROM*: La fotopletismografía remota habilita el monitoreo del pulso de volumen de sangre usando una cámara regular de consumidor. Los estudios que han mejorado en la robustez de movimiento proponen una fuente ciega en técnicas de separación BSS en el espacio de colores RGB, demostrando un éxito limitado. El estudio presenta un análisis del problema del movimiento, basado en los métodos emergentes de crominancia.[7]

c) *ICA*: Los intentos hasta ahora para realizar medidas remotas de pulso cardiaco no son automatizadas y son susceptibles a artefactos de movimiento, y son típicamente costosos. Este estudio propone un nuevo acercamiento para estos problemas, aplicable

a grabaciones de videos en colores de la cara humana, automatizado para el seguimiento facial mientras se realiza separación de señales base correspondiente a los canales de colores, en componentes independientes.

Usando Bland-Altman y análisis correlativo, se compara el ritmo cardiaco extraído de los videos con una cámara web básica, con un sensor BVP (finger blood volumen pulse) aprobado por FDA, logrando una alta precisión y correlación incluso con la presencia de artefactos de movimientos.

Es posible aplicar este método a más de una persona a la vez, y se declara que el modelo se puede utilizar con tres pacientes simultáneos ante la misma cámara [23].

d) LGI: Estimación del ritmo cardiaco desde videos faciales en entornos denominados ruidosos debido a la presencia de movimientos de cabeza, habla, expresiones faciales e iluminación natural en varios escenarios. Se basa en el resultado producto de una reorganización de la señal de volumen sanguíneo, en espacios vectoriales con una distribución concentrada. La incertidumbre en la distribución característica es incorporada en un modelo que aprovecha la invarianza local del ritmo cardiaco. Se logró un rendimiento de estimación eficiente de ritmo cardiaco en videos ruidosos previamente descritos.[8]

e) PBV: Remote PPG detecta los cambios diminutos de absorción óptica causados por las variaciones del volumen de sangre en la piel. Los diferentes espectros de absorción de la sangre arterial y la piel sin sangre causan que las variaciones ocurran a lo largo de un específico vector del espacio normalizado RGB. Este vector puede determinarse por un espectro de luz dado y por características de transferencia dadas de los filtros ópticos de la cámara.[9]

f) PCA: Algoritmo elaborado para un registro eficiente de ritmo cardíaco directamente desde imágenes capturadas por una cámara web. La señal deseada se obtiene por la correcta selección de canales y análisis de componentes principales. Esta técnica fue desarrollada con el propósito de monitorización domiciliaria [10].

g) POS: Introducción a un modelo matemático que incorpora las pertinentes propie-

dades ópticas y fisiológicas para los reflejos en la piel. El modelo es usado para explicar las diferentes decisiones que han sido utilizadas en los modelos actuales de rPPG para la extracción de pulso.[11]

4. Algorithmic Principles of Remote PPG

Esta investigación introduce un modelo matemático que incorpora propiedades visuales y fisiológicas de la piel para aumentar la exactitud de rPPG. Esto se logra mediante el diseño de un método alternativo a rPPG, donde una proyección del plano ortogonal al tono de la piel, se utiliza para la extracción de pulso.[11]

5. Mental Stress PPG

El estrés mental es una respuesta natural a las actividades diarias. Sin embargo, el estrés agudo y prolongado puede causar enfermedades psicológicas y cardíacas. La variabilidad de la frecuencia cardíaca (HRV) se considera un indicador de estrés mental y condición física. La forma estándar de obtener HRV es usando electrocardiografía (ECG) como el intervalo de tiempo entre dos picos R consecutivos. La fotopletiśmografía (PPG) se considera una alternativa para la detección del estrés mental utilizando la variabilidad de la frecuencia del pulso (PRV), el intervalo de tiempo entre dos picos sucesivos de PPG. Mental Stress PPG ayuda a confirmar la utilización de rPPG para la detección de ritmo cardíaco en el paciente de forma remota.[12]

6. Evaluación de sesgos en métodos de fotopletiśmografía remota (rPPG)

Actualmente, los conjuntos de datos faciales disponibles públicamente acompañados de mediciones fisiológicas generalmente se capturan en entornos de laboratorio controlados, sin diversidad en los tonos de piel, la edad y los artefactos culturales de los sujetos (p. ej., bindi que usan las mujeres indias).

En este estudio se han recopilado datos de video faciales y frecuencia del pulso de sujetos humanos en India y Sierra Leona, para cuantificar la incertidumbre en los métodos de estimación de la frecuencia del pulso sin contacto. Los datos de vídeo se utilizan para estimar la frecuencia del pulso utilizando rPPG de última generación basado en cámaras y comparados con mediciones reales capturadas usando un pulso basado en

contacto aprobado por la FDA dispositivo de medición de velocidad. El estudio revela que los métodos rPPG exhiben sesgos similares cuando se comparan con un dispositivo basado en contacto a través de grupos demográficos y condiciones ambientales. La diferencia media entre las frecuencias del pulso medidas por métodos rPPG se encuentra alrededor de un 2 % (1 latido por minuto (bpm)), lo que significa concordancia con los métodos rPPG de contacto. También encontraron que los métodos rPPG muestran una variabilidad de la frecuencia del pulso de 15 % (11 bpm), en comparación con la realidad.[13]

3.2. Detección de atención

1. Webcam-based Attention Tracking in Online Learning: A Feasibility Study

En este artículo, exploran la posibilidad de hacer uso de métricas simples como la presencia de la mirada o la presencia de la cara para detectar una pérdida de atención en el entorno de aprendizaje en línea. Con este fin se evalúa el rendimiento de tres marcos de seguimiento ocular para consumidores y profesionales utilizando un paquete de referencia que diseñaron específicamente para este propósito: contiene un conjunto de actividades y comportamientos comunes de los usuarios de xMOOC (eXtended Massive Open Online Course). Los resultados del estudio muestran que esas métricas simples ya representan un desafío significativo para las soluciones actuales de seguimiento ocular basadas en hardware y software.[14]

3.3. Posibles Datasets

1. UBFC-Phys Dataset

UBFC-Phys consiste en un compilado de videos, donde se muestran las caras de 56 participantes en un experimento de vivir situaciones de estrés con un protocolo riguroso inspirado en la prueba de estrés social de Trier (TSST por sus siglas en inglés) donde se utiliza una pulsera para medir el BVP, y la actividad electrodérmica (EDA), ambos

para obtener un Ground Truth. Los integrantes son conducidos a través de tres estados: descanso, un discurso y realizar tareas aritméticas con diferentes niveles de dificultad. El dataset contiene las grabaciones de las personas y sus niveles de BVP/EDA durante las tres tareas, además, de sus niveles auto calculados de ansiedad.[9]

2. UBFC-RPPG Dataset

Este dataset fue creado utilizando un programa personalizado en C++ para la adquisición de videos de una cámara Logitech C920 HD Pro a 30fps con resolución de 640x480 en formato RGB de 8-bits sin comprimir. También se utilizó un Oxímetro de pulso para dedo CMS50E para obtener el valor de frecuencias cardíacas PPG y poder ser contrastadas con la onda de rPPG.

El conjunto de videos, que muestra los rostros de las personas participantes, se divide en dos sub-datasets:

- El primero consta de personas sentadas a un metro de distancia de la cámara, lo más quietas posible. Se compone de 8 videos (alrededor de 16500 frames).
- En el segundo dataset, igualmente sentados a un metro de distancia de la cámara, se les pide realizar pruebas matemáticas con el objetivo de aumentar su frecuencia cardíaca mientras se emula en simultáneo una interacción normal entre humano-computador. Consta de 42 videos compartidos para investigación [15].

3. KDEF & AKDEF

Karolinska Directed Emotional Faces (KDEF) es un conjunto de 4900 imágenes de expresiones faciales humanas. El conjunto de imágenes contiene 70 individuos que muestran 7 expresiones emocionales diferentes. Cada expresión se ve desde 5 ángulos diferentes.

The Averaged Karolinska Directed Emotional Faces (AKDEF) es un conjunto de 70 imágenes de expresiones faciales humanas. El conjunto de imágenes contiene una mujer promedio y un hombre promedio que muestran 7 expresiones emocionales diferentes. Cada expresión se ve desde 5 ángulos diferentes [16].

4. FER 2013

Fer2013 contiene aproximadamente 30.000 imágenes RGB faciales de diferentes expresiones con un tamaño restringido a 48×48 píxeles, y las etiquetas principales del mismo se pueden dividir en 7 tipos: 0=Enojado, 1=Disgusto, 2=Miedo, 3=Feliz, 4=Triste, 5=Sorpresa, 6=Neutral. La expresión Disgust tiene la cantidad mínima de imágenes: 600, mientras que otras etiquetas tienen casi 5000 muestras cada una.[17]

4. Diseño de la Arquitectura del sistema

Este proyecto cuenta con cuatro módulos, de los cuales son dos de ellos los que son descritos en este documento. Según los requerimientos de FALP, el software final debe cumplir con lo siguiente:

1. Una herramienta que pueda ser ejecutada en tiempo real, durante una teleconsulta.
2. Médico y paciente son los únicos participantes. No debe haber intervención de terceros durante la consulta médica, respetando la confidencialidad del paciente.
3. La información del paciente con respecto a su emocionalidad debe ser visible para el médico durante la consulta.
4. Considerando el contexto situacional del paciente, este sólo debe contar con un dispositivo de comunicación doméstico, como lo es un smartphone, computador de escritorio, tableta o similares que soporten el uso de un micrófono y cámara.
5. Mostrar al médico la emoción que está experimentando el paciente.
6. Para reforzar la interpretación de la emocionalidad por parte del médico, también se debe informar del ritmo cardíaco y el nivel de atención del paciente.

Con esta información es posible describir los requerimientos funcionales y no funcionales del proyecto.

4.1. Requisitos funcionales

Los Requisitos funcionales (RF) son expresiones que definen el comportamiento del sistema desde el punto de vista del usuario. Antes de comenzar con el desarrollo del proyecto, han sido definidos los siguientes RF:

RF0: Se activa el sistema con una conexión previa para la teleconsulta.

RF1: Internamente estará el módulo de detección de emociones detectando el sentimiento del paciente.

RF2: Identificar el nivel de ritmo cardíaco del paciente a través de rPPG.

RF3: Mostrar porcentualmente, la atención del paciente mientras habla el médico.

RF4: Se genera a un costado de la teleconsulta un cuadro donde se exhibirá la información obtenida, es decir, la emoción actual del paciente, su nivel de atención y ritmo cardíaco estimado.

RF5: Generar un reporte con las mediciones obtenidas en la sesión.

RF6: Luego de la consulta, y posterior a la aprobación del médico, idealmente el sistema enviará el video al servidor para su retroalimentación y así mejorar los resultados en sesiones futuras.

4.2. Requisitos no funcionales

Los Requisitos no funcionales (RNF) definen principalmente la calidad del funcionamiento del sistema, en base al juicio del usuario. Para esto, se definieron los siguientes RNF:

RNF0: La precisión del sistema debe ser de al menos 75 %.

RNF1: El modelo entrenado, si aplica para el módulo, deberá actuar en tiempo real.

RNF2: El sistema debe estar montado en un servidor de FALP, y ser accesible en todas las teleconsultas realizadas. Esto es responsabilidad de FALP.

RNF3: Se mantendrá la confidencialidad de los datos adquiridos, así como también de la videollamada, el diagnóstico del profesional de salud, entre otros.

El desglose de información descrita anteriormente nos permite definir la arquitectura del sistema, y el contexto durante y donde el cual será utilizado.

4.3. Diagrama de Contexto

El siguiente diagrama permite comprender los límites de alcance, entidades y sus relaciones, mostrando una vista de alto nivel del sistema. Se ha definido el diagrama mostrado en la figura 4.1.

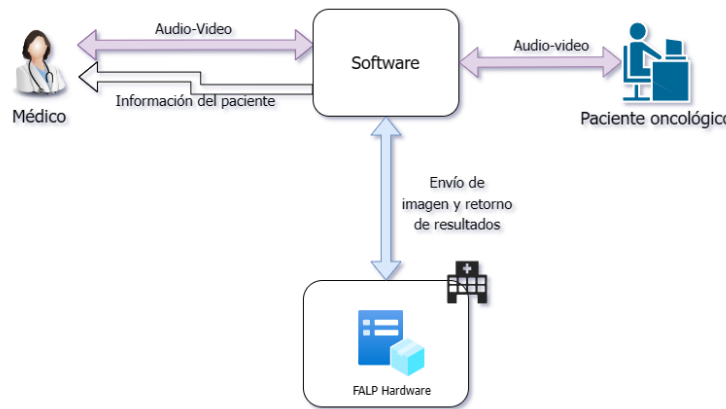


***Figura 4.1:** Diagrama de contexto. Muestra las entidades que interactúan con el sistema, y su flujo de datos.*

El diagrama muestra tres elementos:

- El software, o **sistema** representa el sistema que contiene una serie de funciones para obtener el resultado deseado.
- Las **entidades externas** incluyen a las dos personas involucradas que interactúan con el sistema, los cuales son el médico tratante y el paciente oncológico, presentes en la teleconsulta. También existe el hardware, un servidor que ejecuta las teleconsultas en conjunto con el sistema, procesando los datos necesarios para ser enviados al médico.
- Las **líneas de flujo** muestran el flujo de los datos. El audio-video es transmitido normalmente entre los interlocutores, como en una reunión remota común. Adicionalmente, el sistema envía los datos del paciente al médico (bpm, atención y emoción), que fueron generados gracias a servidor de FALP.

4.4. Diagrama de Arquitectura

La arquitectura del sistema se diseñó en base a cuatro módulos que actuarán de forma paralela. Los tres primeros obtienen los frames (fotogramas) capturados por una webcam, la que recibe las imágenes en tiempo real del paciente. Estos frames son recibidos por el sistema y enviados a los tres módulos principales: Módulo de ritmo cardíaco (CarMod), Módulo de nivel de atención (AtMod) y Módulo de emociones (EMod) (Ver figura 4.2)

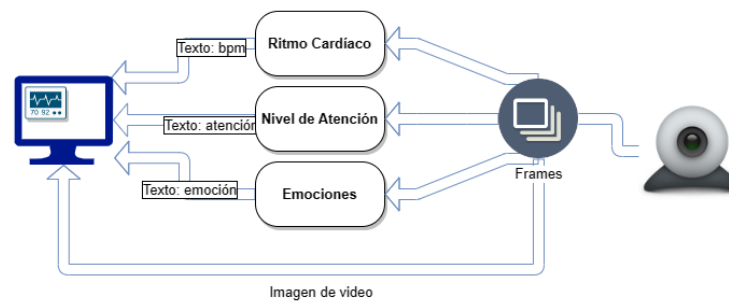


Figura 4.2: Diagrama de arquitectura del sistema.

Estos módulos procesan los frames de modo que obtienen la información que le corresponde a cada uno, en formato de texto. CarMod genera el ritmo cardíaco en bpm, AtMod genera una métrica del nivel de atención del paciente, y EMod genera la emoción que está sintiendo el paciente. Estos datos se envían al cuarto módulo, que corresponde a una interfaz que ordena estos datos por segundo y los envía a la API que lo requiera. En este caso, la API es un sistema de reunión remota utilizada por FALP, la cuál tendrá un recuadro con la información capturada en una esquina para el médico. Ya que este documento hace un acercamiento de los módulos AtMod y CarMod, este cuarto y último módulo es obviado.

5. Desarrollo de los módulos

5.1. Entorno de desarrollo.

El equipo de desarrollo ha tenido a disposición un servidor proveniente de la universidad. Cada estudiante ha trabajado en uno de sus computadores personales tanto por SSH y localmente. Se han configurado entornos de pruebas en máquinas virtuales alojadas en Windows, utilizando el mismo hardware incorporado de las computadoras en las pruebas, como cámaras web. A continuación se describe el entorno de desarrollo en términos de hardware, y la configuración del entorno virtual para la implementación de los módulos CarMod y AtMod.

Servidor:

- CPU: Intel®Core™ 10900
 - 10 núcleos
 - 20 hilos
 - Clock base/turbo: 2.80 GHz/5.20 GHz
- GPU: NVIDIA®GeForce RTX 3060
- 32GB RAM DDR4

Computador personal:

- CPU con GPU integrada: AMD®Ryzen 5 4500U with Radeon Vega 6 Graphics.
- 16 GB RAM LPDDR4 4266 MHZ

En cuanto a software, se ha utilizado principalmente Anaconda, una plataforma de distribución para desarrollo en Python con compatibilidad para la mayoría de sistemas

operativos. En este caso, se ha utilizado la alternativa ‘liviana’ conocida como MiniConda, el cuál posee la cantidad básica de herramientas para generar entornos virtuales y algunas bibliotecas y paquetes básicos para trabajar en Python.

Para establecer el entorno:

- Instalación de MiniConda en Windows 10 21H1.
- Crear entorno con Conda, utilizando Python 3.10.

Las bibliotecas y paquetes utilizados:

- Scikit-Learn
- Pandas
- NumPy
- Matplotlib
- Keras
- TensorFlow
- Cupy (CUDA en el servidor, no estable para trabajar con AMD)

El banco de pruebas para realizar la extracción de datos de rendimientos del software final, evaluar los métodos rPPG y su respectiva minería de datos, se han utilizado Pandas, Matplotlib y Numpy sobre jupyter Notebook. Las demás dependencias se utilizaron dentro de un módulo que contiene el software desarrollado, luego de su implementación.

5.2. Módulo: Detección de Ritmo Cardíaco

En base a la investigación de la Sección 3, para el desarrollo del presente módulo se utilizó un framework basado en Python llamado Python framework for Virtual

Heart Rate (PyVHR). Este paquete es un marco integral para estudiar los métodos de estimación de la frecuencia del pulso cardíaco, basados en video, conocido también como fotopletimografía remota (rPPG). PyVHR ha sido elegido como base por los siguientes motivos:

- **Limitaciones en el contexto situacional:** la utilización de una webcam proveniente de los dispositivos de comunicación domésticos presentes en una teleconsulta es la única herramienta que es utilizada para obtener los resultados. Este framework cumple con que el uso de frames provenientes de un video grabado es suficiente para calcular el HR. Una de las tareas ha sido llevar este procedimiento a tiempo real.
- Las **capacidades del servidor** permiten el uso de funciones que computen resultados basándose en GPU y CPU. El cliente especificó que los procedimientos deben utilizar única o mayormente solo GPU. Python framework for Virtual Heart Rate (PyVHR) tiene en su implementación los métodos clásico de rPPG.
- **pyVHR ya tiene implementado un pipeline** que cuenta con los pasos necesarios para la interpretación de HR, lo que ha permitido la de reutilización de código durante el desarrollo.

El desarrollo del módulo se resume como un desarrollo de un software en el entorno descrito en la Sección 5.1. PyVHR está basado estructuralmente en un pipeline de etapas múltiples, que cubren todo el proceso para la extracción de señales provenientes desde frames, y el análisis de fluctuaciones de Heart rate (HR). En base a este pipeline, se ha determinado la siguiente estructura de software.

5.2.1. Paso 1: Captura de imagen y requisitos de código.

Al inicio es necesario contar con los frames pertenecientes a la fuente del video a tratar. Esta parte se ha especificado con más detalle en el desarrollo de un módulo

CAPÍTULO 5 : DESARROLLO DE LOS MÓDULOS

que no pertenece a este documento, pero por términos prácticos, se asumirá que la imagen será capturada con Open Source Computer Vision Library (OpenCV). El siguiente fragmento de código cumple con esta tarea:

```
import cv2
def _generate(fps = 25, seconds = 1):
    video_file = 'video_sample.avi'

    counter = 0
    replay = 0
    cap = cv2.VideoCapture(video_file)
    frames = []

    while True:
        ret, frame = cap.read()
        if not ret:
            replay = replay + 1
            cap = cv2.VideoCapture(video_file)
            ret, frame = cap.read()
        if replay >= 1:
            break
        if counter >= fps*seconds:
            break
        counter = counter+1
        frames.append(frame)
    cap.release()
    return frames
```

Los argumentos de la función *generate* son *fps* y *seconds*, que permiten indicar la cantidad de segundos desde el inicio del video que se requieren procesar, y los frames por segundo a los que trabaja el video. Es necesario no sobrepasar los fps reales del video en este parámetro, debido que generaría redundancia en el procesado de algunos frames al ser enviados a la función *analysis*. *Counter* y *replay* son parámetros para manejar la cuenta de veces que se repite el video, en forma de bucle con fines de pruebas de software.

Con OpenCV se puede capturar una entrada de video que, en este caso, se utiliza un ejemplo con extensión avi. OpenCV captura la imagen con el método *VideoCapture* desde el paquete *cv2*, lo cual se almacena como parámetro *cap*. Luego, cada iteración

CAPÍTULO 5 : DESARROLLO DE LOS MÓDULOS

hará una lectura (*cap.read()*), cuyos valores de retorno se establecen como (*ret*), para conocer si la lectura fue exitosa, y en *frame*, para obtener los valores del frame. Como las iteraciones dependen de la cantidad deseada de fps y frames, estos son almacenados en una lista para ser retornados como *frames*.

Con este valor se puede proceder a utilizar los métodos desarrollados para el módulo.

```
from mainBPM import *

apply_params()
bpm = analisis(_generate(seconds = 160))
```

Con lo anterior implementado, se procede a explicar el código del software desarrollado para el CarMod. Primeramente, se necesitan los paquetes siguientes:

```
import ast
import numpy as np
import queue
import time

from pyVHR.extraction.skin_extraction_methods import *
from pyVHR.extraction.sig_extraction_methods import *
from pyVHR.extraction.sig_processing import *
from pyVHR.BVP.BVP import *
from pyVHR.BVP.methods import *
from pyVHR.BVP.filters import *
from pyVHR.BPM.BPM import *

from pyVHR.realtime.params import Params
```

La llamada al método *apply_params* se utiliza para aplicar los parámetros de configuración. Por defecto, se aplica el método de extracción de piel con *convexhull*, método rPPG por *cupyPOS*, con un enfoque holístico para los resultados.

5.2.2. Paso 2: Extracción de rostro

PyVHR tiene implementado dos métodos para la extracción de rostro: *convex hull* y *face-parcing*. Ambos métodos se basan en tecnologías diferentes que obtienen resultados comparables. No obstante, el rendimiento y tiempo de respuesta es diferente. La

figura 5.1 muestra una comparación de las regiones de interés extraídas, quedando en negro lo que no es de interés.

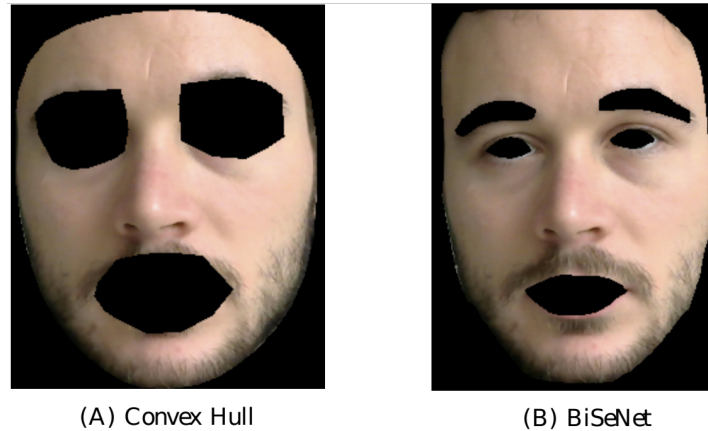


Figura 5.1: Imagen obtenida de [5]. Resultado de Convex hull (A), y face-parsing (B) procesado utilizando la imagen facial de un sujeto del dataset LGI-PPGI.

Face-parsing: Una alternativa es face-parsing[25] de BiSeNet [26], una solución que obtiene la región de interés con una extracción completa y, posteriormente, eliminación de las secciones que no son de interés (ojos, cejas y labios), lo cual ha sido descartado debido a los errores persistentes al ser probado en algunos datasets.

Convex hull: Consiste en obtener solo las regiones de interés del rostro gracias a los 468 puntos fiduciales faciales manejados por MediaPipe Face Mesh [24], donde las coordenadas de los puntos y sus conexiones entregan información de las regiones de forma numérica. MediaPipe Face Mesh es una solución que emplea ML para inferir tridimensionalmente la superficie facial sin la necesidad de ningún sensor además de una cámara en sí. En la figura 5.2 se muestra la malla facial utilizada por MediaPipe con los landmarks (puntos faciales), comparado con su funcionamiento sobre un rostro real.

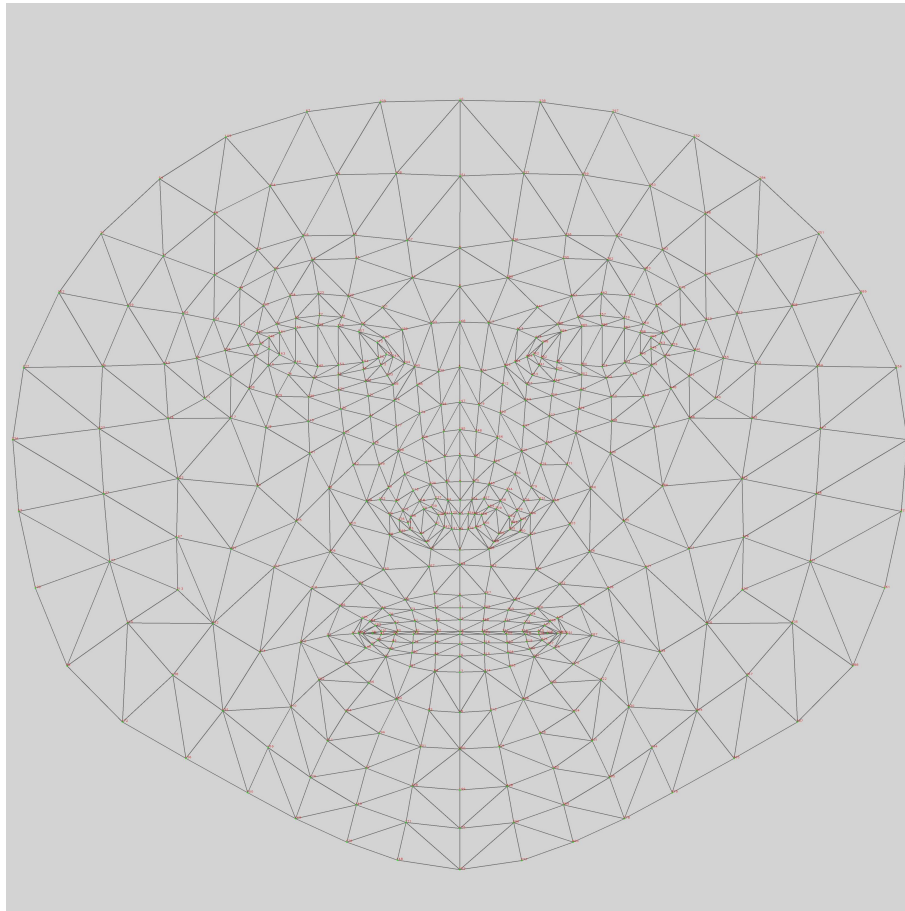


Figura 5.2: Cada punto se encuentra enumerado, lo cuál sirve como índice o identificador del landmark.

Para obtener la región de interés, utilizando convexhull, se debe utilizar el método *SkinExtractionConvexHull*, especificando que la GPU será la encargada de procesar los datos de extracción. Esto crea un objeto cuyo método es extraer la piel facial de interés, almacenado en su atributo *cropped_skin_im*. Este método será llamado más adelante.

```
target_device = 'GPU' if Params.cuda else 'CPU'
skin_ex = None
skin_ex = SkinExtractionConvexHull(target_device)
```

Al configurar el método de extracción, se procede a utilizar la declaración *with*, para asegurar que el uso de FaceMesh no retorne errores en base a la captura de frames no válidos (e.g: rostro no presente).

Listing 1: Bucle de iteración por frame.

```
with mp_face_mesh.FaceMesh(  
    max_num_faces=1,  
    min_detection_confidence=0.5,  
    min_tracking_confidence=0.5) as face_mesh:  
    counter = 0  
    while True:  
        ...
```

Dentro de este *with* comienzan las iteraciones por cada frame capturado. Dentro se manejan la cantidad de frames, una lista de resultados por cada frame y el tratamiento de la imagen temporal para obtener aquellos resultados. Se itera por cada frame, por lo que es necesario aumentar un contador. Para procesar la imagen, se debe transformar el formato de colores de BGR a RGB. También se necesita conocer las dimensiones de la imagen.

Listing 2: BGR a RGB y dimensión de imagen.

```
image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)  
processed_frames_count += 1  
width = image.shape[1]  
height = image.shape[0]
```

FaceMesh tiene un método para obtener los landmarks en base al frame en proceso. El siguiente código genera el resultado de los landmarks, el cual se trata de un arreglo multidimensional, con valores de posición e intensidad de color por cada uno de ellos:

Listing 3: Obtención de landmarks faciales.

```
results = face_mesh.process(image)  
if results.multi_face_landmarks:  
    #print("Hay resultados multi face landmarks")  
    face_landmarks = results.multi_face_landmarks[0]  
    landmarks = [l for l in face_landmarks.landmark]  
    for idx in range(len(landmarks)):  
        landmark = landmarks[idx]  
        if not ((landmark.HasField('visibility') and landmark.visibility < VISIBILITY_THRESHOLD)  
                or (landmark.HasField('presence') and landmark.presence < PRESENCE_THRESHOLD)):  
            coords = mp_drawing._normalized_to_pixel_coordinates(  
                landmark.x, landmark.y, width, height)  
            if coords:  
                ldms[idx, 0] = coords[1]
```

```
ldmks[idx, 1] = coords[0]
### skin extraction ###
cropped_skin_im, full_skin_im = skin_ex.extract_skin(
    image, ldmks)
else:
    cropped_skin_im = np.zeros_like(image)
    full_skin_im = np.zeros_like(image)
```

Anteriormente se obtuvo el objeto *skin_ex* y su método de extracción que recibe la imagen y los landmarks obtenidos de dicha imagen. Hasta este punto, la piel de interés se encuentra en la variable *cropped_skin_im*. Con este dato se puede continuar con el siguiente paso para procesar las señales.

5.2.3. Paso 3: Procesamiento de señal RGB

La región de interés de la imagen obtenida se debe procesar en términos de señal RGB. Primero, se necesita calcular el promedio de la señal RGB, excluyendo los píxeles de la imagen que se encuentran fuera del rango

[*RGB_LOW_TH*, *RGB_HIGH_TH*], sin incluir los extremos.

El siguiente código entrega un arreglo RGB-mean, de forma [1, 3], en donde 1 es un estimado, y 3 son los promedios de rojo, verde y azul.

Listing 4: *Enfoque holístico.*

```
elif Params.approach == 'holistic':
    temp = sig_ext_met(cropped_skin_im, np.int32(
        SignalProcessingParams.RGB_LOW_TH), np.int32(SignalProcessingParams.RGB_HIGH_TH))
    sig.append(temp)
```

Dado que la aproximación holística ya generó un promedio de las intensidades RGB, este paso se encarga de dividir la zona del rostro y generar ventanas superpuestas. Cada ventana es almacenada dinámicamente con su respectiva información de señal en RGB. Lo siguiente muestra los pasos para el filtrado de señal:

```
for filt in Params.pre_filter:
    if filt != {}:
        if 'fps' in filt['params'] and filt['params']['fps'] == 'adaptive' and fps is not None:
            filt['params']['fps'] = float(fps)
        if filt['params'] == {}:
            copy_sig = filt['filter_func'](
                copy_sig)
        else:
            copy_sig = filt['filter_func'](
                copy_sig, **filt['params'])
```

5.2.4. Paso 4: Métodos de rPPG, obtención de BVP

En este punto, el método de rPPG trabaja sobre las señales RGB del rostro extraído. En el estado del arte se han mencionado varios métodos que pyVHR ya tiene implementados. El trabajo principal en esta sección ha sido evaluar cada uno de ellos utilizando procesador gráfico en lugar de CPU. El parámetro 'method_func' tiene, en este caso, el valor 'cupyPOS', que se trata del método electo en base al estudio de rendimiento que se explica en la Subsubsección 5.2.6.

Listing 5: Inicio de la obtención de BVP

```
bvp = signals_to_bvps_cuda(copy_sig, Params.method['method_func'], Params.method['params'])
```

Vale mencionar que el valor 'cupy' hace referencia al uso de GPU como dispositivo para computar esta sección. Se refiere a una biblioteca llamada CuPy de código abierto, para computación acelerada por GPU con Python, que utiliza herramientas de CUDA para hacer uso total de los recursos de hardware gráfico.

Algoritmo Plane-Orthogonal-to-Skin:

El algoritmo Plane-Orthogonal-to-Skin (POS) es una alternativa a los métodos basados en PCA e ICA. Son criterios para diferenciar trazas RGB temporales de señales que permiten capturar el pulso; CHROM, que combina linealmente las señales de crominancia, asumiendo un color estandar para blanquear las imágenes; El algoritmo se explica con el siguiente pseudocódigo:

Listing 6: Algoritmo obtenido de [11]

Input: Secuencia de video con N fotogramas.

```

for n = 1, 2, ..., N do
     $C(n) = [R(n), G(n), B(n)]^T$  Promediado espacial
    if m = n-1+1 > 0 then
         $C_n^i = \frac{C_{m \rightarrow n}^i}{\mu C_{m \rightarrow n}^i}$  Normalización Temporal
         $S = \begin{pmatrix} 0 & 1 & -1 \\ -2 & 1 & 1 \end{pmatrix} C_n$  Proyección
         $h = S_1 + \frac{\theta(S_1)}{\theta(S_2)} S_2$  Tuning
         $H_{m \rightarrow n} = H_{m \rightarrow n} + (h - \mu(h))$  Agregar superposición.
    end if
end for

```

Output: pulsos H.

Para efectuar POS se necesitan los valores RGB de cada fotograma promediados espacialmente, considerando que cada frame tiene pixeles con distintos valores RGB independientes. Esto es para reducir la cantidad de colores utilizados en una imagen, que además es útil para mejorar la eficiencia de compresión. Este promedio espacial se expresa con el valor $C(n)$ dado un frame n. Considerar también que el video de entrada posee N frames.

Con $C(n)$ calculado, se debe obtener una normalización temporal, siendo el cociente entre las señales RGB y su promedio. Este proceso es útil para la eliminación del promedio de las señales producto del reflejo de la piel.

La siguiente etapa es la proyección, en donde se utiliza un plano ortogonal a la matriz correspondiente al tono de la piel. Para esto, se realiza la proyección espacio RGB temporalmente normalizado, que consta de un espacio de 360 ejes de proyección $Z(z_1, z_2, z_3)$ discretos.

La etapa de ajuste, o Tuning, sirve para refinar la dirección de la proyección a modo de suavizado de resultados. Con este paso se obtiene un espacio más ‘probable’ o ajustado a lo esperado.

Por último, en la etapa de superposición, o overlap-adding, combina la señal de salida para ser utilizada en el siguiente proceso del software, que corresponde a la

obtención de BPM.

El código implementado en pyVHR inspirado en el algoritmo POS mencionado es el siguiente:

***Listing 7:** Algoritmo desarrollado por pyVHR*

```
def cupy_POS(signal, **kargs):
    # Run the pos algorithm on the RGB color signal c with sliding window length wlen
    # Recommended value for wlen is 32 for a 20 fps camera (1.6 s)
    eps = 10**-9
    X = signal
    fps = cupy.float32(kargs['fps'])
    e, c, f = X.shape      # e = #estimators, c = 3 rgb ch., f = #frames
    w = int(1.6 * fps) # window length

    # stack e times fixed mat P
    P = cupy.array([[0, 1, -1], [-2, 1, 1]])
    Q = cupy.stack([P for _ in range(e)], axis=0)

    # Initialize (1)
    H = cupy.zeros((e, f))
    for n in cupy.arange(w, f):
        # Start index of sliding window (4)
        m = n - w + 1
        # Temporal normalization (5)
        Cn = X[:, :, m:(n + 1)]
        M = 1.0 / (cupy.mean(Cn, axis=2)+eps)
        M = cupy.expand_dims(M, axis=2) # shape [e, c, w]
        Cn = cupy.multiply(M, Cn)

        # Projection (6)
        S = cupy.dot(Q, Cn)
        S = S[0, :, :, :]
        S = cupy.swapaxes(S, 0, 1) # remove 3-th dim

        # Tuning (7)
        S1 = S[:, 0, :]
        S2 = S[:, 1, :]
        alpha = cupy.std(S1, axis=1) / (eps + cupy.std(S2, axis=1))
        alpha = cupy.expand_dims(alpha, axis=1)
        Hn = cupy.add(S1, alpha * S2)
        Hnm = Hn - cupy.expand_dims(cupy.mean(Hn, axis=1), axis=1)
    # Overlap-adding (8)
```

```
H[:, m:(n + 1)] = cupy.add(H[:, m:(n + 1)], Hnm)
```

```
return H
```

5.2.5. Paso 5: Estimación de BPM

Dada la información generada en el paso anterior por cada una de las ventanas del rostro, la estimación de BPM se genera transformando BVP en función de la frecuencia. Esto es posible aplicando la transformada de Fourier en tiempo discreto (FDT) y estimaciones de Welch para obtener la potencia en el espectro de frecuencias.

Listing 8: *Obtención de BPM desde BVP.*

```
if Params.cuda:
    bvp_device = cupy.asarray(bvp)
    if BPM_obj == None:
        # Asegura que exista el objeto BPM
        BPM_obj = BPMcuda(bvp_device, fps,
                           minHz=Params.minHz, maxHz=Params.maxHz)
    else:
        BPM_obj.data = bvp_device
    if Params.BPM_extraction_type == "welch":
        bpm = BPM_obj.BVP_to_BPM()
        bpm = cupy.asnumpy(bpm)
    elif Params.BPM_extraction_type == "psd_clustering":
        bpm = BPM_obj.BVP_to_BPM_PSD_clustering()
else:
    # Mismo procedimiento, pero utiliza directamente (bvp)
    if BPM_obj == None:
        BPM_obj = BPM(bvp, fps, minHz=Params.minHz,
                       maxHz=Params.maxHz)
....
```

Se han implementado dos caminos para este cálculo: utilizar GPU con tecnologías de CUDA en base a CuPy, o bien utilizar CPU. En primera instancia *Params.cuda* es por defecto *true*, por lo que se procede a utilizar valores de arreglo CUDA. El objeto *BPMcuda* se encarga de manejar la información de entrada, el cual tiene dos métodos internos que realizan la transformación. Por defecto, se ha determinado que se utilice el método de Welch (*BVP_to_BPM()*) para estimar la densidad espectral de la señal

de entrada BVP, lo que luego escoge como BPM a la frecuencia de máxima amplitud.

Listing 9: Método BVP a BPM.

```
def BVP_to_BPM(self):
    # -- interpolation for less than 256 samples
    _, n = self.data.shape
    if self.data.shape[0] == 0:
        return cupy.float32(0.0)
    Pfreqs, Power = Welch_cuda(self.data, self.gpuData[0], self.gpuData[2], self.gpuData[3],
                               self.gpuData[1])
    # -- BPM estimate
    Pmax = cupy.argmax(Power, axis=1) # power max
    return Pfreqs[Pmax.squeeze()]
```

Con lo descrito anteriormente, el software obtiene una estimación en BPM para la frecuencia cardíaca del paciente.

5.2.6. ¿Qué método rPPG utilizar?

Esta subsección explica el procedimiento que se utilizó para evaluar tres métodos rPPG que han sido preseleccionados para el software final. Junto con la estructura general del software establecida, se ha diseñado un procedimiento para elegir el método rPPG que se ajuste mejor para el sistema. El framework pyVHR ya posee varios métodos clásicos de rPPG implementados, por lo que es necesario elegir adecuadamente el método para el producto final, en base a la menor tasa de errores y menor desviación de su Ground Truth.

Se ha implementado un banco de pruebas en Jupyter Notebook, que consta de dos archivos:

- *metrics_generator*: su función principal es crear archivos csv para guardar los datos del funcionamiento de cada método rPPG, por cada video perteneciente a algún dataset.
- *metrics_miner*: genera métricas y gráficos en función de los archivos csv.

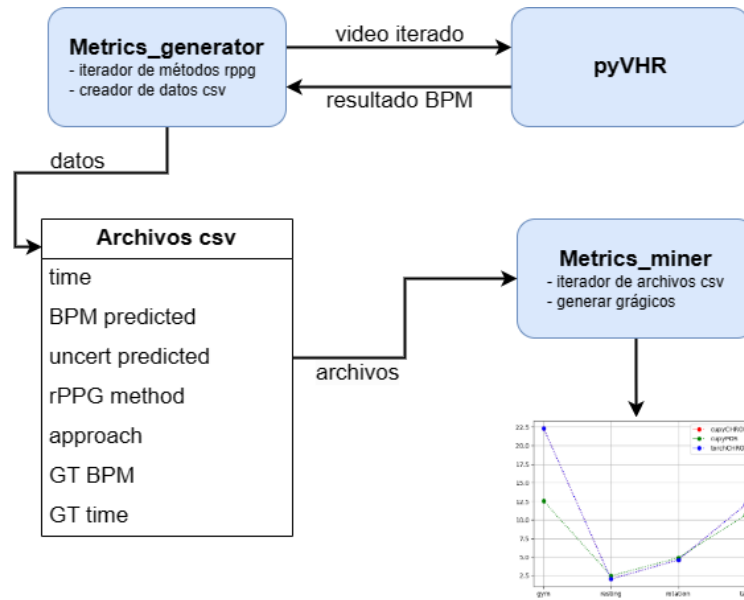


Figura 5.3: Interacción entre los módulos para generar datos y gráficos de rendimiento.

La arquitectura de pruebas se muestra en la figura 5.3. El módulo pyVHR es el encargado de recibir un video y generar los datos de ritmo cardíaco en BPM, como lo hace el software descrito anteriormente. Este módulo se itera varias veces desde metrics_generator, el cual cuenta con las siguientes funciones:

- *get_video_data* recibe como argumentos *convexhull* o *face-parsing* como el método de extracción de piel, el acercamiento (approach) y un índice. Esta función es iterada por cada video del dataset LGI_PPGI, llamando a pyVHR para procesar el video utilizando los tres métodos rppg. Los resultados se mantienen en diccionarios correspondientes a cada método, además de un cuarto diccionario con los datos del Ground Truth provenientes del etiquetado del dataset.
- *get_video_dataframes* recibe las cuatro listas mencionadas en la función anterior y son transformadas en DataFrames de Pandas.

Listing 10: Funciones de metrics generator

```
def get_video_data(met, app, video_index):
```

CAPÍTULO 5 : DESARROLLO DE LOS MÓDULOS

```
BPMs_predicted = []
times_predicted = []
uncerts_predicted = []

bpmgt, timegt = vhr.datasets.lgi_ppgi.LGI_PPGI(videodataDIR= allvideo[video_index],
        BVPdataDIR= allground[video_index]).readSigfile(filename= allground[video_index]).getBPM()
timegt = [int(item) for item in timegt]

for i in allvideo:
    txt = i.split('/')[4]
    txt_ = txt.split('_')
    title = ( txt_[0],txt_[1])

methods = ['cupy_POS', 'torch_CHROM', 'cupy_CHROM']

for method in methods:
    timepd, bpmprd, uncertpd = pipeline.run_on_video(allvideo[video_index],cuda = True,
        roi_method=met, roi_approach=app, method= method, verb=False )
    BPMs_predicted.append(bpmprd)
    times_predicted.append(timepd)
    uncersts_predicted.append(uncertpd)

data_cupy_POS = {
    'time': times_predicted[0],
    'BPM predicted': BPMs_predicted[0],
    'uncert': uncersts_predicted[0],
    'rPPG method' : methods[0],
}
data_torch_CHROM = {
    'time': times_predicted[1],
    'BPM predicted': BPMs_predicted[1],
    'uncert': uncersts_predicted[1],
    'rPPG method' : methods[1]
}
data_cupy_CHROM = {
    'time': times_predicted[2],
    'BPM predicted': BPMs_predicted[2],
    'uncert': uncersts_predicted[2],
    'rPPG method' : methods[2]
}
data_gt = {'extraction': met, 'approach': app, 'BPM gt': bpmgt, 'time gt': timegt}

return(data_cupy_CHROM, data_torch_CHROM, data_cupy_POS, data_gt)
```

```
def get_video_dataframes(data_cupy_CHROM, data_torch_CHROM, data_cupy_POS, data_gt):
    frame_cupy_POS = pd.DataFrame(data_cupy_POS)
    frame_torch_POS = pd.DataFrame(data_torch_CHROM)
    frame_cupy_CHROM = pd.DataFrame(data_cupy_CHROM)
    frame_extraction = pd.DataFrame(data_gt, index = data_gt['time gt'])
    return (frame_cupy_CHROM, frame_torch_POS, frame_cupy_POS, frame_extraction)
```

En base a estas funciones se han generado cuatro archivos csv por cada uno de los veintidós videos del dataset, que corresponden a los datos de cada método rppg evaluado.

- *time*: Tiempo del video en que se capturaron los BPM.
- *BPM Predicted*: Valor BPM estimado por el procedimiento.
- *uncert*: Algunos de los métodos rPPG generan incertezas en sus predicciones. En este caso, este atributo no fue utilizado, ya que cupyCHROM, torchCHROM y cupyPOS quedan excentos de este comportamiento.
- *rPPG method*: Se refiere al método que realizó la predicción.

El cuarto archivo csv almacena los datos del Ground Truth correspondiente al video:

- *time*: Tiempo del video en que se obtuvo los BPM.
- *BPM_gt*: valor de BPM proveniente del etiquetado de datos del dataset.

Con estos resultados se ha podido generar métricas de evaluación y gráficos descritos en la siguiente subsección. Para esto se utilizó el módulo *metrics_miner*.

5.2.7. Rendimiento

Metrics_miner utiliza varias funciones junto con los archivos csv generados en el módulo *metrics_generator*. El dataset posee los videos que corresponden a una de las seis personas realizando una de las cuatro actividades. Las actividades se describen

como *gym*, *resting*, *talk* y *rotation*, que se refieren a actividades post-gimnasio, en descanso, hablando a la cámara y rotando la cabeza respectivamente. Se han calculado los valores Mean Absolute Error (MAE) de cada persona, con respecto a su Ground Truth.

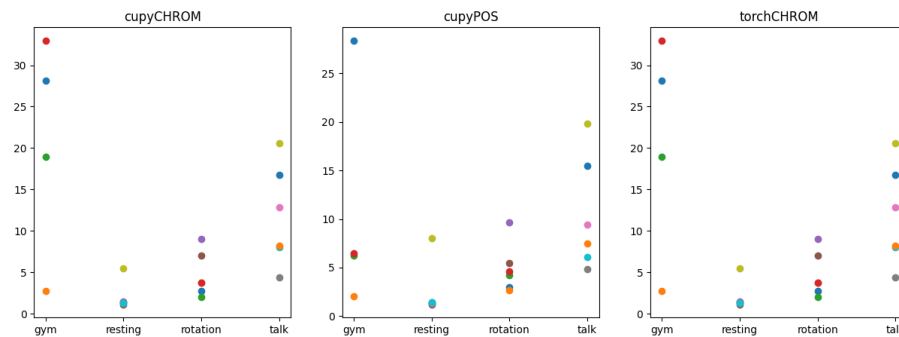


Figura 5.4: Cada punto indica el valor MAE de un video o persona, realizando una actividad.

Los valores MAE en la figura 5.4 muestran una dispersión mayor para ambos métodos CHMOD, dejando levemente una mejor evaluación a cupyPOS. Además nos ayuda a entender que los métodos suelen generar datos más erróneos en ambientes donde los pacientes muestran un mayor cansancio, como se muestran en las columnas ‘gym’.

Por otra parte se han sumado todos los valores predichos por cada video, organizados por actividad realizada, y calculado un MAE general, se ha obtenido una clara ventaja de parte por cupyPOS. Es necesario aclarar que ambos métodos CHROM en el ambiente que se han ejecutado han tenido los mismos resultados. Por esa razón es que en el gráfico de la figura 5.5 se ve la línea torch superpuesta a cupyCHROM.

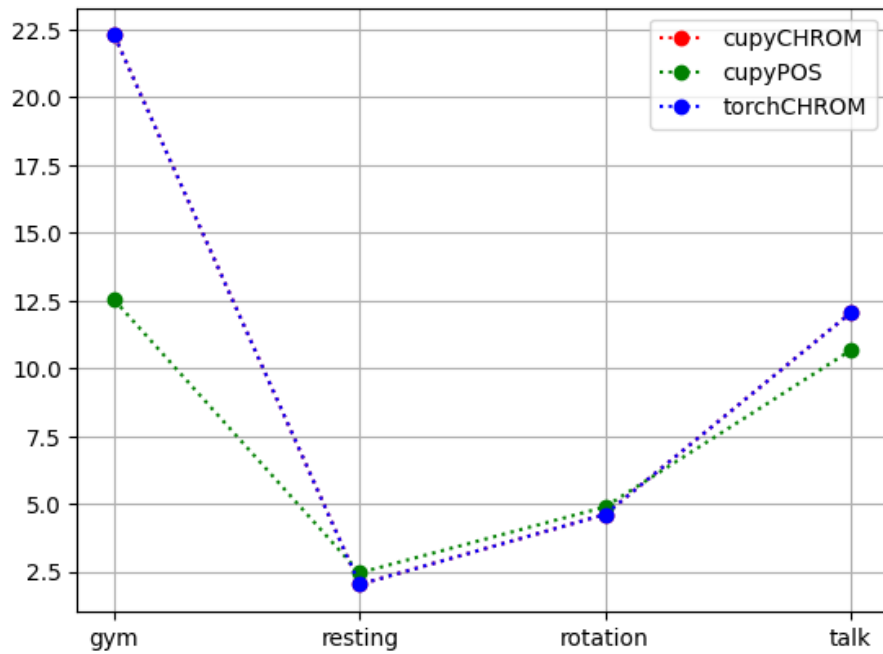


Figura 5.5: Las líneas representan los resultados MAE de cada método.

También es necesario conocer el comportamiento de los métodos utilizando algunos videos. En la imagen 5.6 se muestra un ejemplo de los gráficos generados para comparar los tres métodos interpretando los BPM correspondientes, versus su ground truth. Se observa en varios resultados que los métodos CHMOD tienen muchos saltos abruptos, lo que entorpece los resultados. Esto puede llevar a que una interpretación en tiempo real sea más errónea.

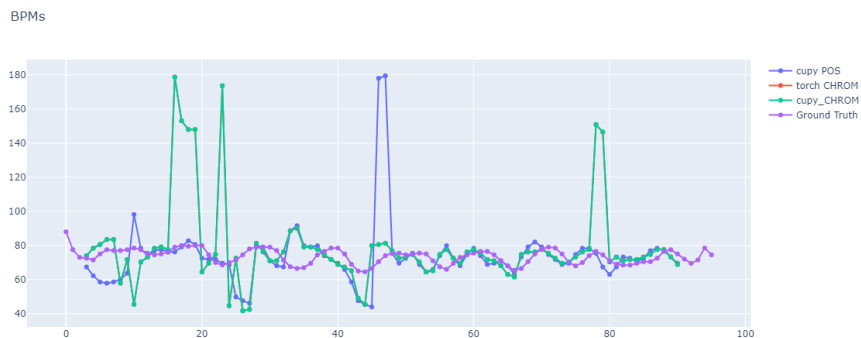


Figura 5.6: BPM predicho versus BPM ground truth.

CAPÍTULO 5 : DESARROLLO DE LOS MÓDULOS

En conclusión, el método cupyPOS ha quedado seleccionado como método predefinido para el software.

5.3. Módulo: Detección de Nivel de Atención

5.3.1. Métricas de cuantificación atencional

Antes iniciar el desarrollo del presente módulo, la investigación previa ha dado con el inconveniente de que no existe ninguna métrica que cuantifique “el nivel de atención del paciente” en tiempo real. Alternativamente existen diversos métodos para la evaluación de algunas situaciones psico-patológicas que llevan a la evaluación de un paciente para su diagnóstico, no en tiempo real, ya que requiere de varios procedimientos con evaluaciones de un profesional. Un ejemplo sería evaluar a un paciente con Trastorno por Déficit de Atención e Hiperactividad (TDAH), fácilmente reconocible como un acercamiento bastante alejado a lo que FALP solicita como detector de atención en tiempo real.

No obstante, se han encontrado alternativas de software que toman algunas características faciales de la persona, haciendo una estimación de si está poniendo más o menos atención, con una calibración previa. También se han encontrado detectores de atención para conductores que no requieren de esta calibración, basándose en el uso de la posición de la cabeza, hacia dónde está mirando y si el paciente se encuentra pestañando a un ritmo acelerado o no.

Se ha desarrollado un software que posee un pipeline similar al descrito en el módulo CarMod. Se estructura de la siguiente forma:

1. **Captura de rostro.** Utilizando Mediapipe FaceMesh, al igual que en CarMod, se obtiene la información facial a través de los landmarks generados. Con la información encontrada en la web, se ha encontrado que los landmarks de interés están enumerados en base a la imagen que se muestra en la figura 5.2.

2. **Rescate de zona de interés.** Para este módulo, la zona de interés está determinada por una cantidad de landmarks específicos. Se ha encontrado en base a la figura 5.2 que los ojos y boca [27], los cuales son necesarios para detectar bostezos y parpadeos, están determinados por los números, o identificadores. Los puntos clave o de interés

CAPÍTULO 5 : DESARROLLO DE LOS MÓDULOS

(keypoints) correspondientes a los ojos están almacenados en las listas que se muestran en el código siguiente:

Listing 11: landmarks como keypoints para los ojos.

```
self.LEFT_EYE =[ 362, 382, 381, 380, 374, 373, 390, 249, 263, 466, 388, 387, 386, 385,384,
                 398 ]
self.RIGHT_EYE=[ 33, 7, 163, 144, 145, 153, 154, 155, 133, 173, 157, 158, 159, 160, 161 ,
                 246 ]
self.LEFT_IRIS = [474, 475, 476, 477]
self.RIGHT_IRIS = [469, 470, 471, 472]
```

Los ojos izquierdo y derecho que han sido denominados por el valor *LEFT_EYE* y *RIGHT_EYE*, análogamente de sus iris por *LEFT_IRIS* y *RIGHT_IRIS*. Vale mencionar que las iris no fueron utilizadas para este trabajo, pero quedan verbalizadas para una posible mejora del producto.

Por otra parte, y de igual modo, se han determinado los keypoints correspondientes a los labios. En este caso, se tuvo que diferenciar los labios interiores de los exteriores. Con los labios interiores se refiere a la zona que delimita los labios en su punto de unión al tener la boca cerrada, mientras que los exteriores son aquellos que delimitan la zona más alta del labio superior, y la más baja del labio inferior.

Listing 12: landmarks como keypoints para los labios.

```
self.lipsUpperOuter = [61, 185, 40, 39, 37, 0, 267, 269, 270, 409, 291]
self.lipsLowerOuter = [146, 91, 181, 84, 17, 314, 405, 321, 375, 291]
self.lipsUpperInner = [78, 191, 80, 81, 82, 13, 312, 311, 310, 415, 308]
self.lipsLowerInner = [78, 95, 88, 178, 87, 14, 317, 402, 318, 324, 308]
```

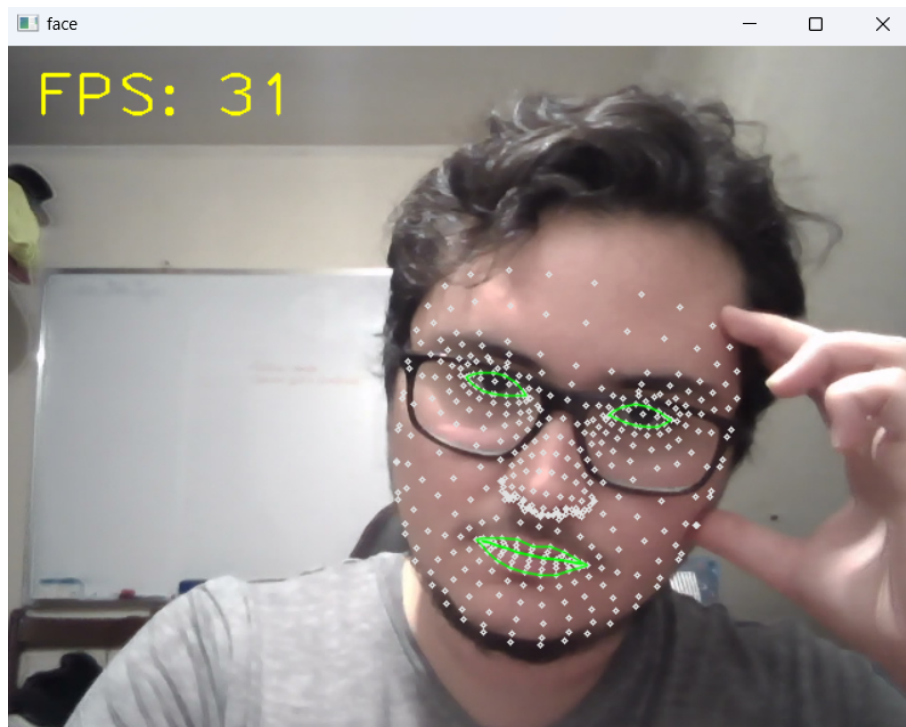


***Figura 5.7:** Cada punto se encuentra enumerado, lo cual sirve como índice o identificador del landmark. También se muestran los labios y ojos remarcados.*

3. **Valores de medición para el conteo de datos.** Los valores de medición se han capturado en base a los landmarks mencionados anteriormente. Para esto, es necesario obtener una relación que se basa en la distancia de dos puntos horizontalmente, y otra distancia entre dos puntos verticalmente. Esto sirve para determinar si los ojos o boca están abiertos o cerrados.

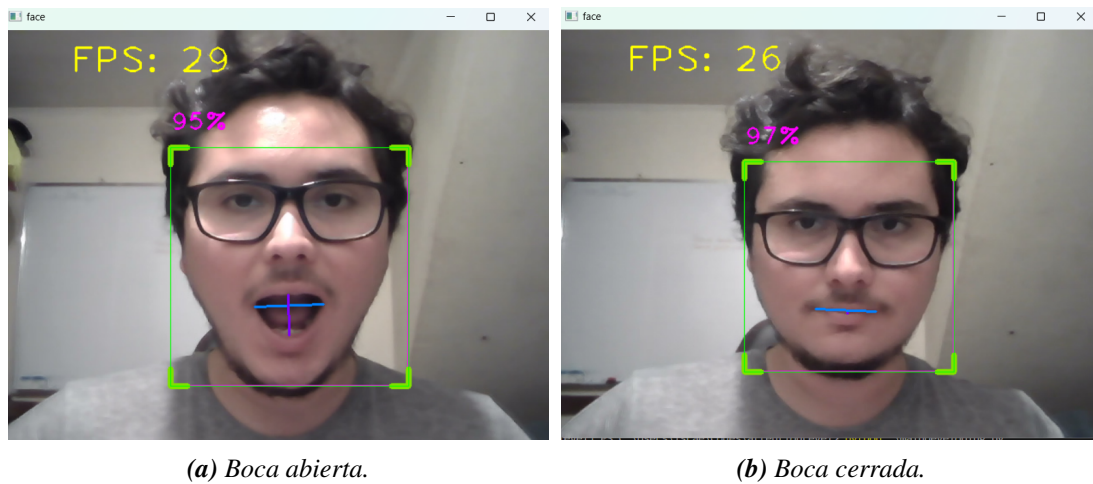


Figura 5.8: Las líneas muestran la distancia vertical y horizontal de la boca.

Primero, se obtiene la *Relación de aspecto de la boca*.

Listing 13: landmarks como keypoints para los labios.

```
def mouth_aspect_ratio(self, img, landmarks, draw = True):
    # Horizontal key points
    mh_right = landmarks[self.lipsUpperInner[0]] # 78
    mh_left = landmarks[self.lipsUpperInner[-1]] # 308

    # Vertical key points
    # upper lip
    mv_upper = landmarks[self.lipsUpperInner[5]] # 13
    # lower lip
    mv_lower = landmarks[self.lipsLowerInner[5]] # 14

    h_distance = self.euclidean_distance(mh_right, mh_left)
    v_distance = self.euclidean_distance(mv_upper, mv_lower)

    if draw:
        cv2.line(img, mh_right, mh_left, (255,128,0), 2)
        cv2.line(img, mv_upper, mv_lower, (255,0,128), 2)

    if v_distance == 0:
        v_distance = 0.000001
    mouth_ratio = (h_distance/v_distance)
    return mouth_ratio
```

Para obtener las distancia euclidiana vertical, se seleccionaron los puntos centrales de

CAPÍTULO 5 : DESARROLLO DE LOS MÓDULOS

los labios interiores superior e inferior. En el caso de la distancia horizontal, los puntos seleccionados son las uniones laterales de los labios superior e inferior. Para esto, se determinaron los landmarks 13 y 14 para el caso vertical, 78 y 308 para el caso horizontal. Por último, la relación se calcula como el cociente entre las distancia horizontal y vertical. En la figura 5.8 se muestran las líneas proyectadas cuando la boca está abierta o cerrada, lo cuál se relaciona directamente con la relación de aspecto de la boca calculada anteriormente.

Este funcionamiento es análogo al cálculo de relación de aspecto para los ojos, el cual se puede ver en la figura 5.9

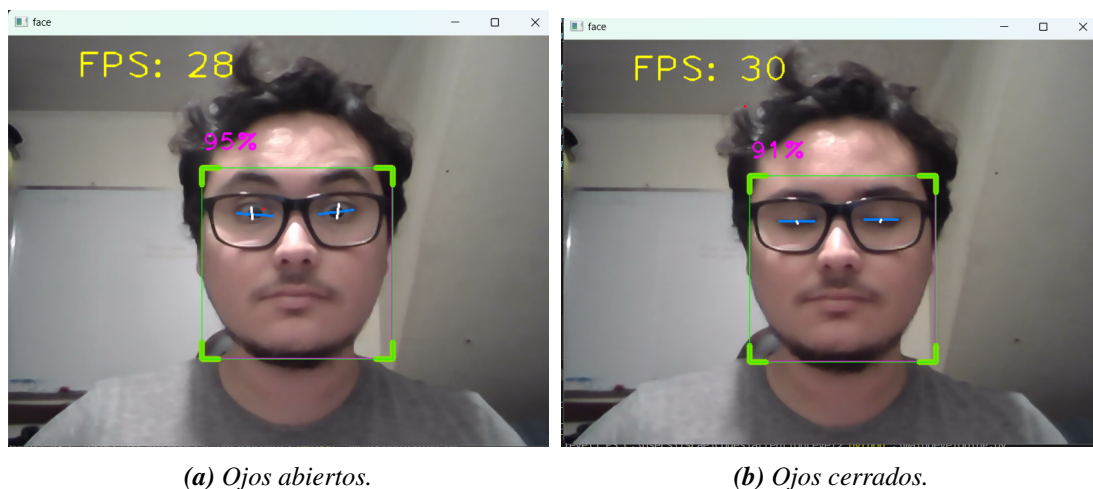


Figura 5.9: Las líneas muestran la distancia vertical y horizontal de la boca.

4. Posición de la cabeza La posición de la cabeza ha ayudado a distinguir si el paciente se encuentra mirando a la pantalla o no, lo que sería también un factor para el nivel de atención. Para esto se utilizan técnicas propuestas por un código abierto en [28], un software propuesto para detectar el nivel de atención en las salas de clases, utilizando bostezos, parpadeos y vista al frente. Este trabajo tiene similitudes con el actual módulo, pero su principal diferencia es en el uso de MediaPipe en lugar de Dlib, para obtener un rendimiento más responsivo y rápido, y algunas diferencias con los pesos en las relaciones de concavidad.

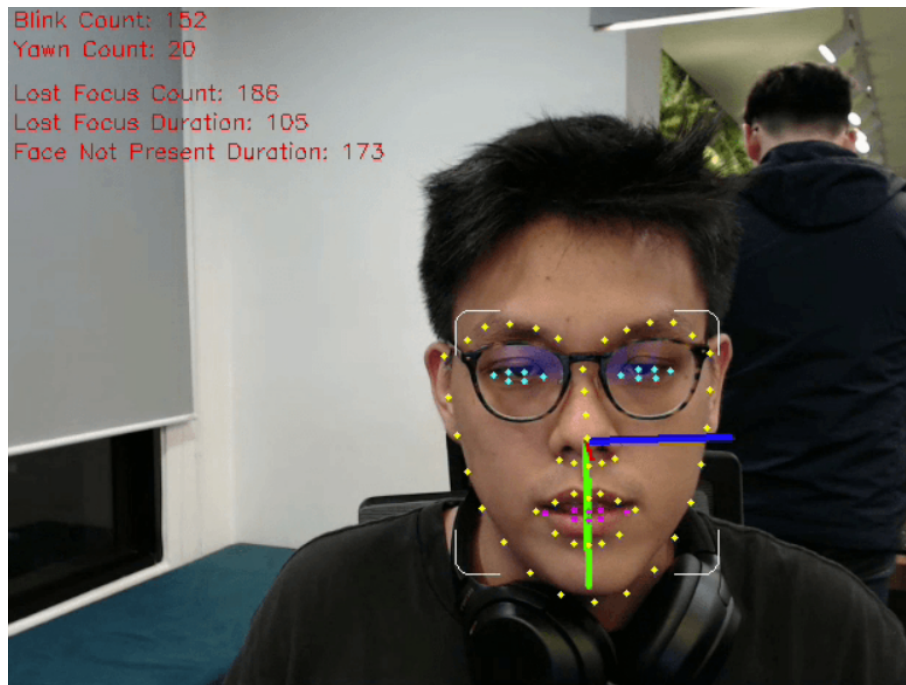


Figura 5.10: Ejemplo de OCAT.

5.3.2. Expresión matemática propuesta

Con los datos obtenidos anteriormente, se ha definido una función matemática para obtener un valor numérico de atención en tiempo real. Como se mencionó antes, los factores que influyen en la detección de atención, tanto en este software como en la vida cotidiana, son los parpadeos, cantidad de bostezos y la inclinación de la cabeza. Los pesos de cada uno de estos aspectos han quedado propuestos para futuras modificaciones, ya que esto ha quedado abierto para una investigación en relación a cómo las personas identifican si una persona está poniendo atención o no.

Los atenuadores se obtienen en función del tiempo, en donde los símbolos tienen los siguientes significados:

- yc_t : contador de bostezos.
- lfc_t : contador “sin vista al frente”.
- fnp_t : contador de ausencia de rostro.

- bfi : frecuencia de parpadeos.

yc_t se obtiene con la cantidad de bostezos que ha habido hasta el momento t : $yc_t = y(t)$. Del mismo modo, lfc_t se obtiene como la cantidad de veces que el paciente no ha tenido la vista al frente, y el tiempo en milisegundos que esto ocurre. Se puede interpretar con la siguiente expresión:

$$lfi = \sum_{t=1}^{ct} lf(t), t > 3000[ms]$$

, donde

$$lf(t) = \begin{cases} 1 & \text{sin vista al frente} \\ 0 & \text{con vista al frente} \end{cases}$$

donde ct es “current time” o tiempo actual. La variable binaria $lf(t)$ es una función que toma el valor 1 cuando el paciente no mira al frente, y 0 cuando sí lo hace. Luego, dado que el tiempo se expresa en milisegundos, se procedió a normalizar la ecuación, manteniendo un mínimo de -1 puntos de atenuación para la expresión final.

$$lfc_t = \max[-1, \frac{lfi}{1000}]$$

El caso de fnp_t es similar al anterior, pero el valor binario viene dado por la presencia de rostro. Lo mismo se ha determinado para yc_t .

$$fnp(t) = fnp_t = \begin{cases} 1 & \text{rostro ausente} \\ 0 & \text{rostro presente} \end{cases}$$

$$yc(t) = yc_t = \begin{cases} 1 & \text{paciente bosteza} \\ 0 & \text{paciente no bosteza} \end{cases}$$

CAPÍTULO 5 : DESARROLLO DE LOS MÓDULOS

Si el rostro ausente o el bostezo dura menos de un segundo, no es relevante para el cálculo y pasa desapercibido.

Finalmente, la variable de atenuación ATN se calcula en base a las variables anteriores cada un segundo (1000ms), mientras que las variables son modificadas cada milisegundo. Es importante mencionar que, para calcular ATN , es necesario esperar a menos tres segundos.

$$ATN = 4 \times [yc_t - yc_{t-3000}] + 4 \times [lfc_t - lfc_{t-1000}] + 0,5 \times [fnp_t - fnp_{t-1000}] + 0,9 \times [bf_t]$$

Como ATN es un atenuador, se ha determinado que el nivel de atención estandar y máximo es 60, por lo que el resultado se expresa de la siguiente forma:

$$\text{result} = \max[0, (60 - ATN)]$$

La razón por la que se ha determinado el valor 60 es parcialmente arbitrario, ya que este valor hace referencia a un total con respecto al sistema de medición del tiempo. 60 hace referencia a un 100 %, lo que puede interpretarse como una cantidad de segundos, dentro de un minuto en que el paciente ha estado atento.

6. Resultados

En esta sección se muestran los resultados del funcionamiento del software.

6.1. CarMod

En primer lugar, los resultados correspondientes al Módulo de ritmo cardíaco (CarMod) se basan en las pruebas realizadas utilizando la estructura descrita en la sección anterior. Estos resultados, presentados en la Tabla 1, muestran el comportamiento de cada uno de los tres métodos preseleccionados, antes de decidir por el método “cupy-POS” como principal.

Tabla 1: MAE por actividad, de cada método rPPG.

Método:	CHROM		POS
Biblioteca:	cupy	torch	cupy
gym	22.279013	22.279013	12.518855
resting	2.027850	2.027850	2.457279
rotation	4.611313	4.611313	4.887072
talk	12.044157	12.044157	10.658393

Gracias a la biblioteca Pandas en Python, el tratamiento de resultados se ha podido trabajar como DataFrame y, con ello, se han exportado las tablas que se encuentran en esta sección. Un ejemplo de código para realizar esta extracción es la siguiente donde, luego de la obtención de datos, agrupación y ordenamiento desde los csv, y creación de DataFrames como tablas, funciones como *DataFrame.to_latex()* exporta una sección de código en LaTeX, lo que ha agilizado el proceso para transferir los datos a este documento.

Listing 14: Obtención de la tabla

```
print(mae_by_method_act)
'''
```

```

{'cupyCHROM': {'gym': 22.279012926706827, 'resting': 2.0278497409326426, 'rotation':
4.611313215648855, 'talk': 12.044157045077434}, 'cupyPOS': {'gym': 12.518855107931726,
'resting': 2.457279185556995, 'rotation': 4.887071604643766, 'talk': 10.65839325221239},
'torchCHROM': {'gym': 22.279012926706827, 'resting': 2.0278497409326426, 'rotation':
4.611313215648855, 'talk': 12.044157045077434}}
"""

table = pd.DataFrame(data=mae_by_method_act)
print(table2.iloc[:,[0,2,1]].to_latex(index=True)) ## obtiene la equivalencia en LaTeX.

```

Este dato ha sido útil para generar la tabla 1. Se ha calculado el error de cada video perteneciente al dataset LGI_PPGI, en base a una suma total separado por actividad. Esto quiere decir que todos resultados de los videos marcados como `_gym` se han sumado, por cada uno de los métodos rPPG. Luego, se obtuvo el MAE en relación a sus correspondientes Ground Truth.

Otra forma de ver los resultados es el MAE por cada uno de los videos del dataset, es decir, cada persona separado por actividad. La tabla 2 muestra esta información.

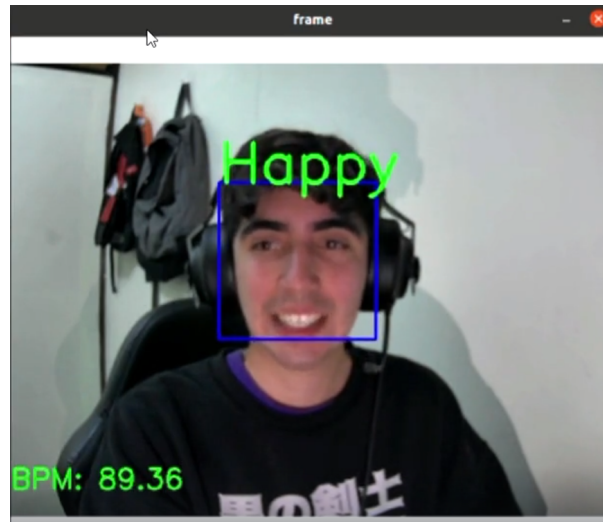


Figura 6.1: Carmod funcionando junto a la detección de emociones.

Las imágenes de las figuras 6.1 y 6.2 muestran el funcionamiento del sistema trabajando la detección de emociones junto a CarMod. La segunda figura muestra una emoción obtenida, pero los BPM aún no son calculados. Esto se debe a que el sistema calcula los BPM cada 10 segundos. Los frames pertenecientes a esta ventana de tiempo

Tabla 2: MAE por cada persona y actividad.

Método:	CHROM		POS
Biblioteca:	cupy	torch	cupy
alex_gym	28.126116	28.126116	28.389220
alex_resting	1.426499	1.426499	1.269388
alex_rotation	2.725968	2.725968	2.962862
alex_talk	12.869207	12.869207	9.391468
angelo_resting	1.126345	1.126345	1.200628
angelo_rotation	3.789290	3.789290	2.642383
angelo_talk	4.417473	4.417473	4.859054
cpi_gym	2.703377	2.703377	2.074288
cpi_resting	1.351475	1.351475	1.366925
cpi_rotation	2.023099	2.023099	4.249219
cpi_talk	20.578331	20.578331	19.844881
david_gym	18.974223	18.974223	6.244331
david_resting	1.417572	1.417572	1.446140
david_rotation	3.702684	3.702684	4.632812
david_talk	8.033781	8.033781	6.069267
felix_resting	5.461839	5.461839	8.000421
felix_rotation	8.999622	8.999622	9.642799
felix_talk	16.731651	16.731651	15.467696
harun_gym	32.980449	32.980449	6.475922
harun_resting	1.330848	1.330848	1.394235
harun_rotation	7.038792	7.038792	5.443990
harun_talk	8.237041	8.237041	7.444785

son utilizados en conjunto por CarMod para hacer un cálculo de un promedio de BPM en los últimos 10 segundos. Si se aumenta el tamaño temporal, la predicción podría ser más adecuada, pero se produce un funcionamiento más lento del sistema al generar sus respuestas.



Figura 6.2: Carmod y emociones, calculando BPM.

6.2. AtMod

Por otro lado, los resultados de AtMod se han obtenido luego de que el sistema de detección de emociones esté totalmente implementado. A pesar de esto, los resultados que se muestran a continuación son imágenes lado a lado con la información visible e invisible para el profesional de la salud.

El “lado invisible” (izquierdo) muestra los datos que utiliza AtMod para calcular el nivel de atención, mientras que el “lado visible” (derecho) sólo muestra el resultado de AtMod, un número que el profesional de la salud puede interpretar como un nivel de atención del paciente.

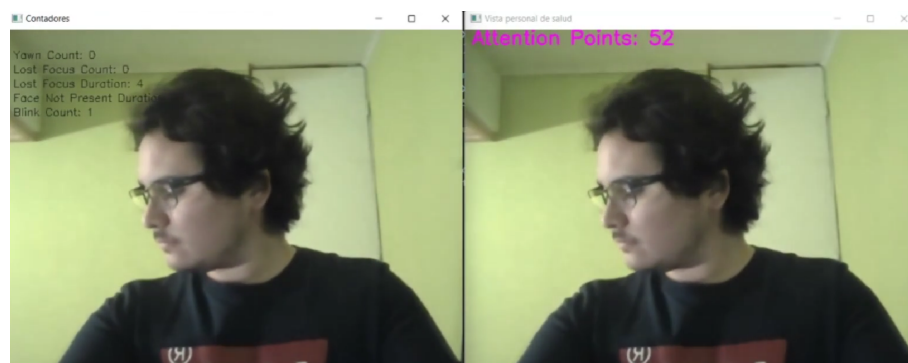


Figura 6.3: AtMod mostrando una caída de atención.

CAPÍTULO 6 : RESULTADOS

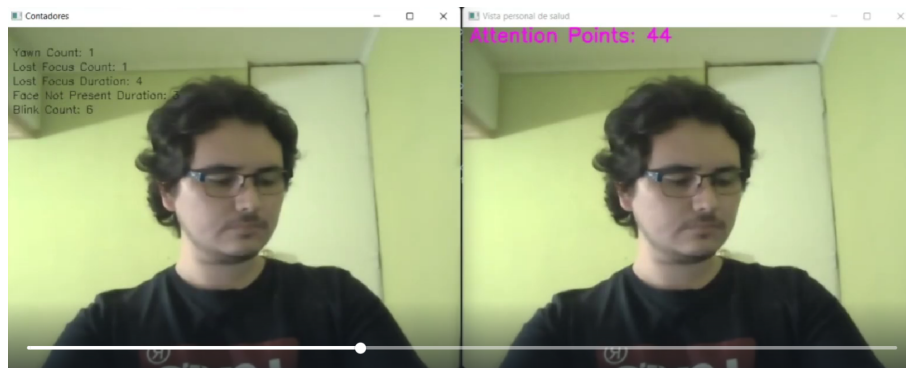


Figura 6.4: AtMod mostrando otra caída de atención, luego de la primera.

En las figuras 6.3 y 6.4 simula el comportamiento de un paciente que ha perdido puntos de atención por no mirar hacia la pantalla fijamente y luego de un bostezo. El nivel de atención calculado ha tenido la influencia de los parámetros mostrados en el lado izquierdo.

Luego, en la figura 6.5, el paciente ha recuperado algunos puntos de atención, ya que este ha empezado a mirar la pantalla fijamente.

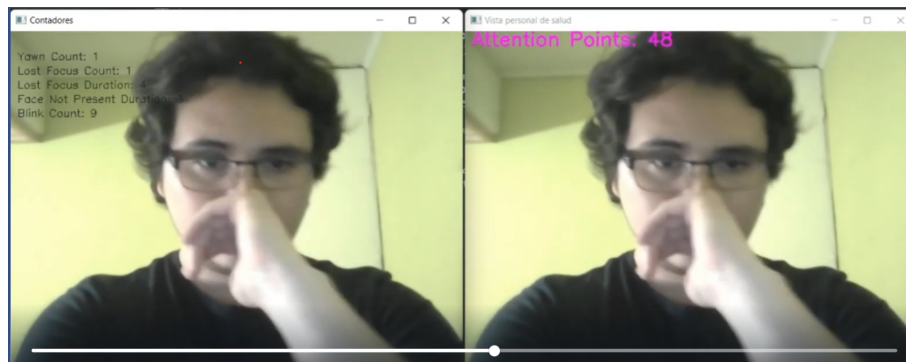


Figura 6.5: AtMod mostrando una recuperación de atención.

7. Verificación y Validación

La sección presente muestra la validación de los resultados expuestos en la sección anterior.

Como se ha visto en la Sección 4, se han determinado los requerimientos funcionales y no funcionales para la totalidad del sistema. También se ha mencionado que este documento aborda dos de los cuatro módulos del sistema: CarMod y AtMod. Por lo tanto, la verificación realizada para este apartado está relacionada con los requisitos específicos.

El siguiente listado describe el cumplimiento de los requerimientos relacionados a CarMod y AtMod:

RF0: Identificar el nivel de ritmo cardíaco del paciente a través de rPPG. El cliente ha quedado satisfecho con el resultado de precisión de CarMod. Se han calculado los MAE relacionados a los BPM predichos en comparación con su Ground Truth, como se ha visto en la sección de resultados, los valores más bajos son aquellos que corresponden a actividades no relacionadas a la actividad física. Es aquí donde cupyPOS ha tenido los mejores resultados como se ve en la tabla 2, y, más general, en la tabla 1.

RF1: Mostrar porcentualmente, la atención del paciente mientras habla el médico. Este requerimiento está relacionado con AtMod. Si bien no ha sido posible medir la precisión de los niveles de atención, el cliente ha quedado satisfecho con el resultado. La razón de este problema es debido a la inexistencia de una métrica que nos permita comparar el nivel de atención con su predicción en AtMod. Este es un tema que ha quedado abierto a una nueva investigación, en donde se podrá proponer una métrica cuyos resultados sean computados con técnicas de ML.

Por otra parte, los módulos que no pertenecen a este documento, han abordado los siguientes requisitos:

CAPÍTULO 7 : VERIFICACIÓN Y VALIDACIÓN

RF0: Se activa el sistema con una conexión previa para la teleconsulta.

Módulo API.

RF1: Internamente estará el módulo de detección de emociones capturando el sentimiento del paciente.

Módulo de emocionalidad.

RF2: Se genera a un costado de la teleconsulta un cuadro donde se exhibirá la información obtenida, es decir, la emoción actual del paciente, su nivel de atención y ritmo cardíaco estimado.

Módulo API.

RF3: Generar un reporte con las mediciones obtenidas en la sesión. Módulo API.

RF4: Luego de la consulta, y posterior a la aprobación del médico, idealmente el sistema enviará el video al servidor para su retroalimentación y así mejorar los resultados en sesiones futuras.

Módulo API.

8. Conclusiones

En este documento se ha presentado el diseño y desarrollo de dos módulos para potenciar la interpretación de las emociones de un paciente en una consulta remota oncológica, los cuales han sido los módulos de interpretación del ritmo cardíaco y el nivel de atención del paciente. Como se observó en la sección 6, los resultados han cumplido con los requerimientos, logrando detectar con éxito el ritmo cardíaco en BPM, utilizando técnicas de Remote Photoplethysmography (rPPG) y Visión por Computador (CV), con una tasa de errores suficientemente baja para ser utilizado en una teleconsulta en tiempo real. También, dado que las métricas de atención son inexistentes, el cliente ha quedado satisfecho con el modelo matemático propuesto en este documento, al final de la sección 5. Con estos módulos desarrollados, en conjunto con el módulo de emocionalidad, ha sido posible una implementación dentro del módulo API, para que los datos de predicción sean capturados dentro del framework que utiliza FALP para sus teleconsultas. No obstante, esa sección no es parte de este documento ni de lo que el cliente ha pedido, ya que solo ha solicitado predicción y envío de datos dentro de una API.

Trabajos futuros:

El trabajo realizado ha cumplido con lo solicitado, pero aún así pueden haber mejoras en los módulos. Por la parte de CarMod, se podría optimizar el pipeline para que la predicción pueda utilizar más de un método rPPG simultáneo, para que el resultado en BPM sea potenciado por más de un algoritmo, como CHMOD y POS.

Por otra parte, el módulo de atención aún tiene aspectos pendientes. Principalmente, la inexistencia de métricas para interpretar el nivel de atención ha llevado al escritor de este documento a planificar una propuesta para determinar una métrica que sea capaz de interpretar el nivel de atención de una persona. Esta planificación tiene los siguientes pasos:

CAPÍTULO 8 : CONCLUSIONES

1. Proponer un modelo de atención arbitrario, basado en lo visto en este documento.
2. Generar un dataset con varias personas que identifiquen su calidad de atención.
3. Encuestar al público para que otorgue un puntaje de atención de varios videos que muestren rostros humanos.
4. Utilizar tecnicas de ML para obtener un modelo matemático que relacione estos resultados.
5. Proponer el modelo a entidades psicológicas y médicas.

Cabe destacar Si esta planificación da resultados positivos, el modelo será propuesto a la comunidad científica, y así se podrá mejorar el software desarrollado en este documento.

Referencias

- [1] Leslie Lamport, *LaTeX: a document preparation system*, Addison Wesley, Massachusetts, 2nd edition, 1994.
- [2] John Camm, Marek Malik, *Heart rate variability: Standards of measurement, physiological interpretation and clinical use. Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology*, 1996.
- [3] Gang Li, Wan-Young Chung, *Detection of Driver Drowsiness Using Wavelet Analysis of Heart Rate Variability and a Support Vector Machine Classifier*, 2013.
- [4] Dasari A., *HeartBeat: Measuring heart rate using remote photoplethysmography (rPPG)*, 2021, Marzo 24
- [5] G. Boccignone, D. Conte, V. Cuculo, A. D'Amelio, G. Grossi, R. Lanzarotti, E. Mortara, *pyVHR: a Python framework for remote photoplethysmography*, Abril 2022.
- [6] Goodfellow. *FER2013 (Facial Expression Recognition 2013 Dataset)*, Obtenido de: FER2013 Dataset, Papers With Code, 2013
- [7] De Haan, G., & Jeanne, V., *Robust pulse rate from chrominance-based rPPG. IEEE Transactions on Biomedical Engineering*, 2013
- [8] Pilz, C. S., Zaunseder, S., Krajewski, J., & Blazek, V., *Local group invariance for heart rate estimation from face videos in the wild*, 2018
- [9] R. Meziati Sabour, Y. Benezeth, P. De Oliveira, J. Chappé, F. Yang., *UBFC-Phys: A Multimodal Database For Psychophysiological Studies Of Social Stress*, IEEE Transactions on Affective Computing, 2021.

- [10] Lewandowska M., Rumiński J., Kocejko T., & Nowak J., *Measuring pulse rate with a webcam—a non-contact method for evaluating cardiac activity*, Septiembre 2011.
- [11] Wang, W. den Brinker A. C. Stuijk, S. & de Haan G., *Algorithmic principles of remote PPG*, IEEE Transactions on Biomedical Engineering 64 (7) (J) 1479-1491.
- [12] Talha Anwar, *Mental Stress PPG*, Obtenido de: Mental Stress PPG | Kaggle, 2022, Mayo 16.
- [13] A. Dasari, S. K. A. Prakash, L. A. Jeni, C. S. Tucker, *Evaluation of biases in remote photoplethysmography methods*, npj Digital Medicine.
- [14] T. Robal, Y. Zhao, C. Lofi, C. Hauff, *Towards Real-time Webcam-based Attention Tracking in Online Learning*, Marzo 2018.
- [15] S. Bobbia, R. Macwan, Y. Benezeth, A. Mansouri, J. Dubois, *Unsupervised skin tissue segmentation for remote photoplethysmography*, Pattern Recognition Letters, 2017.
- [16] F. Calvo, Recio, Ludqvist, *Human Observers and Automated Assessment of Dynamic Emotional Facial Expressions: KDEF-dyn Database Validation*, Frontiers in Psychology, 2018.
- [17] Lixu Wang, Shichao Xu, Xiao Wang, Qi Zhu, *Eavesdrop the Composition Proportion of Training Labels in Federated Learning*, October 2019.
- [18] Center for Disease Control and Prevention, Telemedicina, ¿qué es y por qué debería importar? 5 de agosto 2020.
- [19] Pan American Health Organization, Teleconsulta durante la pandemia.
- [20] How Patient Empowerment Can Improve Health Outcomes May 19, 2020

- [21] Instituto Nacional del Cancer NCI, Los sentimientos y el cáncer, 20 de Agosto 2020.
- [22] M. Mather, J. Thayer How heart rate variability affects emotion regulation brain networks, Feb 1, 2019.
- [23] Poh, M. Z., McDuff, D. J., & Picard, R. W., *Non-contact, automated cardiac pulse measurements using video imaging and blind source separation*, 2010.
- [24] MediaPipe Face Mesh, Sep 9, 2022.
- [25] Hyungjoon K, Hyeonwoo K, Seongkuk C, Eenjun H, *An end-to-end face parsing model using channel and spatial attentions*, 15 March, 2022.
- [26] Using modified BiSeNet for face parsing in PyTorch, May 2019.
- [27] Extracción de repositorio: Pre-trained TensorFlow.js models, May 13, 2021
- [28] Online Classroom Attention Tracker (OCAT), Jul 23, 2020