

DESARROLLO DE UNA HERRAMIENTA PARA LA EVALUACIÓN DE LA FISIOLOGÍA VISUAL USANDO ELECTROENCEFALOGRAFÍA PORTABLE Y DE BAJO COSTO.

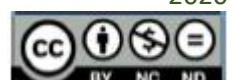
Proyecto de Investigación – Pregrado Bioingeniería- 2020

Verónica Henao Isaza

UNIVERSIDAD
DE ANTIOQUIA

1 8 0 3

Universidad de Antioquia
Facultad de ingeniería
Programa de Bioingeniería
Medellín, Antioquia
2020



DESARROLLO DE UNA HERRAMIENTA PARA LA EVALUACIÓN DE LA FISIOLOGÍA VISUAL
USANDO ELECTROENCEFALOGRAFÍA PORTABLE Y DE BAJO COSTO.

Por:

Verónica Henao Isaza

Informe de práctica Académica
Modalidad: Proyecto de Investigación.

Asesor:

John Fredy Ochoa Gómez

Ingeniero de Sistemas e Informática,
Doctor en Ingeniería Electrónica,
Magister en Ingeniería

Docente en la Universidad de Antioquia.

UNIVERSIDAD
DE ANTIOQUIA

Universidad de Antioquia

Facultad de Ingeniería

Programa de Bioingeniería

Medellín, Antioquia

2020

Contenido

Resumen.....	3
Introducción.....	3
Objetivos.....	6
General:	6
Específicos:.....	6
Marco Teórico.....	6
Sistema visual.....	6
Electroencefalografía (EEG)	7
Potenciales evocados en estado estacionario (SSVEP)	9
Agudeza visual (AV).....	9
Agudeza de Vernier (EEG).....	9
Métodos de procesamiento de señales EEG	10
Método de Welch	12
Diseño y usabilidad de una interfaz de usuario	13
Metodología	16
Descripción	16
Resultados y análisis	17
Implementación del paradigma	17
Estimulación – Agudeza.....	17
Diseño de algoritmos	20
Diseño de interfaz	23
Conclusiones.....	27
Referencias bibliográficas.....	33

UNIVERSIDAD
DE ANTIOQUIA

1 8 0 3

Resumen.

La evaluación de la fisiología visual se ha realizado durante varios años con diferentes tecnologías y herramientas que varían significativamente en costos y precisión. Una de las medidas más estudiadas en este campo es la agudeza visual, ya que esta es fundamental en la examinación oftalmológica con una relevancia eminente en el diagnóstico de patologías, decisiones medico legales y procesos de investigación clínica y de ciencias básicas. Entre las tecnologías de mayor uso para este tipo de evaluación visual se encuentran las cartas con optotipos (Snellen, HOVT, Jaeger, Bailey-Lovie, Pigassou, E Snellen, etc.) el problema con estas cartas es la subjetividad de los resultados, donde la validez de los resultados críticos depende de la fiabilidad de la respuesta de paciente. Los potenciales evocados visuales permiten disminuir la subjetividad de las pruebas de agudeza visual, sin embargo, no deja de ser un campo de investigación amplio que requiere de una gran cantidad de pruebas que permitan generar una base de datos amplia y obtener resultados confiables. Se han realizado diversos estudios referentes a la estimulación, adquisición y procesamiento llevados a cabo para la evaluación visual con potenciales evocados visuales, muchos de ellos con resultados prometedores respecto a la implementación de este tipo de evaluación visual por computador, lograr la repetibilidad de estos métodos con un equipo portable y de bajo costo es un reto que se ha ido desarrollado y por el cual se establece la necesidad de una herramienta que permita agilizar el proceso de adquisición bajo estimulación visual acompañado de un protocolo de entorno en el que se observe el comportamiento de la señal de cada sujeto durante la estimulación y que entregue resultados que puedan ser aplicados en diferentes procesamientos reportados en la literatura.

Introducción.

La pérdida de la visión, incluyendo la ceguera, se mantiene como una causa significativa de discapacidad a nivel mundial. La agencia internacional para la prevención de la ceguera (IABD por sus siglas en inglés) reporta que una enfermedad tratable fue la causa de pérdida de la visión en cuatro de cada cinco personas afectadas y su prevalencia aumenta con la edad. Por lo tanto, el aumento de la esperanza de vida a nivel mundial ha aumentado el número de personas mayores con discapacidad visual [1]. Para distinguir entre las personas que cuentan con un tipo de discapacidad visual y las personas sanas, se hace necesario definir la salud visual como la ausencia de las alteraciones visuales que impidan al ser humano conseguir un estado físico, cultural, estructural y funcional de bienestar social y también es considerada como la ausencia de enfermedad ocular, acompañada de una buena **agudeza visual** [2], adicionalmente

el procesamiento de la información a nivel neuronal en el cerebro juega un papel fundamental en la capacidad que tiene el ser humano para ver el mundo y perder el sentido de la vista significa limitar la percepción de los estímulos externos significativamente.

En Colombia la persistencia de problemas visuales debidos a la falta de sensibilización, detección y tratamiento también impide a una parte de la población tener acceso a las **nuevas tecnologías**, y posibilitar un mayor avance tecnológico en el país [1]. El 90% de las personas ciegas viven en países en vía de desarrollo y el 80% de los problemas visuales presentados, se podrían evitar (50% de las causas es tratable y el 30% es prevenible a través de acciones de nivel primario y secundario) [2], [3]. Se estima que, para un país con las características socioeconómicas de Colombia, según datos de la organización mundial de la salud (OMS) existen aproximadamente 7000 personas ciegas por cada millón de habitantes, las cuales en su mayoría presentan ceguera por causas prevenibles o curables [4].

Examinar y medir objetivamente las respuestas neurológicas de las vías visuales permite ahondar mucho más que un simple examen de visión. Esto se logra gracias a los potenciales evocados visuales (VEP) que generan una respuesta a un estímulo determinado [5]–[7] y que son evaluados por medio de la electroencefalografía, una técnica que registra la actividad eléctrica del cerebro de manera no invasivo [8]–[10] y que proporciona información objetiva sobre la funcionalidad del sistema visual, desde el lente hasta la corteza visual, detectando anomalías mecánicas o neurológicas relacionadas con la visión [7].

Ahora bien, gran parte de la experimentación en la neurociencia actual se da por medio de la presentación de estímulos auditivos o visuales [11] a los sujetos sanos o patológicos, mientras que simultáneamente se mide su capacidad para ver, recordar o interactuar con ese estímulo y/o su actividad cerebral durante la presentación. La tendencia lleva entonces a **generar herramientas** que permitan la presentación precisa de estímulos y la recopilación de respuestas de los participantes, estas herramientas deberían ser lo más fácil posible de usar reduciendo tiempos y mejorando la construcción de experimentos, ofreciendo una variedad de estímulos y diseños experimentales tan amplios como sea posible y logrando la recopilación de **bases de datos** para posteriores estudios. Adicionalmente, el paquete de software ideal debe ser libre e independiente de otras plataformas, de modo que su aplicación sea generalizada.

Respecto al procesamiento, los últimos estudios relacionados con VEP en estado estacionario (SSVEP), mencionan que además del método tradicional que implica la aplicación de la

transformada de Fourier a la señal EEG, implementar una variación que involucre un promediado coherente con los estímulos presentados, previo a la estimación del contenido de frecuencias de la señal EEG mediante análisis de Fourier provee altos desempeños con menos tiempo de estimulación lo que implica una mejora considerable del sistema en términos de aplicabilidad práctica y si bien los informes promedio son principalmente de valor exploratorio y de investigación, se busca lograr esquemas de detección de un solo ensayo [12].

En otros resultados de análisis se muestra que el uso simultáneo de la información de frecuencia y fase, así como la elección de frecuencias y electrodos mejoran la eficiencia de este tipo de BCIs [13],[14].

Finalmente, Norcia y Tyler [15] demostraron que una relación señal a ruido (SNR) de 3: 1 reduce la tasa de falsas alarmas de un solo contenedor a 0.003, por lo que se utilizó como criterio de amplitud para estimar el umbral y mientras que al obtener las siguientes medidas: la magnitud en la frecuencia de estimulación y una estimación de ruido, el promedio de la respuesta de las dos frecuencias vecinas permite encontrar una medida [16] de agudeza visual comparable con la tabla de Snell tradicional.

Cómo auge de los desarrollos tecnológicos se encuentran los análisis de tiempo-frecuencia profundizando en los procesamientos con transformada de Fourier, adicionalmente se busca la integración de análisis relacionados con aprendizaje de máquina (machine learning, ML), las redes de convolución que proveen soluciones procesando los datos de EEG en el dominio espectral [17] y temporal-espectral [18] empleando las transformadas de Fourier o Wavelets en una primera etapa, filtrando las señales en diferentes bandas de frecuencia de acuerdo a cada actividad neuronal bajo estudio conservando la estructura de los datos y finalmente en cuanto a la clasificación, el método de uso de la Máquina de vectores de relevancia (RVM) tiene un mejor rendimiento que las Maquinas de soporte vectorial (SVM) como método de precedencia [12], [19].

Con relación a estos antecedentes en estudios anteriores [20], los datos del EEG se registraron a partir de ocho electrodos (FCz, Oz, O1, O2, PO7, PO8, PO3 y PO4) utilizando el sistema de registro 10-10, los datos fueron registrados por el software de adquisición OpenViBE por medio del dispositivo OpenBCI y procesados en Python. Las señales adquiridas fueron filtradas entre 1 y 30 Hz [21] y se aplicó el método de Welch con ventana Hamming para analizar las señales en frecuencia para cada uno de los canales bipolares.

Finalmente lograr registrar de manera efectiva los potenciales visuales en estado estacionario con una herramienta portable, una interfaz amigable, con un protocolo de montaje sencillo, eficiente y asequible, favorece de gran manera la ejecución de pruebas en diferentes entornos, además, el desarrollo y aplicación de medidas de procesamiento cada vez más especializadas y eficientes pueden permitir el diagnóstico de patologías y la rehabilitación de pacientes con deficiencias visuales partiendo desde los parámetros más simples y que entregan más información para este tipo de sujetos como lo es la agudeza visual.

Objetivos

General:

Desarrollar una herramienta para la evaluación de la fisiología visual mediante el análisis de señales electroencefalográficas obtenidas con un dispositivo portable y de bajo costo.

Específicos:

- Implementar un paradigma de estimulación que permita evaluar una condición fisiológica visual usando librerías de código abierto.
- Diseñar algoritmos de preprocesamiento y análisis para la evaluación de los datos registrados usando técnicas de análisis espectral.
- Diseñar la interfaz de adquisición y visualización de los datos y parámetros visuales que permitan la evaluación de la fisiología visual usando criterios de usabilidad.

Marco Teórico

Sistema visual

La corteza visual del cerebro es la parte de la corteza cerebral responsable del procesamiento de la información visual. Se encuentra en el lóbulo occipital, en la parte posterior del cerebro [22].

El sistema visual se ha subdividido en muchas áreas funcionales. La corteza visual primaria (área 17 de Brodmann; V1) ha interesado a los anatomistas durante más de un siglo, y su posición y tamaño generales se han estimado muchas veces [23]. Además, el estudio de la corteza visual ha sido posible gracias a la fMRI de onda viajera, estas mediciones revelan claramente tres mapas de hemifield humano cerca del surco de calcarina en el lóbulo occipital (Figura 1). La corteza visual primaria (V1), que recibe

información directa de la vía retinogeniculada, ocupa la corteza calcarina y representa un hemifield del espacio visual. Dos mapas adicionales (V2, V3) ocupan una franja de corteza, de aproximadamente 1–3 cm de ancho, que rodea V1[24].

V1 es el área sensorial ubicada dentro y alrededor de la fisura de calcarina en el lóbulo occipital. V1 transmite información a dos vías principales: la corriente dorsal y la corriente ventral. La corriente dorsal se conoce como la ruta de acción, y va hacia arriba, mientras que la corriente ventral es la ruta de percepción, y va hacia abajo.

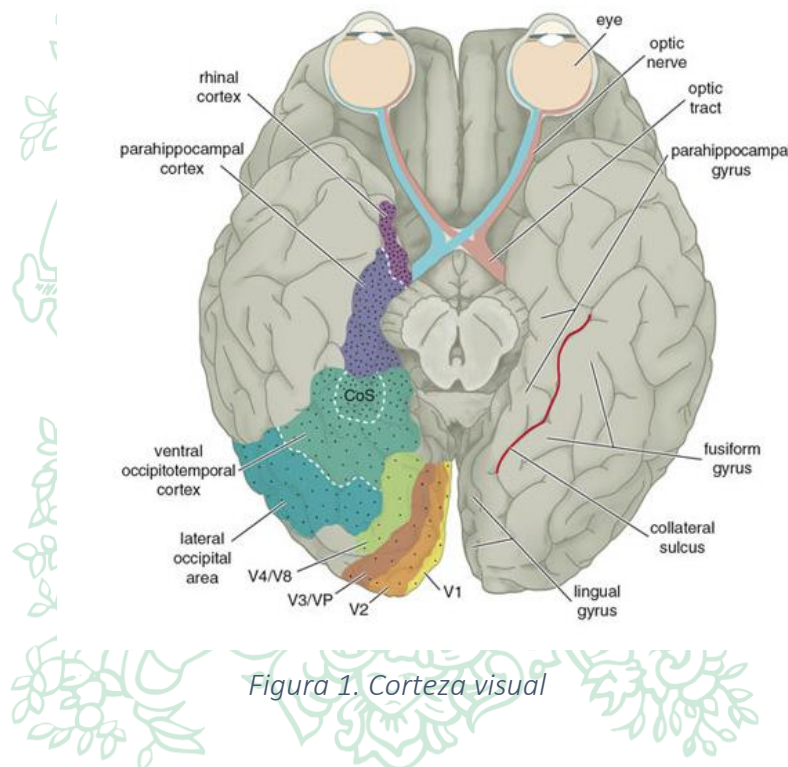


Figura 1. Corteza visual

Electroencefalografía (EEG)

La electroencefalografía es un instrumento que permite registrar la actividad eléctrica del cerebro, gracias al contacto de unos electrodos con la superficie del cuero cabelludo, midiendo las diferencias de potencial eléctrico y su cambio en el tiempo [25].

El principal propósito de esta es tener un método no invasivo y eficiente para medir la actividad eléctrica del cerebro, demostrando ser un método efectivo para diagnosticar muchas enfermedades neurológicas [26], como epilepsia, tumores, lesiones cerebrovasculares, isquemia y problemas asociados con el trauma.

El EEG mide la actividad eléctrica del cerebro causada por el flujo de corrientes eléctricas durante las excitaciones sinápticas de las dendritas en las neuronas. [27], [28]. Esta técnica es portátil, económica y con alta resolución temporal y ofrece la posibilidad de controlar los rápidos cambios dinámicos en la actividad cerebral [29][30].

La señal de EEG grabada es un método de monitoreo para registrar la actividad eléctrica a lo largo del tiempo. Este método contiene componentes de frecuencia que se pueden medir y analizar [27], [31]. El análisis de frecuencia de tiempo de la actividad eléctrica de EEG en el tiempo se puede realizar utilizando la descomposición de Fourier y los componentes de frecuencia se han descrito principalmente en las bandas delta (δ), theta (θ), alfa (α), beta (β) y gamma (γ) de menor a mayor, respectivamente, que se han definido debido a su presencia característica dependiendo de los estados cerebrales. La Figura 2 muestra las ondas o ritmos comúnmente definidos, su frecuencia y sus propiedades.


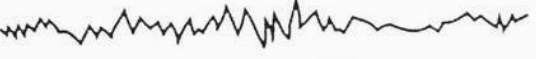


Ondas cerebrales	Frecuencia	Estado mental
	0,5 - 3 Hz	
Onda delta		sueño profundo
	4 - 7 Hz	
Onda theta		sueño ligero
	8 - 13 Hz	
Onda alfa		despierto, relajado
	14 Hz	
Onda beta		despierto, excitado

Figura 2. Ondas cerebrales.

Para encontrar la diferencia de potencial asociada a cada canal se hace necesario tomar de referencia un electrodo en el lóbulo de la oreja o en cualquier otra parte del cuerpo que no reciba una señal eléctrica cerebral, para este caso sería un registro 'monopolar'. Existe también el registro 'bipolar', siendo este de mayor uso [32], [33], en donde se registra la diferencia de voltaje entre dos electrodos colocados en el cuero cabelludo.

Cada electrodo representa un canal y siguiendo el sistema internacional 10-10 [34] permite localizar las zonas de interés de cada estudio. Aplicando este sistema se localizan los electrodos

a trabajar en este proyecto (FCz, Oz, O1, O2, PO3, PO4, PO7, PO8) y en los lóbulos de las orejas se encuentra referencia y tierra.

Potenciales evocados en estado estacionario (SSVEP)

Los potenciales evocados visuales (VEP) son pequeñas respuestas neuro eléctricas del orden de los 5-20 microvoltios, extraídas por una computadora mediante la colocación de electrodos occipitales sobre el área visual y que aparecen después de la aplicación de un estímulo luminoso [4].

Los potenciales evocados visuales en estado estable (SSVEP por sus siglas en inglés) son un fenómeno de resonancia que se puede observar cuando un sujeto mira una fuente de luz que parpadea a una frecuencia constante [35]–[38], modulado a una frecuencia fija superior a 4 Hz, con respuestas de espectro estable y una alta relación señal-ruido, que puede ser evocadas por un estímulo visual con una frecuencia parpadeante que oscila hasta 75 Hz [38]–[41]. Su comportamiento se puede evaluar por medio del análisis de Fourier en donde la amplitud de la potencia aparece en la frecuencia de estimulación y sus armónicos siguientes.

Agudeza visual (AV)

La agudeza visual (AV) se puede definir como la capacidad de percibir y diferenciar dos estímulos separados por un ángulo determinado (α), dicho de otra manera, es la capacidad de resolución espacial del sistema visual [42]. Matemáticamente la AV se define como la inversa del ángulo con el que se resuelve el objeto más pequeño identificado: $AV = 1/\alpha$

Sin embargo, la AV no es sólo el resultado de un ajuste óptico adecuado de las diferentes estructuras oculares (córnea, cristalino, retina, etc.), sino que depende del estado de la vía óptica y del estado de la corteza visual [42]. Por tanto, la visión es un proceso más amplio que la AV por el cual se percibe e integra la información que llega a través de las vías visuales, analizándola y comparándola con otras imágenes o experiencias previas.

Agudeza de Vernier (EEG)

La agudeza vernier es una medida del desplazamiento posicional más pequeño de las características visuales con una precisión mejor que la resolución de muestreo de los receptores de cono en la retina [23], [25]. Existen dos paradigmas ampliamente estudiados para la medición de agudeza visual, estos son Vernier (Vernier Acuity Sweep) y la agudeza a rejilla (Grating Acuity Sweep) [43], están basados en la modulación de patrones simétricos y asimétricos produciendo armónicos en las respuestas VEP de estado estable.

Métodos de procesamiento de señales EEG

1. Análisis de frecuencia: Referente al rango de frecuencia de la actividad eléctrica cerebral. Las características básicas de señales de EEG, usan el proceso de clasificación de SSVEP por medio de las oscilaciones asociadas a la estimulación luminosa y sus armónicos [44]. Este análisis es el más comúnmente usado, en el caso de la transformada discreta de Fourier (DFT), para N series temporales de tamaño X_n , los coeficientes X_k , son determinados por la siguiente relación [44].

$$X_k = \frac{1}{N} \sum_{n=0}^{N-1} X_n e^{-j\left(\frac{2\pi}{N}k\right)n}$$

Ecuación 1

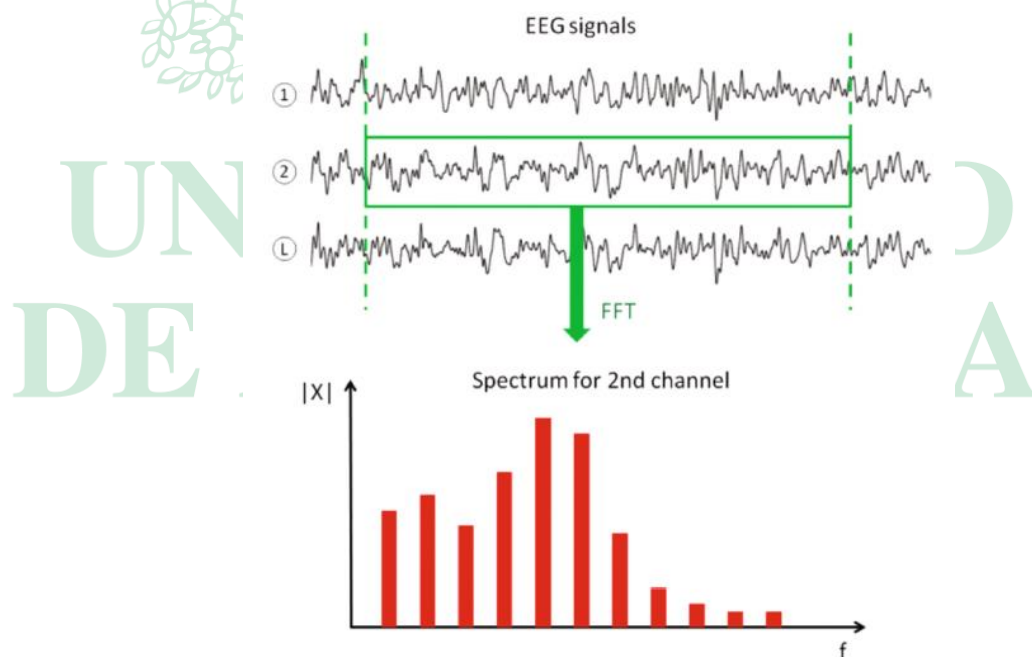


Figura 3. Transformada discreta de Fourier

2. Análisis tiempo-frecuencia: Permite observar la variabilidad en frecuencia de las señales EEG. Este tipo de análisis es particularmente usado en estudios referentes al funcionamiento cerebral [44]. Para SSVEP, las características de las señales EEG deben ser estables en el tiempo. La transformada de Fourier en tiempo corto (STFT), permite confirmar esta estabilidad. La señal EEG se divide en ventanas temporales estrechas y como resultado se obtiene un espectrograma para los valores t/f de cada coeficiente $X_{n,k}$. Para ventanas deslizantes se utiliza la siguiente relación:

$$X_{n,k} = \frac{1}{M} \sum_{m=0}^{M-1} X_m \varphi_{n-m} e^{-j\left(\frac{2\pi k}{M}\right)m}$$

Ecuación 2

Mediante el análisis STFT (espectrograma), se puede determinar cuál de los fragmentos de señal EEG muestra las frecuencias de interés [44].

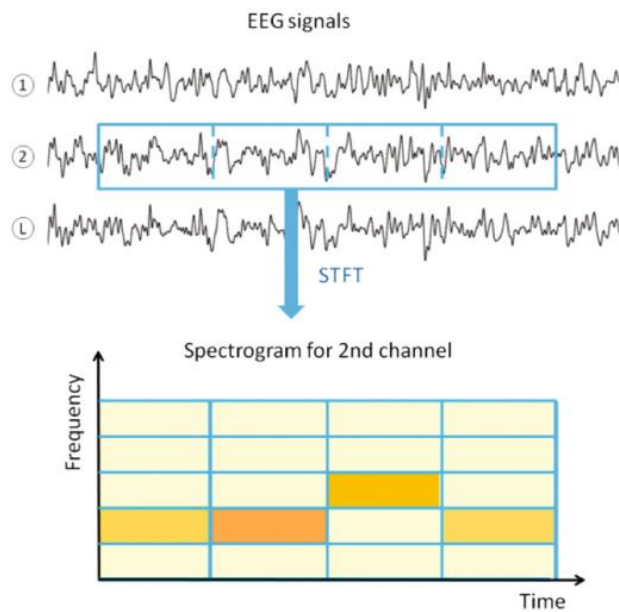


Figura 4. Transformada de Fourier de tiempo corto.

3. Redes neuronales: La neurona artificial es la unidad básica de una red neuronal. Su procesamiento radica en recibir datos de entrada de otras neuronas y producir una salida a través de conexiones con determinado peso que se modifica mediante el aprendizaje, lo que determina el comportamiento de la neurona [45]. Esto es, realiza la suma ponderada de los datos de entrada con los pesos correspondientes, este término es el encargado de generar la señal de activación que se filtra mediante una función de transferencia para obtener la salida de la neurona. La simulación de una neurona artificial considera un dato de entrada, representado por la variable,

que se multiplica por el peso y su correspondiente valor de umbral, llamado bias o sesgo; el resultado obtenido es analizado por una función de activación para inmediatamente obtener la salida esperada [46].

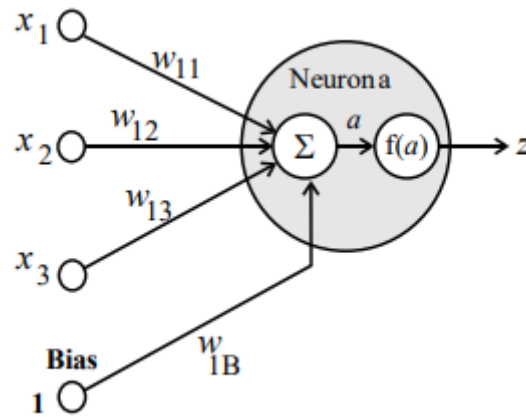


Figura 5. Neurona de red neuronal.

Matemáticamente este cálculo es expresado mediante la siguiente ecuación:

$$f(a) = \begin{cases} 1 & \text{Si } \sum_{j=1}^n w_{nj}x_i + b_n > 0 \\ 0 & \text{en caso contrario} \end{cases}$$

Ecuación 3

Método de Welch

Tiene como principio el análisis de frecuencia con la transformada discreta de Fourier [47]. El método se basa en el concepto de usar estimaciones del espectro del periodograma, que son el resultado de convertir una señal del dominio del tiempo al dominio de la frecuencia [48]. El método de Welch es una mejora en el método estándar de estimación del espectro del periodograma y en el método de Bartlett, ya que reduce el ruido en los espectros de potencia estimados a cambio de reducir la resolución de frecuencia.

Welch propuso un método según el cual se dividía el registro de N puntos original en segmentos de M puntos solapados entre sí L muestras. Si $L=M$, entonces $N=(K+1) M$, donde K sería el número total de segmentos. Posteriormente se aplica una ventana a cada segmento, y finalmente se calcula el periodograma para cada segmento enventanado [48]. El periodograma final se obtiene promediando todos los periodograma parciales.

Se basa en el método de Bartlett y difiere de dos maneras:

1. La señal se divide en segmentos superpuestos: el segmento de datos original se divide en segmentos de datos L de longitud M , la superposición por puntos D [49].
 - a. Si $D = M/2$, se dice que el solape será 50%
 - b. Si $D = 0$, se dice que el solapamiento a 0%. Esta es la misma situación que en el método de Bartlett.
2. Los segmentos superpuestos son entonces de ventana: Después de que los datos se dividen en segmentos superpuestos, los segmentos de datos L individuales tienen una ventana aplicada a ellos (en el dominio del tiempo) [49].
 - a. La mayoría de las funciones de la ventana ofrece una mayor influencia de los datos en los bordes, lo que representa una pérdida de información. Para mitigar esa pérdida, los conjuntos de datos individuales se solapan comúnmente en el tiempo (como en el paso anterior).
 - b. La ventana de los segmentos es lo que hace el método de Welch un periodograma “modificado”.

Después de hacer lo anterior, el periodograma se calcula mediante el cálculo de la transformada de Fourier discreta, y luego calcular la magnitud al cuadrado del resultado. El individuo periodograma se promedia, lo que reduce la varianza de las mediciones de potencia individuales. El resultado final es una serie de mediciones de potencia frente a la frecuencia “bin” [50].

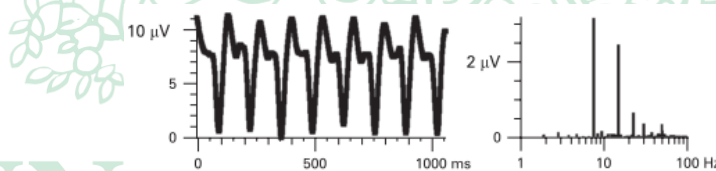


Figura 6. Análisis de frecuencia. Método del Welch.

Diseño y usabilidad de una interfaz de usuario

La Interfaz de Usuario (IU), de un programa, es un conjunto de elementos hardware y software de una computadora que presentan información al usuario y le permiten interactuar con la información y el computador. Una parte importante de una IU es la documentación (manuales, ayuda, referencia, tutoriales) que acompaña al hardware y al software para facilitar y permitir su adecuado uso [51].

Los programas son usados por usuarios con distintos niveles de conocimientos, desde principiantes hasta expertos. Es por esto por lo que una UI debe permitir la libertad del usuario para que elija el modo de interacción que más se adecúe a sus objetivos en cada

momento. La mayoría de los programas y sistemas operativos ofrecen varias formas de interacción al usuario [52].

a. Heurísticas para la Evaluación de IU

Las heurísticas ayudan a analizar las IU y localizar problemas que afecten la utilización de las mismas[53], [54].

Algunas pautas para evaluar una IU son:

- Visibilidad del estado del sistema
- Semejanza del sistema al mundo real
- Control y libertad por parte del usuario
- Consistencia y estandarización
- Prevención de Errores
- Reconocimiento de acciones y opciones
- Flexibilidad y eficiencia en el uso
- Estética y diseño minimalista
- Reconocimiento de errores, diagnóstico y recuperación
- Ayuda y documentación

Para establecer medidas que indiquen la severidad de los problemas en el uso de las interfaces, se deben conocer los factores que determinan el grado de un problema:

- La frecuencia de ocurrencia.
- El impacto que causa la ocurrencia del problema.
- La persistencia del problema.
- El impacto en el mercado.

b. Medidas de severidad de un problema en la IU:

0: No puede llegar a considerarse un problema.

1: Es un problema “cosmético” que no necesita ser corregido a menos que se disponga tiempo extra en el proyecto.

2: Es un problema menor y su corrección puede tener baja prioridad.

3: Es un problema mayor y su corrección debería tener alta prioridad.

4: Es una catástrofe para la utilización de la aplicación y es imperativo corregir el error.

Para la evaluación de los problemas en las IU es conveniente que contar con más de un evaluador; de esta forma los resultados son más confiables.

c. Modelo Vista Controlador (MVC)

El MVC es un patrón de arquitectura de software donde se parte las funciones del sistema en tres componentes (Vistas, Modelos y Controladores) separa la lógica de funcionamiento de la lógica de la vista, permitiendo estructurar sistemas robustos de forma clara y eficiente, sobre todo pensando en que los sistemas sean escalables y requerirán mantenimiento [55].

- *Modelo*: Contiene los datos y la funcionalidad de la aplicación, es decir, se encarga de realizar las funciones de actualizar, búsqueda, consulta, procesamiento de datos, etc.

- *Controlador*: Determina que procesos debe realizar el modelo cuando el usuario interactúa con el sistema, para luego comunicarle a la vista los resultados.
- *Vista*: Gestiona como se muestran los datos en la interfaz gráfica.

Base de datos de los sujetos

Las bases de datos NoSQL o "no solo SQL", no son tabulares y almacenan los datos de manera diferente a las tablas relacionales; vienen en una variedad de tipos basados en su modelo de datos. Los tipos principales son: documento, clave-valor, columna ancha y gráfico. Esto permite que sean flexibles y escalables con grandes cantidades de datos y altas cargas de usuarios. Los modelos de datos NoSQL permiten que los datos relacionados se aniden dentro de una sola estructura de datos[56].

Surgieron a fines de la década del 2000 cuando el costo de almacenamiento disminuyó drásticamente. Para los desarrolladores el lugar del almacenamiento se convirtió en el costo principal del desarrollo de software, por lo que las bases de datos NoSQL se volvieron óptimas para la productividad del desarrollador. Además, los datos se presentaban en todas las formas y tamaños: estructurados, semiestructurados y polimórficos, y definir el esquema preestablecidos era casi imposible. Las bases de datos NoSQL permitieron a los desarrolladores almacenar grandes cantidades de datos no estructurados, dándoles mucha flexibilidad[57].

▪ ¿Cuáles son los tipos de bases de datos NoSQL?

Con el tiempo, surgieron cuatro tipos principales de bases de datos NoSQL: bases de datos de documentos, bases de datos de valores clave, almacenes de columnas anchas y bases de datos de gráficos[56].

Las bases de datos de documentos almacenan datos en documentos similares a los objetos JSON (JavaScript Object Notation). Cada documento contiene pares de campos y valores. Los valores generalmente pueden ser una variedad de tipos que incluyen cosas como cadenas, números, booleanos, matrices u objetos, y sus estructuras generalmente se alinean con los objetos con los que los desarrolladores están trabajando en el código.

Las bases de datos de valores clave son un tipo de base de datos más simple donde cada elemento contiene claves y valores. Por lo general, un valor solo se puede recuperar haciendo referencia a su valor, por lo que aprender a consultar un par clave-valor específico suele ser simple. Las bases de datos de valores clave son excelentes para casos de uso en los que necesita almacenar grandes cantidades de datos pero no necesita realizar consultas complejas para recuperarlos.

Las tiendas de columna ancha almacenan datos en tablas, filas y columnas dinámicas. Los almacenes de columnas anchas proporcionan mucha flexibilidad sobre las bases de datos relacionales porque no se requiere que cada fila tenga las mismas columnas.

Las bases de datos de gráficos almacenan datos en nodos y bordes. Los nodos suelen almacenar información sobre personas, lugares y cosas, mientras que los bordes almacenan información sobre las relaciones entre los nodos.

▪ MongoDB

Es una base de datos distribuida, de propósito general, basada en documentos, creada para desarrolladores de aplicaciones modernas y para la era de la nube.

MongoDB está escrito en C++, aunque las consultas se hacen pasando objetos JSON como parámetro. Es algo bastante lógico, dado que los propios documentos se almacenan en BSON[58].

La manipulación de documentos es un objetivo principal de MongoDB, ya que proporciona varios marcos como MapReduce y formas de interacción de documentos. Los documentos pueden ser iterados, consultados y ordenados con cursores, agregados por otras operaciones[59].

Metodología

Descripción

Con el apoyo de la literatura se realizará la revisión de paradigmas de estimulación que usualmente son utilizados en pruebas de evaluación fisiológica, permitiendo seleccionar la estimulación que ha obtenido los mejores resultados, seleccionando así el paradigma que por sus características entregue información valiosa y confiable en evaluaciones visuales. A continuación, se realizará la programación del estímulo usando Python y se evaluarán los tiempos de estimulación para asociarlos a la adquisición del dispositivo portable permitiendo así la generación de marcas durante los registros.

Para la parte del procesamiento se diseñan filtros que permitan mejorar la calidad de las señales adquiridas con el dispositivo portable OpenBCI [60] y estrategias de rechazo de segmentos atípicos dejando fuera elementos que no son importantes como el ruido.

Finalmente se implementan algoritmos para la estimación de las características oscilatorias o espectrales de la señal, evaluando las características de tiempo/frecuencia.

Antes de concluir la herramienta se debe realizar la revisión de criterios de usabilidad que permitan al usuario entender, que puede hacer, como lo puede hacer y que le ofrece la interfaz, sin esfuerzo, de forma totalmente directa.

Con respecto a la interfaz se realiza el diseño de la vista de registro de usuarios y presentación de datos, que permita unir de manera ordenada la adquisición, procesamiento y resultados de los registros realizados. Se hace necesario también diseñar un formato de presentación de resultados a los usuarios priorizando los datos relevantes en el ámbito médico en este tipo de pruebas. Los diseños para la interfaz se implementan usando PyQt.

Como último parámetro de evaluación se selecciona un grupo de sujetos sanos que participen en el registro de electroencefalografía y permita visualizar los resultados entregados por la herramienta.

Para la elaboración del proyecto se pretende seguir las actividades que se muestran en el diagrama en la (Figura 7).

1 8 0 3

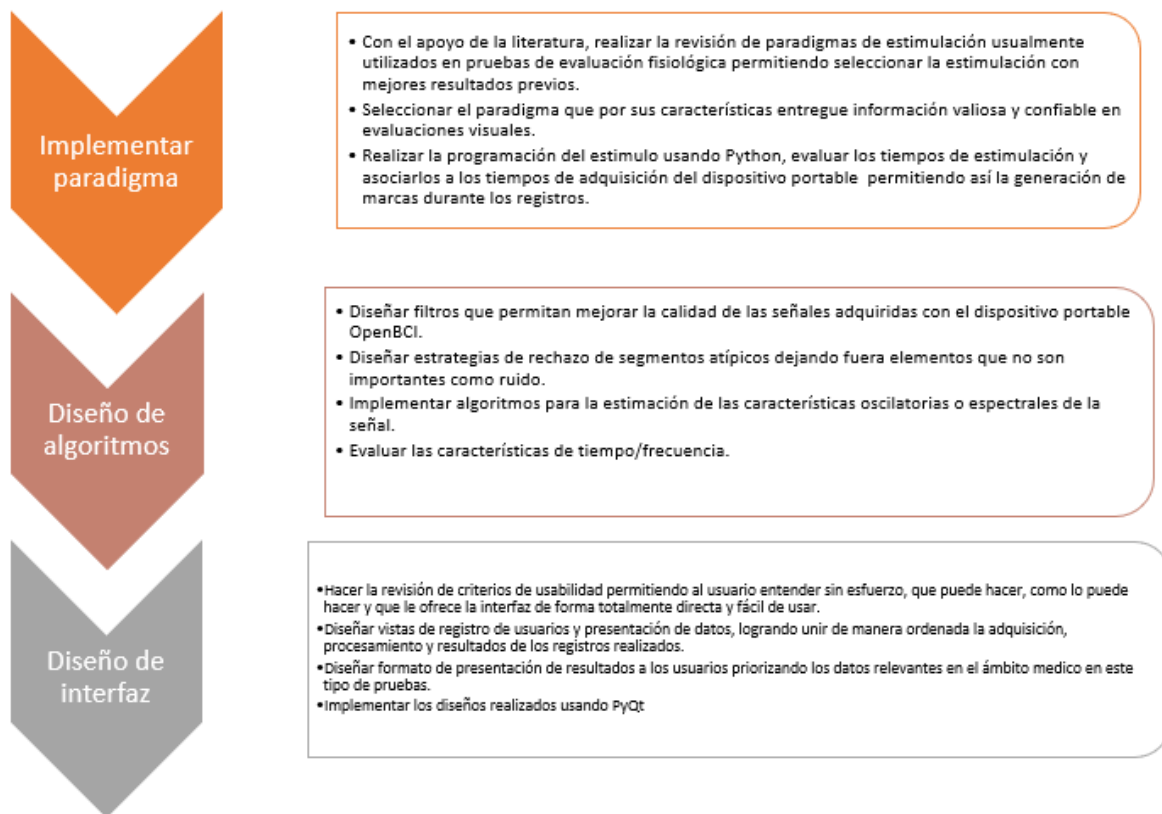


Figura 7. Metodología del proyecto.

*La herramienta se realizará con una base de datos existente de 47 sujetos (Documentación de proceso de adquisición de base de datos existente Anexo 1)

Resultados y análisis

Implementación del paradigma

Estimulación – Agudeza

El estímulo utilizado fue Vernier, que permite evaluar la característica de “Mínimo separable” es decir, la habilidad para ver separados dos objetos muy próximos. Si se presentan dos puntos luminosos suficientemente separados y se van acercando entre sí, llegará un momento en el que será imposible discernir si se trata de un punto o de dos [42]. Si la experiencia se realiza con barras verticales de igual anchura alternativamente blancas y negras (Miras de Foucault) y se trata de detectar cuándo se ven alineadas (similar a la lectura en un nonius) se comprueba que con altas luminosidades y en las mejores condiciones se perciben desalineadas si su anchura subtiende como mínimo un ángulo de 38 segundos de arco. Es, por tanto, una de las máximas capacidades de discriminación del ojo [42]. Esta prueba recibe el nombre de agudeza Vernier (Figura 8).

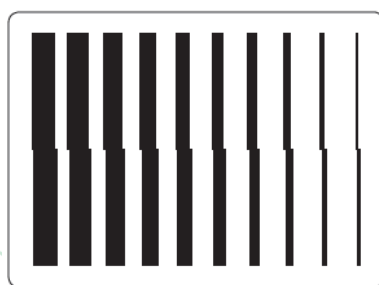


Figura 8. Agudeza de vernier.

Esta elevada capacidad del ojo de discriminación se utiliza en la exploración clínica de alteraciones maculares en diferentes pruebas como la prueba de la rejilla de Amsler y algunos micro perímetros computarizados [42].

Ahora para el presente proyecto se alternó este cambio entre dos estados. La pantalla está cubierta con una rejilla de barra y en un segundo estado, las porciones de la pantalla se desfasan. Todos los estímulos se alternaron entre dos estados a una velocidad de 0.125 segundos = 8 Hz.

Para el paradigma barrido (Figura 9), el tamaño de los desplazamientos osciló entre 2 y 30 ciclos por grado (cpd) en pasos logarítmicos durante un período de 28 segundos, cada desplazamiento con alternando durante 4 segundos, logrando medir 7 niveles de agudeza. (2, 3, 4.7, 7.5, 12, 18.75, 30).

El estímulo fue diseñado en Python y presentado en la herramienta diseñada ViAT para facilitar la adquisición y las marcas, y se realizó en una pantalla Samsung SyncMaster 2243 LNX.

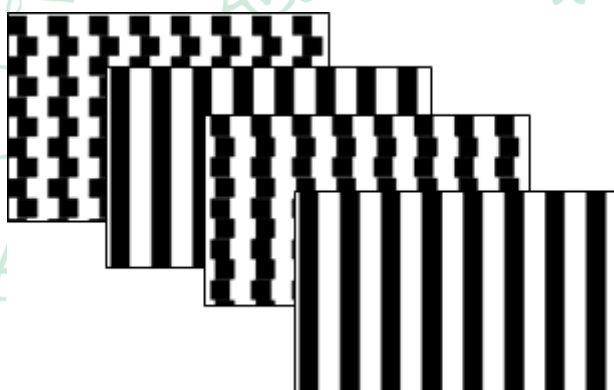


Figura 9. Barrido de agudeza de Vernier.

Se comienza estimulando el ojo derecho durante los 28 segundos, se hace una pausa de 4 segundos y se sigue con el ojo izquierdo el mismo proceso, finalmente se realiza la estimulación de manera binocular, la estimulación total tiene una duración de 1:36 min.

Para el diseño del estímulo se utilizó la librería PyGame (**Documentación Anexo 2**).

Entorno de estimulación

Los sujetos se deben registrar en una habitación oscura y tranquila, se deben sentar con una posición recta y cómoda, sobre una silla estática que permite graduar la altura del sujeto para quedar frente a la pantalla, adicional, se debe fijar la mirada del sujeto con un soporte superior con apoyo mentoniano (Figura 10). La distancia de estimulación para el estímulo diseñado es de 1 m para evaluación de visión intermedia [42].

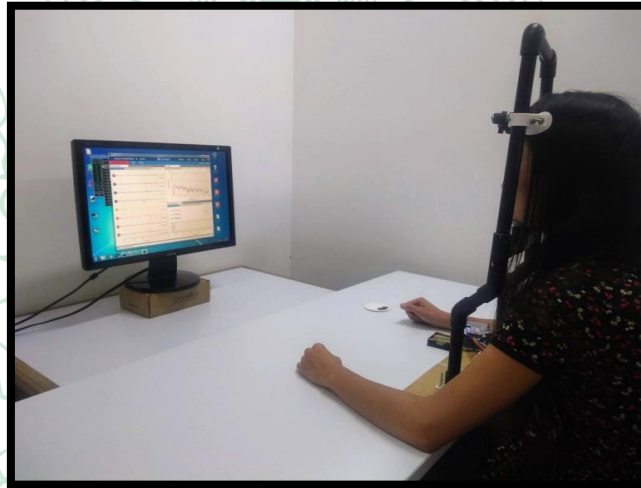


Figura 10. Entorno de estimulación.

Se debe limitar el ruido electromagnético que pueda provenir de los dispositivos electrónicos como celulares, alejándose del entorno de adquisición al menos 2 metros.

Adquisición OpenBCI

Se utiliza un sistema de adquisición portable OpenBCI de 8 canales compatible con Arduino con un procesador de 32 bits para adquirir señales de EEG desde la superficie del cuero cabelludo. El dispositivo funciona con 4 pilas recargables AA 6V y se comunica con la herramienta de adquisición OpenViBE por medio de comunicación bluetooth dongle.

La frecuencia de muestreo fue de 250 Hz. Según el sistema internacional 10-10, los sensores se colocaron en el cuero cabelludo en las ubicaciones FCz, Oz, O1, O2, PO7, PO8, PO3, PO4. Además, se utilizaron dos electrodos en los lóbulos de las orejas izquierda y derecha para las señales de referencia y de tierra, respectivamente.

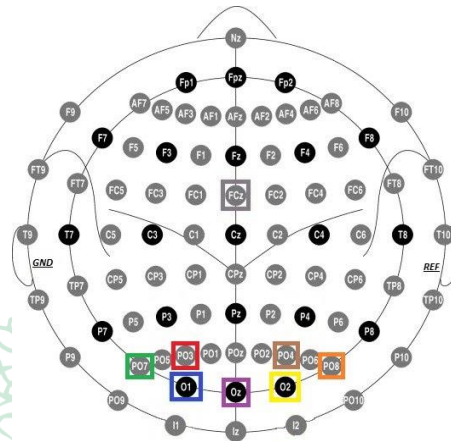


Figura 11. Montaje de electrodos utilizado.

Antes de mostrar el estímulo se realiza la medición de la impedancia por medio del software ViAT para cada uno de los electrodos procurando que quede por debajo de 30 Kohm. Se utiliza un gel conductor Nuprep [61] sobre el cuero cabelludo y una pasta conductora Ten20 [62] en cada electrodo.

Diseño de algoritmos

Filtros

1. Diseño de filtros:

- filter_design:** Diseña el filtro FIR de ventana sinc tipo I de fase lineal. Este filtro fue diseñado por el profesor John Fredy Ochoa e implementado para en la recepción inicial de los datos. El código que encuentra en el archivo **linearFIR.py** consiste en un filtro prototípico pasa-bajo, con el cual se diseña un nuevo filtro pasa-bajo y se transforma. Con una banda de transición; -6 dB.

```
model.py -> class Model

def filtDesign(self):
    order, self.lowpass = filter_design(
        self.__fs, locutoff=0, hicutoff=50, revfilt=0)
    order, self.highpass = filter_design(
        self.__fs, locutoff=5, hicutoff=0, revfilt=1)
```

- hampelFilter:** Procedimiento para implementar el filtro Hampel. Este filtro fue diseñado por el profesor John Fredy Ochoa e implementado como segunda fase de filtración. El código que encuentra en el archivo **nonlinear.py** consiste en sacarle la media a la señal y el valor absoluto de la diferencia entre los elementos de la señal y la media. Este filtro se aplica directamente a la señal completa, es decir, cada canal, como a la operación que se quiera evaluar con Laplace ($a*2-b-c$) siendo a,b,c cualquier canal que se esté registrando.
- signal.filtfilt:** Este filtro es tomado de la biblioteca **scipy**. Esta función aplica un filtro digital lineal dos veces, una vez hacia adelante y una vez al revés. El filtro combinado tiene fase cero y un orden de filtro el doble que el original. La

función proporciona opciones para manejar los bordes de la señal. Se aplica de la misma manera que el filtro anterior tanto a la señal completa como a la operación entre canales.

model.py -> `class Model`

```
def filtData(self):
    self.readData()

    self.senal_filtrada_pasaaltas = signal.filtfilt(
        self.highpass, 1, self.__data)
    self.senal_filtrada_pasaaltas = hampelFilter(
        self.senal_filtrada_pasaaltas, 6)
    self.senal_filtrada_pasabandas = signal.filtfilt(
        self.lowpass, 1, self.senal_filtrada_pasaaltas)

    self.laplace_filtrada_pasaaltas = signal.filtfilt(
        self.highpass, 1, self.__laplace)
    self.laplace_filtrada_pasaaltas = hampelFilter(
        self.laplace_filtrada_pasaaltas, 6)
    self.laplace_filtrada_pasabandas = signal.filtfilt(
        self.lowpass, 1, self.laplace_filtrada_pasaaltas)
```

- d. **signal.welch**: Este filtro es tomado de la biblioteca **scipy**. Estima la densidad espectral de potencia utilizando el método de Welch. El método de Welch calcula una estimación de la densidad de potencia espectral dividiendo los datos en segmentos superpuestos, calculando un periodograma modificado para cada segmento y promediando el periodogramas. Se aplica entro de la aplicación para graficar la señal.

model.py -> `class Model`

```
def Pot(self):
    self.filtData()
    nblock = 250
    noverlap = nblock/2;

    self.f, self.Pxx = signal.welch(self.senal_filtrada_pasabandas, self.__fs,
                                   nperseg=self.__fs*2, noverlap=noverlap);

    self.ftg, self.Pxxtg = signal.welch(self.__laplace,
                                         self.__fs, nperseg=self.__fs*2, noverlap=noverlap);
```

2. Estrategias de procesamiento:

- a. **Marcas**: Es un elemento que permite separar la señal por tramos de interés según el estímulo presentado, se realiza directamente desde el estímulo en

este caso en *Stimulation_Acuity.py* y se guarda en un archivo .csv para posteriormente ser comparado con la señal adquirida y realizar la separación de la señal.

```
dataprocessing.py -> class Processing -> def run(self):
...
Record=pd.read_csv(path_Record, sep=';', index_col=0)
Mark=pd.read_csv(path_Mark, sep=';', index_col=0)
index=list(range(0,len(Record['C1'])))
Record['i'] = index
listMark = Mark.values.tolist()
Markdata = []
for i in range(0,len(listMark)-1):
    start=list(Record.i[Record.H == Mark.iloc[i,0]])
    end=list(Record.i[Record.H == Mark.iloc[i+1,0]])
    MO=Record[start[0]:end[0]].drop(columns=['H','i'])
    Markdata.append(MO)
...
```

- b. **pmtm**: Esta función es tomada de la biblioteca **spectrum**. Por lo general, en la estimación espectral, la media para reducir el sesgo es usar la reducción gradual de ventana. Para reducir la varianza, necesitamos promediar diferentes espectros. El problema es que solo tenemos un conjunto de datos. Por lo tanto, necesitamos descomponer un conjunto en varios segmentos. Tal método es bien conocido: simple periodograma de daniell, método de Welch, etc. El inconveniente de tal método es una pérdida de resolución ya que los segmentos utilizados para calcular el espectro son más pequeños que el conjunto de datos. El interés del método multitaper es mantener una buena resolución mientras reducción de sesgo y varianza. Consiste en calcular un periodograma simple diferente con el conjunto de datos completo (para mantener una buena resolución) pero se calcula cada periodograma con una ventana diferente. Luego, promediamos todo este espectro.

```
dataprocessing.py -> class Processing -> def run(self):
...
Sk_complex , weights , _ = pmtm(data[i], NW=2, k=4, show=False)
Sk = abs(Sk_complex)**2;
Sk = Sk.transpose();
Sk = np.mean(Sk * weights, axis=1);
...
```

- c. **Búsqueda de valores de interés**: Se limita la búsqueda de potencias en un rango de la frecuencia de interés y se guardan estos valores para posteriores análisis estadísticos o de machine learning, que permitan resultados confiables. En este caso la frecuencia de estimulación es de 7.5 Hz por esto el

rango de interés esta entre 7 y 8.5 Hz, puede variar o reducirse según el estímulo presentado.

```
dataprocessing.py -> class Processing -> def run(self):
...
frequencies = np.linspace(0,250,1024)
position = np.where((frequencies >= 7) & (frequencies <= 8.5))
maxValue = np.max(Sk[position[0]])
values.append(maxValue)
posicion= np.where(Sk==maxValue)
maxValuex = frequencies[posicion[0]]
frec.append(maxValuex[0])
doc = pd.DataFrame(values,columns=['Max'])
doc['Fre']=frec
...
```

3. Evaluación de características de tiempo frecuencia:

- Transformada de Fourier de corto tiempo (STFT):** Las STFT se pueden utilizar como una forma de cuantificar el cambio de la frecuencia de una señal no estacionaria y el contenido de fase a lo largo del tiempo.
- Generar imagen descriptiva de frecuencia de interés:** Se utiliza la función `plot_stft` para leer los datos, realizar la STFT, limitar los rangos y generar la matriz grafica de frecuencias.

```
plot_stft.py -> class TimeFrequency -> def plot_stft (self):
...
signal = pd.read_csv(self.loc+'/'+ 'Record_' + name + '.csv',sep=';',
index_col=0)
fs= 250
f, t, Zxx = sig.stft( signal['C2'], fs, nperseg = 500)
levels = MaxNLocator(nbins=15).tick_values(np.abs(abs(Zxx)).min(),
np.abs(abs(Zxx)).max())

cmap = plt.get_cmap('jet')
norm = BoundaryNorm(levels, ncolors=cmap.N, clip=True)
fig,ax0 = plt.subplots(nrows=1)
im = ax0.pcolormesh(t, f, np.abs(abs(Zxx)), cmap=cmap, norm=norm)
...
```

Diseño de interfaz

Se diseña bajo el sistema MVC y cuenta con 6 vistas:

- ViAT -> Inicio
- Agregardatos -> Interacción con la base de datos del sistema (Figura 12).
- Adquisicion -> Conexión con el dispositivo de adquisición (Figura 13).

- Adquisición_accion -> Visualización en tiempo real de la adquisición (Figura 14),
- Buscardatos -> Permite visualizar datos de cada sujeto en la base de datos
- Visualizacion -> Permite visualizar señales adquiridas previamente.

1. Revisión de criterios de usabilidad:

Base de datos

HISTORIA CLINICA

ViAT

ID:

Nombre:

Apellido:

CC:

Sexo:

Ojo dominante:

¿Usa gafas?

Agudeza de Snellen:

Agudeza de Snellen corregida:

Estimulo:

Edad:

¿Tiempo desde el accidente visual?

Responsable:

Verificar si un paciente se encuentra ya registrado reduciendo tiempos

Valores por defecto permitiendo también control y libertad por parte del usuario.

Figura 12. Historia clínica

Adquisición

ADQUISICIÓN

ViAT

Medida de la impedancia en [Kohms] < 30 Kohms

FCz

Oz

O1

O2

PO7

PO8

PO3

PO4

Permite seguir un flujo de aplicación por medio de habilitar y deshabilitar funciones según el estado del sistema.

Consistencia y estandarización

Botones de ayuda

Figura 13. Conexión de dispositivo.

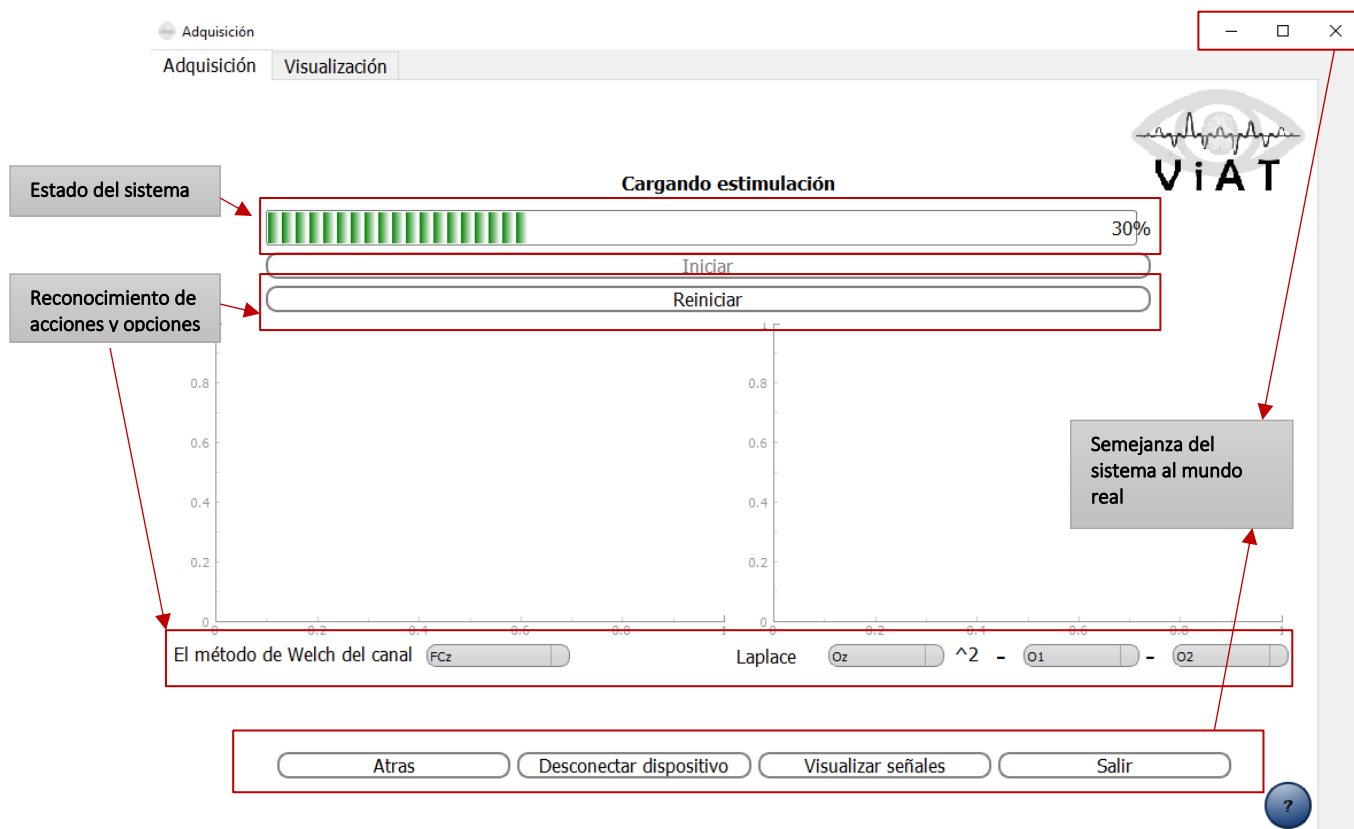


Figura 14. Adquisición de datos.

*Se adjunta manual de usuario para facilitar el uso de la interfaz realizada (Documentación Anexo 3).

2. Diseñar formato de presentación de resultados.

Los resultados se entrenan principalmente en 2 carpetas:

Tabla 1. Estructura de los archivos.

Estructura	Visual
<ul style="list-style-type: none"> Processing <ul style="list-style-type: none"> ID_CC <ul style="list-style-type: none"> FECHA <ul style="list-style-type: none"> Parameters_ID_CC.csv Time-Frequency Oz-FCz_ID_CC.jpg 	<ul style="list-style-type: none"> Processing <ul style="list-style-type: none"> H1_1152207135 <ul style="list-style-type: none"> 06-28-2020 <ul style="list-style-type: none"> Parameters_H1_1152207135 Time-Frequency Oz-FCz_H1_1152207135
<ul style="list-style-type: none"> Records <ul style="list-style-type: none"> ID_CC <ul style="list-style-type: none"> FECHA <ul style="list-style-type: none"> Mark_ID_CC.csv Record_ID_CC.csv 	<ul style="list-style-type: none"> Records <ul style="list-style-type: none"> H1_1152207135 <ul style="list-style-type: none"> 06-28-2020 <ul style="list-style-type: none"> Mark_H1_1152207135 Record_H1_1152207135

Se elijen los archivos .csv por su facilidad de manipulación.

La estructura de los archivos es la siguiente:

Processing

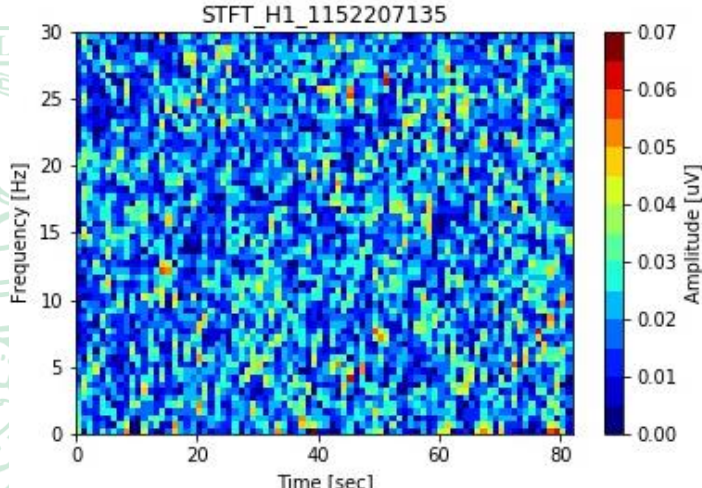
Tabla 2. Estructura procesamiento.

Parameters_ID_CC.csv	Max	Fre
	0.59804803	7.08699902
	0.10245691	8.30889541
	0.40024617	8.30889541
	0.25079922	7.08699902
	0.07403019	8.30889541
	0.31702204	8.30889541

Max: Hace referencia a la amplitud máxima encontrada en el rango de frecuencia de interés, en este caso entre 7 y 8.5 Hz

Fre: Hace referencia a la frecuencia exacta a la que se encontró el valor máximo.

Estos datos son útiles y necesarios para realizar evaluaciones posteriores de la señal con apoyo de Machine Learning [46] u otras técnicas de análisis estadístico.

Time-Frequency Oz-FCz_ID_CC.jpg	<p>STFT_H1_1152207135</p> 

La imagen permite observar a lo largo de la señal diferencial Oz-FCz la frecuencia en el tiempo y con esto validar la eficiencia de la estimulación realizada a una frecuencia determinada.

Records

Tabla 3. Estructura registros.

Record_ID_CC.csv	C1	C2	C3	C4	C5	C6	C7	C8	H
	0.73187554	0.1191259	-0.4066793	-0.2687161	-0.393944	-0.29221848	-0.30618772	-0.00962621	11-22-25
	0.52693057	0.33831501	0.45620424	0.24923992	0.23631233	0.08804327	0.11061227	-0.41921276	11-22-26
	0.92276698	-0.13704956	-0.58744523	-0.76322785	-0.19375223	-0.41278476	-0.42801601	-0.88206647	0
	0.30380246	-0.25563728	0.44580564	-0.19913614	0.23976383	-0.03268635	-0.22412168	-0.19581272	0
	0.30664465	0.64603671	0.38948187	-0.15265588	-0.137475	0.64516971	-0.04174143	0.37839356	0
	0.60575074	-0.12673205	0.18656027	-0.4891784	0.23382193	-0.36466084	0.34857655	0.21318001	0
<p>Contiene todos los registros de los sujetos y sus respectivas marcas al realizar la estimulación.</p> <p>C1: Representa el canal 1 es decir FCz, canal de referencia.</p> <p>C2: Representa la diferencia entre el canal 1 y el canal 2 es decir Oz-FCz</p> <p>C3: Representa la diferencia entre el canal 1 y el canal 3 es decir O1-FCz</p> <p>C4: Representa la diferencia entre el canal 1 y el canal 4 es decir PO7-FCz</p>									

C5: Representa la diferencia entre el canal 1 y el canal 5 es decir O2-FCz
 C6: Representa la diferencia entre el canal 1 y el canal 6 es decir PO8-FCz
 C7: Representa la diferencia entre el canal 1 y el canal 7 es decir PO3-FCz
 C8: Representa la diferencia entre el canal 1 y el canal 8 es decir PO4-FCz
 H: La hora en la que se adquirió el segmento de datos. El formato es:

Hora-Minutos-Segundos

Los datos H = 0, quiere decir que hacen parte del mismo chunk de datos adquiridos durante la última hora guardada.

Mark_ID_CC.csv	H	M
	11-23-09	1
	11-23-14	2
	11-23-18	3
	11-23-23	4
	11-23-27	5
	11-23-31	6
	11-23-36	7
	11-23-44	8

H: Representa la hora en la que se hizo el cambio de imagen del estímulo es decir, al llegar la marca, el formato es:

Hora-Minutos-Segundos

M: Representa el número de marca que llegó en ese momento iniciando en **1**, para el estímulo presentado Vernier se evaluando **7** agudezas, una marca por cada una y una **8va** marca al finalizar el registro.

Conclusiones

El proceso de adquisición realizado mediante la herramienta de adquisición (VIAT) realizada en el presente trabajo, permite integrar en un solo proceso la adquisición, estimulación, visualización, procesamiento y manejo de bases de datos de manera fácil, ágil y confiable con el uso de dispositivos portables y de bajo costo como el dispositivo OpenBCI, además permite obtener resultados útiles para posteriores análisis de características visuales. Por lo tanto, esta herramienta puede ser tomada como una base de partida en estudios a futuros, para adquirir una base más extensa de sujetos, sanos y patológicos, bajo una estimulación visual determinada o variable según sea el interés de los usuarios que la implemente, utilizando el protocolo de entorno presentado, que presenta gran similitud al utilizado previamente con otros software de adquisición como OpenViBE, pero reduciendo significativamente tiempos en los procesos de instalación, ejecución y manipulación de los mismos, mirar **(Anexo 1)**.

La interfaz de usuario desarrollada en este proyecto posee las funciones necesarias para facilitar a un usuario la tarea de agregar o buscar un sujeto en una base de datos empleada en MongoDB, implementada de manera local, permite observar datos relevantes de los sujetos y

modificar datos que puedan cambiar en el tiempo. Durante el registro permite visualizar la ubicación de los electrodos en un montaje de electrodos 10-10, medir la impedancia de cada uno de los electrodos posicionados, conectar y desconectar el dispositivo de adquisición OpenBCI y evaluar la existencia de dos pantallas para realizar la estimulación.

Permite observar durante la adquisición la resta de los canales de la zona occipital con el canal de referencia del montaje mencionado, iniciar la estimulación y visualizar el espectro del método de welch para el electrodo de elección y la configuración de Laplace con la combinación de cualquiera de los electrodos actuales. El software permite realizar todas las acciones de manera simultánea gracias al uso de hilos y subprocesos durante su ejecución sin presentar problemas en la interacción de las diferentes tareas. Finalmente, la interfaz permite visualizar señales previamente guardadas en archivos csv o señales adquiridas mediante la herramienta con una conexión directa a la base de datos para facilitar la búsqueda de dichas señales.

Los resultados entregados como archivos csv que entrega la herramienta tienen un propósito de análisis a futuro, debido a que los enfoques en medidas visuales como agudeza visual, usan la información obtenida en varios tamaños de verificación (o frecuencias espaciales) para derivar una medida de agudeza, por medio de la amplitud de la señal en cada tamaño de verificación, a veces combinada con la fase o una medida de ruido. Por otro lado, cuenta con la capacidad de entregar información gráfica de métodos de tiempo frecuencia, que permitirían emplear herramientas de clasificación al ampliar la base de datos de EEG bajo estimulación visual.

El diseño de software pensando en la separación de tareas por la implementación del MVC, permite facilitar la aplicación de mejoras y realizar mantenimiento a futuro del programa, esta GUI puede ser un buen punto de partida para la creación de base de datos y ampliarse a el desarrollo de un software muy completo de análisis y tratamiento de señales de EEG, enfocado al campo de la investigación y diagnóstico de enfermedades neurodegenerativas visuales.

Al confirmar los procesos de adquisición con el software ViAT y el hardware del OpenBCI con una frecuencia de 250 Hz, con pocos electrodo y un protocolo de fácil aplicación en entornos poco controlados, se genera la posibilidad de desarrollar a futuro un nuevo sistema que pueda ser aplicado a la industria, económico y portable para la adquisición de señales de EEG en la industria neuro-oftalmológica y oftalmológica, el cual permita ampliar la cobertura en centros de salud especializados para la detección y tratamiento temprano de patologías visuales asociadas a parámetros visuales caracterizados.

Los módulos que conforman la herramienta de este proyecto demostraron que tiene la capacidad para ser aplicados en procesos de investigación, ya sea para preparar u obtener información de las señales de EEG, además de garantizar un proceso semiautomático de

procesamiento, el cual puede ser replicable para diferentes estímulos visuales bajo las mismas condiciones de adquisición.

Finalmente se describe la herramienta ViAT como una herramienta que permite registrar de manera efectiva los potenciales visuales en estado estacionario con una herramienta portable, una interfaz amigable, con un protocolo de montaje sencillo, eficiente y asequible, que permite favorecer en gran medida la ejecución de pruebas en diferentes entornos, además, el desarrollo y aplicación de medidas de procesamiento cada vez más especializadas y eficientes pueden permitir el diagnóstico de patologías y la rehabilitación de pacientes con deficiencias visuales partiendo desde los parámetros más simples y que entregan más información para este tipo de sujetos como lo es la agudeza visual.

El proyecto se puede descargar en gitlab, desde el siguiente enlace:

<https://gitlab.com/veronicahenaoisaza/hva.git>



UNIVERSIDAD
DE ANTIOQUIA

1 8 0 3

ANEXOS

Anexo 1



MANUAL DE INSTALACIÓN Y PROTOCOLO

Contenido

Dispositivos	2
Software	2
Materiales de entorno y desechables	2
Librerías	3
Instalación	3
Archivos	3
Implementación	4
Preparar el sujeto para la prueba:	4
Preparar dispositivo de adquisición:	5
Abrir OpenBCI	5
Solución de problemas	9
Dentro del OpenBCI GUI	9
Adquisición y estimulación	10
Ejecutar estimulación	13
Estimulación	16
o OpenViBE:	16
o Psychopy:	16
o Python:	17
Entorno:	18

Verónica Henao Isaza
Proyecto Banco de la república
"Evaluation of a platform for the study of visual physiology based on low cost and
portable electroencephalography"
Universidad de Antioquia
2020

UNIVERSIDAD DE ANTIOQUIA

1 8 0 3

PYGAME ESTIMULOS

SSVEP - Adquisición de señales EEG
Funciones - Clases - Métodos

Pygame agrega funcionalidad sobre la excelente biblioteca SDL; es una biblioteca C multiplataforma para controlar multimedia, comparable a DirectX. Pygame es altamente portátil y se ejecuta en casi todas las plataformas y sistemas operativos. Pygame es gratis. Lanzado bajo la licencia LGPL, puede crear código abierto, freeware, shareware y juegos comerciales con él. Vea la licencia para más detalles.

INICIACIÓN

PYGAME.INIT()

Inicializar todos los módulos de pygame importados. Siempre puede inicializar módulos individuales manualmente, pero `pygame.init()` inicializa todos los módulos `pygame` importados, es una forma conveniente de comenzar todo. Las `init()` funciones para módulos individuales generarán excepciones cuando fallen.

PYGAME.GET_INIT()

Devuelve True si pygame está actualmente inicializado

PYGAME.QUIT()

Desinicializar todos los módulos de Pygame. Cuando el intérprete de Python se apaga, este método se llama independientemente, por lo que su programa no debería necesitarlo, excepto cuando quiere terminar sus recursos de pygame y continuar. Es seguro llamar a esta función más de una vez ya que las llamadas repetidas no tienen efecto.

Nota Llamar a `pygame.quit()` no saldrá del programa. Se debe finalizar de la misma manera que finalizará un programa Python normal.

EVENTO

PYGAME.EVENT()

Módulo de pygame para interactuar los eventos como colas. Pygame maneja todos sus mensajes de eventos a través de una cola de eventos. Las rutinas en este módulo lo ayudan a administrar esa cola de eventos. La cola de entrada depende en gran medida del módulo `pygame.display()`.

La cola de eventos tiene un límite superior en la cantidad de eventos que puede contener (128 para SDL 1.2 estándar). Cuando la cola se llena, los nuevos eventos se descartan silenciosamente. Para evitar eventos perdidos, especialmente eventos de entrada que señalan un comando para salir, su programa debe verificar periódicamente los eventos y procesarlos. Para acelerar el procesamiento de la cola, use el `pygame.event.set_blocked()` control de los eventos permitidos en la cola para limitar los eventos.

Para obtener el estado de varios dispositivos de entrada, puede renunciar a la cola de eventos y acceder a los dispositivos de entrada directamente con sus módulos apropiados: `pygame.mouse` para trabajar con el mouse, `pygame.key` para trabajar con el teclado y `pygame.joystick` para interactuar con joysticks, gamepads y trackballs. Si utiliza este método, recuerde que pygame requiere alguna forma de comunicación con el administrador de ventanas del sistema y otras partes de la plataforma. Para mantener Pygame sincronizado

UNIVERSIDAD
DE ANTIOQUIA

1 8 0 3

Anexo 3

**Manual de usuario de herramienta
para la evaluación de la fisiología visual.**



En el marco del proyecto:
**Desarrollo de una herramienta para la evaluación de la fisiología visual usando
electroencefalografía portable y de bajo costo**

Proyecto de Investigación
Pregrado Bioingeniería
Verónica Henao Isaza

Programa de Bioingeniería
Medellín, Antioquia
2020

UNIVERSIDAD
DE ANTIOQUIA

1 8 0 3

Referencias bibliográficas

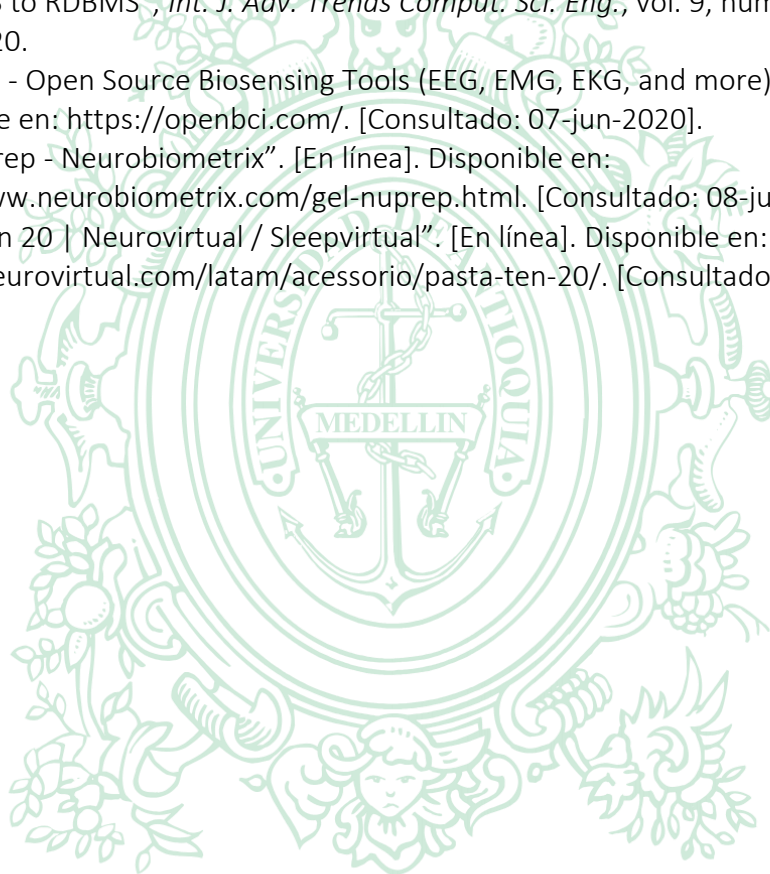
- [1] IMSS, “Diagnóstico y Tratamiento del Síndrome De Privación Sensorial en el Adulto Mayor GPC Guía de Práctica Clínica Catálogo maestro de guías de práctica clínica: IMSS-611-13”.
- [2] G. Motors y W. Europe, “ANÁLISIS DE SITUACIÓN DE SALUD VISUAL EN COLOMBIA”, vol. 519, núm. June, pp. 21–24, 2016.
- [3] C. D. R. Sánchez Hernández, A. González Pérez, y L. Rivadeneyra Espinoza, “Agudeza visual en alumnos de medicina en una universidad privada de Puebla, México.”, *Rev. Médica Risaralda*, vol. 22, núm. 2, pp. 79–82, 2016.
- [4] V. Peterson, Y. Atum, F. Jauregui, I. Gareis, R. Acevedo, y L. Rufiner, “Detección de potenciales evocados relacionados a eventos en interfaces cerebro-computadora mediante transformada wavelet”, *Rev. Ing. Biomédica*, vol. 7, núm. 14, pp. 51–59, 2013.
- [5] “Asociación de neuritis óptica e hipoparatiroidismo”, *Univ. Médica*, vol. 54, núm. 4, pp. 543–548, 2013.
- [6] ONCE, *Discapacidad visual y autonomía personal: enfoque práctico de la rehabilitación*. 2011.
- [7] R. P. Simon, D. A. Greenberg, y M. J. Aminoff, “Alteraciones de la vista”, en *Neurología clínica*, 7e, 2015.
- [8] J. F. Reyes Domínguez y A. Castillo Angulo, “Cobertura Del Servicio Público En Salud Visual En Bogotá, Capital Cosmopolita”, *Cienc. Tecnol. para la Salud Vis. y Ocul.*, vol. 16, núm. 1, pp. 45–71, 2018.
- [9] MINISTERIO DE SALUD Y PROTECCIÓN SOCIAL, “LINEAMIENTO PARA LA IMPLEMENTACION DE ACTIVIDADES DE PROMOCION DE LA SALUD VISUAL, CONTROL DE ALTERACIONES VISUALES Y DISCAPACIDAD VISUAL EVITABLE (ESTRATEGIA VISION 2020) Direccion de Promoción y Prevención Subdirección de Enfermedades No Transmisibles”, pp. 1–49, 2012.
- [10] E. OJEDA, “Potenciales Evocados Visuales Y Electroretinograma”, *Guía Neurológica*, pp. 127–134, 2005.
- [11] J. L. Sirvent, J. M. Azorín, E. Iáñez, A. Úbeda, y E. Fernández, “Interfaz Cerebral no Invasiva basada en Potenciales Evocados para el Control de un Brazo Robot”, *RIAI - Rev. Iberoam. Autom. e Inform. Ind.*, vol. 8, núm. 2, pp. 103–111, 2011.
- [12] P. Durka, R. Kus, J. Zygiereicz, P. Milanowski, y G. Garcia, “High-frequency SSVEP responses parametrized by multichannel matching pursuit”, *Front. Neuroinform.*, 2009.
- [13] “Universidad autónoma de madrid”, 2018.
- [14] R. Wang, W. Wu, K. Iramina, y S. Ge, “The combination of CCA and PSDA detection methods in a SSVEP-BCI system”, en *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, 2015.
- [15] A. M. Norcia y C. W. Tyler, “Spatial frequency sweep VEP: Visual acuity during the first year of life”, *Vision Research*. 1985.
- [16] M. Bach, J. P. Maurer, y M. E. Wolf, “Visual evoked potential-based acuity assessment in normal vision, artificially degraded vision, and in patients”, *Br. J. Ophthalmol.*, vol. 92, núm. 3, pp. 396–403, 2008.
- [17] P. W. Mirowski, Y. LeCun, D. Madhavan, y R. Kuzniecky, “Comparing SVM and Convolutional Networks for Epileptic Seizure”, *2008 IEEE Work. Mach. Learn. Signal Process.*, 2008.
- [18] X. Li, X. Jia, G. Xun, y A. Zhang, “Improving EEG feature learning via synchronized facial

- video”, en *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*, 2015.
- [19] A. Setiawan, A. D. Wibawa, E. S. Pane, y M. H. Purnomo, “EEG-based mental fatigue detection using cognitive tests and RVM classification”, *Proceeding - 2019 Int. Conf. Artif. Intell. Inf. Technol. ICAIIT 2019*, pp. 180–185, 2019.
 - [20] O. R. Daniela y H. I. Ver, “Estudio de SSVEP en visi ´ on monocular y binocular”.
 - [21] P. M. Glover, S. Eldeghaidy, T. R. Mistry, y P. A. Gowland, “Measurement of visual evoked potential during and after periods of pulsed magnetic field exposure”, *J. Magn. Reson. Imaging*, vol. 26, núm. 5, pp. 1353–1356, 2007.
 - [22] “Guyton y Hall. Tratado de fisiología médica - 9788491130246 | Elsevier España”. [En línea]. Disponible en: <https://tienda.elsevier.es/guyton-y-hall-tratado-de-fisiologia-medica-9788491130246.html>. [Consultado: 07-jun-2020].
 - [23] R. F. Dougherty, V. M. Koch, A. A. Brewer, B. Fischer, J. Modersitzki, y B. A. Wandell, “Visual field representations and locations of visual areas v1/2/3 in human visual cortex”, *J. Vis.*, vol. 3, núm. 10, pp. 586–598, oct. 2003.
 - [24] “(PDF) Visual Field Maps in Human Cortex | Alyssa Brewer - Academia.edu”. [En línea]. Disponible en: https://www.academia.edu/16521967/Visual_Field_Maps_in_Human_Cortex. [Consultado: 07-jun-2020].
 - [25] W. V. Good y C. Hou, “Normal vernier acuity in infants with delayed visual maturation”, *Am. J. Ophthalmol.*, vol. 138, núm. 1, pp. 140–142, 2004.
 - [26] T. Talamillo, “Nociones elementales para la interpretación del EEG”, *Enfermería Docente*, vol. 94, pp. 29–33, 2011.
 - [27] L. F. Nicolas-Alonso y J. Gomez-Gil, “Brain computer interfaces, a review”, *Sensors*, vol. 12, núm. 2. Sensors (Basel), pp. 1211–1279, feb-2012.
 - [28] S. Sanei y J. A. Chambers, *EEG Signal Processing*. John Wiley and Sons, 2013.
 - [29] E. Genç, M. L. Schölvinck, J. Bergmann, W. Singer, y A. Kohler, “Functional connectivity patterns of visual cortex reflect its anatomical organization”, *Cereb. Cortex*, vol. 26, núm. 9, pp. 3719–3731, 2016.
 - [30] H. Setiawan, W. R. Islamiyah, A. D. Wibawa, y M. H. Purnomo, “Identifying EEG Parameters to Monitor Stroke Rehabilitation using Individual Analysis”, en *Proceedings - 2019 International Seminar on Intelligent Technology and Its Application, ISITIA 2019*, 2019, pp. 337–342.
 - [31] “Medical Devices and Human Engineering - Google Libros”. [En línea]. Disponible en: [https://books.google.com.co/books?id=r4uZBQAAQBAJ&pg=SA2-PA19&lpg=SA2-PA19&dq=Encyclopedia,+Medical+Devices+and+Human+Engineering,+vol.+5.+2006.&source=bl&ots=Vd9xxirJY2&sig=ACfU3U1fU3OtZtCSkzQev44Li7T7MJnGGA&hl=es-419&sa=X&ved=2ahUKEwjW8tz6ovHpAhWvhOAKHZiHDaoQ6AEwAHoECAkQAQ#v=onepage&q=Encyclopedia%2C Medical Devices and Human Engineering%2C vol. 5. 2006.&f=false](https://books.google.com.co/books?id=r4uZBQAAQBAJ&pg=SA2-PA19&lpg=SA2-PA19&dq=Encyclopedia,+Medical+Devices+and+Human+Engineering,+vol.+5.+2006.&source=bl&ots=Vd9xxirJY2&sig=ACfU3U1fU3OtZtCSkzQev44Li7T7MJnGGA&hl=es-419&sa=X&ved=2ahUKEwjW8tz6ovHpAhWvhOAKHZiHDaoQ6AEwAHoECAkQAQ#v=onepage&q=Encyclopedia%2C%20Medical%20Devices%20and%20Human%20Engineering%2C%20vol.%205.%202006.&f=false). [Consultado: 07-jun-2020].
 - [32] F. Ramos-Argüelles, G. Morales, S. Egozcue, R. M. Pabón, y M. T. Alonso, “Técnicas básicas de electroencefalografía: principios y aplicaciones clínicas.”, *An. Sist. Sanit. Navar.*, vol. 32 Suppl 3, pp. 69–82, 2009.
 - [33] J. W. Osselton, “Acquisition of EEG data by bipolar unipolar and average reference methods: a theoretical comparison”, *Electroencephalogr. Clin. Neurophysiol.*, 1965.
 - [34] V. Jurcak, D. Tsuzuki, y I. Dan, “10/20, 10/10, and 10/5 systems revisited: Their validity as relative head-surface-based positioning systems”, *Neuroimage*, vol. 34, núm. 4, pp.

- 1600–1611, 2007.
- [35] C. Kapeller, C. Hintermuller, M. Abu-Alqumsan, R. Pruckl, A. Peer, y C. Guger, “A BCI using VEP for continuous control of a mobile robot”, *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, núm. February 2018, pp. 5254–5257, 2013.
 - [36] R. Gulbinaite, T. Van Viegen, M. Wieling, M. X. Cohen, y R. Vanrullen, “Individual alpha peak frequency predicts 10 hz flicker effects on selective attention”, *J. Neurosci.*, vol. 37, núm. 42, pp. 10173–10184, 2017.
 - [37] D. J. Peterson, G. Gurariy, G. G. Dimotsantos, H. Arciniega, M. E. Berryhill, y G. P. Caplovitz, “The steady-state visual evoked potential reveals neural correlates of the items encoded into visual working memory”, *Neuropsychologia*, vol. 63, pp. 145–153, 2014.
 - [38] Z. Işcan y V. V. Nikulin, “Steady state visual evoked potential (SSVEP) based brain-computer interface (BCI) performance under different perturbations”, *PLoS One*, vol. 13, núm. 1, pp. 1–17, 2018.
 - [39] Y. Zhang, P. Xu, T. Liu, J. Hu, R. Zhang, y D. Yao, “Multiple frequencies sequential coding for SSVEP-based brain-computer interface”, *PLoS One*, vol. 7, núm. 3, 2012.
 - [40] C. H. Wu *et al.*, “Frequency recognition in an SSVEP-based brain computer interface using empirical mode decomposition and refined generalized zero-crossing”, *J. Neurosci. Methods*, vol. 196, núm. 1, pp. 170–181, 2011.
 - [41] Y. Zhang, P. Xu, Y. Huang, K. Cheng, y D. Yao, “SSVEP Response Is Related to Functional Brain Network Topology Entrained by the Flickering Stimulus”, *PLoS One*, vol. 8, núm. 9, 2013.
 - [42] M. R y V. G, “Factores que afectan a la agudeza visual”, *Man. Optom.*, p. 21.
 - [43] A. M. Skoczinski y A. M. Norcia, “Development of VEP vernier acuity and grating acuity in human infants”, *Investig. Ophthalmol. Vis. Sci.*, vol. 40, núm. 10, pp. 2411–2417, 1999.
 - [44] M. Phothisonothai y K. Watanabe, “Time-frequency analysis of duty cycle changing on steady-state visual evoked potential: EEG recording”, *2014 Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. APSIPA 2014*, 2014.
 - [45] K. Delgado *et al.*, “2007-9621-Au-29-E1672”, pp. 1–24, 2019.
 - [46] M. Bach y S. P. Heinrich, “Acuity VEP: improved with machine learning”, *Doc. Ophthalmol.*, vol. 139, núm. 2, pp. 113–122, 2019.
 - [47] J. Delgado Rivera, “Análisis del electroencefalograma con transformada de Fourier y modelos paramétricos”, *Ing. e Investig.*, núm. 23, pp. 7–13, 1991.
 - [48] P. D. Welch, “The Use of Fast Fourier Transform for the Estimation of Power Spectra”, *Digit. Signal Process.*, núm. 2, pp. 532–574, 1975.
 - [49] A. V. Oppenheim y R. W. Schaffer, *Digital signal processing*. Prentice-Hall, 1975.
 - [50] ro mexico, “Tratamiento Digital de Señales 4 Ed. - John G. Proakis, Dimitris G. Manolakis”. .
 - [51] L. S. M. Gómez, “Diseño de Interfaces de Usuario Principios, Prototipos y Heurísticas para Evaluación”, núm. January 2000, p. 13, 2000.
 - [52] E. H. Sibley, R. Molich, y J. Nielsen, “Improving a H & man- Computer Dialogue”, vol. 33, núm. 3, 1990.
 - [53] J. Nielsen y R. Molich, “Heuristic evaluation of user interfaces”, *Conf. Hum. Factors Comput. Syst. - Proc.*, núm. April, pp. 249–256, 1990.
 - [54] J. Nielsen, “How to Conduct a Heuristic Evaluation”, *Useitcom*, pp. 1–11, 2002.
 - [55] A. Majeed y I. Rauf, “MVC Architecture: A Detailed Insight to the Modern Web

Applications Development”, vol. 1, pp. 1–7.

- [56] “What is NoSQL? NoSQL Databases Explained | MongoDB”. [En línea]. Disponible en: <https://www.mongodb.com/nosql-explained>. [Consultado: 28-jun-2020].
- [57] “NoSQL vs SQL Databases | MongoDB”. [En línea]. Disponible en: <https://www.mongodb.com/nosql-explained/nosql-vs-sql>. [Consultado: 28-jun-2020].
- [58] “La base de datos líder del mercado para aplicaciones modernas | MongoDB”. [En línea]. Disponible en: <https://www.mongodb.com/es>. [Consultado: 28-jun-2020].
- [59] J. Kachaoui y A. Belangour, “MQL2SQL: A proposal data transformation algorithm from mongoDB to RDBMS”, *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 9, núm. 2, pp. 2457–2463, 2020.
- [60] “OpenBCI - Open Source Biosensing Tools (EEG, EMG, EKG, and more)”. [En línea]. Disponible en: <https://openbci.com/>. [Consultado: 07-jun-2020].
- [61] “Gel Nuprep - Neurobiometrix”. [En línea]. Disponible en: <http://www.neurobiometrix.com/gel-nuprep.html>. [Consultado: 08-jun-2020].
- [62] “Pasta Ten 20 | Neurovirtual / Sleepvirtual”. [En línea]. Disponible en: <https://neurovirtual.com/latam/acessorio/pasta-ten-20/>. [Consultado: 08-jun-2020].



UNIVERSIDAD DE ANTIOQUIA

1 8 0 3

MANUAL DE INSTALACIÓN Y PROTOCOLO

Contenido

Dispositivos.....	37
Software	38
Materiales de entorno y desechables.....	38
Librerías	38
Instalación.....	39
Archivos	39
Implementación.....	39
Preparar al sujeto para la prueba:	39
Preparar dispositivo de adquisición:.....	41
Abrir OpenBCI.....	41
Solución de problemas	44
Dentro del OpenBCI GUI.....	44
Adquisición y estimulación	45
Ejecutar estimulación	48
Estimulación	49
o OpenViBE:.....	49
o Psychopy:.....	49
o Python:.....	50
Entorno:.....	50

Dispositivos

- Computador 64 bits o 32 bits con pantalla.
- Pantalla Samsung SyncMaster 2243 LNX o similar, conservando la resolución de (1680x1050)

- Cyton Biosensing Board (8 canales)
- OpenBCI Dongle
- Paquete de baterías AA de 6V y (x4) baterías AA (baterías no incluidas)
- Gold Cup Electrodes (10)

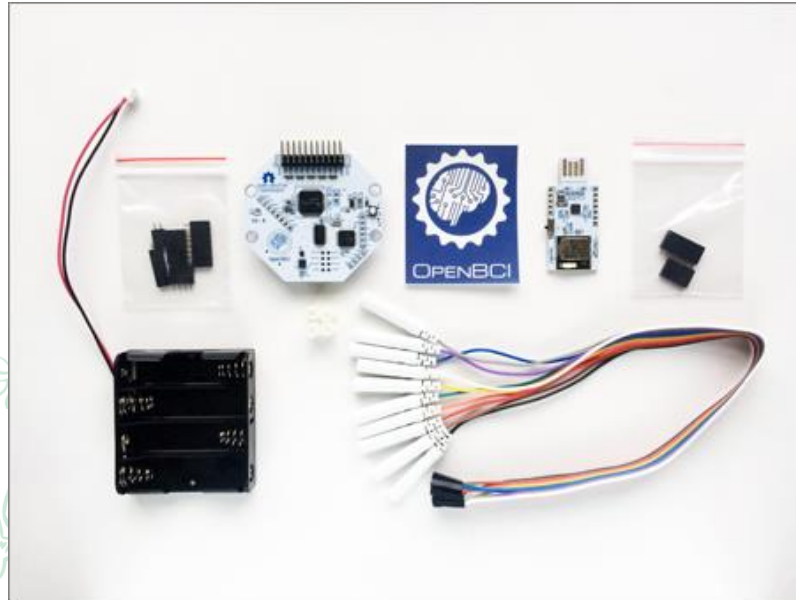


Figura 15. OpenBCI

Software

- Python 2.7
- OpenViBE 2.2.0
- PsychoPy3
- OpenBCI GUI

Se debe ser coherente con la instalación de todos los programas conservando la cantidad de bits en todos los programas.

Materiales de entorno y desechables

- Soporte mecánico
- Gafas protectoras
- Silla con graduación de altura
- Mesa de 1m de anchura
- Gel conductor Nuprep
- Pasta conductora Ten20
- Tabla HOTV
- Gasa estéril
- Alcohol

Librerías

PyGame -> Se adjunta documentación de esta

Instalación

La información para la instalación a continuación se realiza para Windows, pero todos los softwares se encuentran disponibles en Linux siguiendo los mismos requisitos de compatibilidad.

Comience la instalación en el orden que se presentan los softwares, comenzando por Python 2.7, luego OpenViBE, seguido por PsychoPy3 y finalmente OpenBCI GUI.

Tabla 4. Software necesarios.

Software	Link	Comentario
Python 2.7	https://www.python.org/download/releases/2.7/	Necesario para OpenViBE debido a que el módulo de Python es compatible únicamente con esta versión
OpenViBE 2.2.0	http://openvibe.inria.fr/downloads/	
Python 3.6	https://www.python.org/downloads/release/python-360/	Necesario para Psychopy3 de lo contrario deberá instalar una versión anterior de Psychopy compatible con Python 2.7
Psychopy3	https://www.psychopy.org/download.html	pip install psychopy
OpenBCI GUI v4.2.0	https://openbci.com/index.php/downloads	Es necesario descargar el controlador más reciente
Drivers OpenBCI	https://www.ftdichip.com/Drivers/VCP.htm	2017-08-30

Archivos

Una vez termine la instalación de los softwares podrá comenzar a ejecutar los archivos utilizados contenidos en un repositorio Git

- Códigos y estímulos: https://github.com/danni9310/Thesis_stimuli.git

Implementación

Para realizar un registro se debe seguir el siguiente protocolo

- **Preparar al sujeto para la prueba:**
 - Se le realiza la prueba de agudeza con tablas de HOTV
 - Se evalúa ojo dominante de la siguiente manera: Primero extiende completamente tus brazos y ubícalos a la altura de tus ojos. A continuación, con las palmas paralelas a tu cuerpo, forma un triángulo entre tus pulgares y dedos índices, dejando un agujero a través del que puedas ver. Finalmente, mira a través del agujero hacia un objeto que no exceda el tamaño de este, cierra el ojo izquierdo y luego el derecho, aquel con el que sigas viendo la imagen centrada dentro del agujero entre las dos manos es tu ojo dominante.

1 8 0 3

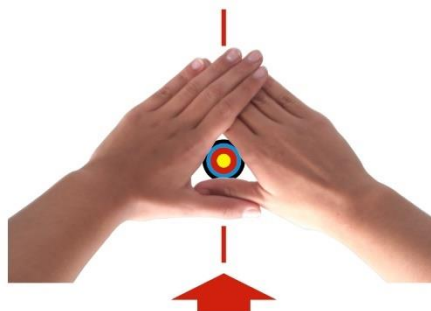


Figura 16. Fijación visual.

- Se ubica el sujeto en una silla graduable frente a la pantalla
- Se realiza una limpieza del cuero cabelludo y las orejas con alcohol
- Se aplica el gel conductor Nuprep en las zonas en las que se colocara cada uno de los electrodos (Zona occipital, sutura sagital y las orejas)
- Se realiza el montaje de los electrodos con el sistema 10-10, aplicando la pasta conductora Ten20 en cada uno de los electrodos. Se describe la forma de colocar los electrodos en orden de derecha a izquierda.

Inion: es la proyección más prominente del hueso occipital en la parte posterior inferior del cráneo humano

Nasion: es el punto de intersección del hueso frontal y de dos huesos nasales del cráneo humano.

Pulgada: Medida de longitud del sistema inglés, de símbolo in, que equivale a 25,40 mm.

- Electrodo 1: Oreja derecha
- Electrodo 2: Posición FCz; se mide la cabeza del sujeto con un metro desde el inion hasta el nasion, y se calcula el 60% y se posiciona en este valor midiendo desde el inion FCz.
- Electrodo 3: Posición OZ; se ubica el inion del sujeto y se acomoda una pulgada sobre este.
- Electrodo 4: Posición O1; se ubica Oz y se mide una pulgada diagonal hacia la izquierda.
- Electrodo 5: Posición PO7; se ubica O1 y se mide una pulgada diagonal hacia la izquierda.
- Electrodo 6: Posición O2; se ubica Oz y se mide una pulgada diagonal hacia la derecha.
- Electrodo 7: Posición PO8; se ubica O2 y se mide una pulgada diagonal hacia la izquierda.
- Electrodo 8: Posición PO3; se ubica O1 y se mide una pulgada sobre el mismo.
- Electrodo 9: Posición PO4; se ubica O2 y se mide una pulgada sobre el mismo.
- Electrodo 10: Oreja izquierda

1 8 0 3

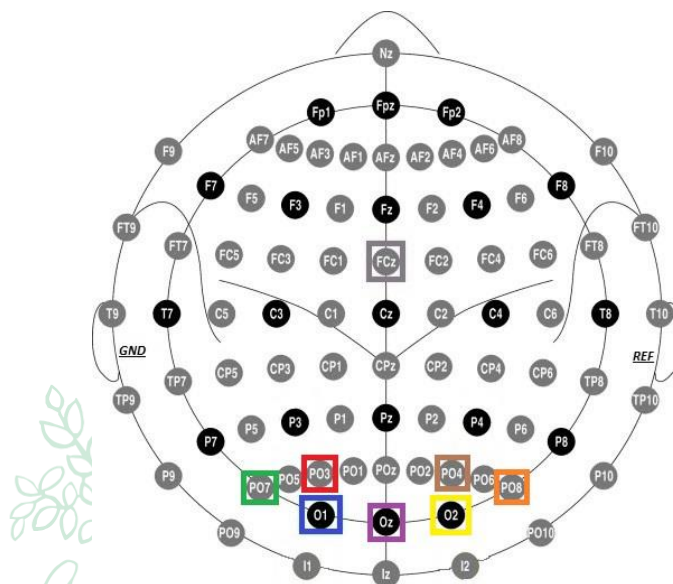


Figura 17. Montaje de electrodos utilizado.

Preparar dispositivo de adquisición:

Abrir OpenBCI

- Conectar el Dongle USB OpenBCI: Enchufe esto (¡mirando hacia arriba!) Y debería ver un LED azul encendido y permanecer encendido, así como un LED rojo parpadeando. Nota: asegúrese de que su Dongle USB esté cambiado a GPIO 6 y no RESET. El interruptor debe estar más cerca de su computadora como se ve en la imagen de la derecha.
- Cambie su placa Cyton a PC: Asegúrese de mover el pequeño interruptor en el lado derecho de la placa de "APAGADO" a "PC". Tan pronto como lo haga, debería ver que se enciende un LED azul. Si no lo hace, presione el botón de reinicio (RST) justo a la izquierda del interruptor. Si el LED aún no se enciende, asegúrese de tener la batería llena. Si está seguro de que sus baterías están completamente cargadas, consulte la sección de hardware de nuestro Foro.

Nota: es importante conectar su Dongle antes de encender su placa Cyton. A veces, si el flujo de datos parece interrumpido, es posible que deba desconectar su Dongle USB y apagar su placa Cyton. Asegúrese de enchufar primero su Dongle USB, luego encienda su placa después.

1 8 0 3

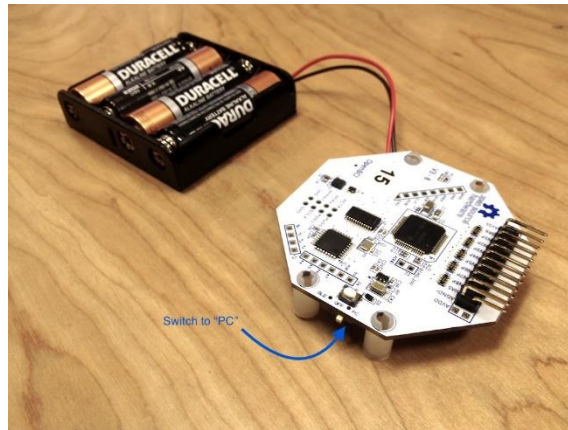


Figura 18. Placa del dispositivo.

- En el computador iniciar la aplicación GUI instalada y ejecutarla como administrador.
- Presionar LIVE (from Cyton): Para conectarse a su Cyton, debe especificar la fuente de datos que se encuentra LIVE (from Cyton) en la primera sección del Panel de control del sistema. Antes de START SYSTEM presionar el botón, debe configurar su placa Cyton (siga los pasos a continuación)

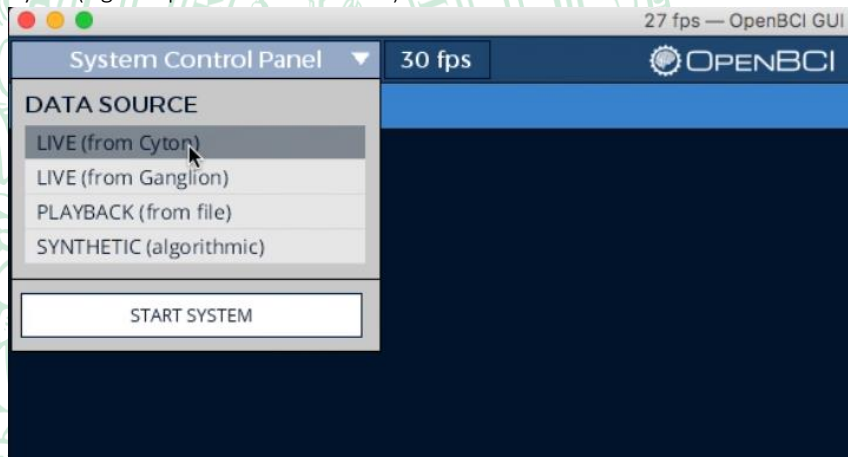


Figura 19. Interfaz OpenBCI.

- Seleccionar transferencia de datos por serial:

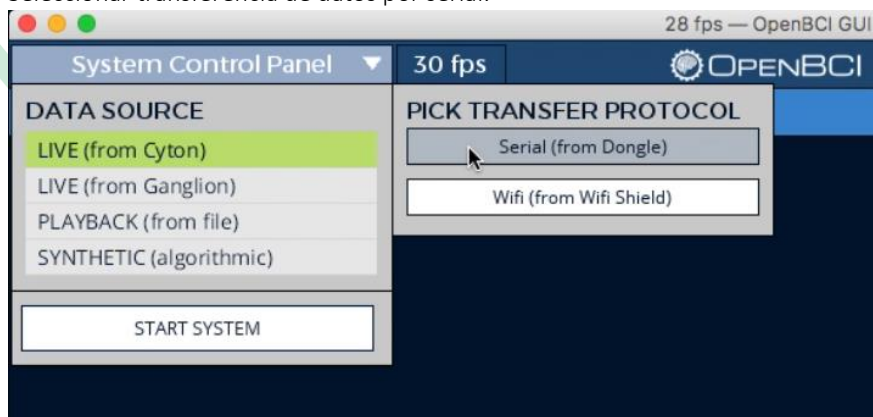


Figura 20. Conexión puerto USB.

- Seleccionar el puerto serial: Para saber en que puerto se encuentra el dispositivo Inicio -> Administrador de dispositivos -> Puertos COM -> USB serial Port (COM#). Por defecto OpenBCI también le mostrara los puertos

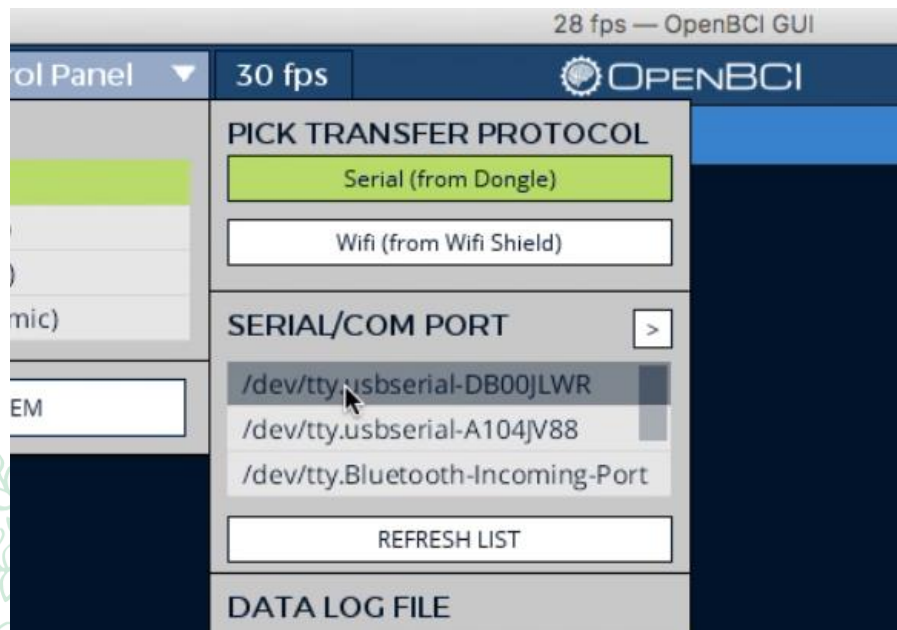


Figura 21. Puerto.

Si está utilizando Windows, aparecerá como:

COM #

El nombre del puerto de su dongle USB probablemente estará en la parte superior de la lista. Si no lo ves:

Asegúrese de que su dongle esté enchufado y cambiado a GPIO 6 (no RESET)

Haga clic en el botón ACTUALIZAR LISTA en la sección PUERTO SERIAL / COM del subpanel

Si todavía tiene problemas para encontrar el nombre del puerto de su Dongle USB, consulte el Foro sobre la depuración de su conexión de hardware.

- Seleccionar los canales: La configuración de CHANNEL COUNT está predeterminada en 8. Si está trabajando con un módulo de margarita OpenBCI y un sistema de placa Cyton (16 canales), asegúrese de hacer clic en el botón 16 CHANNELS antes de iniciar su sistema.



Figura 22. Canales

- Presione "Iniciar sistema": ¡Ahora estás listo para iniciar el sistema! Presione el botón INICIAR SISTEMA y espere a que la GUI de OpenBCI establezca una conexión con su placa Cyton. Esto generalmente toma ~ 5 segundos. Durante este tiempo, la línea de ayuda en la parte inferior de la interfaz gráfica de usuario de OpenBCI debe parpadear las palabras: "Intentando establecer una conexión con su placa OpenBCI ..."

1 8 0 3

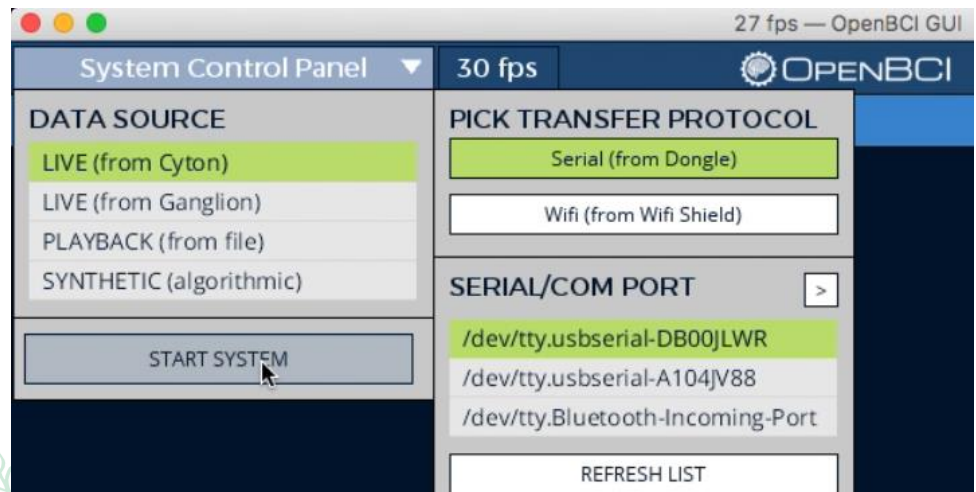


Figura 23. Iniciar sistema.

▪ Solución de problemas

Si la inicialización falla, intente los siguientes pasos en orden:

1. Asegurarse de haber seleccionado el puerto serie / COM correcto
2. Apague su placa Cyton y desconecte su Dongle USB. Luego, vuelva a conectar su Dongle USB y encienda su placa Cyton en ese orden. Luego intente reiniciar el sistema, pero presione el botón START SYSTEM nuevamente.
3. Si esto no funciona, intente reiniciar la aplicación GUI OpenBCI y rehaga el paso 2 anterior. Luego reconfigure la configuración del PANEL DE CONTROL DEL SISTEMA y vuelva a intentar INICIAR SISTEMA.
4. Asegúrese de que sus baterías estén completamente cargadas y vuelva a intentar los pasos anteriores.

▪ Dentro del OpenBCI GUI

Esta herramienta se utiliza en este caso únicamente para observar la señal en primera instancia y verificar la impedancia en cada electrodo para esto entonces inicie la adquisición de los datos

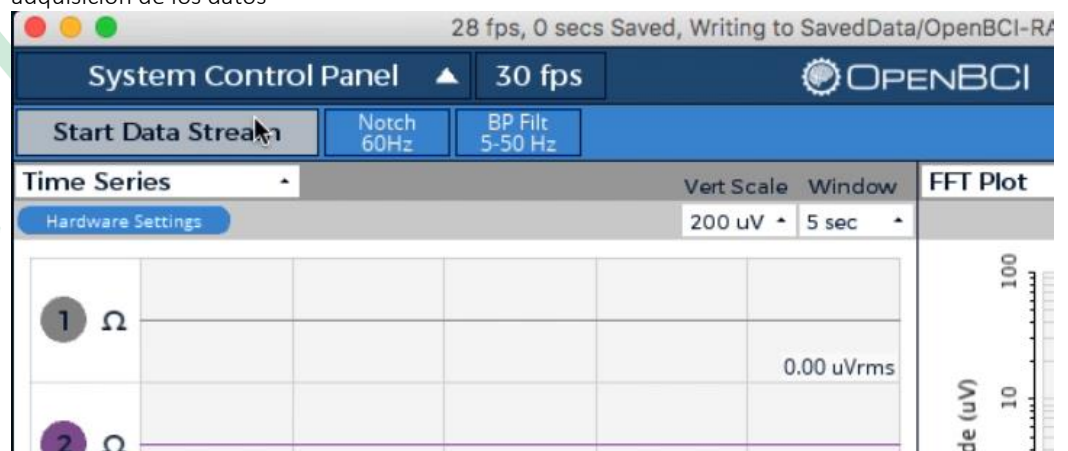


Figura 24. Iniciar visualización.

Comenzara a observar la señal en uVrms en cada canal, para medir la impedancia presione el símbolo Ω y a continuación comenzara a observar la impedancia en cada canal, procure que este por debajo de 30 kohm

Adquisición y estimulación

- Detener el envío de datos desde el OpenBCI y cerrar el OpenBCI GUI, si no se realiza este paso no podrá ejecutar los siguientes.
- Abrir el OpenViBE Acquisition Server como administrador: Conecte su placa OpenBCI y asegúrese de que se reconozca como un puerto COM y que su latencia esté establecida en 1 ms.
- Al abrir la ventana de OpenViBE Acquisition Server, diríjase a las opciones del Driver y seleccione OpenBCI
- Luego en propiedades del Driver, seleccionar el número del puerto que se verificó en el administrador de dispositivos previamente, ingresar a cambiar nombres de canales en la parte inferior y nombrarlos en el orden especificado en el montaje de electrodos, aplique.

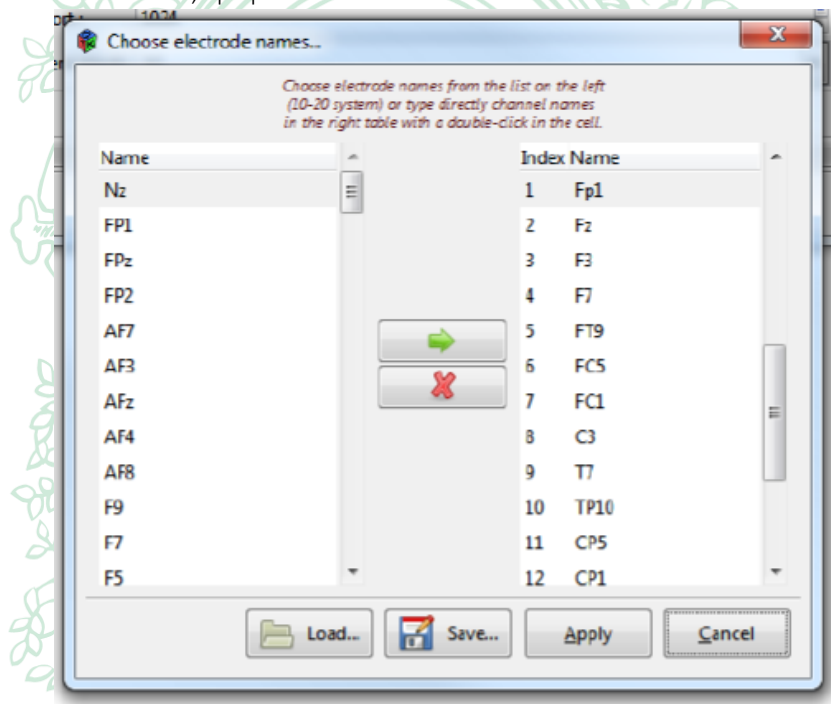


Figura 25. Seleccionar canales.

- En el menú de Preferencia SAS, cambie la tolerancia de deriva de 2 ms (predeterminado) a 10 ms.
- Verificar la conexión en 1024 y el muestreo en 32.
- Presione Conectar. Si se produce un error, solucione el problema:
- Mire la ventana de terminal que abre el SAS. Tiene un informe detallado sobre la condición del SAS.
- A menudo, al presionar el botón de reinicio en la placa OpenBCI o al desconectar / conectar el dongle se solucionarán los problemas de conexión.
- Si el error informa que no puede abrir el puerto seleccionado, asegúrese de que el puerto COM seleccionado en las opciones del controlador sea el mismo que su placa.
- Verifique también que la conexión con el OpenBCI no este activa en otro lugar como en el OpenBCI GUI

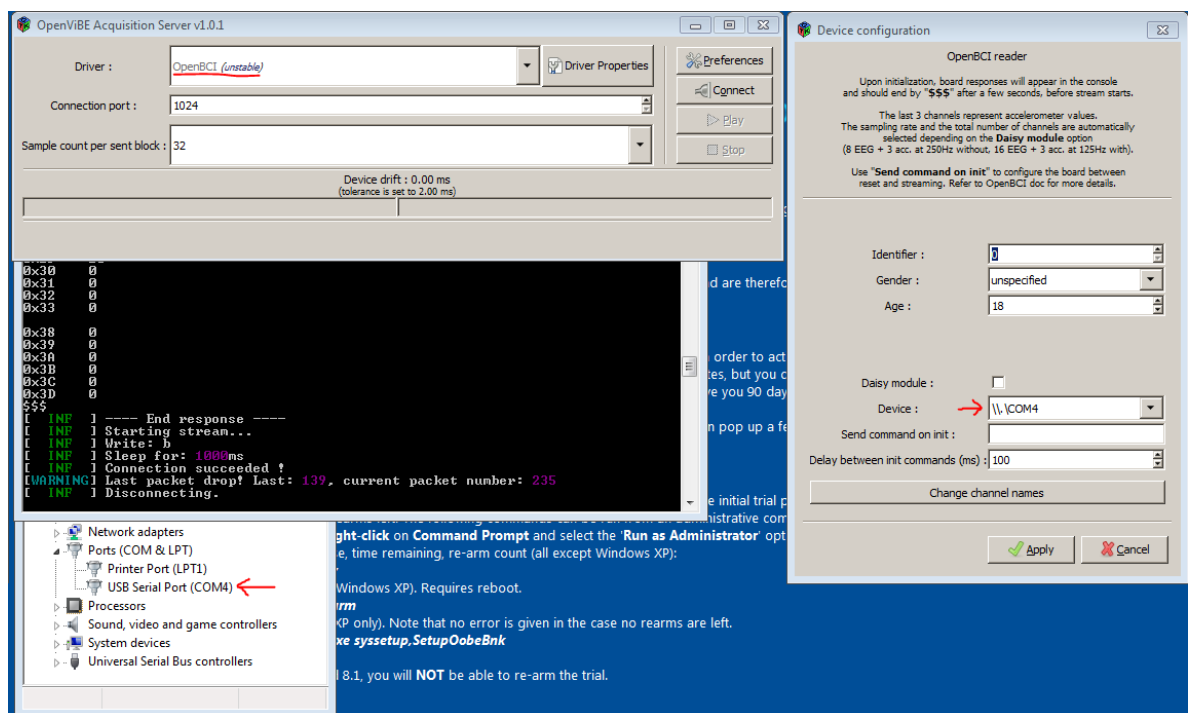


Figura 26. Adquisición.

- Abra el OpenViBE Designer (No en modo administrador):

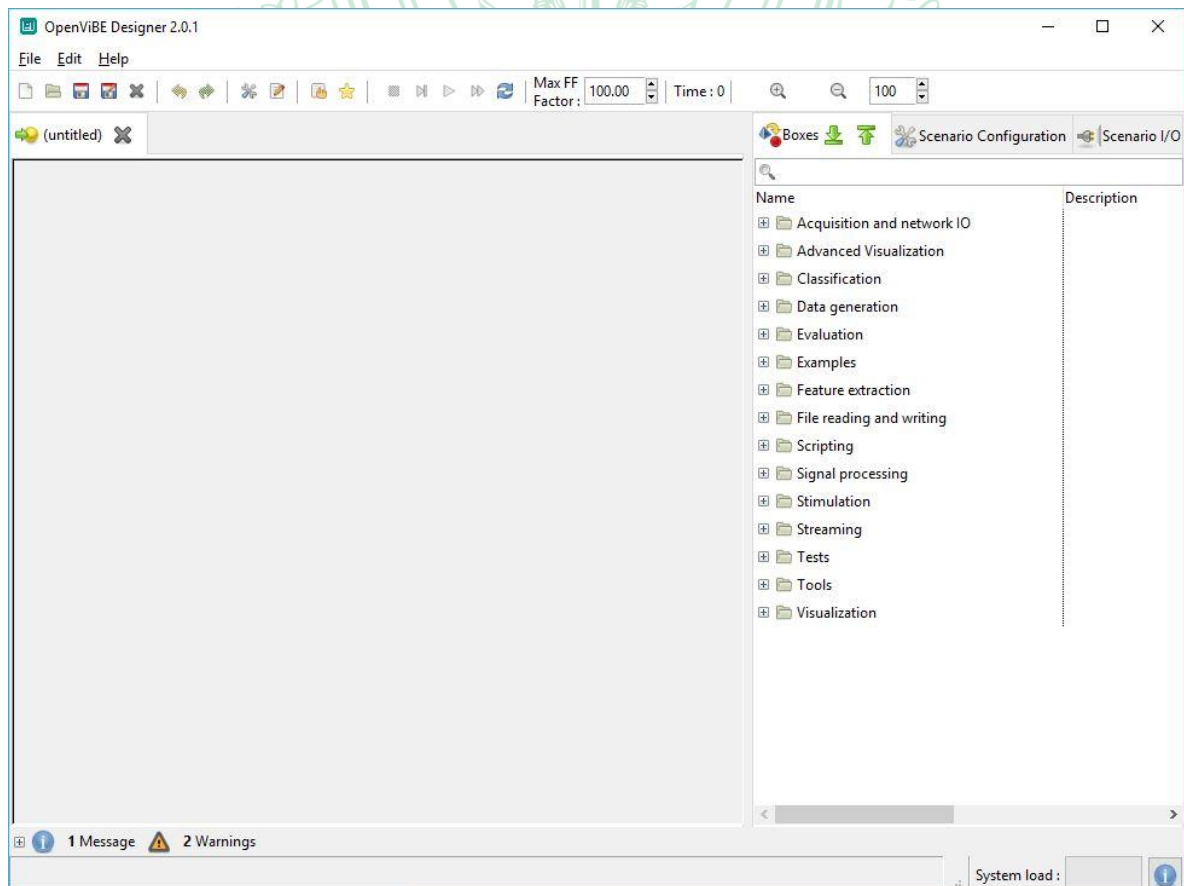


Figura 27. Adquisición designer.

En los archivos suministrados por el Git, ingrese a la carpeta OpenViBE y busque los que tiene extensión .mxs y ábralos.

Para encontrarlos con mayor facilidad se recomienda agregar la parte buscando la carpeta y presionando el símbolo +

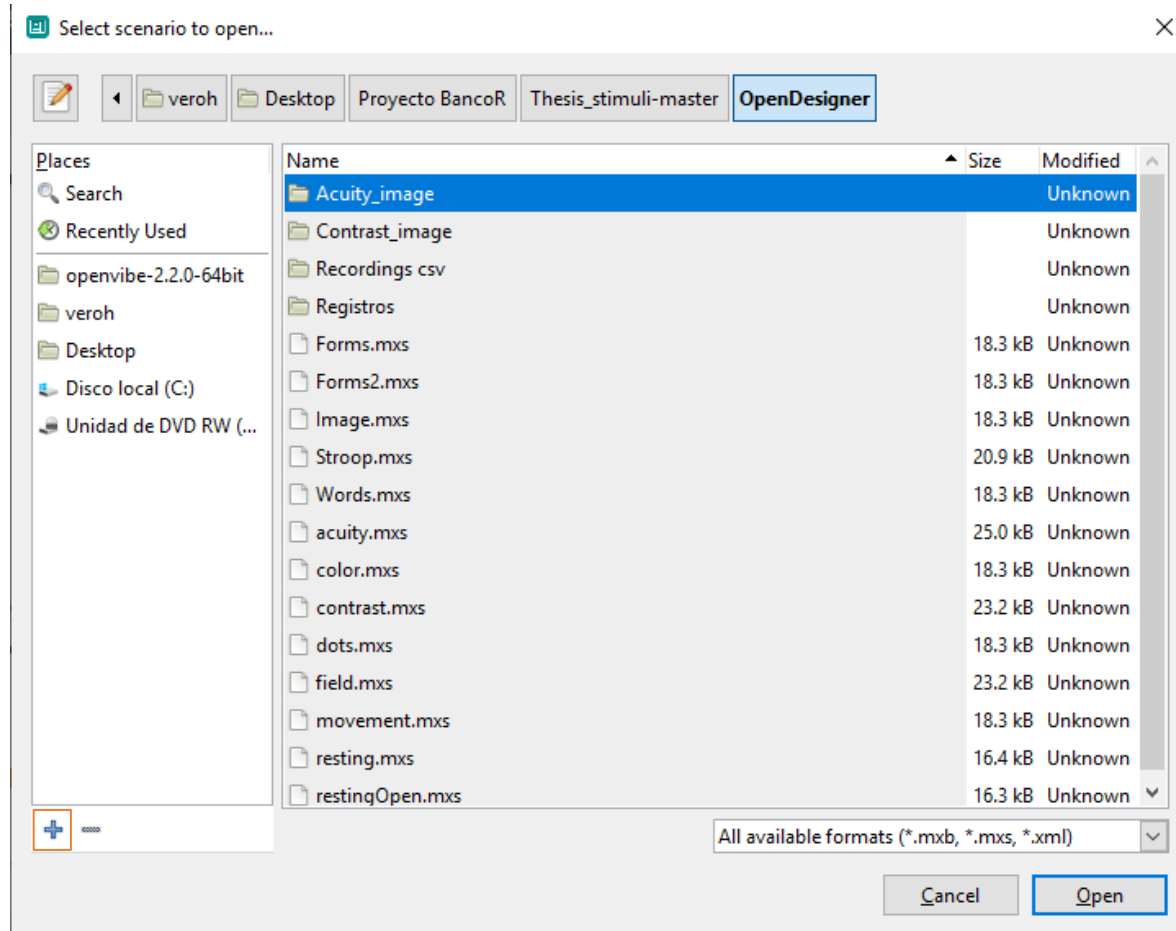


Figura 28. Ubicación.

- El orden de estimulación es el siguiente, se presenta en orden, el archivo de estimulación y el archivo de almacenamiento en cada caso, los estímulos que cuentan con más de un archivo de almacenamiento se deben ir tocando al final de cada estimulación:

Tabla 5. Archivos.

Estimulación	Archivo de adquisición	Archivo de almacenamiento	Ejecución
Reposo ojos cerrados	resting.mxs	resting.csv	OpenViBE
Reposo ojos abiertos	restingOpen.mxs	resting_open_right.csv	OpenViBE
		resting_open_left.csv	OpenViBE
		resting_open_both.csv	OpenViBE
Agudeza	Acuity.mxs	acuity.csv	OpenViBE
Contraste	Contrast.mxs	contrast.csv	OpenViBE
Campo visual	Field.mxs	field_right.csv	OpenViBE
		field_left.csv	OpenViBE
		field_both.csv	OpenViBE
Movimiento	dots.mxs	dots.csv	Psychopy
Formas	Forms.mxs	forms.csv	Python
Color	Color.mxs	color.csv	Python

- Los archivos se seleccionan en la caja de CSV File Writer, dando doble click sobre ella y en el campo de "Filename" colocar la ruta del archivo o presionar sobre la carpeta y buscarla en el directorio.
- Los archivos de estimulación están realizados en Python por lo tanto diríjase a la caja de Python scripting y en el campo Script busque en el directorio el archivo .py que se encuentra en la misma carpeta de los .mxs encontrara un archivo .py por cada estímulo ejecutado en OpenViBE, excepto los reposos que no tienen ninguna estimulación.
- Luego de seleccionar el archivo .py de interés diríjase a el botón con el lápiz, se abrirá un archivo .txt con el código Python, busque "path_initial" y coloque la ruta en que se encuentre "OpenDesigner/ Scripts in Python/...+Estimulación" en su computador, esto permitirá que el código encuentre las imágenes almacenadas en este lugar y poder ejecutar el estímulo.
- Si se busca modificar alguna característica del estímulo por favor diríjase a los tutoriales para diseño de estímulos de OpenViBE.

Ejecutar estimulación

En la barra superior del OpenViBE encontrara los controles que le permitirán inicial o detener la estimulación. Asegúrese de que el OpenViBE Acquisition Server se encuentra activo y recibiendo datos, es decir, la barra en la parte inferior derecha debe aparecer en verde. Presione "Conectar" y luego "Iniciar" este proceso se hace para cada archivo que requiera ser registrado y guardado.

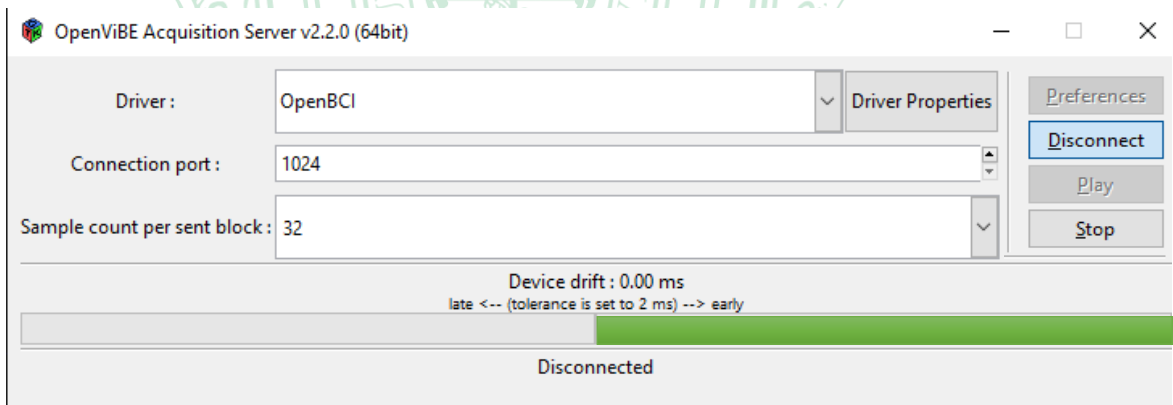


Figura 29. Servidor.

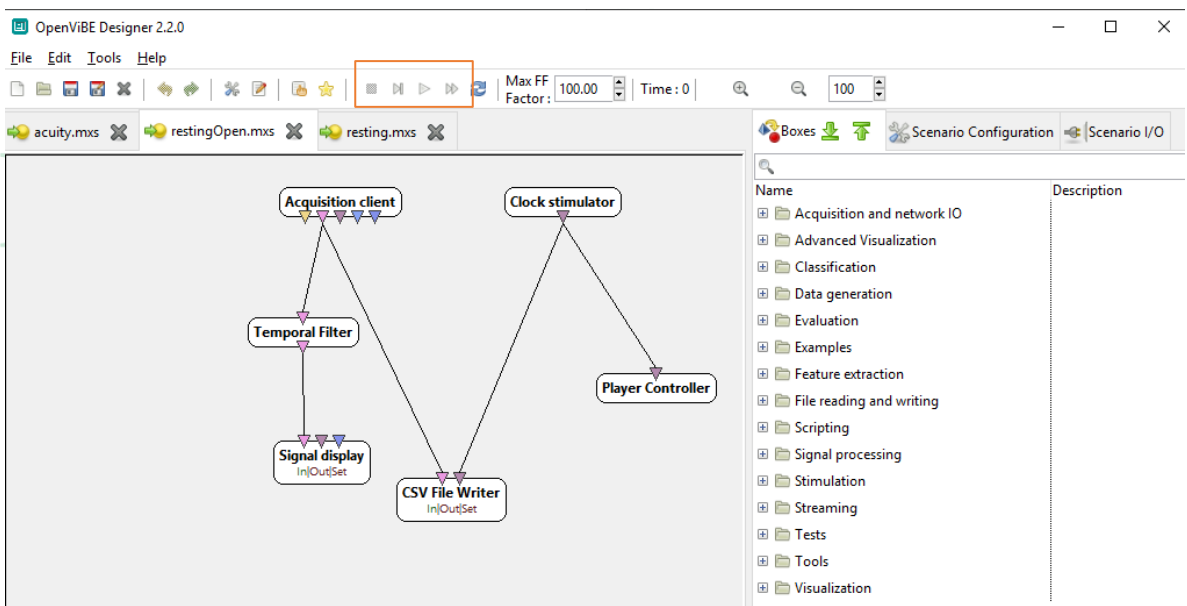


Figura 30. Adquisición.

Para comprender la corriente de deriva, sus variaciones y como afecta la adquisición puede dirigirse al adjunto “Protocolo de pruebas” o “OpenViBE ACQUISITION”.

Estimulación

- **OpenViBE:** Al iniciar la estimulación, se abrirá una ventana de pygame, envíe esta pantalla de manera manual a la pantalla Samsung SyncMaster 2243 LNX en el menor tiempo posible y presione la tecla espacio para iniciar el estímulo y observe en la pantalla del computador el OpenViBE Designer y el OpenViBE Acquisition Server, para observar la señal adquirida y el envío de los datos verificando que ambos se estén dando de manera adecuada.
- **Psychopy:** Abra el programa y dentro abra el archivo dot.psyexp en la carpeta de Psychopy del repositorio

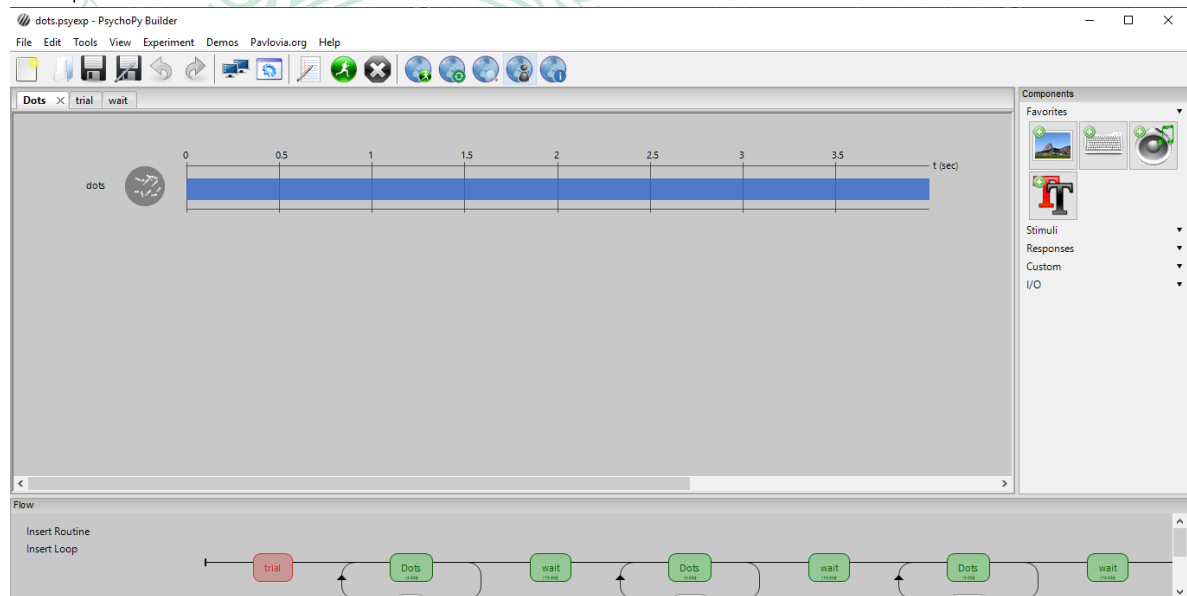


Figura 31 Estímulo en Psychopy.

En la parte inferior encontrara el flujo de trabajo del estímulo, y en la parte superior los controles para iniciar el estímulo.

- Presione el botón verde y después de unos segundos aparecerá un cuadro como el que se muestra a continuación, antes de presionar “OK” asegúrese de tener la segunda pantalla conectada (Todos los estímulos están configurados con la resolución de la SyncMaster, si lo corre en una pantalla con una resolución diferente puede ocasionar errores graves en el sistema).

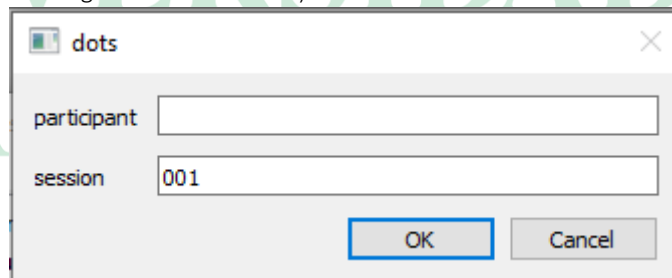


Figura 32. Participantes.

No es necesario completar los campos que se piden en el cuadro únicamente presione “OK”

- Le aparecerá una pantalla gris, el estímulo comenzara inmediatamente al presionar una tecla, antes de presionar comience la adquisición desde OpenViBE de la misma manera que lo hizo en los estímulos anteriores

- **Python:** Finalmente existen dos estímulos ejecutados desde Python, en un mismo Script “CompleteV5.py” abra este archivo en su editor Python y ejecútelo, igualmente, se ejecutará el estímulo inmediatamente después de presionar espacio cuando la pantalla de pygame se haya abierto Por lo tanto inicie la estimulación antes de presionar la tecla espacio.

NOTA: Todos los estímulos están configurados con la resolución de la SyncMaster, si lo corre en una pantalla con una resolución diferente puede ocasionar errores graves en el sistema, asegúrese de conectar una segunda pantalla y configurar las resoluciones en cada programa si es necesario. Adicionalmente se debe ser consciente que durante el procesamiento se eliminan los primeros 5 segundos de cada estimulación, dando tiempo al operario de pasar el estímulo a la pantalla de estimulación, en el caso de Psychopy se recomienda esperar 5 segundos de OpenViBE antes de iniciar el estímulo, ya que los tiempos en este se encuentran más limitados y se podría eliminar tiempo de estimulación si se inicia antes.

Entorno:

Sabiendo el funcionamiento de todos los estímulos se pasa a organizar el entorno. Los sujetos se registraron en una habitación oscura y tranquila, se sentaron con una posición recta y cómoda, sobre una silla estática que permite graduar la altura del sujeto para quedar frente a la pantalla, adicional, se fijó la mirada del sujeto con un soporte superior con apoyo mentoniano. La distancia de estimulación fue de 1 m para evaluación de visión intermedia.

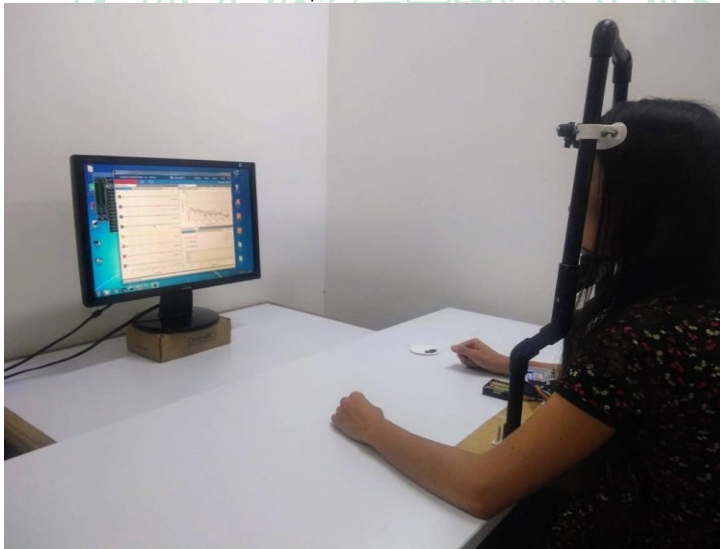


Figura 33. Entorno.

Se limita el ruido electromagnético que pueda provenir de los dispositivos electrónicos como celulares, alejándose del entorno de adquisición 2 metros.

Cuando la persona se encuentra en esta posición se procede a comenzar con la estimulación como se explico en el punto anterior.

● **PAGINAS DE INTERES**

- <https://openbci.com/>
- <https://docs.openbci.com/docs/01GettingStarted/GettingStartedLanding>
- <https://docs.openbci.com/docs/01GettingStarted/01-Boards/CytonGS>
- https://docs.openbci.com/docs/06Software/02-CompatibleThirdPartySoftware/OpenVibe?utm_source=google&utm_medium=cpc&utm_campaign=9572259726&utm_content=&gclid=CjwKCAjw2a32BRBXEiwAUcugiJw7bk58qx0lebl4IMok8TQUb6izUkfn4zJis_6ecue_xVOrq9Lm9RoCiPUQAvD_BwE

- <https://github.com/OpenBCI>
- <https://www.ftdichip.com/Drivers/VCP.htm>
- https://docs.openbci.com/docs/10Troubleshooting/OpenBCI_on_Windows

Anexo 2

PYGAME ESTIMULOS

SSVEP · Adquisición de señales EEG

Funciones · Clases · Métodos

Pygame agrega funcionalidad sobre la excelente biblioteca SDL; es una biblioteca C multiplataforma para controlar multimedia, comparable a DirectX. Pygame es altamente portátil y se ejecuta en casi todas las plataformas y sistemas operativos. Pygame es gratis. Lanzado bajo la licencia LGPL, puede crear código abierto, freeware, shareware y juegos comerciales con él. Vea la licencia para más detalles.

INICIACIÓN

PYGAME.INIT()

Inicializar todos los módulos de pygame importados. Siempre puede inicializar módulos individuales manualmente, pero `pygame.init()` inicializa todos los módulos pygame importados, es una forma conveniente de comenzar todo. Las `init()` funciones para módulos individuales generarán excepciones cuando fallen.

PYGAME.GET_INIT()

Devuelve `True` si pygame está actualmente inicializado

PYGAME.QUIT()

Desinicializar todos los módulos de Pygame. Cuando el intérprete de Python se apaga, este método se llama independientemente, por lo que su programa no debería necesitarlo, excepto cuando quiere terminar sus recursos de pygame y continuar. Es seguro llamar a esta función más de una vez ya que las llamadas repetidas no tienen efecto.

Nota Llamar a `pygame.quit()` no saldrá del programa. Se debe finalizar de la misma manera que finalizará un programa Python normal.

EVENTO

PYGAME.EVENT()

Módulo de pygame para interactuar los eventos como colas. Pygame maneja todos sus mensajes de eventos a través de una cola de eventos. Las rutinas en este módulo lo ayudan a administrar esa cola de eventos. La cola de entrada depende en gran medida del módulo `pygame.display()`.

La cola de eventos tiene un límite superior en la cantidad de eventos que puede contener (128 para SDL 1.2 estándar). Cuando la cola se llena, los nuevos eventos se descartan silenciosamente. Para evitar eventos perdidos, especialmente eventos de entrada que señalan un comando para salir, su programa debe verificar periódicamente los eventos y procesarlos. Para acelerar el procesamiento de la cola, use el `pygame.event.set_blocked()` control de los eventos permitidos en la cola para limitar los eventos.

Para obtener el estado de varios dispositivos de entrada, puede renunciar a la cola de eventos y acceder a los dispositivos de entrada directamente con sus módulos apropiados: `pygame.mouse` para trabajar con el mouse, `pygame.key` para trabajar con el teclado y `pygame.joystick` para interactuar con joysticks, gamepads y trackballs. Si utiliza este método, recuerde que pygame requiere alguna forma de comunicación con el administrador de ventanas del sistema y otras partes de la plataforma. Para mantener Pygame sincronizado con el sistema, deberá llamar a `pygame.event.pump()` los controladores de eventos de Pygame de proceso interno para mantener todo actualizado.

PYGAME.EVENT.PUMP()

Esto garantiza que su programa pueda interactuar internamente con el resto del sistema operativo. Manejo de eventos individuales.

PYGAME.EVENT.GET()

Esto obtendrá todos los mensajes y los eliminará de la cola. Si se proporciona un tipo o secuencia de tipos, solo esos mensajes se eliminarán de la cola.

PYGAME.EVENT.POLL()

Devuelve un solo evento de la cola. Si la cola de eventos está vacía `pygame.NOEVENT`, se devolverá un evento de tipo inmediatamente. El evento devuelto se elimina de la cola.

PYGAME.EVENT.WAIT()

Devuelve un solo evento de la cola. Si la cola está vacía, esta función esperará hasta que se cree una. El evento se elimina de la cola una vez que se ha devuelto. Mientras el programa está esperando, dormirá en un estado inactivo. Esto es importante para los programas que desean compartir el sistema con otras aplicaciones.

PYGAME.EVENT.PEEK()

Devuelve `True` si hay algún evento del tipo dado esperando en la cola. Si se pasa una secuencia de tipos de eventos, esto regresará `True` si alguno de esos eventos está en la cola.

PYGAME.EVENT.CLEAR()

Elimina todos los eventos de la cola.

DISPLAY

Este módulo ofrece control sobre la pantalla de pygame. Pygame tiene una superficie de visualización única que está contenida en una ventana o se ejecuta a pantalla completa. El origen de la pantalla, donde $x = 0$ e $y = 0$, es la esquina superior izquierda de la pantalla. Ambos ejes aumentan positivamente hacia la esquina inferior derecha de la pantalla.

PYGAME.DISPLAY.INIT()

Inicializa el módulo de visualización de pygame. El módulo de visualización no puede hacer nada hasta que se inicialice. Esto generalmente se maneja automáticamente cuando llama al nivel superior `pygame.init()`.

PYGAME.DISPLAY.QUIT()

Esto cerrará todo el módulo de pantalla. Esto significa que cualquier pantalla activa se cerrará. Esto también se manejará automáticamente cuando el programa salga.

PYGAME.DISPLAY.GET_INIT()

Devuelve True si el pygame está actualmente inicializado.

PYGAME.DISPLAY.SET_MODE()

Esta función creará una superficie de visualización. Los argumentos pasados son solicitudes de un tipo de visualización. La pantalla creada real será la mejor coincidencia posible admitida por el sistema.

PYGAME.DISPLAY.GET_SURFACE()

Devuelve una referencia a la superficie de visualización configurada actualmente. Si no se ha configurado el modo de visualización, esto devolverá "Ninguno".

PYGAME.DISPLAY.FLIP()

Actualice la pantalla completa Surface a la pantalla.

PYGAME.DISPLAY.UPDATE()

Esta función es como una versión optimizada de `pygame.display.flip()` para pantallas de software. Solo permite actualizar una parte de la pantalla, en lugar de toda el área. Si no se pasa ningún argumento, actualiza toda la superficie como `pygame.display.flip()`.

PYGAME.DISPLAY.SET_ICON()

Cambiar la imagen del sistema para la ventana de visualización. Establece el icono de tiempo de ejecución que el sistema usará para representar la ventana de visualización. Todas las ventanas tienen un logotipo de pygame simple para el icono de la ventana.

Puede pasar cualquier superficie, pero la mayoría de los sistemas quieren una imagen más pequeña alrededor de 32x32. La imagen puede tener una transparencia de colorkey que se pasará al sistema.

Algunos sistemas no permiten que el icono de la ventana cambie después de que se haya mostrado. Se puede llamar `pygame.display.set_mode()` a esta función antes para crear el icono antes de configurar el modo de visualización.

PYGAME.DISPLAY.SET_CAPTION()

Establecer el título de la ventana actual

Si la pantalla tiene un título de ventana, esta función cambiará el nombre en la ventana.

Algunos sistemas admiten un título alternativo más corto que se utilizará para pantallas minimizadas.

PYGAME.DISPLAY.GET_NUM_DISPLAYS()

Devuelve el número de pantallas de pygame

Devuelve el número de pantallas disponibles. Esto siempre es 1 si `pygame.get_sdl_version()` obtener el número de versión de SDL devuelve un número de versión principal por debajo de 2.

PYGAME.DISPLAY.GET_WINDOW_SIZE()

Devuelve el tamaño de la ventana o pantalla

Devuelve el tamaño de la ventana inicializada con `pygame.set_mode()`. Esto puede diferir del tamaño de la superficie de la pantalla si SCALED se usa.

Nuevo en pygame 2.0.

IMAGE

El módulo de imagen contiene funciones para cargar y guardar imágenes, así como para transferir Superficies a formatos utilizables por otros paquetes.

Tenga en cuenta que no hay clase de imagen; una imagen se carga como un objeto Surface. La clase Surface permite la manipulación (dibujar líneas, establecer píxeles, capturar regiones, etc.).

PYGAME.IMAGE.LOAD()

Cargue una imagen de una fuente de archivo. Puede pasar un nombre de archivo o un objeto similar a un archivo Python.

PYGAME.IMAGE.SAVE()

Guardar una imagen en el disco. Pygame 1.8: guardar archivos PNG y JPEG.

KEY

Este módulo contiene funciones para tratar con el teclado.

El `pygame.event` es un módulo de pygame para interactuar con eventos y colas obtiene `pygame.KEYDOWN` y `pygame.KEYUP` eventos cuando se presionan y sueltan los botones del teclado. Ambos eventos tienen `key` y `mod` atributos.

key: una ID entera que representa cada tecla del teclado

mod: una máscara de bits de todas las teclas modificadoras que estaban presionadas cuando ocurrió el evento

PYGAME.KEY.GET_PRESSED ()

Obtener el estado de todos los botones del teclado

```
for e in pygame.event.get():
    if e.type == pygame.QUIT: break
    if e.type == pygame.KEYDOWN:
        if e.key == pygame.K_p: state = PAUSE
        if e.key == pygame.K_s: state = RUNNING
        if e.key == pygame.K_ESCAPE:
            pygame.quit()
```

SURFACE

Una superficie de pygame se usa para representar cualquier imagen. La superficie tiene una resolución fija y formato de píxel. Las superficies con píxeles de 8 bits utilizan una paleta de colores para asignar a color de 24 bits.

PYGAME. SURFACE. BLIT ()

Dibujar una imagen sobre otra

PYGAME. SURFACE. BLITS ()

Dibujar muchas imágenes en otra

PYGAME. SURFACE. CONVERT ()

Crea una nueva copia de Surface con el formato de píxel cambiado.

PYGAME. SURFACE. COPY ()

Hace una copia duplicada de una superficie. La nueva superficie tendrá los mismos formatos de píxeles, paletas de colores, configuraciones de transparencia y clase que la original.

PYGAME. SURFACE. FILL ()

Rellene la superficie con un color sólido. Si no se da un argumento correcto, se rellenará toda la superficie.

PYGAME. SURFACE. SCROLL ()

Desplaza la imagen de la superficie en su lugar. desplazamiento (dx = 0, dy = 0) -> Ninguno
Mueva la imagen dx píxeles hacia la derecha y dy píxeles hacia abajo. dx y dy pueden ser negativos para los desplazamientos hacia la izquierda y hacia arriba, respectivamente. Las áreas de la superficie que no se sobrescriben conservan sus valores de píxeles originales. El desplazamiento está contenido en el área de recorte de superficie. Es seguro tener valores dx y dy que excedan el tamaño de la superficie.

FONT

El módulo de fuente permite representar fuentes TrueType en un nuevo objeto Surface. Puede cargar fuentes desde el sistema utilizando la `pygame.font.SysFont()` función. Hay algunas otras funciones para ayudar a buscar las fuentes del sistema.

PYGAME. FONT.SYSFONT()

Devuelve un nuevo objeto Font que se carga desde las fuentes del sistema. La fuente coincidirá con los indicadores solicitados en negrita y cursiva. Si no se encuentra una fuente de sistema adecuada, se volverá a cargar la fuente de pygame predeterminada. El nombre de la fuente puede ser una lista separada por comas de nombres de fuentes para buscar.

PYGAME.FONT.GET_FONTS ()

Devuelve una lista de todas las fuentes disponibles en el sistema. Los nombres de las fuentes se establecerán en minúsculas con todos los espacios y signos de puntuación eliminados. Esto funciona en la mayoría de los sistemas, pero algunos devolverán una lista vacía si no pueden encontrar las fuentes.

PYGAME.FONT.MATCH_FONT ()

Devuelve la ruta completa a un archivo de fuente en el sistema. Si negrita o cursiva se establecen en verdadero, esto intentará encontrar la familia correcta de fuentes.

PYGAME. FONT.FONT.SIZE ()

Devuelve las dimensiones necesarias para representar el texto. Esto se puede usar para ayudar a determinar el posicionamiento necesario para el texto antes de que se procese. También se puede usar para envolver palabras y otros efectos de diseño.

PYGAME. FONT.FONT.RENDER ()

Rellene la superficie con un color sólido. Si no se da un argumento correcto, se rellenará toda la superficie.

EJEMPLOS

import pygame

pygame.init()

screen = pygame.display.set_mode((400, 300))

done = False

while not done:

for event in pygame.event.get():

if event.type == pygame.quit():

done = True

import pygame - necesario para acceder al marco de PyGame.

pygame.init() - Inicializa todos los módulos necesarios para PyGame.

pygame.display.set_mode((width, height)) - Esto abrirá una ventana del tamaño

`pygame.display.flip()`

```
import pygame

pygame.init()
screen = pygame.display.set_mode((400, 300))
done = False
is_blue = True
x = 30
y = 30

clock = pygame.time.Clock()

while not done:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            done = True
        if event.type == pygame.KEYDOWN and event.key == pygame.K_SPACE:
            is_blue = not is_blue

    pressed = pygame.key.get_pressed()
    if pressed[pygame.K_UP]: y -= 3
    if pressed[pygame.K_DOWN]: y += 3
    if pressed[pygame.K_LEFT]: x -= 3
    if pressed[pygame.K_RIGHT]: x += 3

    screen.fill((0, 0, 0))
    if is_blue: color = (0, 128, 255)
    else: color = (255, 100, 0)
    pygame.draw.rect(screen, color, pygame.Rect(x, y, 60, 60))

    pygame.display.flip()
    clock.tick(60)
```

deseado. El valor de retorno es un objeto Surface, que es el objeto sobre el que realizará operaciones gráficas.

pygame.event.get() - Esto vacía la cola del evento. Si no llama a esto, los mensajes de Windows comenzarán a acumularse y su juego dejará de responder en opinión del sistema operativo.

pygame.quit() - Este es el tipo de evento que se activa cuando cerrar la ventana.

pygame.display.flip() - Las actualizaciones en la pantalla sean visibles.



DAD
QUIA

1 8 0 3

Anexo 3

Manual de usuario de herramienta
para la evaluación de la fisiología visual.



En el marco del proyecto:
Desarrollo de una herramienta para la evaluación de la fisiología visual usando
electroencefalografía portable y de bajo costo

UNIVERSIDAD
DE ANTIOQUIA

Proyecto de Investigación
Pregrado Bioingeniería
Verónica Henao Isaza

1 8 0 3

Universidad de Antioquia
Facultad de ingeniería
Programa de Bioingeniería
Medellín, Antioquia

Contenido

Introducción	59
Glosario.....	59
Bibliotecas y dependencias.....	59
instalaciones previas.....	59
bibliotecas utilizadas.....	59
Componentes	60
Modelo	61
Controlador.....	63
Vista.....	64
Estimulación agudeza de vernier	72
Procesamiento con multitapering	76
Procesamiento con tiempo frecuencia.....	77
Los resultados entregados por la herramienta, incluyendo los generados en: <i>stimulation_acuity.py</i> , <i>dataprocessing.py</i> y <i>plot_stft.py</i>	78
Servidor.....	80
Simulación	82

UNIVERSIDAD
DE ANTIOQUIA

1 8 0 3

INTRODUCCIÓN

El objetivo principal de este proyecto es desarrollar una herramienta para la evaluación de la fisiología visual mediante el análisis de señales electroencefalográficas obtenidas con un dispositivo portable y de bajo costo, con el fin de lograr registrar de manera efectiva los potenciales visuales en estado estacionario con una herramienta portable, una interfaz amigable, con un protocolo de montaje sencillo, eficiente y asequible, favoreciendo de gran manera la ejecución de pruebas en diferentes entornos, además, el desarrollo y aplicación de medidas de procesamiento cada vez más especializadas pueden permitir el diagnóstico de patologías y la rehabilitación de pacientes con deficiencias visuales partiendo desde los parámetros más simples y que entregan más información para este tipo de sujetos.

GLOSARIO

- Usuario: Persona que manipula la herramienta, puede ser o no la misma persona que le realiza el montaje al sujeto.
- Sujeto: Persona que es registrada por uno o más usuarios, puede ser un paciente o una persona sin ninguna patología asociada. Será a quien se le realice el montaje de los electrodos y se le muestre la estimulación visual.

BIBLIOTECAS Y DEPENDENCIAS

La herramienta fue desarrollada en python 3.7 con el IDE Spyder de Anaconda en Windows 10.

Instalaciones previas

Python, para evitar mayor cantidad de instalaciones se recomienda instalar Anaconda

- pygame -> conda install -c cogsci pygame
- wmi -> conda install -c primer wmi
- pymongo -> conda install -c anaconda pymongo
- serial -> pip install serial
- pyserial -> conda install -c anaconda pyserial
- pylsl -> pip install pylsl
- pyOpenBCI -> pip install pyOpenBCI
- pip install bitstring
- pip install xmldict

Bibliotecas utilizadas

- PyQt5
- Matplotlib
- scipy
- numpy
- pandas
- wmi
- os
- pygame

- time
- traceback
- sys
- pylsl
- datetime
- csv
- errno
- serial
- subprocess
- pymongo
- spectrum
- pyOpenBCI

COMPONENTES

ARCHIVOS .PY	Definición
model.py	Modelo
controller.py	Controlador
View.py	Vista
Stimulation_Acuity.py	Estimulación agudeza de Vernier
dataprocessing.py	Procesamiento multitaper
Plot_stft.py	Procesamiento tiempo-frecuencia
randData.py	Simulador de datos
Server.py	Conexión con OpenBCI
linearFIR.py	Filtro FIR
nonlinear.py	Filtro no lineal

ARCHIVOS .UI	Definición
ViAT	Inicio
Agregardatos	Permite agregar o elegir un sujeto de la base de datos
Adquisicion	Permite realizar la conexión con el openBCI y medir la impedancia

ARCHIVOS .UI	Definición
Adquisicion_accion	Permite visualizar los datos adquiridos en tiempo real
Buscardatos	Permite buscar y observar los datos en la base de datos
Visualizacion	Permite visualizar señales guardadas previamente

MODELO

Contiene los datos y la funcionalidad de la aplicación, es decir, se encarga de realizar las funciones de actualizar, búsqueda, consulta, procesamiento de datos, etc.
model.py

Funciones

1. **init**
Recibe: Nombre de la base de datos, nombre de la colección de la base de datos, datos para grafica.
Función: Realiza la conexión con la base de datos de mongo, diseña los filtros llamando a *filtDesign()* y define las ubicaciones de almacenamiento de los archivos.
2. **newLocation**
Recibe: La nueva ubicación
Función: Seleccionar la ubicación de los nuevos registros adquirir
3. **location**
Recibe:
Función: Redefine la ubicación de los nuevos registros adquirir
4. **startDevice**
Recibe:
Función: Inicia el envío de los datos, utiliza la función *subprocess.Popen* para iniciar un proceso nuevo sin detener el existente.
5. **stopDevice**
Recibe:
Función: Detener el envío de los datos y cerrar el proceso.
6. **startData**
Recibe:
Función: Define una matriz de ceros para los datos con la cantidad de canales y la cantidad de muestras.
 Utiliza la función *resolve_stream* de la biblioteca *pysl* para recibir los datos enviados por el servidor *Server.py* o el simulador de datos *randData.py*.
 Utilizar el objeto *StreamInlet* para recibir datos de transmisión (y metadatos) desde la red del laboratorio, toma las trasmisiones disponibles y finalmente se hace un *pull_chunk()* que extrae un trozo de muestras de la entrada.
7. **stopData**
Recibe:

Función: Detiene la acción de recibir los datos cerrando el stream con *close_stream()* y llama los módulos de procesamiento *dataprocessing.py* y *plot_stft.py*

8. startStimulus

Recibe:

Función: Llama el módulo de estimulación *Stimulation_Acuity.py* y lo inicia.

9. stopStimulus

Recibe:

Función: Detiene la estimulación al cerrar el *pygame*.

10. startZ

Recibe:

Función: Inicia la recepción de datos con *StreamInlet* para posteriormente hallar la impedancia de los datos.

11. stopZ

Recibe:

Función: Cierra la recepción de los datos con *close_stream* al finalizar la lectura de la impedancia.

12. readZ

Recibe:

Función: Hace un *pull_sample()* para tomar un valor y realizar la operación por ley de ohm. V = voltaje rms recibido. i = corriente del dispositivo = 6 nA

$$Z = \frac{V * \sqrt{2}}{i}$$

El valor de Z es dividido por 1000 antes de presentarse en la vista.

13. readData

Recibe:

Función: Hace un *pull_chunk()* para tomar una cantidad de valores y realizar una diferencia entre el canal de referencia y cada uno de los canales de interés. Crea un Dataframe de cada resultado para almacenarlo en un archivo .csv

14. filtDesign

Recibe:

Función: Diseña un filtro pasa-bajas y un filtro pasa-altas

15. filtData

Recibe:

Función: Llama la función *readData()* y utiliza la función *filtfit* de *scipy.signal* para aplicar un filtro a los datos recibidos en tiempo real.

16. Pot

Recibe:

Función: Aplica el método de Welch al canal o configuración definida por el usuario en la interfaz

17. laplace

Recibe: Del menú desplegable de la interfaz gráfica, toma los valores de los canales de interés.

Función: Crea una matriz para realizar la operación de Laplace con los valores entregados.

18. returnLastData

Recibe:

Función: Ejecuta *Pot()* y retorna los valores a la vista para graficarlos.

19. **returnLastZ**

Recibe:

Función: Ejecuta *readZ()* y retorna los valores de la impedancia.

20. **returnLastStimulus**

Recibe:

Función: Ejecuta *readData()*

21. **add_into_collection_one**

Recibe:

Función: Agrega un sujeto a la base de datos

22. **search_one**

Recibe:

Función: Busca una persona de la base de datos

23. **search_many**

Recibe:

Función: Entrega la lista de integrantes de la base de datos

24. **delete_data**

Recibe:

Función: Elimina un sujeto de la base de datos

25. **assign_data**

Recibe:

Función: Entrega los datos a graficar de un archivo .csv

26. **return_segment**

Recibe:

Función: Permitir el avance en el tiempo de la señal

27. **signal_scale**

Recibe:

Función: Permitir realizar la ampliación o disminución de la señal

28. **file_location**

Recibe:

Función: Permite encontrar un archivo de un sujeto determinado.

CONTROLADOR

controller.py

Determina que procesos debe realizar el modelo cuando el usuario interacciona con el sistema, para luego comunicarle a la vista los resultados. Simplemente toma una orden de la vista y se la envía al controlador, por esto las funciones en este modulo hacen referencia a las mismas funciones que hay en el modelo.

El controlador esta compuesto por 3 clases:

- Principal: Inicia el servidor de la base de datos MongoDB en un subproceso de la aplicación, inicia la aplicación y define las variables de la vista y el modelo. Ingresa la vista y el modelo a cada uno de los controladores y asigna los controladores para utilizarlos en la vista.
- Controller: Maneja todas las funciones relacionadas con los datos recibidos del dispositivo y las señales graficadas en la interfaz.

- **Controlador:** Maneja todas las funciones relacionadas con la base de datos de los sujetos y la interacción con MongoDB.

VISTA

view.py

Gestiona como se muestran los datos en la interfaz gráfica por medio de las ordenes que le envía al controlador y los datos que este le devuelve de las operaciones realizadas en el modelo.

Está compuesto 6 vistas diseñadas en QtDesigner y por 8 clases que controlan cada una de las vistas y dos adicionales para el control de hilos para los procesos que se llevan a cabo en simultaneo

Clases:

- **WorkerSignals:** Define las variables que controlan los procesos como:
 - Finalizar, cuando el proceso llega a su fin
 - Error, cuando se presentan tareas combinadas
 - Resultados, los datos esperados en cada proceso
 - Progreso, el seguimiento de proceso.
- **Worker:** Realiza la ejecución de los hilos
- **ViAT (Vista 1):** Ejecuta la vista principal **ViAT.ui** (Figura 34) con el menú para desplazarse por la aplicación. Contiene una breve descripción de la aplicación y cuenta con 3 botones:
 1. Iniciar registro: Permite dirigirse a la vista 2.
 2. Base de datos pacientes: Permite dirigirse a la vista 6
 3. Salir: Permite salir de la aplicación.

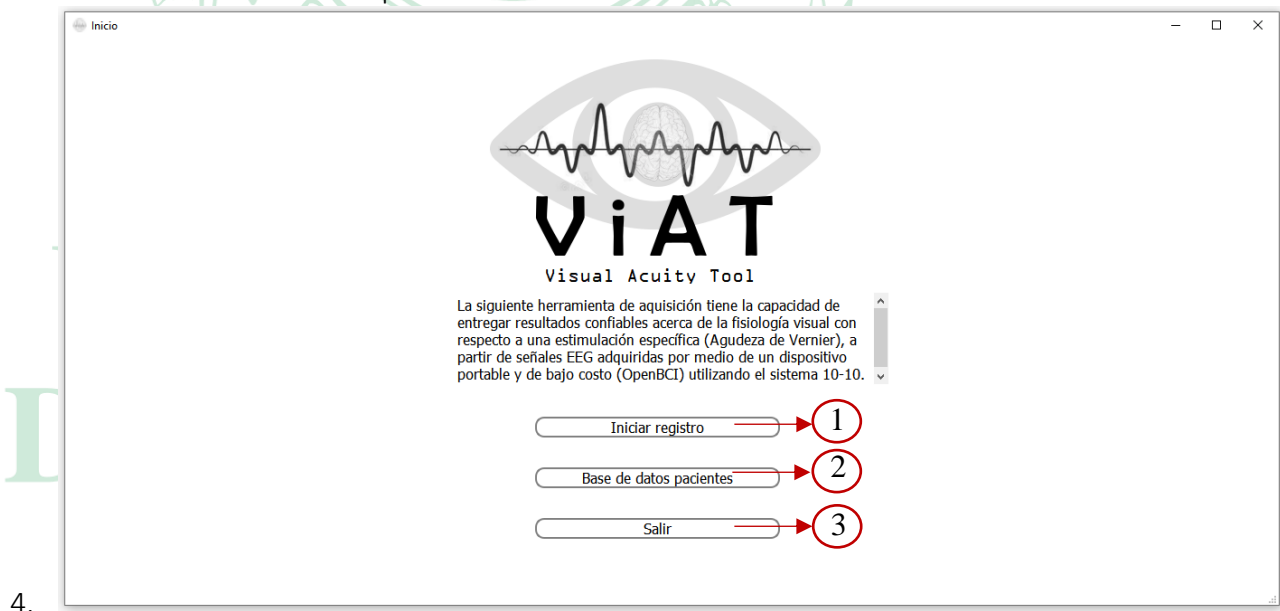


Figura 34. ViAT.ui

LoadRegistration (Vista 2): Ejecuta la vista **Agregardatos.ui** (Figura 35). Tiene la función de crear y administrar una base de datos de cada sujeto registrado con la herramienta ViAT.

1. Campos de libre escritura: Son campos de los datos comunes como: Nombre, Apellido, Cédula (CC), Agudeza de Snellen (Valor que debe encontrarse en la escala de Snellen es decir desde 20/16 hasta 20/200, se debe evaluar sin gafas), Agudeza de Snellen corregida (hace referencia a la agudeza de un sujeto al usar gafas), Edad, ¿Tiempo desde el accidente visual? (Es un dato tomado para sujetos que hayan perdido funciones visuales debido a un accidente cráneo encefálico) y responsable (Persona que realiza el registro de un sujeto, es decir el usuario actual de la aplicación)
2. Botón de verificación: Permite al usuario reducir tiempos de registro al autocompletar los campos de un sujeto que se encuentre previamente registrado en la base de datos (Figura 36).
3. Botones de menú desplegables: Permiten al usuario responder de manera concisa preguntas cortas como: Sexo (Femenino o masculino), Ojo dominante (Derecho o izquierdo), ¿Usa gafas? (Sí o No), Estímulo (Vernier) por el momento es el único estímulo implementado en la herramienta.
4. Permite al usuario tener la autonomía de elegir el lugar de ubicación de los archivos resultantes de la estimulación (Registro, marcas, procesamiento con multitaper y procesamiento con tiempo frecuencia)
5. Permite al usuario desplazarse a la base de datos para buscar un sujeto vista 5
6. Permite agregar un sujeto, luego de completar todos los campos.
7. Al verificar un sujeto de la base de datos, los campos de la columna 2 (Figura 36) permiten ser manipulados y actualizados con el botón "Actualizar" se creará un nuevo registro de la base de datos con los mismos datos de la columna 1 y los nuevos de la columna 2, en la base de datos se conservan ambos registros como historia clínica, si se desea eliminar alguno de los registros se debe ir a la base de datos y confirmar esta acción.
8. Permite ir a la siguiente vista. (Vista 3)
9. Es el botón de ayuda, contiene el siguiente mensaje como instrucción de uso de la vista: "Diríjase al campo de CC y digite la cédula del sujeto a registrar, para verificar que no se encuentre ya en la base de datos, si no está llene todos los campos presentados a continuación y presione agregar, en el botón 'definir ubicación' podrá modificar la ubicación donde desea que quede guardado el registro, al finalizar presione el botón 'Siguiente'. Si el sujeto ya se encuentra en la base de datos y desea modificar un campo, presione 'Actualizar'. Si desea observar todos los sujetos en la base de datos presione 'Mostrar Pacientes'. Para volver al menú anterior presione 'Atrás'."

UNIVERSIDAD
DE ANTIOQUIA

1 8 0 3

Base de datos

HISTORIA CLINICA

VIAT

ID:

Nombre:

Apellido:

CC:

Sexo:

Ojo dominante:

¿Usa gafas?

Agudeza de Snellen:

Agudeza de Snellen corregida:

Estímulo:

Edad:

¿Tiempo desde el accidente visual?

Responsable:

Definir ubicación

Agregar Actualizar Mostrar Pacientes

Atrás Siguiente

Figura 35. Agregar datos.ui

Base de datos

HISTORIA CLINICA

VIAT

ID:

Nombre:

Apellido:

CC:

Sexo:

Ojo dominante:

¿Usa gafas?

Agudeza de Snellen:

Agudeza de Snellen corregida:

Estímulo:

Edad:

¿Tiempo desde el accidente visual?

Responsable:

Definir ubicación

Agregar Actualizar Mostrar Pacientes

Atrás Siguiente

Figura 36. Agregar datos.ui en uso

DataAcquisition (Vista 3): Ejecuta la vista **Adquisicion.ui** (Figura 37). Tiene la función de preparar el registro, permitiendo al usuario observar el montaje de electrodos sugerido (10-10), medir la impedancia de cada electrodo posicionado, conectar el dispositivo y verificar la segunda pantalla en la que se presentará el estímulo.

1. Montaje de electrodos corteza occipital (10-10): El sistema internacional de colocación de EEG toma cuatro puntos de referencia craneales universales (nasion, inion y ambos puntos pre-auricular), y distribuye proporcionalmente los electrodos del EEG sobre la superficie de la cabeza. Según el porcentaje de la distancia entre los sensores, tenemos el layout 10-20 con un total de 21 sensores, si partimos de una distribución de distancias del 10% y el 20% de las curvas de referencia central sagital y coronal. Si estas líneas centrales se dividen en un 10%, entonces tenemos el layout 10-10 con 81 sensores. Tener un menor porcentaje de distancia nos permite concentrar los 8 electrodos en la zona de interés en este

caso, visión. Este esquema permite también presionar los botones de los electrodos para comenzar la medición de la impedancia.

2. Conectar dispositivo: Activa el envío de datos del dispositivo a la “red del laboratorio” en donde se tramite la información de cada electrodo. Este es el primer botón que se debe presionar al interactuar con la vista.
3. Verificar pantalla: Se activa inmediatamente después de presionar “Conectar dispositivo”, el estímulo se debe presentar en una pantalla aparte para poner tener control de la interfaz y observar la señal adquirida durante la estimulación. Al conectar una segunda pantalla por medio de un cable HDMI el software detectara la entrada y permitirá continuar con las demás tareas, si no se tiene una segunda pantalla, el software entregara un mensaje de advertencia solicitando dicha acción, no es posible continuar sin una segunda pantalla conectada. Esta verificación se hace por medio de la biblioteca wmi:
obj = wmi.WMI().Win32_PnPEntity(ConfigManagerErrorCode=0)
displays = [x for x in obj if 'DISPLAY' in str(x)]
4. Al verificar la segunda pantalla y con el dispositivo conectado, se pasa a observar la impedancia, que como se menciona anteriormente, se debe presionar alguno de los botones del montaje para activar la recepción de los valores de impedancia. La impedancia permite evaluar el adecuado montaje de los electrodos, ya que, a menor impedancia, menor será la resistencia hecha por el cuero cabelludo al paso de la señal para ser recibida por el electrodo. Se recomienda que la impedancia sea menor a 30kohms y de debe acomodar el electrodo, agregar mayor cantidad de pasta o gel conductores para disminuir la impedancia.
5. Antes de continuar se debe detener la medición de la impedancia para que la adquisición de los datos no entre en conflicto con los procesos siguientes.
6. Permite volver a la vista anterior Vista 2.
7. Permite continuar a la siguiente Vista 4.
8. Es el botón de ayuda, contiene el siguiente mensaje como instrucción de uso de la vista: Comience identificando que el dispositivo está conectado y que puede comenzar a adquirir los datos, se abrirá una ventana negra la cual le anuncia que se ha comenzado la adquisición, presione 'Minimizar' y a continuación verifique que la pantalla de estimulación se encuentra conectada. Para verificar la impedancia presione cualquiera de los electrodos de la configuración. Antes de pasar a la siguiente etapa recuerde detener la medición de la impedancia.

UNIVERSIDAD
DE ANTIOQUIA

1 8 0 3

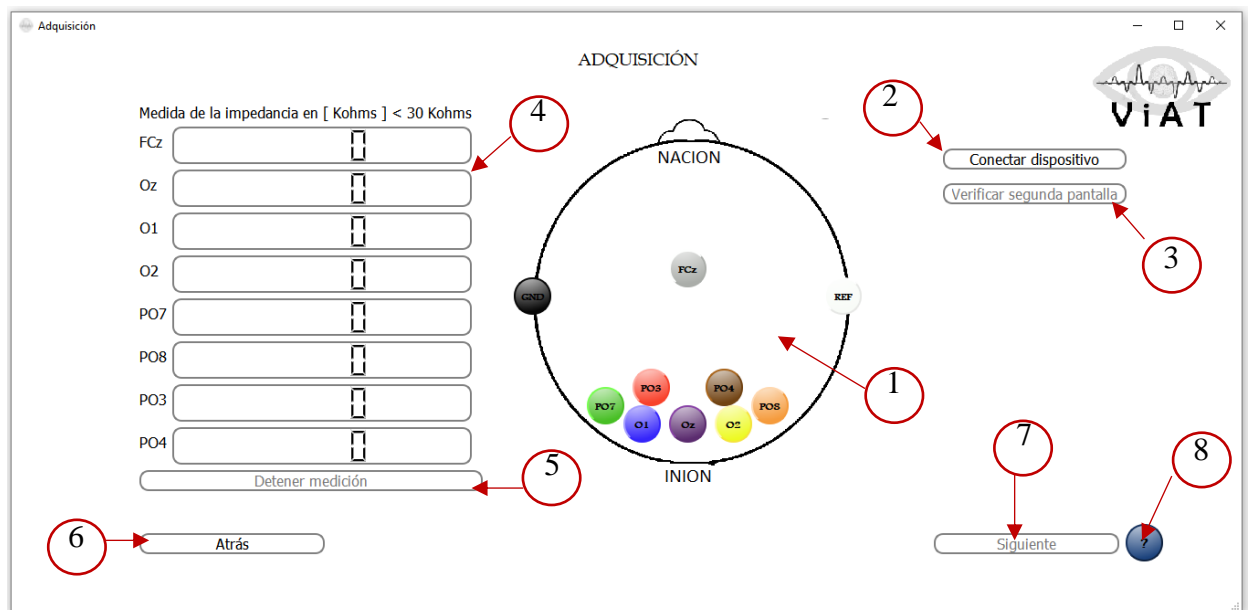


Figura 37. Adquisicion.ui

AcquisitionSignal (Vista 4): Ejecuta la vista **Adquisicion_accion.ui** (Figura 38). Tiene la función de visualizar la adquisición de los datos:

1. Botón de iniciar: Activa una línea de carga que le da tiempo al usuario de comenzar la visualización (2) en la pestaña "Visualización" antes de iniciar el estímulo.
2. Permite iniciar la visualización, toma un tiempo aproximado de 15 segundos comenzar a visualizar las señales, este tiempo está considerado en la línea de tiempo llamada "Cargando estimulación" al presionar el botón iniciar.
3. Reiniciar permite comenzar con la carga del estímulo, se debe parar la visualización antes de iniciar de nuevo el estímulo.
4. El método de Welch: Se utiliza para estimar la potencia de una señal en diferentes frecuencias: es decir, que es un enfoque de estimación de la densidad espectral. En el menú aparece cada uno de los canales de la adquisición, al seleccionar un canal se le aplica el método de Welch y se observa el resultado en el color del canal seleccionado.
5. Laplace: El objetivo de esta operación es eliminar el ruido debido al efecto de resistencia ocasionado por el cráneo, actúa como un filtro de pasa alta. Además, eliminan el efecto del electrodo de referencia. Esta grafica permite observar el comportamiento de la señal con la técnica de Laplace, como método adicional de verificación de una correcta adquisición bajo el estímulo.
6. Se puede detener la estimulación sin desconectar el dispositivo, permitiendo realizar registros continuos durante la ejecución de la aplicación.
7. Desconectar dispositivo, el dispositivo deja de enviar datos y se hace necesario volver a la vista anterior para iniciarlo nuevamente. Esta acción es necesaria para pasar a la base de datos o visualizar las señales adquiridas previamente.
8. Permite volver a la vista anterior, para observar las impedancias o conectar el dispositivo.
9. Cierra el programa.
10. Permite ir a la vista 6 y visualizar las señales de la base de datos.
11. Botón de ayuda contiene el siguiente mensaje: En la parte superior encontrará dos pestañas, *cargar estimulación*, en la que se encuentra actualmente y la de *visualización*, en la cual podrá observar la señal adquirida en tiempo real. Primero

debe presionar el botón inicio en la pestaña actual, luego, pase a la pestaña de visualización y presione 'visualizar', en la primera pestaña podrá observar una barra de proceso que le anunciara que el estímulo está a punto de presentarse. En esta pestaña podrá interactuar con la señal aplicando el método de Welch a cada uno de los canales o modificando la configuración de Laplace según el interés del registro. Podrá reiniciar el estímulo las veces que sea necesario sin necesidad de detener la adquisición. En la parte inferior podrá observar 4 botones que le permiten ir a la vista anterior, desconectar el dispositivo de adquisición, visualizar las señales adquiridas previamente o salir del programa.

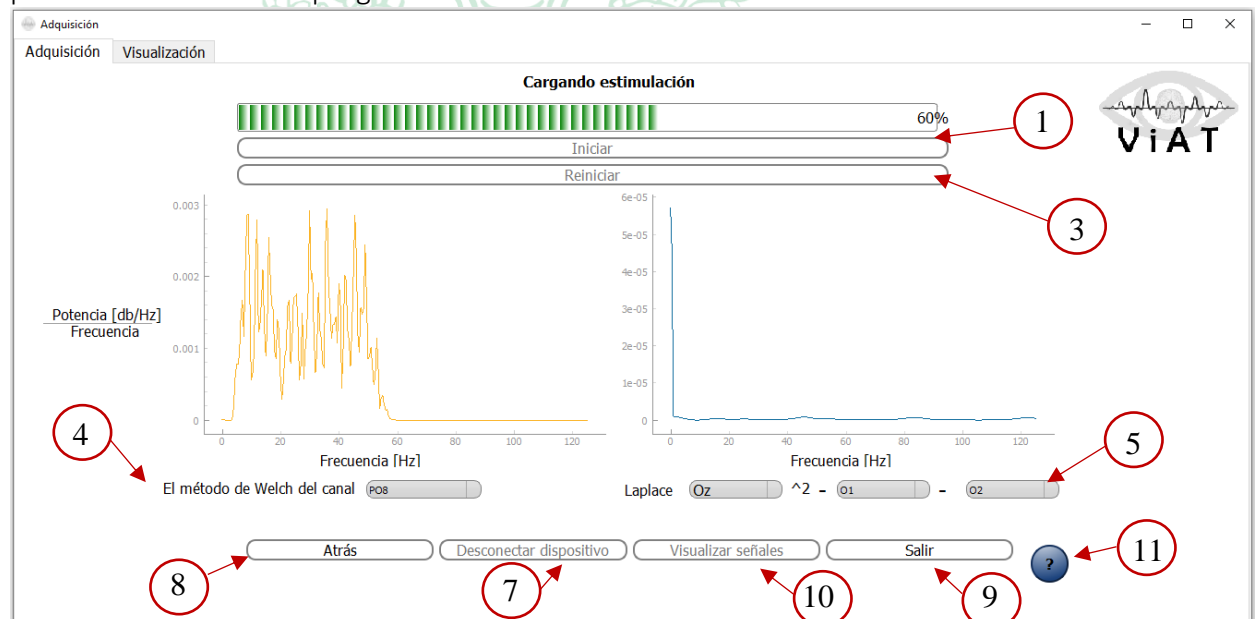


Figura 38. Control vista estímulo.

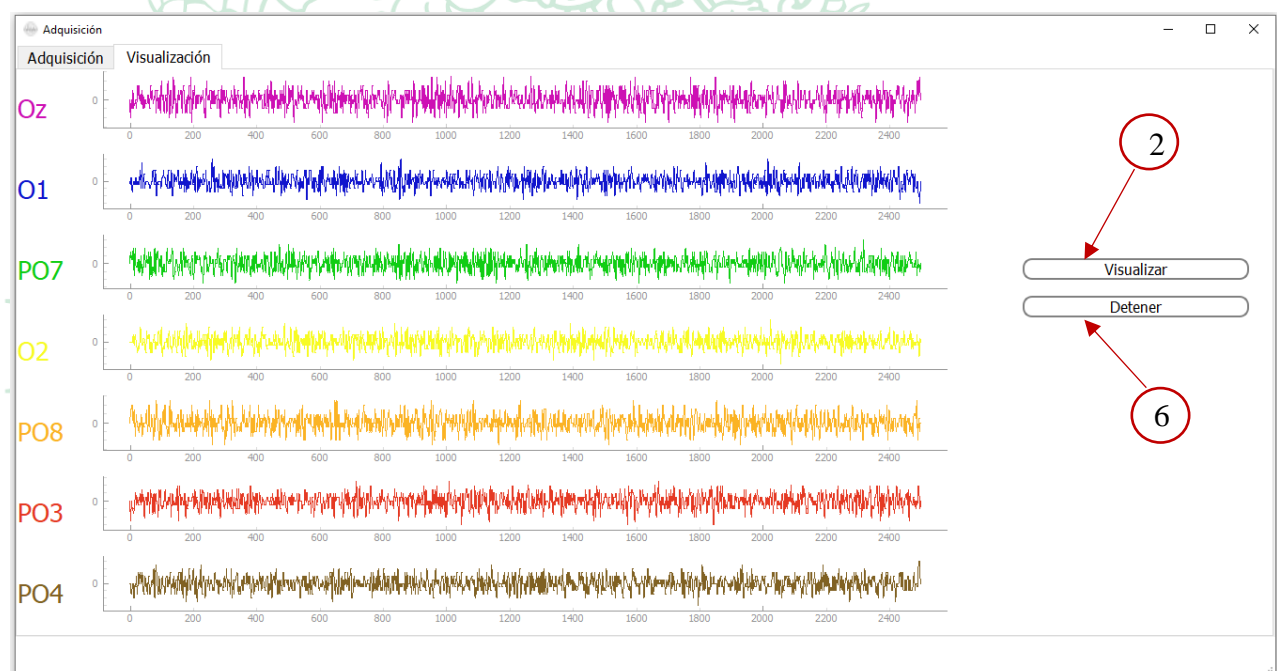


Figura 39. Visualización de señal

DataBase (Vista 5): Ejecuta la vista **Buscardatos.ui** (Figura 40). Tiene la función de visualizar los sujetos registrados en la base de datos:

1. Permite buscar un sujeto en la base de datos por medio del número de Cédula registrado.
2. Es un Menú que permite elegir la función que tendrá al presionar doble clic sobre un sujeto en la base de datos, permitiendo eliminar un registro o direccionar a los archivos registrados localmente del sujeto seleccionado.
3. Este botón permite mostrar todos los sujetos en la base de datos local creada en MongoDB al ejecutar el programa.
4. Permite direccionar a la vista 6, donde se observan señales previamente registradas.
5. Botón de ayuda contiene el siguiente mensaje: En el primer campo que encuentra podrá digitar la CC de un sujeto para buscar su información en la base de datos, o presionar 'Mostrar todo' para visualizar todos los sujetos presentes en la base de datos. Existe un menú desplegable debajo de este campo, este permite eliminar un sujeto de la base de datos o direccionar al usuario al registro del sujeto. En el botón 'Visualizar señales' se dirigirá a la vista que le permite buscar y visualizar señales previamente adquiridas. Para volver al menú anterior presione 'Atrás'.
6. Atrás, este botón permite volver a la vista anterior, esta puede ser; Vista 1 o 2. Según el uso dado por el usuario.

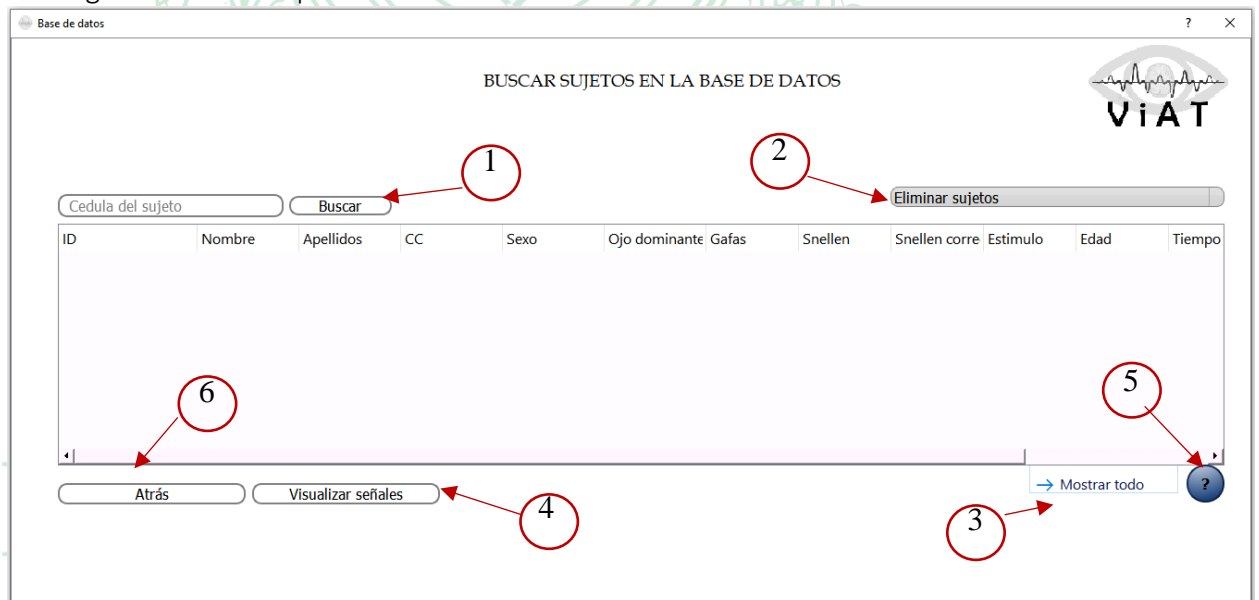


Figura 40. Base de datos

GraphicalInterface (Vista 6): Ejecuta la vista **visualizacion.ui** (Figura 41). Tiene la función de visualizar las señales registradas con el software ViAT o el software OpenViBE:

1. Menú. Las señales adquiridas con OpenViBE tienen una estructura de almacenamiento diferente a la adquirida por ViAT por esta razón es necesario que el usuario especifique de que tipo es la señal antes de cargarla.
2. Cargar señal, abre el explorador de archivos en la carpeta de la aplicación y permite buscar el registro que se quiera observar. Existe una carpeta llamada "Record" en la carpeta principal, ahí deberá encontrar todos los registros realizados por ViAT a no ser que hace cambiado la ubicación de un registro en la Vista 2.
3. Son botones para interactuar con la señal visualizada. Permiten mover hacia la derecha o izquierda la seña y hacer un pequeño zoom de la misma.
4. Permite buscar un sujeto en la base de datos y al aparecer en la tabla y presionar doble clic el usuario se dirigirá a la carpeta de registros del sujeto.
5. Botón de ayuda contiene el siguiente mensaje: En esta vista podrá visualizar señales previamente registradas por ViAT o en el software OpenViBE, en el menú desplegable a la derecha elija el tipo de archivo que va a buscar y presione 'Cargar señal'. Los botones en la parte inferior de este botón le permiten desplazar la señal en el tiempo y realizar un ajuste de la señal para visualizarla mejor. En el campo siguiente podrá interactuar nuevamente con la base de datos buscando un sujeto con su CC para visualizar la señal de manera más ágil. Si desea observar todos los sujetos en la base de datos presione 'Mostrar Pacientes'. Para volver al menú anterior presione 'Atrás'.
6. Atrás, este botón permite volver a la vista anterior, esta puede ser; Vista 4 o 5. Según el uso dado por el usuario.
7. Cierra el programa.

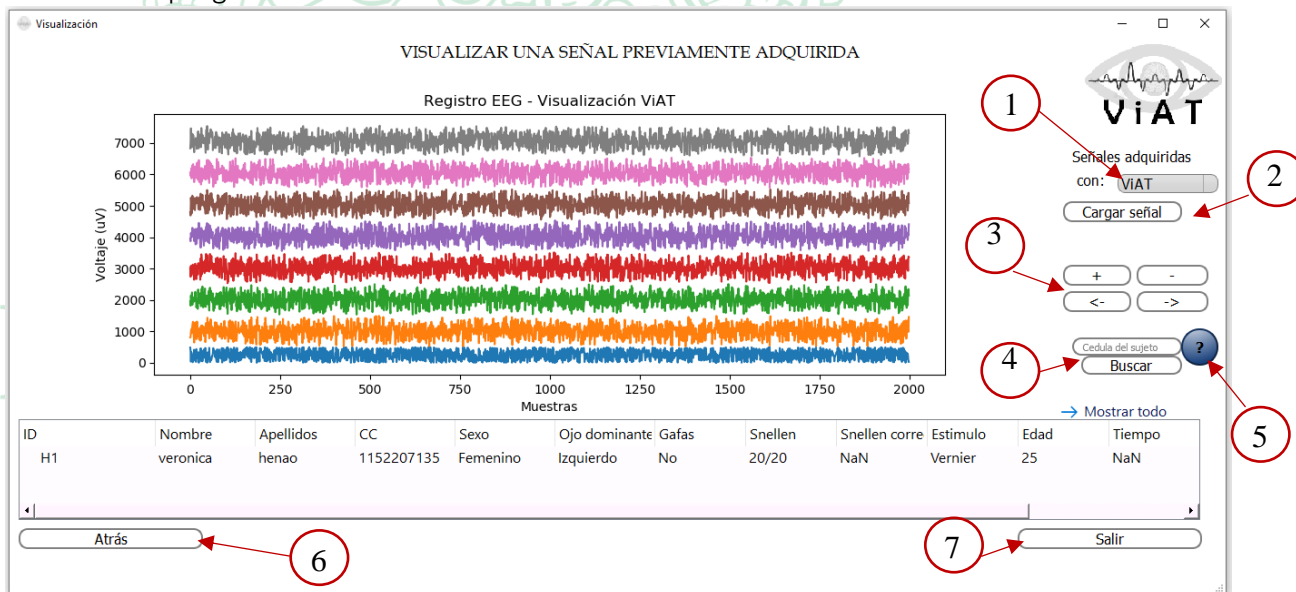


Figura 41. Visualización.

ESTIMULACIÓN AGUDEZA DE VERNIER

Stimulation_Acuity.py

Debido a la versatilidad de los movimientos sacádicos, se ha desarrollado con el tiempo un número importante de tareas sicomotoras para probar distintos mecanismos visuales, entre ellos la agudeza visual. El estímulo (Figura 8) consiste en un conjunto de segmentos verticales blancos sobre fondo negro, separados una distancia determinada para una distancia de 1 metro. Existe una imagen fija que intercala con los segmentos que cambian de distancia en el tiempo. La tarea inconsciente del observador es determinar el desplazamiento máximo horizontal entre las líneas que les permite verlas alineadas.



Figura 42. Estímulo de Vernier.

Stimulation_Acuity.py

Bibliotecas

pygame

Tabla 6. PyGame

Definición permite hacer aplicaciones multimedia como juegos contruidos sobre la excelente biblioteca SDL.	Instalación pip install pygame import pygame
pygame.init() inicializar todos los módulos de pygame importados	pygame.display.set_mode(size, flags, depth, display) Inicializar una ventana o pantalla para mostrar
quit() Finalizar todos los módulos de Pygame	pygame.image.load(filename) cargar nueva imagen de un archivo
pygame.display.flip() Actualice la superficie de visualización completa en la pantalla	pygame.image.load(filename) .get_width() // .get_height()

	Devuelve el ancho y el largo de la superficie en píxeles
pygame.transform.scale() Cambia el tamaño de la superficie a una nueva resolución.	pygame.display.update() Solo permite actualizar una parte de la pantalla, en lugar de toda el área. Si no se pasa ningún argumento, actualiza toda la superficie como <code>pygame.display.flip()</code> .
pygame.display.set_mode().blit(img,(width,height)) generar imagen fotorrealista a partir de un modelo, copiando los píxeles que pertenecen a dicho objeto en el objeto de destino.	pygame.KEYDOWN and event.key == pygame.K_SPACE Eventos cuando se presionan y sueltan los botones del teclado. En este caso al presionar la tecla SPACE
pygame.event.get() Esto obtendrá todos los mensajes y los eliminará de la cola. Si se proporciona un tipo o secuencia de tipos, solo esos mensajes se eliminarán de la cola.	

Time

Tabla 7. Biblioteca Time.

Definición Permite manejar tareas relacionadas con el tiempo.	import time
time.time() Devuelve el número de segundos transcurridos desde la época.	time.sleep(secs) La cantidad de segundos que el programa Python debería pausar la ejecución. Este argumento debe ser un int o un float.

Funciones 1 8 0 3

1. Init

Recibe: ID del sujeto registrado en la base de datos, Cédula del sujeto registrado en la base de datos, ubicación del registro.

Función: Ver: función: `start_stimulus` para más detalles.

StreamInfo: el objeto StreamInfo almacena la declaración de un dato

flujo, para describir sus propiedades y luego construir un StreamOutlet con él para crear la transmisión en la red.

StreamOutlet: los puntos de venta se utilizan para hacer streaming de datos (y los metadatos) disponibles en la red de laboratorio.

2. Display

Recibe: Número de la imagen a presentar.

Función: Ajusta el espacio en la pantalla y recibe la imagen diseñada con el grosor o característica de cada nivel de agudeza visual.

0: corresponde a la imagen intermedia entre cada visual estimulación y cambio en el nivel de agudeza visual.

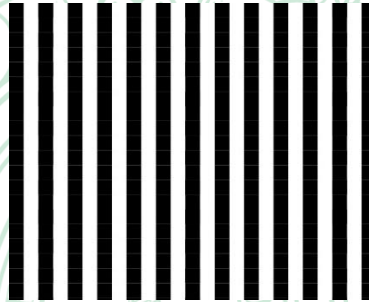


Figura 43. Imagen base. Separación 0.

0.1: corresponde a la imagen intermedia entre cada cambio de estado (binocular o monocular)

3. save

Recibe: Número de la Marca al cambiar imagen.

Función: Le permite guardar la marca de tiempo al cambiar las imágenes. Para este estímulo particular, permite separar las agudezas en 7 niveles

4. start_stimulus

Recibe: Número de la Marca al cambiar imagen.

Función: "Iniciar la estimulación Vernier.

num: niveles transversales del rango de agudeza visual (1,7) para 6 niveles de agudeza visual

marca de tiempo: Devuelve la marca de tiempo POSIX como flotante

push_sample: empuje una muestra en la salida.

Cada entrada en la lista corresponde a un canal.

Argumentos de palabras clave:

x: una lista de valores para insertar (uno por canal).

marca de tiempo - Opcionalmente el tiempo de captura de la muestra, de acuerdo con local_clock (); si se omite, se usa la hora actual. (por defecto 0.0)

event.get (): Pygame registrará todos los eventos del usuario en una cola de eventos

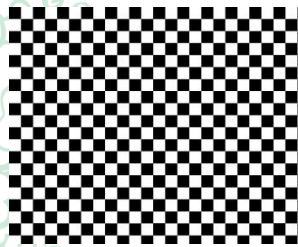
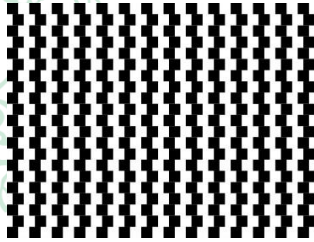
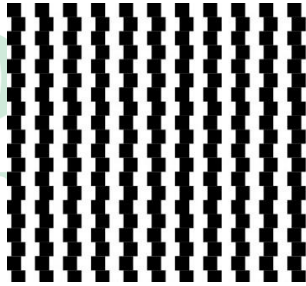
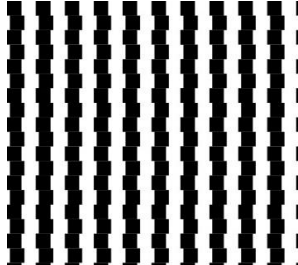
imágenes presentadas

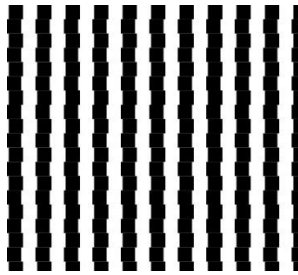
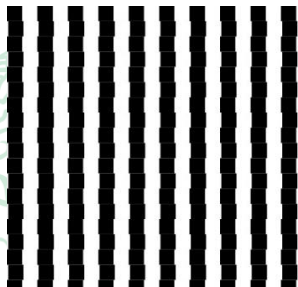
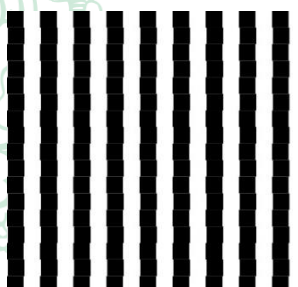
Cada imagen corresponde a un valor de agudeza en la escala de Snellen y se intercambian de la 1 a la 7. Se utilizó la Figura 10 como referente para estimar la separación de los cuadros en la estimulación.

Decimal	Fracción	Snellen (6 m)	Snellen (20 pies)	logMAR
0,10	1/10	6/60	20/200	1,0
0,12	1/8	6/48	20/160	0,9
0,16	4/25	6/37,5	20/125	0,8
0,20	1/5	6/30	20/100	0,7
0,25	1/4	6/24	20/80	0,6
0,32	1/3	6/19	20/63	0,5
0,40	2/5	6/15	20/50	0,4
0,50	1/2	6/12	20/40	0,3
0,63	2/3,2	6/9,5	20/32	0,2
0,80	4/5	6/7,5	20/25	0,1
1,00	1/1	6/6	20/20	0,0
1,25	5/4	6/4,8	20/16	-0,1

Figura 44. Conversión de valores de agudeza visual.

Tabla 8. Separación paradigma.

Número	Separación según Snellen [mm]	Agudeza Snellen	Imagen mostrada
1	5.817764173	>20/200	
2	2.908882087	20/200	
3	1.818051304	20/125	
4	1.163552835	20/80	

Número	Separación según Snellen [mm]	Agudeza Snellen	Imagen mostrada
5	0.727220522	20/50	
6	0.461727315	20/30	
7	0.290888209	>20/20	

PROCESAMIENTO CON MULTITAPERING

dataprocessing.py

Estimación espectral con multitapering

Funciones

1. Init

Recibe: ID del sujeto registrado en la base de datos, Cédula del sujeto registrado en la base de datos, fecha del registro, ubicación del registro y ubicación para guardar.

Función: Establece las variables con los elementos recibidos.

2. run

Recibe:

Función: Crea el directorio para guardar los resultados del procesamiento, lee los datos del registro y de las marcas, compara ambos archivos y separa la señal por cada marca, luego toma cada grupo de marca y aplica el procesamiento de Multitaper así:

`Sk_complex, weights, _ = pmtm(data[i], NW=2, k=4, show=False)`

La función *pmtm* es de la biblioteca *spectrum* y contiene las siguientes características

`def pmtm(x, NW=None, k=None, NFFT=None, e=None, v=None, method='adapt', show=False):`

: param array x: los datos
 : param float NW: el parámetro de ancho de banda medio del tiempo (los valores típicos son 2.5,3,3.5,4). Debe proporcionarse de lo contrario las ventanas cónicas y se deben proporcionar valores propios (salidas de dpss)
 : param int k: utiliza las primeras k secuencias de Slepian. Si no se proporciona * k *, * k * se establece en * NW * 2 *.

Por lo general, en la estimación espectral, la media para reducir el sesgo es usar la reducción gradual de ventana. Para reducir la varianza, necesitamos promediar diferentes espectros. El problema es que solo tenemos un conjunto de datos. Por lo tanto, necesitamos descomponer un conjunto en varios segmentos. Tales métodos son bien conocido: el periodograma de daniell, método de Welch, etc. El inconveniente de tales métodos es una pérdida de resolución ya que los segmentos utilizados para calcular el espectro son más pequeño que el conjunto de datos. El interés del método multitaper es **mantener una buena resolución** mientras se da la reducción de sesgo y varianza.

¿Como funciona? Primero se calcula un periodograma simple diferente con el conjunto de datos completo (para mantener una buena resolución) pero se calcula cada periodograma con ventanas diferentes, luego, promediamos todo este espectro para evitar la redundancia y el sesgo debido a los conos mtm.

Procesamiento con tiempo frecuencia.

plot_stft.py

Este procesamiento se da por medio del cálculo la transformada de Fourier de corto tiempo.

Funciones

1. Init

Recibe: ID del sujeto registrado en la base de datos, Cédula del sujeto registrado en la base de datos, fecha del registro, ubicación del registro y ubicación para guardar.

Función: Definir variables y ubicaciones de los archivos entregados.

2. plot_stft

Recibe:

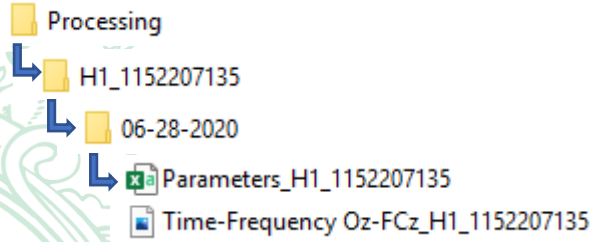
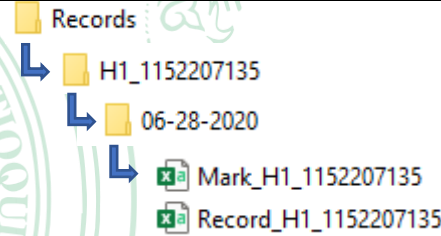
Función: Lee el registro en el canal Oz-FCz y aplica transformada de Fourier de corto tiempo *stft* de la biblioteca *scipy.signal*

Las STFT se pueden usar como una forma de cuantificar el cambio de la frecuencia de una señal no estacionaria y el contenido de fase en el tiempo.

1 8 0 3

Los resultados entregados por la herramienta, incluyendo los generados en: *Stimulation_Acuity.py*, *dataprocessing.py* y *plot_stft.py*

Tabla 9. Estructura.

Estructura	Visual
<ul style="list-style-type: none"> Processing <ul style="list-style-type: none"> ID_CC <ul style="list-style-type: none"> FECHA <ul style="list-style-type: none"> Parameters_ID_CC.csv Time-Frequency Oz-FCz_ID_CC.jpg 	
<ul style="list-style-type: none"> Records <ul style="list-style-type: none"> ID_CC <ul style="list-style-type: none"> FECHA <ul style="list-style-type: none"> Mark_ID_CC.csv Record_ID_CC.csv 	

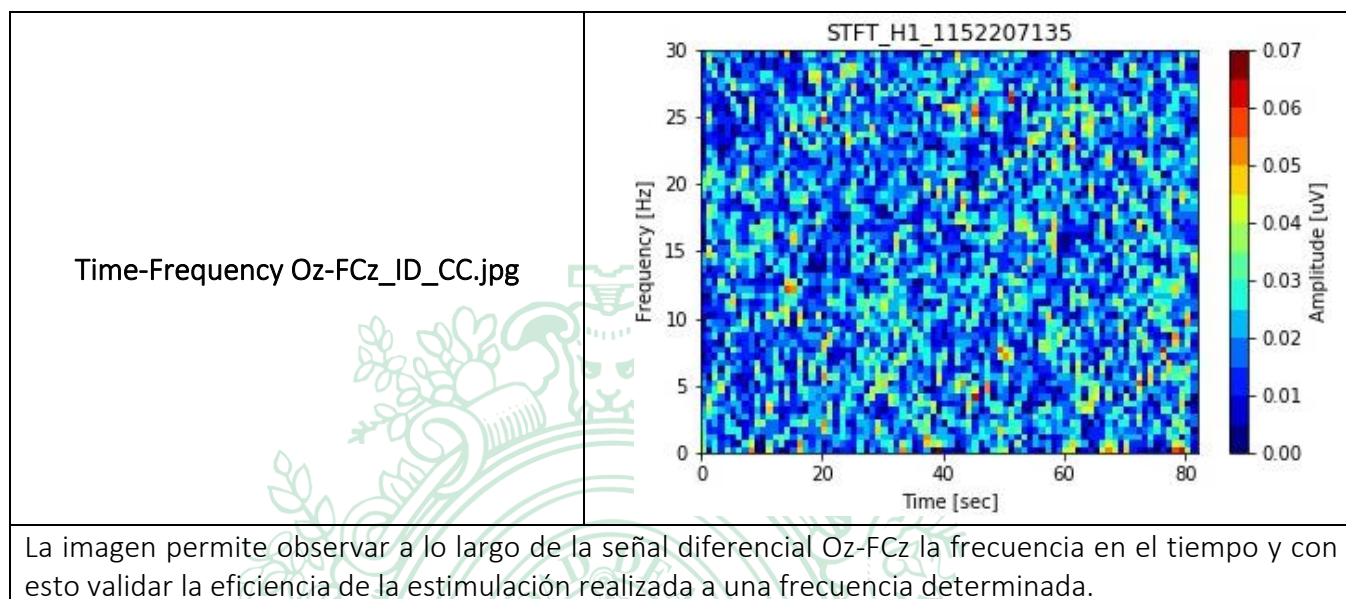
Se elijen los archivos .csv por su facilidad de manipulación.

La estructura de los archivos es la siguiente:

Processing

Tabla 10. Procesamiento.

Parameters_ID_CC.csv	Max	Fre
	0.59804803	7.08699902
	0.10245691	8.30889541
	0.40024617	8.30889541
	0.25079922	7.08699902
	0.07403019	8.30889541
	0.31702204	8.30889541
<p>Max: Hace referencia a la amplitud máxima encontrada en el rango de frecuencia de interés, en este caso entre 7 y 8.5 Hz</p> <p>Fre: Hace referencia a la frecuencia exacta a la que se encontró el valor máximo.</p> <p>Estos datos son útiles y necesarios para realizar evaluaciones posteriores de la señal con apoyo de Machine Learning [46] u otras técnicas de análisis estadístico.</p>		



Records

Tabla 11. Registros.

Record_ID_CC.csv	C1	C2	C3	C4	C5	C6	C7	C8	H
	0.73187554	0.1191259	-0.4066793	-0.2687161	-0.393944	-0.29221848	-0.30618772	-0.00962621	11-22-25
	0.52693057	0.33831501	0.45620424	0.24923992	0.23631233	0.08804327	0.11061227	-0.41921276	11-22-26
	0.92276698	-0.13704956	-0.58744523	-0.76322785	-0.19375223	-0.41278476	-0.42801601	-0.88206647	0
	0.30380246	-0.25563728	0.44580564	-0.19913614	0.23976383	-0.03268635	-0.22412168	-0.19581272	0
	0.30664465	0.64603671	0.38948187	-0.15265588	-0.137475	0.64516971	-0.04174143	0.37839356	0
	0.60575074	-0.12673205	0.18656027	-0.4891784	0.23382193	-0.36466084	0.34857655	0.21318001	0

Contiene todos los registros de los sujetos y sus respectivas marcas al realizar la estimulación.

C1: Representa el canal 1 es decir FCz, canal de referencia.

C2: Representa la diferencia entre el canal 1 y el canal 2 es decir Oz-FCz

C3: Representa la diferencia entre el canal 1 y el canal 3 es decir O1-FCz

C4: Representa la diferencia entre el canal 1 y el canal 4 es decir PO7-FCz

C5: Representa la diferencia entre el canal 1 y el canal 5 es decir O2-FCz

C6: Representa la diferencia entre el canal 1 y el canal 6 es decir PO8-FCz

C7: Representa la diferencia entre el canal 1 y el canal 7 es decir PO3-FCz

C8: Representa la diferencia entre el canal 1 y el canal 8 es decir PO4-FCz

H: La hora en la que se adquirió el segmento de datos. El formato es:
Hora-Minutos-Segundos

Los datos H = 0, quiere decir que hacen parte del mismo chunk de datos adquiridos durante la última hora guardada.

Mark_ID_CC.csv	H	M
	11-23-09	1
	11-23-14	2
	11-23-18	3
	11-23-23	4
	11-23-27	5
	11-23-31	6
	11-23-36	7
	11-23-44	8

H: Representa la hora en la que se hizo el cambio de imagen del estímulo es decir, al llegar la marca, el formato es:

Hora-Minutos-Segundos

M: Representa el número de marca que llegó en ese momento iniciando en **1**, para el estímulo presentado Vernier se evaluando **7** agudezas, una marca por cada una y una **8va** marca al finalizar el registro.

Servidor

Server.py

Un **servidor** es una aplicación en ejecución capaz de atender las peticiones de un cliente y devolverle una respuesta en concordancia.

Unas bibliotecas útiles para crear este tipo de servidor compatible con OpenBCI son pyOpenBCI y pylsl.

pyOpenBCI proporciona un controlador Python estable para todos los biosensores OpenBCI.

Pylsl proporciona un conjunto de funciones para crear datos transmitidos en tiempo real dentro de una red.

Inicialmente se deben instalar e importar las bibliotecas necesarias:

```
from pyOpenBCI import OpenBCICyton
from pylsl import StreamInfo, StreamOutlet
import numpy as np
from serial.tools import list_ports
from datetime import datetime
```

Para iniciar la comunicación se escribe:

```
board = OpenBCICyton(port='COM3', daisy=False)
```

El valor de “port” se puede encontrar en -Panel de control-Administrador de dispositivos - Otros dispositivos- o usando serial.tools.list_ports.comports()

El dispositivo OpenBCI contiene 8 salidas para los canales y 3 salidas auxiliares. Para los canales se recomienda convertir los canales_datos a uVolts multiplicando por **(4500000) / 24 / (2 ** 23 - 1)** actualmente se multiplica únicamente por **(4500000)** para amplificar los datos y para los auxiliares se recomienda multiplicar por **0.002 / (2**4)**

El objeto StreamInfo almacena la declaración de una secuencia de datos.

```
StreamInfo('OpenBCIEEG', 'EEG', 8, 250, 'float32', 'OpenBCItestEEG')
```

En este se incluye la siguiente información: Nombre, Tipo, número de canales, frecuencia de muestreo formato o tipo de dato, identificador del dispositivo (ID)

Primero creará un StreamInfo para describir sus propiedades y luego se construye un StreamOutlet con él para crear la transmisión en la red.

El objeto StreamOutlet se utilizan para hacer que los datos de transmisión (y los metadatos) estén disponibles en la red del laboratorio.

```
outlet_eeg = StreamOutlet(info_eeg)
```

Requiere del StreamInfo para tomar la información, opcional se puede incluir tamaño del Chunk (muestras en la transmisión), y cantidad máxima de datos (buffered) para almacenar.

Predeterminado 6 min.

`streams = resolve_stream('type', 'Markers')` devuelve todas las transmisiones disponibles desde cualquier salida en la red en tiempo real. Y al utilizar el objeto StreamInlet para recibir datos de transmisión (y metadatos) desde la red del laboratorio, toma las transmisiones

disponibles y finalmente se hace un `pull_chunk()` que extrae un trozo de muestras de la entrada.

Comenzar a manejar la transmisión de datos, haciendo pull de las marcas, push de los datos y evaluando continuamente la existencia de las marcas.

Pull: Recibe los datos de la red

Push: Envía los datos a la red

```
def lsl_streamers(sample):
    sample_mark, timestamp = inlet.pull_sample(timeout=0.0)
    outlet_eeg.push_sample(np.array(sample.channels_data)*SCALE_FACTOR_EEG)
    outlet_aux.push_sample(np.array(sample.aux_data)*SCALE_FACTOR_AUX)
    if sample_mark is not None:
        print(sample_mark)
```

Para llamar la transmisión se ejecuta:

```
board = OpenBCICyton(port='COM3', daisy=False)
board.start_stream(lsl_streamers)
```

Bibliotecas

OpenBCI: Significa interfaz de cerebro-computadora de código abierto (BCI). Es una herramienta que proporciona a cualquier persona con una computadora, las herramientas necesarias para tomar muestras de la actividad eléctrica de su cuerpo.

Primero, asegúrese de tener las dependencias necesarias.

```
pip install numpy pyserial bitstring xmltodict
```

pyOpenBCI

Tabla 12. PyOpenBCI.

Definición Proporcionar un controlador Python estable para todos los biosensores OpenBCI	<code>pip install pyOpenBCI</code> <code>from pyOpenBCI import OpenBCICyton</code>
OpenBCICyton(port, daisy=False) Maneja la conexión a una placa OpenBCI Cyton. La clase OpenBCICyton interactúa con el Cyton Dongle y la placa Cyton para analizar los datos recibidos y enviarlos a Python como un objeto OpenBCISample.	

pysl

Tabla 13. Pysl

Proporciona un conjunto de funciones para crear datos de instrumentos accesible en tiempo real dentro de una red. A partir	<code>pip install pylsl</code> <code>from pylsl import StreamInfo, StreamOutlet</code>
--	---

de ahí, las corrientes pueden ser recogido por programas de grabación, programas de visualización o experimentos personalizados para aplicaciones que acceden a flujos de datos en tiempo real.	from pylsl import StreamInlet, resolve_stream
resolve_stream('type', 'Markers') Esta función devuelve todas las transmisiones disponibles actualmente desde cualquier salida en la red.	StreamInlet(streams[0]) Las entradas se utilizan para recibir datos de transmisión (y metadatos) desde la red del laboratorio.
StreamInlet() .pull_chunk() Saque un pedazo de muestras de la entrada	OpenBCICyton.start_stream(self, callback) Comience a manejar la transmisión de datos desde el tablero. Llame a una devolución de llamada proporcionada por cada muestra procesada.

Funciones

1. Init

Recibe:

Función: Define las variables para *StreamInfo* y para *StreamOutlet*

2. IsStreamers

Recibe: Las muestras desde el dispositivo (Samples)

Función: Realiza la conversión de los datos sugerida por el OpenBCI y realiza un push de los datos. Saca un pedazo de muestra y empuja una muestra por canal a la salida.

3. Port

Recibe: Las muestras desde el dispositivo (Samples)

Función: Evalúa la lista de puertos en uso y establece el puerto a utilizar.

Simulación

Permite crear una nueva transmisión de información y establecer un número de serie para el dispositivo o identificador único local para la transmisión (también se podría omitir, pero las conexiones interrumpidas no se recuperarían automáticamente).

Funciones

1. Init

Recibe:

Función: Define las variables para *StreamInfo* y para *StreamOutlet*

2. sample

Recibe:

Función: Crea un arreglo de datos aleatorios en un ciclo continuo y genera un *push_sample*, empujando una muestra por canal a la salida.



UNIVERSIDAD DE ANTIOQUIA

1 8 0 3