Diseño e Implementación de un Sistema Generador de Pulsos Binaurales para el Estudio del Impacto de los Pulsos en la Calidad de Sueño

Luis André Guerrero Sifontes

UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Diseño e Implementación de un Sistema Generador de Pulsos Binaurales para el Estudio del Impacto de los Pulsos en la Calidad de Sueño

Trabajo de graduación presentado por Luis André Guerrero Sifontes
para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala,

2020

Vo.Bo.:

(f) _____

Dr. Luis Alberto Rivera Estrada

Tribunal Examinador:

(f) _____

Dr. Luis Alberto Rivera Estrada

(f) _____

(f) _____

Fecha de aprobación: Guatemala, de de 2020.

Prefacio

Índice

Lista de figuras

Lista de cuadros

Resumen

El presente trabajo se llevó a cabo debido a los distintos trastornos del sueño que sufre una buena porción de la población en la actualidad. Existen varios factores que afectan nuestra calidad de sueño, tales como el estrés, presión, entre otros.

Independientemente de los factores que causan el problema, en este trabajo se planteó explorar una posible solución, la cual es la utilización de Estimulación por Pulsos Auditivos (ABS, por sus siglas en inglés - *Auditory Beat Stimulation*). ABS consiste en pulsos auditivos que estimulan las señales Electroencefalográficas (EEG) del cerebro, de manera que puedan afectar positivamente tanto la cognición de las personas como la calidad del sueño de las mismas.

De los distintos pulsos ABS, este trabajo se enfocó en los llamados pulsos binaurales. Estos pulsos son el resultado de una interpretación del cerebro, que sucede cuando dos fuentes de sonido, a frecuencias distintas, se colocan en cada uno de los oídos. La meta de este trabajo fue poder replicar pulsos binaurales en un dispositivo compacto y de bajo costo, y también cómodo y fácil de usar para los usuarios.

Luego de tener un prototipo funcional de replicación de pulsos binaurales, se realizaron pruebas con algunos individuos, para obtener la opinión que ellos tienen acerca de los pulsos y cómo estos afectan el sueño. Se espera que el prototipo y los resultados preliminares sirvan como punto de partida para estudios más profundos sobre la calidad del sueño y su posible corrección.

Abstract

This is an abstract of the study developed under the

CAPÍTULO 1

Introducción

El trabajo realizado desarroló un protoripoque busca mejorar la calidad del sueño de las personas, ya que muchas personas sufren distintos trastornos que no les permiten conciliar el sueño de la mejor manera posible, [[REVISAR: Según investigaciones realizadas en el ámbito científico los pulsos binaurales, podrían ser la solucion a las afecciones de sueño de las presonas.

[[INFORMAL: A grandes rasgos]] se busca teniendo un dispositivo fácil de utilizar, reproducir pulsos binaurales que logren estimular el cerebro de manera tal, que la persona que sea expuesta a estos pulsos sea capaz de dormir más plácidamente, esto ocurre cuando la persona escucha estos pulsos binaurales, la mejor manera de escuchar los mismos es por medio de audífonos conectados a alguna fuente de sonido. El sonido al cual el sujeto de prueba este expuestodebe de ser *surround*, ya que esto influye a la hora de oír los pulsos binaurales.

El trabajo se divide en tres etapas, la primera, tomando en consideración el trabajo previo de José Pablo Muñoz en [1] se origina la tarea de adaptar lo realizado por el estudiante a un dispositivo el cual fuese más cómodo y mas práctico a la hora de utilizarlo (en este caso se utilizó la Raspberry Pi y una pantalla compatible de 7"). Luego de haberse adaptado trabajo anterior, se desarrolló una interfaz gráfica amigable con el usuario, para poder generar, y reproducir los pulsos binaurales con las especificaciones deseadas por el usuario. Finalmente se hizo una evaluación la cual mide de manera cualitativa la calidad del sueño de las personas al momento de ser expuestas a pulsos binaurales al momento de dormir, para poder comprobar de una manera científica que este tipo de excitación sonora puede ayudar a los distintos trastornos de sueño.

CAPÍTULO 2

Antecedentes

Este trabajo es la continuación de la tesis realizada por el estudiante José Pablo Muñoz [1], la cual se basaba en varios aspectos de las etapas del sueño y observar cómo estas se ven afectadas por distintos estímulos, así como la clasificación de las etapas por medio de Aprendizaje Automático (*Machine Learning*).

En el caso del trabajo anterior se utilizó una computadora portátil para generar pulsos binaurales que afectan las etapas de sueño. Los pulsos binaurales se generan digitalmente, seguidamente se transmite el sonido al sujeto de prueba por medio de auriculares. Lo que se busca en este trabajo es encontrar una manera más conveniente y eficiente de generar y transmitir estos pulsos, por medio de un dispositivo más compacto y menos costoso, como una RaspberryPi.

El trabajo de José Pablo Muñoz tiene un espectro de estudio más general, debido a que en este se trabaja en la reproducción de los pulsos binaurales, y la lectura e interpretación de las señales EEG. En el caso de esta tesis, la misma se centrará mayormente en la reproducción efectiva de los pulsos binaurales.

Los pulsos binaurales son una 'ilusión' cerebral, ya que estos son la interpretación que el cerebro le da a dos sonidos con frecuencias distintas, cada una en las dos terminales auditivas del ser humano. Muchas veces estos pulsos son utilizados para modificar la cognición, y estados de ánimo de las personas, ya que los mismos afectan las ondas cerebrales que están estrechamente relacionadas las etapas del sueño. Esto ha llevado a científicos en el mundo ha investigar este fenómeno para poder investigar las nuevas aplicaciones que pueden tener estas señales.

Estudios recientes en [2] indican que los procedimientos ABS han llegado a afectar a la creatividad, atención y memoria de las personas. Estos estudios concluyen que los procedimientos pueden ser utilizados para potenciar la cognición e incluso alterar el sentido del humor.

Otro estudio bajo la misma linea de procedimientos ABS y pulsos binaurales, más enfo-

cado en el área del sueño [3], indica que es posible lograr inducir las señales EEG necesarias para el sueño, siempre que el sonido utilizado no sea molesto para los sujetos de prueba.

CAPÍTULO 3

Justificación

Muchas personas sufren trastornos del sueño. Según [4], aproximadamente el 6.5 por ciento de la población sufre apnea, 34 por ciento ronca, aunque solo 16.4 por cierto lo hace frecuentemente. Además, en poblaciones como la mexicana, por ejemplo, el 30 por ciento de la población sufre trastornos del sueño, lo cual es un porcentaje significativo. Normalmente, estos tipos de padecimientos no son tratados, y llegan a afectar de manera significativa la vida de las personas, debido a que un buen descanso está directamente relacionado con la productividad de las mismas.

Por todo lo antes mencionado, se quiere explorar un área que puede llevar a una solución. Como se indicó anteriormente, los pulsos binaurales pueden tener un efecto en el sueño, por lo que se propuso crear un dispositivo fácil de usar y de bajo costo que replique estos pulsos. El dispositivo puede ser muy útil en futuros estudios de análisis de sueño.

Generalmente, para controlar las enfermedades del sueño, se utilizan variedad de medicamentos que muchas veces ayudan a lograr conciliar el sueño pero tienen efectos secundarios los cuales no son deseados por los usuarios. El uso de pulsos binaurales puede ayudar a tratar enfermedades relacionadas con el sueño sin la necesidad de medicamentos.

CAPÍTULO 4

Objetivos

Objetivo General

Diseñar e implementar un generador de pulsos binaurales compacto y de bajo costo, que permita estudiar los efectos de los pulsos en la calidad de sueño de las personas.

Objetivos Específicos

- Optimizar el algoritmo de generación de pulsos binaurales desarrollado en la fase previa del proyecto.
- Implementar el algoritmo de generación de pulsos en un dispositivo compacto y de bajo costo.
- Diseñar e implementar una interfaz de usuario que permita configurar y generar los pulsos de forma fácil y eficiente.
- Diseñar un protocolo para la aplicación de los pulsos binaurales de forma segura, y para la evaluación del impacto de los pulsos en la calidad del sueño de las personas.

CAPÍTULO 5

Alcance

El alcance de este proyecto es como los objetos lo indican lograr un dispositivo pequeño y cómo para le usuario que logre replicar eficazmente pulsos binaurales, esto implica tambien a grandes rasgos que los usuarios puedan utilizar fácilmente el dispositivo con la aplicación correspondiente. Para lograr esto se desarrolló tanto una aplicación sencilla de utilizar para el usuario como un manual de usuario que indica a detalle como funciona cada uno de los módulos del programa.

Este proyecto no busca resolver los trastornos del sueño aún, pero busca la implementación del dispositivo antes mencionado, para poder reproducir los pulsos, y definir un rango en el cual los mismos induzcan al sueño de mejor manera. Para esto se generó un protocolo el cual determina los rangos de frecuencias de los pulsos que impactan de mejor manera el sueño.

Para trabajos futuros se tiene la implementación de otra tesis que se trabajó en paralelo, para lograr obtener las señales EEG del usuario, y poder aplicar pulsos binaurales para que el cerebro genere ciertas ondas EEG, llegando así a la etapa de mejora del sueño.

CAPÍTULO 6

Marco teórico

6.1. Señales EEG

Las señales electroencefalográficas (EEG), son causadas por los pulsos eléctricos generados por el cerebro. El interés de investigar este tipo de señales es que el cerebro es el órgano que manda instrucciones a los demás órganos para que estos realicen sus actividades adecuadamente. Se sabe que ciertas partes del cerebro controlan ciertos movimientos de partes del cuerpo. Dependiendo del lugar del cerebro que estemos recibiendo las señales, podemos tener una noción de las instrucciones que este está enviando.[5]

El espectro de frecuencia que las señales EEG abarca son de 1-30 Hz en su mayoría, pero a continuación se describirán los distintos espectros para los distintos tipos de señales que tiene el cerebro.

Delta: Tiende a ser la señal más lenta (0.1 - 4Hz), pero con mayor amplitud comparada a las demás, generalmente estas señales se dan por actividad en el lóbulo frontal del cerebro. Es anormal en adultos generalmente se da en bebés durmiendo.

Figura 1: Onda Delta [5]

Theta: De igual manera es una señal un poco lenta de 4 - 8 Hz, es normal este tipo de actividad en niños o en adultos durmiendo, pero no es común en adultos despiertos. Dependiendo de las señales y del individuo que las causa este tipo de señales puede llegar a diagnosticar lesiones subcorticales y encefalopatías. Contribuye a la hormona de crecimiento humano, y otras hormonas que ayudan a la memoria.

Figura 2: Onda Theta [5]

Alpha: Señales de 8 a 13 Hz, se presentan tanto en adultos como en niños, para el grupo de adultos tenemos que este tipo de señales se presentan cuando están relajados o bien con los ojos cerrados, se registran en la parte parietal y occipital del cerebro. Este tipo de señales contribuyen a la generación de serotonina que es un químico que relaja a los seres humanos.

Figura 3: Onda Alpha [5]

Beta: Son señales rápidas mayores de 13-30 Hz, tienen que ver con actividades que hacen a la persona estar concentrada, causan la generación de cortisol en el cuerpo que es una hormona que acelera el cuerpo. Estas señales ocurren cuando el individuo esta despierto ya que se necesita una gran actividad cerebral, generalmente de los lóbulos frontales y parietales.

Figura 4: Onda Beta [5]

Gamma: Son las señales más rápidas (30 - 100 Hz) causadas por el cerebro generalmente son causadas en situaciones de alerta extrema que hace que el individuo utilice dos sentidos al mismo tiempo para lograr su objetivo.

Figura 5: Onda Gamma [5]

6.2. Etapas del sueño

Como se ha mencionado a lo largo de este trabajo tenemos que el sueño es una parte vital en la vida de las personas, llegando a abarcar la tercera parte de nuestra vida.

El sueño está compuesto por 2 etapas, REM, que por sus siglas en inglés significa Movimiento rápido de los ojos, y NREM, que por sus siglas en inglés significa Movimiento no rápido en los ojos.

La etapa REM, es identificable por tener amplitudes más bajas y una frecuencia más rápida que está directamente relacionada con el movimiento de los ojos, cabe mencionar que en esta etapa no se clasifica como sueño profundo, ya que se registra bastante actividad cerebral. Cabe mencionar que este movimiento de los ojos en la etapa es fuertemente relacionado a los sueños debido a que estudios que nos mencionan que muchas personas al ser despertada en esta etapa dicen haber soñado [6]. Otra situación interesante de esta etapa es que nuestros músculos de los brazos y piernas están temporalmente paralizados.

Seguidamente la etapa NREM, se pude dividir en 3 sub-etapas N1, N2, N3, en las cuales va decreciendo la velocidad de las señales EEG, teniendo frecuencias más bajas, esta etapa es fuertemente relacionada con las etapas más profundas del sueño.

Como podemos ver en las gráficas, entre las señales EEG tienen una frecuencia más baja, su amplitud aumenta, estas figuras muestran las distintas etapas tanto como la REM, como las distintas etapas de NREM.

Figura 6: Señales EEG etapa REM [6]

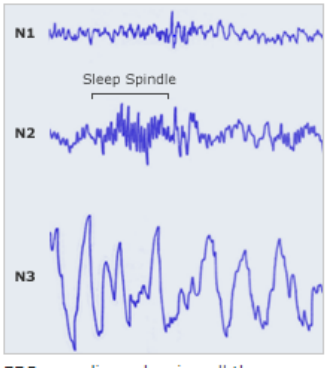

Figura 7: Señales EEG etapa NREM [6]

6.3. Ondas Sonoras

6.3.1. Interpretación física de las ondas sonoras

El sonido, según [7], es producido debido a las vibraciones de presión externas al oído. Físicamente estas vibraciones, u ondas mecánicas, chocan con el tímpano, lo cual causa que se repliquen ondas de la misma frecuencia, obteniendo de esta manera la sensación de sonido.

Un oído normal, puede distinguir ondas de frecuencias desde los 16 Hz, hasta los 20,00 Hz, y la amplitud de umbral de audición y hasta que las mismas causen sensaciones de dolor. Hay que mencionar que en este caso, las frecuencias más bajas corresponden a sonidos más graves, mientras que frecuencias más altas son propias de sonidos más agudos, esto no tiene nada que ver con la amplitud de variación de presión, que es el rango en el cual los humanos podemos tanto percibir sonido como soportarlo.

De los conceptos mencionados anteriormente dos definiciones muy importantes que existen son:

- La onda sonora u onda mecánica que causa vibraciones en su medio
- La sensación de sonido que el oído tiene a la onda sonora antes mencionada

Hay que diferenciar estos dos conceptos, ya que el sonido como tal, parecido a la luz, es la interpretación de los sistemas biológicos del ser humano para poder entender de mejor manera el ambiente que lo rodea.

La sonoridad [7] (sensación de sonido) no puede ser medida por un aparato debido a que cada una de las personas tiene una percepción distinta al sonido, se pueden estar recibiendo las mismas ondas mecánicas, pero depende del organismo de cada persona como las va a percibir. El oído humano no tiene la capacidad de identificar la intensidad de un sonido "linealmente", si la intensidad de un sonido se duplica, el oído no tiene la capacidad para percibir ese incremento. La audición de los seres humanos tiene un tipo de comportamiento logarítmico con relación a la intensidad de las ondas sonoras que percibe. Generalmente las ondas sonoras que percibe el oído humano son provocados por la unión de varios componentes sonoros, que tienen distintas frecuencias, y depende de la intensidad de cada uno de los componentes que tanto afecte a la onda sonora final (es decir entre más intensidad sonora tiene una onda, su componente en frecuencia, tenderá a oírse más).

Se debe de tener una manera efectiva, la cual nos indique el rango auditivo, de las personas las cuales tienen una audición normal. Esto se logra generando algunas curvas que nos muestran el espectro audible del ser humano, tanto en frecuencia como en intensidad acústica, como se mencionó anteriormente el oído humano tiene la capacidad de percibir de 16 Hz a 20,000 Hz, mientras que el nivel de presión sonora (que se mide en dB) puede variar desde valores un poco menores de 0 hasta valores de 120-130. Cabe mencionar que hay valores mucho más altos de presión sonora, pero el oído humano no es capaz de soportarlos.

Con los conceptos previamente aclarados, se muestra la gráfica de espectro auditivo del ser humano, la cual demuestra el comportamiento de las ondas sonoras, dependiendo tanto de su frecuencia como de su presión sonora.

[[SÉ CONSISTENTE AL COLOCAR LAS REFERENCIAS EN LOS CAPTIONS DE LAS FIGURAS. EN ALGUNAS TENÉS "obtenidas de: ", Y EN OTRAS NO. QUE SEA LO MISMO EN TODAS LAS FIGURAS. YO QUITARÍA LO DE "obtenidas de: ", PERO ES TU DECISIÓN.]]

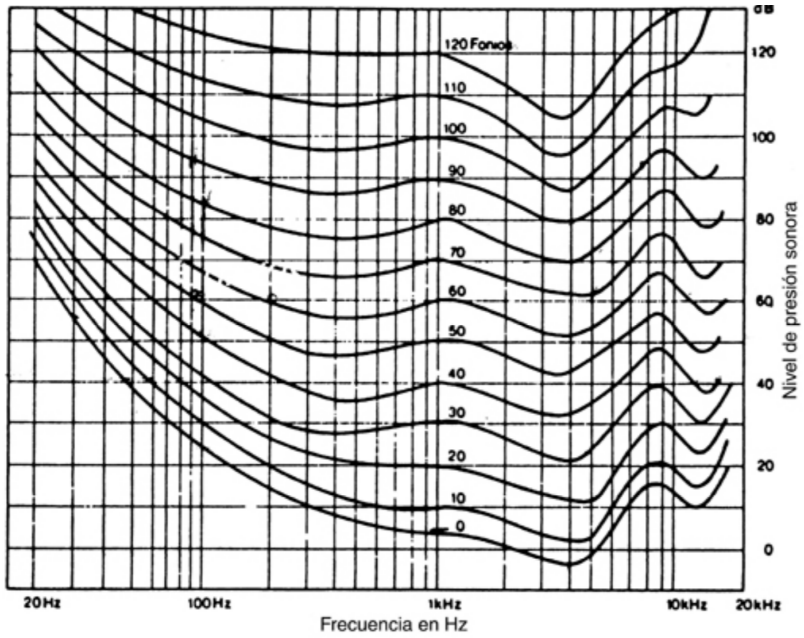

Figura 8: Rango Audible de Frecuencias y su nivel de presión sonora en dB [7].

6.3.2. Interpretación neuronal de las ondas sonoras

En la sección anterior se ahondó sobre el tema físico de las ondas sonoras, de igual manera del oído y una breve descripción de como se replican las ondas sonoras, pero en esta sección se abordará en tema neuronal, y la interpretación que el sistema nervioso le da a las ondas sonoras que el tímpano recibe. Luego que el sonido es recibido por el tímpano, el sonido se transfiere por algunos órganos antes de ser interpretado por el sistema nervioso, llegando hasta la cloquea que es el órgano más grande (del sistema auditivo) y el encargado de conectarse al sistema nervioso, para la interpretación del mismo sonido.

[MEJOR: EN LA FIGURA [REF] A continuación tenemos una imagen de los distintos órganos que están involucrados (Conexión entre órganos auditivos y sistema neuronal):

Teniendo una idea de que órganos están involucrados en la audición por parte del sistema nervioso, empezamos con que luego de que la cloquea recibe el sonido esta envía el mismo al núcleo cloquear, que, por conexiones nerviosas son transmitidas a distintas secciones del sistema Olivar en el cerebro, siempre considerando que estas tienen la interpretación de sonido que es proporcionada por la cloquea.

Una parte muy importante de para descifrar el sonido es el núcleo del sistema olivar, ya que este realiza el trabajo de ubicar la posición del sonido, ya que es el primer sistema nervioso el cual detecta el sonido y ya le da una interpretación, también cabe añadir que en

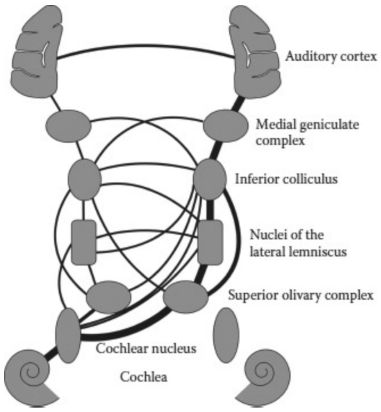

Figura 9: Órganos del sistema nervioso para la interpretación del sueño [8]

núcleo cloquear se da la decodificación que le indica al cerebro la frecuencia e intensidad del sonido, obteniendo así información valiosa pero no tan procesada.

Tálamo auditivo

Habiendo procesado la información el sonido llega a tálamo auditivo, el cuál está fuertemente controlado por el colículo inferior. Hay otros caminos que el sonido tiene para poder llegar a la corteza auditiva pero es mucho más fácil describir este comportamiento desde la respuesta y comportamiento del tálamo al sonido.

Podemos subdividir al tálamo auditivo en tres secciones las cuales cumplen distintas funciones(todas consideradas por sus siglas en inglés):

- MGv *ventral*
- MGd *dorsal*
- MGm *medial*

Con respecto a la primera sección del tálamo auditivo (MGv) esta se encarga de responder bien a los tonos de frecuencias puras, gracias a esta sección del tálamo la respuesta auditiva es bastante rápida logrando distinguir pulsos rápidos de estimulación. Además de las cuestiones antes mencionadas la MGv ayuda a ubicar la posición de la fuente del sonido.

Seguidamente MGd es la parte del tálamo encargada del reconocimiento de patrones en el sonido. Al contrario que la parte MGv encontramos que esta sección no reacciona de buena manera a los tonos puros, estas neuronas se especializan en tonos complejos los cuales contienen varias frecuencias, y juntos con MGv puede distinguir ese espectro de tonos, funcionando al mismo tiempo para distinguir distintos sonidos.

Finalmente con la sección MGm, la cual no contiene neuronas especializadas a tonos puros o a tonos complejos, sino que es una especia de híbrido entre ambos ,se cree que busca

llenar el espacio en medio de los dos sistemas anteriores, por ejemplo cuando no es un tono tan complejo pero tampoco es un tono puro.

Corteza auditiva

Habiendo descrito las distintas partes del tálamos auditivo, se llega al final del sistema nervioso encargado de interpretar el sonido, con la corteza auditiva, en la cual el sonido ya puede ser memorizado y puede llegar a causar una reacción física, es la última parte del cerebro involucrada con la interpretación de sonido.

La corteza auditiva se puede dividir, entres áreas mayores, las cuales se ven definidas por distintos parámetros tales como:

- Conexiones
- Arquitectura
- Organización funcional

Habiendo definido el porque de las divisiones, nos topamos con que las divisiones son(se utilizaran los términos en ingles a los cuales no se les encontró una traducción idónea):

- *Core*
- *Belt*
- *Parabelt*

El *Core* recibe información directa desde las neuronas de MGv y reibe alguna información de las neuronas de MGm, esta área de la corteza auditiva está conectada mucho a si misma, y está conectada con otra área de la corteza auditiva, la cual es el *Belt*, comparado a las otras regiones esta región tiene las siguientes características:

- Latencias de respuesta corta
- Funciones de sintonización de frecuencia espectral estrecha
- Sintonización de frecuencia de modulación relativamente rápida

La región *belt* recibe información tanto de MGm como de las neuronas de MGd , la misma esta subdivididas en dos subregiones, las cuales son distintas en cuanto a conectividad. La parte media esta conectada mayormente consigo misma y con la región *core* y tiene algunas conexiones con *parabelt*. La parte lateral de la región *belt* está mayormente Las neuronas de conectada consigo misma agregando algunas conexiones a las otras dos regiones.

Las neuronas de esta región tienen las siguientes características:

- Funciones de sintonización de frecuencia espectral más ancha.
- Entiende de buena manera los sonidos con frecuencias espectrales complejas.
- Exhiben latencias de respuesta más largas que *core*.

Finalmente la región *parabelt* igual qu la región mencionada anteriormente recibe información de las neuronas de las regiones del tálamo MGm Y MGd, y está conectada mayormente con si misma pero tiene algunas conexiones con *belt*. Esta sección de la corteza auditiva no se ha podido descubrir tanto de ella, pero se cree que esta es más responsiva para sonidos que son complejos, tanto espectralmente como en el tiempo, un ejemplo de esto serían las vocalizaciones.

Estos son los mecanismos que utiliza el sistema enrvioso para poder identificar y clasificar el sonido, cabe anadir que la información de esta sección fue obtenida de [8].

6.4. Pulsos binaurales

Por definición los pulsos binaurales son el entendimiento que le da el cerebro a dos señales de frecuencias distintas (obviamente en el espectro auditivo del ser humano). Estos son transmitidos por medio de auriculares para garantizar que se están escuchando dos frecuencias distintas en cada oído. La parte del cerebro encargada de procesar esta información es el núcleo olivar superior de cada hemisferio del cerebro correspondiente. [2]

El concepto de pulso binaural es un tanto más complejo que una interpretación cerebral, para aclarar el concepto, se dará la siguiente explicación, según [9]: La señal recibida por cada oído es transmitida a la corteza auditiva. Al llegar al tronco encefálico, las señales llegan al núcleo olivar superior, el cual es la primer parte del cerebro que recibe la señal bilateralmente, ya que hasta el momento las señales estaban separadas. Ya en el núcleo olivar el pulso binaural se genera espontáneamente teniendo una frecuencia igual a la resta de las dos señales auditivas iniciales. Después se envía a la corteza auditiva primaria ya como un pulso binaural, reenviandose a áreas cerebrales relacionadas con el sentido del oído, haciendo que las señales EEG oscilen a la frecuencia de oscilación del pulso binaural generado.

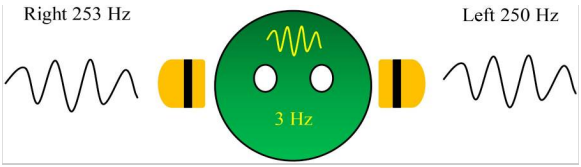

Figura 10: Diagrama pulso binaural [9]

Como se explicó anteriormente dependiendo de la actividad cerebral son las ondas cerebrales, y estas están directamente relacionadas con las etapas del sueño, ya que se ha observado que los pulsos binaurales logran afectar de manera más significativa la etapa del sueño más profundas tenemos que evaluar estas al momento de reproducir nuestros pulso.

Considerando que los pulsos binaurales al final son un sonido que se desea transmitir, debemos de saber cual es el rango de frecuencias en los cuales estos pulsos son percibidos por el ser humano, en [10] se indica que para frecuencias mayores a 1000 Hz, la mayoría de los sujetos de prueba, no detectaban el pulso binaural, o bien, se tenían que concentrar mucho para lograr detectarlo, además de esto se utilizaban diferencias de frecuencia muy bajas para poder distinguir los pulsos binaurales con esta frecuencia que en este marco de referencia se considera alta. También se concluyó que la diferencia máxima (siendo el pulso auditivamente perceptible) se encuentra en 400 Hz aproximadamente, logrando una diferencia aproximada de 35 Hz entre cada pulso.

Debido a que los pulsos binaurales necesitan tener frecuencias distintas en cada receptor de audio, en este caos los oídos, la manera más fácil de aplicarlos es utilizando un sonido surround de los audífonos, ya que estos pueden manejar frecuencias distintas en cada terminal, lo cual los hace ideales para esta situación.

Una ventaja de los pulsos binaurales es que son un método no invasivo que altera las señales EEG propias del cerebro haciéndolas resonar a la frecuencia del pulso binaural generado internamente. Además estudios anteriores que evalúan el comportamiento de las personas que son expuestas a los pulsos binaurales concluyen que hay un aumento en la fase electro-córtica, que puede causar los siguientes beneficios:

- Comunicación Neuronal
- Plasticidad Neuronal
- Memoria

Entre otros mencionados anteriormente, notamos que los pulsos binaurales si tienen un efecto tangible sobre los sujetos de prueba. [11]

6.5. Formatos digitales de sonido

[[REDACCIÓN: EVITAR ORACIONES TAN LARGAS:]] Debido a la variedad de sistemas operativos y las distintas maneras de almacenar información en estos días se llega a que hay muchos formatos digitales de audio, cada uno con sus características particulares que lo hacen único, algunos cambian debido a que el sistema operativo puede reproducir los mismos de manera más fácil, o bien la forma de guardar el archivo es diferente, de la gran variedad de formatos digitales de sonido, en el presente trabajo escrito solo nos interesan 2, los cuales son:

- Archivos .ogg
- Archivos .wav

A continuación se dará una explicación mucho más detallada de cada uno de los formatos digitales de sonido, sus ventajas, desventajas y usos.

6.5.1. Archivos .ogg

Ogg o Vorbis dependiendo de como se conozca, [12] es el mismo acrónimo para un formato digital de audio creciente en la industria, este tipo de formato es mayormente utilizado para la distribución de música en línea. Sabiendo que mp3 sigue siendo el formato de audio más utilizado tenemos que otros tipos de formatos tal como el ogg han surgido últimamente dándole competencia a los otros formatos, una ventaja muy grande que tiene ogg es que no tiene licencia, lo cual hace que su uso sea gratuito tanto como para generación de audio. Un usuario particular no paga por utilizar archivos mp3, pero ya las plataformas mucho más grandes la licencia para la utilización de archivos de este formato puede llegar a tornarse un costo considerable, por lo que los formatos como ogg que no tienen licencia surgen como competidor por esta razón.

Otra característica interesante del formato Vorbis es que el decodificador de referencia está basado en matemática de punto flotante, aunque haya implementaciones que utilicen punto fijo. También el formato de audio ogg a diferencia de otros, tiene en su flujo de datos ciertas estructuras de datos las cuales permiten que se aumente el uso dinámico de memoria. Estas características mencionadas anteriormente, muchas veces hacen que este formato no sea tan utilizado ya que su uso no es tan común y también influye el poco tiempo que este formato tiene en el mercado a comparación de otros.

6.5.2. Archivos . wav

Este es un formato digital [13] de audio desarrollado especialmente para ser reproducido por varios sistemas operativos con relativa facilidad, este está basado en el formato Microsoft WAVE , una característica de este formato es que se tiene la información más básica para que el archivo se reproduzca de la manera más sencilla posible.

Es un formato de audio basado en fragmentos, el cual está basado en RIFF (por sus siglas en inglés), lo cual significa Formato de archivo de intercambio de recursos. Este formato de audio está compuesto por varios fragmentos, estos fragmentos son:

- Identificador de carácter
- La longitud del fragmento
- Y la información del fragmento

Seguidamente tiene un header RIFF el tipo de formato WAVE, seguido por otros fragmentos. En los siguientes fragmentos siempre se debe de tener el dato de extensión de audio de transmisión el cual contiene la megadata correspondiente para el intercambio de datos ente transmisores. Cabe añadir que este tipo de formato de audio puede reproducir el siguiente tipo de tipos de reproducción de audio:

- Mono
- Estéreo

- Estéreo conjunto
- Doble canal

CAPÍTULO 7

Implementación de Generador de Pulsos Binaurales en Raspberry Pi 3

En la tesis de José Pablo Muñoz [1] se logró generar pulsos binaurales satisfactoriamente desde un programa de Python, en el cual se variaban distintos parámetros para poder determinar la frecuencia que los pulsos binaurales generados iban a tener. Derivado de estos resultados satisfactorios, en este trabajo se migrará este programa a una Raspberry Pi y se verificará el funcionamiento del programa ya migrado a la Raspberry Pi 3.

7.1. Familiarización con Python en Raspberry Pi 3

7.1.1. Raspberry Pi

La Raspberry Pi (RPi) es un dispositivo lanzado en el año 2012, el cual busca ser una computadora de bajo costo. Con solo añadirle periféricos de entrada como teclado, mouse, y periféricos de salida como el monitor, se puede usar como una computadora común. La RPi corre sistemas operativos basados en Linux, por ejemplo, Raspbian. Una de las ventajas de estos dispositivos es su precio debido a que es bastante bajo a comparación de computadoras portátiles laptops o de escritorio.

Muchas veces se refieren a dispositivos como la RPi como *Single Board Computer* (SBC). Aunque algunas de estas SBC ya estuvieran disponibles en el mercado, no habían tenido el impacto que el RPi tuvo al ser lanzado, debido a que eran utilizadas para desarrollos industriales.

La RPi cuenta con los protocolos de comunicación y características como USB, UART, SPI, I^2C, e interrupciones.

Debido al incremento de la popularidad de distintas áreas de la tecnología, tales como

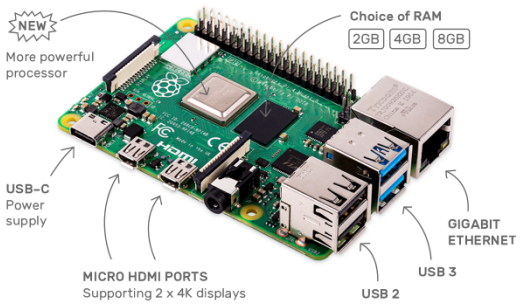

Figura 11: Fotografía Raspberry Pi obtenido de: [14]

Cyber Physical Systems (CPS) e *Internet of Things* (IoT), el mercado para dispositivos como la RPi se ha ampliado de una manera exponencial [15].

7.1.2. Python

La Raspberry Pi, tiene incorporados varios modulos para poder correr programas en Python en la misma, estos son:

- Python (IDLE)
- Thonny Python IDE

En este caso se utilizó Thonny, ya que es un IDE en el cual podemos correr el programa de una manera más sencilla que en el IDLE propio de Python .

Hay una cosa muy importante al momento de migrar este tipo de programas a la Raspberry Pi, los cuales son, que este dispositivo, puede no tener todas las librerias que se necesitan para que el programa funcione eficientemente. Debido a esto se encuentran algunos errores al momento de implementación, que con la investigación debida se van solucionando.

7.2. Migración de programa

A grandes rasgos el algortimo es bastante similar al desarrollado por José Pablo Muñoz en [1] ya que esta tesis se enfoca mayormente en el desarrollo de la aplicación funcional en la raspberry Pi. Para el generador se tuvieron que utilizar las siguientes librerias de Ppython:

- numpy
- scipy

Que son dos librerias que son utilizadas para realizar distintas operaciones dentro del código. Numpy es utilizada para las operaciones matemáticas y scipy es utilizada para la generación de un archivo wav.

Ya habiendo instalado nuestras librerias correspondientes en la Raspberry Pi, habiendo corroborado esto en "Manage Packages"de Thonny, se dió la tarea de generar audios con este código migrado, obteniendo los siguientes resultados:

```
>>> binauralgenerator(200,44100,20,10,0)
('BinauralFrecuenciaCentral200.0FrecuenciaPulso20.0.wav', array([[ 0.        ,  0.        ],
       [ 0.02855605,  0.03141076],
       [ 0.05708881,  0.06279052],
       ...,
       [-0.19859047, -0.09410831],
       [-0.22649677, -0.06279052],
       [-0.25421834, -0.03141076]], dtype=float32))
```

Figura 12: Salida generador de Pulsos Binaurales

En este caso se evaluó la función llamada *binauralgenerator* con ciertos parámetros para observar si había una salida correspondiente, al correr el programa , obteniendo que se generaba un archivo como este:

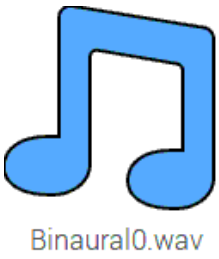

Figura 13: Archivo .wav

Habiendo generado satisfactoriamente un archivo, se corroboré que el mismo se oyera de manera correcta, antes de hacer esto cabe mencionar que hay que forzar la salida de audio que sea por audífonos, porque de lo contrario la Raspberry puede estar transmitiendo por la salida HDMI. Tomando en cuenta lo anterior se comprobó que el audio estuviese sonando de manera correcta, esto de primero con el programa de VLC que viene incorporado al sistema operativo, obteniendo un resultado satisfactorio, se escuchaba un tono distinto en cada uno de los canales de los audífonos utilizados. A la espera de una verificación más exhaustiva se da un visto bueno a la migración del programa de Python a la Raspberry Pi.

7.2.1. Problemática de reproducción de sonido entre Python y Raspberry Pi

Considerando que en la sección anterior solo se tomó en cuenta la generación de archivos de sonido utilizando Python, se dejó un poco del lado la reproducción del audio en el mismo programa de Python, ya que este proporcionó muchos problemas a la hora de reproducir los archivos generados por el programa. Esto no es un problema al azar, debido a que la librería que se está usando en este caso es:

- pygame

Esta libreria tiene un problema, no se pueden reproducir archivo wav de 32 bits, sino

que solamente de archivos de la misma denominación de 16 bits. Se corroboró la teoría con la práctica utilizando el siguiente código:

```
def playbinaural(filename):
    pygame.init()
    sonido  = pygame.mixer.Sound(filename)
    sonido.play()
```

Figura 14: Código para reproducir archivo de audio

El siguiente código utiliza la librería de pygame y la función mixer de la misma para poder reproducir archivos de audio. Haciendo pruebas, se trató de utilizar el archivo generado por el programa de *binaural generator* pero no se logró obtener una salida de audio, obteniendo el siguiente error:

```
Traceback (most recent call last):
  File "<pyshell>", line 1, in <module>
  File "/home/pi/Desktop/Tesis Luis Guerrero/ArchivosPrueba/GeneradoryReproductordePulsosBinaurales.py", line 97, in playbinaural
    sonido  = pygame.mixer.Sound(filename)
pygame.error: Unable to open file 'Binaural0.wav'
```

Figura 15: Error al utilizar archivo .wav

Seguidamente se utilizó un audio de prueba obtenido de internet, y este si pude ser reproducido de manera correcta.

Un problema en general de utilizar archivos wav en Debian, sistema operativo de la Raspberry Pi, es que no son el archivo que mejor maneja el sistema operativo, en cambio este maneja de mejor manera, los archivos multimedia OGG, tomando en cuenta esto, se hizo una conversión desde una página web, y se hizo la prueba, obteniendo resultados satisfactorios, la solución del problema es utilizar este tipo de archivos para que sean reproducidos con mayor facilidad en Python. El problema es que no se ha hallado una manera desde Python para hacer una conversión de archivo wav a OGG o directamente una generación directa de un archivo OGG, esto se resolverá en la etapa de la realización de interfaz gráfica.

7.2.2. Resolución problemas de reproducción de sonido

Como se mencionó anteriormente, no se encontró un método útil para poder reproducir pulsos binaurales, este no es un problema tanto del usuario final, como del desarrollador, ya que para el usuario final se le debe de dar un rango más pequeño, pero debido a que se está estudiando el efecto de los pulsos en las personas, es estudiará la ma,yoría del rango audible, por lo cual no es eficiente tener audios pregrabados. Como se mencionó la librería de pygame no reproduce archivos wav, entonces no se estaba podeiendo reproducir los sonidos que eran deseados.

Se pensaron muchas soluciones, una de las cuales fue la utilización remota del programa integrado en la Raspberry para reproducir audio llamado *VLC media player* el cual es capaz de convertir a distintos formatos de audio,como cualquier otro programa de audio. Para la aplicación de este proyecto se necesitaba operar la aplicación externa desde Python, lo

cual resultó muy tedioso y complicado, debido a esto se encontró otra manera de resolver el problema.

La solución anterior fue planteada ya que luego de exhaustiva investigación, no se encontraba una librería específica de python, la cual pudiese. convertir archivos a formato OGG, que como se mencionó anteriormente, son el tipo de archivos que la librería de reproducción soportaba, y debido a que no se encontraba una manera de guardar los archivos en un formato que no fuese wav, no se tenía ninguna otra solución. Luego de haber fallado implementando la solución que empleaba la utilización de *VLC media player* se investigó aún más y finalmente se encontró la librería pydub, que permite la conversión entre formatos de audio.

El primer paso fue generar los archivos con formato OGG, para luego utilizando ahora sí *VLC media player* para comrpobar, que los archivos fuesen realmente convertidos y que se pudieran reproducir por una herramienta externa que no fuera Python:

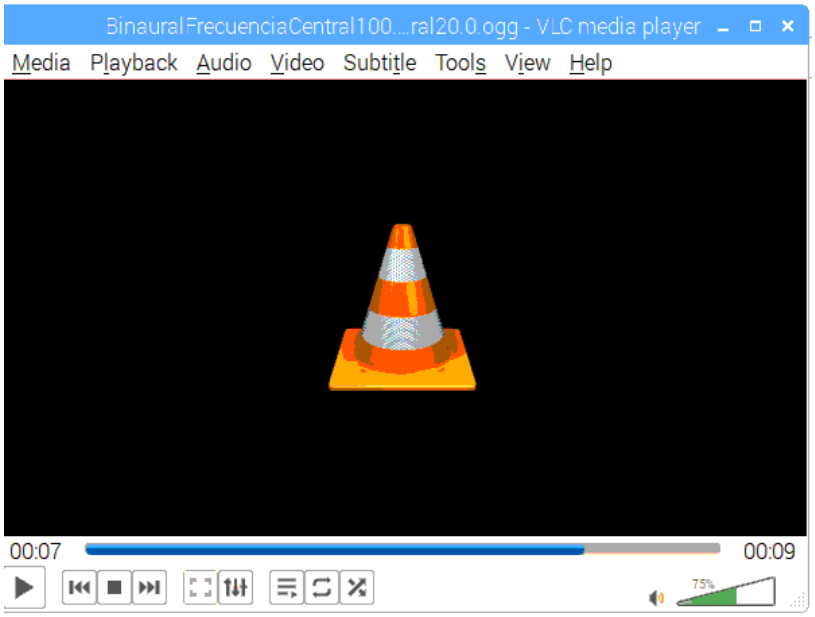

Figura 16: Prueba de reproducción con VLC Media Player

Habiendo obtenido un resultado satisfactorio en la prueba de sonido con un reproductor externo, ya se pudieron realizar pruebas pertinentes para finalmente verificar el funcionamiento de la librería pygame para reproducir los archivos generados, y efectivamente se verificó que los archivos OGG generados, podían ser reproducidos por dicha librería.

Es muy importante mencionar el procedimiento que se llevo a cabo fue el de una conversión entre formatos de audio, no está generando los archivos como OGG, desde la etapa del generador de pulsos binaurales, sino que, se está generando un archivo wav, el cual puede ser generado por el algoritmo desarrollado en la tesis [1] , y al momento que este archivo es generado, esté se ubica en un fólder en específico, el cual tiene almacenados todos los archivos wav generados. Este archivo queda guardado con ciertos parámetros, tanto la frecuencia central de los pulsos como la frecuencia de pulso binaural, cuando el archivo .wav es guarado, su nombre incluye los parámetros anteriormente mencionados. Luego para la

conversión de formato OGG a wav el comando utilizado propio de pydub, busca el archivo con las características que se acaban de guardar, y luego lo convierte, utilizando un método propio de la librería, obteniendo esto:

Figura 17: Archivo OGG generado

Esta información que contiene el nombre del archivo es muy importante, debido a que luego en desarrollos posteriores de esta tesis es más fácil identificar las carácterísticas del audio si el nombre tiene la forma indicada.

7.3. Evaluación funcionamiento de programa

Luego de haber logrado la generación del pulso binaural deseado, nos encontramos con la problemática de la verificación del buen funcionamiento del programa. debido a esto se tuvo que realizar un análisis en el espectro de frecuencias en Matlab. Esto debido a que los análisis con respecto al tiempo no son útiles en este caso ya que los mismos solo nos mostrarían como es la amplitud del pulso binaural (que es representada como una onda sonora).

El primer paso de la verificación fue generar un pulso binaural con el programa ya migrado, el cual tiene las siguientes características:

- Frecuencia de los pulsos binaurales generados: 20 Hz
- Duración del pulso binaural: 10s
- Frecuancia central de los pulsos binaurales: 200 Hz

Se realizó un código simple en Maltab, en el cual de primero se verifica, que el sonido tenga dos canales, el izquierdo y el derecho, lo cual es muy importante para esta tesis, ya que esta depende de sobremanera de la diferencia de frecuencia que haya entre ambos canales, para que el cerebro interprete estos sonidos como un pulso binaural. El proceso de ïnterpretación"que tiene el programa para detectar al pulso binaural es que resta el canal que tiene una frecuencia más alta contra el que tiene la frecuencia más baja, obteniendo así el siguiente resultado:

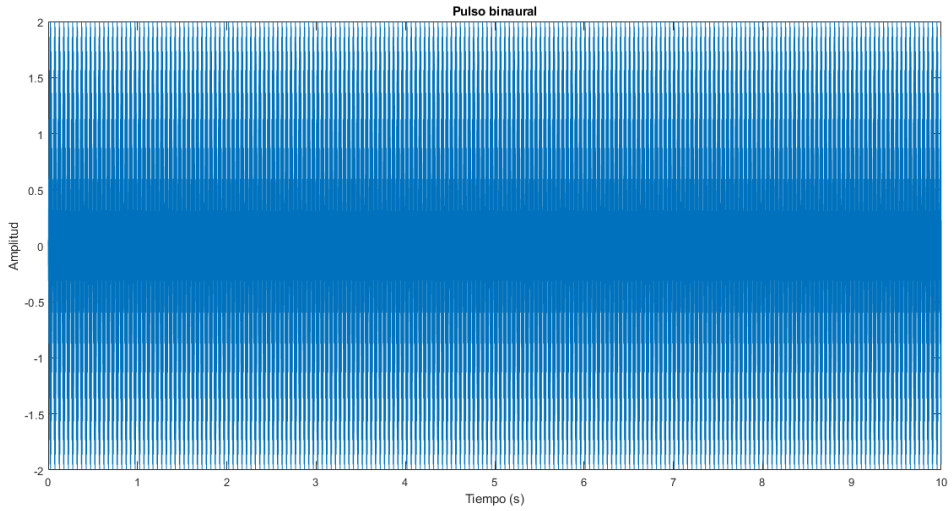

Figura 18: Pulso Binaural obtenido

Como se aprecia en la figura anterior, el programa está generando de manera correcta el tiempo del pulso, el cual se determinó que fuese de 10 s, cabe mencionar que en Matlab para ajustar este tipo de mediciones hay que hacer algunos cambios sencillos, ya que generalmente cuando se gráfica archivos de sonido, o cualquier archivo que tenga una frecuencia de muestreo, antes de hacer un ajuste, la gráfica está definida por el número de muestras, lo que se debe de realizar para tener un buen marco de referencia en segundos y no por muestras es demostrado en el siguiente código de Matlab:

```
%Definimos el pulso binaural que se interpretaría
binaural_beat= right_channel-left_channel;
%Definimos t, dependiendo del tamaño de nuestro archivo wav y la frecuencia
%de muestreo fs
t= (0:441000-1) / fs;
plot(t,binaural_beat);
title('Pulso binaural');
xlabel('Tiempo (s)')
ylabel('Amplitud')
```

Figura 19: Código para análisis en el tiempo

Que a grandes rasgos lo que hace es tomar en cuenta la frecuencia de muestreo para poder dividir esta por la cantidad de muestras, obteniendo así la cantidad de segundos que en este caso el archivo .wav de sonido tiene.

Para apreciar de mejor manera la imagen, podemos ver que la onda sonora que el cerebro interpreta como pulso binaural es de la siguiente manera:

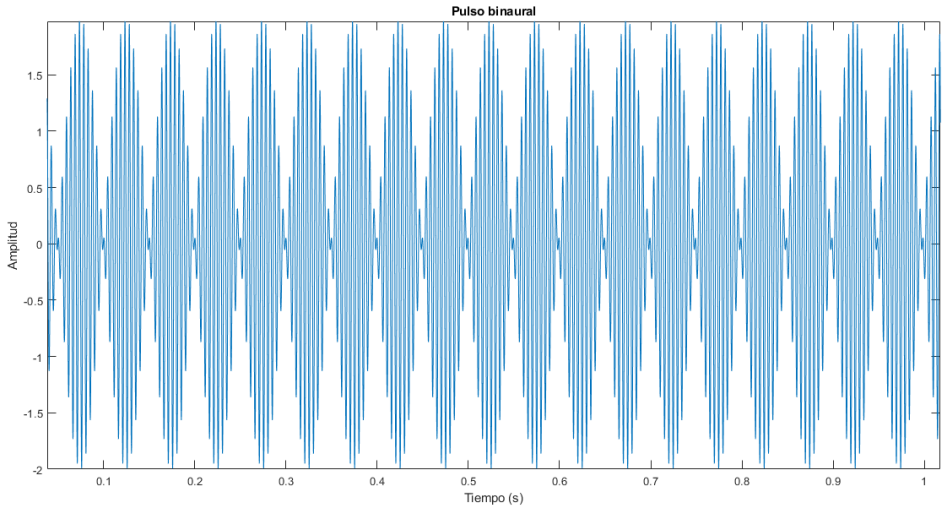

Figura 20: Pulso Binaural obtenido (1 s)

Aunque se puede apreciar de cierta manera el sonido y como se comporta con respecto al tiempo, no su puede tomar conclusiones con respecto al buen funcionamiento del algoritmo. Debido a esto debemos de realizar un análisis en el espectro de frecuencias, el cual nos dé como resultado la frecuencia central de nuestro pulso binaural, y la diferencia de frecuencia entre los canales. Esto se realizó utilizando la fft en Matlab, haciendo algunos ajustes que se lograrán apreciar a continuación.

Hay un método en Matlab, que es utilizado especialmente en la FFT (Fast Fourier Transform), para poder obtener la frecuencia en Hz para nuestro cálculo en el espectro de frecuencias, que es llamado *Single-sided amplitude spectrum* , el cual mediante el siguiente procedimiento, nos ayuda a llegar a una escala en Hz deseada.

```
%Definimos el pulso binaural que se interpretaría
binaural_beat= right_channel-left_channel;
Fs = 44100;                    % Frecuencia de muestreo
T = 1/Fs;                      % Periodo de la señal
L = length(y);                 % Tamaño de la señal
t = (0:L-1)*T;                 % Vector de Tiempo
NFFT = 2^nextpow2(L);
Y = fft(binaural_beat,NFFT)/L;
f = Fs/2*linspace(0,1,NFFT/2+1);

% Graficamos espectro de frecuencias single-sided
plot(f,2*abs(Y(1:NFFT/2+1)))
title('Análisis de Espectro de frecuencias canal izquierdo')
xlabel('Frecuencia (Hz)')
ylabel('|Y(f)|')
```

Figura 21: Código Análisis de Espectro de frecuencias

Ya teniendo este código, se realizó la medición tanto de ambos canales por separado, tanto como de la resta de ambos, obteniendo los siguientes resultados:

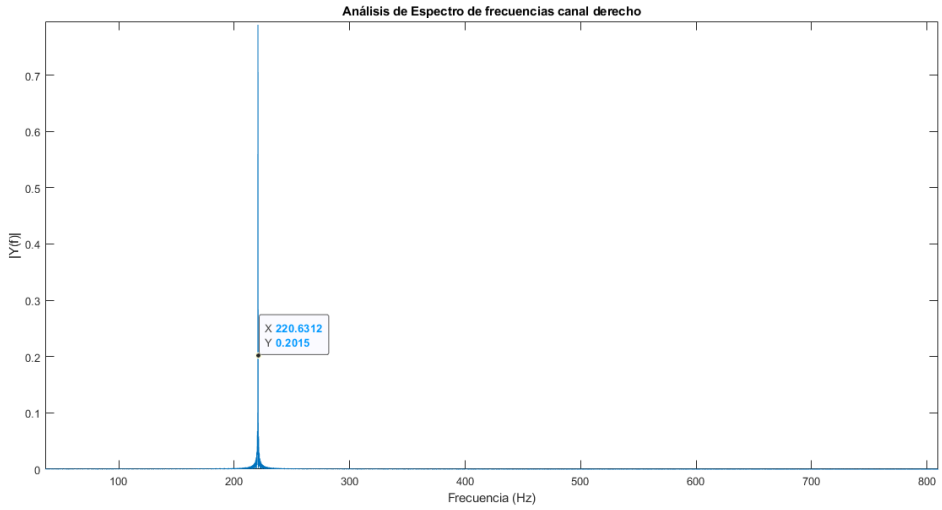

Figura 22: Análisis Espectro de Frecuencias Canal Derecho

En la figura anterior se nota que el espectro de frecuencias para el canal derecho se encuentra en los 220 Hz.

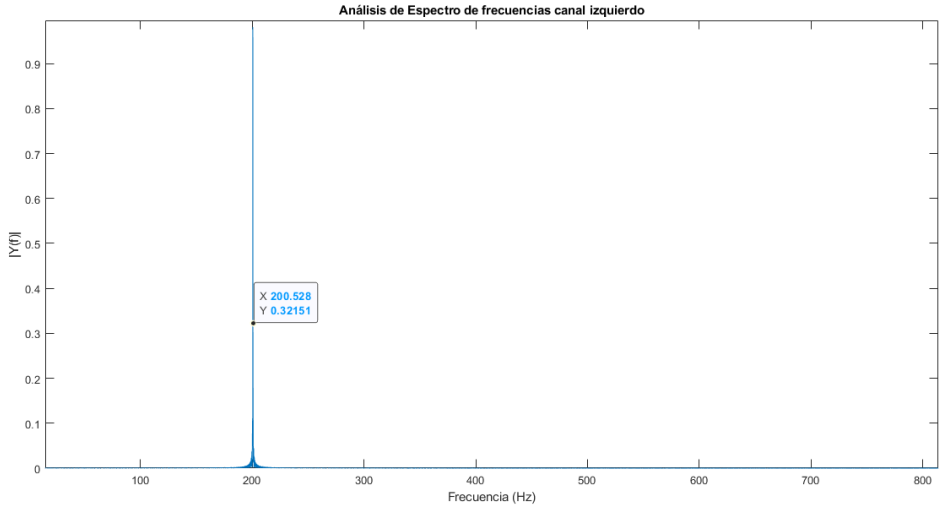

Figura 23: Análisis Espectro de Frecuencias Canal Izquierdo

Analizando la gráfica del canal izquierdo, se aprecia que el espectro de frecuencia del canal, es de 200 Hz, considerando esto se puede corroborar el buen funcionamiento del programa tanto para generar tanto la frecuencia central del audio, que en este caso son 200 Hz, y la diferencia entre ambos canales, que en este caso debería ser 20 Hz, que de igual manera se cumple.

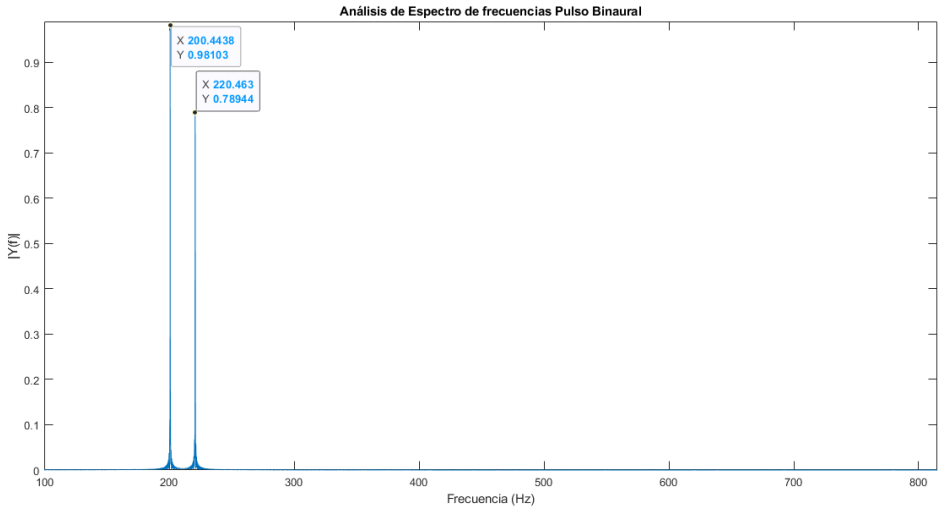

Figura 24: Análisis Espectro de Frecuencias Canal Pulso Binaural

Para finalizar con la verificación se analiza también el espectro de frecuencias del pulso binaural generado, en este caso se verifica el buen funcionamiento de los dos canales en conjunto, obteniendo los mismos valores de frecuencia que anteriormente se encontraron de manera separa.

Con todas estas verificaciones podemos concluir que el programa está generando los archivos wav de manera correcta, con esto se termina la verificación del funcionamiento correcto del generador de pulsos binaurales en la Raspberry Pi.

CAPÍTULO 8

Interfaz Gráfica

Uno de los objetivos más importantes de este trabajo, es el desarrollo de una interfaz gráfica fácil de utilizar para el usuario, ya que los que utilizarían esta aplicación no son los desarrolladores (a gran escala por supuesto). Debido a esto se utilizaron ciertas herramientas, que ayudaron al desarrollo de la interfaz gráfica, de los cuales se dará más detalles a continuación.

En cuanto a la interfaz gráfica obtenida, se puede verificar tanto la funcionalidad de la misma, como estética, acorde a las limitaciones de las herramientas utilizadas, para desarrollar la misma.

8.1. Herramientas utilizadas para la Interfaz Gráfica

8.1.1. Pantalla Táctil 7"

Algo muy importante que se discutió con el asesor, fue la dificultad de utilizar la Raspberry Pi con un monitor para la utilización del programa, debido a que esto sería lo mismo que haberlo implementado en una computadora portátil, a lo cual se solucionó utilizando una pantalla LCD que es compatible con la Raspberry Pi. Esta pantalla es fabricada por la misma empresa que diseña la Rasberry Pi.

Sin querer ahondar en aspectos técnicos de la Pantalla, es un dispositivo que es similar al funcionamiento un monitor conectado por HDMI, a la Rasberry Pi. Esta pantalla se puede controlar con los periféricos de entrada convencionales, siendo estos el teclado y el mouse. Y debido a que es una pantalla táctil, se puede interactuar con la Rasberry Pi por medio de tacto con la pantalla, de la misma manera se utiliza un case donde quedan descubiertos todos los puertos de la Rasberry Pi, para obtener un *setup* más estético.

Conexiones Pantalla Táctil 7”

Las conexiones de la pantallas soon bastantes sencillas, estas solamente son conexiones de 4 jumpers de la pantalla LED a la Raspberry P, y un *Ribbon Cable* o cable de multiples hilos que va a la entrada de este tipo en la raspberry Pi, como se muestra en las siguientes fotos.

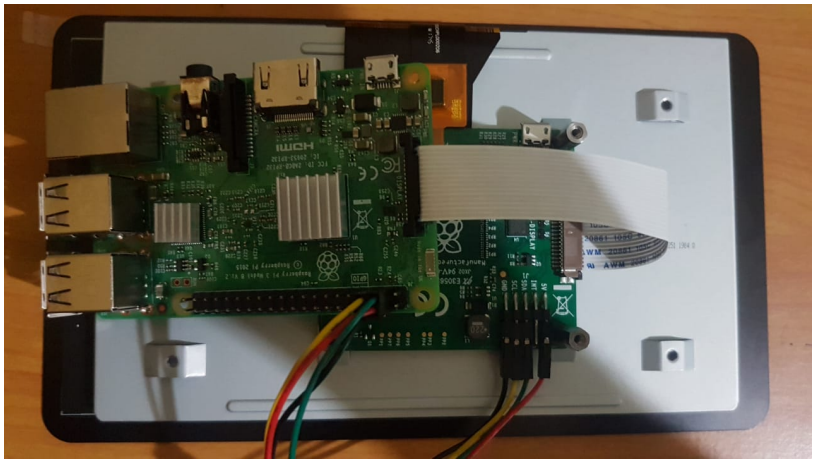

Figura 25: Conexiones de Raspberry Pi a Pantalla Táctil de 7”

Cabe mencionar que se utilizó un case para que la pantalla tuviera un apecto más estético.

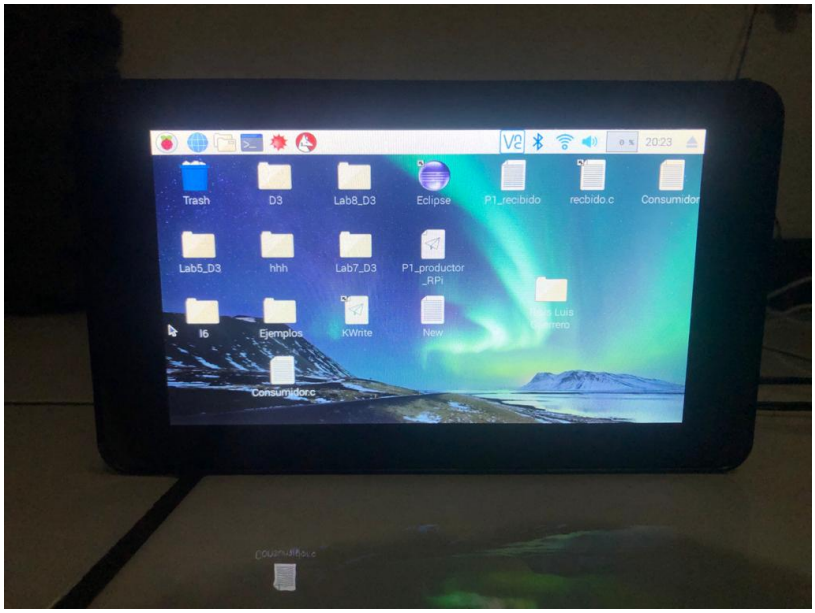

Figura 26: Vista de Frente Sistema Pantalla Táctil-Raspberry

8.2. Tkinter

Como se mencionó anteriormente, la interfaz gráfica es una parte importante de esta teses, luego de investigar sobre distintos paquetes que son compatibles para realizar interfaces gráficas, se llegó a que se utilizaría Tkinter que es un paquete incluido dentro de Python (en este caso se tuvo que instalar las librerías correspondientes en la Rasberry Pi). Es el generador de interfaces gráficas por defecto, y por esa misma razón se eligió la utilización de esta librería, debido a que hay mucha más documentación de esta librería que de otras, tales como Kivy (Otra librería de para generación de interfaz gráfica compatible con Python). Además la utilización de esta libre es simple y aunque no deje estilizar mucho los botones y los inputs en general, siempre se pueden utilizar imágenes, lo cual es algo muy bueno, ya que provoca que esta librería de Tkinter sea tanto funcional como bastante estética, lo cual se irá mostrando en las secciones posteriores de este capítulo.

8.3. Desarrollo de Interfaz Gráfica

Considerando todas las herramienta mencionadas anteriormente, se realizó el diseño de la interfaz gráfica, debido a que por separado se habían probado los módulos correspondientes,los cuales eran:

- Generación de pulsos binaurales
- Reproducción de pulsos binaurales

El siguiente paso a seguir era la implementación de la interfaz gráfica con todos los componentes mencionados anteriormente, y algunas características que hicieran a la misma, amigable con el usuario. La primera versión de la interfaz gráfica (de la cual es la que se está discutiendo en la presente sección) no tiene todas las características al 100 por ciento, pero es totalmente funcional, y cumple con las características antes mencionadas.

8.3.1. Creación de etiquetas y botones

Algo muy interesante que surgió al momento de realizar la interfaz gráfica es que Tkinter no tiene incorporado en si una manera muy eficiente de estilizar los distintos componentes que se pueden generar, tales como:

- Etiquetas
- Botones
- etc

Debido a esto se tienen que exportar imagenes las cuales cumplirán la función de estilizar la interfaz gráfica, algo raro de Tkinter es que no detecta si una imagen tiene transparencia,

lo que significa esto es que si una imagen tiene algún lugar el que sea transparente, cuando se añade a la ventana, no lo reconoce, así que al momento de exportar imágenes, estas deben de tener el color de fondo que tenga predeterminado la ventana. Se iba a utilizar un fondo distinto (una foto en específico), pero a esta problemática, se optó por un diseño más simple en el cual el color de fondo fuera un color sólido y no una imagen.

8.3.2. Multiples ventanas en Tkinter

Como se mencionó anteriormente se está utilizando la librería de Tkinter propia de Python para el desarrollo de la interfaz gráfica. La idea inicial del desarrollador era realizar las distintas páginas de la aplicación, y luego ejecutar las demás dependiendo de la página que se quisiera utilizar. Esta manera de realizar la aplicación no es muy inteligente, ya que uno de los objetivos nos restringe a que la aplicación pueda realizar múltiples subprocesos de forma paralela, entonces, no se estaría cumpliendo con este requerimiento.

En el presente trabajo se tomó la mala decisión de realizarlo de la manera anteriormente mencionada, pero no solamente el objetivo mencionado no sería cumplido, sino que la implementación se torno un tanto complicada, al punto en el que se estaban corriendo programas de Python desde otras programas, como ejecutables, esto no solamente no cumple con las restricciones de los objetivos sino que no es una manera correcta de programar en Python, debido a que si se va a utilizar un programa externo al actual, se deben de importar sus clases.

Debido a esto se tuvo que buscar otra solución a esta problemática, debido a que si se estaba desarrollando cada ventana de la aplicación por separado, pero no se podían integrar entre sí, debido a esto se encontró la solución de los *Frames* en Tkinter , la traducción directa de *frames* al español, es ventanas, entonces lo que se lograba con estas ventanas, es que todas se corren paralelamente, pero hay un método el cual logra que alguna estas sea la que se despliegue, siendo este, el método *showframes* , el cual dependiendo de la entrada, despliega la ventana correspondiente, con eso, se logro algo muy importante que fue la implementación de todas las ventanas, el prototipo es el siguiente:

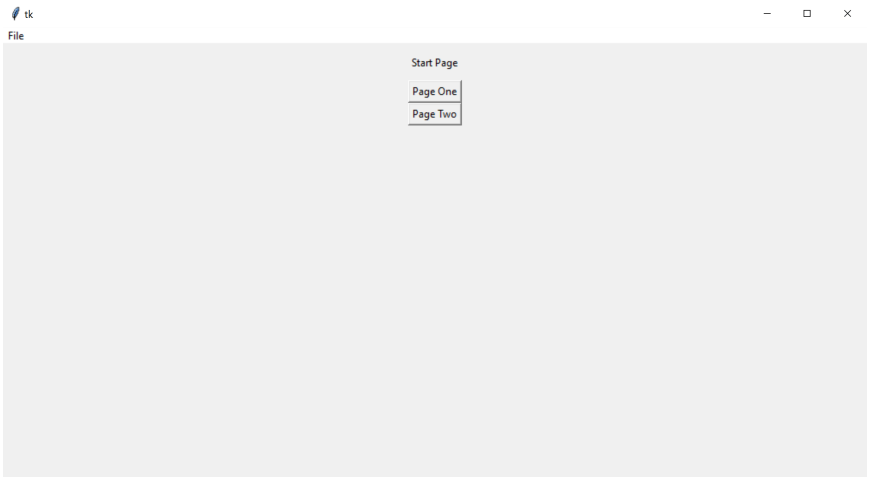

Figura 27: Ventana principal prototipo

Como se logra apreciar, hay una pagina principal que se despliega al momento de correr la aplicación, y de esta forma se puede recorrer las múltiples ventanas de la aplicación, como por ejemplo:

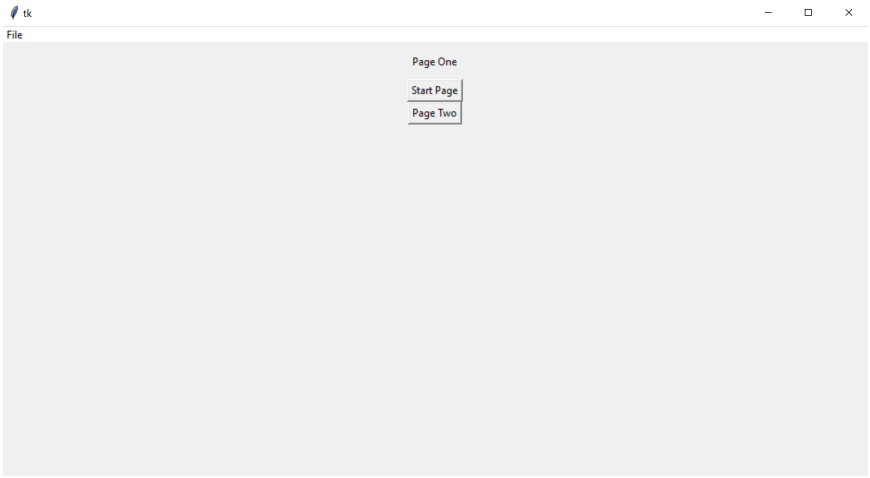

Figura 28: Ventana secundaria prototipo

Con estos métodos descritos anteriormente ya se puede implementar una aplicación en si, porque lo que se intentó realizar al principio fue la implementación de varias aplicaciones y la conexión entre ellas, este método es mucho más sencillo. Cabe añadir que todos las ventanas son del mismo tamaño utilizando este método, se puede estilizar todo lo demás, pero esta variable permanece constante para todas las ventanas.

8.3.3. Características generales de las ventanas

Todas las ventanas de la aplicación tienen algunas características generales, estas son:

- Ajuste de Brillo
- Botón de apagado
- Botón de regreso

Los cuales se pueden distinguir claramente:

Ajuste de Brillo

Debido a que el dispositivo está pensado para que sea utilizado al momento de dormir (en la noche) se tiene que tener una capacidad de ajustar el brillo de la pantalla para que no sea molesto al momento de utilizarlo, este botón cumple esta función.

Figura 29: Botones principales prototipo

Botón de apagado

En todo momento es posible terminar la aplicación desde cualquiera de las ventanas, este botón se encuentra en la esquina superior derecha a un lado de l ajuste de brillo.

Botón de regreso

Excluyendo la página de inicio, todas las páginas tienen la opción de regresar a la página anterior, este botón se encuentra en la esquina inferior izquierda.

Estas son las funcionalidades generales de la interfaz gráfica, que logran tanto facilidad de navegación como adaptabilidad de la aplicación. Cabe añadir que no se agregó un botón de sonido, ya que eso se puede modificar en las opciones de la Raspberry, así que sería un tanto redundante la utilización del mismo.

8.3.4. Problemática tamaño pantalla LCD

Algo importante que hay que notar, es que toda la interfaz gráfica se estaba programando en un monitor convencional, pero el usuario debería de poder utilizar la pantalla táctil mencionada anteriormente para poder navegar por la aplicación, pero esta pantalla táctil tiene dimensiones mucho menores a las de un monitor convencional, por lo cual se tuvo que hacer el cambio, no solo del tamaño de la ventana que se despliega como de las distintas etiquetas y botones que ya se habían posicionado anteriormente a encontrar este error.

8.3.5. Página de Inicio

La siguiente página, es la primera ventana que se muestra al momento de ingresar la aplicación, recordando el método antes mencionado de *showFrames* hay una opción para poder dejar una página predeterminada, en este caso la de inicio, esta página no tiene mucha

funcionalidad, debido a que la misma solo es la presentación de la aplicación al usuario. A continuación tenemos una imagen de la página de inicio:

Figura 30: Ventana principal de Interfaz Gráfica

Para esta ventana se utilizaron las siguientes librerías:

- Tkinter
- PIL

Esta página tiene algunas diferencias con las demás, una de las cuales es que no hay botón de regreso, lo cual tienen sentido porque es la página principal, seguidamente el botón de *play* es el botón que hace que otra ventana se despliegue, siendo este el de Submenu.

En este caso la ventana actual no tiene muchas funcionalidades pero se describirá la manera de hacer cambios entre ventanas:

En el primer paso, causado porque Tkinter no tiene una gran capacidad de modificación de botones, se guarda una variable con un una imagen específica, en este caso el botón de *play*

Seguidamente se asigna el tipo de variable *button*, para tener la posibilidad de generar una acción cuando este sea presionado, también está variable de botón incluye la variable de imagen anteriormente mencionada para tener la forma y color de la imagen anteriormente especificada. Y en la misma instancia del botón asignamos que ocurra una acción cuando sea presionado, la cual es el método *showFrames*.

Finalmente se hace asigna una posición al botón. Esta es la manera de utilizar botones, se pueden utilizar otras funciones, en este caso solo se menciona a *showFrames* porque es el caso, pero se puede utilizar otra función o método al momento de llamar a la misma., por ejemplo si se desea utilizar la función apagar, en vez de colocar *showFrames()* se escribe apagar() y sucederá lo que la función este programada para hacer.

Para las imágenes que no son botones, el procedimiento es el mismo, solo que en ese caso se utiliza el tipo de variable propio de Tkinter *label* en lugar de *button*, y debido a que no tienen una acción al ser presionadas, solo deben de ser ubicadas en la ventana.

Debido a la utilización de Tkinter, para poder desplegar las imagenes correctamente, las variables correspondientes a las mismas, deben de ser globales, se investigó un tanto acerca del tema, pero no se pudo encontrar la razón específica de esto.

8.3.6. Sub Menu

Para esta ventana se utilizaron las siguientes librerías:

- Tkinter
- PIL

Esta ventana tiene la funcionalidad, de poder navegar a todas las ventanas, es un paso obligado para poder generar y reproducir pulsos binaurales, tiene el mismo método anteriormente mencionada para mostrar ventanas. La ventana de SubMenu se ve de la siguiente manera:

Figura 31: Sub Menu de Interfaz Gráfica

Los botones son explícitos en este caso, en el cual si se quiere generar un audio de presiona el botón de GENERAR AUDIO, de la misma manera para REPRODUCIR AUDIO. Tiene las funcionalidades generales de ventana mencionadas en secciones anteriores, y los botones son instanciados de la misma manera.

8.3.7. Generar Audio

Debido a que el fin de esta tesis es poder reproducir cualquier pulso binaural (en esta etapa) para poder determinar posteriormente el rango de frecuencias que induce positivamente al sueño, se debe tener la habilidad, de generar cualquier pulso binaural, tanto de frecuencia central, como frecuencia de pulso binaural para poder recopilar estos datos, para lograr lo antes mencionados, se implementó la siguiente ventana:

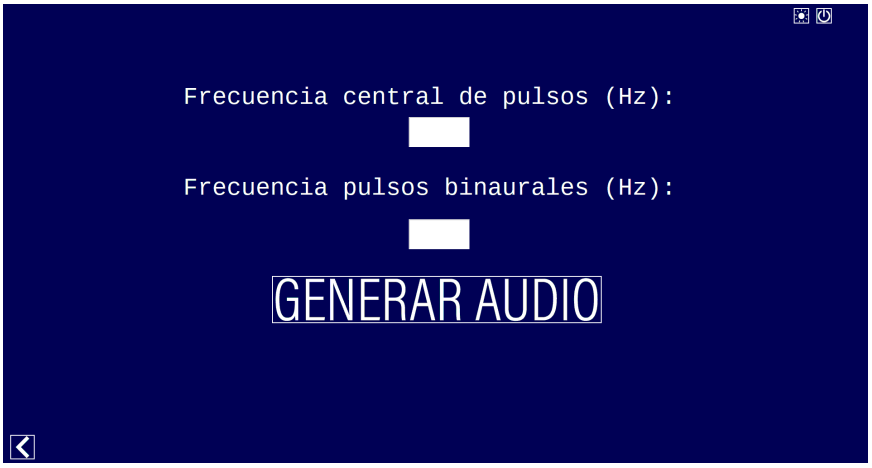

Figura 32: Generar Audio de Interfaz Gráfica

Para esta ventana se utilizaron las siguientes librerías:

- Tkinter
- PIL
- Tkinter File Dialog
- Scipy.io.wavfile
- Pydub

La primera diferencia que tiene esta página con respecto a las demás son las entradas de texto, que definen:

- Frecuencia central de los pulsos
- Frecuencia de pulso binaural

De esta definición surge algo muy interesante, en este tipo de paginas donde se necesita que el valor de una variable en una clase , sea utilizado en un método posteriormente hay que utilizar el comando *self* de Python, debido a que estando dentro de la misma clase, si se tiene un método el cual se debe de utilizar, la variable no está definida en ese método, por lo que es necesario utilizar esta manera de acceder a las variables, de lo contrario el programa tendrá errores, debido a que la variable es diferente dentro del método que fuera del método.

También se debe recordar que cuando se ingresan caracteres a los cuadros de texto, este tipo de variable es de Cadena de caracteres, o en inglés tipo *String*, por lo cual al momento de hacer operaciones matemáticas con estas variables, debe de hacerse una conversión, ya sea a una variable tipo entero, o bien a una variable tipo flotante. Un problema con respecto al ingreso de datos fue muy notorio, el cual se pudo resolver pero no se encontró una explicación

especifica. El problema es que la forma de utilizar la variable que está dentro de la entrada de texto, es llamando a toda la entrada de texto, debido a esto se tuvo que hacer una variable global, la cual tuviera como valor la totalidad de la entrada de texto para poder ser utilizada correctamente en el código. Se sospecha que por ser un tipo de objeto, no puede ser llamado con la instancia *self* que se utilizó para poder llamar a otro tipo de variables.

Se utilizó como base el programa generado por José Pablo Muñoz en la tesis que precede al presente trabajo, con la diferencia que se utilizaron variables de las entradas de texto correspondientes. Seguidamente se guardan los archivos wav en una carpeta, y el mismo programa automáticamente los convierte a archivos OGG, para su fácil reproducción en Python. Los archivos son guardados en carpetas específicas, las cuales pueden ser accesadas desde otras ventanas del programa.

Cabe mencionar que para esta generación de sonidos, el tiempo de todos los sonidos generados es de segundos, el motivo de esto es debido a que entre más tiempo de sonido sea requerido, ocuparemos mucha memoria, por ejemplo no podríamos almacenar una gran variedad de sonidos de distintas duraciones, aproximando el tiempo de sueño a unas 7 horas, por lo cual se ideó que todos los sonidos tuviesen la misma duración, con la diferencia, que este sonido se repetiría n cantidad de veces dependiendo del tiempo de reproducción. Esto resuelve tanto problema de memoria, la cual es limitada (aunque es muy grande no es eficiente tener muchos archivos de 7 horas, cuando uno de 10 segundos puede realizar la misma función), y también resuelve el problema de tiempo de generación, el cul sería absurdo si fuese un archivo de 7 horas, debido a que el algoritmo logra generar el sonido en aproximadamente el mismo tiempo que este dura.

Al momento que el archivo de audio se genera, se le da un nombre en clave, el cual describe las características del pulso, ya que luego de ser guardados como archivo OGG, son localizados en una carpeta en específico en la cual debe de ser posible identificar las características de los audios para elegir el correspondiente.

8.3.8. Reproducir Audio

La presente ventana tiene muchas funcionalidades que se tuvieron que implementar para que funcionara de la mejor manera, posible. Esta ventana a grandes rasgos cumple la función de reproducir audio, elegir audio, elegir tiempo de reproducción, la siguiente imagen presenta la ventana:

Para esta ventana se utilizaron las siguientes librerías:

- Tkinter
- PIL
- Pygame

La primera distinción de esta página, es la entrada de tiempo de reproducción, en la cual la primera versión de la Interfaz gráfica, queda libre para que el desarrollador pudiese encontrar los pulsos binaurales que propiciaran el sueño, esta parte se logra de la misma manera

Figura 33: Generar Audio de Interfaz Gráfica

que las entradas de texto de la ventana de generación de audios. Al momento de ya haber definido el tiempo de reproducción lo que prosigue es presionar el botón que dice: "Tiempo de reproducción", y de esta manera cargar un valor que será utilizado posteriormente para determinar cuantas veces se reproduce el sonido.

Seguidamente, se tiene el botón: "Tiempo de Reproducción", el cual habiendo ingresado, el tiempo de reproducción carga un valor a una variable en específico, que hará que el audio se reproduzca una cantidad de tiempo determinado. Inicialmente este procedimiento se había pensado implementarlo con un ciclo *while*, el cual repitiese el método la cantidad de veces necesarias para llegar al tiempo de reproducción, pero ya pygame (la librería de sonidos que se utilizó para la reproducción de sonidos), trae implementado en su función de reproducción que repite el sonido cierto numero de veces. Esto es una opción que tiene la librería cuando se utiliza la función *play* por defecto tiene la opción que el numero n de veces que se repita es igual a 1, entonces el sonido se repetiría una vez, en este caso 10 segundos que dura el audio, y de esta manera se logró esto. Cabe mencionar que algo útil, es poder generar un sonido por tiempo indefinido, ingresando en la función el valor de -1.

Otro botón que está presente en la ventana que se está analizando, es el botón de parar, el cual es bastante simple, lo que hace es que cuando es presionado, detiene al sonido que se está reproduciendo, y lo regresa al tiempo 0, por ejemplo si se tiene un audio que dura 10 segundos, el mismo se está reproduciendo y en el segundo 5 se presiona parar, al momento de volver a reproducir el sonido, este se reproducirá por 10 segundos, ya que no queda guardado en que momento fue detenido el sonido.

Ahora bien con el botón de pausa es un poco más complicado, debido a que debemos de considerar que si se presiona el botón de pausa se debe de guardar en que momento del audio se encuentra el usuario, en concreto esto no se hace ya que hay funciones específicas de pygame que ya implementan esto, que son pause y unpause, pero debido a que sería un poco innecesario tener solo un boton para play y otro para pausar, lo que se realizó fue que el programa tiene un estado, el cual al momento cuando se presiona play, este asigna un valor a una variable, que indica que el botón de play ya no inicia en 0, sino que es un botón que reanuda el audio.

Finalmente se tiene el botón de elegir audio, el cual es el encargado de desplegar lo siguiente:

Figura 34: Seleccionar Audio de Interfaz Gráfica

Cuando se despliega esta ventana, se redirecciona a un folder predeterminado donde se encuentran todos los archivos de audio que se han generado, estos archivos como se mencionó anteriormente, tienen la información correspondiente (Frecuencia central y frecuencia de pulso binaural) para que el usuario los pueda elegir correctamente. Hay que mencionar que por obvias razones sino se ha generado ningún archivo desde el generador de sonidos, no se podrá seleccionar ningún archivo.

8.3.9. Implementación multihilos

Uno de los objetivos del presente trabajo es la implementación efectiva del método multihilos, para poder obtener un funcionamiento óptimo de la aplicación, por ejemplo, mientras se está ejecutando un subproceso en una ventana, se tiene que poder realizar otros subprocesos, y no interrumpir los mismos, esto se logra utilizando multihilos entre otros métodos.

Inicialmente se planteó como solución del problema la programación en multhililos, pero debido a las condiciones de programación que sucedieron, se encontró una solución diferente pero eficaz, que lograse ejecutar varios subprocesos al mismo tiempo, siendo esto la utilización de clases y métodos. Siendo Python un lenguaje de programación orientado a objetos, y cuales son las ventajas más grandes de la programación orientada a objetos, en este caso se sabe que es la definición de clases y la manera en la que estas clases pueden heredar sus atributos entre ellas.

Tomando en cuenta lo mencionado anteriormente, el problema se solucionó programando en Python, con programación orientada a objetas, en la cual en el script, se programaron todas las clases en un mismo script. Recordando los métodos mencionados anteriormente, específicamente el método *showFrame* indica cual de las ventanas será desplegada, pero si una de las ventanas no está desplegada, puede seguir realizando proceso, debido a que todo se corre simultáneamente. Por ejemplo, si se está reproduciendo un audio, y se cambia de ventana, lo que sucede realmente es que se está desplegando visiblemente la ventana nueva,

pero el proceso el cual está realizando la ventana anterior que en este caso es de reproducción sigue ejecutándose en paralelo, debido a esto no es necesaria la implementación de multihilos. Cabe añadir que está solución no fue planeada, solamente debido a la manera de programar en Tkinter de manera ordenada y efectiva se logra obtener esta funcionalidad de ejecutar en paralelo varios subprocesos.

Es importante mencionar que no se establece que la forma de programa =r por clases es un sustituro para la programación multihilos, sino que para la aplicación que se desarrolló, no es necesaria la implementación multihilos, debido a que la función de la misma se ve cubierta con el método de programación orientada a objetos, en este proyecto en específico no es necesaria la implementación del modelo multihilos.

CAPÍTULO 9

Pruebas e implementación

Luego de haber logrado una implementación efectiva de la interfaz gráfica, considerando que esta es una parte muy importante de la presente tesis, se tiene que la misma solo es una herramienta, para poder empezar a observar patrones de comportamiento que las personas tienen al momento de ser expuestas a pulsos binaurales, en este caso, se tienen que hacer pruebas preliminares para poder detectar a grandes rasgos, que tipo de frecuencias funcionan mejor para dormir más plácidamente, También se deben de descartar los rangos de frecuencia con los cuales sea imposible descartar. Este capítulo ahonda un poco más en la experiencia del usuario, dejando por un lado los prototipos y desarrollos, llegando a una opinión personal.

Cabe destacar que por la pandemia causada por el COVID-19 las pruebas se realizaron a un grupo pequeño de personas, comenzando solamente por el estudiante Luis Guerrero, y su familia.

9.1. Frecuencias centrales y diferencia de tonos no idóneos

Esta etapa es clave debido a que, esta será una restricción que se le tendrá que dar al usuario, haciendo pruebas preliminares, las frecuencias más altas son insoportables, debido a esto se presume que los mejores resultados se darán a frecuencias más bajas, a continuación se detallará de mejor manera como se llevo acabo la experimentación.

El primer análisis se hizo con las frecuencias centrales de los pulsos, como se mencionó anteriormente los humanos tenemos un espectro de aproximadamente de 16 Hz a 20,000 Hz en audición. Lo que se realizó fue una prueba utilizando la misma frecuencia de pulso binaural de 20 Hz, y también el mismo tiempo de muestreo, lo que garantizaba que los resultados fueran más homogéneos entre si. (Se hicieron experimentos con otras diferencias

de frecuencia entre canales y el resultado fue el mismo, se varío la frecuencia de pulso binaural de 1 Hz a 100 Hz y se obtuvo el mismo resultado) Se empezó en una frecuencia alta de 10,000 Hz, la cual aunque es audible, no presenta ningún tipo de relajación ni de confort, se fue bajando la frecuencia gradualmente, por ejemplo se evaluaron frecuencias de 2,000 Hz, 1000 Hz, 500 Hz, hasta llegar a 1000 Hz, que aunque no es un sonido muy agradable para el examinador, no es insoportable como los sonidos mencionados anteriormente, seguidamente se fue bajando la frecuencia y el examinador concluyó que el rango de 16 Hz a 250 Hz es el adecuado para poder llegar a facilitar el sueño. Cabe mencionar que es muy importante la intensidad del sonido, ya que a frecuencias más altas, se puede tornar el sonido insoportable, también hay que considerar que cuando el sonido no es tan intenso se puede perder la frecuencia del pulso binaural, debido a esta prueba preliminar debemos de tomar en cuenta lo antes descrito/

El experimento siguiente a realizar es calcular las frecuencias más agradables de pulso binaural, esto se llevo a cabo analizando una frecuencia central del sonido que fuese agradable, en este caso 150 Hz, e ir variando la frecuencia de pulso binaural, desde un punto bastante agradable, 5 Hz, hasta que llegamos a los x Hz, como fue el caso anterior , con frecuencias de pulso binaural más bajas se obtuvo un resultado más placentero y que pudiese llevar a un sueño más plácido. Se fue aumentando la frecuencia de pulso binaural, y se llego a la conclusión que este valor hasta 1000 Hz puede llegar a no molestar tanto, como se estableció en el párrafo anterior se tiene la hipótesis que tanto frecuencias centrales más bajas (o graves) como frecuencias de pulso binaural más pequeñas pudiesen contribuir más al sueño.

Considerando esta experimentación previa que se llevó a cabo, se tienen que:

- La frecuencia central debe de ser de 16 - 1000 Hz para no ser tan molesta
- La frecuencia de pulso binaural, debe der ser de 1 a 1000 Hz
- Hay que tomar en consideración la intensidad del sonido a mayores frecuencias

Cabe mencionar que se había cometido un error previo, en el cual se había descartado todas las frecuencias mayores a 250 Hz, pero haciendo una prueba extensiva e incluyendo la intensidad del sonido se llegó a los postulados descritos anteriormente.

9.2. Primeras pruebas con pulsos binaurales

La sección anterior proporcionó datos para poder reducir de todo el espectro auditivo, cuales frecuencias tanto centrales como de pulso binaural, son más idóneas para poder dormir con ellas ya que esto es el fin del presente trabajo. Tomando en consideración lo antes mencionado se empezaron a realizar pruebas con el primer sujeto (el escritor de esta tesis) para poder analizar los distintos efectos que los pulsos binaurales pueden tener en nuestro sueño.

El experimento tiene varios pasos, el primero consiste en encontrar la frecuencia central de los tonos, la cual sea más cómoda para la persona esté siendo expuesta a los pulsos

binaurales, se empezará con una frecuencia de 1000 Hz y se irá decrementando la frecuencia poco a poco hasta encontrar la mas idónea, cabe mencionar que estas pruebas se realizarán con una frecuencia de pulso binaural de 20 Hz, la cual por experimentación anterior se considera una frecuencia estándar. Luego de haber encontrado la frecuencia central de los pulsos, se prosigue con la elección de la frecuencia de pulso binaural, esto se hará de primero, subiendo la frecuencia original de 20 Hz hasta llegar a un punto que no sea muy cómodo utilizar el dispositivo, al llegar este momento, también se evaluarán frecuencias más bajas a 20 Hz, para poder llegar a una frecuencia de pulso binaural que sea lo más agradable posible, esta va a ser la metodología a llevar a cabo con esta experimentación, para obtener un rango de valores en los que el sujeto de prueba se beneficie de las pruebas.

9.2.1. Frecuencia central 1000 Hz, frecuencia de pulso binaural 20 Hz

En este caso el sujeto de prueba no fue capaz de conciliar el sueño debido a que el sonido es muy agudo, aunque el pulso binaural tenga una interpretación a nivel neurológico, debido a este experimento se cree que de primero al pasar por los órganos auditivos las frecuencias muy altas producen rechazo o molestia a la persona.

En esta frecuencia central es imposible dormir, aun variando la intensidad del sonido no se logra llegar a niveles de relajación óptimos para dormir, esta frecuencia mantiene alerta al sujeto de prueba.

9.2.2. Frecuencia central 500 Hz, frecuencia de pulso binaural 20 Hz

Con esta frecuencia central el sujeto pudo conciliar un sueño superficial, ya que oyendo el pulso binaural el mismo se dormía por etapas, eran breves etapas en los que se lograba dormir, se puede dormir con esta frecuencia central pero no es muy recomendable. A criterio del sujeto de prueba esta frecuencia central sigue siendo muy aguda, aunque ya se muestra resultados interesantes ya que en este caso el individuo si pudo dormir. Cabe mencionar que después de dos horas de pruebas el sujeto decidió no seguir debido a que era un poco molesto no llegar a un sueño profundo.

Características de la presente frecuencia central:

- Sueño superficial, no se puede dormir profundamente
- Sonido un tanto agudo

Cabe mencionar que en este caso se tuvo problema con los audífonos ya que era un poco incómodo dormir con audífonos de casco que cubren toda la oreja, esto se hizo debido a que el sonido se encuentra más aislado, pero en la siguiente prueba por comodidad se hará con audífonos *in ear* para observar que tanto influye el tamaño de los mismos, de no ser el caso se seguirá utilizando audífonos de casco.

9.2.3. Frecuencia central 250 Hz, frecuencia de pulso binaural 20 Hz

9.2.4. Frecuencia central 100 Hz, frecuencia de pulso binaural 20 Hz

9.2.5. Frecuencia central 50 Hz, frecuencia de pulso binaural 20 Hz

9.2.6. Frecuencia central 25 Hz, frecuencia de pulso binaural 20 Hz

9.2.7. Frecuencia central 10 Hz, frecuencia de pulso binaural 20 Hz

9.3. Pruebas con frecuencia de pulso binaural

En la sección anterior se pudo determinar cual frecuencia de pulso binaural propicia el sueño, hay que mencionar que esto fue una desición tomada por el examinador, la cual puede entrar en discusión, pero debido a esto se planteó que el rango de frecuencias de frecuencia central más aceptable para el presente trabajo fuese de tandos Hz a tantos Hz (Falta llegar a esta conclusión).

El siguiente paso para el estudio del efecto de los pulsos binaurales, en la calidad de sueño de las personas, es determinar un rango de frecuencias de pulso binaural, con las que se duerman, mejor, esto se llevó a cabo con la misma metodología descrita anteriormente

CAPÍTULO 10

Mejoras Interfaz Gráfica

La interfaz gráfica descrita en anteriores capítulos, es funcional, pero está pensada más para la realización de pruebas por parte del examinador, esta aplicación tendrá algunos cambios con referencia al primer prototipo, ya que ya habiendo realizado alguna experimentación anteriormente, se puede hacer que las opciones de los usuarios sean más sencillas en relación a:

- Frecuencia central de pulso binaural
- Frecuencia de pulso binaural

También se realizarán algunos cambios los cuales mejorarán tanto la experiencia del usuario como la estética de la aplicación, esto añadiendo algunos componentes, y también estilizando algunas herramientas de la aplicación.

10.1. Entrada de usuario y contraseña

10.2. Estilización de botones y etiquetas

10.3. Resolución de errores de botones de apago y brillo

CAPÍTULO 11

Implementación de datos en la Interfaz Gráfica

Esta sección se enfocará mayormente en la implementación parcial entre datos en vivo y la aplicación que se desarrolló junto con su Interfaz gráfica correspondiente, esto es para que la tésis tenga un poco más de extensión, debido a que se han cumplido algunos objetivos generales, y se puede tener más resultados.

En este caso se utilizarán los datos en vivo para poder determinar que tipo de pulso binaural será empleado, esto puede conllevar un estudio posterior, en el cual dependiendo de las etapas del sueño de cada persona, y de las señales EEG, que estas produzcan, se puede llegar a encontrar un patrón personal para poder propiciar el sueño en esta persona, hay que mencionar que solo es un estudio adicional, y no se busca resolverlo totalmente en esta tesis.

11.1. Importación de gráficas en vivo en Tkinter

11.2. Cambios de frecuencia según la información

CAPÍTULO 12

Conclusiones

La implementación en dispositivos como la raspberry pi puede ser un tanto complicada, ya que hay muchas adaptaciones que se tienen que hacer si se está migrando un programa en windows.

Es más fácil desarrollar una aplicación pensando en un dispositivo en específico, ya que las migraciones pueden resultar muy complicadas, ya que dos sistemas nunca son iguales.

Las implementaciones de interfaces gráficas pueden resultar más complicadas de lo que se espera, debido a que los mismos módulos no es tan fáciles implementarlos en conjunto.

La frecuencia central de los pulsos binaurales es un rango entre 10 y 200 Hz.

La frecuencia de pulso binaural debe de encontrarse entre 1 y 50 Hz.

CAPÍTULO 13

Recomendaciones

Investigar e implementar una manera de poder analizar un análisis cuantitativo de los efectos de los pulsos binaurales en las señales electro encefalográficas.

Si se va a utilizar una Raspberry analizar todas l,as librerias que se instalaron en la Raspberry Pi utilizado, para que esto no cause problemas.

Analizar todas las opciones antes de dar una que de el resultado correcto.

Investigar antes de realizar un proyecto, no empezar a investigar sobre la marcha, cuando se tienen bases de conocimientos es mucho más fácil trabajar.

CAPÍTULO 14

Bibliografía

[1] J. P. M. Nuñez, *Diseño de un sistema inteligente de monitoreo de ondas EEG y generador de pulsos binaurales para combatir desordenes de sueño en los atletas*, Trabajo de graduación, nov. de 2019.

[2] L. Chaileb, E. Wilpert, P. Reber y J. Fell, "Auditory Beat Stimulation and its Effects on Cognition and Mood States", *Frontiers in Psychiatry*, vol. 6, n.º 70, mayo de 2015.

[3] M. Lee, C. Song, G. Shin y S. Lee, "Possible Effect of Binaural Beat Combined With Autonomous Sensory Meridian Response for Inducing Sleep", *Front Hum Neurosci*, n.º 13, pág. 425, dic. de 2019.

[4] A. Collado, O. Sánchez, A. Almanza, E. Arch e Y. Arana, "Epidemiología de los trastornos del sueño en población mexicana: seis años de experiencia en un centro de tercer nivel.", *Asosiación médica ABC*, vol. 61, n.º 2, jun. de 2016.

[5] J. Satheesh y P. Bhunavaneswari, "Analysis of Electroencephalography (EEG) Signals and Its Categorization–A Study", *Procedia Engineering*, vol. 38, jun. de 2012, Special Issue: Edición especial.

[6] *Natural Patterns of Sleep*, `http://healthysleep.med.harvard.edu/healthy/science/what/sleep-patterns-rem-nrem`, Accessed: 2019-10-5.

[7] P. L. Fernández, "Conceptos físicos de las ondas sonoras", *Física y Sociedad, Revista del Colegio Oficial de Físicos*, n.º 11, 2000.

[8] J. A. Gottfried, *Neurobiology of Sensation and Reward*. Boca Raton (FL): Department of Neurology Northwestern University, Feinberg School of Medicine, Chicago, Illinois, 2011, ISBN: 978-1-4200-6726-2.

[9] N. Jirakittayakorn e Y. Wongsawat, "A Novel Insight of Effects of a 3-Hz Binaural Beat on Sleep Stages During Sleep", *Frontiers in human neuroscience*, vol. 12, n.º 387, sep. de 2018.

[10] J. Licklider, "On the Frequency Limits of Binaural Beats", *The Journal of the Acoustical Society of America*, vol. 322, n.º 468, 1950.

[11] C. Beauchene, N. Abaid, R. Moran, R. Diana y A. Leonessa, "The Effect of Binaural Beats on Visuospatial Working Memory and Cortical Connectivity", *PLoS ONE*, vol. 11, n.º 11, nov. de 2016.

[12] E. Montnémery y J. Sandvall, *Ogg/Vorbis in embedded systems*, 2004.

[13] L. of Congress, *Broadcast WAVE Audio File Format*, https://www.loc.gov/preservation/digital/format Obtenido de la Library of Congress, dic. de 2009.

[14] *Raspberry Pi 4*, `https://www.raspberrypi.org/products/raspberry-pi-4-model-b/`, Accessed: 2019-15-5.

[15] S. Johnston y S. Cox, "The Raspberry Pi: A Technology Disrupter, and the Enabler of Dreams", *Electronics*, vol. 6, n.º 3, pág. 51, jul. de 2017.

CAPÍTULO 15

Anexos