

**Desarrollo de Plugin VST, para la reproducción binaural de fuentes virtuales en movimiento, utilizando bases de datos de Funciones de Transferencia Asociadas a la Cabeza.**

**Daniel Cardona Rojas  
Marcel Valderrama Pulgarín**

**Asesor**

**Héctor García Mayén**



**Universidad San Buenaventura Seccional Medellín**

## **Agradecimientos**

Agradecemos especialmente al docente Antonio Escamilla por su constante interés y soporte durante varios semestres a la investigación que aquí se presenta. Fue quien nos asesoró en los conceptos de la síntesis binaural y presento las herramientas esenciales para el desarrollo del plugin al igual que los conceptos fundamentales para el procesamiento de audio en tiempo real. También agradecemos a todos aquellos que en algún momento nos hayan remitido a fuentes bibliográficas importantes que nutrieron y aceleraron el proceso de este proyecto.

## Contenido

Contenido.....	3
1. Objetivo General. ....	5
2. Objetivos Específicos. ....	5
3. Descripción General .....	5
3.1. Planteamiento del Problema de Investigación y su Justificación en Términos de Necesidades y Pertinencia.....	5
3.2 Propósito del Proyecto de Grado. ....	6
3.3 Marco Teórico.....	6
3.3.1 Localización de Fuentes Sonoras .....	7
3.3.1.1 Diferencias Interaurales de Tiempo (ITD) .....	7
3.3.1.2 Diferencias Interaurales de Intensidad (IID) .....	8
3.3.1.3 Función de Transferencia Relacionada con la Cabeza, Torso y Pabellón Auditivo (HRTF) .....	9
3.3.2 Reproducción Binaural .....	10
3.3.3 Base de Datos Medida por el IRCAM. ....	10
3.3.4 Base de Datos Medida por el MIT. ....	11
3.3.5 Comparación de las Bases Datos (IRCAM, MITKemar). ....	11
3.3.6 Interpolación Bilineal.....	12
3.3.7 Convolución en Tiempo Real .....	13
3.3.8 Transformación a Fase Mínima .....	15
3.3.9 Generalidades Sobre los Plugins de Audio .....	16
3.3.10 Virtual Studio Technology VST Plugin.....	17
3.4 Estado del Arte.....	17
3.5 Análisis Previo al Procesamiento .....	19
3.5.1 Justificación del Modelo de ITDs. ....	19
3.5.2 Justificación del Método Interpolación. ....	21
3.5.3 Prueba del Procesamiento. ....	24
3.6 Implementación .....	25
3.6.1 Pre Procesamiento Utilizado con la Base de Datos. ....	25
3.6.2 Procesamiento Utilizado. ....	26
3.6.3 Desarrollo en MATLAB (Scripts).....	27
3.6.4 Desarrollo del VST .....	28

3.6.4.1 Rutinas de Procesamiento del Plugin VST .....	28
3.6.4.2 Clases Desarrolladas. ....	30
3.6.4.3 Librerías Utilizadas. ....	32
4. Descripción General de la Prueba Subjetiva .....	33
4.1. Descripción Sección 1: Reconocimiento Discreto de Fuentes .....	34
4.2. Descripción Sección 2: Reconocimiento de Continuidad.....	35
4.3. Análisis de Datos .....	35
4.3.1. Sección 1: Reconocimiento Discreto de Precedencia .....	35
4.3.2. Sección 2: Reconocimiento Continuo de Precedencia .....	41
5. Conclusiones .....	45
6. Trabajos Futuros .....	46
7. Bibliografía .....	47
8. Glosario.....	49
ANEXO 1: FICHA TÉCNICA DEL PROYECTO DE GRADO.....	51
ANEXO 2: PRUEBA SUBJETIVAS .....	52
ANEXO 3: INSTALACION DEL PLUGIN (WIN/MAC) .....	64

# **Desarrollo de Plugin VST, para la reproducción binaural de fuentes virtuales en movimiento, utilizando bases de datos de Funciones de Transferencia Asociadas a la Cabeza.**

## **1. Objetivo General.**

Desarrollar un plugin VST, que permita la reproducción y espacialización binaural en tiempo real de fuentes virtuales en movimiento, mediante la utilización de una base de datos de HRTFs.

## **2. Objetivos Específicos.**

- Descomponer la HRTF en uso en su equivalente de fase mínima, y estimar la diferencias interaurales de tiempo (ITD) equivalente a partir de un modelo geométrico, para obtener HRTFs que no tengan diferencias Interaurales de tiempo sesgadas.
- Utilizar el método de interpolación bilineal y realizar el filtrado de la señal, utilizando una rutina que permita la convolución en tiempo real.
- Realizar una prueba subjetiva, en la población de estudiantes de ingeniería de Sonido, para corroborar los métodos de interpolación y procesamiento empleados.

## **3. Descripción General**

### **Pregunta de investigación**

¿Cuál es el desempeño del algoritmo de interpolación bilineal que utiliza una representación en fase mínima de las HRTFs, desde una perspectiva subjetiva y cómo se puede refinar este algoritmo para dar lugar a pistas binaurales más precisas?

### **3.1. Planteamiento del Problema de Investigación y su Justificación en Términos de Necesidades y Pertinencia.**

Con el auge de nuevas tecnologías y aparición de productos interactivos y multimediales, tales como videojuegos, aplicaciones para dispositivos móviles, etc. Aparece el audio 3D como una herramienta que puede nutrir este tipo de productos, es en estos casos donde la facilidad y la movilidad juegan un papel importante en las próximas tecnologías, por esto depender de un sistema binaural para tener una percepción real del posicionamiento de fuentes sonoras, abre las posibilidades de suplir la falta de entornos realmente virtuales e interactivos.

Se plantea la idea de poder reproducir fielmente la localización de una fuente sonora en un sistema binaural, con el fin de mejorar la experiencia de un usuario en entornos virtuales, producciones audiovisuales, o multimediales, es por eso que el desarrollo de un plugin que permita el posicionamiento y movimiento de una fuente sonora en un espacio

tridimensional, será una herramienta que cualquier usuario de un Digital Audio Workstation (DAW) pueda utilizar.

Desarrollar una herramienta de código abierto para los estudiantes de la universidad San Buenaventura, será de gran ayuda a que este tipo de tecnologías se pongan en práctica para un mercado que siempre está en crecimiento como lo es el mercado del ocio enfocado a redes sociales, aplicaciones de dispositivos móviles o producciones audiovisuales y publicitarias.

### **3.2 Propósito del Proyecto de Grado.**

Desarrollar herramientas que permitan la movilización de fuentes sonoras virtuales en espacios tridimensionales, en producciones audiovisuales y multimediales, mejora la experiencia que el usuario tiene en este tipo de productos interactivos. Demostrar que el desarrollo de este tipo de herramientas no está tan alejado de la realidad y que con el auge de nuevos dispositivos y el mejoramiento de procesadores, desarrollar software con características de procesamiento que en un pasado sería casi imposible, se pone al alcance de la mano.

A pesar de los alcances de muchos sistemas de reproducción de audio 3D, tales como los sistemas Dolby Atmos, que mejora la respuesta en el plano vertical que tiene un sistema 5.1 convencional, o sistemas Ambisonics<sup>1</sup>, que trabajan con arreglos numerosos de altavoces. Sigue existiendo la necesidad de contar con herramientas de bajo costo y portabilidad. Es aquí donde la reproducción binaural brinda movilidad y portabilidad, dado que la practicidad que este sistema brinda, facilita las producciones enfocadas a la interactividad. La síntesis binaural permite posicionar fuentes sonoras virtuales en un espacio tridimensional, de esta manera desarrollar un plugin óptimo en procesamiento, podría ser útil en audio publicitario, producciones multimediales, visitas virtuales, aplicaciones para dispositivos móviles, y video juegos.

### **3.3 Marco Teórico**

La reproducción binaural depende necesariamente de funciones de transferencia asociadas a la cabeza más comúnmente llamadas HRTFs, las cuales codifican las pistas sonoras que utiliza el cerebro para poder definir la precedencia de un sonido. El filtrado no solo se hace por las reflexiones del frente de onda en el pabellón auditivo sino que también el conjunto de cabeza torso y canal auditivo tienen influencias en este proceso, de ahí que se le otorgue a la cabeza la representación de este conjunto de órganos. El proceso de filtrado y por tanto las HRTFs (vistas como un filtro) dependen de dos parámetros esenciales, los cuales son el ángulo de elevación y ángulo de azimut con el que incide el sonido. No existe un filtro paramétrico analítico que pueda representar todas las variables y conjunto de curvas para cada par (elevación y azimut posible). Por esta razón, es necesario muestrear y obtener un conjunto finito de respuestas al impulso asociadas a la cabeza. Antes de discutir el procesamiento necesario para caracterizar dichos filtros se debe entender cuáles son las pistas utilizadas por el cerebro para localizar una fuente

---

<sup>1</sup> Véase Ambisonics: <http://www.ambisonic.net/>

sonora.

### 3.3.1 Localización de Fuentes Sonoras

El reconocimiento de la procedencia de un sonido se realiza a través de dos pistas sonoras, de las cuales nuestro cerebro extrae la información para dar una impresión acertada de la dirección de incidencia del sonido. Estas pistas son las diferencias Interaurales de Tiempo (ITDs) y las Diferencias Interaurales de Intensidad (IIDs), ambas son variables que toman valores diferentes en función de la frecuencia, producto de sombras acústicas, refracciones y otros fenómenos acústicos. Como se verá más adelante, una de las dos pistas es prioritaria y provee mejor información que la otra en un rango de frecuencias específico [1].

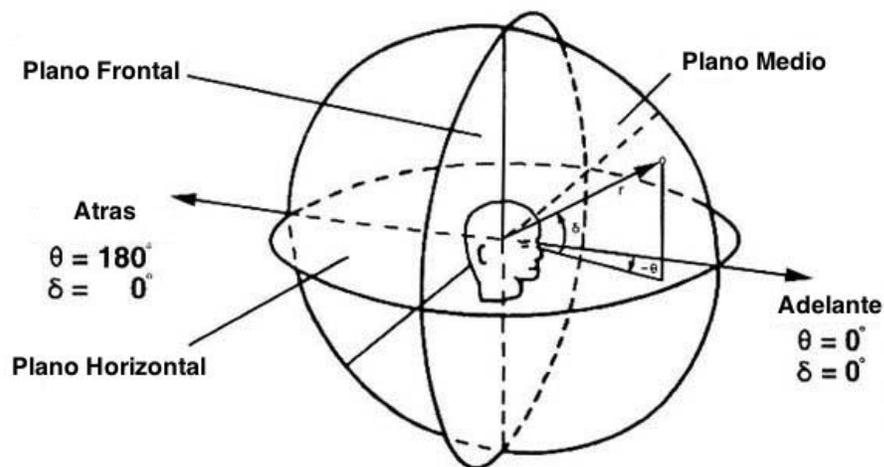


Figura 1. Planos característicos para la localización de fuentes sonoras. imagen modificada de [2].

#### 3.3.1.1 Diferencias Interaurales de Tiempo (ITD)

Las ITD son producidas por que el sonido recorre diferentes distancias para llegar a cada oído con excepción de ángulos de azimut de  $0^\circ$  y  $180^\circ$  por ende el cerebro procesa esta diferencia para obtener una referencia de la procedencia del sonido en el plano horizontal (lateralización), esto solo es válido para frecuencias por debajo de los 1600Hz pues en frecuencias superiores, la longitud de onda es más pequeña que el tamaño de la cabeza lo que complica la detección del ITD. En la Figura 2, se ve representado el camino que realiza un frente de onda y la diferencia en tiempo con respecto a cada oído.

Las diferencias de tiempo pueden ser calculadas teniendo en cuenta los ángulos de incidencia de la onda con respecto a los oídos permitiendo identificar esta pista en la determinación de la posición de la fuente [1].

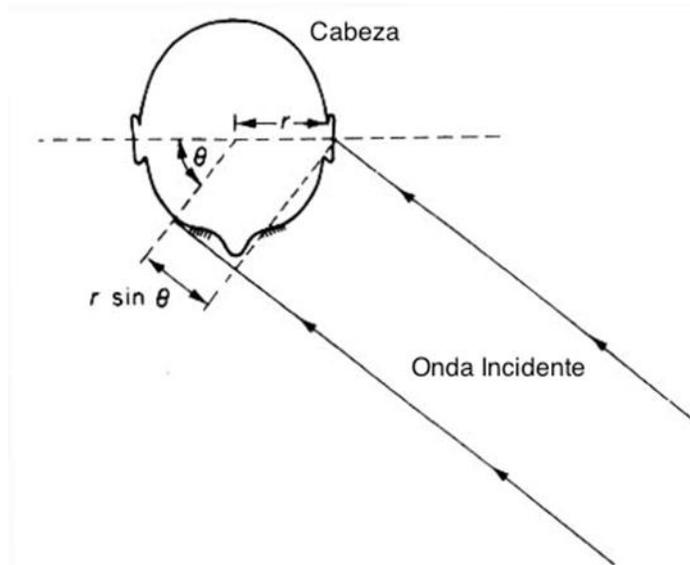


Figura 2. Diferencias Interaurales de Tiempo. imagen modificada de [1].

Sin embargo en un sistema de tres dimensiones, es necesario considerar también el ángulo de elevación “ $\varphi$ ”, el cual permite obtener una representación más fiel de las diferencias de tiempo entre ambos oídos. Teniendo en cuenta estos conceptos se presenta una fórmula para calcular el ITD a partir de los ángulos de incidencia (azimut y elevación) y el radio de la cabeza.

$$ITD = \left(\frac{r}{C}\right) (\text{sen}\theta + \theta) \cos \varphi \quad (1)$$

Donde

r: Radio de la cabeza

C: Velocidad del sonido

### 3.3.1.2 Diferencias Interaurales de Intensidad (IID)

Las IID son producto de la divergencia geométrica de la onda sonora incidente como de la sombra acústica producida por la cabeza. Sin embargo para frecuencias bajas con longitudes de onda relativamente mayores al diámetro de la cabeza humana (frecuencias menores a 500Hz), este fenómeno se hace imperceptible, dejando las diferencias interaurales de intensidad prácticamente despreciables, siendo esta pista útil solo para frecuencias mayores a 1500Hz como lo muestra la figura 3.

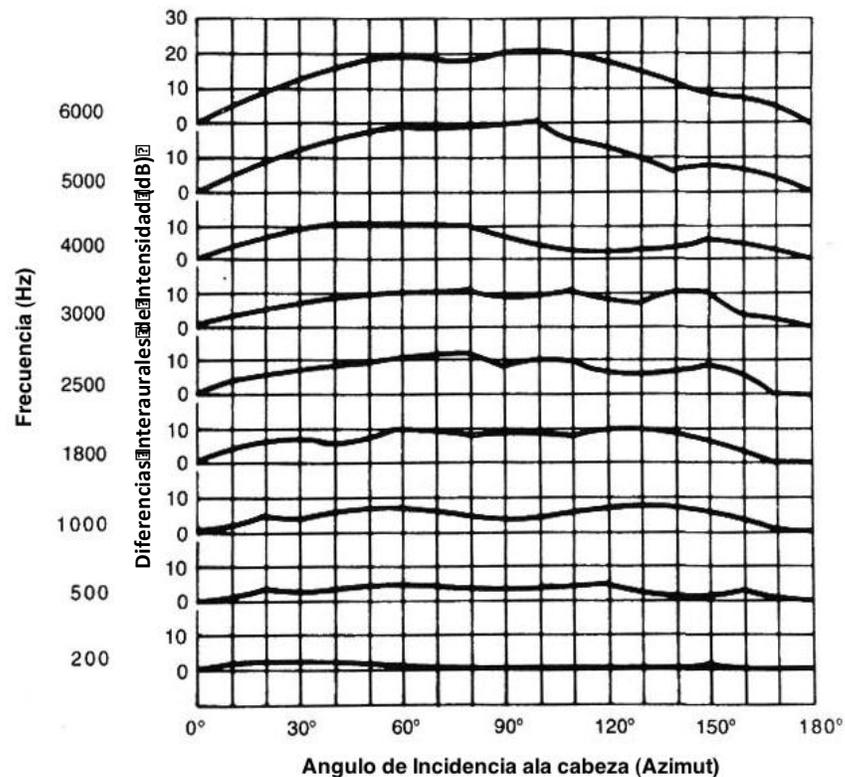
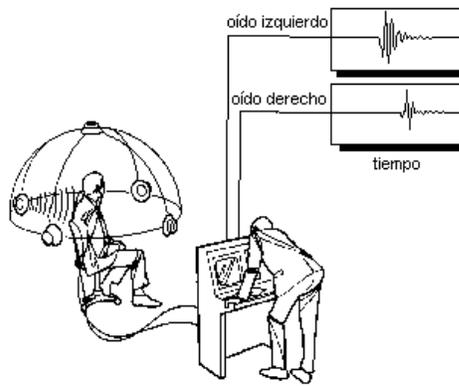


Figura 3. Gráfica de las IID según la frecuencia con respecto al ángulo de incidencia con la cabeza. (Estudio de la gráfica de "An Introduction to the Psychology of hearing").

### 3.3.1.3 Función de Transferencia Relacionada con la Cabeza, Torso y Pabellón Auditivo (HRTF)

Una HRTF (*Head Related Transfer Function*) contiene la información de precedencia sonora que está implícita en las múltiples reflexiones producidas por la cabeza, torso y pabellón auricular que se suman en el canal auditivo o membrana timpánica. Dicho de otra manera una HRTF codifica las pistas binaurales (ITDs e IIDs) necesarias para la localización de una fuente sonora en el espacio. En consecuencia las HRTFs producen un filtrado que depende de dos parámetros esenciales, los cuales son el ángulo de elevación y ángulo de azimut con el que incide un sonido.

Las mediciones de las HRTFs se realizan variando un punto de emisión, desde el cual se reproducen señales de prueba como barridos senoidales, señales MLS, y se registra mediante micrófonos colocados al interior del canal auditivo de un auditor o maniquí. De esta manera se genera una grilla esférica al rededor del receptor. como lo muestra la Figura 4.



Medición de las HRTF

Figura 4. Ilustración de la medición de una base de datos HRTF. Imagen modificada de [3].

Como producto de estas mediciones se obtiene un conjunto de respuestas impulsivas asociadas a la cabeza (HRIR “Head Related Impulse Response”) de duración finita. De esta manera se ha obtenido indirectamente los coeficientes de un filtro (HRTF) tipo FIR.

### 3.3.2 Reproducción Binaural

Una señal binaural es aquella que ha sido procesada con HRTFs y debe encontrarse la manera de ser entregada independientemente a cada oído típicamente a través de audífonos, pues esta contiene información de posición espacial, características acústicas del entorno, y cualidades del movimiento de una fuente, que una señal estéreo reproducida por altavoces no tiene. Sin embargo esta reproducción tiene varios problemas, entre ellos el más notable es el efecto denominado “In Head Localization” [4], que causa problemas en la percepción de profundidad y distancia de las fuentes sonoras y también la sensación de estar dentro de la cabeza a falta de externalización.

### 3.3.3 Base de Datos Medida por el IRCAM.

Esta base de datos se realizó en la cámara Anecoica de  $(8.1 \times 6.2 \times 6.45 = 324 \text{ m}^3)$  del instituto de investigación IRCAM, cuyo montaje de medición se realizó con una tarima metálica configurable, de acuerdo a la posición de la medición (Figura 5). Para la medición de esta base de datos se utilizó un sistema de altavoz TANNON 600 pasivo, un amplificador Yamaha, y para la medición de la respuesta al impulso del dicho altavoz, se utilizó un micrófono 4149 de 0,5 pulgadas de B&K. Para la medición de la HRTF, se utilizaron un par de micrófonos Knowles FG3329. La resolución de esta base de datos es de  $15^\circ$ , variando en altura de  $-45^\circ$  a  $90^\circ$  y en azimut de  $0$  a  $360^\circ$  [5].

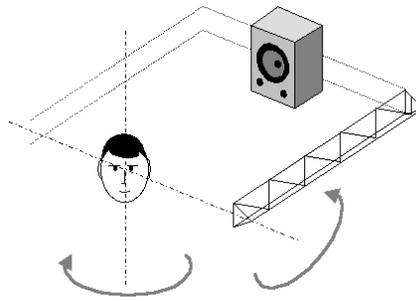


Figura 5. Montaje de medición del IRCAM. Imagen sacada de [5].

Esta base de datos se puede descargar de manera gratuita, y se divide entre la base de datos cruda, con los datos tal cual se midieron, y la base de datos compensada o ecualizada para campo difuso utilizando el método "decoupled equalization", las características de esta ecualización son: la eliminación del retardo por propagación, utilizar la componente en fase mínima de las HRIR, normalización de la fase mínima con respecto al campo difuso utilizado. A su vez, los datos que se presentan pueden ser en formato ".Wav" o como una estructura en MATLAB ".mat" [5].

### 3.3.4 Base de Datos Medida por el MIT.

Para la realización de esta base de datos se utilizó un maniquí KEMAR (Knowles Electronics Manikin for Acoustic Research), esta misma se completó en 1994 donde se utilizaron 7 altavoces puestos a 1,4 metros del maniquí [6], las mediciones se realizaron a una tasa de muestreo de 44,1KHz con un total de 710 posiciones diferentes, que varían entre  $-40^\circ$  y  $90^\circ$  en elevación, y  $360^\circ$  en azimut, cabe resaltar que el incremento azimutal de estas mediciones depende de la elevación. Para su la reducción de las respuestas al impulso a 128 coeficientes y ecualización se utilizaron las mediciones del altavoz en campo libre [7].

La manera como se presentan los datos, es en formato ".Wav", subdivididas las HRTFs en carpetas para cada elevación correspondiente, de igual manera se presenta la base de datos para diferentes auriculares y diferentes micrófonos [6].

### 3.3.5 Comparación de las Bases Datos (IRCAM, MITKemar).

Como ya se ha mencionado anteriormente, cada persona tiene una fisiología específica que implica que las HRTFs varíen. Basados en la Tabla 1, se optó por utilizar la base de datos del MIT, al ser una base de datos menos pesada, con mayor resolución angular, y en la que sus HRTFs han sido sometidas a ecualización y reducción de orden (FIR de 128 Coeficientes). Sin dejar a un lado que la mayoría de la bibliografía consultada, utiliza esta base de datos en sus investigaciones.

Tabla 1. Comparación entre Bases de datos, IRCAM vs el MIT.

Base de Datos	KEMAR MIT	IRCAM
Número de Mediciones (Archivos)	368	187
Longitud de las IR	128 samples	512 samples
Incremento Acimutal	variable (5-30)	15 grados
Incremento Elevación	10 grados	15 grados
Formatos Disponibles	.wav	.wav y .m (estructura MATLAB)
Rango Elevación	[-40 90]	[-45-90]

### 3.3.6 Interpolación Bilineal

Como ya se mencionó existe un único par de HRTF para cada punto en el espacio. Realizar el registro de cada una de ellas forma parte de un extenso proceso de medición, el cual se especifica en la documentación existente de cada base de datos [8] [9] [10] [11]. Es así donde aparecen librerías de acceso gratuito como las proporcionadas por el MIT [8], por el CIPIC [12], por el FIU [13], o por el IRCAM [10]; estas cuentan con una gran cantidad de funciones de transferencia que representan una gran cantidad de puntos en el espacio. Igualmente con estas mediciones realizadas, se obtienen filtros con una baja resolución angular. Una forma de disminuir el número de filtros a utilizar, y con ello generar un sistema de audio 3D con mayor resolución angular y por ende más eficiente, es realizando una interpolación entre puntos adyacentes. Esto ha sido ampliamente estudiado durante las últimas décadas [14] [15] [16] [17] [11] y hasta la fecha se han desarrollado múltiples métodos de interpolación de filtros HRTF eficientes que no generan una mayor degradación en la percepción de la localización de la fuente sonora. Estos métodos han sido analizados en una variedad de artículos, como los mencionados por Fabio P. Freeland [14], el método más popular se ha denominado interpolación bilineal, este método considera las HRTF de los cuatro puntos más cercanos a la función de transferencia deseada, incluyendo ángulos de azimut y elevación.

El método de interpolación bilineal se puede efectuar directamente (sin pre procesar la base de datos), o preparando la base de datos para tener fase. La ecuación que describe dicha interpolación es la siguiente, donde  $h_A$ ,  $h_B$ ,  $h_C$ ,  $h_D$  : son los cuatro datos cercanos a  $h_E$ , que se muestran en la Figura 6;  $c\theta$  es el coeficiente de interpolación de azimut, y se describe como la ecuación (3) [15].

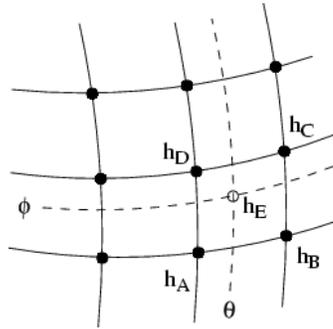


Figura 6. Esquema del método de interpolación bilineal, considerando los cuatro puntos más cercanos. Modificado de [15].

$$h_E = (1 - C_\theta)(1 - C_\phi)h_A + C_\theta(1 - C_\phi)h_B + C_\phi C_{\theta 2}h_C + (1 - C_{\theta 2})C_\phi h_D \quad (2)$$

$$C_\theta = \frac{\theta \text{ mod } \theta_{grid}}{\theta_{grid}} \quad (3)$$

$$C_{\theta 2} = \frac{\theta \text{ mod } \theta_{grid2}}{\theta_{grid2}} \quad (4)$$

$$C_\phi = \frac{\phi \text{ mod } \phi_{grid}}{\phi_{grid}} \quad (5)$$

### 3.3.7 Convolución en Tiempo Real

Una característica importante de la operación de convolución que resulta atractiva para procesos de tiempo real es la linealidad de la operación. En la práctica enfocada al audio la señal de entrada que se quiere convolucionar con una respuesta al impulso es infinita. Es decir esta en las manos del usuario decir la extensión de la señal de entrada y por ende el momento en que el proceso de convolución debe parar. La propiedad asociativa permite en este caso que la secuencia de entrada pueda ser fragmentada en bloques. De esta manera si la respuesta al impulso es considerablemente corta entonces la señal de entrada puede ser segmentada en bloques de igual tamaño a la Respuesta al Impulso y de esta manera realizar convoluciones cortas y entregar bloques de salida de audio. Es así como es posible realizar el proceso de convolución en tiempo real sin tener conocer toda la señal de entrada de ante mano.

Es posible llevar a cabo la operación de convolución desde el dominio de la frecuencia. La multiplicación de dos espectros frecuenciales vía la DFT, corresponde a la convolución circular en el dominio temporal, no obstante la DFT calculada de forma directa tampoco conlleva a un algoritmo rápido. Para que la operación de convolución en el dominio de la frecuencia sea más eficiente que su equivalente temporal es necesario realizar dichas transformaciones usando uno de varios algoritmos de transformadas rápidas de Fourier. El proceso de ir al dominio de la frecuencia, multiplicar ambos espectros y regresar al dominio temporal tiene una complejidad computacional inferior a la convolución llevada a cabo en el dominio temporal cuando se utiliza la transformada rápida de Fourier.

Adicionalmente es necesario linealizar la convolución y que no haya “aliasing”. Si se tiene una señal de  $N$  muestras y un filtro (FIR) de  $L$  muestras, se rellenan o anexan ceros a estas secuencias hasta llegar a  $L+N-1$  muestras, y se realiza la operación de convolución como ha sido descrita arriba, el resultado dará una convolución lineal, como se muestra en la ecuación (5)

$$x(n) * h(l) \Leftrightarrow X(k)H(k) \quad (6)$$

Una manera eficiente de realizar la convolución en tiempo real, es utilizar el método Overlap Add; cuya metodología consiste en dividir la señal en varios fragmentos sin solapamiento, y realizar la convolución entre estos fragmentos de señal con la IR. Esta metodología se utiliza para señales muy largas, o para aplicaciones en tiempo real, un ejemplo práctico de la metodología se puede apreciar en la Figura 7. Donde la señal  $x[n]$  se divide en secuencias que no se solapan. Luego de realizar la transformada discreta de Fourier de las secuencias  $x_k[n]$  y  $h[n]$  se multiplican para dar como resultado un bloque de salida  $Y_k[n]$ . Como paso siguiente se realiza la FFT inversa de  $Y_k[n]$  cuya señal resultante  $y_k[n]$  se reconstruye mediante la superposición y el solapamiento con  $y_{k-1}[n]$ . Anteriormente “ $L$ ” se elegía teniendo en cuenta la capacidad de procesamiento, sin embargo con transformadas rápidas de Fourier desarrolladas eficientemente en la actualidad, se pueden tener transformaciones eficientes para grandes factores de  $L$ .

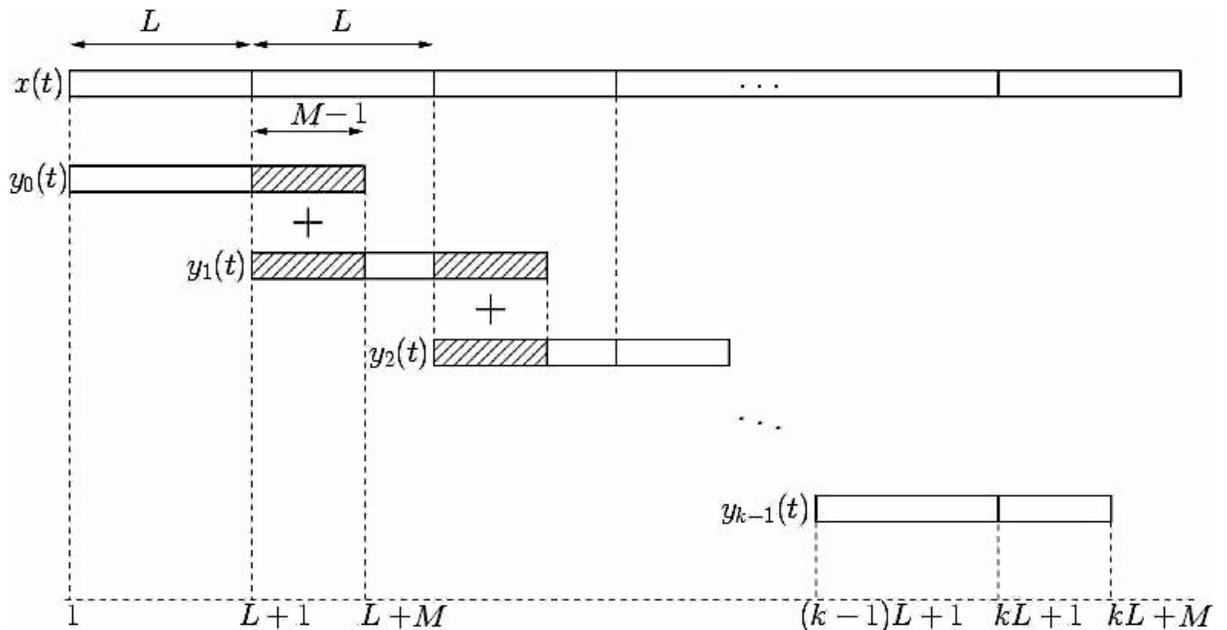


Figura 7. Método de convolución Overlap-add. Figura modificada de [4].

### 3.3.8 Transformación a Fase Mínima

En la teoría de sistemas y señales se establece que cualquier sistema puede ser descompuesto como el producto, de dos sistemas uno de ellos que tiene la propiedad de ser de fase mínima y otro que es allpass.

$$H(z) = H_{min}(z)H_{ap}(z) \quad (7)$$

Un sistema en fase mínima conserva el espectro de magnitud de su equivalente de fase mixta, un sistema de fase mínima es un sistema en el cual ninguna de sus componentes espectrales ha sido sometida a un corrimiento fuera del rango  $[-90 \ 90]$  grados. Otra interpretación es que un sistema de fase mínima tiene ambos polos y ceros dentro del círculo unitario  $|z| < 1$ , lo que hace que el sistema y su inverso por aparte sean estables y causales. La siguiente es la expresión con la que se puede hallar la componente de fase mínima de un sistema, la cual tiene una formulación matemática larga y compleja representada en la ecuación 7.

$$\angle_{mph} = \text{hilbert}(-\log|HRTF_{inter}|) \quad (8)$$

Proviene de las llamadas relaciones de hilbert de transformaciones, de manera simple esta teoría explica bajo que circunstancias es posibles relacionar de manera unica la parte real con imaginaria del espectro de una señal/sistema, o relacionar su magnitud con fase. En el primer caso se puede demostrar que si una señal es causal, entonces, existe una relación unica entre la parte real e imaginaria de una DFT, en el segundo caso (relación entre fase y magnitud) es necesario imponer la restriccion que la señal/sistema tengan un cepstrum causal. El logaritmo en esta

operación esta relacionado con el cepstrum, y la transformada de hilbert en cambio sirve para imponer causalidad.

En vista de la propiedad de relacionar el espectro de fase y el de magnitud en los sistemas de fase minima y dado que el reto principal en la interpolacion de HRTFs es la interpolacion del espectro de fase, es de gran interes en la sintesis binaural. Si una base de datos es descompuesta en componentes de fase minima y allpass, entonces es posibles interpolar las HRTFs de fase minima y a partir de la magnitud interpolada recuperar la fase del sistema. Una simplificación adicional puede adoptarse. En [18] se ha probado que la componente de fase de la sección allpass del sistema es aproximadamente lineal hasta 10 kHz. La linealidad de la fase implica en retraso igual para todas las componentes espectrales por lo que se puede tomar este exceso de fase y reemplazarlo por un retraso temporal. Si recordamos que fase lineal implica retrasos temporales iguales para todas las componentes espectrales, entonces nos permite reemplazar esta informacion en fase por un retraso temporal.

### 3.3.9 Generalidades Sobre los Plugins de Audio

Un audio plugin es una aplicación o ejecutable que corre sobre un host, secuenciador, o Digital Audio Workstation (DAW). Este tipo de aplicativos permiten desarrollar diferentes herramientas sobre el host de soporte, teniendo variables de entrada y de salida que pueden ser los controles que tendrá el aplicativo, una señal de entrada (Mono/Estéreo), y la señal de salida la cual será procesada teniendo en cuenta las características del mismo. En la Figura 8, se puede apreciar de una manera más práctica el funcionamiento interno de un plugin de audio.

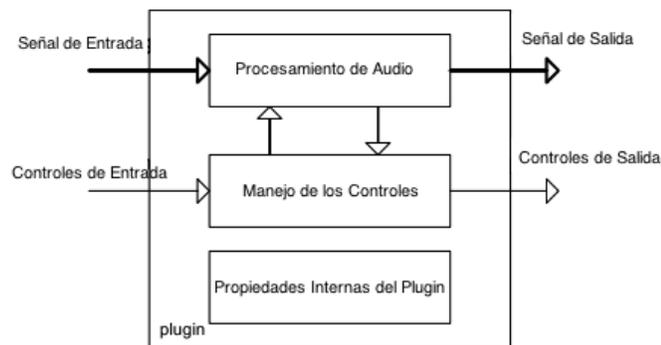


Figura 8. Diagrama de bloques explicativo sobre el procesamiento de un plugin de audio. Imagen modificada de [19].

Una de las principales ventajas de los audio-plugin, es permitir que otros desarrolladores generen contenido para el DAW o Host, extendiendo las funciones, sin que esto implique aumentar la complejidad del mismo.

El funcionamiento de la mayoría de Plugins de audio que corren en tiempo real está fundamentado sobre dos esquemas de programación básicos, el primero es la función callback, esta es llamada cada vez que hay un nuevo buffer de audio por

procesar, es importante generalizar esta función para que sea compatible con cualquiera que sea la longitud del buffer y tasa de muestreo a la que trabaje el Host. Algo importante aquí y que pasa bajo la mesa es que esta función es pasada como argumento a otra encargada de manejar y procesar los bloques de audio de entrada y salida de los canales de nuestro DAW. El segundo esquema de programación consiste en manejar eventos, esto es fundamental dado que es necesario que el plugin conozca y esté atento a los cambios de su interfaz gráfica. El programador por lo tanto debe suplir una función que se encargue de actualizar variables y hacer cálculos adicionales, cada vez que el usuario hace cambios en los parámetros del plugin.

### **3.3.10 Virtual Studio Technology VST Plugin**

Virtual Studio Technology (VST) es una interfaz estándar desarrollada para ser un complemento de Cubase 3.0 por Steinberg Media, sin embargo las ventajas de este protocolo, lo hicieron un estándar en plugins de audio, esta misma se puede definir como una interfaz estándar de aplicativos de audio, cuya función es permitir crear herramientas utilizadas en un estudio virtual, ya sea un procesador de audio, un instrumento virtual, etc.

Una de las principales características, es que su entorno de desarrollo es libre, lo que permite que cualquier programador pueda crear herramientas o aplicativos dentro de este protocolo. Fue con el estándar VST1.0 con el cual se dieron las bases para generar complementos dentro de los hosts, Secuenciadores, o DAW, el cual permitió implementar dentro de las características del complemento un uso arbitrario de entradas y salidas. Luego con la actualización al estándar VST 2.0 se introdujo el soporte para trabajar con mensajes MIDI, con lo cual se consolidó este estándar como una herramienta indispensable en el desarrollo de herramientas de audio.

La extensión del archivo del plugin varía según la plataforma, en Windows, se tiene un archivo “.dll” o librería de enlace dinámica la cual se ejecuta sobre un programa madre o Host, y para los sistemas operativos de Apple se tiene un “.vst” el cual aloja el archivo que se ejecutara sobre el Host.

### **3.4 Estado del Arte.**

Hablar de fuentes sonoras y la manera como el cerebro procesa las características de posición de una fuente, es remontarse a 1907 cuando Lord Rayleigh desarrolló su teoría dúplex (combinación de dos) [8], dando las bases para la localización de una fuente sonora, en la que definió que 2 señales (binaural) se diferenciaban en su fase y en su intensidad, así se empezó a hablar de Diferencias Interaurales de tiempo ITD y Diferencias Interaurales de Intensidad IID, estas 2 son claves para la lateralización y la localización de fuentes sonoras.

Con el tiempo, se han desarrollado diferentes modelos para la espacialización de fuentes sonoras, se han diseñado varios modelos geométricos para estimar las ITD dentro de los cuales se destacan los métodos de Woodworth/Schlossberg, el método de Larcher/Jot, y el método de Savioja. En la actualidad también existen otros métodos para calcular las ITD utilizando las bases de datos HRTFs, como el método de correlación cruzada, regresión lineal sobre la fase de exceso y utilizando el group delay, sin embargo los métodos geométricos siguen siendo eficientes y sencillos de programar [20]. Pauli Minnaar et al. en [20], realizaron un estudio donde se comparan los diferentes métodos de cálculo y aproximación a las ITDs de personas.

Sin embargo identificar la posición de una fuente sonora no depende solamente de estas diferencias Interaurales, de hecho, más estudios demostraron que para ciertas frecuencias se generaba una zona de confusión y la clave de la localización de una fuente la daban las características fisiológicas del pabellón auditivo y el torso [8] [20] [1]. Fue entonces y con el desarrollo de sistemas de procesamiento más avanzado que se logró caracterizar la función de transferencia relacionada a la cabeza, y aparecieron las primeras bases de datos HRTFs [3].

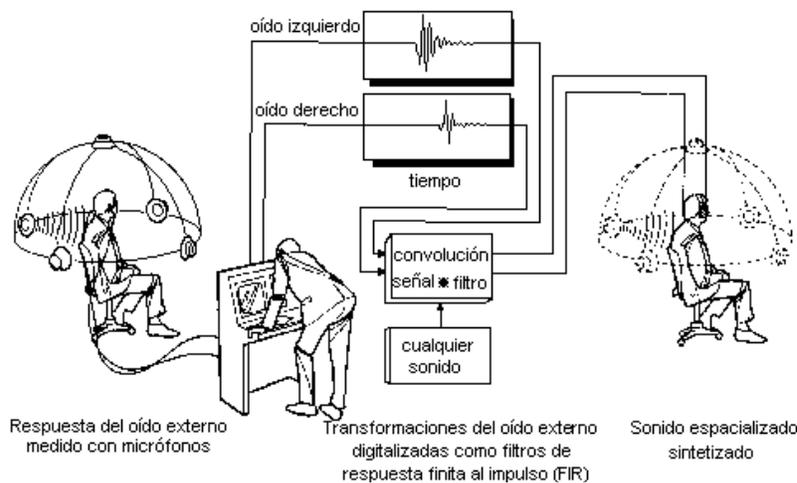


Figura 9. Metodología de medición y reproducción utilizando HRTFs. Imagen modificada de [9].

Las bases de datos de HRTFs se pueden encontrar en distintas resoluciones angulares, y categorizadas según el sujeto de prueba sobre el cual se ha medido la base de datos (maniquí o persona). En casos más particulares se especifica también el tipo de pabellón auricular de la persona. Las mediciones de dichas bases de datos se hacen bajo condiciones de laboratorio en particular en cámaras anecóicas, las cuales son hechas para que las características acústicas del recinto no interfieran en la obtención y caracterización de la HRTF asociada al punto de medición [17]. Majdak et al [21], proponen un sistema de medición rápida de HRTFs, donde el tiempo de medición es reducido a una cuarta parte, sin perder la relación señal a ruido, usando múltiples barridos exponenciales alternados y con solapamiento.

Bill Garner en [8] propuso un sistema de síntesis binaural utilizando altavoces y un algoritmo de cancelación de Crosstalk basado en el posicionamiento de la cabeza de un auditor.

Jean Marc Jot et al [22]. menciona los propósitos de realizar la reducción y ecualización de una base de datos de HRTFs, mencionando el pre procesamiento que se realiza sobre estas, igualmente se describe las implementaciones utilizando HRTFs representadas como filtros FIR e IIR.

Tacksung Choi et al. proponen un algoritmo alternativo a los esquemas convencionales de externalización para dar solución al efecto producido por la utilización de auriculares como sistema de reproducción [23]. Los sistemas convencionales usualmente consisten en algoritmos que filtran las reflexiones tempranas con un par de HRTFs, para tomar en cuenta la dirección de incidencia, lo cual conlleva a un procesamiento costoso. En el sistema propuesto por Choi, se consideran tan solo las primeras dos de las reflexiones tempranas, mientras que el filtrado requerido para simular directividad, se implementa con un par de filtros shelving de primer orden.

### **3.5 Análisis Previo al Procesamiento**

Parte importante de la investigación realizada, es la justificación del modelo de ITDs y la manera como se realizará la interpolación, para lograr pistas binaurales más precisas. En los siguientes apartados se exponen las pruebas realizadas en Matlab con las cuales se justifica el procesamiento desarrollado.

#### **3.5.1 Justificación del Modelo de ITDs.**

Para justificar los modelos esféricos es preciso que se comparen los ITDs calculados de las propias HRTFs con los ITDs estimados por modelos esféricos. Los modelos geométricos de cabeza esférica más comunes en la bibliografía son los de Woodworth, Savioja y Jot [7]. Por medio del script de Matlab (Plot\_ITDs.m) comparamos los modelos esféricos con los calculados por regresión lineal sobre la componente de fase de exceso y con el ITD estimado por el método de correlación cruzada. Se puede apreciar que el modelo esférico de Savioja tiene mayor semejanza a los modelos analíticos. En particular el método de Jot parece estar correlacionado con los ITDs para ángulos de elevación grandes, mientras que el modelo de Woodworth (que no toma en cuenta el ángulo de elevación) solo se asemeja a los modelos analíticos cuando el ángulo de elevación es pequeño, ver las Figuras 10, 11 y 12.

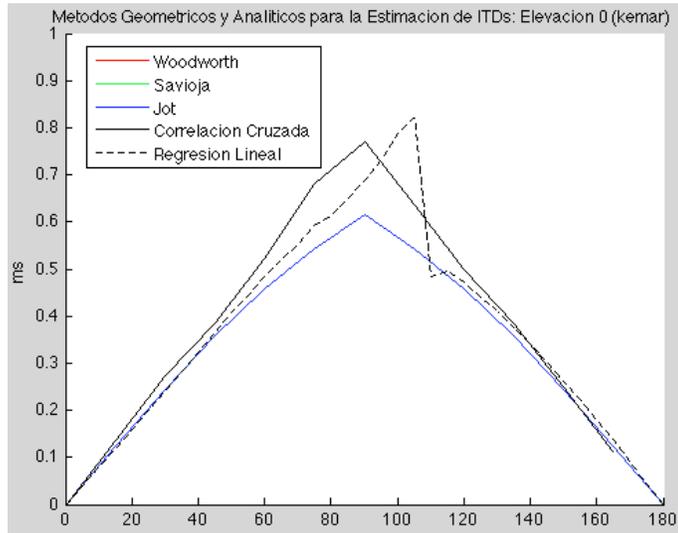


Figura 10. Gráfica de comparación entre métodos analíticos y geométricos del calculo de las ITD a una elevación de 0°.

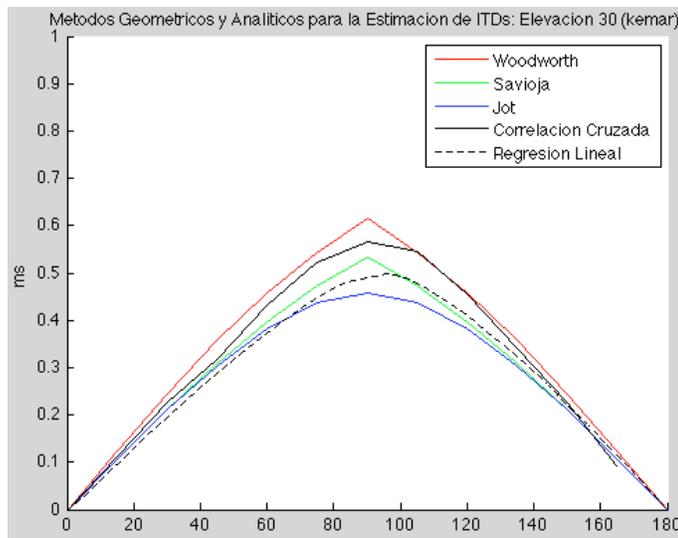


Figura 11. Gráfica de comparación entre métodos analíticos y geométricos del calculo de las ITD a una elevación de 30°.

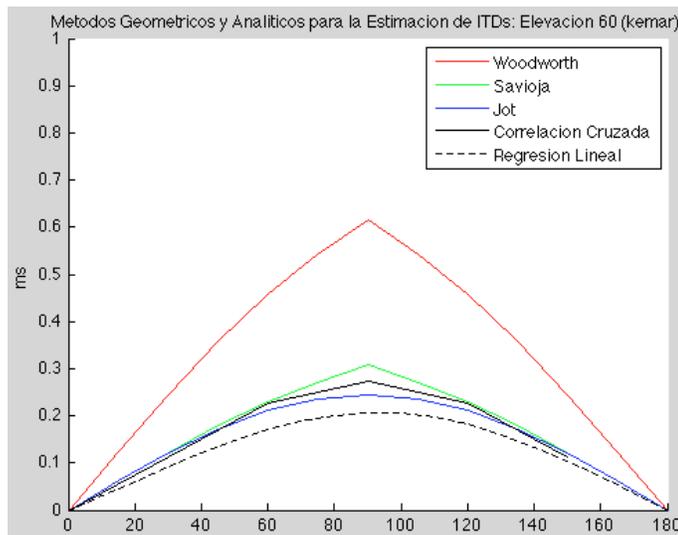


Figura 12. Gráfica de comparación entre métodos analíticos y geométricos del calculo de las ITD a una elevación de 60°.

### 3.5.2 Justificación del Método Interpolación.

Dentro del desarrollo del procesamiento de señal utilizado, se indago sobre la manera en que se debía realizar la interpolación, pues de esta depende el resultado final, se realizaron pruebas comparativas en Matlab, las cuales muestran la diferencia de realizar la interpolación bilineal directa, y las HRIRs circunscritas en representación de fase mínima (FM) más retraso en la base de datos. En las Figuras 13 y 14, se muestra la respuesta en magnitud comparadas.

La interpolación bilineal, es una interpolación lineal que se hace por segmentos, horizontalmente y verticalmente, una aclaración importante aquí, y que debe expandir la expresión matemática planteada en [14]; es que en una base de datos generalmente los incrementos azimutales no son constantes por lo que es necesario tener 3 coeficientes de interpolación en lugar de dos como lo plantean ellos. En cualquier caso cuando en la literatura se refieran a interpolación lineal el termino bilineal también es apropiado y equivalente.

Para esta prueba se utilizó el script "pruebaKemar.m", se analizaran dos gráficas. En la Figura 13, se puede ver la respuesta en magnitud de 2 HRIR localizadas en la base de datos, en este caso para las coordenadas 0 azimut, 0 elevación y 10 azimut, 0 elevación; y la respuesta en magnitud de la HRIR resultante de la interpolación bilineal, para la coordenada 5 azimut, 0 elevación. En el segundo Gráfico se compara la misma HRIR sacada de la base de datos contra la HRIR resultante de la interpolación. Nótese que cuando la estimación se realiza con su componente en fase mínima, la respuesta en magnitud es prácticamente la misma. Por esta razón es indispensable trabajar con la base de datos pre procesada en fase mínima.

En las figuras 15 y 16 se realiza la siguiente prueba: Se inspeccionan las HRTFs de un plano horizontal (elevación 0° o elevación 30°), luego se seleccionan HRTFs haciendo saltos de (2 veces el incremento azimutal) es decir se seleccionan intercaladas. Por ejemplo para una elevación 0° el incremento azimutal es 5° en la base del MIT de esta manera se seleccionan las HRTFs a 0°,10°,20°, .....180° en azimut y se interpolan todas las HRTFs intermedias a 5°,15°,25°....175°. Para efectos de comparación la interpolación bilineal es llevada a cabo con las HRIRs sin pre procesamiento y con las HRIRs en fase mínima con retraso (utilizando el modelo de Savioja). Los ITDs son extraídos de las HRTFs interpoladas por medio de una regresión lineal sobre la diferencia de exceso en fase de la HRTF izquierda con derecha. De esta manera se comparan los ITDs de los puntos intermedios interpolados e implícitos de la base de datos. Analizando estos gráficos se puede ver que la interpolación bilineal de HRTFs con representación en FM más retraso dan como resultado, ITDs simétricas pero menos similares a las obtenidas de las HRIRs sin pre procesamiento. Adicionalmente para visualizar las diferencias en milisegundos de las interpolaciones respecto a las HRTFs adyacentes, se sitúan estas ultimas como las laterales de cada grupo de barras (para la el grupo de barras en 5° la primera barra corresponde a la ITD de la HRTF adyacente en 0° y la quinta barra a 10°). Es de esperar que las ITDs interpoladas estén siempre entre el valor de las ITDs adyacentes.

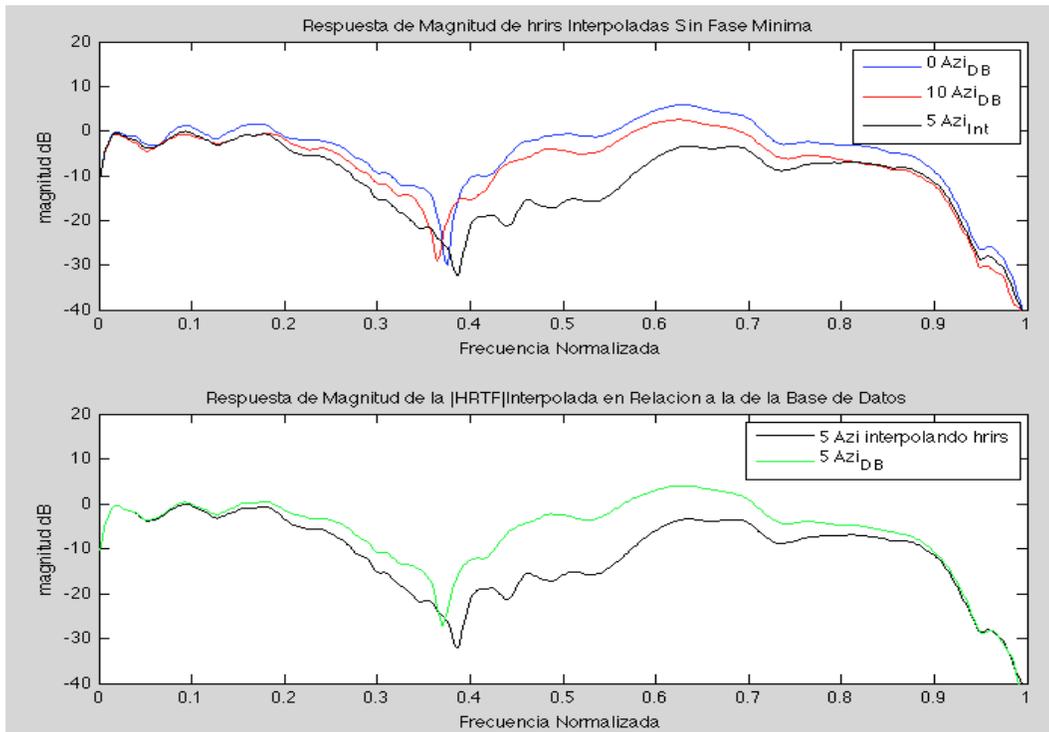


Figura 13. Gráfica de comparación de la respuesta en frecuencia sin fase mínima.

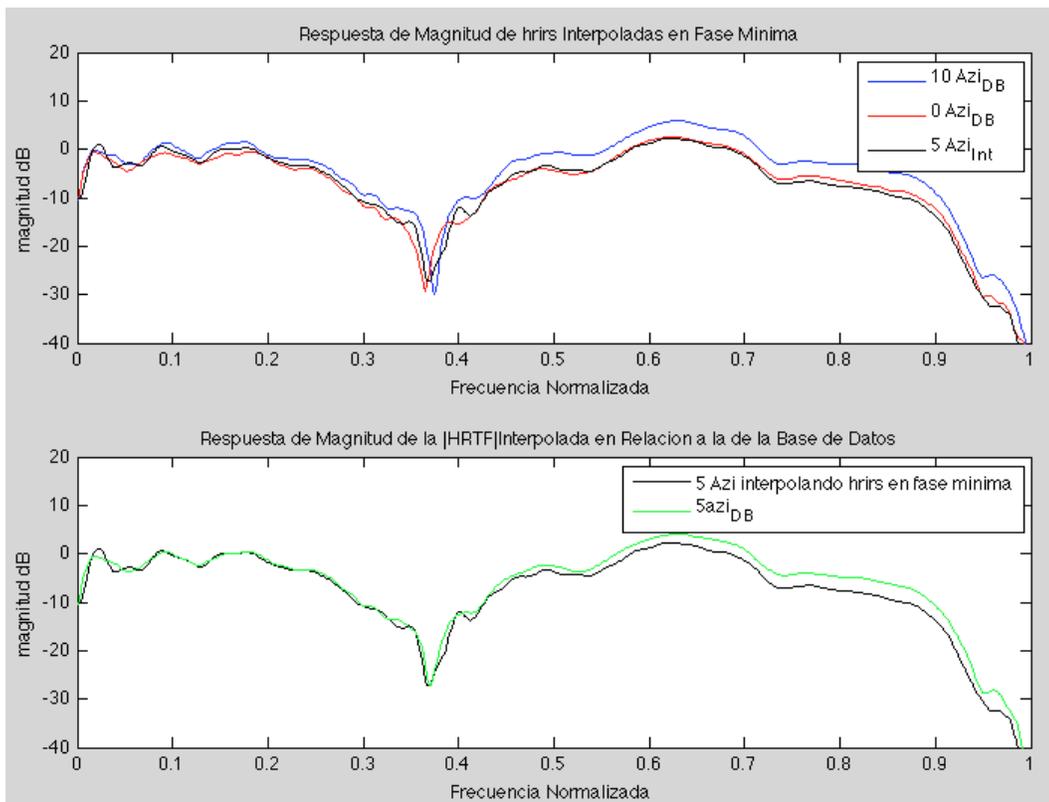


Figura 14. Gráfica de comparación de la respuesta en frecuencia con fase mínima.

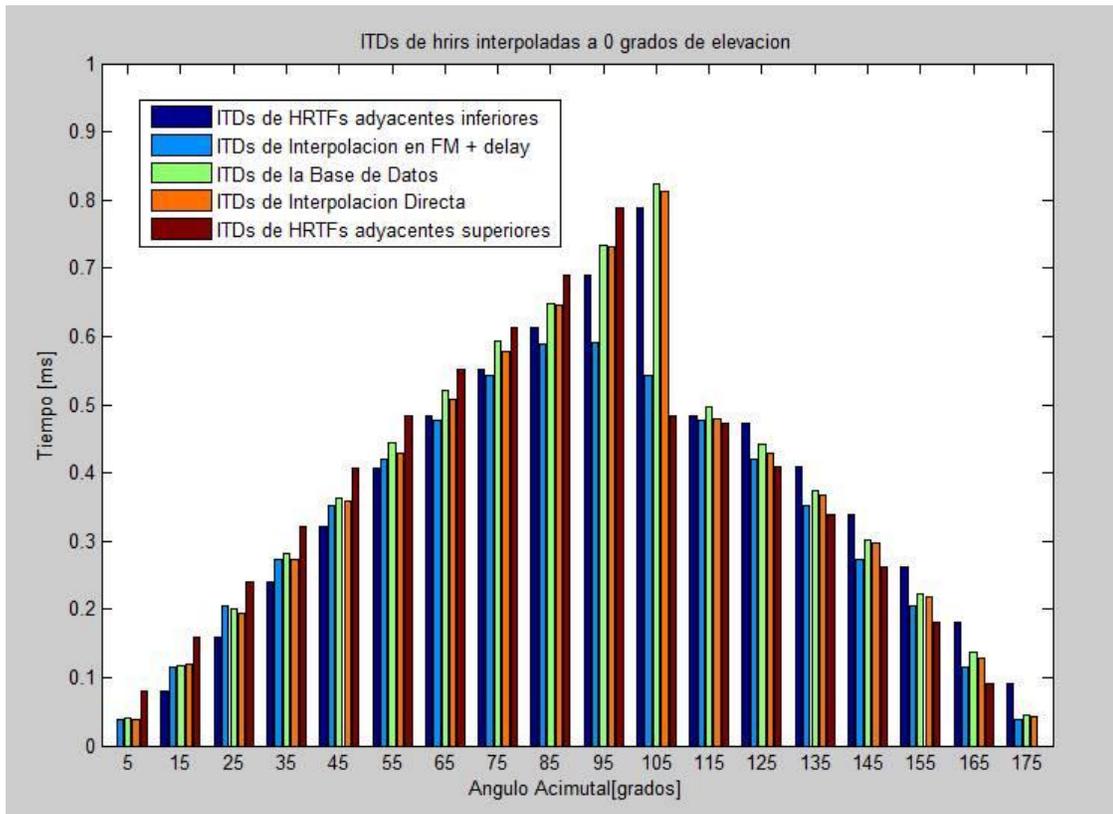


Figura 15. Gráfica de comparación entre las ITDs de HRTFs interpoladas e implícitas en la base de datos a 0° de elevación.

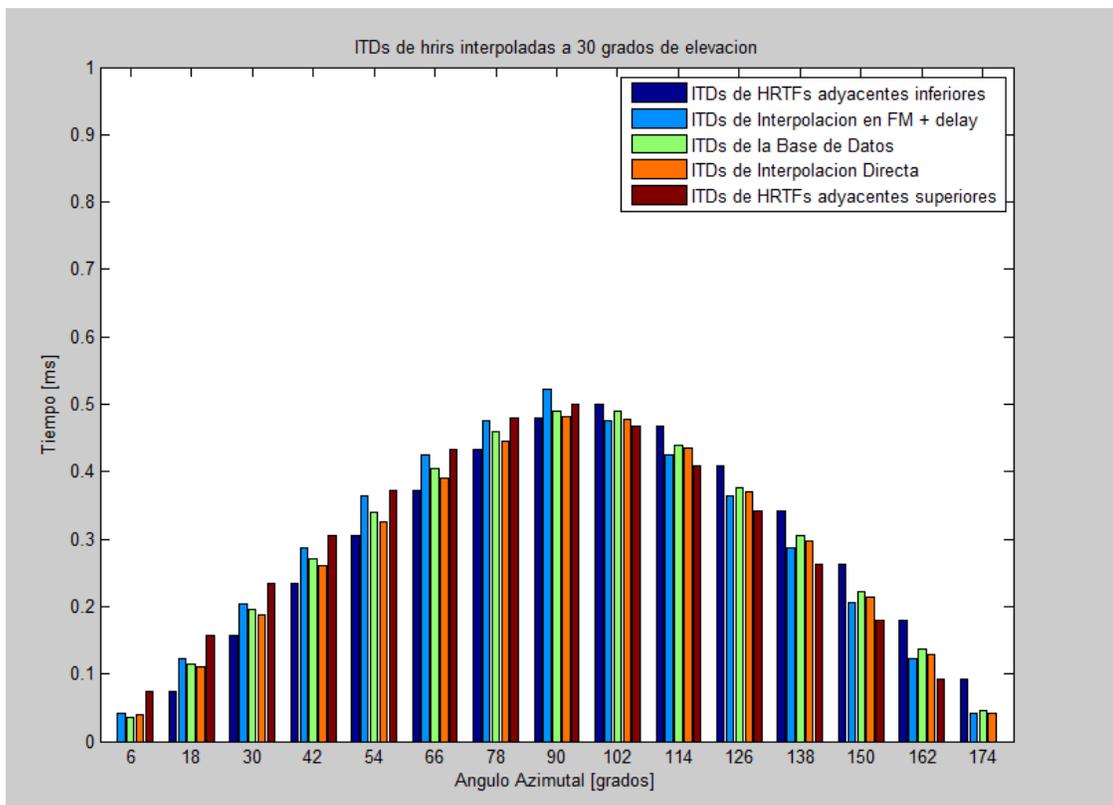


Figura 16. Gráfica de comparación entre las ITDs de HRTFs interpoladas e implícitas en la base de datos a 30° de elevación.

### 3.5.3 Prueba del Procesamiento.

Para desarrollar esta prueba se utilizo el script “hrtfProcess.m”, en este se auraliza una fuente monofónica usando la base de datos elegida por el usuario y escogiendo si se procesa con HRIRs en representación de fase mínima o mixta. Finalmente se grafican las respuestas en frecuencia de las HRTFs adyacentes con la interpolada. Las Figuras 17 y 18, contienen un ejemplo del script mencionado, teniendo en cuenta la base de datos y la componente de fase que se utilizara.

Datos de Entrada:

Ingrese elevación 34

Ingrese azimuth 163

Base de Datos a usar [kemar/IRCAM]: kemar

Representación en fase mínima o mixta [min/mix/mix2]: min/mix

Con este script se ve claramente como la interpolación en fase mínima es superior en calidad a la interpolación en representación de fase mixta.

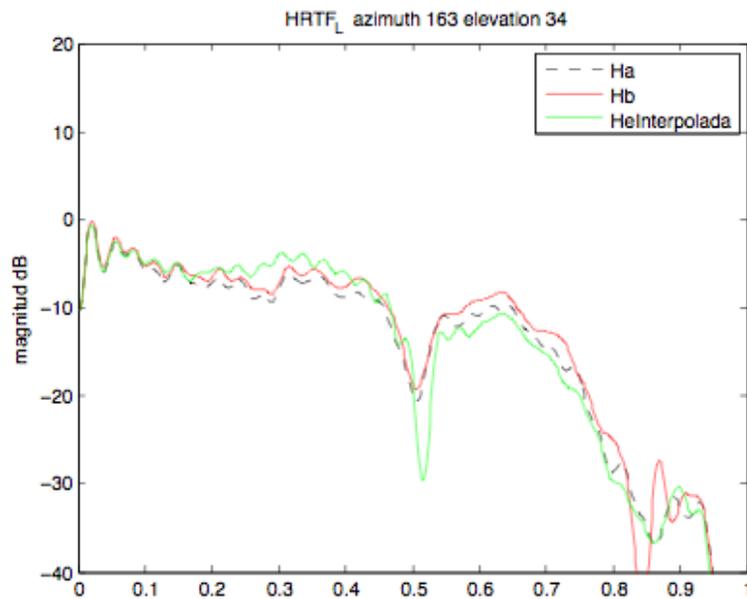


Figura 17. Comparación entre una HRIR interpolada, contra los puntos mas cercanos incluidos en la base de datos, utilizando su componente de fase mínima.

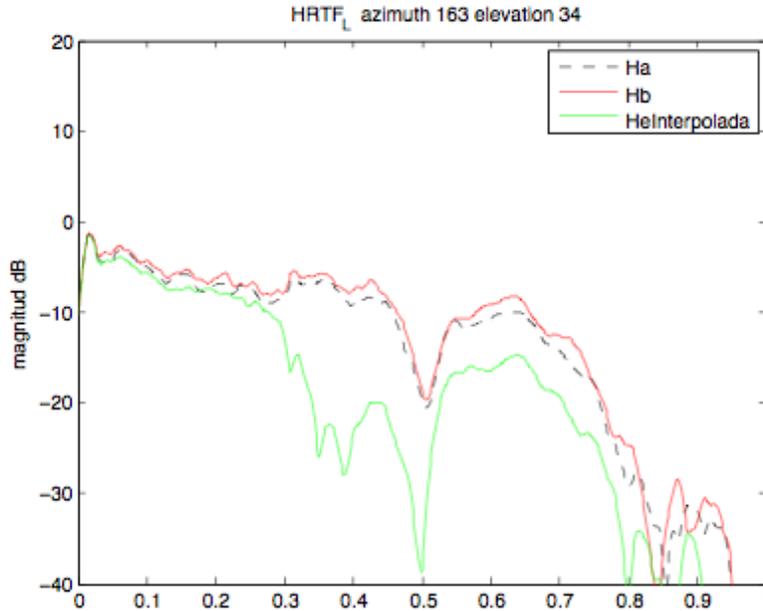


Figura 18. Comparación entre una HRIR interpolada, contra los puntos mas cercanos incluidos en la base de datos, utilizando su componente en fase mixta.

### 3.6 Implementación

Con base en los resultados obtenidos en el apartado 3.5 se tomaron decisiones determinantes para el esquema de procesamiento a implementar. En particular el modelo de Savioja para la estimación de las ITDs fue el más próximo a los métodos analíticos de estimación de las ITDs. Por otro lado el método de interpolar las HRIRs sin pre procesar conlleva a una magnitud incorrecta sin embargo interpolar las HRIRs en fase mínima produce una interpolación correcta del espectro de magnitud.

#### 3.6.1 Pre Procesamiento Utilizado con la Base de Datos.

- Modificación del Código de Brian Carty para obtener la base de datos en dos estructuras (archivo binario “.raw”)

Realizando ligeras modificaciones al “datapreparation.cpp” [24] y usando las clases propuestas en este proyecto, se logró escribir la base de datos en dos estructuras de respuestas al impulso como archivos binarios. Cada archivo pesa 368 kb, la idea detrás de este tipo de empaquetamiento de la base de datos es que al no ser archivos pesados pueden ser cargados desde el momento que se instancie el efecto en el programa anfitrión del plugin. De esta manera se evita tener que abrir y cerrar archivos para un punto indicado cada vez que se solicita una coordenada espacial nueva. Adicionalmente, se puede modificar para que la base de datos sea estática y por lo tanto compartida entre cada instancia del efecto binaural.

- Script de escritura de la base de datos IRCAM y Kemar en fase mínima o mixta como archivos de texto.

Otra manera de utilizar las bases de datos de HRTFs, la cual fue utilizada en una primera instancia pero reemplazada por la opción anteriormente mencionada. Consiste en convertir dicha base de datos en Archivos de texto, En Matlab el archivo HRIR2txt.m encontrado en la carpeta de entregas, escribe ya sea la base de datos IRCAM o Kemar en fase mínima o mixta como archivos de texto separados, los nombres de los archivos siguen la siguiente convención: pos\_elevacion\_azimuth.txt.

### **3.6.2 Procesamiento Utilizado.**

Luego de tener claros los conceptos mencionados en el análisis previo, se analiza el procesamiento que se utilizará en el desarrollo del plugin. En el diagrama de bloques presentado en la Figura19, se pueden ver los objetos utilizados en el procesamiento de la señal, cuya salida será una señal Binaural, que contiene las características de posición de acuerdo a los controles de entrada que el usuario asignará Cabe resaltar que este procesamiento se efectúa en tiempo real, sin embargo antes de mencionar lo utilizado en la programación del plugin, se mencionaran los resultados y pruebas realizadas en Matlab, para su posterior desarrollo en C++.

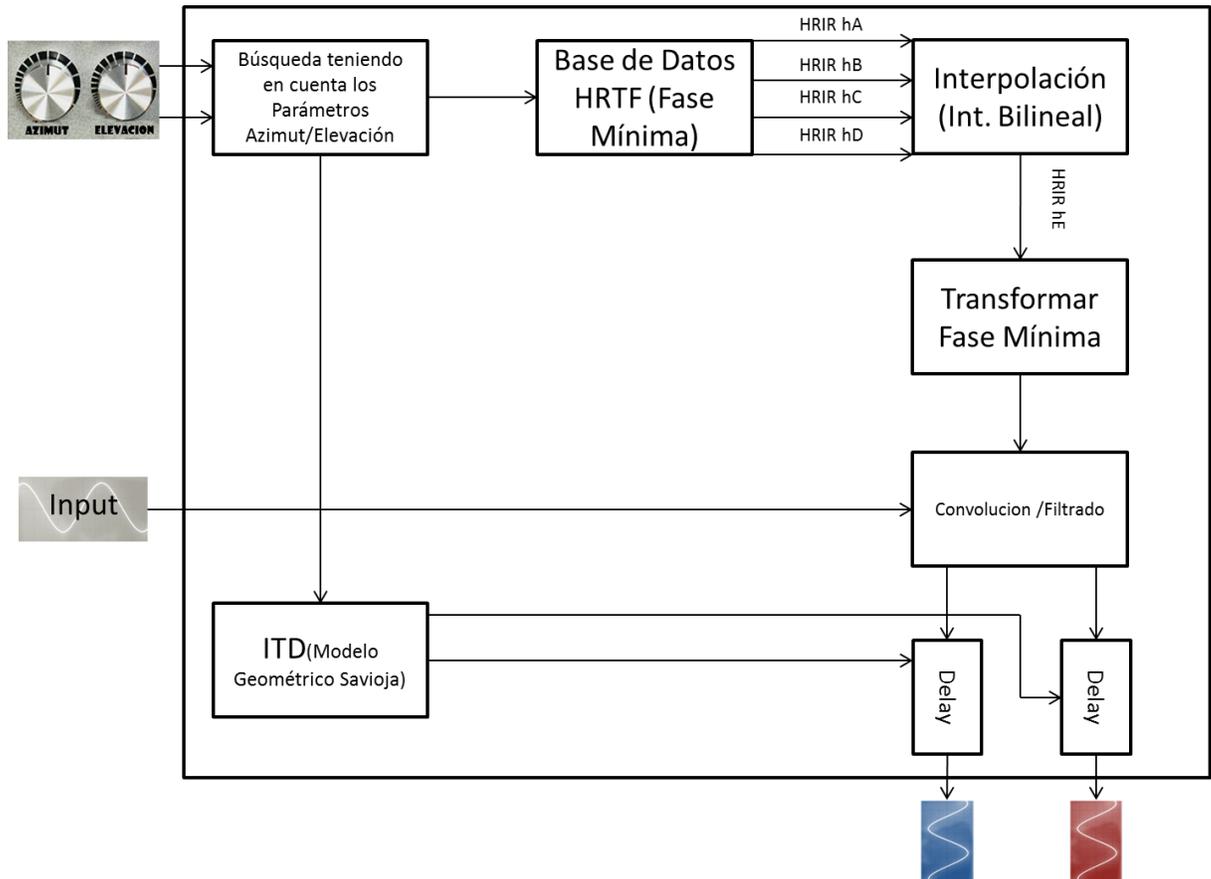


Figura 19. Diagrama de bloques del procesamiento utilizado en el desarrollo del plugin.

### 3.6.3 Desarrollo en MATLAB (Scripts).

Para la investigación planteada en este proyecto se desarrollaron diferentes scripts en Matlab, los cuales son un insumo importante para el análisis y la realización del trabajo, estos scripts se encuentran para su uso libre en la carpeta de entregables, y a continuación se describirá cada uno de los mismos para un mejor entendimiento.

- **hrtfprocess.m**

Este Script contiene todo el procesamiento propuesto en la Figura 10. Depende de varios datos de entrada: los ángulos de azimut y elevación, la base de datos que se desea utilizar kemar/IRCAM, realizar el filtrado con su componente en fase mínima o mixta. Los datos de salida son: la gráfica en magnitud de la HRIR Interpolada, contra las HRIR más cercanas implícitas en la base de datos, y la señal binaural resultante del filtrado entre una señal de entrada, y la HRIR interpolada.

- **kemarDataBaseReading.m / IrcamDataBaseReading.m**

Estas dos funciones permiten realizar la lectura de las bases de datos kemar/IRCAM, los datos de entrada son los ángulos de azimut y elevación, y como que resultado la función entrega la respuesta al impulso para la coordenada deseada.

- **ITD.m**

Esta función calcula las diferencias interaurales utilizando un modelos geométricos, dando la opción de elegir entre el modelo de Woodworth, Savioja, y Jot.

- **ITD\_cross.m**

Es una función mas, que ejecutada en matlab permite calcular las diferencias interaurales entre la HRTF izquierda y derecha, usando la base de datos KEMAR o IRCAM.

- **ITD\_phase.m**

Esta función calcula las diferencias interaurales de tiempo usando un método en el dominio de la frecuencia el cual consiste en hacer una regresión lineal en [1-5]kHz sobre la fase de exceso de la HRTF.

- **fdelay.m**

Esta es una clase en MATLAB que permite hacer retrasos fraccionales, luego de especificar el tipo de interpolación deseada (en el constructor) y el retraso equivalente en muestras usando el método `setDelay( )` (no necesariamente un numero entero), se procesa mediante la llamada a la función `process(signal)` .

- **getMinPhase.m**

Esta función es muy importante en el flujo de procesamiento, pues entrega la fase mínima utilizando la transformada de Hilbert, los datos de entrada son: el sistema de entrada “h”, y el dominio en el que se quiere recibir la fase mínima, ya sea en el dominio del tiempo `time`, o el dominio de la frecuencia `freq`

### 3.6.4 Desarrollo del VST

El VST `BinauralMoverCoMa3D` fue desarrollado bajo la API del SDK2.4, el Plugin fue escrito en el lenguaje de programación C++ y para este se utilizo la interface gráfica por defecto. El SDK2.4 brinda dos funciones que deben ser sobre cargadas, la primera de ellas es el “audio callback” nombrado “`processReplacing()`” o “`process()`”, y la segunda encargada de inicializar los atributos de la clase “binaural”, esta es la clase general del Plugin.

La tasa a la que se pueden actualizar los parámetros y por tanto las HRTFs necesarias para procesar un buffer de entrada de audio, es igual al tamaño del buffer, para este caso estamos inicializando los bloques de procesamiento para tener igual longitud a la de las respuestas al impulso del kemar (128 = 3ms). El incremento mínimo que tienen los parámetros de azimut y elevación son de 1 grado.

#### 3.6.4.1 Rutinas de Procesamiento del Plugin VST

- **processReplacing**

El efecto VST tiene la propiedad de ser usado como un insert en lugar de ser un efecto al cual se envíe una señal, por lo que la función que se debe sobre

escribir el VST SDK es el processReplacing() en lugar del process().

- **setParameter**

En el setParameter se llevan a cabo 5 tareas importantes, según las posiciones de los controles de azimut y elevación:

1. Se leen los 4 puntos que circunscriben a la HRTF deseada mediante dataBaseObj->read4()
2. Se interpola y se almacena la HRTF interpolada en los arreglos del efectos mediante databaseObj->interpolate(HRIR\_L,HRIR\_R).
3. Se actualiza la respuesta al impulso con la que se va convolucionar dentro del objeto para realizar este proceso mediante, convObj->setImpulse().
4. Se estima el ITD correspondiente mediante la fórmula del modelo esférico de Savioja.
5. Se elige, que lado retrasar y se sitúan los punteros de las líneas de retardo a sus ubicaciones correctas.

```
//-----  
void binaural::setParameter (VstInt32 index, float value)  
{  
    binauralProgram *pp;  
    if(curProgram >=5)  
        pp = &setting[getProgram()];  
  
    switch (index)  
    {  
        case AZIMUTH : if(curProgram >=5)pp->azi = value;  
                       azimuth=(int)(value*360.f);  
        break;  
        case ELEVATION :if(curProgram >=5) pp->ele= value;  
                       elevation=(int)(value*130.f-40.f); break;  
    }  
  
    dataBaseObj->read4(elevation,azimuth);  
    dataBaseObj->interpolate(hrir_L,hrir_R);  
    getMinPhase(hrir_L, hrir_L, hrirLength);  
    getMinPhase(hrir_R, hrir_R, hrirLength);  
    convObjL ->setImpulse(hrir_L);  
    convObjR ->setImpulse(hrir_R);  
  
    ITD =ITD_savioja(elevation,azimuth);  
  
    //Choose Side to delay  
    if(azimuth < 180 && azimuth >0){  
        dlineR->setDelay(ITD*44100.f);  
        dlineL->setDelay(0.f);  
    }  
    else{  
        dlineL->setDelay(ITD*44100.f);  
        dlineR->setDelay(0.f);  
    }  
}
```

Una nota importante a tener en cuenta en esta función de la clase “binaural”, es que no se deben realizar operaciones de escritura y lectura de archivos ( dataBaseObj->write( ) , dado que el plugin no responde correctamente cuando los parámetros del plug-in son automatizados.

### 3.6.4.2 Clases Desarrolladas.

- **Convolver**

Esta clase implementa el método Overlap Add para lograr la convolución entre la entrada de Audio y la HRIR, el método ha sido adaptado a nuestras necesidades por lo que solo se toman particiones de la señal de entrada iguales a la longitud en muestras de la HRIR(128 muestras). Esta clase no puede ser utilizada para convolución con respuestas al impulso de salas las cuales usualmente son más largas. Con ligeras modificaciones se podría lograr esto pero para los requerimientos de este plugin no es necesario.

Tabla 2. Características de la clase convolver.

<code>class convolver()</code>		Clase para realizar la convolución rápida de dos señales usando el método overlap add, esta clase permite particionar la señal de entrada en bloques de igual longitud con respuestas al impulso de longitud menor igual a estos.
return type	Métodos Públicos	Descripcion
	<code>convolver(int blockSize)</code>	Constructor de la clase, trabaja convoluciones de entrada/salida de tamaño blockSize, haciendo convoluciones internas del tamaño de la IR.
	<code>convolver(int blockSize, int IRlength)</code>	Constructor de la clase para realizar convoluciones de bloques de entrada mas grandes a los tamaños a de las IR.
<code>void</code>	<code>conv(float* input, float* output)</code>	Realiza la convolución de input con la respuesta al impulso guardada después de la llamada a setImpulse ( ) y la guarda en output.
<code>void</code>	<code>setImpulse(float* impulse)</code>	Método para actualizar la respuesta al impulso.

- **delay**

La clase delay en el archivo fdelay.cpp, es la clase que define la línea de retardo fraccional con interpolación Allpass utilizada en el VST. Es implementada mediante un buffer circular de un tamaño especificado como segundo argumento del constructor. Sus dos métodos mas importantes son setDelay(float ) y process() el primero se encarga de actualizar el delay de la línea de retardo y el segundo es el que entrega la señal retrasada. Para la aplicación dada en este caso es preciso recordar que los ITDs no llegan si quiera a 1ms de retardo lo que implica que es suficiente tener una línea de retardo de 44 muestras de longitud o incluso menos.

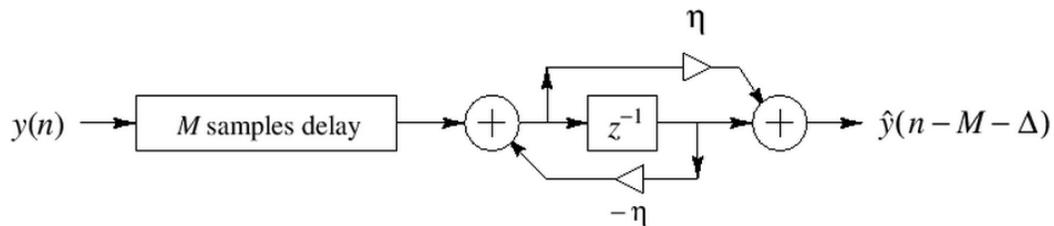


Figura 20. Diagrama de bloques del retraso fraccional. Esquema modificado de [4]

Tabla 3. Características de la clase delay.

class delay		Clase que implementa un delay fraccional usando interpolación allpass con warping, en un buffer circular.
return type	Métodos Públicos	Descripción
	delay(float initDelay)	Constructor de la clase que fija el delay en una línea de retardo de 44100 muestras
	delay(float initDelay, bufferSize)	Constructor de la clase que fijando el delay inicial en una línea de retardo de bufferSize muestras
void	setDelay(float newDelay)	Fija el nuevo delay en la línea de retardo
void	setDelayRelativeToBuffer(float del)	Fija el delay de la línea de retardo independiente de la longitud del buffer circular tomando un argumento en el rango [0 1]
float	process(float sample)	Ingresa nueva muestra a la línea de retardo y devuelve la salida actual de la línea de retardo
int	getSampDelay( )	Devuelve la parte entera del delay actual (numero de muestras de retraso)
float	getFracDelay()	Devuelve la parte fraccional del delay actual

- **readHRIR**

La clase readHRIR es un clase diseñada para leer e interpolar la base de Datos kemar del MIT, después de ser escrita en dos archivos binarios "datal.raw" y "datar.raw" tal y como se hace en el ejecutable compilado de datapreparation.cpp [4].

Tabla 4. Características de la clase readHRIR.

class readHRIR		Clase para la lectura e interpolación de la base de datos escrita en archivos binarios.
return type	Métodos Públicos	Descripción
	readHRIR( )	Carga la base de datos del MIT (kemar), de los archivos binarios nombrados datal.raw y datar.raw
void	read(int ele, int azi)	Lee la HRIR mas cercana a las coordenadas en grados
void	read4(int ele, int azi)	Lee cuatro HRIR que circunscriben a las coordenadas en grados
void	interpolate(float* hL, float* hR)	Interpola los 4 HRIRs y guarda el resultado en los arreglos hL y hR de al menos 128 muestras de amplios, después de la llamada a read4(int ele, int azi)
void	writeHRIR( )	Escribe la HRIR como texto después de la llamada a read(ele,azi)
void	write4HRIR( )	Escribe las 4 HRIR como archivos de texto, los nombres relacionados a las HRIRs son: low1 ->ha, low2 ->hb, high1 ->hd, high2 ->hc

- **getMinPhase - Hilbert**

En general una combinación lineal de sistemas en fase mínima, no da un sistema en fase mínima. Razón por la cual a pesar de tener la base de datos escrita en fase mínima, será necesario repetir el cambio a fase mínima de la HRIR interpolada. getMinPhase es una función que se encarga de retornar el equivalente en fase mínima de un sistema. Internamente esta función depende de la transformada de Hilbert la cual, se implementa en el dominio de la frecuencia como un corrimiento en fase de -90 grados.

Para asegurar que esta función converja es necesario que el algoritmo de la magnitud también, por tal razón no pueda existir un bin en el dominio de la frecuencia donde su magnitud sea cero. Una solución simple es sumar un offset (constante), lo más pequeño posible.

### 3.6.4.3 Librerías Utilizadas.

- **Librería FFTW**

Actualmente todas las clases que impliquen hacer transformadas Discretas de Fourier utilizan esta librería. La fftw funciona definiendo planes en los cuales se establece que se transforma y en donde se guarda, esto es crucial y por eso la librería cuenta con diferentes planes para diferentes tipos de datos y ordenamiento de los mismo. En nuestro caso, la señal de entrada es real y el espectro de salida tiene simetría Hermitiana, (magnitud simétrica y fase anti simétrica). Existen dos planes especiales para este tipo de escenario, el plan r2c con su inversa c2r y los planes r2r. La diferencia con los planes real a real es que en este caso se usa un formato llamado, Half Complex, donde los datos complejos son organizados de manera que la primera

mitad son las componentes reales y la segunda los imaginarios de manera descendente [25].

- **Librería libsndfile**

Esta librería es necesaria para abrir y leer archivos de tipo .WAV, y con esta fue posible leer la base de datos en archivos de audio y ser reescrita en archivos binario, disponible en [26].

## **4. Descripción General de la Prueba Subjetiva**

Según Soren [27], toda prueba subjetiva se divide en tres dominios que caracterizan las etapas del desarrollo experimental y que categorizan las variables del mismo. El primero de estos, es el dominio físico, donde se genera el estímulo sonoro que genera una reacción sensorial pasando así al dominio perceptual en que el individuo interpreta psicoacústica mente esta información. Luego de estas dos etapas viene el dominio afectivo, donde se cuantifica el estímulo subjetivo.

En vista que las etapas o dominios que conforman una prueba subjetiva son complejas, las variables experimentales son con frecuencia numerosas. Para poder diseñar un experimento es útil asociar las variables en categorías. Las categorías que mejor agrupan las variable de experimentos de este tipo han sido sugeridas por [27] y son las siguientes: Variables dependientes(variables de interés para el experimentador), variables independientes(variables completamente controladas por el experimentador), y variables aleatorias (variables no tomadas en cuenta en el experimento pero que tienen incidencia en los resultados).

En el experimento descrito las variables dependientes fueron: el reconocimiento espacial, la concordancia visual-auditiva y el reconocimiento de cambios continuos de la posición de una fuente sonora.

Las variables independientes fueron los controles en la interfaz gráfica del plugin, es decir los parámetros de azimuth y elevación al igual que los archivos anecoicos seleccionados para el experimento.

En el caso de las variables aleatorias el ruido de fondo, el sistema de reproducción y la fisiología de los individuos son las responsables de gran parte de la variabilidad del experimento.

A lo largo de esta sección se presentarán los resultados obtenidos durante el proceso de evaluación subjetiva. Esta prueba consiste en una encuesta realizada a una muestra de 30 individuos de la población de estudiantes de ingeniera de sonido de la universidad San Buenaventura seccional Medellín. Luego de una prueba piloto realizada el mes de septiembre de 2013 a amigos y personas conocidas, se determinó escoger una muestra de la población de ingenieros de sonido para economizar tiempo, dado que los individuos encuestados no familiarizados con la terminología y poco entrenamiento auditivo solicitaban repetir las explicaciones y preguntas.

La encuesta final se dividió en dos secciones, la primera de 6 preguntas que evalúan el comportamiento estático (con parámetros constantes) del Plugin, la segunda de 4 que evalúa directamente el método de interpolación bilineal realizando finos desplazamientos de los parámetros del plugin y por ende de la fuente virtual. En el anexo 2 se encuentra la encuesta llevada a cabo.

Estadísticamente se ha fijado un nivel de confianza de 95 % para la estimación de la media, en las preguntas de la primera sección y para la proporción de la segunda. Con un muestra de 30 estudiantes tomados de una población de 314 estudiantes de ingeniería de sonido, se obtiene un factor de elevación aproximado de 10.5.

#### 4.1. Descripción Sección 1: Reconocimiento Discreto de Fuentes

La primera sección está dividida en dos partes. En esta se indaga sobre la calidad de percepción de fuentes en puntos fijos del espacio. La primera parte está compuesta por un conjunto de 6 preguntas que evalúan la localización en una escala de 1 a 5 (ver Figura 12) del posicionamiento visual de la fuente con la percepción espacial de ella. La parte 2 en cambio es una prueba donde el origen de la fuente lo debe estimar el individuo y consiste en decidir hacia dónde se ha movido la fuente con una opción de repuesta tipo A B (ver Figura 13).

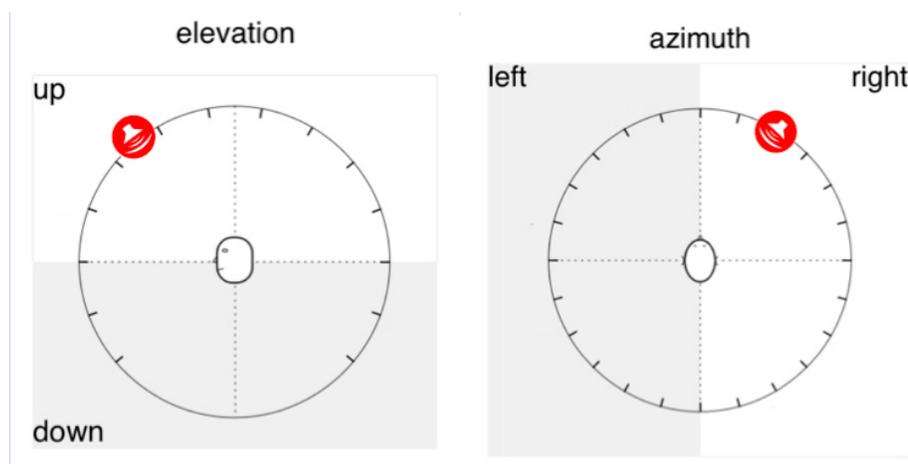


Figura 21. Tipo de figuras utilizadas en la primera parte de la sección 1, preguntas 1 a 6.

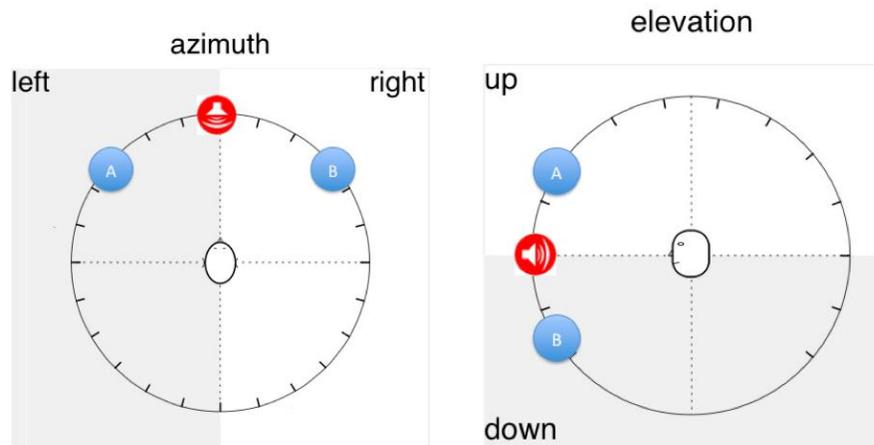


Figura 22. Tipo de figuras utilizadas en la segunda parte de la sección 1, preguntas 7 a 12.

## 4.2. Descripción Sección 2: Reconocimiento de Continuidad

En la segunda sección de la prueba se evalúa si el sujeto percibe una mayor continuidad en el desplazamiento de la fuente cuando se lleva a cabo una interpolación espacial. Esta sección de la prueba consiste en reproducir dos archivos de audio auralizados con duración de 9 segundos, los cuales realizan la misma trayectoria binaural pero en la cual una de ellas pasa únicamente por puntos específicos de la base de datos, mientras la segunda es una interpolación fina de puntos intermedios, es decir una automatización suave de los parámetros del plugin. De las cuatro preguntas de esta sección dos representan un desplazamiento horizontal y dos un desplazamiento vertical, las preguntas 1 y 2 utilizan la interpretación de una flauta como fuente sonora, y las preguntas 3 y 4 utilizan la narración de un locutor como fuente sonora.

## 4.3. Análisis de Datos

En los siguientes apartados, se presentan los resultados de la prueba subjetiva desarrollada, realizando una descripción estadística de cada una de las secciones propuestas en la encuesta.

### 4.3.1. Sección 1: Reconocimiento Discreto de Precedencia

En la primera parte del experimento, en la cual se juzga el grado de concordancia de la imagen con la percepción auditiva se puede ver que los resultados son bastante fluctuantes dado que la desviación estándar en las preguntas 1 a 6 es grande (coeficiente de variación C.V es mayor a 30 ver tablas 4 y 5 ). Por otro lado el experimento parece estar bien diseñado, dado que en el Gráfico el intervalo de confianza para un nivel de confianza del 95% es considerablemente estrecho (menor a 1), lo cual habla de la repetibilidad del experimento. Dicho de otra manera de los 314 estudiante matriculados en el programa de sonido en el periodo 2 del 2013, si se tomaran el 95% de todas las formas de elegir una muestra de tamaño 30, es decir  $6.92665 \times 10^4$  combinaciones, se podría estar seguro de que las medias muestrales para cada una de las preguntas estaría capturado en el intervalo de confianza ilustrado en la gráfica mostrada. Lo que significa que habría bastante variabilidad en las respuestas de esta parte del experimento.

Sin embargo la mayoría de la muestra evaluó en más de 3 cada una de las preguntas dado que la media para cada pregunta excede el valor de 3 en la escala de 1 a 5 y el porcentaje de la muestra que puntuó por encima del puntaje de corte (3 en la escala 1 a 5) fue mayor a 65% (ver percentiles en la tabla 4), lo que se podría interpretar como que todas las personas aunque no satisfaciendo sus expectativas sonoras de la auralización entregada reconoce que la fuente sonora está ubicada en la cercanía de la fuente mostrada en la imagen. La mejor puntuación de esta prueba fuera de las expectativas fue la pregunta 5 donde la posición de la fuente se encontraba en la parte inferior del eje de elevación. También se puede notar lo difícil que es reconocer desplazamientos cortos (pregunta 1 y 4).

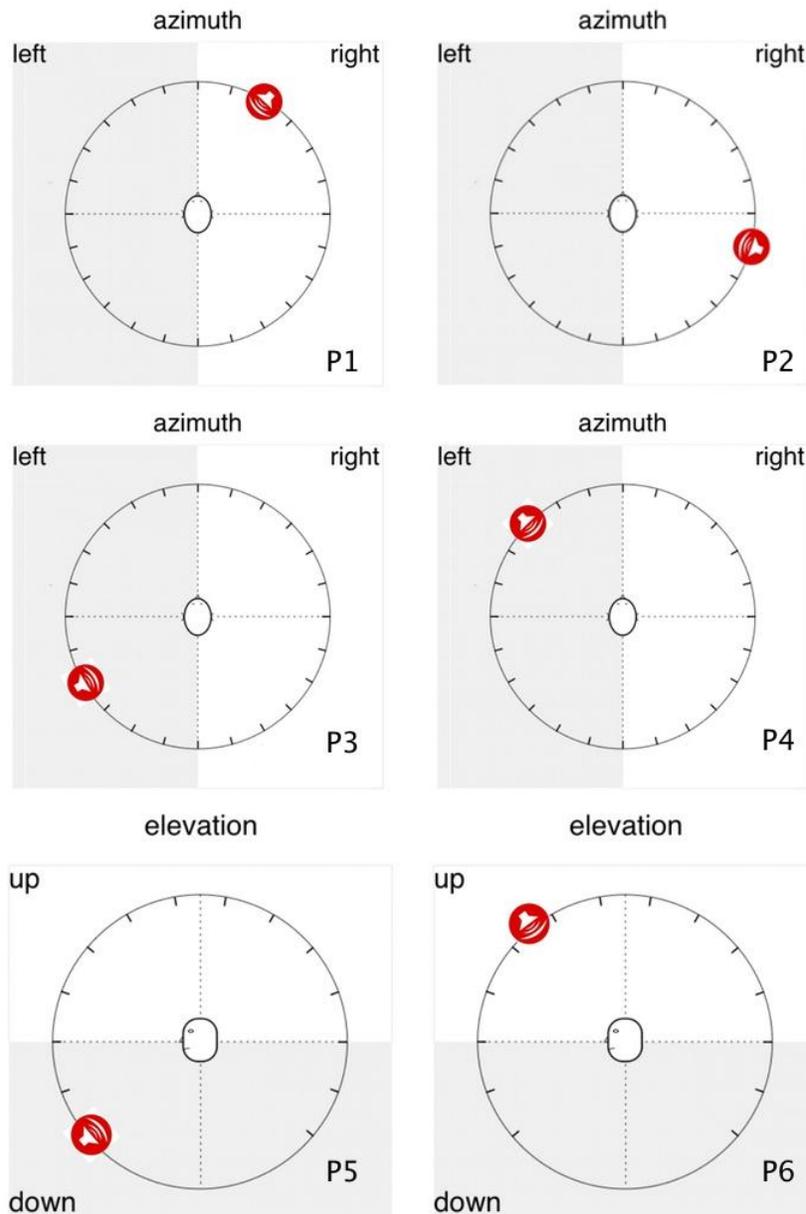
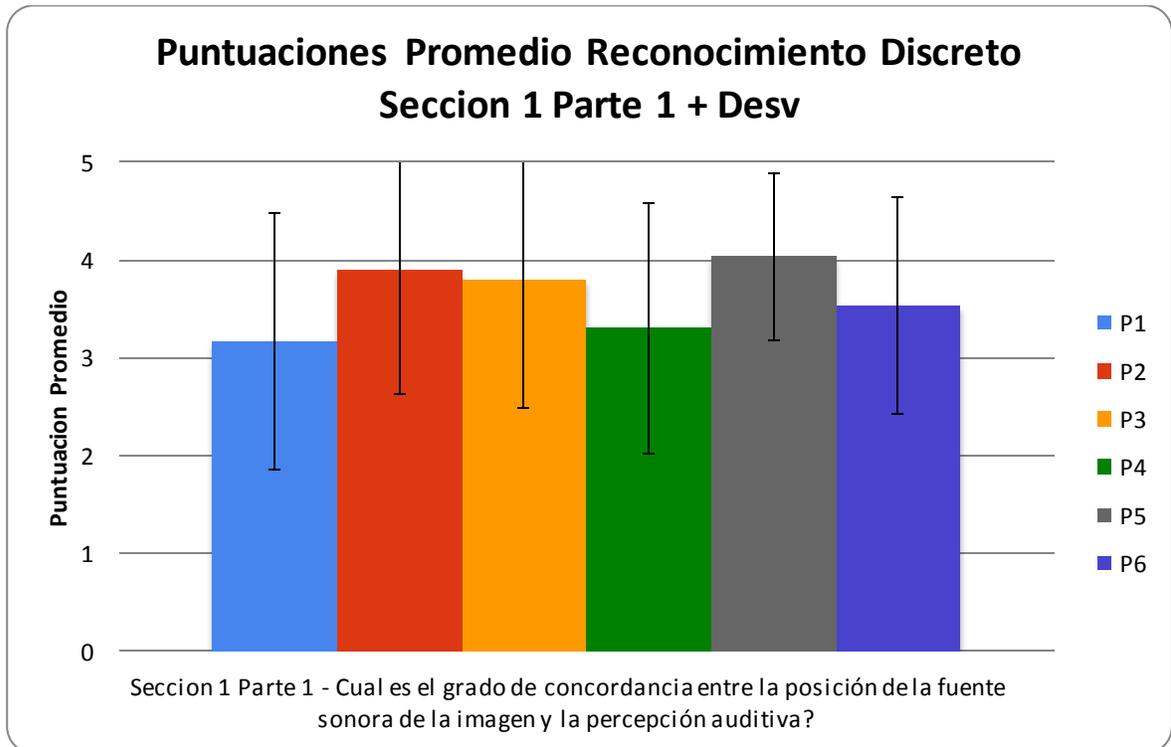
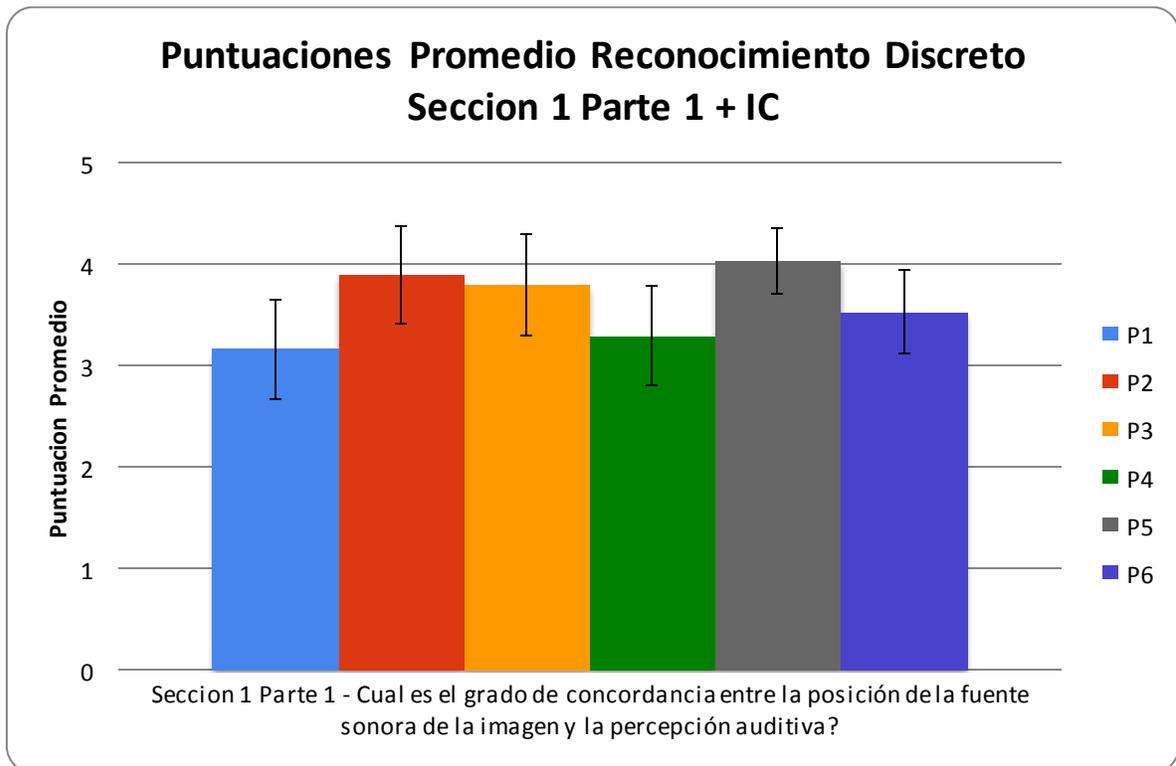


Figura 23. Preguntas 1 a 6 de la parte 1 de la sección 1.



Gráfica 1. Distribución de las puntuaciones promedio, preguntas 1 a 6 de reconocimiento puntual, parte 1.



Gráfica 2. Intervalos de confianza correspondientes a las preguntas 1 a 6 de reconocimiento puntual, parte 1.

Tabla 5. Análisis estadístico realizado para la parte 1 sección 1 de la prueba.

<b>Reconocimiento Discreto de Precedencia - Sección 1</b>							
Parte 1 - Cual es el grado de concordancia entre la posición de la fuente sonora de la imagen y la percepción auditiva?							
	Media Muestral	Desviación	C.V	IC - Normal Std	IC - T Student	Percentil Punto Corte	1-Percentil Punto Corte
P1	3,166666667	1,315250948	41,53424048	0,470647859	0,491122776	34,4	65,6
P2	3,9	1,268993628	32,53829815	0,454095193	0,473850008	17,2	82,8
P3	3,8	1,323527158	34,82966205	0,473609408	0,494213163	20,6	79,4
P4	3,3	1,290549201	39,10755156	0,461808615	0,481898992	31	69
P5	4,033333333	0,850287308	21,0815035	0,304265814	0,317502499	6,8	93,2
P6	3,533333333	1,105888107	31,29872002	0,395729705	0,412945406	27,5	72,5

En la segunda parte del experimento se trabajaron preguntas con respuestas tipo A-B la variable aleatoria discreta en este caso es binomial. Analizando las gráficas de esta sección de la prueba subjetiva se puede notar una vez más que la muestra pasa esta prueba dado que la proporción de aciertos de todas las preguntas excede el 50%. También se puede concluir que esta prueba es más coherente dado que todos los sujetos tienden a responder de la misma manera las preguntas (todos tienden a equivocarse en iguales proporciones en las mismas preguntas) por que la variabilidad de todas las preguntas tiende a ser muy similar. Otro hecho importante en esta parte de la evaluación es que las preguntas de lateralización con referencias en 90 y -90 grados son las menos acertadas mientras que las preguntas con referencias en 0 y 180 son las más concluyentes. Además se hace evidente el desafío en reconocer cambios en el plano de elevación, como se puede notar en la pregunta 6 con puntuación menor a todas las demás.

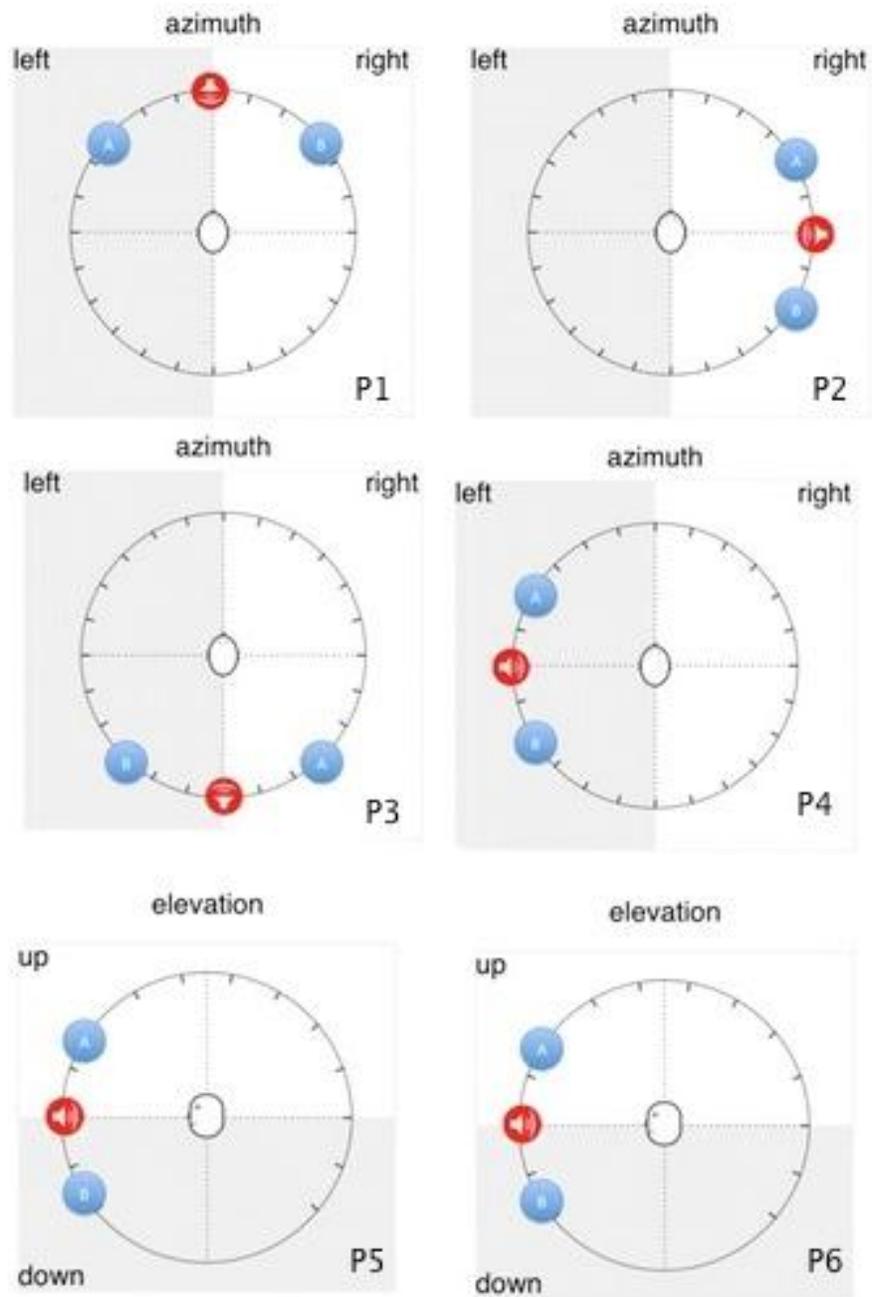
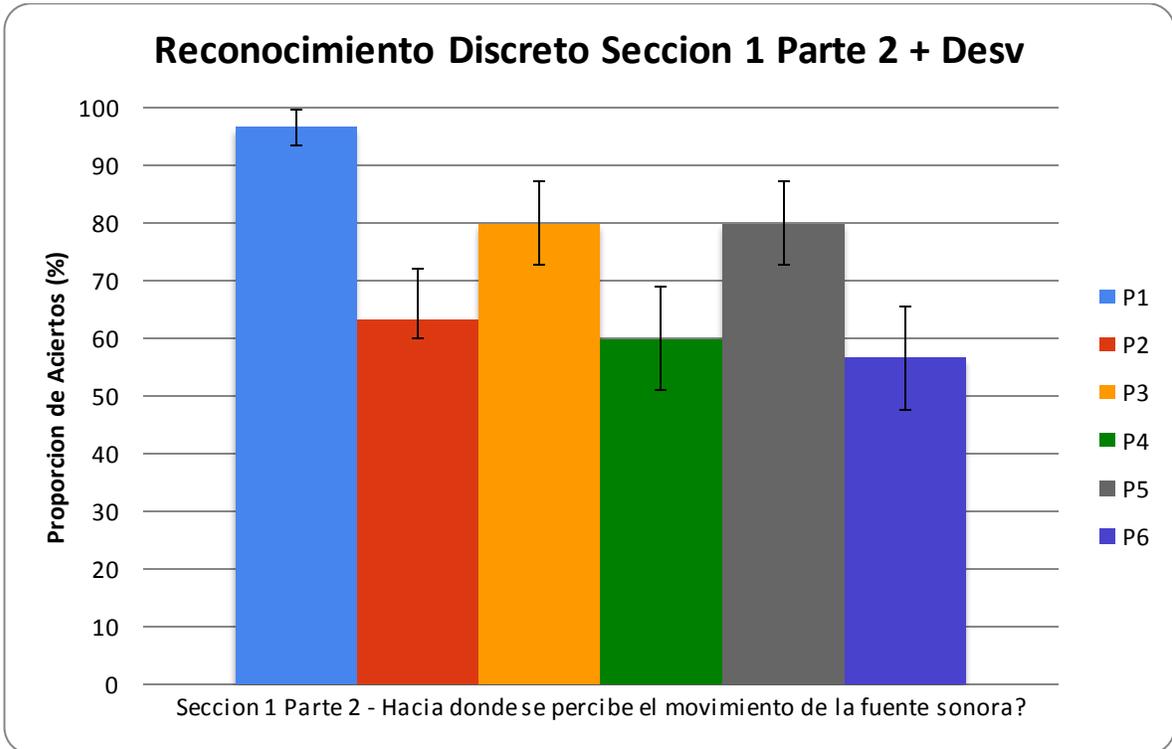
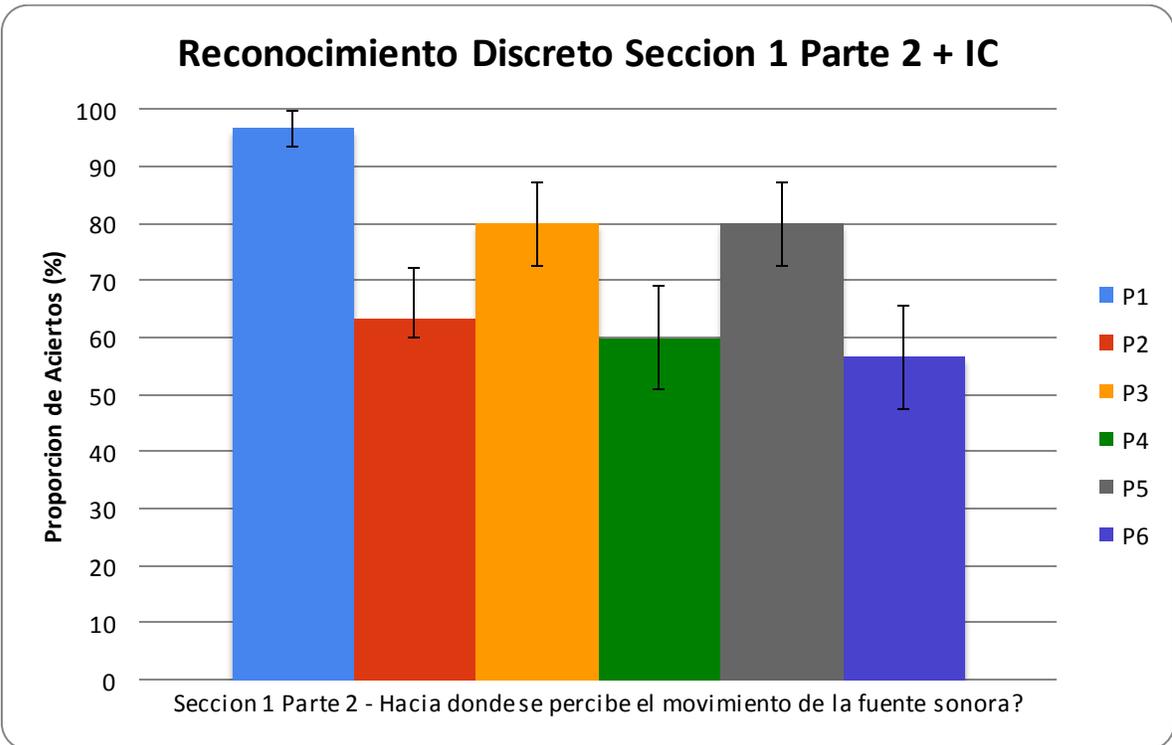


Figura 24. Preguntas 1 a 6 de la segunda parte de la sección 1.



Gráfica 3. Distribución de las puntuaciones promedio, preguntas 1 a 6 de reconocimiento puntual, parte 2.



Gráfica 4.. Intervalos de confianza correspondientes a las preguntas 1 a 6 de reconocimiento puntual, parte 2

Tabla 6. Análisis estadístico realizado para la parte 2 seccion1

<b>Reconocimiento Discreto de Precedencia - Seccion 1</b>					
Parte 2 - Hacia donde se percibe el movimiento de la fuente sonora?					
	Proporcion (%)	Desviacion(%)	C.V	IC - Norm (%)	Aciertos
P1	96,66666667	3,277306934	3	3,277306934	29
P2	63,33333333	8,798147953	14	8,798147953	19
P3	80	7,302967433	9	7,302967433	24
P4	60	8,94427191	15	8,94427191	18
P5	80	7,302967433	9	7,302967433	24
P6	56,66666667	9,047201327	16	9,047201327	17

Una observación que no se manifiesta en los gráficos es que en las preguntas de elevación los sujetos encuestados la mayoría del tiempo pedían repetir la primera pregunta de elevación (60 grados), cuando escuchaban la segunda (-40 grados) y en muchos casos solicitaban cambiar su respuesta anterior.

#### **4.3.2. Seccion 2: Reconocimiento Continuo de Precedencia**

En la prueba piloto se puede notar como los sujetos encuestados no respondían bien a trayectorias rápidas, es decir les parecía de comportamiento caótico y poco inteligible. Es preciso recordar que el plugin carece de la capacidad de variar el radio de la esfera sobre el que se mueve la fuente, lo que se traduce en una relación de desplazamiento a ángulo pequeña. En razón de esto la trayectoria realizada por la fuente sonora se ajustó a 9 segundos o más.

En los Gráficos 14 y 15 sin embargo a pesar de la dificultad que significaba para los sujetos encuestados percibir las diferencias, se insinúa la mejor calidad sonora de los audios interpolados horizontalmente. Esto se hace evidente por los dos picos del Gráfico 14 (primera y tercera pregunta) correspondientes a las pruebas con automatizaciones azimutales de los parámetros del plugin. Se puede ver que más del 80% de la muestra logra percibir la sensación de continuidad de la fuente satisfactoriamente cuando se trata de un desplazamiento horizontal. Sin embargo las diferencias entre columnas adyacentes muestra la dificultad que presentaban los individuos para apreciar los mismos cambios en el plano de elevación.

La desviación de la proporción refleja la cantidad de desacuerdo que hay en que la proporción dada no sea verdadera por parte de la muestra poblacional. La desviación es más pequeña para las puntuaciones altas (preguntas 1 y 3 Gráfica 14). Esto refleja una pregunta más clara (fácil y rápidas de contestar) para el grupo en general a comparación de las preguntas de menor puntuación. Esto también se puede respaldar tomando en cuenta el tiempo adicional que se tomaban los encuestados para decidir su respuesta en las preguntas 2 y 4.

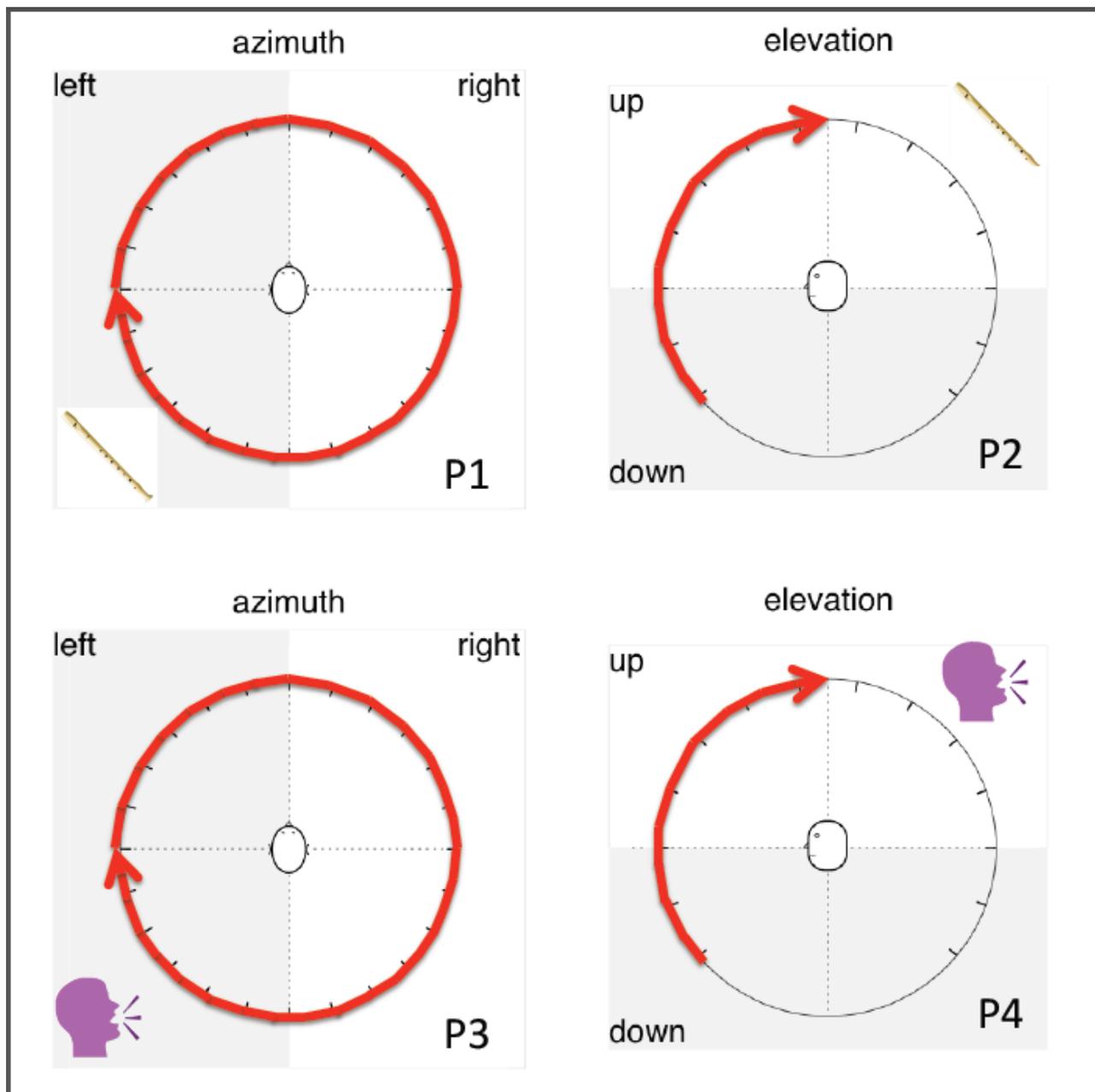
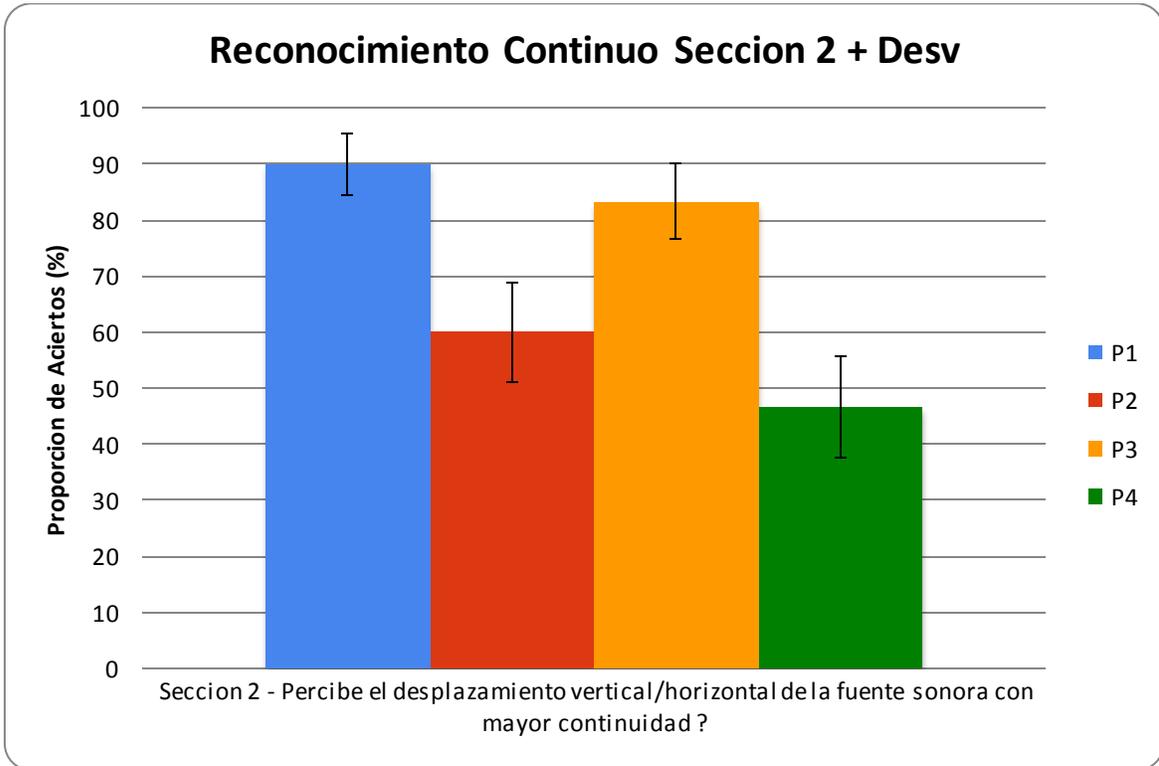
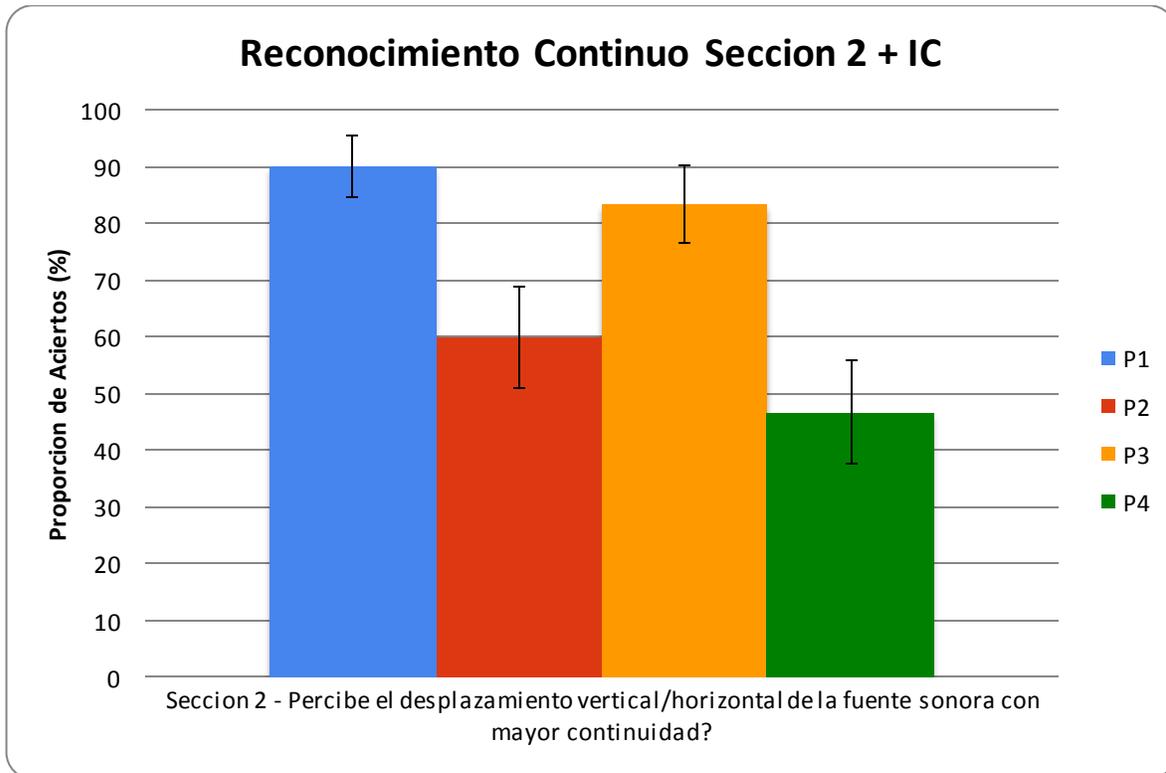


Figura 25. Preguntas 1 a 4 de la sección 2.



Gráfica 5. Desviación estándar correspondientes a las preguntas 1 a 4 de reconocimiento continuo.



Gráfica 6. Intervalos de confianza correspondientes a las preguntas 1 a 4 de reconocimiento continuo.

Tabla 7. Análisis estadístico realizado para la sección 2 de la prueba.

<b>Reconocimiento Continuo (interpolación) - Sección 2</b>					
Parte 1 - Percibe el desplazamiento horizontal con mayor continuidad de la fuente sonora?					
	Proporcion (%)	Desviacion (%)	C.V	IC - Norm (%)	Aciertos
P1	90,0	5,5	6	5,48	27
P2	60,0	8,9	15	8,94	18
P3	83,3	6,8	8	6,80	25
P4	46,7	9,1	20	9,11	14

## 5. Conclusiones

De acuerdo al experimento realizado, la primera parte de la primera sección fue la prueba de más variabilidad (Coeficientes de Variación Mayores) entre la muestra, no obstante la media fue superior a 3 en la escala de 1 a 5. Por lo que en la sección 1 del experimento todas las preguntas son aprobadas si consideramos el puntaje de aprobación 3 para las preguntas de puntuación 1 a 5 y 60% para la proporción de acierto en las preguntas tipo A B. En general la repetibilidad del experimento es buena y se puede esperar resultados semejantes de otra muestra, a excepción de la primera parte de la sección 1 que podría considerar algunos cambios de diseño.

La peor puntuación se presentó en la pregunta de elevación en la sección de reconocimiento continuo, que fue la única pregunta con una proporción de aciertos inferior a 50%. Las demás preguntas de esta sección tenían una proporción de aciertos de 60% o más.

A partir de las pruebas realizadas en MATLAB se puede comprobar de qué manera se interpola fielmente las HRTFs. En primer lugar la interpolación de las respuestas al impulso asociadas a la cabeza en representación de fase mínima, dan una interpolación de la magnitud de la HRTF correcta. La interpolación bilineal de las magnitudes también, genera un espectro de magnitud interpolado correctamente. Haciendo válido que dentro del procesamiento, se utilicen solo las magnitudes para realizar el método de interpolación.

Para la base de datos kemar el modelo esférico que más se asemeja a los ITDs calculados a partir de los datos (método de correlación cruzada y regresión lineal) es el modelo de Savioja. Esto se pudo comprobar con las pruebas realizadas en el apartado 3.5.1 (ver Figuras 10, 11, 12), donde se observó que las diferencias son pequeñas entre este modelo geométrico en particular y los métodos analíticos como el de correlación cruzada. Este modelo sin embargo necesita ser calibrado para asemejarse aún más a los ITDs extraídos de la base de datos como se puede observar en las Gráficas 6 y 7 y no sobrepasar el valor de las ITDs propias de las HRTFs que circunscriben el punto deseado.

Es indispensable tomar en cuenta varios refinamientos en el proceso de síntesis binaural, en primer lugar se ha concluido que el factor de profundidad que se logra combatiendo el efecto de percepción al interior de la cabeza, es crucial para enriquecer la percepción binaural. Implementar una reverberación binaural sería de gran beneficio y probablemente daría mejores resultados en pruebas subjetivas (incrementa la relación de desplazamiento ángulo), en las preguntas 1 y 4 de la primera partes de la sección de reconocimiento discreto se puede evidenciar esto.

## 6. Trabajos Futuros

Analizando herramientas existentes como el plugin “Panorama”, desarrollado por Wave Arts, empresa fundada por Bill Gardner, y “H3D” desarrollado por Long Cat Audio, se puede diseñar una interfaz gráfica más versátil e intuitiva que permita el movimiento y espacialización de una fuente sonora.

Un trabajo importante sería desarrollar un algoritmo de externalización, que mitigue el efecto “In Head Localization”, producido por la utilización de auriculares, siendo este paso importante, si se desea enfocar esta herramienta a dispositivos móviles.

La reverberación y la caracterización de un recinto, es importante para recrear fielmente la posición de la fuente sonora dentro de un espacio determinado, esta herramienta permitiría recrear espacios virtuales, muy útil para producciones multimediales, videojuegos, etc.

La posibilidad de reproducir una señal binaural por medio de un par de altavoces, o el estudio de los filtros de cancelación “Canceladores de Crosstalk”, esto permitiría que la herramienta pueda ser utilizada como parte de un sistema de sonido multimedia. Existen sistemas de reproducción de audio 3D como el OPSODIS que implementan estos filtros para reproducir una señal binaural, pero que no se acomodan al presupuesto y el objetivo de todas las producciones audio visuales.

Una característica importante en el movimiento de una fuente, es el efecto Doppler causado por la fuente sonora, incluir un módulo que permita el cálculo y la reproducción de dicho efecto, recreara de manera más fiel el movimiento de la fuente sonora virtual.

Extender la clase del delay fraccional para dar soporte a mas métodos de interpolación en la línea del retardo, como las que se mencionan en [17].

Probar el modelo funcional de fase para la estimación de las ITD calibrado en bajas frecuencias como sugiere Lazzarini [28]. En este artículo se sugiere un método más efectivo que el de fase mínima y retraso temporal.

## 7. Bibliografía

- [1] F. Alton Everest, *THE MASTER HANDBOOK OF ACOUSTICS*.
- [2] Antonio Jose Moya Diaz, "Sonido 3D Ingenieria Acústica," ETSIIT, Granada, 2008.
- [3] Matti Karjalainen Jyri Huopaniemi. (1996, June) HRTF FILTER DESIGN BASED ON AUDITORY CRITERIA.
- [4] Richard Boulanger, *The Audio Programming Book*: Max MATHEWS.
- [5] IRCAM instituto de investigación sobre acústica y música. IRCAM HRTF DATA BASE. [Online]. HYPERLINK "<http://recherche.ircam.fr/equipements/salles/listen/>"
- [6] MIT. HRTF Measurements of a KEMAR Dummy-Head Microphone. [Online]. HYPERLINK "<http://sound.media.mit.edu/resources/KEMAR.html>"
- [7] Keith Martin Bill Garner. (1994, mayo) HRTF Measurement of a KEMAR Dummy-Head Microphone.
- [8] William G. Gardner. (1997, September) 3-D Audio Using Loudspeakers.
- [9] Jyri Huopaniemi, "Virtual Acoustics and 3D Sound in Multimedia Signal Processing," Helsinki University of technology Laboratory of Acoustics and audio signal processing, Espoo, Finland, 1999.
- [10] IRCAM. (2013, Abril) LISTEN HRTF DATABASE. [Online]. HYPERLINK "<http://recherche.ircam.fr/equipements/salles/listen/>"
- [11] Karmelo Usategi de la Peña. (2010) Procesador de sonido y estudio de métodos de interpolación de la localización de fuentes basados en HRTFs para la generación de audio 3D.
- [12] the CIPIC Laboratory. (2013, febrero) The CIPIC HRTF Database. [Online]. HYPERLINK "<http://interface.cipic.ucdavis.edu/sound/hrtf.html>"
- [13] FIU DSP Lab. HRTF/Anthropometric Measurement Database. [Online]. HYPERLINK "<http://dsp.eng.fiu.edu/HRTFDB/main.htm>"
- [14] LUIZ WAGNER P. BISCAINHO, PAULO SERGIO R. DINIZ FABIO P. FREELAND. EFFICIENT HRTF INTERPOLATION IN 3D MOVING SOUND.
- [15] M. de Sousa Gustavo H. and Queiroz Marcelo. Two approaches for HRTF interpolation.
- [16] Pedro Godoy Alvarez and Cristobal Ilabaca Grez, "Implementacion de un espacio sonoro virtual mediante reproduccion binaural," Universidad Tecnologica de Chile INACAP, Santiago de Chile, Tesis de Titulación 2009.
- [17] Anne Sédès David Doukhan. A binaural synthesis external for Pure Data.
- [18] S. Mehrgardt y V. Mellert, "Transformation characteristics of the external human ear," *Journal of the Acoustic Society of America*, vol. 61, no. 6, Junio 1977.
- [19] Remy Muller Vincent Gourard. (2003, junio) Real-Time audio plugin architectures.
- [20] Jan Plogsties, Soren Krarup Olesen, Flemming Christensen, Henrik Moller Pauli Minnaar. The Interaural Time Difference in Binaural Synthesis.
- [21] Peter Balazs, Bernhard Laback Piotr Majdak, "Multiple exponential Sweep Method for fast measurement of Head Related transfer functions," *AES*, 2007.

- [22] Veronique Larcher, Oliver Warusfel Jean Marc Jot, "Digital signal processing issues in the context of binaural and transaural stereophony".
- [23] Young Cheol Park, Dae Hee Youn Tacksung Choi. (2006, may) Efficient Out of Head Localization System for Mobile Applications.
- [24] Brian Carty, "Movements in Binaural Space: Issues in HRTF," NUI Maynooth, 2010.
- [25] FFTW. FFTW. [Online]. HYPERLINK "http://www.fftw.org/download.html"  
<http://www.fftw.org/download.html>
- [26] libsndfile. Mega Nerd. [Online]. HYPERLINK "http://www.mega-nerd.com/libsndfile/"  
<http://www.mega-nerd.com/libsndfile/>
- [27] Soren Bech Ban, *Perceptual audio evaluator - Theory Method and application*. Dinamarca: John Wiley & Sons ltd.
- [28] Victor Lazzarini Brian Carty, "Frequency-domain interpolation of empirical HRTF data," AES, Mayo 2009.
- [29] Elena Blanco-Martin, Silvia Merino Saez-Miera, Juan Jose Gomez-Alfageme, and Luis Ortiz-Berenguer. (2011, Mayo) Repeatability of localization cues in HRTF data bases.
- [30] Christian Müller-Tomfelde. Low-Latency convolution for real-time applications.
- [31] Young Cheol Park , Dae Hee Youn Tacksung Choi, *Efficient Out of Head Localization System*. Paris, France: AES, 2006.
- [32] Kiyofumi Inanaga, Yuji Yamada, and Hiroshi Koizumi, "Headphone System with Out-of-Head Localization Applying Dynamic HRTF (Head-Related Transfer Function)," Sony Corp, Tokyo, 1995.
- [33] Steinberg Media Technologies. (2013, febrero) VST SDK 2.4 Documentation. [Online]. HYPERLINK  
"file://localhost/Users/marcelvalderrama/VSTs/vstsdk2.4/doc/html/index.html"  
<file://localhost/Users/marcelvalderrama/VSTs/vstsdk2.4/doc/html/index.html>

## 8. Glosario

**Auralización:** Es el proceso de renderizar datos de audio por medios digitales para simular un espacio acústico.

**Bin:** se refiere al índice de una muestra espectral. Informalmente se puede pensar que una transformada discreta de Fourier de tamaño N, divide el espectro en N recipientes frecuenciales igualmente espaciados.

**Buffer Size:** Tamaño de los bloques de entrada y salida en un DAW.

**Buffer:** Nombre que se le da a un arreglo donde se alojan datos numéricos, generalmente audio digital.

**C** Lenguaje de programación.

**C++:** Un súper conjunto del lenguaje C orientado a objetos.

**Clase** Es una plantilla para definir los atributos y funciones de un objeto.

**Class:** Palabra clave para definir una clase en C++.

**Compilar:** Traducir un lenguaje de alto nivel (C++) a lenguaje de binario o de máquina.

**Constructor:** Método o función presente en toda clase, encargado de inicializar los atributos o variables de la misma.

**Crosstalk:** Fenómeno por el cual la señal de un canal acústico, o electro acústico interfiere con otro.

**DAW:** Por sus siglas en inglés Digital Audio Workstation (Estación de Trabajo Para Audio), es un programa para la edición y mezcla de audio.

**Destructor:** Método encargado de liberar la memoria alojada para un objeto.

**Double:** Tipo de dato que representa números reales con mayor precisión que un float.

**Filtro Shelving:** Filtro que deja inalteradas algunas porciones del espectro, quitándole o dándole ganancia a otras.

**FIR:** Filtro de respuesta al impulso finita, es un filtro no recursivo (salida solo depende de la entrada presente y algunas pasadas), donde sus coeficientes son las muestras de la respuesta al impulso.

**Float:** Tipo de dato que representa números reales en C y C++.

**Función de Transferencia:** Función compleja que contiene las características de un sistema, las cuales permiten saber las variaciones que se le aplicaran a una señal de

entrada, típicamente los cambios en fase y amplitud que experimentaran sus componentes espectrales.

**Host:** Huésped del plugin, hace referencia al DAW donde se utiliza.

**Int:** Tipo de dato que representa números enteros C y C++.

**Objeto:** Instancia de una clase en memoria.

**Offset:** Desplazamiento de una variable.

**Plug In:** Componente adicional que corre sobre un programa y que permite realizar tareas, no incluidas en el programa.

**Void:** tipo de dato vacío del lenguaje de C y C++.

## ANEXO 1: FICHA TÉCNICA DEL PROYECTO DE GRADO

Título del Proyecto de Investigación: Desarrollo de Plugin VST , para la reproducción binaural de fuentes en movimiento, utilizando bases de datos HRTFs.
Grupo de Investigación: No Aplica
Línea de Investigación: Procesamiento Digital de Señales
Semillero de Investigación: No Aplica
<b>Integrantes:</b>
1. Daniel Cardona Rojas
2. Marcel Valderrama Pulgarin
Descriptores / Palabras Claves:HRTFs, Binaural, Plugin VST.
Problema u Objeto de Investigación: ¿Cuál es el desempeño del algoritmo de interpolación bilineal que utiliza una representación en fase mínima de las HRTFs, desde una perspectiva subjetiva y como se puede refinar este algoritmo para dar lugar a pistas binaurales más precisas?
Resumen: Este prototipo implementa los fundamentos teórico/prácticos necesarios para desarrollar un plugin VST, que permita la reproducción binaural de fuentes sonoras en movimiento y en tiempo real. Los desafíos que vienen con este prototipo incluyen el mejoramiento de la resolución espacial de las mediciones de una base de datos convencional de Funciones de Transferencia Asociadas a la Cabeza (HRTFs), por medio del método de interpolación bilineal. Implementando optimas herramientas, para procesar una señal de manera que posea las características y pistas que determinan la precedencia de un sonido, a costos computacionales bajos, dentro de los cuales se encuentran algunas rutinas para calcular la convolución en tiempo real de dos tramas de datos, así como el método de interpolación bilineal. Por último, se indaga sobre desempeño del Plugin llevando a cabo una prueba subjetiva a la población de estudiantes de ingeniería de sonido de la San Buenaventura tomando como muestra de la población 30 estudiantes.
Objetivo General:Desarrollar un plugin VST, que permita la reproducción binaural en tiempo real de fuentes virtuales en movimiento, mediante la utilización de HRTFs.
Productos Esperados: <ul style="list-style-type: none"> <li>• Código de MATLAB(.m) que permita la lectura de datos de la base de datos HRTF proporcionada por el IRCAM, realizando la convolucion entre una señal de entrada y la respuesta al impulso según la posición que el usuario defina.</li> <li>• Código de MATLAB(.m) que realice la interpolación bilineal realizando la comparación entre la señal resultante con el punto interpolado, y un punto medido de la base de datos cercano.</li> <li>• Código de MATLAB(.m) que permita la estimación de las diferencias interaurales de tiempo ITD por los métodos de Woodworth/Schlosberg, Larcher/Jot, y Savioja.</li> <li>• Código presentado en C++ Que permita realizar la transformada de Hilbert.</li> <li>• Archivo .dll y .vst con el Plugin VST compilado que permita su implementación en DAWs compatibles.</li> </ul>

## ANEXO 2: PRUEBA SUBJETIVAS

### Cuestionario

La encuesta se realizó sobre la plataforma de Google Drive, a continuación se anexan las imágenes de la parte visual de la encuesta realizada.

**ENCUESTA Binaural Plugin VST**

\*Obligatorio

**Nombre \***

**Edad \***

**Sexo**

**Semestre \***

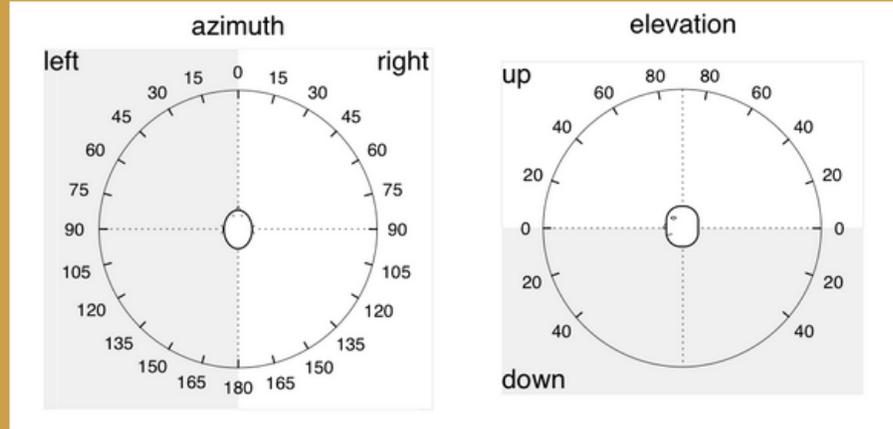
Con la tecnología de Google Drive

Este contenido no ha sido creado ni aprobado por Google.  
[Informar sobre abusos](#) - [Condiciones del servicio](#) - [Otros términos](#)

# ENCUESTA Binaural Plugin VST

## Reconocimiento Discreto de Precedencia

Las siguientes preguntas tienen como objetivo determinar el grado de concordancia que existe entre la posición de la fuente según los parámetros del plugin y la percepción auditiva resultante, para definir el reconocimiento discreto o puntual de precedencia de una fuente sonora.



### PARTE 1

En las siguientes preguntas se mostrará una imagen que representa la posición de la fuente sonora, igualmente se reproducirá una señal binaural auralizada con el plugin; con las características de la posición representativa de la imagen, califique el grado de concordancia de 1 a 5 entre la imagen y la percepción auditiva.

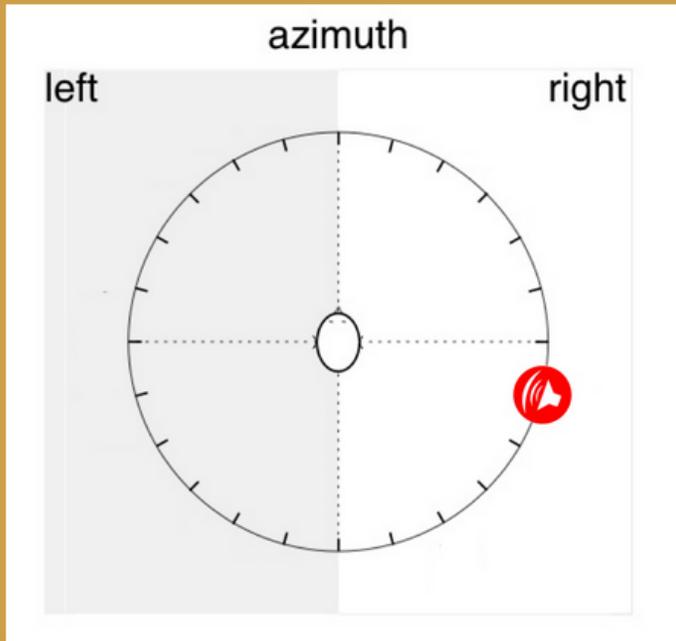


**Cual es el grado de concordancia entre la posición de la fuente sonora de la imagen y la percepción auditiva?**

(IMAGEN DE AZIMUT)

1 2 3 4 5

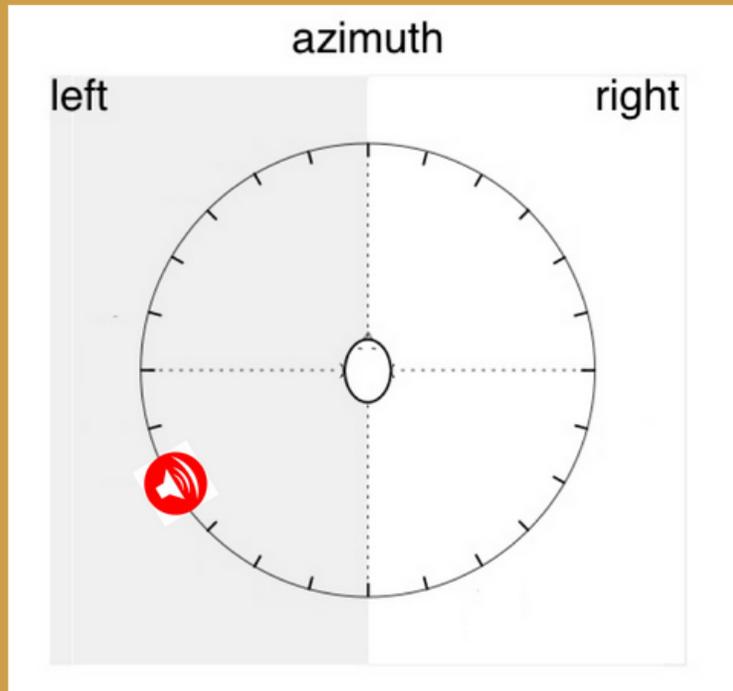
Poca Concordancia      Muy buena Concordancia



Cual es el grado de concordancia entre la posición de la fuente sonora de la imagen y la percepción auditiva?  
(IMAGEN DE AZIMUT)

1 2 3 4 5

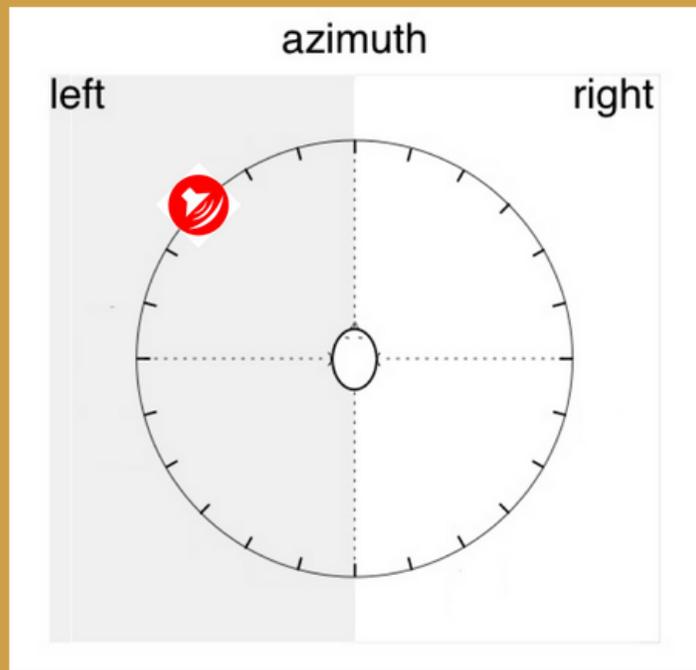
Poca Concordancia ● ● ● ● ● Muy buena Concordancia



Cual es el grado de concordancia entre la posición de la fuente sonora de la imagen y la percepción auditiva?  
(IMAGEN DE AZIMUT)

1 2 3 4 5

Poca Concordancia ● ● ● ● ● Muy buena Concordancia

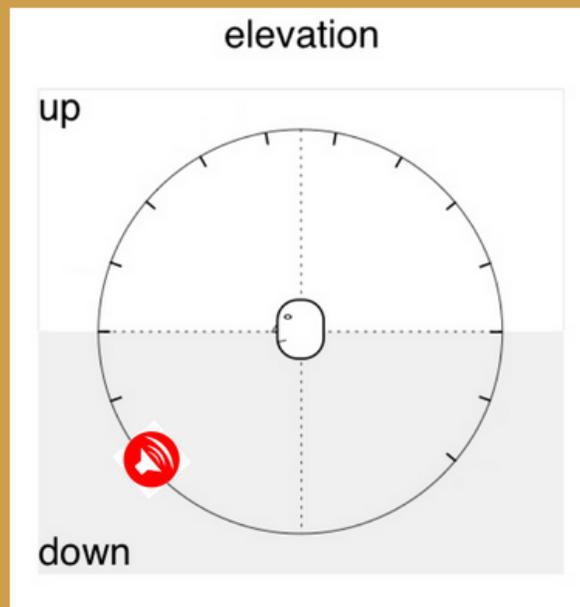


Cual es el grado de concordancia entre la posición de la fuente sonora de la imagen y la percepción auditiva?

(IMAGEN DE AZIMUT)

1 2 3 4 5

Poca Concordancia ● ● ● ● ● Muy buena Concordancia

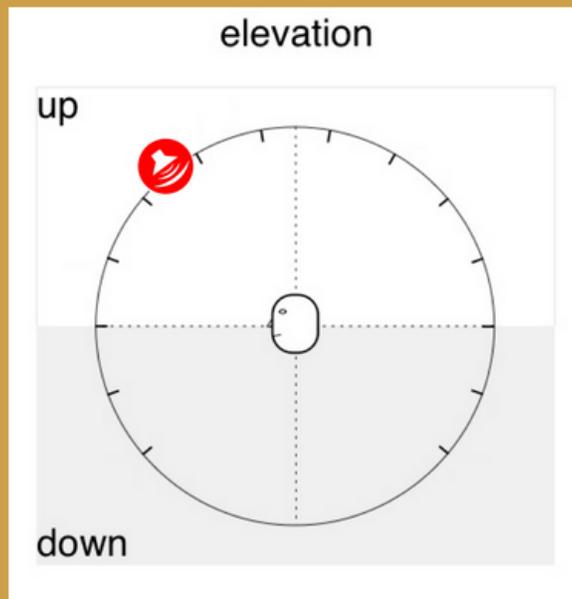


Cual es el grado de concordancia entre la posición de la fuente sonora de la imagen y la percepción auditiva?

(IMAGEN DE ELEVATION)

1 2 3 4 5

Poca Concordancia ● ● ● ● ● Muy buena Concordancia



Cual es el grado de concordancia entre la posición de la fuente sonora de la imagen y la percepción auditiva?

(IMAGEN DE ELEVATION)

1 2 3 4 5

Poca Concordancia ● ● ● ● ● Muy buena Concordancia

## PARTE 2

En las siguientes preguntas se reproducirá una señal binaural en una posición específica, luego se realizara un desplazamiento hacia la izquierda, derecha, arriba o abajo.

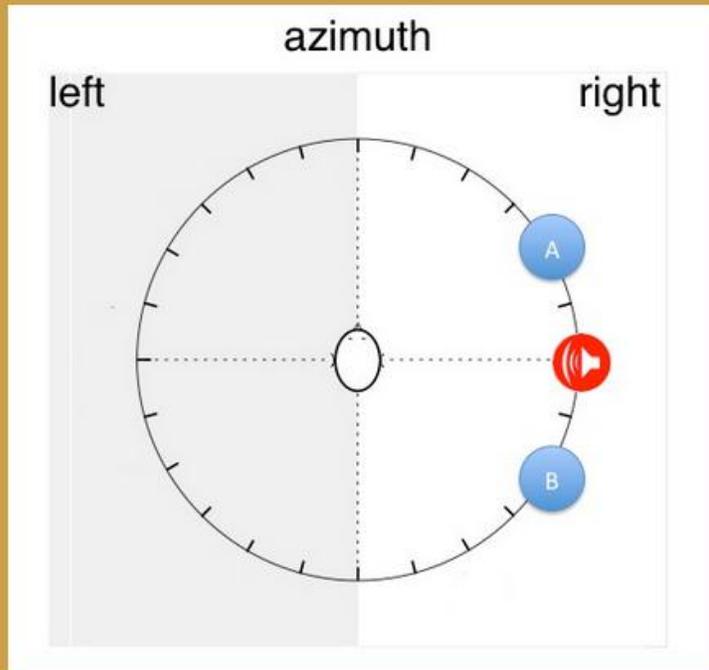
Hacia donde se percibe el movimiento de la fuente sonora?

- A
- B



Hacia donde se percibe el movimiento de la fuente sonora?

- A
- B



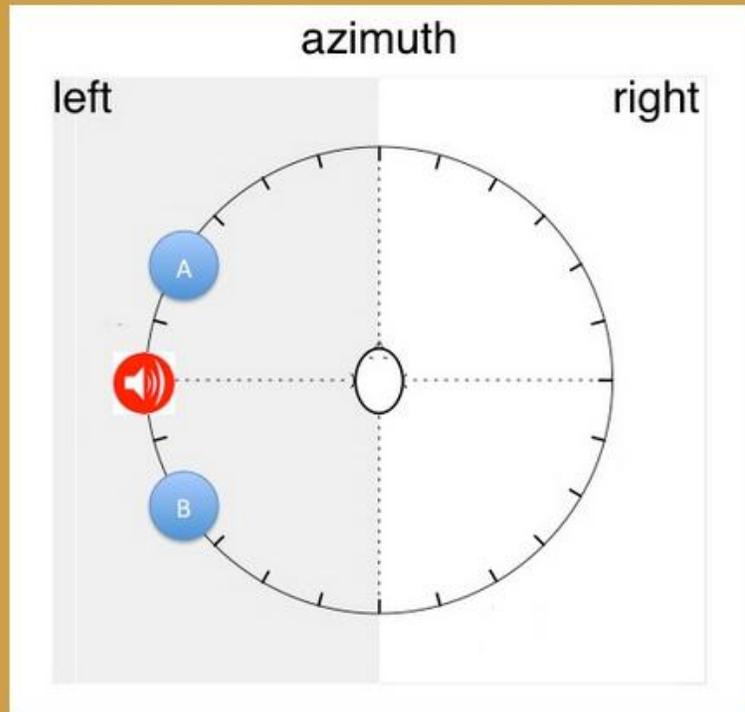
Hacia donde se percibe el movimiento de la fuente sonora?

- A
- B



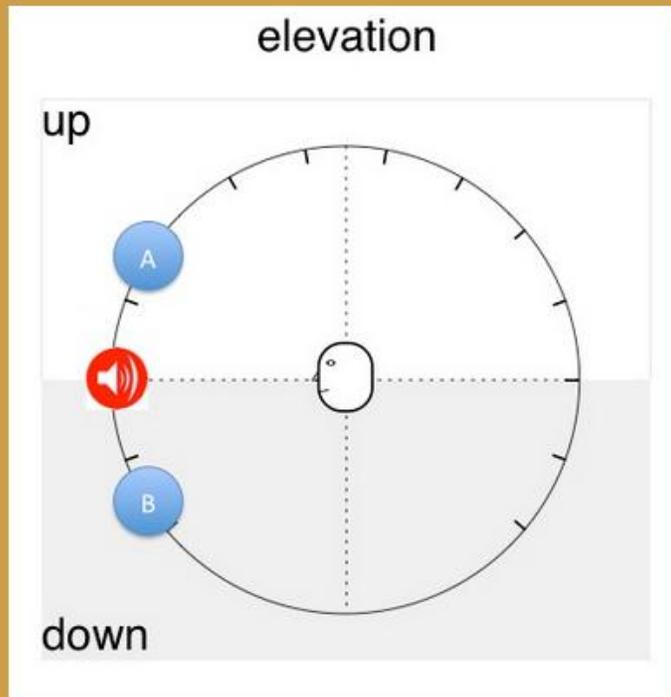
Hacia donde se percibe el movimiento de la fuente sonora?

- A
- B



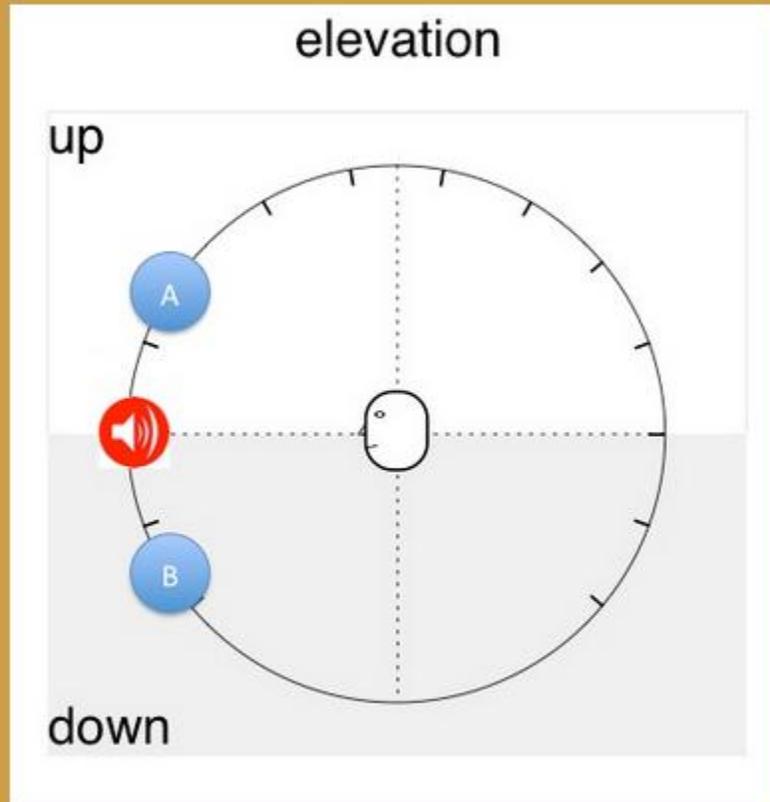
Hacia donde se percibe el movimiento de la fuente sonora?

- A
- B



Hacia donde se percibe el movimiento de la fuente sonora?

- A
- B



« Atrás Continuar »

66% completado

Con la tecnología de  
Google Drive

Este contenido no ha sido creado ni aprobado por Google.  
[Informar sobre abusos](#) - [Condiciones del servicio](#) - [Otros términos](#)

# ENCUESTA Binaural Plugin VST

## Reconocimiento Continuo (Interpolación)

En las siguientes preguntas se evaluara el desplazamiento continuo de la fuente sonora virtual, para analizar el funcionamiento de la interpolación bilineal implementada en el plugin.

### PARTE 1

En esta sección se reproducirá un audio de referencia, el cual tiene un desplazamiento de la fuente sonora virtual con puntos que están implícitos en la base de datos HRTF utilizada, luego se realizara un desplazamiento mas fino para evaluar la interpolación utilizada.

**Percibe el desplazamiento horizontal con mayor continuidad de la fuente sonora?**

- SI
- NO

**Percibe el desplazamiento vertical con mayor continuidad de la fuente sonora?**

- SI
- NO

**Percibe el desplazamiento horizontal con mayor continuidad de la fuente sonora?**

- SI
- NO

**Percibe el desplazamiento vertical con mayor continuidad de la fuente sonora?**

- SI
- NO

Nunca envíes contraseñas a través de Formularios de Google.

 100%: has terminado.

Con la tecnología de  


Este contenido no ha sido creado ni aprobado por Google.  
[Informar sobre abusos](#) · [Condiciones del servicio](#) · [Otros términos](#)

## Ficha Técnica de la Encuesta

		<b>Especificaciones</b>	
<b>Hardware</b>	<b>Computador Mac Book White</b>	Nombre del modelo:	MacBook
		Identificador del modelo:	MacBook6,1
		Nombre del procesador:	Intel Core 2 Duo
		Velocidad del procesador:	2,26 GHz
		Número de procesadores:	1
		Número total de núcleos:	2
		Caché de nivel 2:	3 MB
		Memoria:	4 GB
		Velocidad del bus:	1,07 GHz
		Versión de la ROM de arranque:	MB61.00C8.B00
		Versión SMC (sistema):	1.51f53
		Número de serie (sistema):	W8008DBY8PW
		UUID de hardware:	61F65517-5901-5ECB-97C8-957A3EC8CCF3
	<b>Interface de audio Audo Box Presonus</b>	Salida de Auriculares	Tipo de conector ¼" TRS, hembra, estéreo
			Potencia máxima 30 mW/ch @ 60Ω load
			Respuesta en frecuencia 20 Hz - 20 kHz, +/- 0.5 dB, ganancia max
			THD+N 0.08%, 1 kHz, ganancia max, 20 kHz BW, A-wtd
			Relación S/R 90 dB, 1 kHz, ganancia max, 20 kHz BW, unwtd
		Alimentación	Alimentación USB bus
		Digital	Tipo de puerto de conexión USB 1.1
			Resolución de bits 24-bit
			Frecuencia de muestreo 44.1 kHz, 48 kHz,
			Nivel de referencia de 0dBFS +4 dBu
			Rango dinámico ADC 102 dB, 48 kHz de frecuencia de muestreo, A-wtd
			Rango dinámico DAC 110 dB, 48 kHz de frecuencia de muestreo, A-wtd
			E/S MIDI I Conectores 5-pin DIN
	<b>Audifonos Sony MDR 7506</b>	Audifonos tipo: Cerrados	Cerrados
		Medidas del Driver:	40 mm día., tipo dinámico
		Impedancia: 63 Ohms a 1 Khz	63 Ohms a 1 Khz
		Sensibilidad:	106 dB/mW
		Potencia medida:	0.5 W
Potencia máxima:		1 W	
Respuesta en Frecuencia:		10 a 20 000 Hz	
Cable:		3 metros (longitud total) en forma de bobina que permite más movilidad	
Peso: Aprox.		230 g (con el cable)	
<b>Software</b>		<b>Ableton Live</b>	Version
	Serie		3006-3803-8CFA-A6BE-7701-C263
	Sample Rate		44100
	Buffer Size		248 Samples
	Input latency		0 ms
	Output latency		7,60 ms
	<b>BinauralPlugin VST</b>	Desarrolladores	Daniel Cardona - Marcel Valderrama
		Tipo de plugin	VST
		Interfaz Estandar	Steinberg

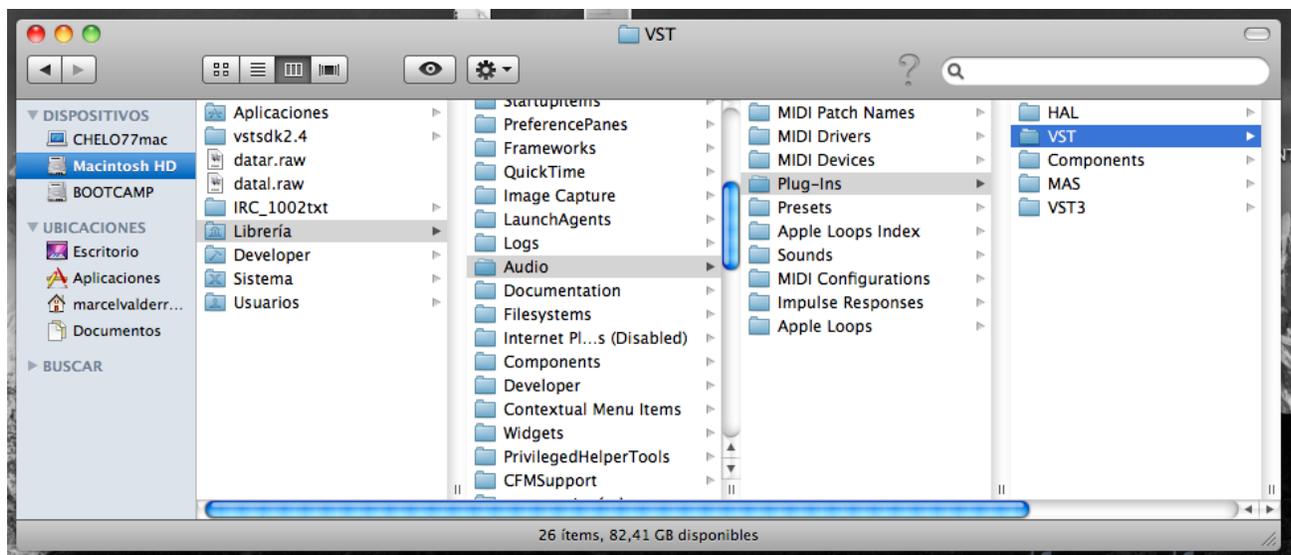
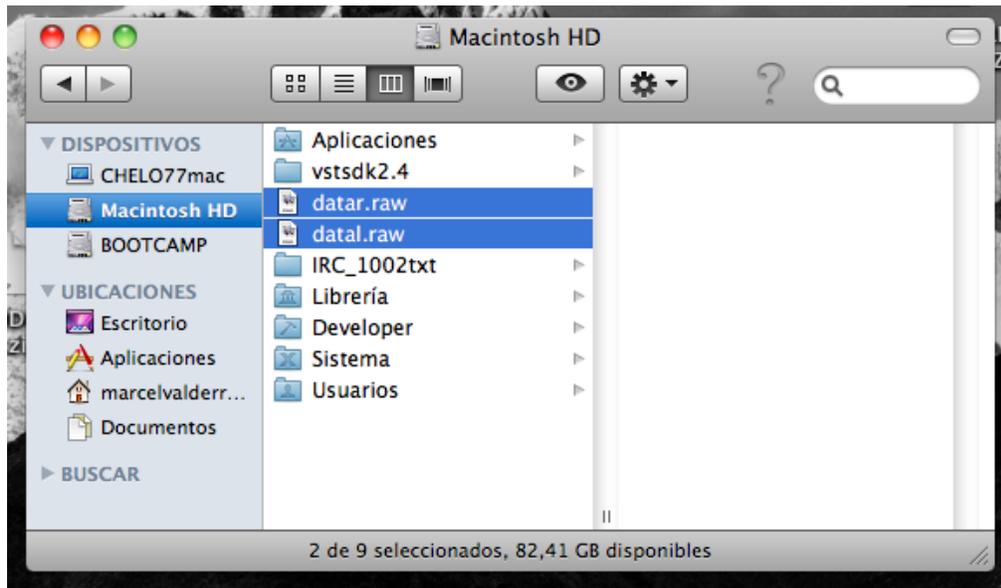
### Respuestas del Formulario

					Sección 1												Sección 2			
					Parte 1						Parte 2						Parte 1			
					P1	P2	P3	P4	P5	P6	P1	P2	P3	P4	P5	P6	P1	P2	P3	P4
Marca temporal	Encuestado	Edad	Sexo	Semestre																
11/19/2013 15:20:01	1	22	Masculino	10	2	5	1	1	3	3	B	B	A	A	A	A	SI	SI	SI	SI
11/19/2013 15:37:33	2	22	Masculino	10	5	5	5	5	5	3	B	B	A	B	A	B	SI	NO	SI	NO
11/19/2013 15:53:41	3	23	Masculino	10	1	2	3	3	4	2	B	B	B	A	A	B	NO	SI	SI	NO
11/19/2013 16:04:19	4	23	Masculino	9	2	5	5	3	3	5	B	A	A	B	A	B	SI	SI	NO	NO
11/19/2013 16:23:33	5	23	Masculino	N/A	2	5	5	2	2	3	B	A	B	B	A	B	SI	NO	SI	SI
11/19/2013 16:38:19	6	23	Masculino	10	4	4	5	4	5	3	B	B	A	A	B	A	SI	SI	SI	SI
11/19/2013 17:26:22	7	22	Masculino	8	5	4	5	5	4	5	B	A	A	B	B	A	SI	SI	SI	SI
11/19/2013 17:42:00	8	21	Femenino	9	4	4	5	4	5	2	B	B	A	B	A	B	SI	NO	SI	NO
11/19/2013 17:56:37	9	25	Masculino	9	2	5	5	5	5	2	B	A	A	B	B	A	SI	NO	SI	NO
11/20/2013 14:31:02	10	21	Masculino	9	3	5	4	2	4	5	B	A	A	B	A	A	SI	SI	SI	SI
11/20/2013 14:41:40	11	22	Masculino	10	5	5	4	5	5	5	B	A	A	B	A	B	SI	SI	SI	SI
11/20/2013 14:48:38	12	22	Masculino	10	5	5	5	3	4	3	B	A	A	A	A	B	SI	SI	SI	SI
11/20/2013 15:01:14	13	21	Masculino	10	4	5	5	4	5	5	B	B	A	B	A	B	SI	SI	SI	SI
11/20/2013 15:12:44	14	22	Masculino	7	3	4	2	3	3	4	B	A	A	B	A	B	SI	NO	SI	NO
11/20/2013 15:32:26	15	19	Femenino	6	1	1	5	3	5	2	B	A	A	A	B	A	SI	SI	SI	SI
11/20/2013 15:45:09	16	21	Masculino	7	3	4	2	3	3	3	B	B	A	B	A	B	SI	NO	SI	NO
11/20/2013 15:56:29	17	21	Masculino	7	4	5	4	3	4	4	A	A	B	A	A	B	SI	SI	NO	SI
11/20/2013	18	24	Masculino	5	3	4	4	5	4	2	B	B	A	B	A	A	SI	SI	SI	NO

16:05:37																				
11/20/2013 17:28:36	19	24	Masculino	10	5	5	3	3	5	5	B	A	A	A	A	A	SI	SI	SI	NO
11/20/2013 17:43:36	20	20	Masculino	8	2	3	4	2	3	4	B	B	A	A	A	B	SI	NO	SI	SI
11/20/2013 17:55:13	21	22	Masculino	7	4	5	4	5	5	4	B	A	A	B	A	A	SI	NO	NO	NO
11/21/2013 14:51:27	22	24	Masculino	8	3	4	2	2	4	2	B	A	A	B	A	B	SI	SI	SI	NO
11/21/2013 15:15:53	23	23	Masculino	7	4	4	5	5	4	5	B	A	B	B	A	B	SI	NO	NO	NO
11/21/2013 15:26:40	24	21	Masculino	7	4	4	5	5	5	4	B	A	A	B	A	B	SI	SI	SI	NO
11/21/2013 16:10:10	25	30	Masculino	9	1	1	3	3	4	4	B	A	B	A	A	A	SI	SI	SI	NO
11/21/2013 16:21:56	26	21	Femenino	8	3	1	1	1	4	4	B	A	A	A	A	A	SI	SI	SI	SI
11/21/2013 16:32:18	27	21	Masculino	6	4	3	3	2	3	4	B	B	A	B	B	A	NO	NO	SI	NO
11/21/2013 16:45:54	28	24	Masculino	10	1	3	2	2	4	3	B	A	A	A	B	A	SI	NO	SI	SI
11/21/2013 17:00:05	29	23	Masculino	10	2	3	3	2	3	2	B	B	B	B	A	B	NO	NO	NO	NO
11/21/2013 17:18:58	30	20	Masculino	5	4	4	5	4	4	4	B	A	A	A	A	B	SI	SI	SI	SI

## ANEXO 3: INSTALACION DEL PLUGIN (WIN/MAC)

Para la instalación satisfactoria del plugin, es necesario, en el caso de ser instalado en un Mac poner el plugin (archivo llamado binaural.vst) en la carpeta donde el sistema guarda sus plugins de esta clase, usualmente ubicada en el disco local específicamente en: /Library/Audio/Plug-Ins/VST. El siguiente paso es tomar los dos archivos donde se almacena la base de datos (data.raw y datar.raw) y copiarlos en la carpeta donde se ubique la aplicación desde la cual se correrá el plugin (e.g Ableton Live), si al abrir la aplicación, no aparece en la lista de efectos nuestro VST, es posible que sea necesario adicionar una ruta adicional de escaneo de plugins manualmente.



En el caso de estar en un sistema operativo de Windows es necesario realizar los pasos anteriores con el plugin en este caso de extensión .dll y ubicarlo en la carpeta de VSTs propia de la aplicación usualmente ubicada en la carpeta de la aplicación por ejemplo: C:/archivos de programa/Ableton Live/Plug Ins/Vst.