UNTREF UNIVERSIDAD NACIONAL DE TRES DE FEBRERO

INGENIERÍA DE SONIDO

Implementación de filtros HRTF para síntesis binaural en tiempo real†

Autor: Ezequiel Santiago Blanco
Tutor: Ing. Elio De María

(†) Tesis para optar al título de Ingeniero de Sonido

Octubre 2019

Agradecimientos

En primer lugar, deseo agradecer a las autoridades de la Universidad Nacional de Tres de Febrero (UNTREF), Rector Lic. Aníbal Jozami, Vicerrector Lic. Martín Kaufmann y Coordinador de la Carrera de Ingeniería de Sonido Mg. Alejandro Bidondo, a todo su personal docente y no docente. Por promover un espacio y recursos acordes para el desarrollo de profesionales.

Agradezco a toda mi familia por el apoyo incondicional. En especial a Andrea mi compañera de vida y aventuras a lo largo de estos años.

A mis amigos, compañeros y profesores que siempre ayudaron durante el proceso de aprendizaje.

A la comunidad de SOFAcoustics por compartir libremente diferentes bases de datos con filtros HRTF. A Kris Winer por facilitar de forma libre las bibliotecas que hicieron posible la realización del sensor de movimiento.

Gracias a todas las personas que participaron de las pruebas subjetivas en esta investigación en pos de la ciencia.

Ezequiel Blanco

Resumen

En esta tesis de grado se diseñó y evaluó un sistema capaz de realizar síntesis binaural en tiempo real. El objetivo principal fue generar una herramienta que aporte al campo de la percepción subjetiva orientada a la espacialización de fuentes sonoras con auriculares. La síntesis binaural se logró implementando algoritmos que convolucionan una señal de audio con un par de filtros relativos de la cabeza (HRTF, *Head-Related Transfer* Functions). Los filtros HRTF son la función transferencia de las respuestas al impulso que se obtienen a partir del sistema acústico conformado por una fuente sonora puntual y cada uno de los oídos en el campo libre. Estos algoritmos utilizan la convolución particionada para lograr el procesamiento de las señales en tiempo real. Además, se diseñó y construyó un sensor de movimiento para integrar las rotaciones de la cabeza en el sistema.

Una vez desarrollado el sistema, el siguiente paso fue evaluarlo por medio de pruebas subjetivas de localización de fuente sonora, percepción de latencia del sensor de movimiento y pérdida de calidad debido a los filtros. Los resultados obtenidos demostraron que el sistema cumple con los objetivos planteados, tanto en lo referido a percepción subjetiva como en su funcionamiento.

Palabras claves: “Filtros HRTF”, “Síntesis binaural dinámica”, “Convolución particionada en tiempo real”, “Sensor de movimiento de la cabeza”.

Abstract

In this thesis, a system capable of performing binaural synthesis in real-time was designed and evaluated. The main objective was to generate a tool that contributes to the field of subjective perception oriented to the spatialization of sound sources with headphones. The binaural synthesis was achieved by implementing algorithms that convolves an audio signal with a pair of head-related transfer function filters (HRTF). HRTF filters are the transfer function of the impulse responses that are obtained from the acoustic system consisting of a specific sound source and each of the ears in the free field. These algorithms use the partitioned convolution to achieve real-time processing of the signals. Also, a motion sensor was designed and built to integrate the rotations of the head into the system.

Once the system was developed, the next step was to evaluate it through subjective tests of sound source localization, perception of the motion sensor latency and loss of quality due to the filters. The results obtained showed that the system reaches the propose objectives, both in terms of subjective perception and its functionality.

Keywords: "HRTF filters", "Dynamic binaural synthesis", "Real-time partitioned convolution", "Head tracking".

Índice

Índice de figuras

Índice de tablas

CAPÍTULO I – FUNDAMENTACIÓN E INTRODUCCIÓN

En los últimos años, el consumo del contenido audiovisual ha ido creciendo y evolucionando rápidamente. Cada vez es más fácil acceder a computadoras, tabletas (*tablets*), celulares, televisores inteligentes (*smartTV*) o equipos de reproducción tanto de audio como de video. Dejando de lado la componente visual, hoy en día hay muchas áreas donde se busca mejorar la experiencia inmersiva de las personas al darle espacialidad a las fuentes sonoras. Algunas de estas son la realidad virtual, realidad aumentada, simulación, producción musical, auralización, filmaciones en 360°, videojuegos, películas, telepresencia, desarrollo de audífonos y pruebas subjetivas de localización. Si la reproducción del sonido es mediante auriculares la espacialidad se puede lograr utilizando tecnología binaural.

La tecnología binaural abarca varios métodos de grabación, síntesis y reproducción, los cuales permiten recrear la forma en que las personas escuchan las fuentes sonoras en el espacio tridimensional. Una manera de lograrlo, es realizar una grabación binaural con micrófonos de cápsulas muy pequeñas en cada entrada del canal auditivo de una persona real o un maniquí. La reproducción posterior con auriculares va a generar una experiencia auditiva con espacialidad similar a como si el oyente hubiese estado situado en la misma posición donde se realizó la grabación. Por otro lado, las señales binaurales se pueden generar mediante síntesis con procesamiento digital de señales. En este caso se procesan señales de audio monoaurales con los filtros HRTF.

Los filtros HRTF se corresponden con la función transferencia entre una fuente sonora en el espacio y cada oído. En otras palabras, estos filtros describen como se modifica el sonido directo en el campo libre al llegar a los tímpanos de un sujeto receptor. Dependiendo de la dirección de la fuente, la misma onda sonora arriba a cada oído con una información temporal y espectral diferente. Las curvas de los filtros HRTF van a depender de la anatomía del sujeto receptor. Las dimensiones de los pabellones auditivos, las características faciales, ancho de hombros, etc., aportan a generar diferencias en la respuesta en frecuencia y fase de la señal. Para obtener los filtros HRTF, se graban respuestas al impulso binaurales (HRIR, *Head-Related Impulse Responses*) con diferentes posiciones de una fuente sonora en el campo libre.

En 1994, se crea la primera base de datos pública de filtros HRTF en el Massachusetts Institute of Technology (MIT). Los investigadores Gardner y Martin utilizan una cabeza maniquí para registrar las respuestas al impulso [1]. A partir de ese momento, las investigaciones sobre síntesis binaural y las asociadas a percepción psicoacústica han ido en aumento. Varias universidades han creado sus propias bases de datos, tanto de sujetos reales como de cabezas maniquí. El mayor inconveniente era que cada base de datos adoptaba su propio formato para almacenar la información. Por lo tanto, era engorroso realizar implementaciones de estos filtros de manera flexible. No obstante, en el 2015, se estandariza un formato llamado *Spatially-Oriented Format for Acoustics* (SOFA) [2] con el objetivo de poder almacenar datos como filtros HRTF, respuestas al impulso direccionales de salas (DRIR), respuestas al impulso binaurales de salas (BRIR) o patrones polares de altavoces.

En esta investigación se contribuye al conocimiento sobre el tema de la síntesis binaural por medio de los filtros HRTF, y se desarrolla una herramienta que facilita la espacialización de fuentes sonoras mediante síntesis binaural para favorecer investigaciones científicas con relación a la percepción subjetiva. Inicialmente, se aborda la teoría psicoacústica relacionada a cómo las personas percibimos el sonido de manera espacial, y luego la teoría sobre el procesamiento digital de señales para el posterior desarrollo de una interfaz con la finalidad de realizar la síntesis binaural.

1.1 OBJETIVOS

1.1.1 Objetivo general

Esta investigación se enfoca en el diseño y desarrollo de un sistema destinado a la síntesis binaural en tiempo real utilizando los filtros HRTF. La misma permite cargar la base de datos de los filtros, darle una ubicación a la fuente virtual e interactuar en forma dinámica con un sensor de movimiento de la cabeza.

1.1.2 Objetivos específicos

- Recopilar y estudiar material bibliográfico sobre los filtros HRTF y sus implementaciones.
- Investigar la manera de cargar bases de datos de filtros HRTF con formato SOFA y poder manipular la información dentro de Matlab.

- Diseñar la estructura del procesamiento con filtros HRTF en tiempo real para una fuente sonora.
- Plantear la utilización de un sensor de movimiento de la cabeza para poder interactuar en tiempo real con las fuentes sonoras.
- Desarrollar una interfaz gráfica que facilite al usuario cargar archivos de audio y poder especificar una ubicación espacial.
- Evaluar el sistema mediante pruebas objetivas y subjetivas.

1.2 ESTRUCTURA DE LA INVESTIGACIÓN

En el Capítulo II se describen algunas investigaciones sobre el tema de desarrollo de software aplicando síntesis binaural. Se pretende dar un panorama del estado del arte sobre el tema para entender como son los bloques de procesamiento que integran la síntesis binaural.

En el Capítulo III se asientan las bases teóricas para poder entender el tema en cuestión. Comienza con la teoría psicoacústica sobre la percepción de fuentes sonoras, con el fin de comprender cuales son los factores que influyen en la misma. Luego, se trata el tema de los filtros HRTF y métodos de implementación en tiempo real.

En el Capítulo IV comienza la descripción de los pasos lógicos para lograr el objetivo del desarrollo del sistema a fin de realizar la síntesis binaural utilizando los filtros HRTF y un sensor de movimiento para la cabeza.

En el Capítulo V se realiza la validación del sensor de movimiento, y del algoritmo de síntesis binaural mediante mediciones de latencias y pruebas de percepción subjetivas.

En el Capítulo VI se expresan las conclusiones luego de los resultados obtenidos.

En el Capítulo VII se listan las posibles investigaciones a futuro sobre el tema en cuestión.

CAPÍTULO II – ESTADO DEL ARTE

En los últimos años, se han desarrollado sistemas de síntesis binaural que utilizan los filtros HRTF con diferentes propósitos y aplicaciones. Se realiza una breve reseña de algunos sistemas de síntesis binaural para entender su funcionamiento. La implementación de esta investigación se apoya en la estructura central de estos sistemas.

la NASA (*National Aeronautics and Space Administration*) desarrolla el software *Sound Laboratory* (SLAB) para poder llevar a cabo investigaciones psicoacústicas [3-5]. Es un software escrito en el lenguaje C++ que realiza la síntesis binaural en tiempo real. Además, permite especificar el patrón polar de la fuente, las reflexiones, absorción del ambiente y del aire. Las reflexiones de primer orden son modeladas con el método de fuente imagen. Para representar la absorción de los materiales aplica filtros de respuesta al impulso infinita (IIR), ya que resultan más eficientes de implementar que los filtros de respuesta al impulso finita (FIR). También, aplica una ecualización de auriculares por medio de filtros IIR. Utiliza el método de convolución *overlap-add* para realizar el procesamiento con los filtros HRTF, y un fundido cruzado para la variación de filtros en el tiempo. La longitud de los filtros HRTF que implementa es de 128 muestras. En la Figura 1 se muestra un diagrama del flujo y los procesamiento de la señal. La latencia interna del sistema ronda los 24 ms, excluyendo la latencia externa del sensor de movimiento de la cabeza.

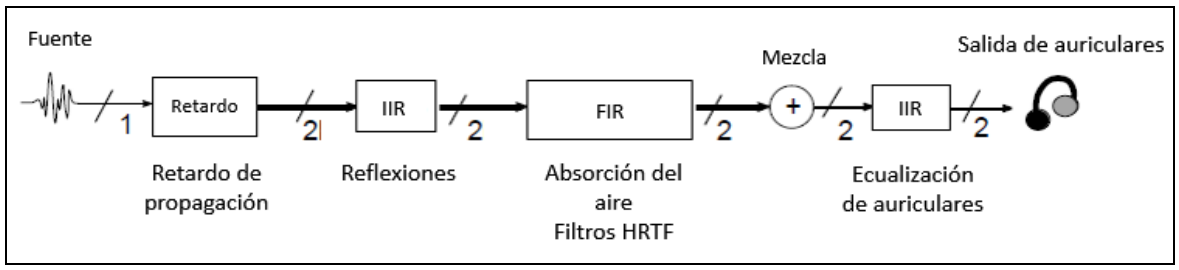

Figura 1. Diagrama del flujo de la señal del sistema SLAB [3].

El sistema *Digital Interactive Virtual Acoustics* (DIVA) desarrollado por la *Helsinki University of Technology* (actualmente fusionada en *Aalto University*) se creó para investigaciones sobre multimedia y realidad virtual [6-7]. Está implementado en el sistema operativo de UNIX y escrito en lenguaje C++. DIVA puede simular un entorno auditivo virtual (*Virtual auditory environments*) con fuentes sonoras estáticas o en movimiento en tiempo real. Incluye los mismos parámetros para modificar el entorno como el SLAB. Las reflexiones de primer y segundo orden son modeladas con el método de fuente imagen,

las reflexiones tardías son generadas con reverberación artificial. La latencia del sistema se encuentra entre 110 y 160 ms.

La Universidad Ruhr de Bochum comenzó a realizar estudios sobre entornos auditivos virtuales a mediados de 1980s. Con el apoyo del *European ESPRIT Research Project*, la primera generación del sistema SCATIS (*Spatially Co-ordinated Auditory/Tactile Interactive Scenario*) fue desarrollado entre 1993 y 1996 [8]. El sistema se diseñó para investigaciones psicoacústicas y realidad virtual en lenguaje C/C++. El procesamiento es similar a los sistemas anteriores. La latencia alcanza los 66 ms.

La misma universidad, en 1998, comienza una segunda generación del sistema IKA-SIM [9], escrito en lenguaje C++ para el sistema operativo de Microsoft Windows. El sistema se implementa en módulos de procesamiento. La idea de un sistema modular es permitir intercambiar bloques de procesamientos según la aplicación que se desee. Además, integra el método de reproducción por altavoces utilizando panorama de amplitud vectorial (VBAP, *Vector Base Amplitude Panning*). En la Figura 2 se muestra el diagrama en bloques del sistema. El núcleo del sistema es el módulo de campo sonoro. Dentro del mismo, se encuentra el módulo de campo sonoro estático el cual realiza la simulación para una habitación con fuentes sonoras y receptores estáticos. Por lo tanto, el sistema es lineal e invariante en el tiempo, las respuestas al impulso son constantes y los caminos de las reflexiones también. Utiliza el método de fuente imagen o de rayos de cualquier orden para las reflexiones. Por otro lado, el módulo dinámico permite desplazar las fuentes sonoras y los receptores. Este módulo se diseña para poder sintetizar espacios sonoros interactivos. Una frecuencia de actualización de 100 Hz permite calcular fuentes imágenes de segundo y tercer orden, dependiendo la complejidad de la simulación. La latencia promedio total es de 80 ms. En el 2008, plantearon desarrollar un sistema montado en un servidor y poder realizar la simulación con acceso remoto a través de internet [10]. El enfoque estaba puesto en aplicaciones móviles. En este caso, el sistema se desarrolló en lenguaje JAVA. La latencia del sistema completo resultó en 350 ms.

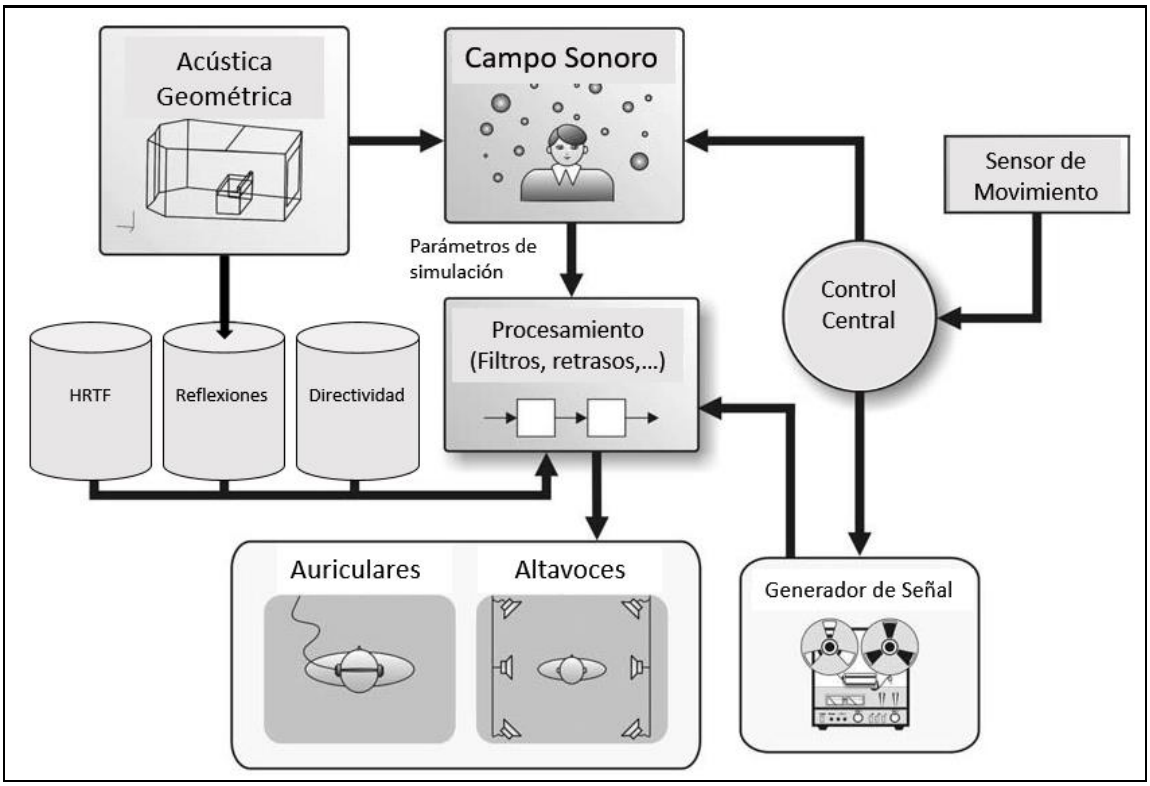

Figura 2. Diagrama en bloque del sistema de síntesis binaural IKA-SIM [9].

El *SoundScape Renderer* (SSR) de la Universidad Técnica de Berlín (*Technische Universität Berlin*), es un software creado para la simulación de fuentes virtuales en tiempo real [11]. Se encuentra en lenguaje C++. Implementa síntesis de frente de onda (WFS, *Wave Field Synthesis*), VBAP y síntesis binaural. En la Figura 3 se muestra el esquema del módulo de síntesis binaural. Es un procesamiento más simplificado que los otros sistemas. Aún así, tiene la estructura básica para realizar la síntesis binaural por medio de filtros HRTF.

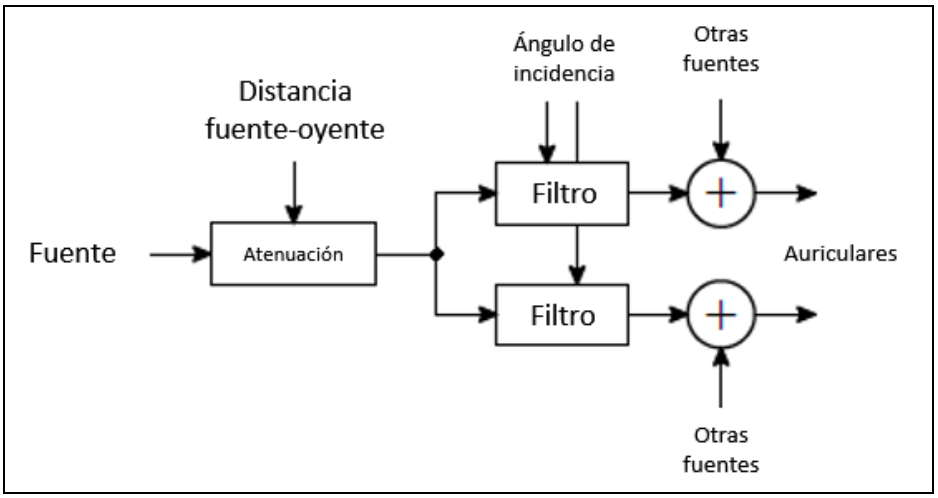

Figura 3. Esquema del módulo de síntesis binaural del SSR [11].

El *Acoustic Lab* de la *South China University of Technology* desarrolla un sistema para investigaciones psicoacústicas, también en lenguaje C++ [12]. El sistema simula información dinámica de la cabeza con seis grados de libertad, y puede simular campo cercano y lejano. Utiliza dos algoritmos para la síntesis binaural. El primero utiliza como base los filtros HRTF, y el segundo utiliza una descomposición de los filtros en componentes principales (PCA - *Principal Component Analysis*) y utiliza un banco de 16 filtros para la síntesis [13]. El segundo es un algoritmo de mayor complejidad, pero permite lograr un mejor rendimiento computacional al disminuir la longitud de los coeficientes de los filtros. En la Figura 4 se muestra el diagrama del sistema. La latencia del sistema ronda los 25,4 ms.

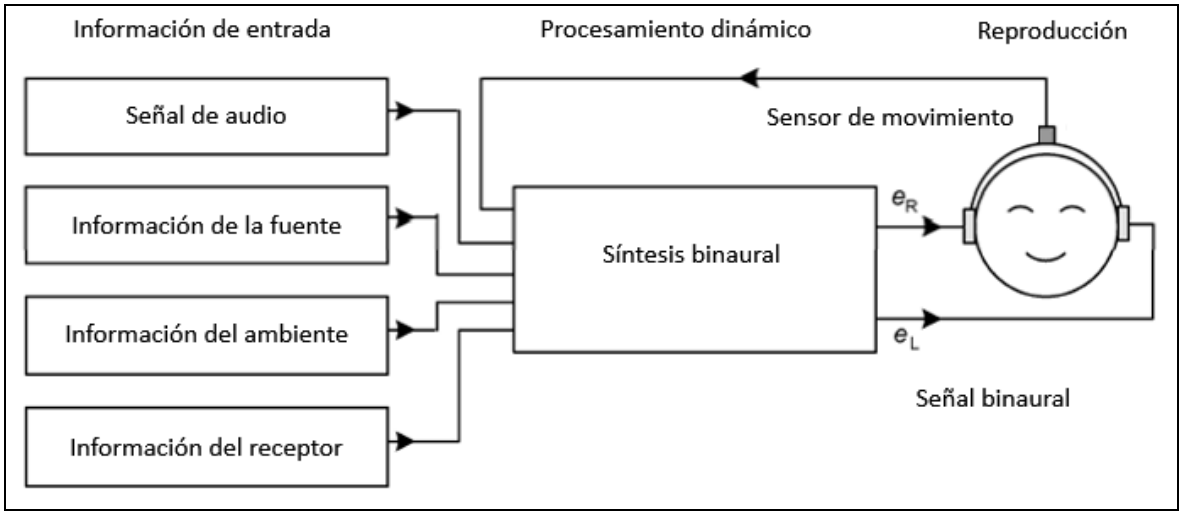

Figura 4. Diagrama del sistema del Acoustic Lab de la South China University of Technology [13].

A pesar de que existen diferentes sistemas de síntesis binaural, estos comparten los mismos bloques de procesamiento. El núcleo del procesamiento es la convolución en tiempo real de la señal con las respuestas al impulso de los diferentes filtros. Algunos integran procesamiento de acústica geométrica al integrar las reflexiones y absorciones de materiales. También, utilizan un sensor de movimiento para interactuar con el sistema y realizar una síntesis dinámica.

Las latencias de los diferentes sistemas son relativas. Las mismas dependen del lenguaje de programación, las etapas de procesamiento y el hardware utilizado para el procesamiento.

CAPÍTULO III – MARCO TEÓRICO

3.1 SISTEMA DE COORDENADAS ESPACIALES

En las investigaciones sobre percepción espacial, la posición de la fuente sonora se especifica en términos de dirección y distancia con respecto a la cabeza del sujeto receptor. Se suele utilizar el centro de la cabeza a la altura de los canales auditivos como origen de coordenadas del sistema. El plano paralelo al suelo que corta a ambos oídos se denomina plano horizontal, el plano perpendicular al suelo que corta a ambos oídos se denomina plano frontal, y el plano que divide la cabeza de manera simétrica se denomina plano medio.

La posición espacial de una fuente se puede representar como un vector en coordenadas esféricas utilizando ángulos azimutales (θ), de elevación (φ) y una distancia al origen del sistema (r). El ángulo azimutal se desplaza sobre el plano horizontal, comenzando con el valor de 0° enfrente a la cabeza, y aumenta hasta 359° en sentido anti horario. El ángulo de elevación comienza en 0° sobre el plano horizontal, y alcanza los valores de 90° y -90° para la posición encima y debajo de la cabeza respectivamente. En la Figura 5 se representa el sistema de coordenadas descrito previamente.

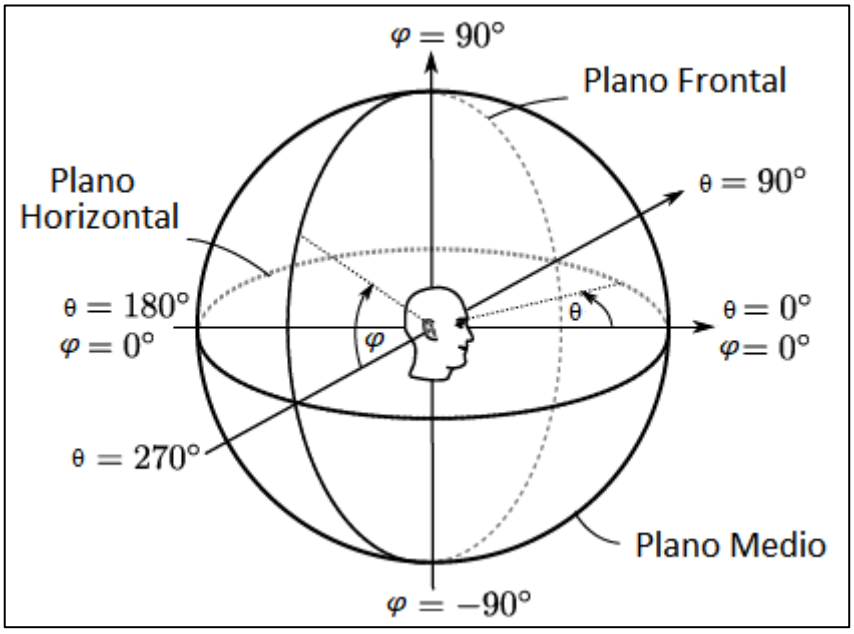

Figura 5. Sistema de coordenadas aplicado para la percepción espacial de fuentes sonoras [14].

3.2 PERCEPCIÓN ESPACIAL

Los seres humanos están expuestos a sonidos por transmisión acústica desde el nacimiento. Gracias al sistema auditivo (oído interno, medio y externo), la anatomía y la habilidad de mover la cabeza, se tiene la capacidad de poder percibir distancia, dirección y sonoridad de las fuentes sonoras en las tres dimensiones espaciales. Estudios psicoacústicos demostraron que esta habilidad depende de las diferencias de tiempo interaural (ITD, *Interaural Time Differences*), diferencias de intensidad interaural (ILD, *Interaural Level Differences*), factores dinámicos y espectrales [15].

La ITD se refiere a la diferencia de tiempo de arribo de una onda sonora al oído izquierdo y derecho. En el caso particular en que la onda se produce en el plano medio, el valor de la ITD es aproximadamente cero. Sin embargo, en la mayoría de los casos la onda está fuera del plano medio, por lo tanto, las distancias al oído izquierdo y derecho son diferentes, dando como resultado un valor de la ITD diferente de cero. De acuerdo a la ecuación de Woodworth [16], en 0 y 180° se encuentra el mínimo que es aproximadamente 0 s, mientras que para 90°, se encuentra el valor máximo que es aproximadamente 662 µs. Estudios subjetivos revelaron que la ITD es un factor importante en la localización para frecuencias aproximadamente por debajo de 1,5 kHz [17].

La ILD es otro factor importante en la localización de fuentes sonoras. Cuando una fuente sonora se encuentra fuera del plano medio, la presión sonora para el oído más lejano estará atenuada (especialmente en altas frecuencias) debido al efecto de sobra acústica de la cabeza. Por otro lado, el oído más cercano tendrá un mayor nivel de presión sonora por efecto de estar a una menor distancia a la fuente.

Los experimentos psicoacústicos han demostrado que las ILD no son un factor eficiente de localización hasta que se presentan variaciones en la dirección de la fuente, por arriba de 1,5 kHz. Esto se debe a que es una función multivariable que depende de la dirección de la fuente y la frecuencia. Adicionalmente, el máximo valor no se presenta a 90°. Esto se debe a un aumento en el nivel de presión sonora en el oído opuesto por el arribo en fase de las ondas difractadas alrededor de la cabeza. Para ondas sonoras complejas, no tonos puros, las ILD varían de manera suave con la variación del ángulo [17].

3.2.1 Cono de confusión y movimiento de la cabeza

Las ITD y ILD son considerados los dos factores dominantes de la localización de fuentes sonoras para bajas y altas frecuencias, respectivamente. Los mismo fueron propuestos en la teoría clásica dúplex (*Duplex Theory*) por Lord Rayleigh en 1907 [18]. Sin embargo, son insuficientes para determinar una única posición en el espacio. De hecho, hay infinitas posiciones que poseen idénticas distancia a cada oído. Estas posiciones se pueden representar como un cono alrededor del eje interaural. En la literatura se lo conoce como el cono de confusión (*cone of confusion*). Un caso particular de confusión es el que se produce en el plano horizontal, el cual se lo conoce como *front-back confusion*.

Wallach en su investigación planteó como hipótesis que los cambios de ITD y ILD producidos por el movimiento de la cabeza son otro tipo de factor de localización (factor dinámico) [19]. Por ejemplo, cuando la cabeza esta fija, las ITD y ILD para una fuente sonora ubicada a 0° (anterior) y 180° (posterior) son cero, debido a la simetría de la cabeza. Si se utiliza el movimiento de la cabeza, las variaciones en las ITD y ILD ayudan a identificar la posición de la fuente sonora. Por ejemplo, si se rota la cabeza hacia la derecha, la fuente sonora que se encontraba a 0° se desplaza hacia la izquierda del plano medio de la cabeza. Se produce una variación en las ITD y ILD que dan información de que la fuente sonora se encuentra más cerca del oído izquierdo. Lo mismo sucede para la fuente que se encuentra a 180°, solo que, al rotar la cabeza hacia la derecha, el oído derecho es el que se encuentra más cerca de la fuente sonora. En la Figura 6 se muestra un esquema lo de previamente explicado.

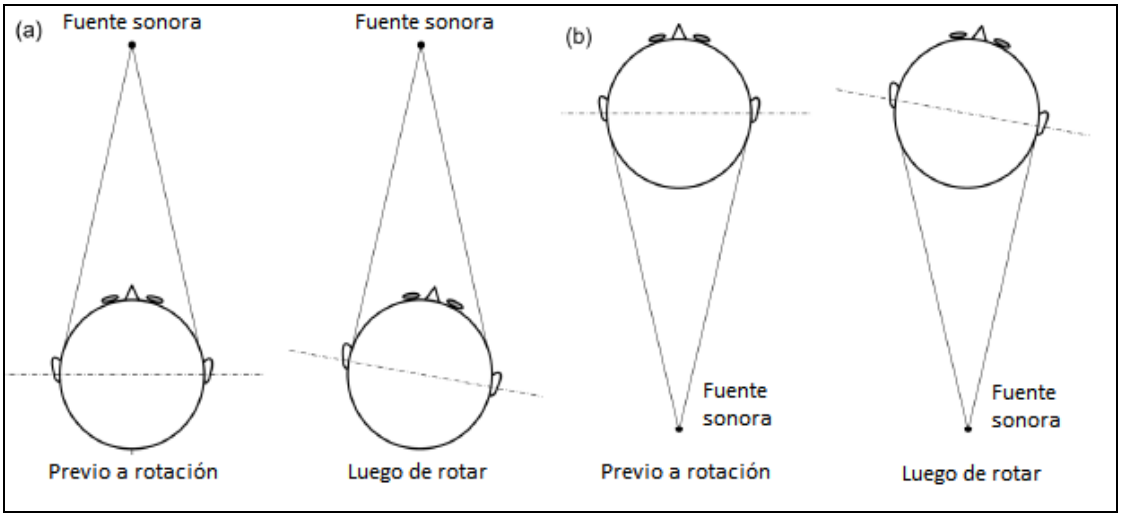

Figura 6. Variación de ITD y ILD por efecto de movimiento de la cabeza: (a) fuente sonora anterior; (b) fuente sonora posterior [15].

El factor de rotación ayuda a resolver el problema de *front-back confusion* para fuentes estáticas. Esta conclusión se verifica en la investigación de Wightman y Kistler [20]. Además, otras investigaciones han demostrado que la rotación de la cabeza con dos grados de libertad ayuda a la localización en el plano medio para bajas frecuencias [21-22].

3.2.2 Factor espectral

Muchos estudios sugieren que las características espectrales causadas por las reflexiones y difracciones en el pabellón auditivo, así como también por la cabeza y el torso, contribuyen con información para la localización de fuentes con ángulo de elevación y diferenciar fuentes frontales y posteriores. En contraste con los factores binaurales (ITD y ILD), el factor espectral se considera monoaural [23]. En la Figura 7 se muestra un ejemplo de la incidencia del sonido directo y una reflexión dentro del pabellón auditivo, y el espectro resultante con dos posiciones de fuentes sonoras posibles.

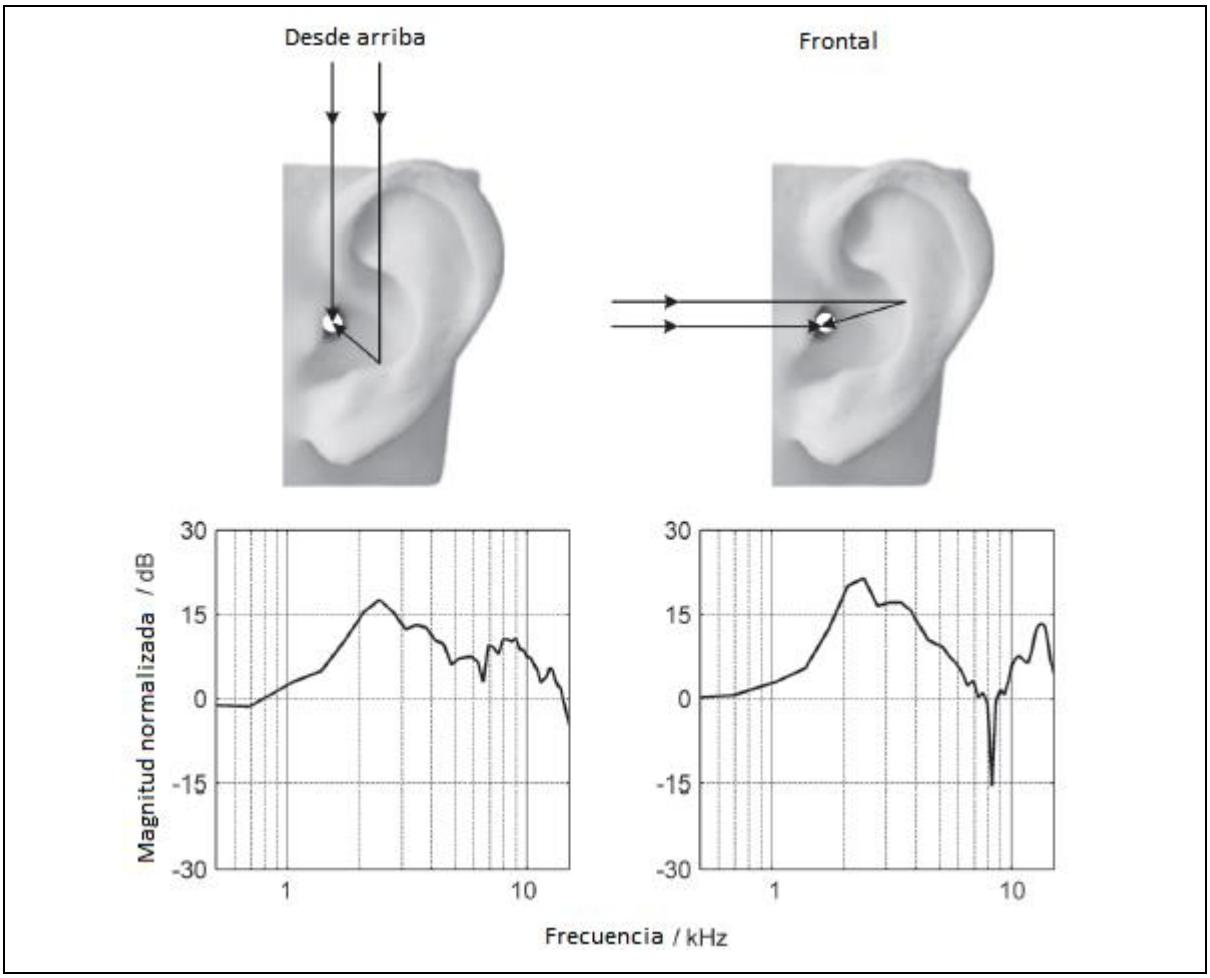

Figura 7. Espectro resultante debido a la incidencia del sonido directo y una reflexión dentro del pabellón auditivo para dos posiciones de fuente [15].

Debido a que el pabellón auditivo mide aproximadamente 65 mm, las reflexiones y difracciones comienzan a producirse a partir de los 1,5 kHz, en donde las longitudes de onda comienzan a ser comparables a las del pabellón auditivo. Estudios muestran que a partir de 3 ó 3,5 kHz ya comienza a ser un factor importante para la localización de fuentes sonoras en el plano medio con ángulos de elevación [17, 24-26].

Los picos y valles en el espectro resultantes actúan como un filtro de la onda sonora originaria. Los mismos son dependientes de la forma y dimensiones del pabellón auditivo. Por esta razón, la información espectral es diferente para cada individuo.

3.2.3 Percepción de distancia

El sistema auditivo humano estima en menor medida la distancia de una fuente sonora que la dirección de la misma. La distancia real a una fuente sonora y la percepción de distancia no suelen ser las mismas en muchos casos. Estudios subjetivos han demostrado que el sistema auditivo tiende a subestimar la percepción de distancia de una fuente sonora con respecto a su distancia física para valores más altos de 1,6 m, y sobreestimar para distancias más cortas. Zahorik [27], tras analizar resultados de investigaciones, encuentra una relación entre la distancia percibida (*r'*) y la distancia física real (*r*) que se puede aproximar con:

$$r' = k\, r^{a}, \tag{1}$$

donde *k* es una constante con valor superior a uno (aprox. 1,32); y *a* es un valor que depende de cada sujeto, en promedio tiende a 0,4. En la Figura 8 se muestra la relación entre *r'* y *r*, obtenida con interpolación lineal para un sujeto típico.

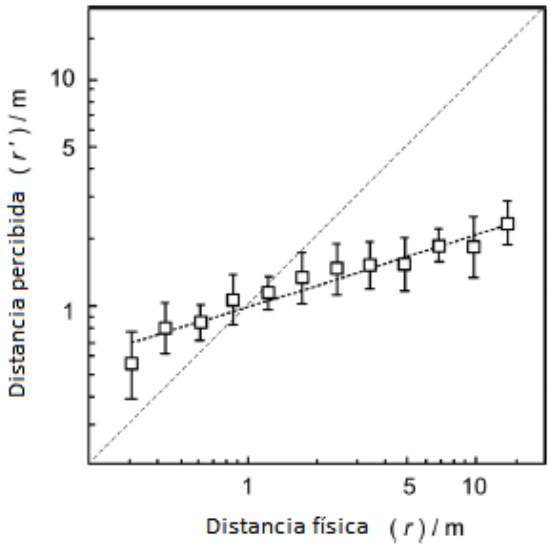

Figura 8. Relación entre *r'* y *r* para un sujeto típico, con *a* = 0,32, *k* = 1 [28].

Sin embargo, la percepción de distancia es un proceso complejo que depende de muchos factores. El factor subjetivo de sonoridad (*loudness*) ha sido considerado un factor efectivo para percibir distancia de una fuente sonora. En el campo libre el nivel de presión sonora de una fuente puntual es proporcional al cuadrado inverso de la distancia[1]. Al duplicar la distancia el nivel de presión sonora decrece 6 dB. Por lo tanto, una mayor distancia corresponde a un menor nivel de presión sonora, menor nivel de sonoridad, y consecuentemente una mayor distancia percibida. Si bien, este fenómeno acústico aplica en el campo libre, para un ambiente con reflexiones el comportamiento es diferente. Más aún, la percepción de sonoridad va a depender del patrón polar de la fuente, como también de ya poseer conocimientos del comportamiento de la fuente sonora [29].

La atenuación en altas frecuencias debido a la absorción del aire es otro factor que influye en la percepción de distancia. Para distancias grandes, la absorción del aire actúa como un filtro pasa bajo.

3.2.4 Percepción de ambiente

Cuando se trata de percepción espacial en salas, lo que escucha un sujeto receptor es el sonido directo más una serie de reflexiones características de la sala. En la Figura 9 se muestra un ejemplo de la transmisión acústica dentro de una sala. Luego de que arriba el sonido directo, llegan las reflexiones tempranas que corresponden a las reflexiones de primer orden sobre las superficies de la sala. Los tiempos de arribo van a depender

[1] En la literatura se la conoce como la ley del cuadrado inverso.

directamente de las distancias de los recorridos de la onda acústica. Finalmente, el número de reflexiones se incrementa provocando una gran densidad de energía, pero con un menor nivel debido a la absorción. La información de la transmisión de una sala (en un punto específico) se puede obtener a través de la respuesta al impulso (RIR, *Room Impulse Response*).

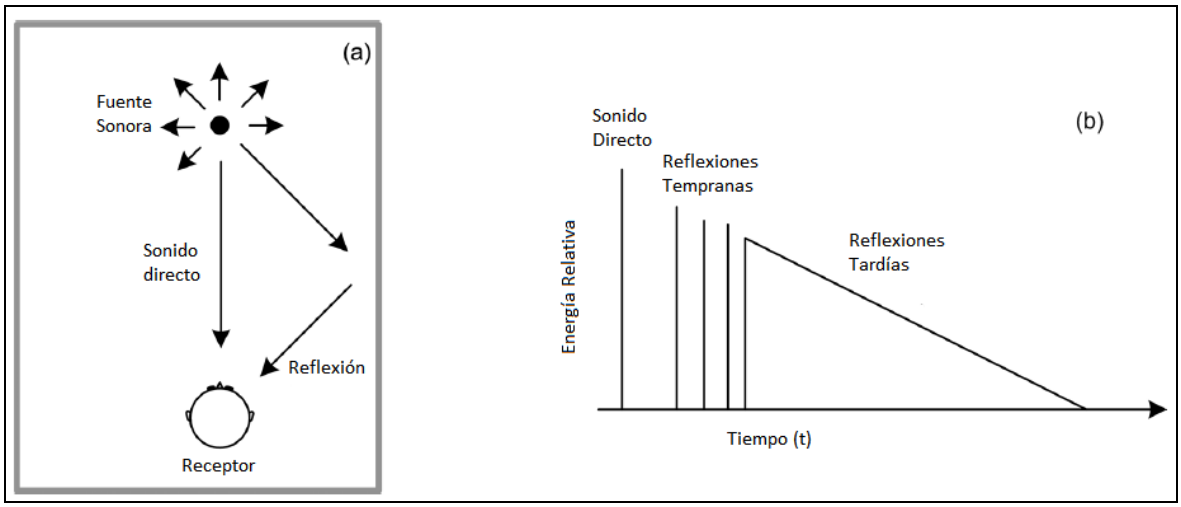

Figura 9. (a) Transmisión acústica geométrica de una sala; (b) Ejemplo típico de la respuesta al impulso en un punto dentro de una sala [15].

La relación entre el sonido directo y las reflexiones, aportan información a lo que se conoce como externalización de fuente (*sound source externalization*) y a la percepción de distancia [30].

3.3 FILTROS RELATIVOS DE LA CABEZA (HRTF)

En general, los procesos de transmisión en sistemas acústicos o de audio digital se pueden aproximar a sistemas lineales e invariantes en el tiempo (LTI). Por lo tanto, se pueden predecir sus reacciones frente a señales tanto en el dominio temporal como en el dominio de frecuencias. El temporal va a estar caracterizado por la respuesta al impulso del sistema y el de frecuencias por la función transferencia.

Como se presentó previamente, la HRTF es la función transferencia del sistema que integran una fuente sonora en el espacio, el medio de propagación y cada oído del sujeto receptor (Figura 10). Estos filtros contienen todos los factores estáticos de localización mencionados previamente.

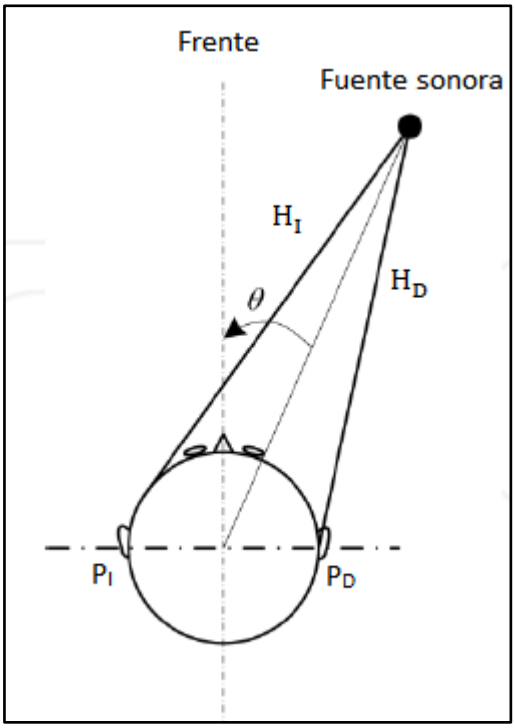

Figura 10. Diagrama de la transmisión de una fuente sonora a ambos oídos [15].

La HRTF para cada oído se define como:

$$H_I(r,\theta,\varphi,f,a) = \frac{P_I(r,\theta,\varphi,f,a)}{P_0(r,f)}, \qquad H_D(r,\theta,\varphi,f,a) = \frac{P_D(r,\theta,\varphi,f,a)}{P_0(r,f)}, \tag{2}$$

donde P_I y P_D representan el nivel de presión sonora en Pa en el oído izquierdo y derecho, respectivamente; P_0 representa el nivel de presión sonora al centro de la cabeza en ausencia de la misma, cuando el sistema se compone de una fuente sonora y un micrófono; a es un factor individual. Para $r > 1$ - 1,2 m, las HRTFs son aproximadamente independientes de la distancia (HRTFs de campo lejano). Por otro lado, para $r < 1$ m las HRTFs se vuelven sensibles con la distancia (HRTFs de campo cercano) [15].

3.3.1 Medición de las HRTFs

Las mediciones de las HRTFs por lo general utilizan la misma configuración (Figura 11), una cámara anecoica o semi anecoica, una computadora, conversores AD/DA, preamplicador, altavoces como fuentes, y micrófonos de cápsula pequeña en la entrada del canal auditivo (Figura 12).

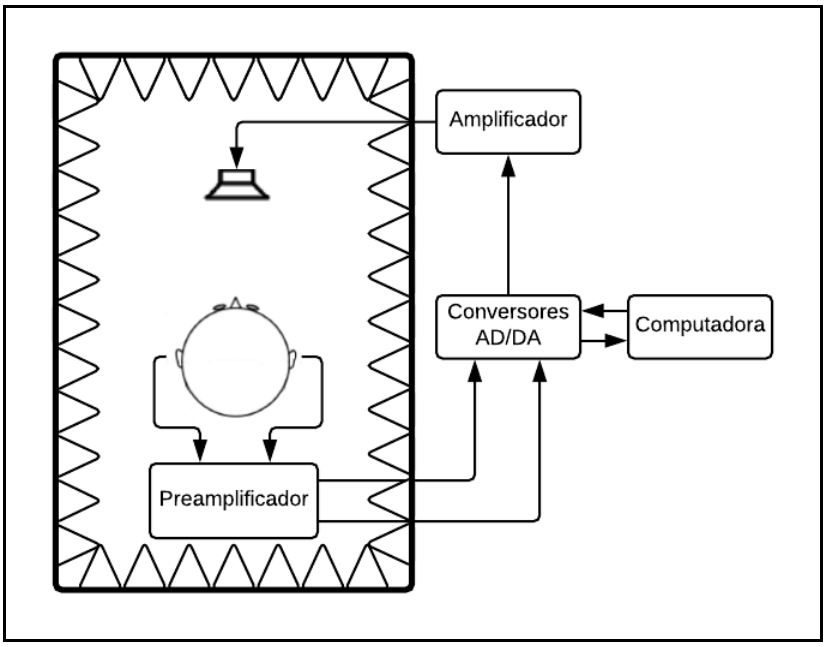

Figura 11. Diagrama de la medición de las respuestas al impulso de la cabeza en cámara anecoica.

Figura 12. Sujeto con micrófono en la entrada del canal auditivo [15].

A lo largo del tiempo se han utilizado varios tipos de señales con el objetivo de obtener las HRIR, tales como, Secuencias de Golay (*Golay Sequences*), secuencia de ruido cuasialeatoria (MLS, *Maximum Length Sequences*), barrido senoidal exponencial (ESS, *Exponential Swept-Sine*), etc. [30].

En general, los laboratorios suelen utilizar un arreglo de altavoces en forma de arco y una mesa giratoria para rotar al sujeto de medición. En la Figura 13 se muestra un ejemplo de la configuración que utilizó el Instituto de Acústica Técnica (ITA) de la Universidad de Aachen para hacer su base de datos de filtros HRTF.

Figura 13. Ejemplo de una configuración para la medición de HRIRs en este caso de una cabeza y torso genéricos. (Institude of Technical Acoustics) [31].

3.3.2 Formato de las HRTFs

A partir del 2015 el formato estándar para almacenar los filtros HRTF es SOFA. Este formato se basa en la estructura de NetCDF, el cual utiliza HDF5 (*Hierarchical Data Format*) [32]. NetCDF es un conjunto de bibliotecas y formatos de datos auto descriptivos e independientes que admiten la creación, el acceso y el uso compartido de datos científicos orientados a matrices [33]. En el caso de las HRTF, se almacenan las respuestas al impulso grabadas para cada posición de fuente dentro una matriz, al igual que toda la información relevante, por ejemplo, modelo de la fuente utilizada, micrófonos para la medición, etc. Algunas universidades e investigadores comparten sus bases de datos de manera pública en un repositorio online como parte de la comunidad SOFAcoustics [34].

3.4 SÍNTESIS BINAURAL

Para poder lograr la síntesis binaural y externalizar fuentes sonoras se requiere procesar una señal de audio con los factores estáticos, dinámicos, y ambientales. Los factores estáticos están integrados en los filtros HRTF, los factores dinámicos se logran con el desplazamiento de fuentes y los ambientales con la utilización de reflexiones tempranas y reverberación [35].

Los filtros HRTF se consideran de respuesta al impulso finita (FIR). Por lo tanto, el procesamiento se obtiene mediante la convolución de las HRIR con la señal en el dominio temporal; o lo que se conoce como una convolución basada en una transformación. La más utilizada es la Transformada de Fourier Discreta (DFT).

En la Figura 14 se muestra un diagrama del flujo de la señal para realizar la síntesis binaural.

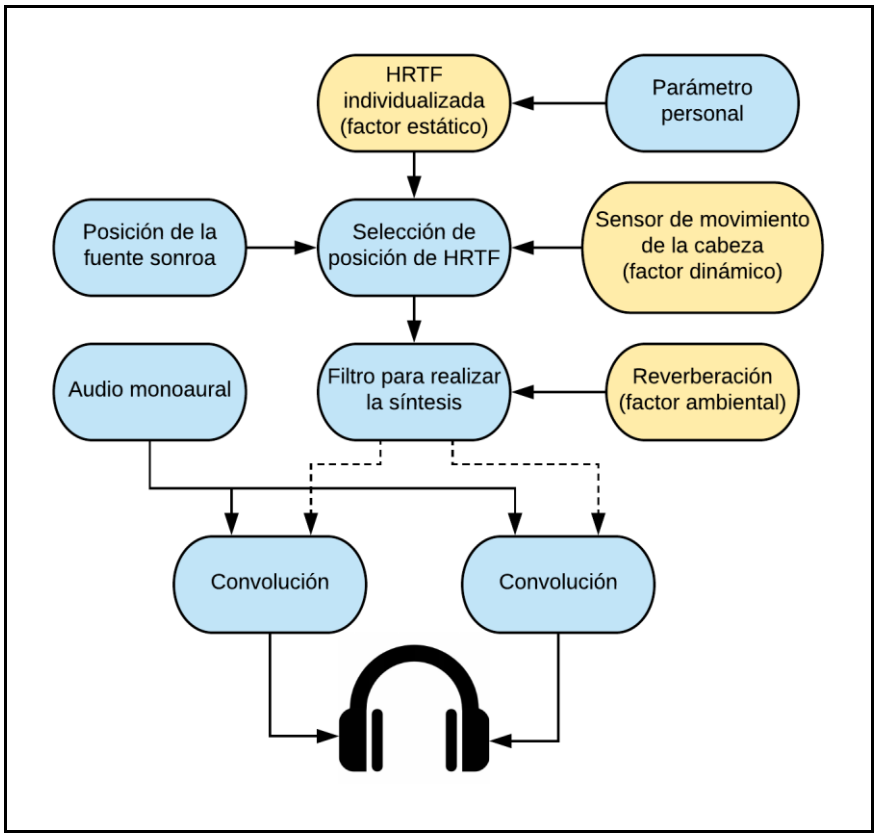

Figura 14. Diagrama del flujo del procesamiento de la señal para realizar la síntesis binaural [30].

3.4.1 Convolución en tiempo real

La convolución discreta es una operación que se define para dos secuencias *x(n)* y *h(n)* de acuerdo a la Ecuación (3).

$$y(n) = x(n) * h(n) = \sum_{k=-\infty}^{+\infty} x(n-k) \cdot h(k) \quad (3)$$

El resultado de la convolución en una secuencia *y(n)*. Generalmente, *h(n)* es la respuesta al impulso de un sistema o filtro, que suele tener una longitud finita (FIR). Si la señal *x(n)* tiene una longitud finita sería el caso de una convolución lineal, por ejemplo, de una señal de audio con la respuesta al impulso de un filtro. Si las longitudes de *x(n)* y *h(n)* son *B* y *N*, respectivamente, la longitud de *y(n)* es igual a *B*+*N*+1.

Por otro lado, si la señal *x(n)* tiene longitud infinita, en la práctica se la divide en pequeños bloques y luego se aplica la convolución. En este caso se suele llamar convolución particionada o segmentada (*partitioned convolution* o *running convolution*). Un ejemplo, es el caso de una transmisión de audio en tiempo real en donde la longitud de la señal de audio sería "infinita" y se la desea filtrar con una respuesta al impulso *h(n)* de una longitud *N.*

Muchos algoritmos de convolución rápida realizan la convolución en el dominio de alguna transformada, por ejemplo, la DFT. De esta manera, se realiza de forma más eficiente. Para este caso, se debe asumir que las secuencias son periódicas. La convolución deja de ser lineal y se la conoce como circular o cíclica. Si se le aplica la DFT de longitud *k*, la transformada de la convolución circular es igual a la multiplicación punto a punto de los coeficientes de cada secuencia (Ecuación (4)). Luego se realiza la transformada inversa de Fourier (IDFT) y se obtiene el resultado original de la convolución (Ecuación (5)).

$$DFT_k\{y(n)\} = DFT_k\{x(n) \circledast h(n)\} = X(k) \cdot H(k) \tag{4}$$

$$IDFT_n\{Y(K)\} = IDFT_n\{X(K) \cdot H(K)\} = x(n) * h(n) \tag{5}$$

Los esquemas de implementación más utilizados que realizan la convolución por medio de la DFT son *Overlap-Add* (OLA) y *Overlap-Save* (OLS) [36].

3.4.2 Overlap-Add

En la Figura 15 se muestra un esquema con la implementación de la convolución OLA utilizando la FFT. Funciona de la siguiente manera:

- El bloque de audio que ingresa con longitud B se rellena con ceros a fin de lograr la longitud K, para luego realizar la FFT.
- La respuesta al impulso de longitud N se rellena con ceros a fin de lograr la longitud K, para luego realizar la FFT.
- La transformada de la señal de entrada y el filtro se multiplican en el dominio de las frecuencias punto a punto.
- El resultado se vuelve a transformar al dominio temporal por medio de la IDFT con la función IFFT. Formando de esta manera un resultado parcial de longitud *B*+*N*-1.

- El bloque *B* de salida se obtiene de la suma con los resultados parciales de convoluciones previas de acuerdo al salto de superposición utilizado.

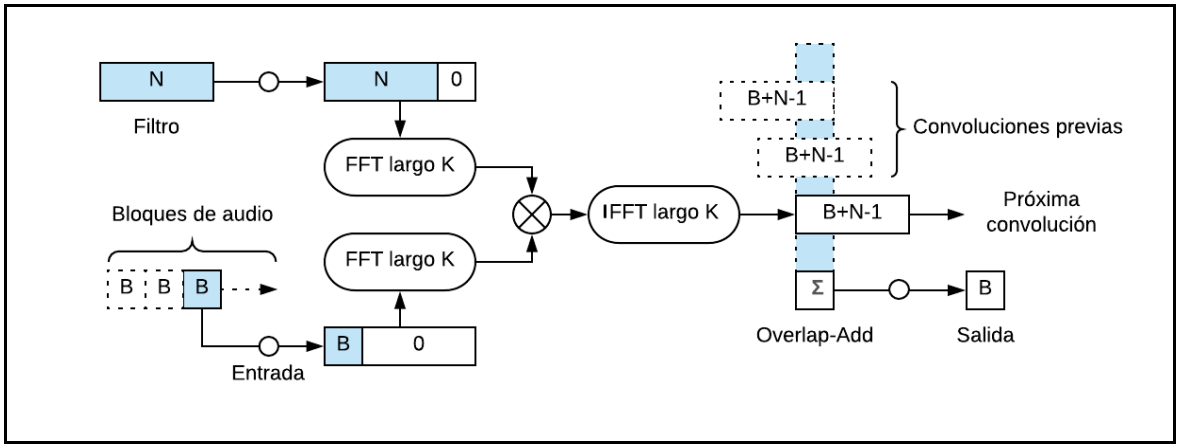

Figura 15. Esquema de la convolución Overlap-Add utilizando la FFT (Esquema adaptado [36]).

El valor de *K* se debe elegir suficientemente largo para evitar el *aliasing* temporal. En la Ecuación (6) se expresa la relación. Además, *K* debe ser un valor potencia de 2 para poder computar de manera eficiente la FFT.

$$K \geq B + N - 1 \tag{6}$$

3.4.3 Overlap-Save

El esquema con la implementación de la convolución OLS utilizando la FFT se muestra en la Figura 16. El método es similar al OLA al utilizar la convolución en el dominio de las frecuencias. Sin embargo, difiere en los siguientes aspectos:

- Para realizar la transformada del bloque B con un valor de K, se le realiza el rellenado con los valores de las muestras previas de señal hacia la izquierda. En el caso de que el bloque sea el inicio de la señal de audio se rellena con ceros.
- En la transformación al dominio temporal, las K-B muestras se descartan debido al aliasing temporal. Las B muestras del extremo derecho son guardadas para la salida.

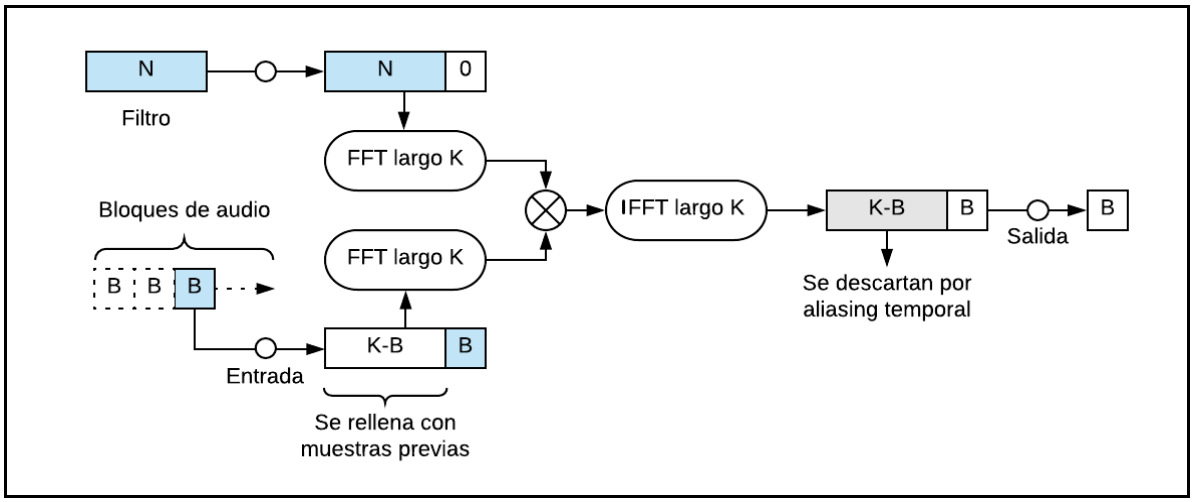

Figura 16. Esquema de la convolución Overlap-Save utilizando la FFT (Esquema adaptado [36]).

En principio, ambos métodos requieren la misma cantidad de FFTs, IFFTs y multiplicaciones. Una pequeña ventaja del OLS sobre el OLA es que se evita el paso de sumar los valores de convoluciones previas.

3.5 FUNDIDO CRUZADO DE FILTROS

La síntesis binaural dinámica incluye desplazamientos de fuentes sonoras o rotaciones del oyente receptor. Por lo tanto, requiere que los filtros vayan cambiando con el tiempo. Esta variación instantánea que se produce en el procesamiento por bloques en tiempo real produce una discontinuidad audible en la forma de onda de la señal. La solución que propone Wefers [36] es realizar un fundido cruzado (*crossfading*) entre bloques procesados con los filtros previos y los nuevos filtros.

En la Ecuación (7) se expresan los resultados del procesamiento de un bloque de longitud *B* de señal $x(n)$ con cada par de filtros; siendo $h_0(n)$ los filtros de la posición previa, y $h_1(n)$ los filtros de la nueva posición:

$$y_0(n) = x(n) * h_0(n) \qquad \text{y} \qquad y_1(n) = x(n) * h_1(n) \tag{7}$$

Luego, para atenuar las posibles discontinuidades se aplican dos funciones envolventes. $f_{in}(n)$ como fundido de entrada y $f_{out}(n)$ como fundido de salida, ambas con una longitud $L \leq B$:

$$y(n) = \begin{cases} y_0(n) \cdot f_{out}(n) + y_1(n) \cdot f_{in}(n) & n<L \\ y_1(n) & en\ otro\ caso \end{cases} \tag{8}$$

La longitud *L* de fundido suele rondar entre las 8 y 32 muestras en aplicaciones de síntesis binaural [36]. Las funciones envolventes se definen para valores $0 \leq n < L$. $f_{out}(n)$ es monótona decreciente y $f_{in}(n)$ es monótona creciente. Es fundamental que la suma de ambas envolventes mantenga una amplitud constante. Si no es el caso, se produce una modulación a la salida:

$$f_{out}(n) + f_{in}(n) = 1 \qquad (0 \leq n < L) \tag{9}$$

Dos ejemplos de envolventes que se suelen utilizan para procesamiento de audio:

Lineal $f_{out}(n) = 1 - \frac{n}{L}$ $f_{in}(n) = \frac{n}{L}$ (10)

Coseno cuadrado $f_{out}(n) = cos^2(\frac{\pi n}{2L})$ $f_{in}(n) = sin^2(\frac{\pi n}{2L})$ (11)

3.5.1 Fundido temporal

El fundido cruzado de la Ecuación (8) se puede integrar sin inconvenientes al algoritmo de OLS. En la Figura 17 se muestra el esquema del algoritmo. Se realizan dos convoluciones en paralelo, por un lado los filtros previos $h_0(n)$; y por otro los nuevos filtros $h_1(n)$. El fundido cruzado se realiza luego multiplicando las envolventes ($f_{out}(n)$ y $f_{in}(n)$) con las señales procesadas en el dominio temporal. Finalmente, se suman los resultados.

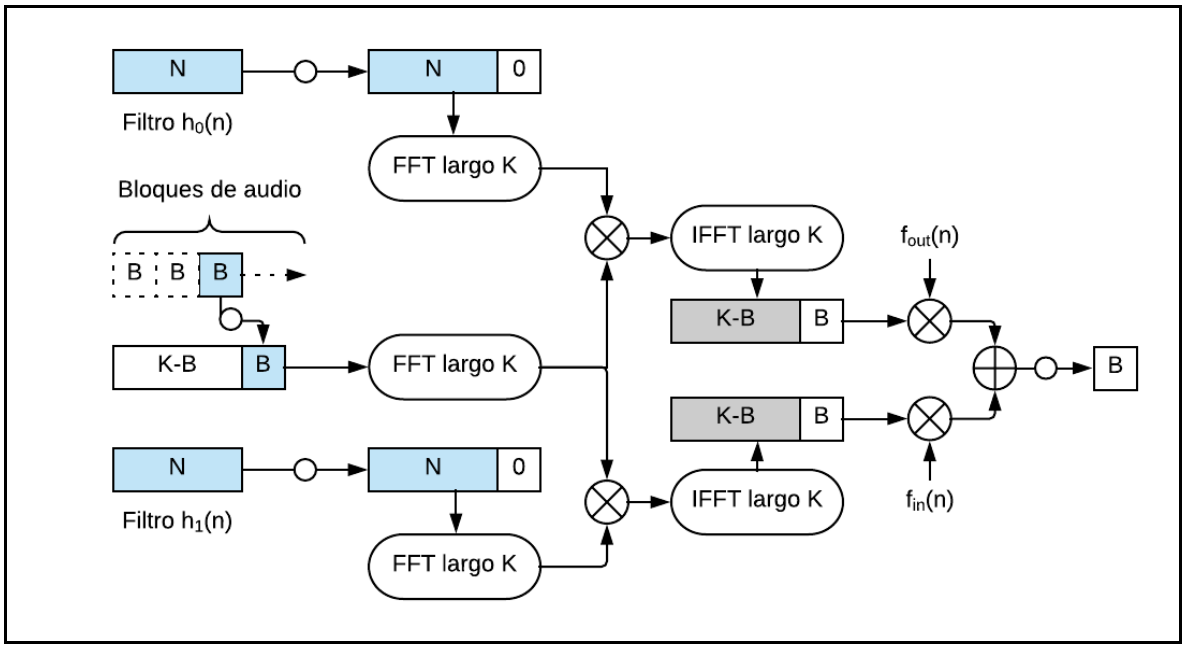

Figura 17. Esquema de la convolución Overlap-Save con intercambio de filtros utilizando fundido cruzado en el dominio temporal [36].

3.5.2 Fundido espectral

Otro método que propone Wefers es realizar el fundido en el dominio de las frecuencias [36-37]. En la Figura 18 se muestra el esquema con la implementación dentro del algoritmo de OLS. El beneficio es que se puede ahorrar la segunda IFFT en comparación con el fundido temporal. Sin embargo, la técnica solo aplica para el esquema OLS, y se encuentra limitada al tener que utilizar el fundido en la longitud completa del bloque *B*.

La Ecuación (8) se puede expresar en el dominio de las frecuencias como

$$y(n) = y_0(n) \cdot f_{out}(n) + y_1(n) \cdot f_{in}(n) \tag{12}$$

$$\downarrow \qquad \downarrow \qquad \downarrow$$

$$Y(k) = Y_0(k) \circledast F_{out}(k) + Y_1(k) \circledast F_{in}(k) \tag{13}$$

El fundido de las señales $y_0(n)$ y $y_1(n)$ en el dominio temporal corresponden a convoluciones circulares de longitud *K* en el dominio espectral con las transformadas discretas de Fourier $F_{out}(k)$ y $F_{in}(k)$ de las envolventes $f_{out}(n)$ y $f_{in}(n)$.

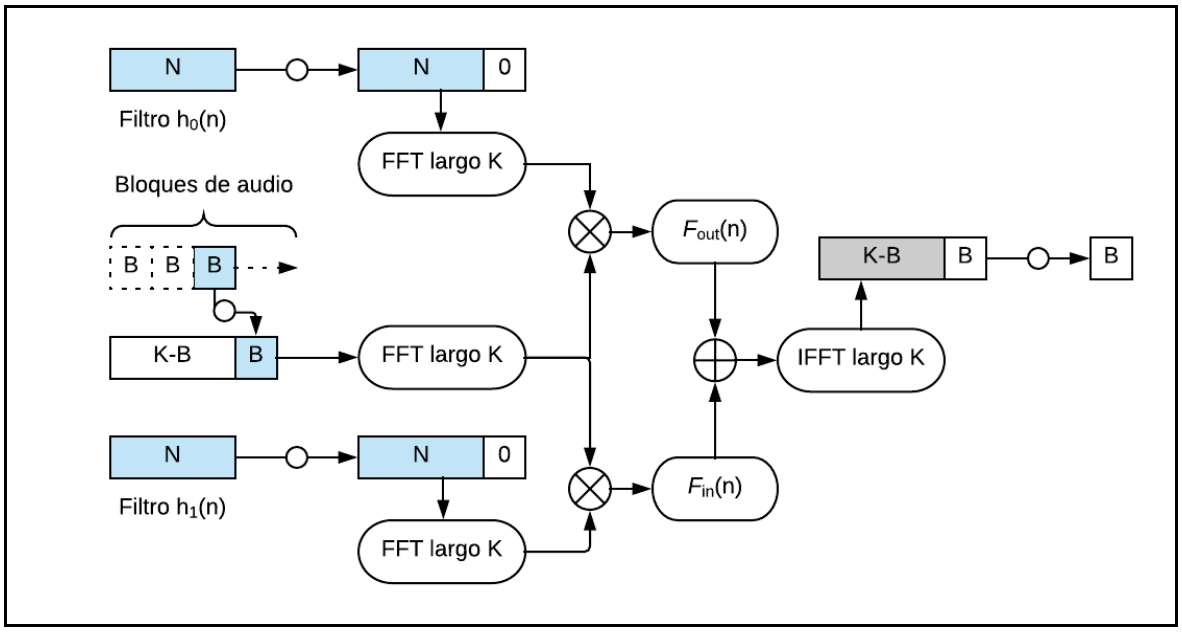

Figura 18. Esquema modificado de convolución Overlap-Save con fundido cruzado en el dominio de las frecuencias [36].

Esta implementación es más eficiente que el fundido temporal si las operaciones de las convoluciones circulares son menores que las que realiza una IFFT. Para lograrlo, se aprovecha la zona de descarte del esquema de OLS para utilizar una función cosenoidal

periódica. Este principio se esquematiza en la Figura 19. Particularmente, la envolvente que se adapta en este caso es el coseno cuadrado:

$$f_{out}(n) = sin^2\left(\frac{\pi nP}{K}\right) = \frac{1}{2} - \frac{1}{2}cos\left(\frac{2\pi nP}{K}\right) \quad (14)$$

$$f_{in}(n) = cos^2\left(\frac{\pi nP}{K}\right) = \frac{1}{2} + \frac{1}{2}cos\left(\frac{2\pi nP}{K}\right) \quad (15)$$

siendo *K* el número de muestras espectrales, y *P* el número de periodos. Es estrictamente necesario que las *B* muestras del extremo derecho coincidan con la mitad de un periodo de la envolvente (Figura 19). Esto requiere que el tamaño de la transformada, *K,* sea un múltiplo entero del doble de longitud del bloque *B.*

$$P = \frac{K}{2B} \quad (16)$$

La transformada discreta de Fourier de las Ecuaciones (14) y (15) genera solo tres coeficientes reales.

$$F_{out}(k) = DFT\big(f_{out}(n)\big)$$

$$F_{out}(k) = K(-\frac{1}{4}\delta(k+P) + \frac{1}{2}\delta(k) - \frac{1}{4}\delta(k-P)) \quad (17)$$

$$F_{in}(k) = DFT\big(f_{in}(n)\big)$$

$$F_{in}(k) = K(\frac{1}{4}\delta(k+P) + \frac{1}{2}\delta(k) + \frac{1}{4}\delta(k-P)) \quad (18)$$

Por ejemplo, para el caso de *K*, *B*, *P* iguales a 1024, 512, 1, respectivamente, los coeficientes son:

$$F_{in,out}(k) = 1024 \cdot [\ \frac{1}{2}\ \ \pm\frac{1}{4}\ \ 0\ \ \ldots\ \ 0\ \ \ \pm\frac{1}{4}\ ] \quad (19)$$

Finalmente, la Ecuación (13) se puede expresar como:

$$Y(k) = \frac{K}{2}(\ Y_0(k)\ + Y_1(k) + \frac{1}{2}(\ Y_1(k+P) - Y_0(k+P) + Y_1(k-P) - Y_0(k-P)) \quad (20)$$

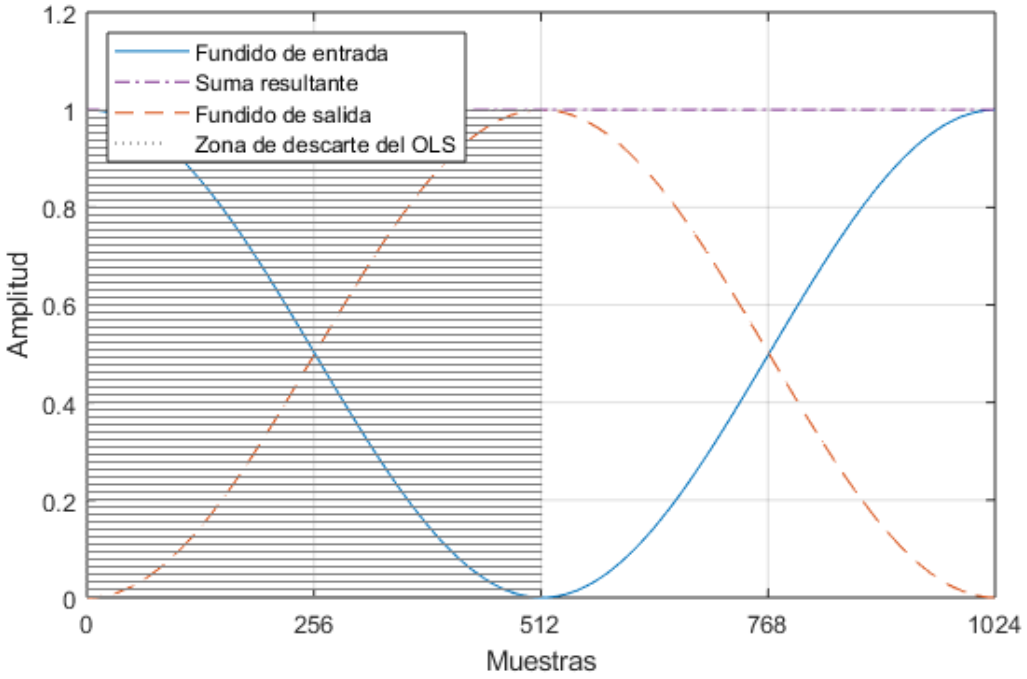

Figura 19. Esquema de la función envolvente cosenoidal periódica utilizando la zona de descarte del método OLS. Con *K, B, P* iguales a 1024, 512, 1, respectivamente [36].

Wefers realiza una comparación entre los costos computacionales entre ambos métodos [36]. Llega a la conclusión de que el fundido espectral solo incrementa entre 17-34 % el costo computacional, en cambio, el fundido temporal incrementa entre 50-60 %. Sin embargo, el fundido temporal tiene la ventaja de ser más flexible al poder ajustar la longitud del fundido dependiendo la aplicación. No se encuentra limitado a realizar el fundido en un bloque entero, como es el caso del fundido espectral.

CAPÍTULO IV - DISEÑO DEL SISTEMA

El núcleo de la investigación se centró en el procesamiento de señales de audio con los filtros HRTF a fin de lograr la síntesis binaural.

Para realizar la implementación se empleó el entorno de Matlab por ser un lenguaje de programación que se aplicó a lo largo de la carrera. Por medio de las funciones del *Audio System Toolbox* (AST) se pudo lograr un procesamiento en tiempo real de forma simple [38].

4.1 PROCESAMIENTO EN TIEMPO REAL

Inicialmente se planteó un sistema con el fin de leer un archivo de audio y reproducirlo. El AST utiliza sistemas de objetos con el objetivo de poder realizar el procesamiento en tiempo real. Los mismos están optimizados para realizar iteraciones de grandes cantidades de flujo de bloques de datos (en inglés, los bloques y el flujo se conocen como *frames* y *audio stream*, respectivamente).

Con el fin de leer un archivo de audio se utilizó la función dsp.AudioFileReader [39]. Esto permite cargar el archivo de audio en pequeños bloques de datos al espacio de trabajo de Matlab. Cada bloque que lee el objeto está compuesto por un número de muestras determinado. El valor por defecto suelen ser 1024 muestras por bloque.

Luego, en la reproducción de los bloques se empleó la función dsp.audioDeviceWriter [40]. La misma se encarga de enviar las muestras de audio por medio del driver al buffer de salida de la placa de audio, después al conversor D/A y por último a los auriculares. En la Figura 20 se muestra un esquema con el flujo de las muestras de audio. El bloque de “Proceso” corresponde al procesamiento de la señal mediante los filtros.

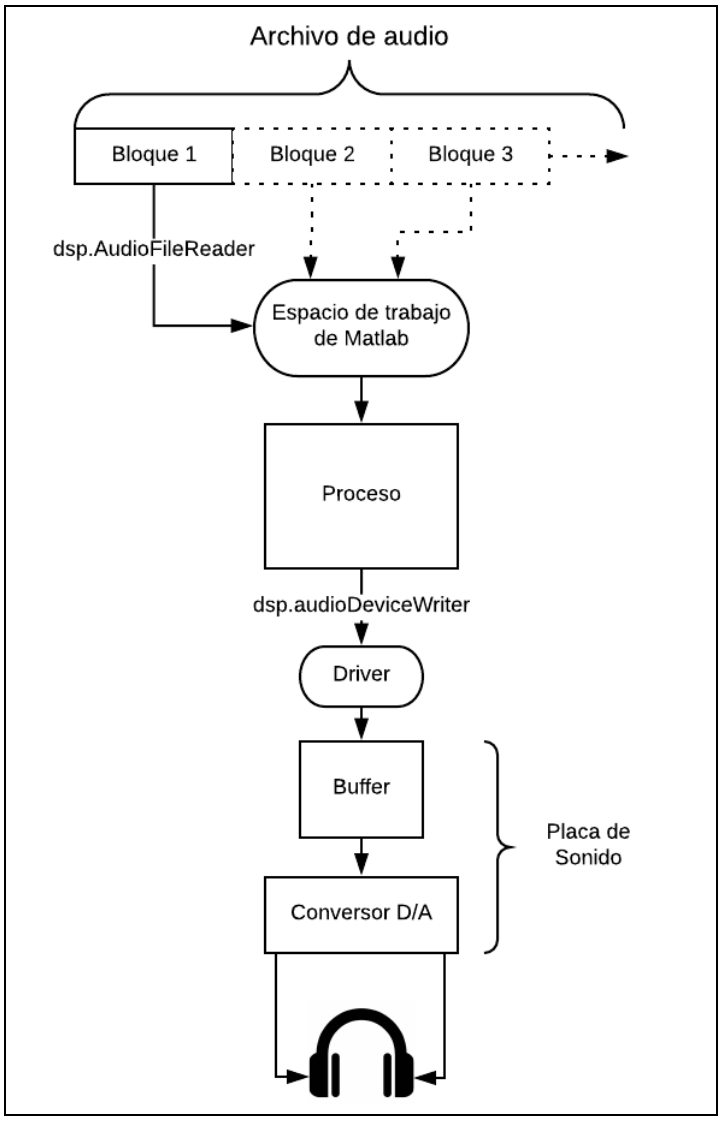

Figura 20. Esquema con el flujo de muestras de audio.

4.2 FILTROS HRTF

El objetivo de cargar las HRIR al entorno de Matlab se logró por medio de una interfaz de programación de aplicaciones (API) desarrollada por Majdak [41]. La misma se encuentra bajo la Licencia Pública de la Unión Europea V.1.1 (EUPL). La API consta de funciones diseñadas para escribir y leer los archivos SOFA. En este caso, solo se recurrió a las funciones capaces de leer la información.

Para poder utilizar el API se necesitó agregar el directorio al espacio de trabajo de Matlab, luego, ejecutar la función “SOFAstart” que se encarga de agregar las carpetas y funciones disponibles. La información de un archivo SOFA se logró leer mediante la función “SOFAload”. La misma utiliza otra función llamada “NETCDFload” que se ocupa de cargar toda la información dentro de un objeto. Dentro del objeto se encuentras las respuestas al

impulso almacenadas en una matriz de *M x R x N*, además de toda la información respecto a la medición en forma de metadatos. *M* corresponde a la cantidad de mediciones para cada posición de fuente; *R* es la cantidad de receptores, para las HRIR corresponde al valor 2, oído izquierdo y derecho; *N* es la cantidad de muestras de cada respuesta al impulso.

La función "SOFAcalculateAPV" se empleó con el propósito de generar una matriz con la información de la posición de la fuente aparente (grados en el plano horizontal, grados en el plano vertical y distancia en metros). De esta manera, se simplifica la búsqueda de un índice correspondiente a un determinado par de grados azimutal y elevación. Luego, con este índice se obtienen las respuestas al impulso correspondientes dentro de la matriz con los filtros.

4.3 MÉTODO DE CONVOLUCIÓN

EL método de convolución a utilizar se evaluó por medio de una comparación de la latencia entre el OLA, OLS y la función dsp.FIRFilter del *Audio System Toolbox*. Estos tres métodos realizan el filtrado de la señal con los mismos resultados. Los códigos con la implementación del OLA y OLS se encuentran en el ANEXO A. En esta prueba se utilizó la versión de Matlab R2018a con una computadora Dell i5-7200u, CPU 2.5 GHz, 8 GB de RAM y Windows 10.

Los filtros HRTF se obtuvieron de la base de datos pública de la cabeza maniquí KEMAR publicada por el MIT. Los mismos usan 512 muestras a una frecuencia de muestreo de 44,1 KHz. La fuente virtual se dejó fija a 0° azimutal y 0° elevación. Como señal de prueba se generó un tono puro de 1 kHz con duración de 20 s a una frecuencia de muestreo de 44,1 KHz. Se realizó un bucle para filtrar la señal 100 veces con cada método. Con la función "Tic-Toc" de Matlab se midió el tiempo que se tardó en realizar solamente el proceso de convolución para cada bloque de audio. Al final de cada ciclo se realizó un promedio del tiempo acumulado sobre el número de bloques de audio procesados. En la Figura 21 se muestran los resultados de la latencia para los tres métodos. Los resultados de los valores promedio se presentan en la Tabla 1.

Tabla 1. Resultados promedios de las pruebas de latencia para los tres métodos.

	OLA	OLS	dsp.FIRFilter
Latencia promedio [ms]	0,141 ± 0,028	0,109 ± 0,022	0,209 ± 0,009

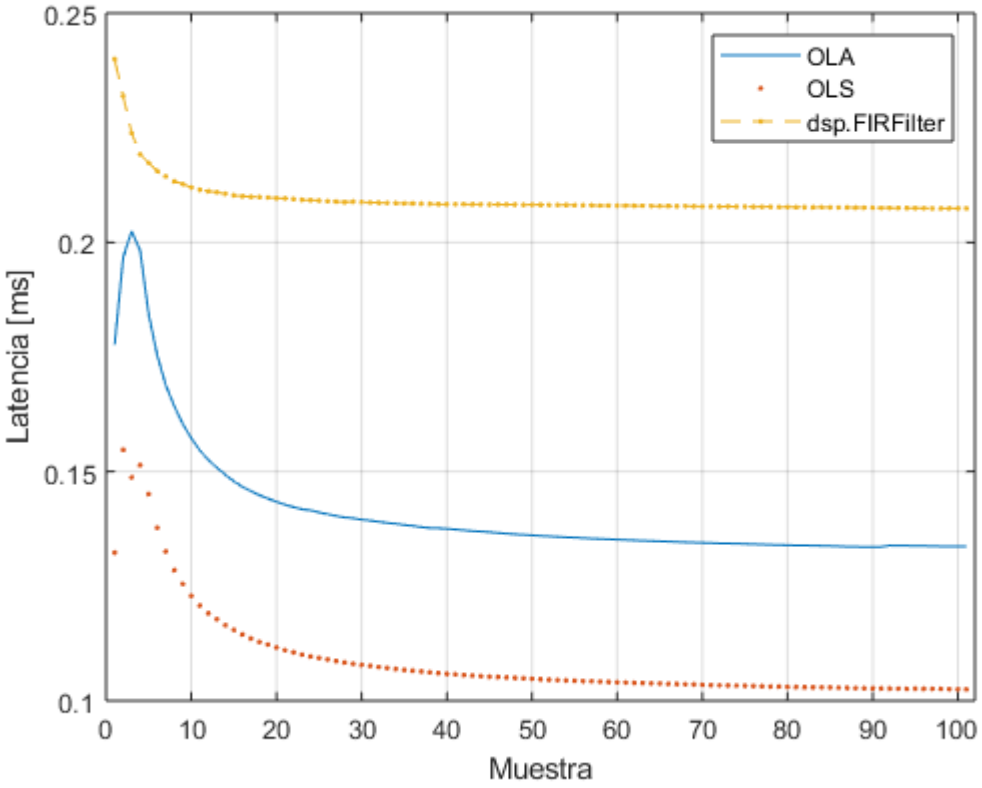

Figura 21. Latencia para los métodos de convolución OLA, OLS y la función dsp.FIRFilter.

El objetivo de esta prueba no fue hallar un valor absoluto, sino comparar la eficiencia entre los tres métodos propuestos. Los resultados de la prueba confirmaron que el método más eficiente fue el OLS para la configuración de filtros propuesta.

4.4 FILTROS VARIANTES CON EL TIEMPO

Una parte importante en la síntesis binaural es el desplazamiento de la fuente sonora. Una primera aproximación se logró variando las HRIR de manera discreta dentro del bucle de procesamiento. Para variar los ángulos en tiempo real se crearon objetos que almacenaban los valores de los ángulos, y por medio de una interface básica de usuario con dos deslizadores se pudieron variar los valores de los ángulos.

El problema que surgió al realizar la variación de ángulos, y por ende de HRIR para realizar las convoluciones, es que tras realizar los cambios de ángulos se producen discontinuidades en la forma de onda a la salida. En la Figura 22 se muestran ejemplos de la discontinuidad en el dominio temporal producida por el cambio de filtros durante el procesamiento. Estos ejemplos, son las señales resultantes que se obtuvieron al procesar un tono puro de 1 KHz y variar los filtros para un salto discreto de ángulo vecino y un salto

de ángulo grande en el plano horizontal. Estas discontinuidades se perciben como “clics” (en la literatura también pueden aparecer como artefactos audibles o *audible artifacts*).

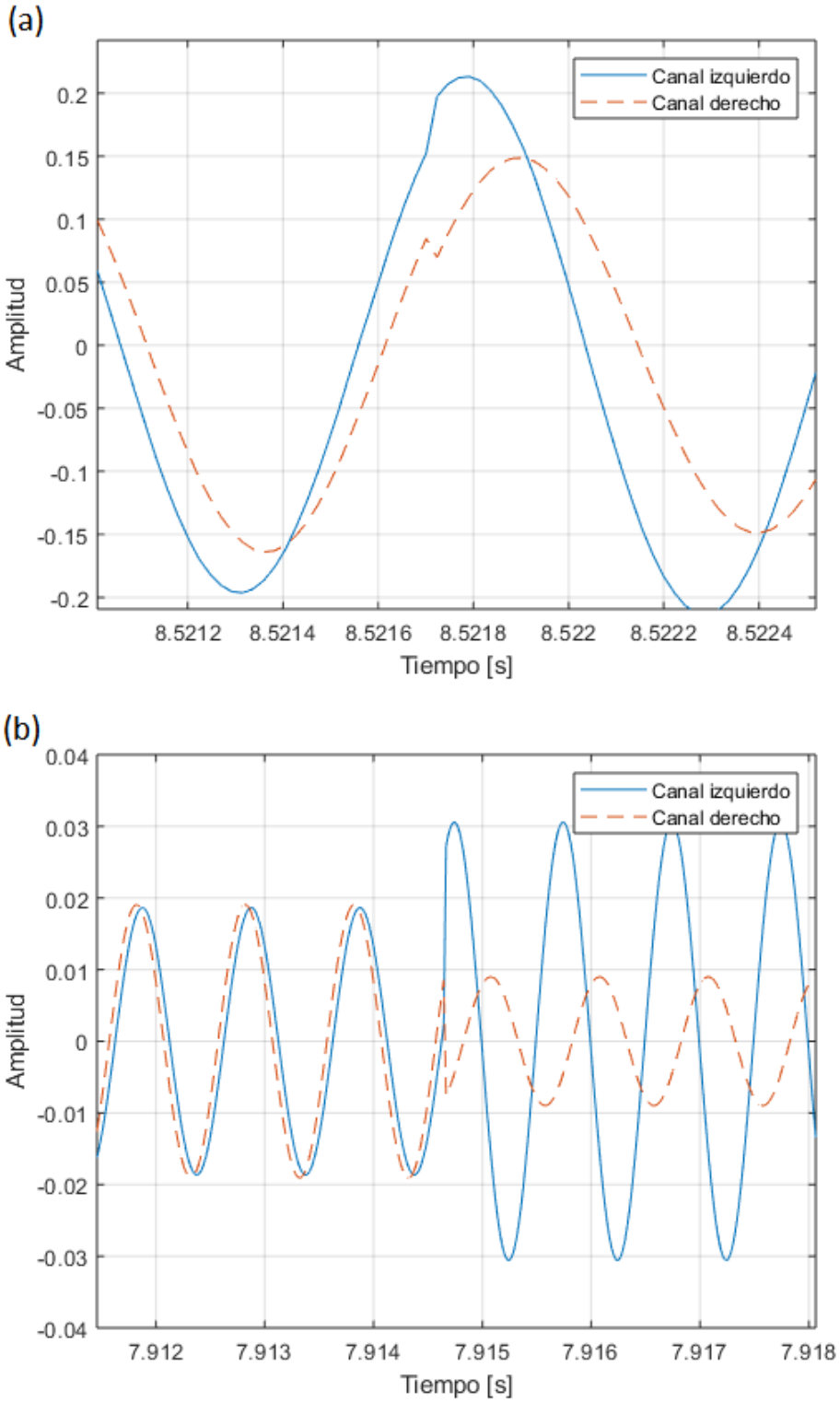

Figura 22. Ejemplos de discontinuidad de la señal producida por el cambio de filtros al variar el ángulo azimutal. Como señal de prueba se utiliza un tono puro de 1 kHz. (a) corresponde a un salto discreto de un ángulo vecino; (b) corresponde a un salto de ángulo grande.

La solución fue implementar un fundido cruzado entre los filtros de la posición de fuente previa y la nueva posición. La teoría de la implementación se explicó en la sección 2.6.

4.4.1 Implementación de fundido cruzado

Con el propósito de analizar la eficiencia de los algoritmos de fundido cruzado se realizaron ambas implementaciones, tanto en el dominio de las frecuencias como en el temporal. El objetivo fue realizar una comparación de la latencia generada por cada algoritmo. En el ANEXO B se muestras los códigos de ambas funciones.

Se adoptó la misma configuración de hardware y software que en la sección 3.3. Se generó un bucle con la finalidad de procesar mil veces el mismo segmento de audio de un tono puro, dejando fijos dos pares de filtros para simular la transición de posición fuente sonora. Mediante la función “Tic-Toc” se midió la latencia de cada iteración. En la Figura 23 se muestras los resultados obtenidos de la prueba. En la Tabla 2 se presentan los valores promedio para ambos fundidos.

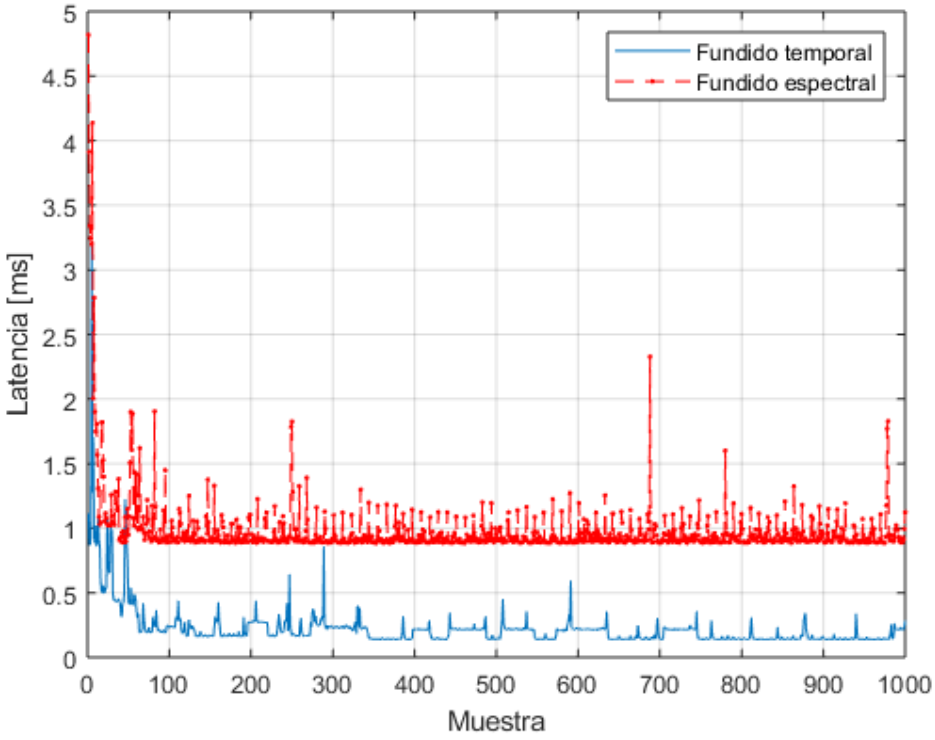

Figura 23. Resultados de latencia de los algoritmos de convolución OLS con la implementación de fundido cruzado temporal y espectral detallados en el ANEXO B.

Tabla 2. Resultados promedios de las pruebas de latencia para los tres métodos.

	Fundido Temporal	Fundido Espectral
Latencia promedio [ms]	0,223 ± 0,174	0,391 ± 0,255

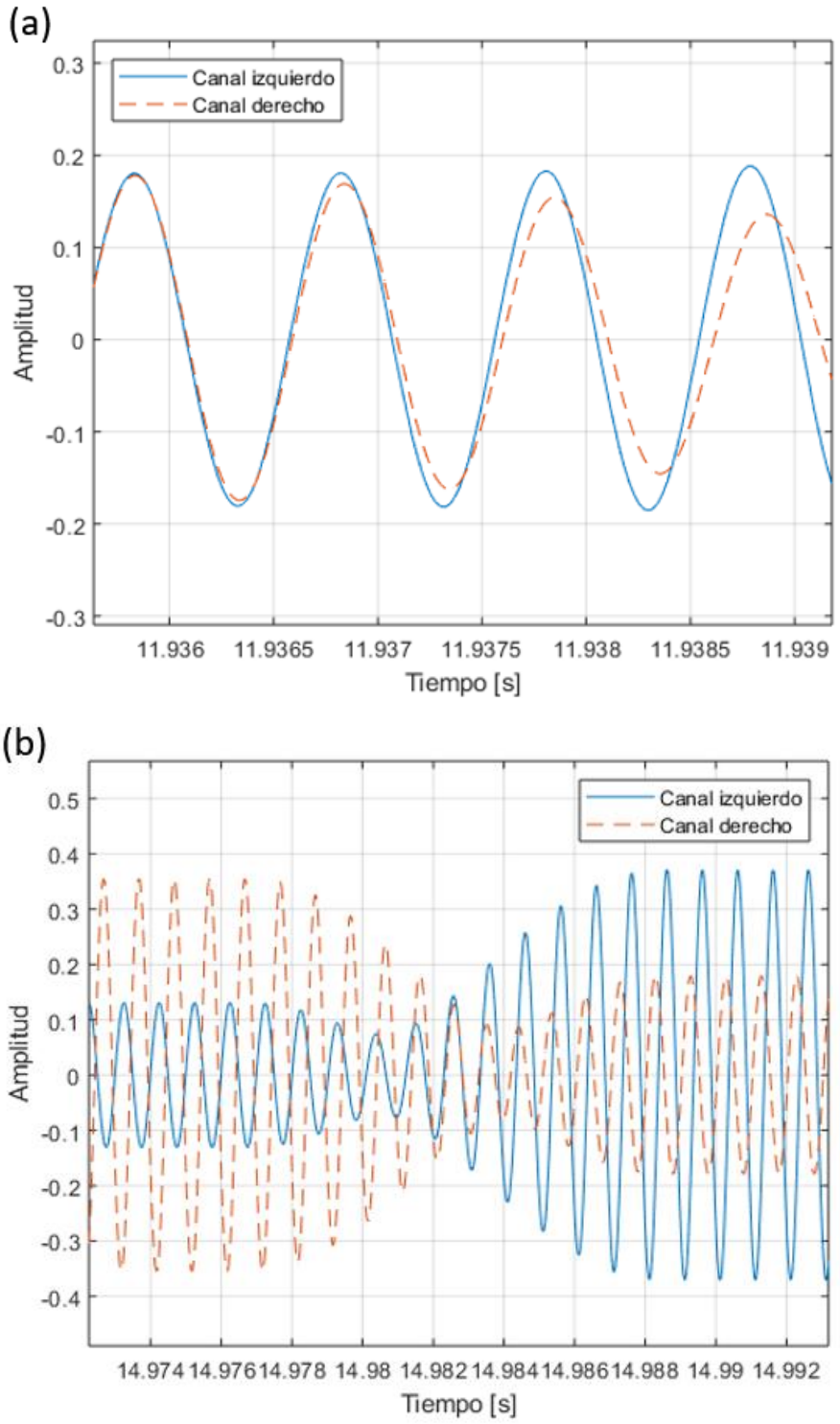

Figura 24. Ejemplos de una señal procesada con los filtros HRTF con variación del ángulo azimutal luego de realizar el fundido cruzado temporal detallado en el ANEXO B. Como señal de prueba se utiliza un tono puro de 1 kHz. (a) corresponde a un salto discreto de un ángulo vecino; (b) corresponde a un salto de ángulo grande.

Los resultados muestran que el fundido temporal es más eficiente para la configuración seleccionada, a diferencia de los resultados obtenidos por Wefers. Puede deberse a que la implementación del fundido espectral en el código de Matlab no es lo suficientemente eficiente, a pesar de que las operaciones matemáticas sean menores. Este tema se puede seguir profundizando en investigaciones futuras. No obstante, esta implementación se procedió a utilizar el fundido temporal, debido a su mejor rendimiento y versatilidad frente al fundido espectral. En la Figura 24 se muestran dos tonos puros procesados con los filtros HRTF con variación del ángulo azimutal luego de realizar el fundido cruzado temporal. Es evidente que la señal no presenta discontinuidad luego de variar los filtros.

4.5 DISEÑO DEL SENSOR DE MOVIMIENTO

Como se mencionó anteriormente, el movimiento de la cabeza del oyente cumple un rol fundamental en la localización de la fuente. Por este motivo, es necesario utilizar un sensor de movimiento para obtener las variaciones de ángulos de la cabeza.

Los dispositivos utilizados más comúnmente son los sensores inerciales (IMU, *inertial measurement units*) con tecnología MEMS (*microelectromechanical systems*, sistemas microelectromecánicos). Los sensores inerciales están formados por acelerómetros y giroscopios. Los mismos tienen la capacidad de medir fuerzas y velocidades, lineales y angulares. Con la utilización de los algoritmos adecuados se pueden obtener las variaciones de los ángulos de rotación [42]. En general, estos sensores se combinan con un magnetómetro con el propósito lograr una medición referida al campo magnético de la tierra. En estos casos se suelen llamar MARG (*Magnetic, Angular Rate, and Gravity*) o AHRS (*Attitude and Heading Reference Systems*). El magnetómetro otorga la referencia estable la cual permite corregir un error que produce el giroscopio de desvío (*drift*) [43].

Estos dispositivos se volvieron de uso frecuente en navegación de aeronaves, embarcaciones o vehículos terrestres con un alto grado de confiabilidad. En los últimos años, gracias a su reducido costo de fabricación se han vuelto masivos en consolas de videojuegos, tabletas, *smartphones* y sensores para el cuidado de la salud y para deportes. Por esta razón, se optó por estos dispositivos en el diseño del sistema de sensor de movimiento de la cabeza.

El sensor elegido para esta investigación fue el MPU-9250. Es un circuito integrado múltiple desarrollado por Invensense que contiene el acelerómetro y giroscopio del MPU-6500, ambos de tres ejes, y el magnetómetro AK8963 de Asahi Kasei Microdevices Corporation, también de tres ejes. Posee un conversor AD de 16 bits por cada eje del giroscopio, acelerómetro y magnetómetro. El giroscopio posee un rango selectivo de ±250, ±500, ±1000 y ±2000 °/s; el acelerómetro posee un rango selectivo de ±2, ±4, ±8 y ±16 g; y el magnetómetro posee un rango de ±48 Gauss. Permite comunicación SPI o I2C [44].

Debido a su reducido tamaño (3x3x1 mm) se optó por adquirir la placa comercial GY-9265, la cual incluye el sensor y conectores para alimentar el integrado y extraer las señales. En la Figura 25 se muestra el integrado montado en la placa comercial con identificación de los ejes.

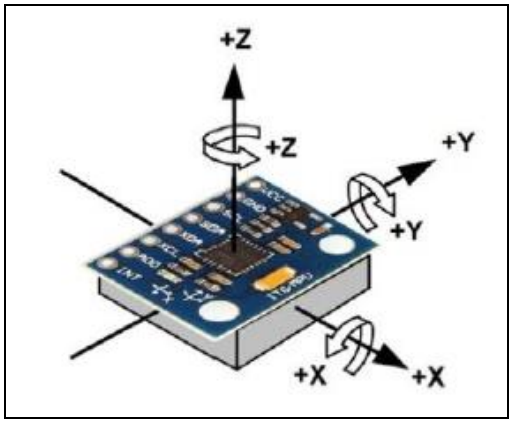

Figura 25. Integrado montado en la placa GY-9265 con la identificación de los ejes. Esquema adaptado de las especificaciones del fabricante [44].

Los ángulos de rotación en tres ejes están asociados comúnmente a la navegación de aeroplanos. Los mismos son *yaw* (dirección), *pitch* (elevación) y *roll* (alabeo) [45]. Tomando como origen del sistema de coordenadas el centro de la cabeza se pueden trasladar los ángulos de rotación antes mencionados. El ángulo *yaw* es rotación sobre el eje perpendicular al plano horizontal o eje z, el ángulo *pitch* es sobre el eje perpendicular al plano medio o eje y, y el ángulo *roll* sobre el eje perpendicular al plano frontal o eje x. En la Figura 26 se muestra un esquema de lo mencionado.

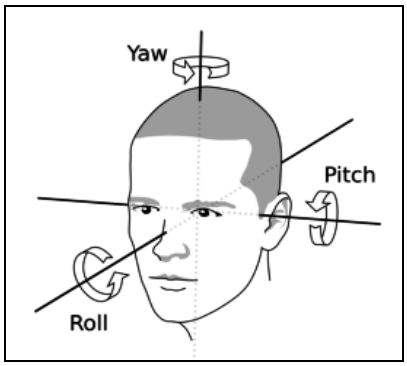

Figura 26. Esquema con los ángulos de rotación tomando como origen del sistema de coordenadas el centro de la cabeza.

4.5.1 Representación en cuaterniones

Con la finalidad de obtener los ángulos de rotación previamente mencionados, primero se representa la orientación en cuaterniones (*Quaternions*), un vector con cuatro componentes complejas que permiten representar la posición en el espacio de un objeto tridimensional. Suponiendo un sistema de ejes de coordenadas (x_A, y_A, z_A) denominado *A*, una rotación arbitraria de otro sistema de coordenadas (x_B, y_B, z_B) *B* referido a *A* se puede obtener mediante la rotación de un ángulo θ alrededor de un eje ${}^{A}\hat{r}$ definido en *A*. Esta representación se muestra en la Figura 27. El cuaternión que describe esta orientación, ${}^{A}_{B}\hat{q}$, se define en la Ecuación (21).

$$ {}^{A}_{B}\hat{q} = [q_1 \ q_2 \ q_3 \ q_4] = [cos\frac{\theta}{2} \ -r_x \ sin\frac{\theta}{2} \ -r_y \ sin\frac{\theta}{2} \ -r_z \ sin\frac{\theta}{2}] \qquad (21) $$

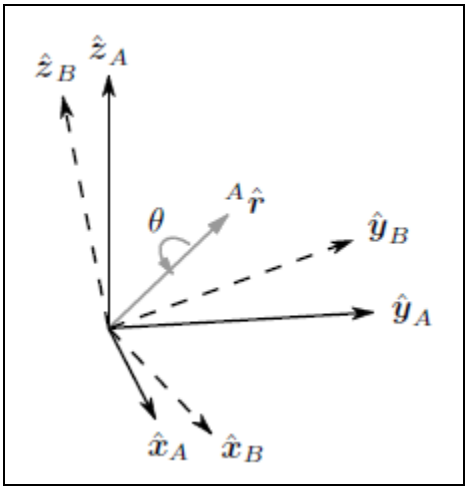

Figura 27. Rotación del sistema B alrededor del eje ${}^{A}\hat{r}$ y con ángulo ϑ referido al sistema A [46].

Los ángulos de rotación *yaw*, *pitch* y *roll*, que se obtienen de la rotación sobre cada eje, se definen en las Ecuaciones (22-24) [47].

$$yaw = arctan(2 \cdot (q_1 \cdot q_2 + q_0 \cdot q_3)/(q_0^2 + q_1^2 - q_2^2 - q_3^2)) \quad (22)$$

$$pitch = arcsen(2 \cdot (q_1 \cdot q_3 - q_0 \cdot q_2)) \quad (23)$$

$$roll = arctan(2 \cdot (q_0 \cdot q_1 + q_2 \cdot q_3)/(q_0^2 + q_1^2 - q_2^2 + q_3^2)) \quad (24)$$

Con el propósito de obtener cada valor del cuaternión (*Q*) se suelen utilizar lo que se conoce como filtros de fusión de sensores. La estructura básica de estos filtros asume que la variación del cuaternión se puede pensar como función de los valores medidos por el giroscopio (*G*) en un delta de tiempo para cada eje:

$$Q(t) = \sum_t (Q(t) - 1) \cdot (1 + \frac{1}{2} G\Delta t) \quad (25)$$

Luego se le agrega un término para corregir errores en donde se halla la diferencia entre el cuaternión referido al sistema de aceleración dado por la gravedad de la tierra (*AQ*) con los valores medidos por el acelerómetro (*Amed*), también, se introduce una constante β que controla el grado de convergencia.

$$Q(t) = \sum_t (Q(t) - 1) \cdot (1 + \frac{1}{2} G\Delta t) + (AQ - Amed) \cdot \beta \cdot \Delta t \quad (26)$$

Finalmente, se le agrega otra referencia ortogonal registrada por el magnetómetro. De esta manera, se logra reducir el error de desvío en el ángulo de *yaw*. Dentro del término de corrección, se compara el cuaternión estimado por el campo magnético de la tierra (*MQ*) con los valores medidos por el magnetómetro (*Mmed*).

$$Q(t) = \sum_t (Q(t) - 1) \cdot (1 + \frac{1}{2} G\Delta t) + (AQ - Amed) + (MQ - Mmed) \cdot \beta \cdot \Delta t \quad (27)$$

Los filtros que suelen aparecer en la literatura son los de Kalman, Mahony y Madgwick. Todos utilizan la misma estructura básica, pero difieren en los métodos de construcción y aproximación para corregir los errores. El más común y más complicado de implementar es el filtro de Kalman. Es el más preciso en resultados, pero requiere niveles complejos de inversión de matrices que suelen requerir un procesador rápido capaz de manejarlo. Por otro lado, los filtros de Mahony y Madgwick están optimizados para poder funcionar en pequeños procesadores con una precisión muy buena [43]. En esta

implementación se aplicó el filtro de Madgwick por su bajo requerimiento de procesador y además por ser de código libre.

4.5.2 Microcontrolador

Con el fin de leer los datos de los sensores, procesar la información y transmitir la información a una computadora fue necesario utilizar un microcontrolador. No es más que una pequeña computadora dentro de un circuito integrado. Tiene la capacidad de ser programado. Cuenta con una CPU (unidad central de procesamiento), memoria, entradas y salidas.

Para la elección del microcontrolador se tuvieron en cuenta las siguientes condiciones: costo, stock en comercios, lenguaje de programación y de fácil conexión a fin realizar un prototipo rápido. Se optó por utilizar el ATmega328 integrado en la placa Arduino Nano 3.0. Sus principales características son: 14 conectores digitales de entrada y salida con comunicación I2C o SPI, 8 conectores de entrada analógica, memoria flash de 32 kb y velocidad de reloj de 16 MHz. La placa cuenta con un conector de salida de 3,3 V de tensión para alimentar el MPU-9250. También cabe destacar que Arduino es una placa muy popular entre aficionados y expertos, lo cual hace que sea fácil encontrar información para los que no están familiarizados con el mundo de los microcontroladores. El software de desarrollo de Arduino es una herramienta que facilita la tarea de programar el microcontrolador de forma rápida.

4.5.3 Diseño del circuito electrónico

Como se mencionó anteriormente, se utilizó la placa GY-9265 con el sensor MPU-9250 para adquirir la información del giroscopio, acelerómetro y magnetómetro. Como microcontrolador el ATmega328 integrado en la placa Arduino Nano 3.0. Este es el encargado de recibir los datos, calcular los ángulos de rotación y luego transmitirlos a la computadora personal. Para realizar la transmisión, se optó por utilizar el módulo Bluetooth HC-05, el cual utiliza Bluetooth V2.0. De esta manera, se evitó la utilización de un cable para la transmisión de los datos, permitió integrar el sensor con los auriculares y que el usuario pueda realizar movimientos libremente. La comunicación entre el microcontrolador y el HC-05 es serie asincrónica, por lo tanto, utiliza un conector para

transmitir datos y otro para recibir. En este caso, es importante que ambos estén configurados con la misma velocidad de transmisión (*Baud rate*).

En la Figura 28 se muestra el esquema del circuito en cuestión. Debido a que los valores lógicos que admite el HC-05 deben tener un valor máximo de 3,3 V, se aplicó un divisor resistivo con la finalidad de reducir el valor de tensión de 5 V de los datos transmitidos por el Arduino a 3,3 V (conector D3 con el conector RXD). El conector A4 del Arduino se conecta con el conector SDA del GY-9265 para la transferencia de datos por I2C. El conector A5 conectado al SCL, es para pulsos de reloj (*clock*).

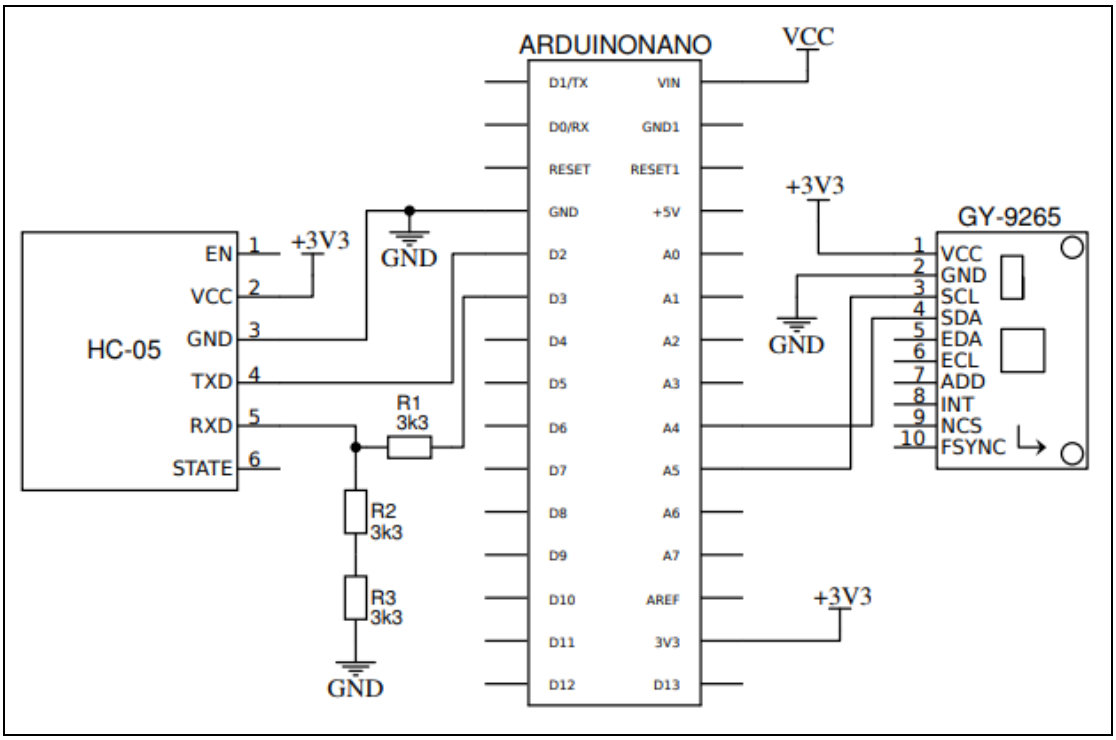

Figura 28. Circuito del sensor de movimiento.

Dentro del diseño de la fuente de alimentación del Arduino se utilizó una batería recargable de li-ion 18650 de 3,7 V con un módulo de carga y elevador de tensión a 5 V modelo T6845-C. En la Figura 29 se muestra el esquema de la fuente de alimentación. Para la alimentación del GY-9265 y HC-05 se conectó la salida regulada de 3,3 V del Arduino directamente.

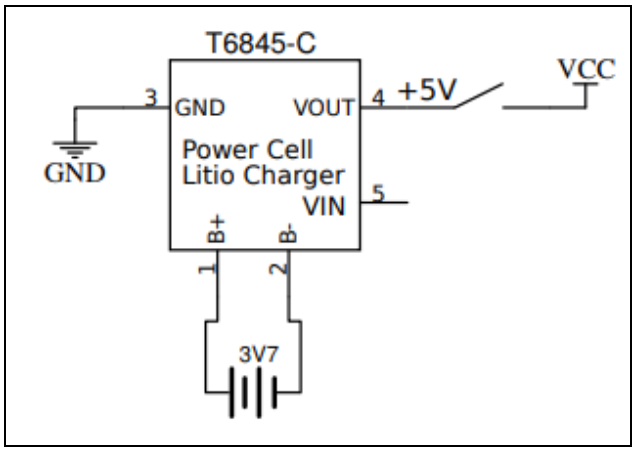

Figura 29. Circuito de fuente de alimentación de 5 V con batería recargable de li-ion 18650.

4.5.4 Programación del Microcontrolador

Con el motivo de programar del microcontrolador ATmega328 se empleó el software de desarrollo de Arduino. El lenguaje de programación de Arduino está basado en C++. El código se desarrolló en gran medida basándose en las implementaciones de Kris Winer [48] y SparkFun [49] para el sensor MPU-9250. Ambas de código libre.

El microcontrolador se programó con dos funciones principales. Una llamada "*setup*" en donde se inicializan las variables, se establecen los puertos de conexión a una velocidad de transmisión de 19200 Baudios, se realizan las pruebas y autocalibración de los sensores. Esta función se ejecuta solo una vez. La otra función, llamada "*loop*", es un bucle que se repite indefinidamente. Dentro de esta función se ejecutan las tareas para lectura de datos de los sensores, procesamiento con el filtro de Madgwick y envío de datos al módulo Bluetooth.

La calibración del magnetómetro es de suma importancia para el correcto funcionamiento del algoritmo. La misma se logró aplicando una función que requiere que el sensor se desplace en figura de 8 en el espacio a fin de obtener un muestreo representativo de los valores máximos y mínimos. Luego, se promediaron los valores y se obtuvieron los desvíos. De esta manera, se corrigieron los errores conocidos como *soft* y *hard iron* [50]. Este proceso se puede realizar una sola vez y luego simplemente guardar los datos de los desvíos en la memoria no volátil.

Para obtener la orientación del Norte magnético de la Tierra con el ángulo de *yaw* se debió realizar una corrección por la declinación magnética de la ubicación geográfica del

sensor. En este caso, la declinación magnética de la ciudad de Buenos Aires (-9°12’ ó -9,2° [51]) se le debe compensar al resultado del ángulo de *yaw*.

$$yaw = yaw + 9{,}2° \tag{28}$$

Las tareas de la programación del microcontrolador se resumen en la Tabla 3. En la Figura 30 se muestran las tareas principales dentro de un diagrama de flujo.

Tabla 3. Tareas de la programación del microcontrolador para la lectura y procesamiento de los datos del sensor MPU-9250.

Tareas	Función
• Inicialización de variables • Configuración de puerto de conexión con una velocidad de transmisión de 19200 Baudios. • Prueba y autocalibración de sensores.	Setup
• Lectura de los datos de los sensores. • Conversión a cuaterniones utilizando el filtro de Madgwick. • Transformación de cuaterniones a ángulos de rotación (*yaw*, *pitch* y *roll*) por medio de las ecuaciones (21-23). • Compensación del ángulo de rotación *yaw* debido a la declinación magnética de la ciudad de Buenos Aires. • Envió de datos al módulo Bluetooth HC-05.	Loop

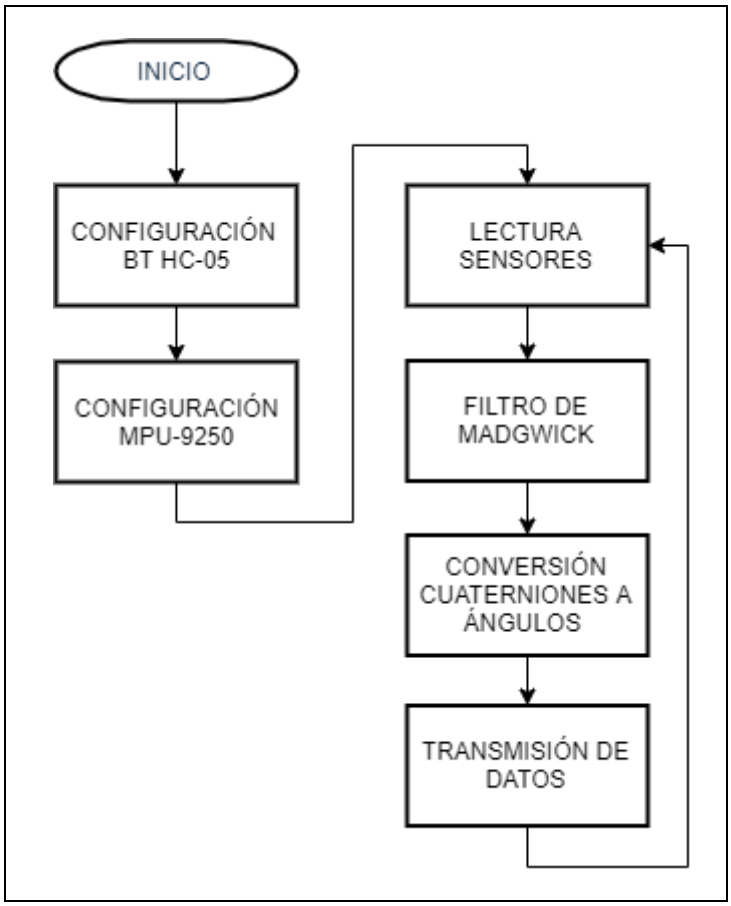

Figura 30. Diagrama de flujo del microcontrolador destinado a la lectura de los sensores y transmisión de datos al módulo Bluetooth HC-05.

4.6 BLOQUE DE PROCESAMIENTO CON SENSOR DE MOVIMIENTO

La última etapa del sistema fue integrar la lectura del sensor de movimiento al bloque de procesamiento en tiempo real dentro de Matlab.

La transmisión entre el módulo Bluetooth y la computadora se realizó emulando un puerto serie. Con la función “Bluetooth” de Matlab se creó un objeto al cual se le asocian los datos del dispositivo. Luego, con la función “fopen” se inició la transmisión de datos como un puerto serie convencional. Para la lectura de datos se utilizó la función “fscanf”.

Los códigos del sistema completo se encuentran cargados en un repositorio en línea de acceso libre [52]. En la Figura 42 del ANEXO C se muestra un diagrama que representa las tareas principales del programa diseñado en Matlab para realizar la síntesis binaural con el sensor de movimiento. En la Figura 43 del ANEXO D se muestra la interfaz gráfica con el sistema de síntesis binaural que integra el sensor de movimiento.

Primero, se cargan las respuestas al impulso de los filtros HRTF dentro de una matriz. Prosigue la carga de un archivo de audio monoaural. Cuando se inicia la

reproducción, comienzan a leerse bloques de señal de audio dentro de un bucle hasta que finaliza el archivo de audio. Dentro del bucle, se leen los datos del controlador de la posición de la fuente sonora. Luego, en caso de que el sensor este encendido, se leen los ángulos de rotación administrados por el mismo. Para esta aplicación solo se leen los ángulos de *yaw* y *pitch* con el objetivo de hacer las variaciones correspondientes en los ángulos azimutal y elevación. Los primeros valores obtenidos se ajustan como 0° tanto para los ángulos azimutal y 0° elevación. El siguiente paso es seleccionar el par de filtros correspondientes a la posición inicial de la fuente. Por ejemplo, si se establece la posición de la fuente en 0° azimutal y elevación, se buscan las respuestas al impulso correspondientes para ese ángulo. Después, se realiza la convolución por el método de *Overlapp-Save* entre el bloque de señal y el par de filtros. En caso de que haya variación de filtros se utiliza el fundido cruzado temporal. Finalmente, se generan los datos de salida y se envían al controlador de audio.

Para el caso de que haya rotación del sensor de movimiento o desplazamiento de la fuente por medio de un controlador, se debe ajustar la posición de la fuente para la selección de los filtros. En la Figura 31 se ejemplifican los casos donde se desplaza la fuente. En (a) se muestra un desplazamiento debido a la rotación del sensor de movimiento partiendo de una posición inicial de fuente en 0°. Por ejemplo, si se registra una rotación del sensor en un ángulo $\theta°$ en sentido horario, se debe desplazar a la fuente el mismo ángulo $\theta°$ pero en sentido antihorario. De esta manera, la fuente se percibe en la misma posición espacial a pesar de que el oyente rote su posición. En (b) se da el caso en que se desea desplazar la fuente por medio de un controlador. Por ejemplo, si se desea desplazar la fuente en un $\theta°$ en sentido antihorario, simplemente se le suma ese ángulo a la posición inicial. El mismo procedimiento aplica para desplazamientos en el ángulo de elevación.

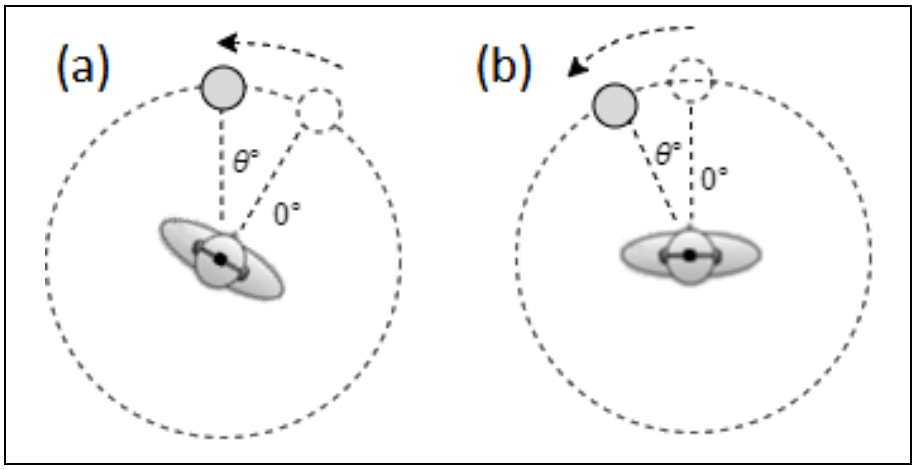

Figura 31. Esquemas con los casos en que se varía la posición de la fuente partiendo de una posición inicial en 0°: (a) Desplazamiento de fuente en un ángulo $\theta°$ por rotación del sensor de movimiento; (b) Desplazamiento de fuente en un ángulo $\theta°$ por medio de un controlador.

Lo anteriormente mencionado se expresa en la ecuación:

$$\theta' = \theta_{controlador} - (\theta_{yaw} - \theta_{cero}) \tag{29}$$

donde θ_{cero} es el ángulo de *yaw* medido por el sensor como valor inicial para posicionar el sistema de referencia; θ_{yaw} es el ángulo de *yaw* medido por el sensor; $\theta_{controlador}$ es el ángulo desplazado por un controlador en el plano horizontal; θ' es el ángulo azimut resultante con la posición final de la fuente.

La misma ecuación se aplica en forma análoga para los ángulos de elevación. En este caso, utilizando el ángulo de *pitch* registrado por el sensor.

$$\varphi' = \varphi_{controlador} - (\varphi_{pitch} - \varphi_{cero}) \tag{30}$$

donde φ_{cero} es el ángulo de *pitch* medido por el sensor como valor inicial para posicionar el sistema de referencia; φ_{pitch} es el ángulo de *pitch* medido por el sensor; $\varphi_{controlador}$ es el ángulo desplazado por un controlador en el plano medio; φ' es el ángulo de elevación resultante con la posición final de la fuente.

CAPÍTULO V – EVALUACIÓN DEL SISTEMA

5.1 VALIDACIÓN DEL SENSOR DE MOVIMIENTO

Una vez ensamblado el sensor de movimiento se procedió a validar la medición de los ángulos en el plano horizontal. Para ello se tomó como referencia una mesa giratoria Outline ET230-3D. Según los datos del fabricante puede medir ángulos con una precisión de 0,5 ± 0,01 °.

Inicialmente, se intentó medir con el sensor situado directamente sobre la mesa (Figura 32). Sin embargo, se observó que las mediciones eran influidas por los campos magnéticos residuales de la mesa giratoria. Este mismo problema lo tuvo el Ing. Piegari al validar el sensor de movimiento en su tesis de grado [53]. Al igual que en su investigación, se aumentó la distancia entre ambos sistemas. En este caso, se apoyó una caja de 1 m de altura sobre la mesa, luego, se colocó el sensor sobre la cabeza y torso de un maniquí de la marca G.R.A.S. KEMAR a fin de generar una mayor distancia (Figura 33).

Figura 32. Sensor de movimiento directamente apoyado sobre la mesa giratoria.

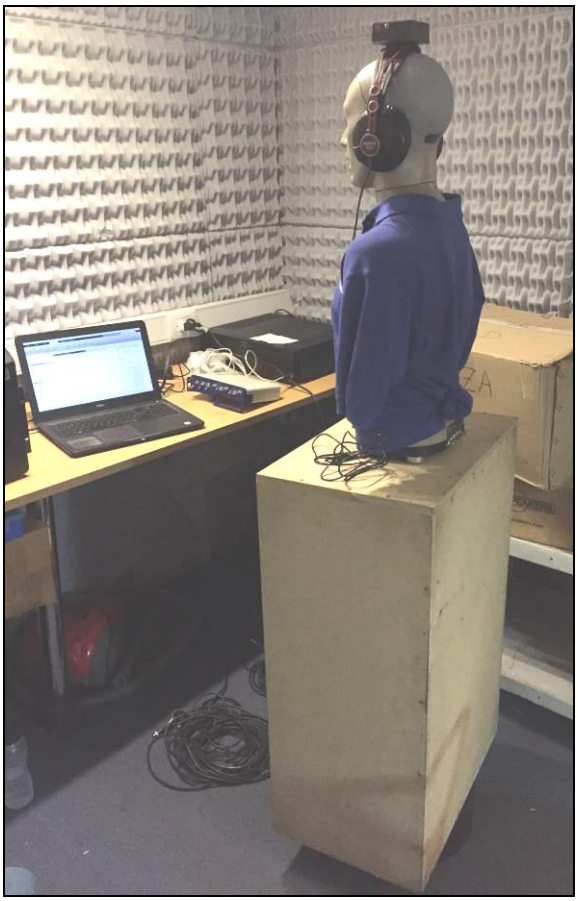

Figura 33. Sistema de medición de ángulos con el sensor de movimiento posicionado sobre la cabeza y torso de un maniquí G.R.A.S. KEMAR y una caja de 1 m de altura. La rotación de ángulos se hizo por medio de la mesa giratoria (debajo de la caja).

La medición se realizó en pasos de 45 ° en sentido antihorario. Se tomaron 300 muestras por cada paso. Luego, se calculó el valor medio, el error absoluto y la incertidumbre de la medición. En la Tabla 4 se observan los resultados de la medición.

Tabla 4. Resultados de la medición angular del sensor de movimiento utilizando la mesa giratoria.

Ángulos de medición [°]	Valor Medio [°]	Error Absoluto [°]
0	0 ± 1,8	0
45	43,3 ± 2,3	1,7
90	87,4 ± 1,3	2,6
135	140 ± 0,7	5
180	190,6 ± 0,6	10,6
225	231,7 ± 0,5	6,7
270	271 ± 0,9	1
315	315,2 ± 1,2	0,2

Se puede observar en los resultados que para los valores de los ángulos en el plano frontal (90, 45, 0, 315 y 270 °) el error absoluto es relativamente bajo, teniendo en cuenta la finalidad del dispositivo. Para los ángulos 135, 180 y 225°, los errores absolutos fueron más grandes en comparación. Se estima que se debe a una calibración del magnetómetro no del todo precisa para compensar los fenómenos de *soft* y *hard iron*.

5.2 EVALUACIÓN DE LATENCIA DEL SISTEMA

Se realizó una evaluación de la latencia del sistema con el fin de analizar los procesos que producen los mayores retardos temporales. En esta prueba se empleó la versión de Matlab R2018a con una computadora Dell i5-7200u, CPU 2.5 GHz, 8 GB de RAM y Windows 10. Los filtros se obtuvieron de la base de datos pública de la cabeza maniquí Neumann KU100 publicada por la Universidad Técnica de Colonia (*Technische Hochschule Köln*) [54-55]. Los filtros tienen una longitud de 128 muestras con una frecuencia de muestreo de 48 kHz.

Los bloques de procesamiento bajo análisis están representados en la Figura 42 del ANEXO C dentro del bucle de procesamiento. Los retardos temporales fueron medidos con la función “Tic-Toc” de Matlab. En la Tabla 5 se muestran los resultados obtenidos de la medición.

Tabla 5. Resultados de la medición de latencia en cada etapa del sistema.

Proceso	Valor Medio [ms]	Muestras Utilizadas
Lectura sensor BT	26,8 ± 20,1	3461
Selección de filtros	0,32 ± 0,24	1398
Convolución	0,31 ± 0,26	1398
Controlador de audio	19,8 ± 10,9	1398
Actualización de variables	0,66 ± 0,25	1398
Total sin sensor	21,09 ± 11,65	
Total con sensor	47.89 ± 31,75	

Los resultados muestran que la lectura del sensor es el proceso con mayor retardo, aproximadamente 56 % del valor total del sistema, el controlador de audio presenta aproximadamente 41,3 %, y el resto de los procesos influyen aproximadamente 2,7 % del retardo total.

5.3 PRUEBA SUBJETIVA DEL SISTEMA

Se diseñaron tres pruebas subjetivas para evaluar el sistema. La primera prueba se planteó como una prueba de localización de fuente sonora en el plano horizontal. La finalidad fue determinar si los sujetos reconocían la dirección de una fuente virtual por medio de la síntesis binaural dinámica con los filtros HRTF. La segunda, se trató de una prueba ABX en donde se analizaron diferentes tiempos de latencia con el sensor de movimiento. Y la tercera, constó de una evaluación de pérdida de calidad entre el audio original y el audio procesado por un par de filtros a 0°. La duración de las tres pruebas no superó los 15 minutos.

En las tres pruebas se empleó la versión de Matlab R2018a con una computadora Dell i5-7200u, CPU 2.5 GHz, 8 GB de RAM y Windows 10. Los filtros se obtuvieron de la base de datos pública de la cabeza maniquí Neumann KU100 publicada por la Universidad Técnica de Colonia. La reproducción se realizó con los auriculares Focusrite Scarlett Hp60.

Las pruebas fueron realizadas con 27 personas, 9 mujeres y 18 hombres, con promedio de edad de 32,9 años y desvío estándar de 8,3 años. Todos manifestaron ser normo oyentes.

5.3.1 Prueba 1: localización

Las pruebas de localización de fuente sonora son comunes dentro de la literatura de síntesis binaural [15]. Sin embargo, se carece de una metodología estandarizada. En general, se presenta un ruido blanco en algún ángulo de azimut (o elevación), y luego, el sujeto de prueba tiene que reconocer la dirección de donde proviene. Para pruebas donde se permite la síntesis dinámica se suelen utilizar estímulos de duración mayor a 1 s.

En esta prueba, se decidió analizar la localización solo en el plano horizontal a fin simplificar el análisis. Como señal de estímulo se eligió ruido blanco con duración de 10 s. Esto permitió que el sujeto pueda buscar el sonido utilizando la rotación de la cabeza. El sujeto tenía como objetivo localizar la fuente de frente y quedarse quieto hasta finalizar los 10 s de reproducción. Los ángulos azimut se eligieron desde 0 a 300 en pasos de 60° (Figura 34). El orden de aparición de la fuente se dispuso en forma aleatoria en cada prueba. Se desarrolló una interfaz gráfica que contenía las instrucciones, un botón capaz de comprobar el funcionamiento del sensor de movimiento, un botón encargado de reproducir la señal de prueba y otro para iniciar la prueba subjetiva en sí. En la Figura 44

del ANEXO D se muestra la interfaz gráfica utilizada. Permitir la escucha de la señal de prueba fue importante para dar a entender al sujeto qué se iba a evaluar. También, se utilizaron las dos primeras pruebas de localización como entrenamiento, repitiendo los ángulos 60 y 240°. Esto generó un total de ocho pruebas de localización.

Con el fin de determinar si el sujeto encontró la fuente, se integraron los resultados del sensor dentro del último segundo almacenado, y finalmente, se promedió el valor acumulado.

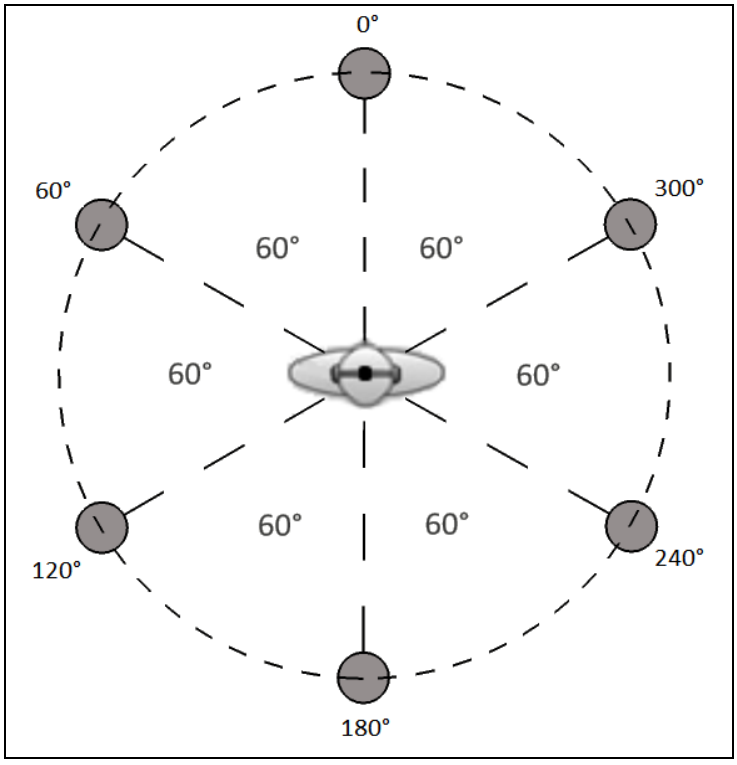

Figura 34. Esquema con las posibles posiciones de la fuente virtual en la prueba de localización.

El diagrama de flujo de la prueba subjetiva de localización de fuente sonora se muestra en la Figura 47 del ANEXO E.

5.3.2 Resultados prueba 1

El objetivo de esta prueba era probar estadísticamente que los sujetos pueden localizar un ruido blanco para diferentes ángulos azimut utilizando la síntesis binaural dinámica.

Los resultados de la prueba de localización se expresan en diagrama de cajas en la Figura 35. Las líneas horizontales rojas representan los valores promedios de los ángulos localizados por todos los participantes, las cajas enmarcan los percentiles 25 y 75. Los

bigotes se extienden hasta ± 2,7 σ (desvío estándar) desde la media. Los valores más allá de los bigotes se consideran *outliers* y se representan con el símbolo "+".

A pesar de que hay *outliers* para todos los ángulos de prueba, la tendencia de los valores promedios revela que los ángulos fueron localizados por la mayoría. Se detectaron 3 valores *outliers* considerados efectos de la confusión *front-back* (símbolos "+" marcados con circunferencias azules). Estas confusiones son aproximadamente el 1,9 % de las respuestas.

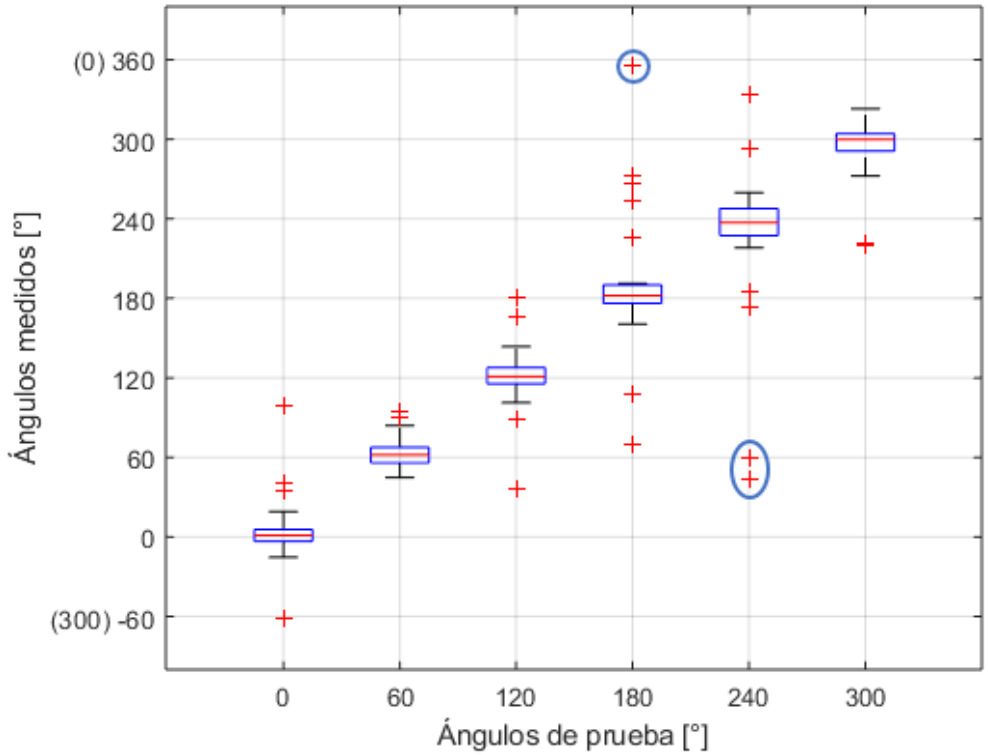

Figura 35. Diagrama de cajas con los resultados de la prueba de localización de fuente sonora.

Los promedios de errores absolutos para cada ángulo de prueba se presentan en la Figura 36. Previamente se despreciaron los resultados de confusión *front-back* por no ser representativos de la tendencia. El ángulo que presentó el mayor error fue el de 180°, el sujeto debía realizar una mayor rotación angular para poder localizarlo, se cree que la dificultad estuvo en este aspecto. No obstante, el error absoluto global fue 13,34°.

Rothbucher en su investigación obtuvo un error absoluto de localización de 16,14° y 4,2 % de confusión *front-back* en el plano horizontal con filtros no individuales [56]. En comparación, los resultados obtenidos en esta investigación son menores. Esto aporta certeza en los valores encontrados.

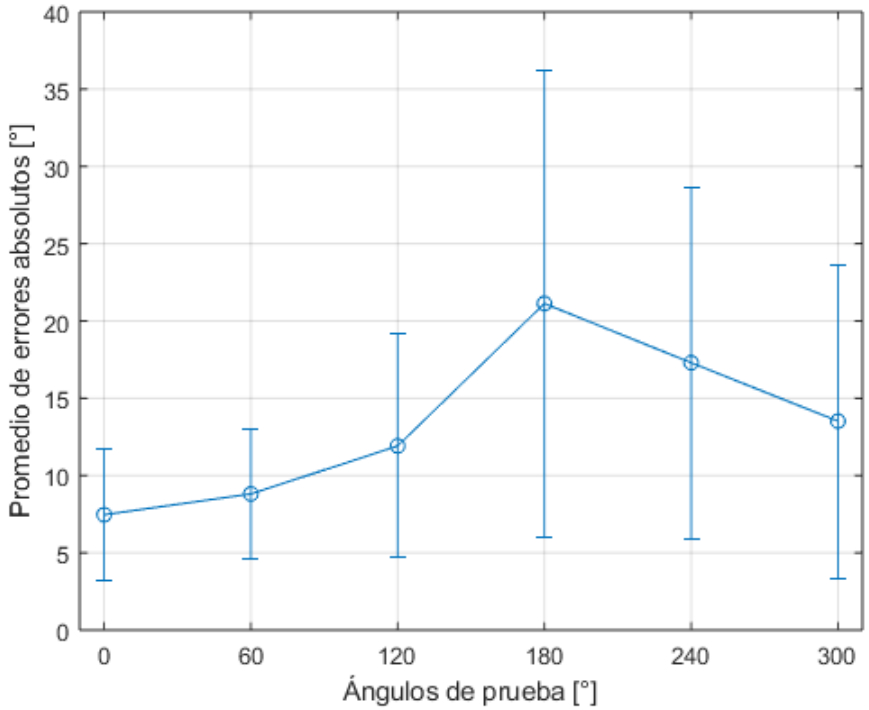

Figura 36. Promedio de errores absolutos en la prueba de localización.

Otra forma de expresar los resultados es ubicar los valores promedios y trazar una regresión lineal (Figura 37). En el caso ideal de que los sujetos hubiesen encontrado los ángulos sin error, el resultado generaría una recta con pendiente igual a 1 y ordenada al origen igual a 0.

Como puede apreciarse en la Figura 37, las respuestas se encuentran cercanas a los ángulos de prueba, dado que la pendiente de la recta de regresión lineal adquiere un valor cercano a 1. El coeficiente de determinación (R^2) dio un valor de 0,9996 ($p < 0,01$), por consiguiente, el modelo de regresión lineal se ajusta a los resultados obtenidos y se rechaza la hipótesis nula de que los sujetos no localizan la fuente sonora para los ángulos de prueba.

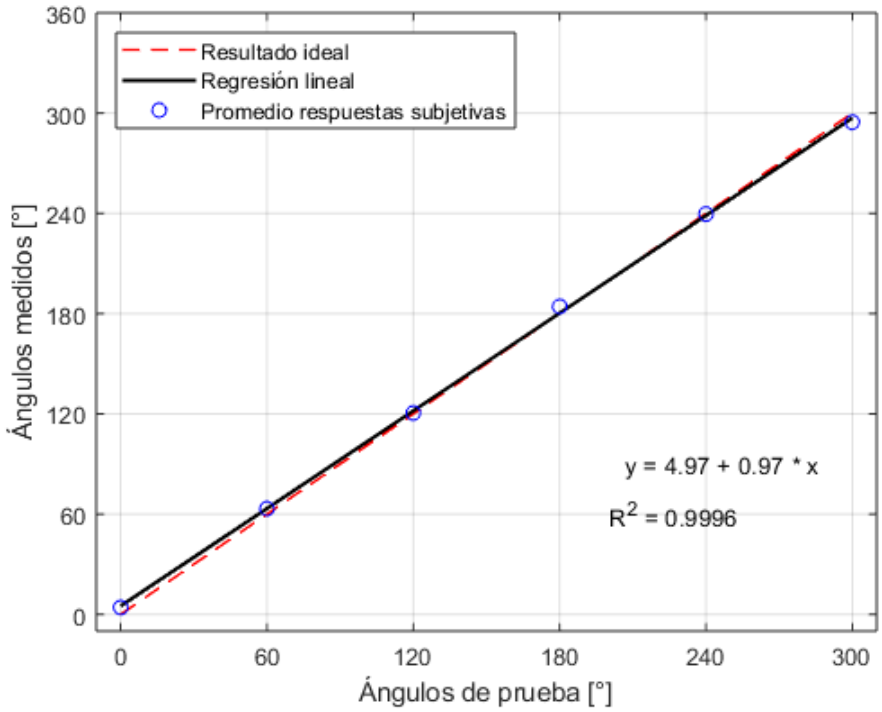

Figura 37. Resultados promedios y regresión lineal de la prueba de localización.

5.3.3 Prueba 2: latencia del sensor de movimiento

La lectura del sensor de movimiento resultó ser la tarea que mayor latencia aporta al sistema. Una forma de optimizar el sistema es realizar menor cantidad de lecturas a lo largo del bucle de reproducción. Sin embargo, esto conlleva a aumentar el retardo en el desplazamiento de la fuente sonora. La interrogante que surgió fue: ¿qué umbral de latencia del sensor de movimiento es perceptible? Para tal fin, se decidió realizar una prueba subjetiva del tipo ABX con empate.

Se utilizaron dos estímulos en esta prueba, 6 s de un pasaje musical lento de orquesta y 5 s de una señal de audio de batería (Figura 38). En cada prueba AB se presentó la misma señal de audio, pero con dos condiciones diferentes de latencia en el sensor de movimiento. Una es la condición con latencia agregada y la otra la referencia sin latencia agregada. El estímulo X es la condición A o B. Al sujeto se le pidió girar la cabeza durante la reproducción, y luego, reconocer si el estímulo X era la condición A, B o sin diferencia. Los valores de latencia agregada fueron 42, 84, 126 y 168 ms (valores múltiplos de la latencia del bucle de reproducción). Todos los grupos de estímulos fueron presentados en forma aleatoria con el fin de disminuir los desvíos producidos. En la Figura 45 del ANEXO D se

muestra la interfaz gráfica utilizada para esta prueba subjetiva, y en la Figura 48 del ANEXO E se puede ver el diagrama de flujo.

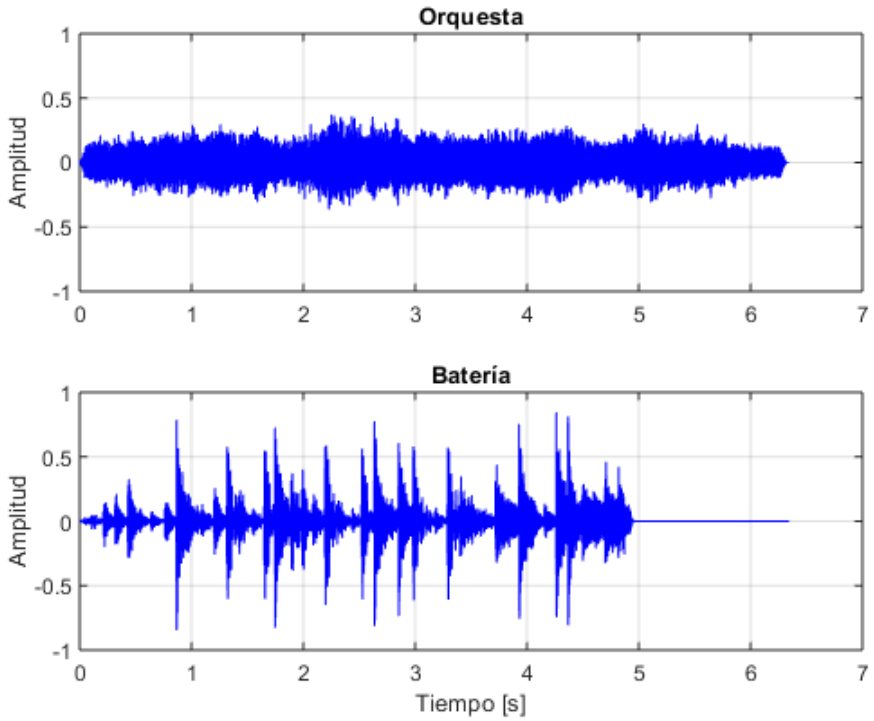

Figura 38. Forma de onda de las señales de audio utilizadas en la prueba de latencia del sensor de movimiento.

5.3.4 Resultados prueba 2

El umbral de respuestas correctas necesarias en una prueba ABX se obtiene a partir de una distribución binomial. Se buscó que la suma de las probabilidades de respuestas incorrectas o azarosas sea menor al 5 % (Ecuación (31)).

$$\sum_{k=0}^{k=n}\left(\frac{{}^{N!}/_{k!\cdot(N-k)!}}{2^N}\right) < 0{,}05 \qquad (31)$$

Siendo *N* el número de la población analizada, *k* la variable de respuestas incorrectas o azarosas, y *n* el valor máximo para satisfacer la condición.

Como en este caso se realizó la prueba con 27 personas (*N* = 27), la cantidad de respuestas incorrectas o azarosas que se ajusta a la condición es igual a 8 (*n* = 8). Entonces, el umbral de respuestas correctas se obtiene como:

$$respuestas\ correctas\ necesarias = \left(1 - \frac{n}{N}\right) \cdot 100 = 70{,}37\ \% \qquad (32)$$

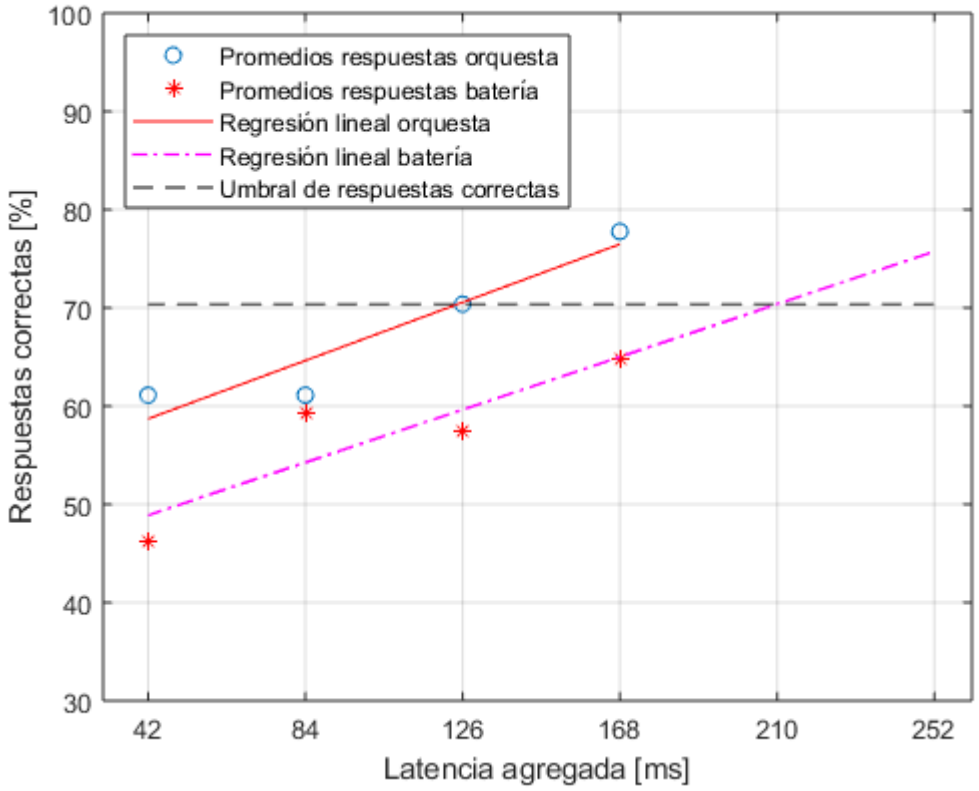

Figura 39. Resultados de la prueba ABX para detectar el umbral de latencia del sensor de movimiento.

Los resultados de la prueba ABX de la detección del umbral de latencia del sensor de movimiento se muestran en la Figura 39. Con las respuestas de cada estímulo se trazó una regresión lineal. Para el audio de orquesta, el coeficiente de determinación fue igual a 0,8982 (p = 0,0522), y con la batería, fue igual a 0,7972 (p = 0,1072).

El valor de latencia detectado con un 95 % de confianza se encuentra en la intersección entre la regresión lineal de los resultados y el umbral de respuestas correctas (en este caso 70,37 %). Cabe aclarar que, en el caso del audio de batería, las repuestas correctas no alcanzaron el umbral de 70,37 % dentro de los intervalos de latencia agregados. Se tuvo que extender la regresión lineal a modo de predicción con el fin de poder alcanzar el umbral. Los resultados finales se expresan en la Tabla 6.

Tabla 6. Resultados estadísticos de latencia del sensor de movimiento detectados.

	Orquesta	Batería
Latencia [ms]	124,7	210

Se puede apreciar que el valor de latencia para la batería es mayor que para la orquesta (un 68,4 % mayor). Varios sujetos de prueba expresaron que les resultó más difícil diferenciar las latencias agregadas para el audio de batería. Esto se debe a la característica impulsiva de la señal, en comparación con el audio de orquesta que es continuo en el tiempo. Los saltos de ángulos con latencia agregada se percibían más fácil con un sonido continuo.

En la investigación de Stitt et al. realizaron una prueba similar para detectar el umbral de latencia del sensor de movimiento en la síntesis binaural dinámica [57]. Con la reproducción de una señal de audio compleja (voces e instrumentos musicales) llegaron al resultado de aproximadamente 60 ms como valor de latencia detectado. Un resultado inferior que el obtenido en esta investigación que denota una mayor sensibilidad a la latencia agregada.

5.3.4.1 Movimiento de la cabeza

Adicionalmente, en esta prueba se tomaron registros de la velocidad angular media en cada respuesta. El objetivo fue ver si la velocidad angular media era una variable que afecta en las respuestas subjetivas. Se procedió a realizar un análisis ANOVA con el fin de probar si existe diferencia significativa entre las respuestas de los sujetos para cada valor de latencia agregada y la velocidad angular media del movimiento de la cabeza.

Tabla 7. Resultados del análisis de ANOVA entre las respuestas de los sujetos y la velocidad angular media del movimiento de la cabeza.

	***P* valor**	
Latencia [ms]	Orquesta	Batería
42	0,9671	0,6354
84	0,8663	0,9893
126	0,1349	0,1364
168	0,0158	0,6298

Los resultados del análisis de ANOVA se muestran en la Tabla 7. Los resultados de *p* valor mayores a 0,05 indican que no hay diferencia significativa entre la velocidad angular media y las respuestas de los sujetos, como se puede apreciar en la mayoría de los casos. Por otro lado, el caso para la latencia agregada de 168 ms y el audio de orquesta se encontró una diferencia significativa. Las respuestas incorrectas, empate y correctas

mostraron una velocidad angular media de 55,1, 95,7 y 83,1°, respectivamente. Es el único caso que se puede justificar estadísticamente que el movimiento lento de la cabeza influenció a los sujetos a elegir la respuesta incorrecta.

El siguiente análisis fue ver si existe diferencia significativa de la velocidad angular media entre ambos audios de prueba. Para ello se realizó la prueba de t de student. En la Tabla 8 se representan los valores de la velocidad angular media y los valores de significación resultantes de la prueba de t de student. En los cuatro casos los *p* valor son mayores a 0,05, por lo tanto, no hay evidencia suficiente para afirmar que existe una diferencia significativa de la velocidad angular media entre ambos audios de prueba.

Tabla 8. Velocidad angular media del movimiento de la cabeza y valor de significación de la prueba de t de student.

	Velocidad angular media [°/s]		
Latencia [ms]	Orquesta	Batería	*P* valor
42	91,1 ± 25,5	97,1 ± 28,6	0,1129
84	90,8 ± 26,9	91,4 ± 25,5	0,8913
126	83,8 ± 28,2	85 ± 21,9	0,7402
168	82,8 ± 20,8	88,2 ± 27,3	0,1957

En la investigación de Stitt et al. [57] también realizaron un análisis de la velocidad angular media y las respuestas de los sujetos. Ellos no encontraron datos suficientes para afirmar que las respuestas eran afectadas por la velocidad angular media. Resultados similares a los de esta investigación, excluyendo el caso particular del audio de orquesta y latencia agregada de 168 ms.

5.3.5 Prueba 3: calidad

La última prueba que se realizó fue de percepción de pérdida de calidad debido a los filtros HRTF. Se aplicó la escala de percepción de deficiencia de la ITU-R BS.562-3 [58] para evaluar diferentes estímulos monoaurales contra los mismos estímulos procesados con los filtros HRTF a 0° azimut (Figura 40). Se les indicó a los sujetos que prestaran atención a si percibían pérdida de calidad en el audio procesado, y que lo marcaran en uno de los niveles como: imperceptible; perceptible, pero no molesta; ligeramente molesta; molesto; muy molesto. Para el análisis esta escala se puede asociar a valores numéricos entre 0 y 1, en pasos de 0,25. En la Figura 46 del ANEXO D se muestra la interfaz gráfica utilizada en

esta prueba subjetiva. Las señales de prueba que se eligieron se muestran en la Tabla 9. En cada prueba se presentaron los audios en orden aleatorio.

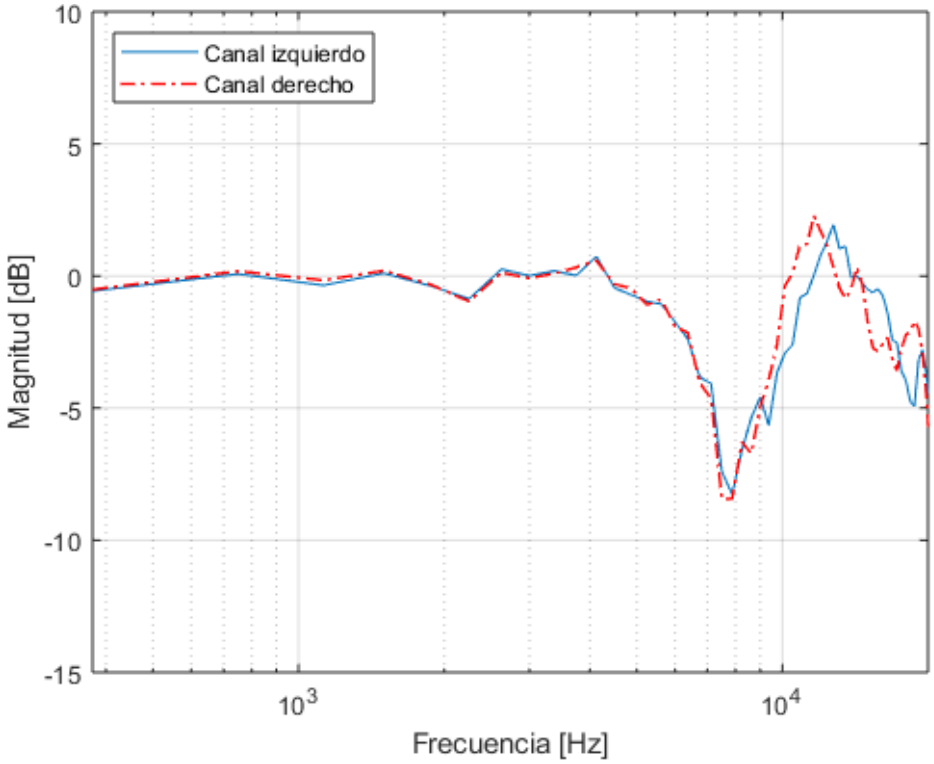

Figura 40. Espectro de los filtros HRTF a 0° azimut.

Tabla 9. Señales de audio para la prueba de calidad con la duración.

Señal de audio	Batería	Guitarra	Música electrónica	Música pop	Música clásica	Saxofón	Violín	Voces hombre y mujer
Duración [s]	4	7	7	8	12	9	12	5

5.3.6 Resultados prueba 3

Los resultados de esta prueba se muestran en diagrama de cajas en la Figura 41. Prácticamente todos los valores medios de las respuestas se encuentran en el nivel de perceptible, pero no molesta. El análisis de ANOVA arrojó un valor de significación de 0,1565, por lo tanto, no hay evidencia suficiente para afirmar que las medias de los grupos presentan diferencias significativas.

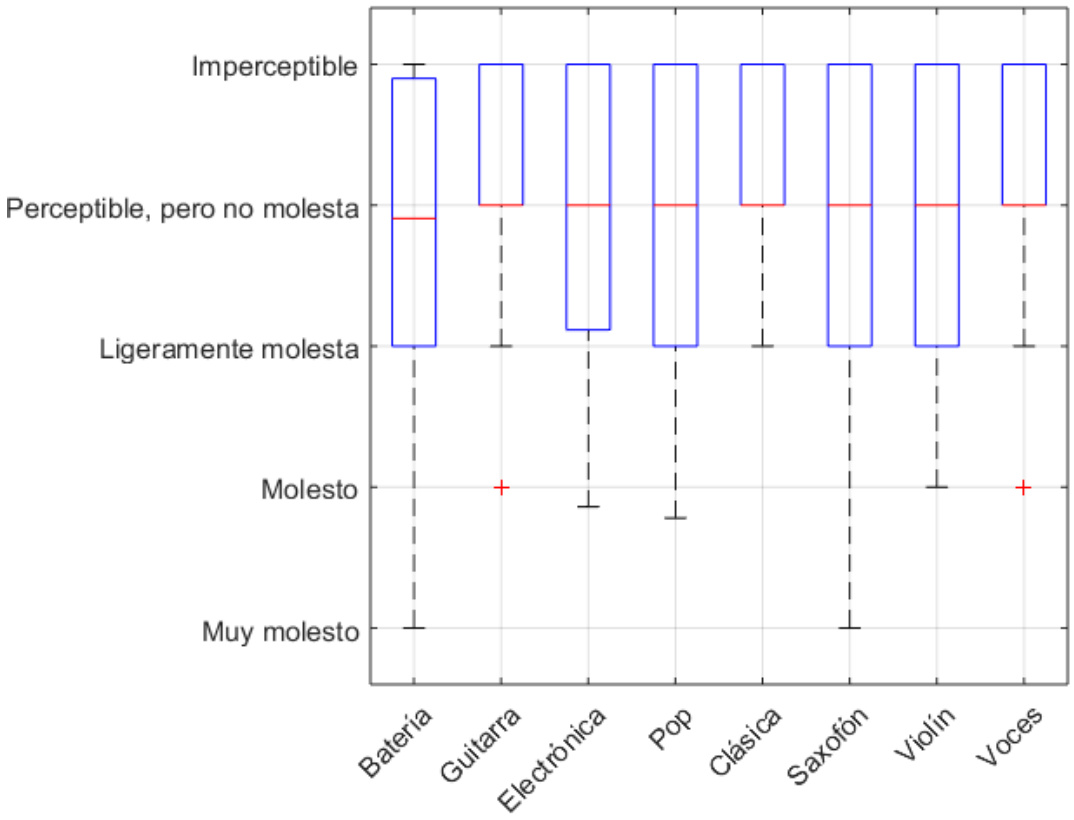

Figura 41. Diagrama de cajas con las respuestas de la prueba de percepción de pérdida de calidad.

CAPÍTULO VI - CONCLUSIONES

La principal conclusión que se remarca es que fue posible lograr la implementación de síntesis binaural aplicando la convolución en tiempo real, los filtros HRTF de una base de datos pública y un sensor de movimiento de la cabeza en el entorno de Matlab. De esta forma, se desarrolló una herramienta que facilita la espacialización de fuentes sonoras mediante síntesis binaural para favorecer investigaciones científicas con relación a la percepción subjetiva. Esto se evidencia con las pruebas subjetivas realizadas en la sección 5.3. Aunque se crearon interfaces diferentes en cada prueba, el núcleo del sistema seguía siendo la convolución en tiempo real con los filtros HRTF.

El sensor de movimiento desarrollado cumplió con los requerimientos necesarios para poder realizar la síntesis binaural dinámica. La latencia generada por la lectura de los datos no impidió el funcionamiento de la síntesis en tiempo real. La calibración del magnetómetro del sensor MPU-9250 es fundamental para reducir los desvíos producidos por los errores de *soft* y *hard iron*.

Los resultados estadísticos obtenidos en la prueba de localización de fuente demuestran que los sujetos de prueba pudieron detectar los ángulos azimut evaluados. Por lo tanto, se puede inferir que la espacialización por medio de la síntesis binaural dinámica fue percibida correctamente.

Las respuestas de la prueba de percepción de latencia del sensor, mostraron que los sujetos eran más sensibles a la latencia agregada con la señal de audio continuo de orquesta en lugar de la señal de audio de batería. Solo en la condición de latencia agregada de 168 ms se encontró evidencia de que la velocidad angular media de rotación de la cabeza fue un factor que influyó en las respuestas. En el resto de los casos, no se puede afirmar que la velocidad angular media es un factor influyente.

No se encontró diferencia significativa de percepción de pérdida de calidad entre las diferentes señales de audio utilizadas en la prueba 3. Se puede concluir que la pérdida de calidad producida por los filtros HRTF de la base de datos de la cabeza maniquí Neumann KU100 es perceptible, pero no molesta.

CAPÍTULO VII – LÍNEAS FUTURAS DE INVESTIGACIÓN

Dentro de las líneas futuras se podrían realizar procesamientos e implementaciones adicionales con el fin de mejorar el sistema. Un caso, sería extrapolar el algoritmo desarrollado en esta investigación para realizar la síntesis con múltiples fuentes. Asimismo, se podrían integrar algoritmos de acústica de salas capaces de simular las reflexiones tempranas por el método de fuente imagen o de rayos. En las reflexiones tardías se suelen utilizar reverberaciones artificiales. Por otro lado, se podría adaptar el algoritmo con la convolución particionada para interactuar con filtros BRIR. Similares a los HRTF, pero con reflexiones de salas integradas en las respuestas al impulso.

Otro tema de investigación sería analizar con profundidad los algoritmos de fundido cruzado en el dominio espectral con el objetivo de lograr una mayor eficiencia.

Siguiendo el análisis de percepción subjetiva, se podría continuar realizando pruebas con los filtros HRTF relacionadas a la inteligibilidad o a sistemas de comunicaciones. También, se podrían realizar pruebas de percepción de salas y/o distancia de fuentes sonoras. Por otro lado, se podría evaluar diferentes respuestas subjetivas frente a fuentes sonoras con ángulos de elevación. O incluso, sería posible analizar los efectos de la velocidad angular media de rotación de la cabeza sobre las respuestas de percepción subjetivas utilizando diferentes estímulos.

Una importante línea de investigación sería encontrar la manera de poder aproximar los filtros con parámetros personales, tales como medidas antropométricas del pabellón auditivo y la cabeza. Asimismo, se podrían implementar algoritmos de interpolación temporal y espectral para aumentar la resolución de la síntesis binaural.

BIBLIOGRAFÍA

[1] HRTF Measurements of a KEMAR Dummy-Head Microphone, Extraído el 24 de Junio de 2019, http://sound.media.mit.edu/resources/KEMAR.html.

[2] Majdak, P. y Noisternig, M., AES69-2015: AES standard for file exchange - Spatial acoustic data file format, Audio Engineering Society, (2015).

[3] Wenzel, E. M., Miller, D. J. y Abel, J. S. , Sound Lab: a real-time, software-based system for the Study of Spatial hearing, Audio Engineering Society Convention 108, Audio Engineering Society, Paris, Francia, (2000).

[4] Miller, J. D. y Wenzel, E. M., Recent developments in SLAB: A software-based system for interactive, The 2002 International Conferece on Auditory Display, Kyoto, Japan, (2002).

[5] Begault, D. R., Godfroy, M., Miller, J. D., Wenzel, E. M. y Anderson, M. R., Applying spatial audio to human interfaces: 25 years of NASA experience, Audio Engineering Society Conference: 40th International Conference: Spatial Audio: Sense the Sound of Space, Audio Engineering Society, Tokyo, Japan, (2010).

[6] Saviojia, L., Huopaniemi, J., Lokki, T. y Väänänen, R., Creating interactive virtual acoustic environments, J. Audio. Eng. Soc., vol. 47, nº 9, pp. 675-705, (1999).

[7] Saviojia, L., Lokki, T. y Huopaniemi, J., Auralization applying the parametric room acoustic modeling technique - The DIVA Auralization system, The 2002 International Conferece on Auditory Display, Kyoto, Japan, (2002).

[8] Blauert, J., Lehnert, H., Sahrhage, J. y Strauss, H., An interactive virtual-environment generator for psychoacoustic research I: architecture and implementation, Acustica united with Acta Acustica, vol. 86, nº 1, pp. 94-102, (2000).

[9] Silzle, A., Novo, P. y Strauss, H., IKA-SIM: A System togenerate auditory virtual environmets, Audio Engineering Society Convention 116, Audio Engineering Society, Berlín, Alemania, (2004).

[10] Borß, C., Silzle, A. y Martin, R., Internet-based interactive auditory virtual environment generators, The 14th International Conference on Auditory Display, París, Francia, (2008).

[11] Geier, M., Ahrens, J. y Spors, S., The SoundScape Renderer: A Unified Spatial Audio Reproduction Framework for Arbitraty Rendering Methods, Audio Engineering Society Convention 124, Audio Engineering Society, Amsterdam, Holanda, (2008).

[12] Zhang, C. Y. y Xie, B. S., Platform for virtual auditory environment real time rendering system, J. Acoust. Soc. Am., vol. 131, nº 4, p. 3269, (2012).

[13] Xie, B. S. y Zhang, C. Y., An algorithm for efficiently synthesizing multiple near-field virtual sources in dynamic virtual auditory display, Audio Engineering Society Convention 132, Audio Engineering Society, Budapest, Hungría, (2012).

[14] Blauert, J., The technology of binaural listening, Springer Science & Business Media, (2013).

[15] Xie, B., Head-related transfer funtion and virtual auditory display, J. Ross Publishing, (2013).

[16] Woodworth, R. S. y Schlosberg, H., Experimental Phychology, Oxford and IBH Publishing, (1954).

[17] Blauert, J., Spatial Hearing: the psychophysics of human sound localization, MIT Press, Cambridge, MA, (1997).

[18] Rayleigh, L., On Our Perception of Sound Direction, Philosoph. Mag., vol. 13, (1907).

[19] Wallach, H., The role of head movements and vestibular and visual cues in sound localization, Journal of Experimental Psychology, vol. 27, nº 4, p. 339, (1940).

[20] Wightman, F. L. y Kistler, D. J., Resolution of front–back ambiguity in spatial hearing by listener and source movement, J. Acoust. Soc. Am., vol. 105, nº 5, pp. 2841-2853, (1999).

[21] Perrett, S. y Noble, W., The effect of head rotations on vertical plane sound localization, J. Acoust. Soc. Am., vol. 102, nº 4, pp. 2325-2332, (1997).

[22] Rao, D. y Xie, B., Head rotation and sound image localization in the median plane, Chinese Science Bulletin, vol. 50, nº 5, pp. 412-416, (2005).

[23] Wightman, F. L. y Kistler, D. J., Monaural sound localization revisited, J. Acoust. Soc. Am., vol. 101, nº 2, pp. 1050-1063, (1997).

[24] Gardner, M. B. y Gardner, R. S., Problem of localization in the median plane: Effect of pinna cavity occlusion, J. Acoust. Soc. Am. 53, pp. 400-408, (1973).

[25] Middlebrooks, J. C. y Green, D. M., Sound localizaion by human listeners, Annu. Rev. Psychol. 42, pp. 135-159, (1991).

[26] Carlile, S., Virtual Auditory Space: Generation and Applications, Springer, Berlin, Heiderberg, (1996).

[27] Zahorik, P., Assessing auditory distance perception using virtual acoustics, J. Acoust. Soc. Am. , vol. 111, nº 4, pp. 1832-1846, (2002).

[28] Zahorik, P., Auditory display of sound source distance, The 2002 International Conferece on Auditory Display, Kyoto, Japan, (2002).

[29] Zahorik, P., Brungart, D. S. y Bronkhorst, A. W., Auditory distance perception in humans: A summary of past and present research, ACTA Acustica united with Acustica, vol. 91, nº 3, pp. 409-420, (2005).

[30] Zhang, W., Samarasinghe, P. N., Chen, H. y Abhayapala, T. D., Surround by sound: A review of spatial audio recording and reproduction, Applied Sciences, vol. 7, nº 5, p. 532, (2017).

[31] Institute of Technical Acoustics, The ITA HRTF-database: A high-resolution head-related transfer function (HRTF) and three-dimensional (3D) ear model database, Extraído el 26 de Junio de 2019, http://www.akustik.rwth-aachen.de/cms/Technische-Akustik/Forschung/~lsly/HRTF-Datenbank/?lidx=1.

[32] HDF Group, Extraído el 26 de Junio de 2019, https://support.hdfgroup.org/HDF5/.

[33] UCAR, Extraído el 26 de Junio de 2019, https://www.unidata.ucar.edu/software/netcdf/docs/.

[34] SOFAcoustics, Extraído el 26 de Junio de 2019, http://sofacoustics.org/data/.

[35] Zotkin, D. N., Duraiswami, R. y Davis, L. S., Rendering localized spatial audio in a virtual auditory space, IEEE Transactions on multimedia, vol. 6, nº 4, pp. 553-564, (2004).

[36] Wefers, F., Partitioned convolution algorithms for real-time auralization, Logos Verlag Berlin GmbH, (2015).

[37] Wefers, F. y Vorländer, M., Efficient time-varying FIR filtering using crossfading implemented in the DFT domain., Proceedings of the 2014 7th Medical and Physics Conference Forum Acusticum, Cracow, Poland, (2014).

[38] MathWorks, Extraído el 26 de Junio de 2019, https://www.mathworks.com/help/audio/gs/real-time-audio-in-matlab.html.

[39] MathWorks, Extraído el 26 de Junio de 2019, https://www.mathworks.com/help/dsp/ref/dsp.audiofilereader-system-object.html.

[40] MathWorks, Extraído el 26 de Junio de 2019, https://www.mathworks.com/help/audio/ref/audiodevicewriter-system-object.html.

[41] GitHub, SOFAcoustics, Extraído el 26 de Junio de 2019, https://github.com/sofacoustics/API_MO.

[42] Wikipedia, Extraído el 26 de Junio de 2019, https://en.wikipedia.org/wiki/Inertial_measurement_unit.

[43] GitHub, Winer, K., Extraído el 26 de Junio de 2019, https://github.com/kriswiner/MPU6050/wiki/Affordable-9-DoF-Sensor-Fusion.

[44] InvenSense, MPU-9250 Product Specification Revision 1.1, (2016).

[45] Wikipedia, Extraído el 26 de Junio de 2019, https://es.wikipedia.org/wiki/%C3%81ngulos_de_navegaci%C3%B3n.

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

[47] Wikipedia, Extraído el 26 de Junio de 2019, https://en.wikipedia.org/wiki/Conversion_between_quaternions_and_Euler_angles.

[48] GitHub, Winer, K., Extraído el 26 de Junio de 2019, https://github.com/kriswiner/MPU9250.

[49] GitHub, SparkFun, Extraído el 26 de Junio de 2019, https://github.com/sparkfun/SparkFun_MPU-9250_Breakout_Arduino_Library.

[50] GitHub, Winer, K., Extraído el 26 de Junio de 2019, https://github.com/kriswiner/MPU6050/wiki/Simple-and-Effective-Magnetometer-Calibration.

[51] Extraído el 26 de Junio de 2019, http://www.magnetic-declination.com.

[52] Repositorio en línea, creado el 11 de Julio de 2019, https://github.com/ezeblanco/SintesisBinaural.

[53] Piegari, A., Desarrollo de dispositivo electrónico de bajo costo para evaluaciar localización de fuentes sonoras, Tesis de Grado, Universidad Nacional de Tres de Febrero, (2017).

[54] Arend, J. M., Neidhardt, A. y Pörschmann, C., Measurement and Perceptual Evaluation of a Spherical Near-Field HRTF Set, Proc. 29th Tonmeistertagung - VDT international convention, Köln, (2016).

[55] SOFAcoustics, Technische Hochschule Köln, Extraído el 26 de Junio de 2019, http://www.sofacoustics.org/data/database/thk/.

[56] Rothbucher, M., Development and Evaluation of an Immersive Audio Conferencing System, Tesis de Doctorado, Facultad de Ingeniería Eléctrica y Tecnología de la Información, Universidad Técnica de Munich, (2014).

[57] Stitt, P., Hendrickx, E., Messonnier, J. C. y Katz, B. F., The role of head tracking in binaural rendering, Tonmeistertagung TMT, pp. 350-355, (2016).

[58] I. T. U. R. Recommendation, Subjective Assessment of Sound Quality., International Telecommunication Union, BS: 562-3., (1990).

ANEXO A – CÓDIGOS IMPLEMENTACIONES DE CONVOLUCIÓN EN TIEMPO REAL

Función con implementación de la convolución Overlap-Add

```
function [out, overlap_block] = partitioned_convolution_OA(audio_in,filtros,overlap_block)
% Convolución por bloques en el dominio de las frecuencias
% Entrada:
%   audio_in          - Buffer de audio
%   filtro            - par de respuestas al impulso a convolucionar
%   overlap_block     - Bloque del procesamiento previo. La primera vez son ceros
% Salida:
%   out               - Buffer de salida procesado
%   overlap_block     - Bloque excedente de la convolución para sumar en el próximo procesamiento
%% Parámetros
L1 = length(audio_in);       % longitud de la señal de audio
N1 = length(filtros);          % longitud de los filtros
%% Ajuste de la longitud de la señal para que sea multiplo entero de los filtros
audio_in = [audio_in;zeros(N1-mod(L1,N1),2)]; % se agregan ceros para que sea múltiplo entero del filtro
L2 = length(audio_in);
%% Convolución overlap-add
out = zeros(L1+N1,2);
fft_filtro = fft(filtros,2*N1);            % FFT filtro

ii=0;
iiend=L2-N1;
while ii<iiend
      segT = audio_in(ii+1:ii+N1,:);   % segmento en el dominio temporal
      segF = fft(segT,2*N1);           % segmento en el dominio de las frecuencias con agregado de ceros
      segFO = fft_filtro.*segF;        % Multiplicación
      segTO = real(ifft(segFO));       % vuelve al dominio temporal
      out(ii+1:ii+2*N1,:)=out(ii+1:ii+2*N1,:)+segTO;    % superposición y suma
      ii=ii+N1;
end

out(1:N1,:) = out(1:N1,:) + overlap_block;    % sumo el bloque excedente del buffer previo
overlap_block = [out(L1+1:L1+N1,1) out(L1+1:L1+N1,2)];       % bloque excedente para el próximo
procesamiento
out=[out(1:L1,1) out(1:L1,2)];       % muestras para entregar al buffer de salida
end
```

Función con implementación de la convolución Overlap-Save

```
function [out,pad_block] = partitioned_convolution_OS(audio_in,filtros,pad_block)
% Convolución por bloques en el dominio de las frecuencias
% Entrada:
%   audio_in        - Buffer de audio
%   filtro          - par de respuestas al impulso a convolucionar
%   pad_block       - bloque de audio previo para rellenar. La primera vez son ceros
% Salida:
%   out             - Buffer de salida procesado
%   pad_block       - Bloque excedente de la convolucion para agregar en el próximo procesamiento (save
and reuse)
%% Parámetros
L1 = length(audio_in);      % longitud de la señal de audio
N1 = length(filtros);       % longitud de los filtros
%% Ajuste
filtros = [filtros;zeros(N1,2)];        % pad de ceros al filtro
%% Convolución overlap-save
out = zeros(L1,2);
fft_filtro = fft(filtros);      % FFT filtro

ii=0;
iiend=L1;
while ii<iiend
    segT = [pad_block;audio_in(ii+1:ii+N1,:)];          % segmento en el dominio temporal
    segF = fft(segT);
    segFO = fft_filtro.*segF;       % Multiplicación
    segTO = real(ifft(segFO));      % Dominio temporal
    out(ii+1:ii+N1,:) = segTO(N1+1:2*N1,:);             % se descartan las primeras N1 muestras debido al
aliasing temporal
    pad_block = audio_in(ii+1:ii+N1,:);
    ii = ii+N1;
end
```

ANEXO B – CÓDIGOS IMPLEMENTACIONES DE FUNDIDO CRUZADO DE FILTROS VARIANTES EN EL TIEMPO

Función con implementación de la convolución Overlap-Save con fundido cruzado en el dominio temporal.

```
function [out,pad_block] = partitioned_convolution_OS_CF(audio_in,filtros,pad_block,filtros_previos)
% Convolución por bloques en el dominio de las frecuencias
% Entrada:
%   audio_in        - Buffer de audio.
%   filtro          - Par de respuestas al impulso a convolucionar.
%   pad_block       - Bloque de audio previo para rellenar. La primera vez son ceros.
%   filtros_previos - Son los filtros con la posición de fuente previa para realizar el fundido cruzado.
% Salida:
%   out             - Buffer de salida procesado.
%   pad_block       - Bloque excedente de la convolución para agregar en el próximo procesamiento (save
and reuse).
%% Argumentos
if nargin < 4
   crossfade = 0;              % no se activa el crossfade
else
   crossfade = 1;
end
%% Parámetros
L1 = length(audio_in);      % longitud de la señal de audio
N1 = length(filtros);        % longitud de los filtros
%% Ajuste
filtros = [filtros;zeros(N1,2)];         % pad de ceros al filtro
%% Convolución overlap-save
out = zeros(L1,2);
fft_filtro = fft(filtros);

ii=0;
iiend=L1;
if crossfade == 1
   filtros_previos = [filtros_previos;zeros(N1,2)];
   fft_filtro_previo = fft(filtros_previos);
   % se generan las funciones de fundido
   long_cf = N1;   % Longitud del fundido cruzado
   fade_out = (0.5+(0.5*cos(pi*linspace(1,2*long_cf,long_cf)/(2*long_cf))))';
   fade_in = (0.5-(0.5*cos(pi*linspace(1,2*long_cf,long_cf)/(2*long_cf))))';

   while ii<iiend
      segT = [pad_block;audio_in(ii+1:ii+N1,:)];          % segmento en el dominio temporal
      segF = fft(segT);
      segFO = fft_filtro.*segF;          % Multiplicación en el espectro
      segTO = real(ifft(segFO));         % Dominio temporal

      if ii == 0
      segFO2 = fft_filtro_previo.*segF;
      segTO2 = real(ifft(segFO2));
      % crossfading
      if long_cf == N1
      out(ii+1:ii+N1,1) = fade_in .* segTO(N1+1:2*N1,1) + fade_out .* segTO2(N1+1:2*N1,1);
```

```
      out(ii+1:ii+N1,2) = fade_in .* segTO(N1+1:2*N1,2) + fade_out .* segTO2(N1+1:2*N1,2);
      pad_block = audio_in(ii+1:ii+N1,:);
      ii = ii+N1;
      else            % si se utiliza una longitud menor a N1
      out(ii+1:ii+long_cf,1) = fade_in .* segTO(N1+1:N1+long_cf,1) + fade_out .* segTO2(N1+1:N1+long_cf,1);
      out(ii+1:ii+long_cf,2) = fade_in .* segTO(N1+1:N1+long_cf,2) + fade_out .* segTO2(N1+1:N1+long_cf,2);
      out(ii+long_cf+1:ii+N1,:) = segTO(N1+long_cf+1:2*N1,:);
      pad_block = audio_in(ii+1:ii+N1,:);
      ii = ii+N1;
      end
      else
      out(ii+1:ii+N1,:) = segTO(N1+1:2*N1,:);
      pad_block = audio_in(ii+1:ii+N1,:);
      ii = ii+N1;
      end
   end
else      % crossfade == 0
   while ii<iiend
      segT = [pad_block;audio_in(ii+1:ii+N1,:)];          % segmento en el dominio temporal
      segF = fft(segT);
      segFO = fft_filtro.*segF;       % Multiplicación
      segTO = real(ifft(segFO));      % Dominio temporal
      out(ii+1:ii+N1,:) = segTO(N1+1:2*N1,:); % se descartan la primeras N1 muestras debido al aliasing
temporal
      pad_block = audio_in(ii+1:ii+N1,:);
      ii = ii+N1;
   end
end
```

Función con implementación de la convolución Overlap-Save con fundido cruzado en el dominio de las frecuencias.

```
function [out,pad_block] = partitioned_convolution_OS_CF_DFT(audio_in,filtros,pad_block,filtros_previos)
% Convolución por bloques en el dominio de las frecuencias
% Entrada:
%   audio_in         - Buffer de audio.
%   filtro           - Par de respuestas al impulso a convolucionar.
%   pad_block        - Bloque de audio previo para rellenar. La primera vez son ceros.
%   filtros_previos - Son los filtros con la posición de fuente previa para realizar el crossfading.
% Salida:
%   out              - Buffer de salida procesado.
%   pad_block        - Bloque excedente de la convolución para agregar en el próximo procesamiento (save
and reuse).
%% Argumentos
if nargin < 4
   crossfade = 0;  % no se activa el crossfade
else
   crossfade = 1;
end
%% Parámetros
L1 = length(audio_in);      % longitud de la señal de audio
N1 = length(filtros);       % longitud de los filtros
%% Ajuste
```

```
filtros = [filtros;zeros(N1,2)];          % pad de ceros al filtro
%% Convolución overlap-save
out = zeros(L1,2);
fft_filtro = fft(filtros);

ii=0;
iiend=L1;
if crossfade == 1
   filtros_previos = [filtros_previos;zeros(N1,2)];
   fft_filtro_previo = fft(filtros_previos);

   while ii<iiend
      segT = [pad_block;audio_in(ii+1:ii+N1,:)];          % segmento temporal
      segF = fft(segT);

      if ii == 0
      % crossfading en el espectro con convolución circular
      segFO = zeros(L1,2);
      Y0 = segF.*fft_filtro_previo;
      Y1 = segF.*fft_filtro;

            for j=1:2
            segFO(1,j) = Y0(1,j) + Y1(1,j) + 0.5*(Y1(2,j) - Y0(2,j) + Y1(L1,j) - Y0(L1,j));
            for i=1:L1/2-2
            segFO(i+1,j) = Y0(i+1,j) + Y1(i+1,j) + 0.5*(Y1(i+2,j) - Y0(i+2,j) + Y1(i,j) - Y0(i,j));
            end
            end
      segTO = real(ifft(segFO));        % Dominio temporal
      out(ii+1:ii+N1,:) = segTO(N1+1:2*N1,:);
      pad_block = audio_in(ii+1:ii+N1,:);
      ii = ii+N1;

      else
      segFO = fft_filtro.*segF;         % Multiplicación
      segTO = real(ifft(segFO));        % Dominio temporal
      out(ii+1:ii+N1,:) = segTO(N1+1:2*N1,:);
      pad_block = audio_in(ii+1:ii+N1,:);
      ii = ii+N1;
      end
   end
else
   while ii<iiend
      segT = [pad_block;audio_in(ii+1:ii+N1,:)];          % segmento temporal
      segF = fft(segT);
      segFO = fft_filtro.*segF;         % Multiplicación
      segTO = real(ifft(segFO));        % Dominio temporal
      out(ii+1:ii+N1,:) = segTO(N1+1:2*N1,:);             % se descarta la primeras N1 muestras debido al
aliasing temporal
      pad_block = audio_in(ii+1:ii+N1,:);
      ii = ii+N1;
   end
end
```

ANEXO C – DIAGRAMA DE FLUJO DEL SISTEMA IMPLEMENTADO EN MATLAB

En este anexo se muestra el diagrama de flujo del sistema implementado en Matlab donde se realiza la síntesis binaural integrando el sensor de movimiento por Bluetooth.

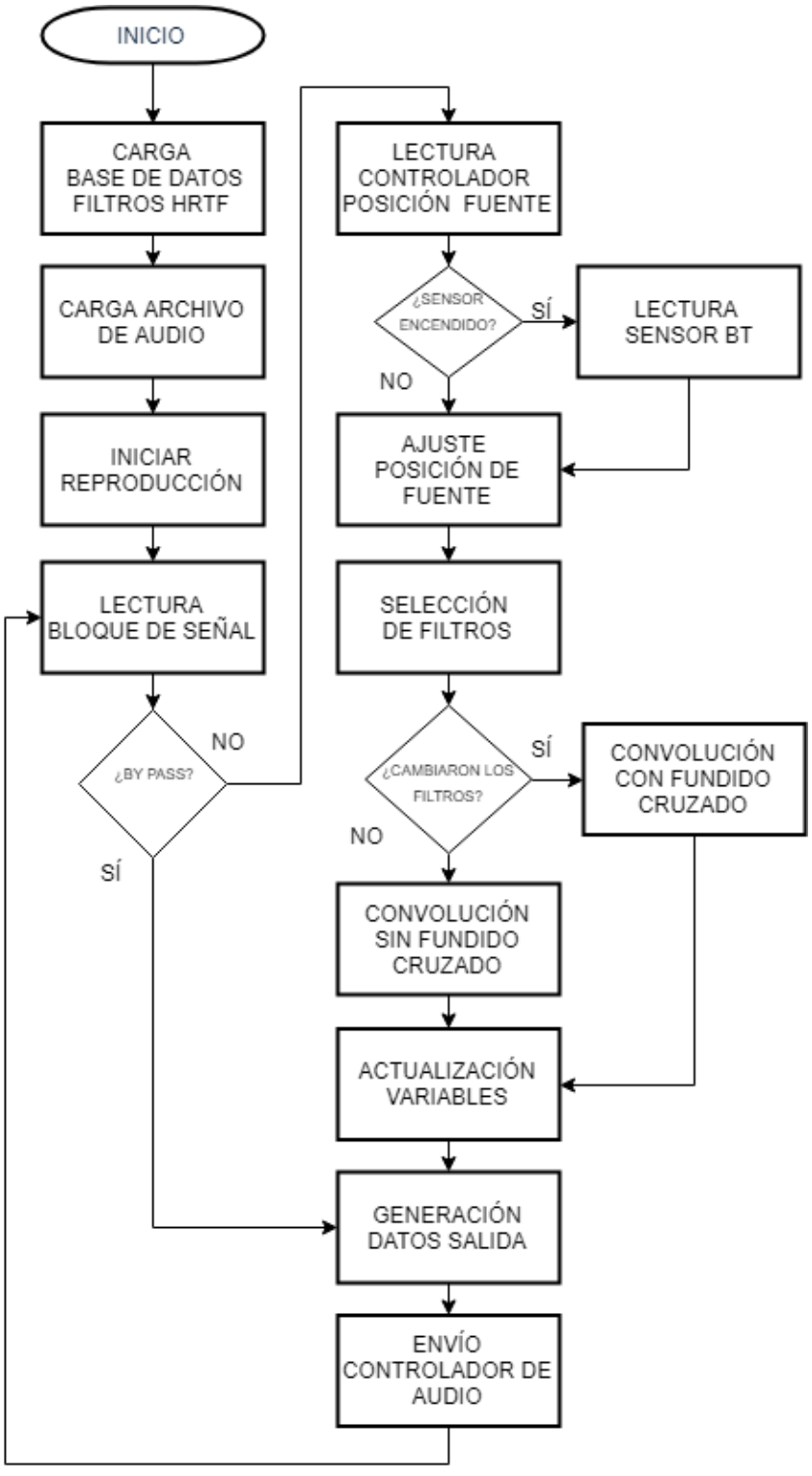

Figura 42. Diagrama de flujo del sistema implementado en Matlab con el sensor de movimiento.

ANEXO D – INTERFACES GRÁFICAS

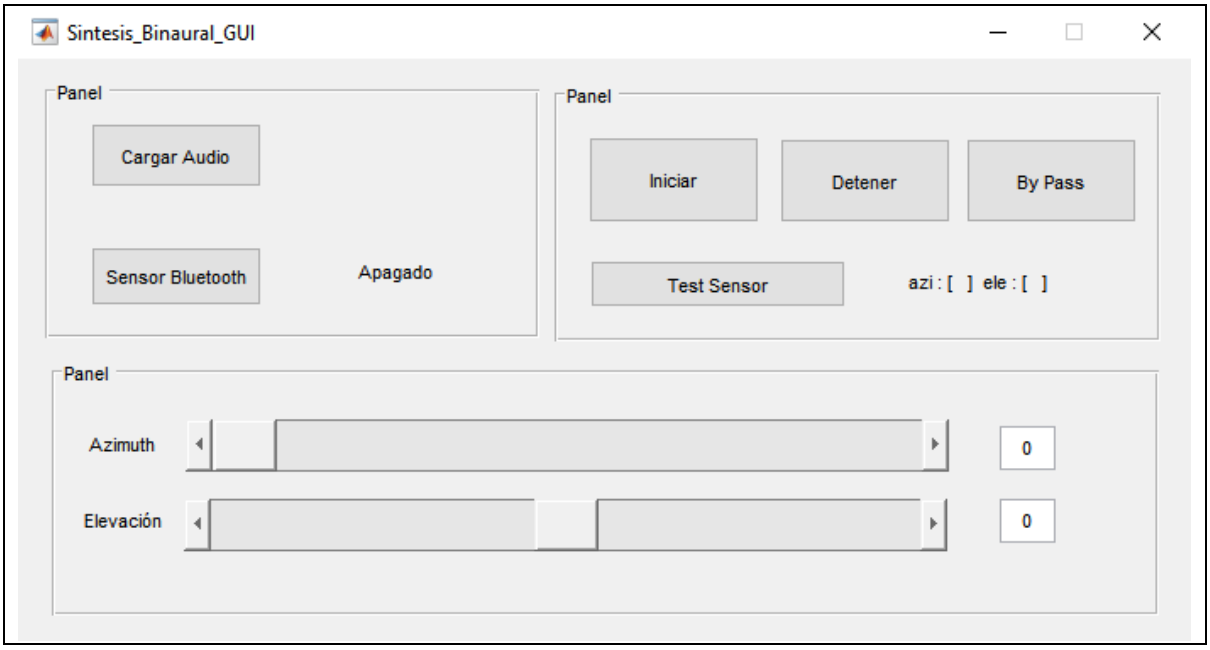

Figura 43. Interfaz gráfica del sistema de síntesis binaural que integra el sensor de movimiento.

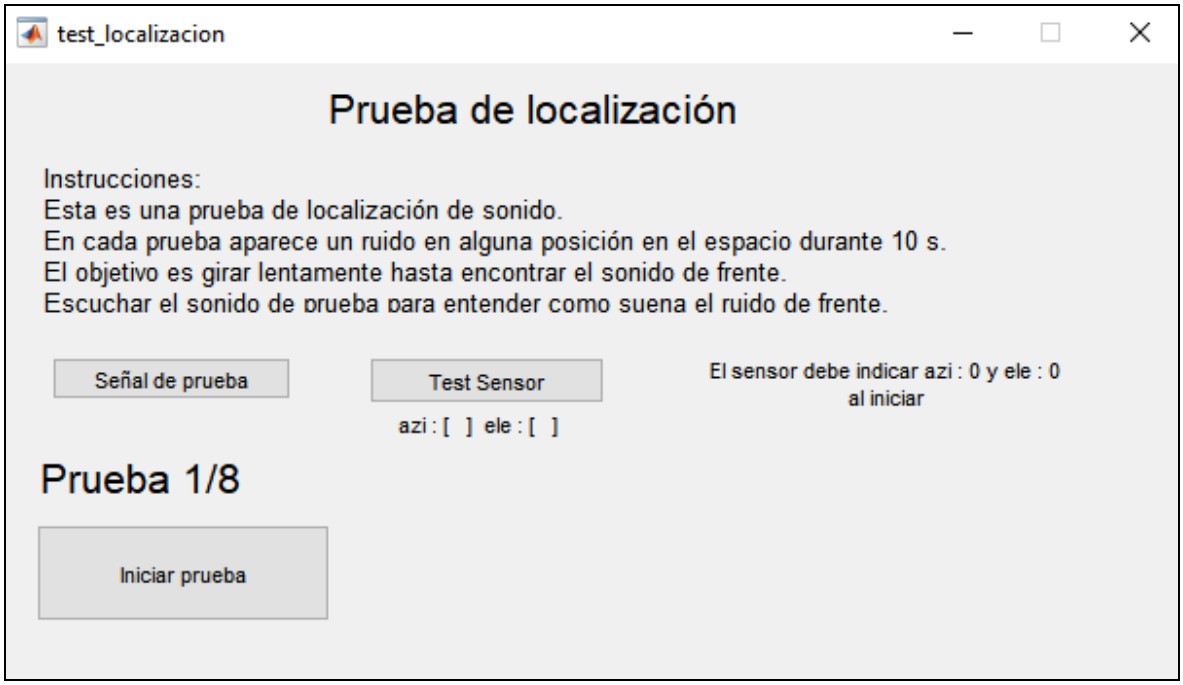

Figura 44. Interfaz gráfica para la prueba subjetiva de localización de fuente sonora.

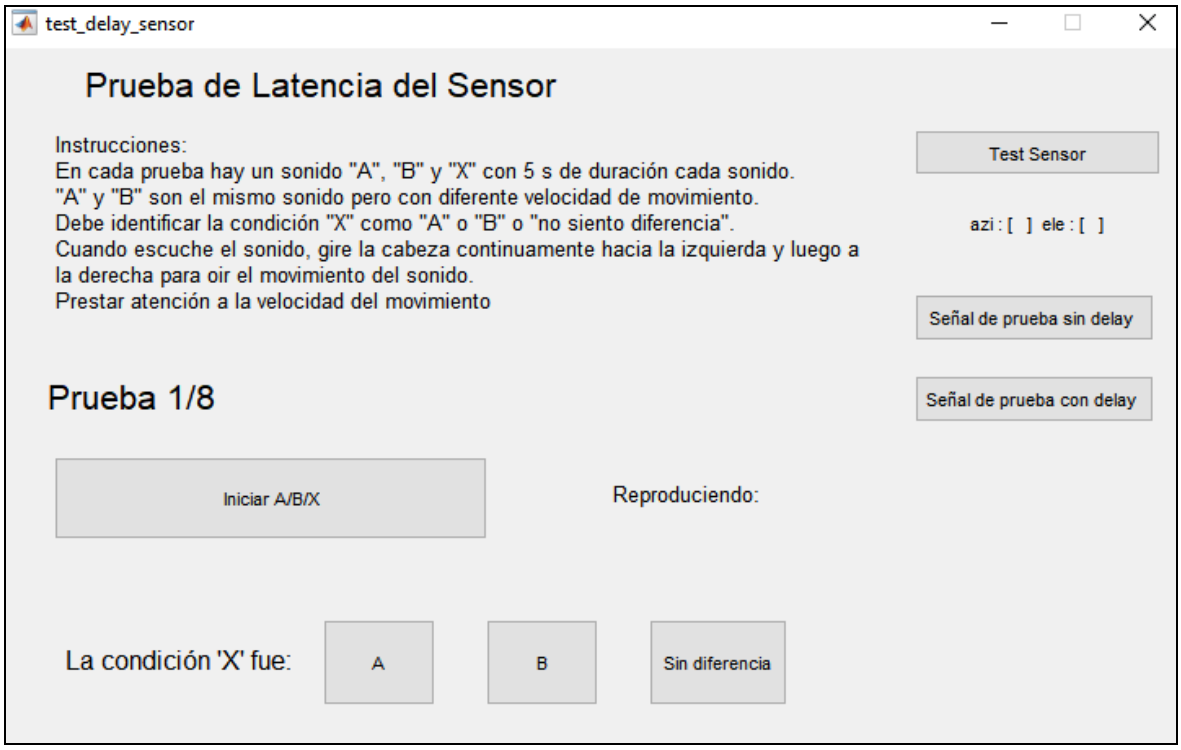

Figura 45. Interfaz gráfica para la prueba subjetiva de latencia del sensor.

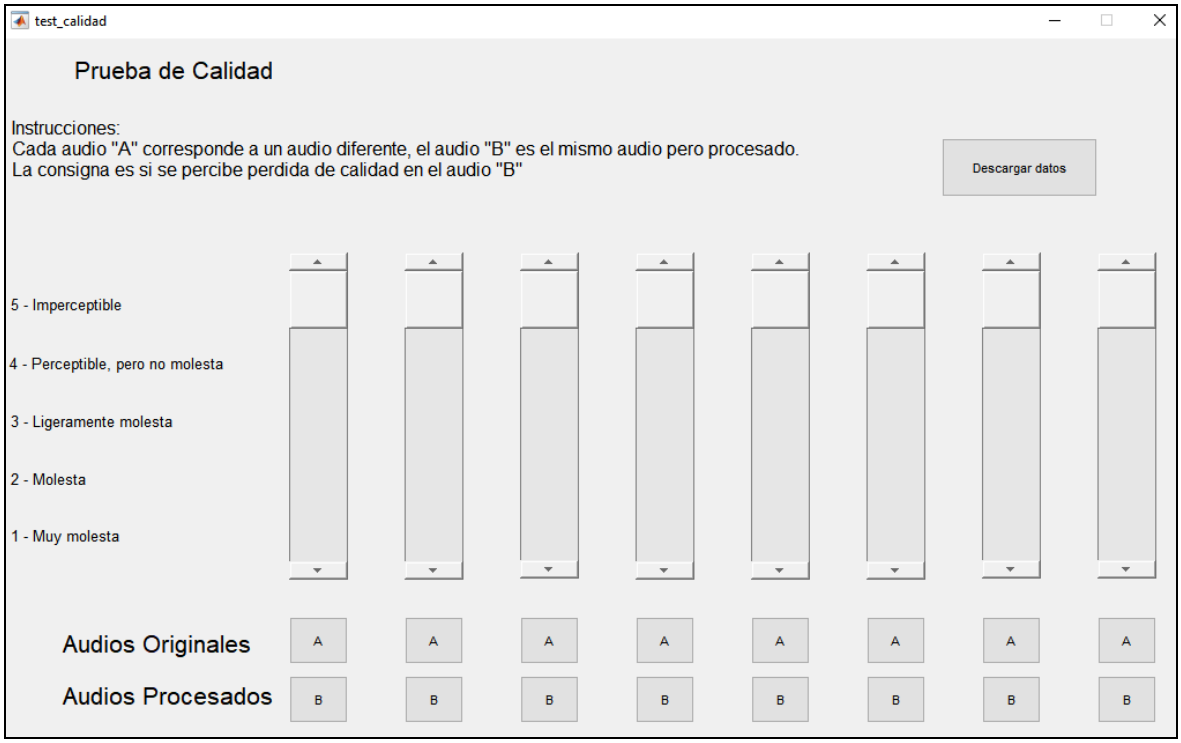

Figura 46. Interfaz gráfica de la prueba subjetiva de calidad.

ANEXO E – DIAGRAMAS DE FLUJO DE LAS PRUEBAS SUBJETIVAS

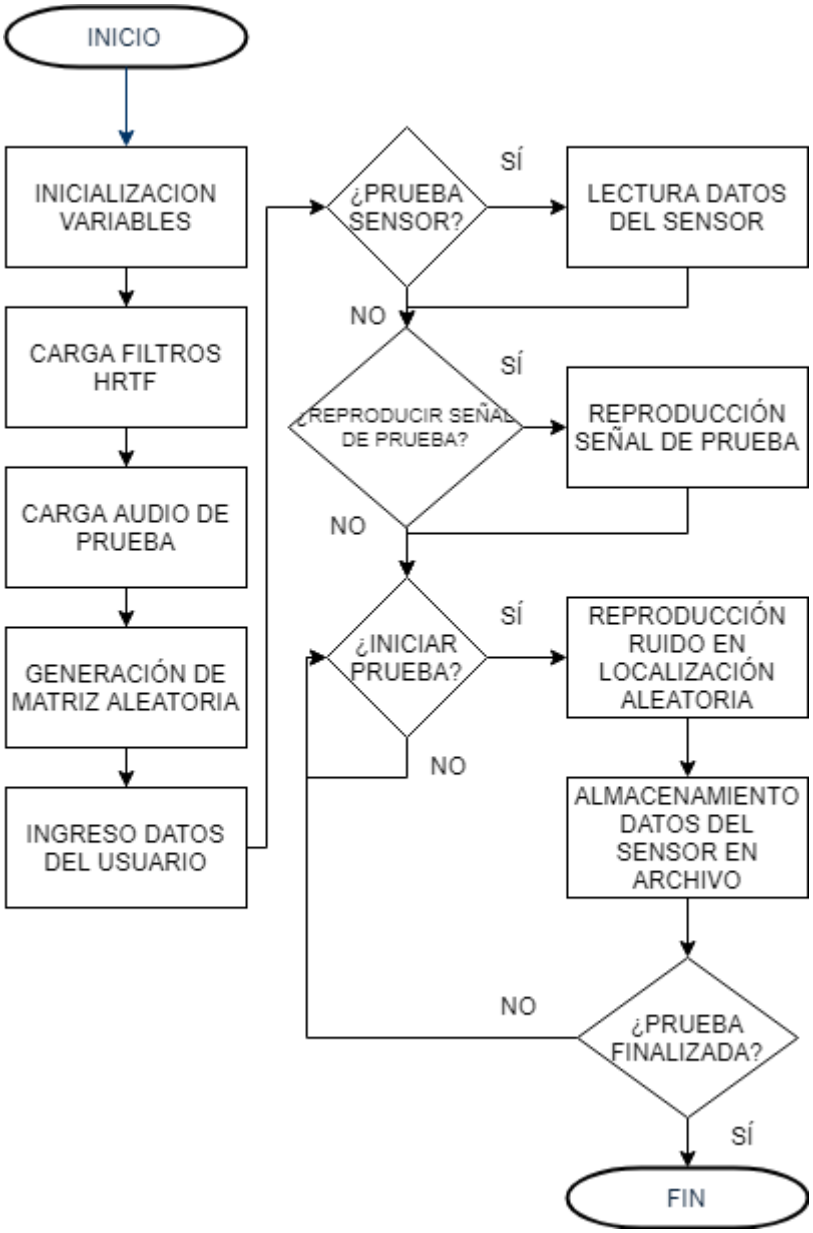

Figura 47. Diagrama de flujo de la prueba subjetiva de localización de fuente sonora.

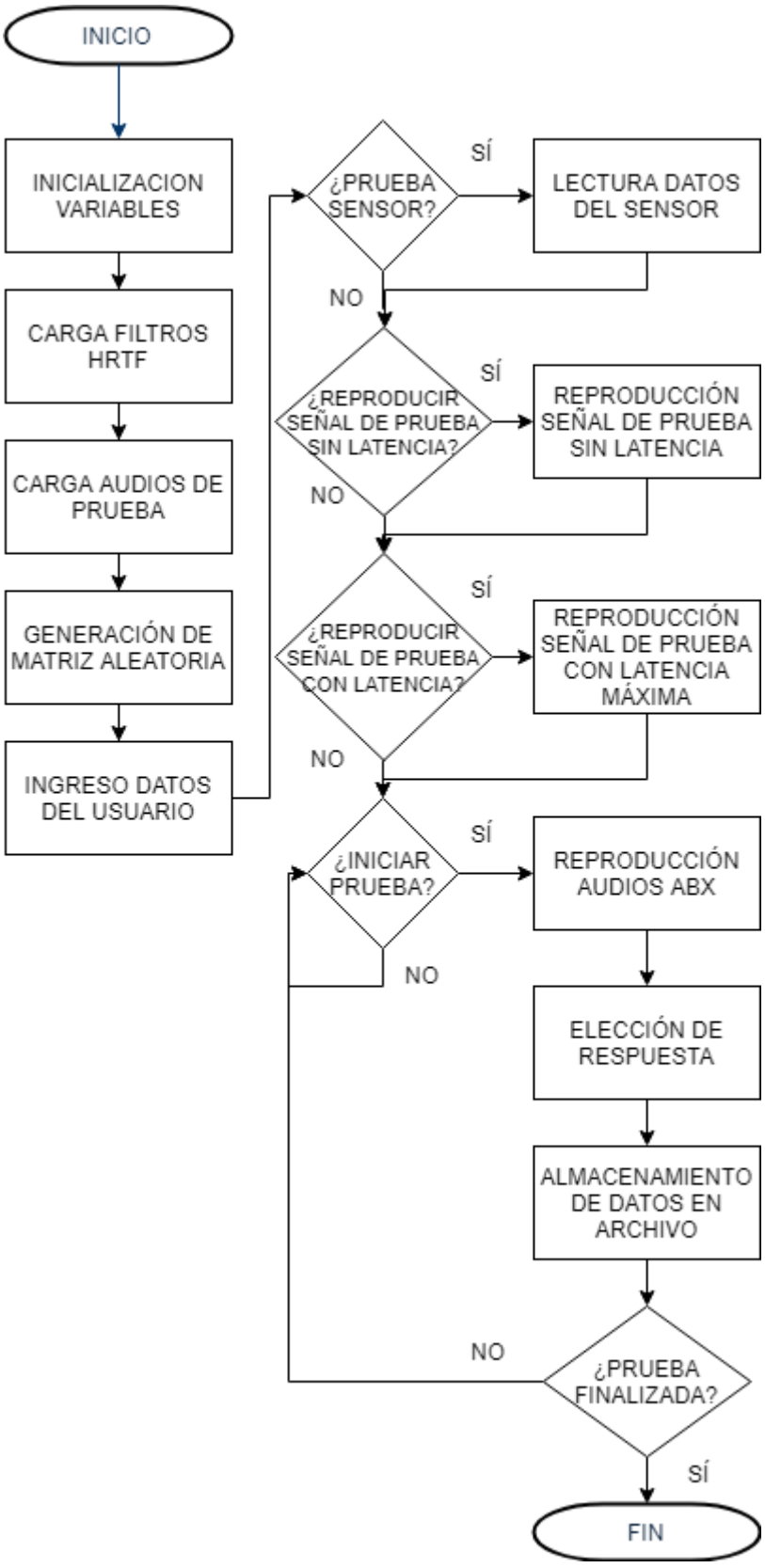

Figura 48. Diagrama de flujo de la prueba subjetiva de latencia del sensor de movimiento.