INTERACCION
Revista Digital de AIPO
Asociación Interacción Persona-Ordenador
Vol. 6, No 1 (2025)
ISSN 2695-6578

Renderizado de Audio Espacial en Realidad Virtual con el Binaural Rendering Toolbox (BRT)

Spatial Audio Rendering in Virtual Reality with the Binaural Rendering Toolbox (BRT)

Arcadio Reyes-Lecuona
Dpto. Tecnología Electrónica
Instituto TELMA
Universidad de Málaga, España
areyes@uma.es

Daniel González-Toledo
Dpto. Tecnología Electrónica
Instituto TELMA
Universidad de Málaga, España
dgonzalezt@uma.es

María Cuevas-Rodríguez
Dpto. Tecnología Electrónica
Instituto TELMA
Universidad de Málaga, España
mariacuevas@uma.es

Recibido: 01.04.2025 | Aceptado: 23.05.2025

Palabras Clave

Audio espacial
Realidad virtual
Renderizado binaural
Interacción inmersiva
Ciencia reproducible
Código abierto

Resumen

Este artículo presenta el Binaural Rendering Toolbox (BRT), que incluye una librería en C++ para el renderizado en tiempo real de escenas sonoras espaciales en entornos de realidad virtual, así como, un conjunto de herramientas diseñadas para la simulación de audio espacial. La librería del BRT es modular, extensible y de código abierto, permitiendo la simulación de fuentes dinámicas junto con modelos de entorno acústico y escucha binaural. Su arquitectura basada en bloques interconectables facilita su adaptación a distintos escenarios, desde simulaciones simples hasta entornos complejos. Desarrollada en el marco del proyecto europeo SONICOM, esta librería continúa la línea del 3D Tune-In Toolkit, integrando varios de sus algoritmos. Al estar implementada como librería de cabeceras (header-only), se integra fácilmente en proyectos en C++. Dentro del BRT, también se incluye BeRTA, una aplicación independiente que se comunica con motores gráficos como Unity mediante el protocolo OSC. El artículo expone los fundamentos del audio espacial, describe la arquitectura del sistema, su integración en flujos de trabajo en VR y sus opciones de configuración. Además, se presenta un caso de uso centrado en un estudio de interacción inmersiva y se discuten sus ventajas como herramienta científica, especialmente en el contexto de investigación reproducible en percepción auditiva e interacción multisensorial.

Keywords

Spatial audio
Virtual reality
Binaural rendering
Immersive interaction
Reproducible science
Open-source

Abstract

This paper introduces the Binaural Rendering Toolbox (BRT), which includes a C++ library for real-time spatial audio rendering in virtual reality environments, as well as a suite of tools for spatial audio simulation. The library is modular, extensible and open-source, enabling the simulation of dynamic sound sources along with acoustic environment models and binaural listening. Its architecture, based on interconnectable blocks, allows adaptation to various scenarios, from lightweight simulations to complex environments. Developed within the European project SONICOM, this library builds on the 3D Tune-In Toolkit, integrating several of its algorithms. As a header-only library, it seamlessly integrates into C++ projects. BRT also includes BeRTA, a standalone application that communicates with engines like Unity via the OSC protocol. The article explores the spatial hearing principles, describes the system architecture, its integration into VR workflow, and its configuration options. Additionally, it presents a use case in an immersive interaction study and discusses its advantages as a scientific tool, particularly for reproducible research in auditory perception and multisensory interaction.

1. Introducción

En los últimos años, la realidad virtual (RV) se ha consolidado como una tecnología clave en numerosos campos, desde la formación profesional y la educación hasta la medicina, la arquitectura o el diseño de experiencias inmersivas. En muchos de estos contextos, el foco principal ha estado tradicionalmente en el aspecto visual, lo que ha llevado a desarrollar motores gráficos muy avanzados y plataformas que ofrecen un gran realismo en la representación tridimensional. Sin embargo, el sonido espacial, especialmente el que se reproduce mediante auriculares, desempeña un papel igualmente importante en la construcción de entornos plausibles e inmersivos (Bergstrom et al., 2017), y su integración adecuada puede tener un impacto significativo tanto en la experiencia subjetiva del usuario como en el rendimiento de tareas perceptivas o interactivas (Serafin, 2023).

Diversos trabajos han señalado que el sonido espacial sigue siendo, en muchos casos, una componente poco desarrollada en el diseño de aplicaciones inmersivas especialmente fuera de contextos altamente especializados (Serafin et al., 2018). A ello se suma la falta de herramientas abiertas y configurables que permitan un control preciso sobre los parámetros acústicos y que puedan integrarse fácilmente en plataformas de desarrollo. Esta situación limita tanto el potencial de las simulaciones como la posibilidad de llevar a cabo investigaciones sistemáticas y reproducibles en percepción auditiva tridimensional.

Esta falta de conexión entre el desarrollo de experiencias inmersivas en VR y los avances en audio espacial son un obstáculo para alcanzar una verdadera integración multisensorial, capaz de reforzar la sensación de presencia y mejorar la calidad de la interacción (Reyes-Lecuona, A et al., 2022). En este contexto, el presente artículo ofrece una presentación de la librería de C++ del Binaural Rendering Toolbox (BRT), una herramienta de código abierto diseñada precisamente para facilitar la incorporación de sonido espacializado realista en aplicaciones de investigación y en entornos de realidad virtual.

A lo largo del artículo, se describe el diseño y funcionamiento de la librería BRT, poniendo especial énfasis en sus capacidades para ser utilizada dentro de contextos de investigación científica reproducible. En este sentido, uno de los principios fundamentales que han guiado su desarrollo ha sido la necesidad de disponer de una herramienta que no solo permita generar sonido espacializado de alta calidad, sino también hacerlo de forma controlable, transparente y documentada. En muchos estudios científicos, especialmente aquellos relacionados con la percepción o la interacción, es esencial poder compartir no solo los resultados, sino también los procedimientos exactos utilizados en la generación de los estímulos. Frente a soluciones comerciales cerradas o cajas negras acústicas, una herramienta como BRT permite describir, replicar y verificar todas las etapas del proceso de renderizado sonoro, contribuyendo así a los objetivos de la ciencia abierta y reproducible.

La estructura del artículo está organizada de la siguiente manera: en la Sección 2, se presentan los fundamentos del audio espacial, incluyendo los principios perceptivos que permiten la localización de sonidos en el espacio y el papel del entorno acústico en esta percepción. La Sección 3 introduce la arquitectura de la librería BRT, explicando su organización modular y los tipos de modelos disponibles para simular fuentes, entornos y oyentes. La Sección 4 describe en detalle cómo la librería puede integrarse en aplicaciones de realidad virtual, particularmente a través de su comunicación con motores gráficos como Unity mediante mensajes del protocolo Open Sound Control (OSC) (Wright et al., 1997). En la Sección 5, se analiza el sistema de configuración basado en ficheros JSON, que permite definir la arquitectura de procesamiento acústico y establecer los parámetros iniciales de la simulación. A continuación, la Sección 6 ilustra un caso de uso completo, centrado en un estudio sobre interacción tridimensional inmersiva, en el que se combina Unity, un visor Meta Quest 3 y el BRT para generar un entorno acústico realista y controlado. Finalmente, la Sección 7 ofrece una discusión de las posibilidades de esta herramienta, tanto desde el punto de vista técnico como metodológico, subrayando su utilidad como plataforma abierta para investigación y su contribución a la construcción de soluciones científicas robustas y replicables.

2. Fundamentos del Audio Espacial

La percepción espacial del sonido permite a las personas localizar fuentes sonoras en el entorno, contribuyendo a una interacción natural y efectiva con el medio ambiente. La localización del sonido en condiciones de campo abierto se basa en varios fenómenos fisiológicos (Blauert, 1997; Vorländer, 2020):

- **Diferencias de Tiempo Interaural (ITD):** el sonido llega con pequeñas diferencias de tiempo a cada oído, dependiendo de la ubicación espacial de la fuente. Estas diferencias son interpretadas por el cerebro como información direccional relativa ala lateralidad del sonido.
- **Diferencias de Nivel Interaural (ILD):** Cuando una fuente sonora está más cerca de un oído que del otro, la cabeza actúa como barrera acústica, reduciendo la intensidad del sonido que llega al oído más alejado, especialmente para altas frecuencias. Al igual que el ITD, el ILD nos proporciona información de la lateralidad del sonido.

- **Indicios Espectrales:** el pabellón auditivo humano filtra el sonido según la dirección desde la cual proviene éste, proporcionando indicios adicionales sobre la localización vertical del sonido.

Aunque estos tres elementos son fundamentales para la localización de fuentes en espacio abierto, la percepción espacial del sonido va mucho más allá. En contextos realistas, las fuentes sonoras no solo se perciben en un punto determinado del espacio, sino también dentro de un entorno acústico específico, ya que éste modifica el sonido que llega al oyente.

Cuando una fuente sonora se encuentra en un recinto cerrado o semirrígido, como una habitación, una sala de conferencias o una nave industrial, su sonido no llega directamente al oyente sin más. Parte de la energía acústica se propaga en línea recta, pero otra parte interactúa con las superficies del entorno: paredes, suelo, techo, objetos. Estas interacciones generan reflexiones y reverberaciones que enriquecen y complican el patrón acústico recibido por el oyente (Välimäki et al, 2012 & 2016). Esta información reverberante contribuye de forma decisiva a la percepción del entorno sonoro. Por ejemplo, permite identificar si estamos en un espacio cerrado o abierto, si se trata de un lugar amplio o estrecho, o incluso si las superficies son duras o absorbentes.

Además, el entorno acústico también influye en aspectos específicos de la localización, como por ejemplo la percepción de la distancia. En general, una fuente que produce muchas reflexiones y tiene un nivel relativo de sonido directo bajo será percibida como más lejana, mientras que una fuente con un componente directo fuerte y poca reverberación parecerá más próxima.

Por tanto, simular correctamente el entorno acústico es un componente esencial en el proceso de espacialización del sonido. Y es precisamente en esta capacidad donde herramientas como el BRT permiten un control detallado y flexible para aplicaciones de realidad virtual, en las que la recreación del entorno sonoro tiene un impacto directo en la sensación de inmersión del usuario.

2.1 ¿Qué es el Audio Binaural?

Una técnica ampliamente utilizada para recrear la percepción espacial del sonido en entornos virtuales es el audio binaural. Esta técnica se basa en la reproducción del sonido a través de auriculares, entregando señales distintas en cada oído, de forma que se simule el modo en que el cerebro recibe y procesa la información auditiva en situaciones reales (Algazi et al., 2021). Para conseguir esta simulación, se utilizan las denominadas funciones de transferencia relativas a la cabeza (HRTF, por sus siglas en inglés), que describen cómo el sonido es filtrado por la anatomía del oyente (especialmente por la cabeza y los pabellones auditivos) dependiendo de la dirección desde la que proviene (Xie, 2013). Aplicar estas funciones a una señal monofónica permite generar una versión binaural del sonido, es decir, una señal estereofónica especialmente procesada que reproduce la impresión de espacialidad cuando se escucha con auriculares.

Además de los indicios generales que permiten la localización del sonido, otro factor determinante en la espacialización binaural es la individualización del oyente. Aunque existen modelos auditivos basados en HRTFs promedio, que funcionan razonablemente bien en muchos casos, lo cierto es que las funciones de transferencia propias de cada individuo reflejan de manera mucho más precisa cómo su anatomía afecta al sonido. Cuando se utilizan HRTFs genéricos, es habitual que la espacialización resulte menos natural, con una localización de las fuentes menos precisa e incluso confusa. Por el contrario, las HRTFs individualizadas permiten una reconstrucción espacial mucho más fiel, mejorando la sensación de inmersión y la credibilidad del entorno virtual.

2.2 Renderizado Binaural y Realidad Virtual

En el contexto de la realidad virtual, el renderizado del sonido espacial plantea una serie de exigencias particulares que no siempre están presentes en otras aplicaciones que manejan sonido. A diferencia de una escena sonora estática o pregrabada, una aplicación de realidad virtual es, por definición, interactiva: el usuario se mueve, gira la cabeza, cambia de posición y desencadena eventos en el mundo virtual que modifican continuamente la escena sonora (Serafin et al., 2018). Además, los propios elementos del entorno virtual, como fuentes sonoras, objetos, superficies, condiciones ambientales, pueden cambiar también en tiempo real. Esto implica que el renderizado espacial debe actualizarse de forma dinámica, procesando el sonido para reflejar todos estos cambios. Técnicamente, esto supone aplicar procesos de convolución con HRTFs y simular la contribución del entorno en un flujo continuo de audio. Si dicho entorno se simula mediante convolución con la respuesta al impulso de una sala, hay que considerar que dichas convoluciones pueden implicar respuestas de hasta cientos de miles de muestras, se requiere el uso de algoritmos eficientes, como la convolución particionada, que permite distribuir el procesamiento en bloques más manejables con baja latencia y optimiza los algoritmos en el dominio de la frecuencia (Wefers, 2015). En paralelo, pueden emplearse modelos acústicos simplificados para reducir la carga computacional, evitando la necesidad de usar respuestas al impulso completas y sustituyéndolas por aproximaciones más ligeras. En cualquier caso, la clave está en encontrar un equilibrio entre realismo acústico y viabilidad computacional, de modo que se mantenga la interactividad fluida que caracteriza a las experiencias en realidad virtual.

Frente a estos desafíos, resulta fundamental contar con herramientas capaces de gestionar en tiempo real toda esta dinámica sonora de forma precisa y eficiente. No se trata únicamente de generar sonidos en distintas posiciones, sino de mantener actualizada, en cada instante, la representación espacial del conjunto de fuentes que componen la escena virtual, considerando también la posición y orientación del oyente. Además, este procesamiento debe ser flexible, permitiendo combinar distintos tipos de modelos acústicos, aplicar transformaciones binaurales y adaptar el flujo de señal según las necesidades específicas de cada aplicación. En este sentido, el BRT ha sido concebido como una solución modular, optimizada y de código abierto para abordar exactamente este tipo de escenarios: una herramienta que permite renderizar en tiempo real la espacialización binaural de fuentes puntuales dinámicas en entornos virtuales, manteniendo un control preciso sobre los modelos de fuente, entorno y oyente. Al ser completamente transparente en sus algoritmos e implementaciones, el BRT garantiza la replicabilidad experimental necesaria en la investigación científica. Proporciona los mecanismos necesarios para simular y actualizar continuamente la escena acústica conforme evoluciona el entorno virtual, lo que lo convierte en una opción particularmente adecuada para aplicaciones interactivas que requieran rigor metodológico y documentación exhaustiva del proceso de auralización. A continuación, se describe en detalle la arquitectura de esta herramienta.

3. Arquitectura de la librería BRT

El BRT es un conjunto de herramientas para la simulación de audio espacial, dentro del cual se incluye una librería *header-only* escrita en C++ y código abierto (*BRTLibrary*, 2025). Esta librería ha sido diseñada con una arquitectura modular que facilita su personalización y extensión según las necesidades específicas de cada proyecto. Su desarrollo se enmarca en el proyecto europeo SONICOM[1], centrado en la investigación sobre comunicación inmersiva mediante tecnologías auditivas personalizadas. La arquitectura de BRT ha sido esbozada en trabajos previos (González-Toledo et al, 2023 & 2025), donde se detallan sus principios de diseño, su organización en bloques funcionales y su orientación hacia aplicaciones interactivas. Además, ofrece una documentación on-line para facilitar su uso[2]. El BRT integra y expande los fundamentos establecidos en el 3D Tune-In Toolkit (Cuevas-Rodríguez et al., 2019), que sirvió como precursor y del cual se han heredado e integrado algunos algoritmos clave de procesado espacial. La librería del BRT representa, por tanto, una evolución directa de estos desarrollos anteriores, reforzada con nuevas funcionalidades, mayor modularidad y un enfoque más claro hacia la investigación científica y la integración con plataformas modernas de realidad virtual.

La arquitectura modular de BRT permite descomponer el proceso de espacialización de audio en distintos componentes especializados (ver Fig. 1). A continuación, se describen los diferentes módulos que conforman la herramienta, cada uno encargado de una etapa específica en la simulación acústica.

Modelos de Fuentes Sonoras

BRT implementa modelos específicos para representar fuentes sonoras en entornos virtuales:

- **Omnidireccionales**: emiten sonido con igual intensidad en todas direcciones.
- **Direccionales**: utilizan archivos SOFA (SOFA, 2025) que almacenan patrones complejos de directividad, permitiendo simular fuentes sonoras realistas.

Modelos del Entorno

BRT ofrece modelos para simular cómo el entorno afecta la propagación del sonido:

- **Modelo Free-Field**: simula el sonido en espacios abiertos, considerando distancia y efecto Doppler.
- **Modelo SDN (Scattering Delay Networks)**: simula reflexiones y reverberaciones en espacios cerrados mediante redes complejas de retardo.

Modelos de Oyente

Estos modelos determinan cómo llega el sonido al usuario hasta el momento, se han implementado estos modelos:

- **Convolución directa con HRTFs**: procesamiento individual por fuente sonora mediante convolución con HRTFs (Xie, 2013). El renderizador permite cargar ficheros de HRTF en formato SOFA (SOFA, 2025), lo que permite usar HRTFs individualizados para cada oyente.
- **Convolución ambisónica con HRTFs**: mezcla múltiples fuentes en un campo ambisónico antes de convolucionar, reduciendo la carga computacional.
- **Uso de Respuestas al Impulso Binaural de Sala (BRIR)**: combinan información del entorno para mayor realismo, con la opción de crear modelos híbridos mediante técnicas de ventana de tiempo.

[1] SONICOM: Transforming auditory-based social interaction and communication in VR/AR. https://www.sonicom.eu

[2] https://grupodiana.github.io/BRT-Documentation/library/

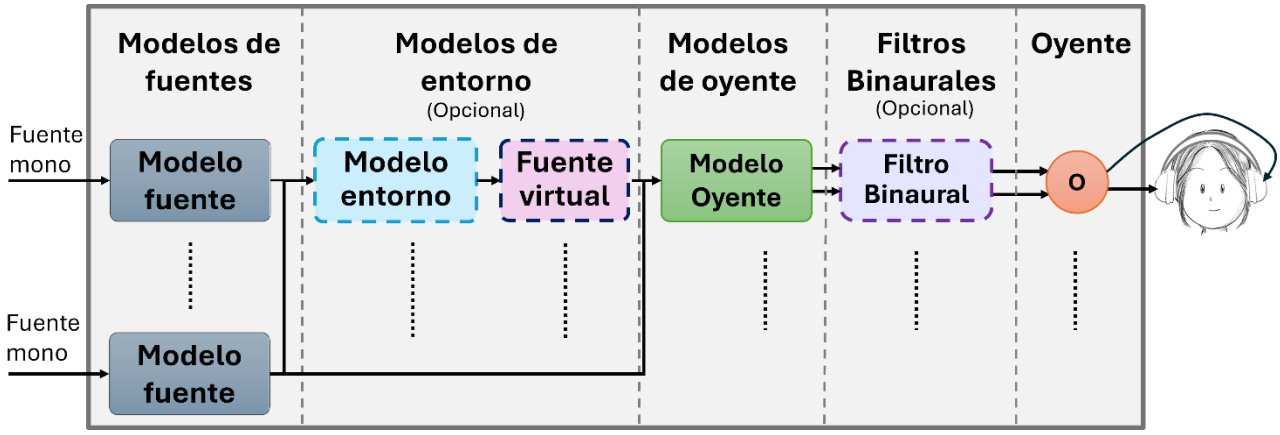

Figura 1: Diagrama de la arquitectura de BRT

Filtros Binaurales

Por último, los filtros binaurales añaden un procesamiento adicional, tras la espacialización binaural, tales como compensación de auriculares específicos, simulación de dispositivos protectores auditivos. En el futuro, estos filtros podrían incluir procesos no lineales avanzados, como la simulación de pérdida auditiva.

4. Integración de BRT en entornos de RV

La integración de una librería de procesamiento de audio en tiempo real, como la del BRT, en aplicaciones de realidad virtual presenta desafíos técnicos relacionados con la arquitectura de software y la interoperabilidad entre plataformas. Escrita en C++, el diseño modular de esta librería está enfocado en lograr un renderizado binaural eficiente en tiempo real. Por ello, su uso en entornos como Unity o Unreal Engine, motores ampliamente empleados en el desarrollo de aplicaciones de RV, requiere de un mecanismo robusto y flexible para la comunicación entre ambos sistemas.

Una opción habitual sería compilar BRT en forma de una biblioteca compartida (por ejemplo, como una DLL en Windows) e integrarla directamente dentro del motor gráfico. Esta vía permitiría una comunicación directa entre la lógica del entorno virtual y los procesos de renderizado sonoro. Sin embargo, este enfoque presenta ciertas limitaciones prácticas. En primer lugar, se trata de una integración que habría que hacer para cada plataforma. Es más, requeriría un mantenimiento continuo para mantener la compatibilidad con las futuras actualizaciones de los entornos de desarrollo. Por esta razón, aunque esta integración directa se contempla como una posibilidad futura, no ha sido la vía adoptada en la implementación inicial.

En su lugar, la estrategia seguida en el desarrollo del BRT se basa en una arquitectura más flexible y desacoplada. Se ha creado una aplicación autónoma, que encapsula la funcionalidad completa de la librería del BRT, y que se encarga de realizar todo el renderizado en tiempo real de las escenas sonoras. Esta aplicación, llamada Binaural Rendering Toolbox Renderer Application (en breve BeRTA Renderer), se ejecuta en segundo plano y se comunica con la aplicación de realidad virtual mediante el protocolo OSC, un protocolo ampliamente utilizado en entornos de audio profesional, que funciona sobre UDP y permite el envío de mensajes estructurados entre procesos o máquinas.

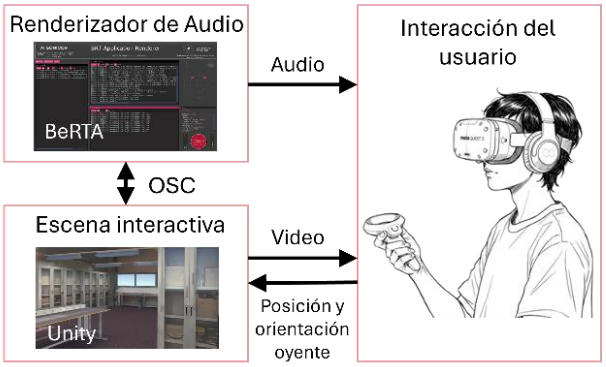

Figura 2: Ejemplo de configuración de BeRTA controlado por OSC

La Figura 2 muestra esta configuración, donde la aplicación de realidad virtual —por ejemplo, una escena interactiva desarrollada en Unity— se encarga de calcular continuamente la posición y orientación del oyente, así como de las fuentes sonoras presentes en la escena. Toda esa información se transmite en forma de mensajes OSC a la instancia de BRT, que está ejecutándose como proceso independiente. Los mensajes incluyen, entre otros, comandos de inicialización de fuentes, configuración de modelos, activación y detención de sonidos, y actualización de parámetros dinámicos. Esto permite que BeRTA reciba en tiempo real toda la información necesaria para mantener actualizada la representación acústica del entorno.

Una ventaja adicional de esta arquitectura distribuida es que permite, si se desea, ejecutar la aplicación BeRTA Renderer en

una máquina distinta a la del motor gráfico. De este modo, se puede distribuir la carga computacional entre varios sistemas, optimizando el rendimiento global, lo cual resulta especialmente útil en simulaciones complejas o con múltiples fuentes sonoras dinámicas. En la siguiente sección, se proporcionan más detalles de esta aplicación, y sus posibilidades.

5. El renderizador de audio BeRTA

BeRTA Renderer, que se ofrece tanto para Windows x64 (Windows 10 y 11) como para macOS Universal, ofrece un alto grado de configurabilidad para el renderizado de audio espacial en tiempo real. Su control se basa exclusivamente en OSC[1] por lo que su interfaz gráfica se centra en la monitorización del sistema más que en la manipulación directa de parámetros. La interfaz, la cual se muestra en la Fig. 3 presenta la información en formato de terminal, incluyendo el estado de la renderización, la configuración de fuentes y oyentes, así como registros detallados de los comandos OSC enviados y recibidos, acciones ejecutadas y mensajes del sistema de diagnóstico.

Entre sus características clave, BeRTA Renderer permite el renderizado en tiempo real con control dinámico de las posiciones de las fuentes sonoras y los oyentes mediante OSC. Soporta múltiples fuentes de sonido, intercambio de HRTF y BRIR en tiempo real, configuración de escenarios virtuales y exportación de datos de simulación.

Una de las características fundamentales de la arquitectura de la librería BRT es su modularidad. Esto significa que funcionalidades como la directividad de las fuentes sonoras, la propagación del sonido en el entorno o la percepción del oyente están implementadas como módulos independientes, permitiendo su combinación según las necesidades de cada aplicación.

BeRTA Renderer aprovecha esto para facilitar la configuración de escenarios virtuales adaptados a distintos requisitos. Por ejemplo, en aplicaciones donde las fuentes de sonido representan emisores direccionales (como altavoces, bocinas o la voz humana), es posible activar y conectar un módulo de directividad. Para simulaciones acústicas avanzadas, se puede integrar un módulo de convolución con la respuesta al impulso completa de un recinto. Alternativamente, si se requiere un equilibrio entre realismo y eficiencia computacional, es posible emplear un modelo híbrido, en el que el sonido directo de cada fuente se procesa individualmente mientras que la reverberación global se maneja con una representación ambisónica de primer orden (Zotter, 2019), optimizando el rendimiento en entornos con múltiples fuentes. En escenarios donde la carga computacional es una limitación, se puede optar por modelos simplificados que omiten la reverberación compleja.

La flexibilidad de BeRTA Renderer requiere una configuración inicial explícita que defina la arquitectura del sistema y los parámetros por defecto. A diferencia de los comandos OSC, que permiten modificar la escena en tiempo real, esta configuración se establece mediante un archivo en formato JSON. Este archivo describe los módulos activos, sus interconexiones y sus parámetros operativos, incluyendo los modelos de fuentes, entorno y oyente, así como ajustes técnicos esenciales como la frecuencia de muestreo, el tamaño del frame de procesamiento y el número de buffers de seguridad para garantizar una reproducción estable.

Además de la estructura del sistema, el archivo de configuración permite definir una escena inicial, especificando elementos como las posiciones de las fuentes y del oyente, los archivos de audio disponibles al inicio y los parámetros predeterminados de volumen y reproducción. Esto asegura que BeRTA Renderer arranque con una configuración coherente y lista para recibir comandos OSC que actualicen la escena de manera dinámica.

Secciones del archivo de configuración

El archivo JSON está estructurado en cinco secciones principales, cada una con un propósito específico dentro del proceso de renderizado binaural:

- **Ajustes generales**: Contiene parámetros globales de la aplicación, como la frecuencia de muestreo y la gestión de buffers, que afectan al rendimiento y estabilidad del sistema.
- **Arquitectura del modelo**: Define los modelos empleados para la representación del sonido, incluyendo los modelos de oyente y entorno, así como su interconexión. Esta es la sección central del archivo, ya que determina la estructura del renderizado y cómo se procesan las señales de audio espacial.
- **Recursos**: Lista los archivos de datos necesarios para el renderizado, como HRTFs, BRIRs, filtros de campo cercano y funciones de directividad. Estos recursos se cargan al inicio y pueden ser modificados dinámicamente mediante comandos OSC.

[1] Para facilitar su integración en distintos entornos, el BRT incluye varias soluciones de control para BeRTA Renderer, desde ejemplos en PureData, Unity y Matlab hasta una aplicación dedicada llamada BeRTA GUI.

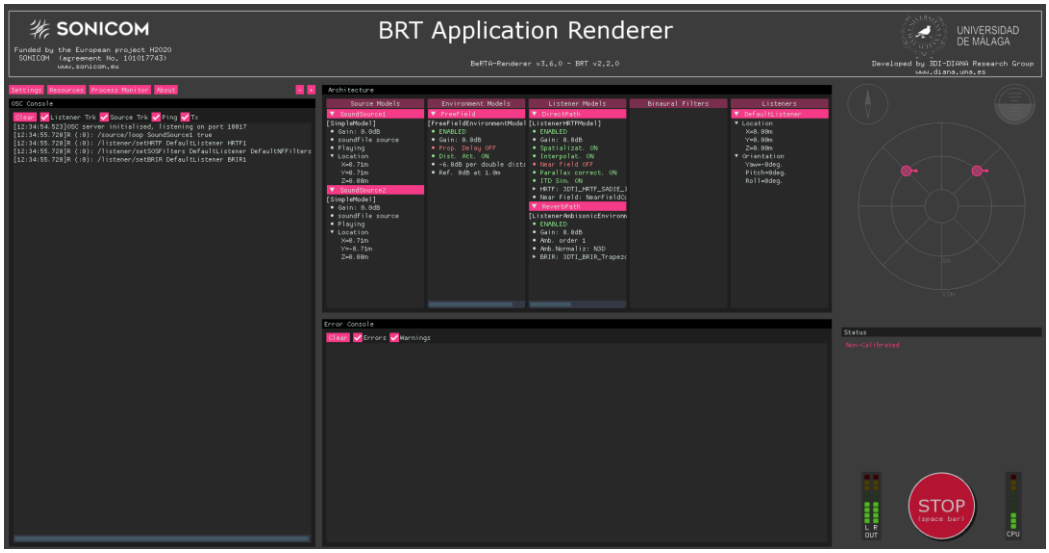

Figura 3: Interfaz de BeRTA, la cual es puramente informativa. BeRTA muestra los comandos OSC enviados y recibidos (consola de la izquierda), la configuración de la aplicación y posiciones de las fuentes y los oyentes (consola central arriba), los errores producidos en el renderizado y la comunicación OSC (consola central abajo) y un radar donde se ve gráficamente las fuentes que se están renderizando y su posición en el espacio (derecha).

Figura 2: Caso de uso donde se simula una sala real de la cual se ha capturado su respuesta al impulso.

- **Fuentes de sonido**: Define las fuentes de audio que estarán disponibles al iniciar el sistema. Cada fuente debe especificar el archivo de sonido asociado y el modelo de renderizado a utilizar, como *SimpleModel* o *DirectivityModel*.
- **Configuración de la escena**: Permite establecer una escena inicial mediante comandos en formato OSC. Esto incluye configuraciones como la activación de fuentes en bucle, la selección de HRTFs y la definición de propiedades ambientales, como el tamaño y la absorción acústica de la sala.

6. Caso de uso: Auralización de un entorno de interacción tridimensional en Realidad Virtual

Uno de los contextos más relevantes para la aplicación del BRT es la investigación en interacción tridimensional en entornos de Realidad Virtual. En esta sección se presenta un caso de uso que actúa como prueba de concepto, con el propósito de demostrar la viabilidad técnica de una investigación sobre cómo las características del entorno acústico afectan a la percepción espacial y a la calidad de la interacción inmersiva. Para ello, sería necesario no solo simular la geometría visual de la sala, sino también auralizar la experiencia de forma que el sonido percibido sea coherente con la posición del oyente y las fuentes dentro del entorno virtual. La *Figura 2* muestra una implementación de este caso de uso, donde se simula la geometría visual de una sala través de unas gafas Meta Quest. El renderizado de audio se lleva a cabo mediante BeRTA Renderer la cual simula la reverberación de la sala donde se encuentra el usuario, gracias a la captura previa del BRIR en esa misma sala.

En este escenario, un motor de Realidad Virtual, como por ejemplo Unity, se encarga de gestionar la visualización del mundo tridimensional (*Figura 2*, imagen derecha) y de rastrear en tiempo real la posición y orientación de la cabeza del

usuario y de las fuentes sonoras virtuales asociadas a objetos, interlocutores simulados o elementos interactivos. La escena puede incluir, por ejemplo, una conversación entre varios avatares, la manipulación de objetos con respuesta sonora y/o señales acústicas emitidas por elementos móviles en la sala.

Como se señaló en la sección anterior para incorporar el renderizado sonoro espacializado, se lanza en segundo plano la aplicación *standalone* BeRTA renderer, que encapsula la librería BRT y se encarga del procesamiento en tiempo real de toda la escena acústica.

En la *Figura 3* se muestra un ejemplo de configuración de BeRTA Renderer para este caso de uso. En esta configuración se declaran dos fuentes de sonido, ambas fuentes alimentarán dos rutas de audio distintas: una ruta de sonido directo ("DirectPath"), basada en convolución con HRTFs, y una ruta de reverberación ("ReverbPath"), basada en la convolución de una respuesta al impulso ambisónica de sala. En el caso del camino directo, antes de aplicar el modelo del oyente, se simula un modelo de entorno "FreeField", encargado de simular la propagación directa en espacio abierto. Esto permite una configuración híbrida donde las distintas componentes acústicas se procesan por caminos separados. Finalmente, ambos modelos de oyente se conectan al mismo oyente ("Default Listener"), que mezclará todos los canales para ofrecer una señal para canal derecho y otra para el canal izquierdo de los auriculares. El fichero JSON a utilizar para esta configuración se muestra en la Figura *4*.

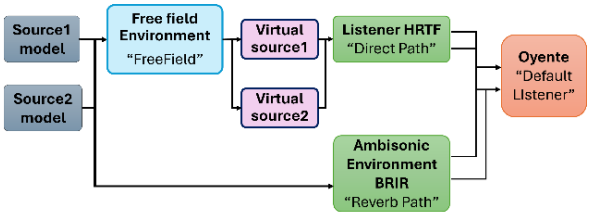

Figura 3: Diagrama de configuración de BeRTA

6.1 Configuración de BeRTA

Es importante señalar que, en el ejemplo anterior, las fuentes se conectan al modelo de entorno FreeField únicamente en la ruta que lleva al modelo de oyente DirectPath. En cambio, la conexión hacia el modelo ReverbPath se realiza directamente, sin pasar por un modelo de entorno intermedio. Esto se debe a que el modelo ReverbPath no representa únicamente al oyente, sino que encapsula tanto al oyente como al entorno acústico en una única respuesta al impulso. Al tratarse de una BRIR (Binaural Room Impulse Response), la señal ya incluye tanto el filtrado asociado a la posición del oyente como la respuesta reverberante de la sala. En otras palabras, la sala y el oyente forman parte inseparable del mismo modelo. Por esta razón, no es necesario anteponer un modelo de entorno como FreeField en esa rama de la arquitectura.

```
"GeneralSettings": {

    "SampleRate": 48000,

    "BufferSize": 512,

    "BufferFrames": 4,

    "OSCListenPort": 10017,

},


"ModelsArchitecture": {

    "Listeners":["DefaultListener"],

    "ListenerModels":[

    {"ID":"DirectPath","Model":ListenerHRTFModel"},

    {"ID":"ReverbPath","Model":
"ListenerAmbisonicEnvironmentBRIRModel"}],

    "EnvironmentModels":[

    {"ID":"FreeField", "Model":
"FreeFieldEnvironmentModel"}],

    "BinauralFilters":[],
```

. *Figura 4: Fichero JSON de configuración para un caso de uso específico, donde se declaran un modelo de entorno de tipo FreeField y dos modelos de oyente: uno que realiza la convolución directa con HRTFs y otro que realiza convolución en el dominio Ambisónico con el BRIR de la sala*

Esta arquitectura híbrida permite mantener el detalle direccional de las fuentes cercanas, mientras que se optimiza el procesamiento de la parte reverberante, que puede compartirse entre múltiples fuentes. Además, al ser una arquitectura flexible y declarativa, permite adaptar fácilmente el sistema a diferentes casos de uso, desde simulaciones ligeras hasta configuraciones acústicamente detalladas.

6.2 Comunicación de BeRTA con Unity

La integración de Unity con BeRTA Renderer se realiza mediante un paquete específico de control OSC desarrollado para facilitar la comunicación entre ambos sistemas. Este paquete, disponible junto a BeRTA Renderer, permite enviar comandos para gestionar la escena sonora directamente desde Unity, sin necesidad de integrar la librería de audio en el motor gráfico. El diagrama de secuencia de la Figura 5 muestra el flujo de trabajo entre el paquete de Unity y BeRTA Renderer.

Para establecer la comunicación, el paquete de Unity envía un comando OSC inicial, el mensaje `/control/connect`. Este mensaje indica la dirección IP y el puerto en el que Unity espera recibir respuestas. BeRTA Renderer responde con el mismo comando, proporcionando su propia dirección IP y el puerto de escucha. Una vez completado este intercambio, Unity

queda suscrito a BeRTA Renderer y puede recibir tanto respuestas a sus solicitudes como actualizaciones de la simulación, incluidas aquellas generadas por terceras aplicaciones.

Una vez establecido el canal de comunicación, Unity debe enviar actualizar de forma continua la posición y orientación del oyente, es decir, del usuario que lleva el visor de realidad virtual. Estos datos se extraen, habitualmente, del objeto cámara principal en Unity, que sigue el movimiento del usuario en la escena.

El paquete OSC incluye scripts que, en cada frame, generan y envían los mensajes necesarios para actualizar la posición del oyente. Para ello, utilizan los comandos `/listener/location` y `/listener/orientation`, proporcionando como argumentos el identificador del oyente, sus coordenadas tridimensionales y los ángulos de rotación.

Del mismo modo, para cada fuente sonora virtual presente en la escena, Unity debe mantener actualizada su posición y orientación enviando mensajes OSC como `/source/location` y `/source/orientation`, usando como argumentos el identificador de la fuente y la posición y orientación en coordenadas tridimensionales, respectivamente.

Además, Unity puede enviar comandos adicionales para gestionar las fuentes de sonido en tiempo real. Algunos ejemplos son: `/source/loadSource`, para cargar una nueva fuente; `/source/play`, para iniciar la reproducción; y `/source/gain`, para ajustar su volumen. También se pueden gestionar detalles del renderizado como la activación o no de la simulación del ITD con el comando `/listener/enableITD`.

Muchos de los comandos OSC tienen implementada una respuesta de BeRTA Renderer, a través del comando `control/actionResult`. Con este comando se informa sobre el resultado de la acción ejecutada por parte del renderizador, si se ha llevado a cabo correctamente o si se ha producido un error, incluyendo una descripción del mismo.

BeRTA Renderer ofrece muchos otros comandos OSC, todos ellos operables en tiempo real. Estos permiten modificar parámetros de configuración de los modelos, así como cargar o eliminar distintos recursos. Todos los comandos OSC y su descripción están disponibles on-line en la documentación de la herramienta[1].

[1] https://grupodiana.github.io/BRT-Documentation/osc/

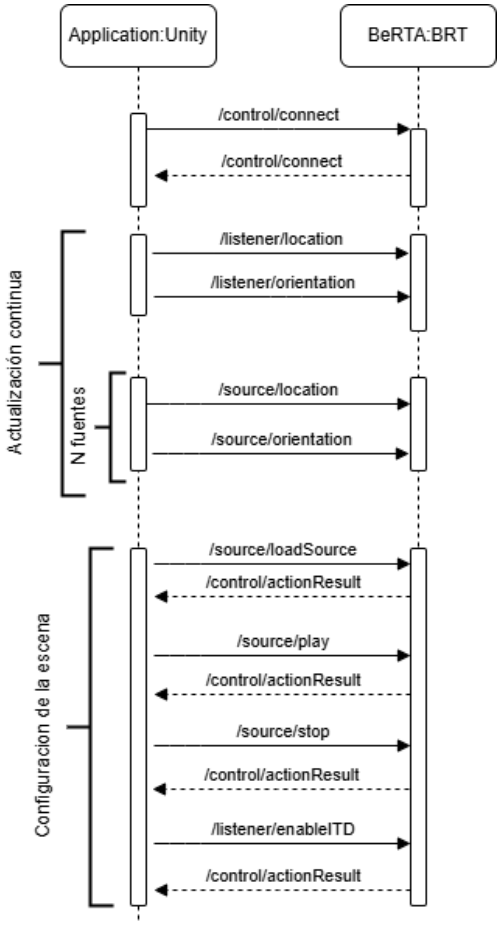

Figura 5: Diagrama de actividad del flujo de trabajo usando el paquete de Unity

El paquete OSC facilita la construcción de estos mensajes y ofrece funciones auxiliares para adaptar el sistema de coordenadas de Unity al que utiliza BRT, en caso de que sea necesario. Gracias a este enfoque, el motor gráfico Unity se mantiene completamente independiente del motor de renderizado acústico, lo que permite una arquitectura modular y escalable. Además, esta separación habilita configuraciones distribuidas en las que BeRTA Renderer puede ejecutarse en un ordenador distinto al que ejecuta la simulación visual, una ventaja especialmente útil en entornos exigentes o de investigación colaborativa.

7. Discusión y conclusión

La propuesta presentada en este trabajo combina un motor de renderizado acústico en tiempo real con una arquitectura modular y desacoplada del entorno gráfico, permitiendo su integración fluida en plataformas de realidad virtual como Unity. Esta separación entre el sistema de visualización y el de auralización no solo facilita la implementación técnica en distintos escenarios, sino que aporta una ventaja metodológica clara: la posibilidad de mantener el procesamiento sonoro

como un componente independiente, intercambiable y controlable, lo cual puede resultar útil en contextos de experimentación interactiva.

Aunque los motores de desarrollo como Unity y Unreal Engine incluyen soporte para audio espacial mediante integraciones con librerías como Steam Audio[1], Resonance Audio[2] o el Oculus Audio SDK[3], estas soluciones están orientadas principalmente a la creación de experiencias comerciales. Priorizan el rendimiento y la portabilidad, pero ofrecen escasa flexibilidad para modificar sus modelos, no están pensadas para experimentación científica y algunas han sido discontinuadas o carecen de soporte activo. Para los investigadores en psicoacústica, estas herramientas presentan importantes limitaciones: no permiten un control detallado de los modelos acústicos o binaurales, ni facilitan la reproducibilidad o la integración de nuevas simulaciones auditivas.

Existen algunas plataformas abiertas más comparables desde el ámbito científico, como Virtual Acoustics[4], que ofrece una potente infraestructura para la simulación acústica con salida binaural. Sin embargo, su enfoque está más centrado en el modelado físico del entorno, mientras que nuestra propuesta, BRT, prioriza una arquitectura modular ligera orientada al renderizado perceptual de bajo nivel, con baja latencia y a la experimentación con modelos auditivos. Esta modularidad permite construir diferentes cadenas de procesamiento acústico según el caso de uso, combinando distintos modelos de fuente, entorno y oyente. Esto resulta especialmente valioso en estudios donde se desee analizar el impacto perceptual de cada componente o personalizar la calidad de la interacción sonora. Además, gracias a su diseño como librería de cabeceras y su integración mediante OSC con motores gráficos como Unity, BRT facilita el uso en entornos de realidad virtual sin instalaciones complejas.

Otro aspecto fundamental es que BRT ha sido concebido desde el inicio como una herramienta abierta, documentada y trazable, orientada a la comunidad científica. A diferencia de muchas soluciones comerciales, que no detallan sus procedimientos internos, BRT permite conocer con precisión qué transformaciones se aplican a cada señal en cada etapa del proceso. Esta transparencia es esencial en el contexto de la investigación en acústica espacial e interacción sonora, donde la reproducibilidad exige que los métodos sean verificables y bien definidos. El uso de formatos estándar como SOFA, la documentación detallada y la estructura modular refuerzan este compromiso con la ciencia abierta y replicable.

Además, el hecho de que BRT sea una herramienta extensible ofrece una vía directa para la innovación. Los usuarios pueden modificar los modelos existentes o implementar otros nuevos, adaptando el sistema a objetivos de investigación específicos, ya sea en simulación física del entorno, en nuevos algoritmos de filtrado, o en estudios perceptuales. Esta apertura convierte a BRT no solo en una herramienta de aplicación, sino también en una plataforma para la investigación experimental en auralización y renderizado espacial.

En conclusión, la librería BRT ofrece un enfoque robusto, flexible y científicamente transparente para incorporar audio espacializado de alta calidad en entornos de realidad virtual. Su diseño modular, su arquitectura desacoplada del renderizado gráfico y su orientación hacia la ciencia reproducible lo hacen especialmente valioso para la comunidad investigadora en interacción, percepción e inmersión. Frente a soluciones cerradas o genéricas, BRT se posiciona como una herramienta que no solo permite usar audio espacializado, sino también entenderlo, controlarlo y explorarlo activamente en contextos de investigación y desarrollo.

Más allá del desarrollo técnico, el futuro de BRT también pasa por su uso como plataforma para la investigación empírica en psicoacústica. Las distintas funcionalidades que ofrece se basan en algoritmos que modelan fenómenos acústicos con implicaciones perceptuales concretas. La validación empírica de estos modelos, en términos de su eficacia para reproducir los indicios espaciales que emplea el sistema auditivo humano, constituye no solo una fase futura del desarrollo de la herramienta, sino también una línea de investigación en sí misma. En este sentido, BRT no se concibe únicamente como un motor de renderizado, sino como un laboratorio virtual para la exploración sistemática de los mecanismos de percepción espacial, abriendo así nuevas posibilidades en el estudio científico del oído humano en condiciones ecológicamente válidas y controladas.

Agradecimientos

Este trabajo se ha llevado a cabo en el contexto del proyecto SONICOM (www.sonicom.eu), acuerdo de subvención nº 101017743 dentro del programa Horizonte 2020 de la Unión Europea; y del proyecto nacional español SONIX (PID2023-152547NB-I00).

[1] https://valvesoftware.github.io/steam-audio/doc/capi/index.html

[2] https://developers.google.com/resonance-audio/develop/overview

[3] https://developer.oculus.com/documentation/native/audio-spatializer-features//

[4] https://www.virtualacoustics.org

Referencias

Algazi, V. R., & Duda, R. (2011). Headphone-based spatial sound. *IEEE Signal Processing Magazine*, *28*(1), 33–42. https://doi.org/10.1109/MSP.2010.938756

Bergstrom, I., Azevedo, S., Papiotis, P., Saldanha, N., & Slater, M. (2017). The Plausibility of a String Quartet Performance in Virtual Reality. IEEE Transactions on Visualization and Computer Graphics, 23(4), 1352–1359. https://doi.org/10.1109/TVCG.2017.2657138.

Blauert, J. (1997). Spatial hearing - the psychophysics of human sound localization. MIT Press. https://mitpress.mit.edu/books/spatial-hearing.

BRTLibrary. 2025. Grupo Diana. GitHub repository. https://github.com/GrupoDiana/BRTLibrary. Accessed March 24, 2025

Cuevas-Rodríguez, M., Picinali, L., González-Toledo, D., Garre, C., de la Rubia-Cuestas, E., Molina-Tanco, L., & Reyes-Lecuona, A. (2019). 3D Tune-In Toolkit: An open-source library for real-time binaural spatialisation. PLOS ONE, 14(3), e0211899. https://doi.org/10.1371/journal.pone.0211899.

González-Toledo, D., Molina-Tanco, L., Cuevas-Rodríguez, M., Majdak, P., & Reyes-Lecuona, A. (2023). "The Binaural Rendering Toolbox. A virtual laboratory for reproducible research in psychoacoustics". In Proc. of Forum Acusticum 2023. Turin (Italy).

González-Toledo, D., Cuevas-Rodríguez, M., Molina-Tanco, L., Picinali, L., Reyes-Lecuona, A. "New Release of the Binaural Rendering Toolbox: Introducing Version 2.0". In Proc. of Forum Acusticum Euronoise 2025. Malaga (Spain).

Reyes-Lecuona, A., Bouchara, T., & Picinali, L. (2022). Immersive sound for XR. In M. Alcañiz, M. Sacco, & J. G. Tromp (Eds.), Roadmapping extended reality: Fundamentals and applications (pp. 75-102). Springer.

Serafin, S., Geronazzo, M., Erkut, C., Nilsson, N. C., & Nordahl, R. (2018). Sonic Interactions in Virtual Reality: State of the Art, Current Challenges, and Future Directions. *IEEE Computer Graphics and Applications*, 38(2), 31–43. https://doi.org/10.1109/MCG.2018.193142628

Serafin, S. "Audio in Multisensory Interactions: From Experiments to Experiences," in Sonic Interactions in Virtual Environments (M. Geronazzo and S. Serafin, eds.), ch. 10, pp. 305–318, Cham: Springer International Publishing, 2023.

Serafin, S., Geronazzo, M., Erkut, C., Nilsson, N. C., & Nordahl, R. (2018). Sonic Interactions in Virtual Reality: State of the Art, Current Challenges, and Future Directions. *IEEE Computer Graphics and Applications*, 38(2), 31–43. https://doi.org/10.1109/MCG.2018.193142628

"SOFA (Spatially Oriented Format for Acoustics) - Sofaconventions." [Online]. Available: https://www.sofaconventions.org/mediawiki/index.php/Main_Page. [Accessed: 01/04/2025].

Välimäki, V., Parker, J. D., Savioja, L., Smith, J. O., & Abel, J. S. (2012). Fifty Years of Artificial Reverberation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(5), 1421–1448. https://doi.org/10.1109/TASL.2012.2189567.

Vorländer, M. (2020). Auralization: Fundamentals of Acoustics, Modelling, Simulation, Algorithms and Acoustic Virtual Reality (RWTH Aachen University, Ed.; second). Springer International Publishing. https://doi.org/10.1007/978-3-030-51202-6

Wefers, F. (2015). Partitioned convolution algorithms for real-time auralization. In RWTH Aachen University (Vol. 20, Issue August). http://publications.rwth-aachen.de/record/466561

Wright, M., Freed, A., "Open Sound Control: A New Protocol for Communicating with Sound Synthesizers", International Computer Music Conference, Thessaloniki, Greece, 1997.Xie, B. (2013). "Head-Related Transfer Function and Virtual Auditory Display (Second Edi)". www.jrosspub.com.

Zotter, F., & Frank, M. (2019). Ambisonics. A Practical 3D Audio Theory for Recording, Studio Production, Sound Reinforcement, and Virtual Reality. Springer. http://www.springer.com/series/8109

.