

**ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA DE TELECOMUNICACIÓN**

UNIVERSIDAD DE MÁLAGA



PROYECTO FIN DE CARRERA

*SISTEMA DE PROCESADO BINAURAL CON
SEGUIMIENTO DE MOVIMIENTO*

**INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN
SONIDO E IMAGEN**

**ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA DE TELECOMUNICACIÓN**

UNIVERSIDAD DE MÁLAGA

**Titulación: Ingeniería Técnica de Telecomunicación
*Sonido e Imagen***

Reunido el tribunal examinador en el día de la fecha, constituido por:

D./D^a. _____

D./D^a. _____

D./D^a. _____

para juzgar el Proyecto Fin de Carrera titulado:

**SISTEMA DE PROCESADO BINAURAL CON SEGUIMIENTO DE
MOVIMIENTO**

del alumno/a D./D^a. *Juan Fco. Campos Valladares*

dirigido por D./D^a. *Salvador Luna Ramírez*

ACORDÓ POR _____ OTORGAR LA
CALIFICACIÓN DE _____

Y, para que conste, se extiende firmada por los componentes del tribunal, la presente diligencia

Málaga, a _____ de _____ de _____

El/La Presidente/a

El/La Vocal

El/La Secretario/a

Fdo.: _____ Fdo.: _____ Fdo.: _____

**ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA DE TELECOMUNICACIÓN**

UNIVERSIDAD DE MÁLAGA

***SISTEMA DE PROCESADO BINAURAL CON SEGUIMIENTO DE
MOVIMIENTO***

REALIZADO POR:

Juan Fco. Campos Valladares

DIRIGIDO POR:

Salvador Luna Ramírez

DEPARTAMENTO DE: *Ingeniería de Comunicaciones*

TITULACIÓN: Ingeniería Técnica de Telecomunicación
Sonido e Imagen

PALABRAS CLAVE: Binaural, HRTF, Tracking

RESUMEN: Este proyecto diseña e implementa un sistema de procesado binaural mediante el filtrado con respuestas de la cabeza humana (HRTF) destinado a recrear un ambiente virtual tridimensional fijo en el espacio, gracias a la incorporación de un sistema de seguimiento de movimiento fijado a la cabeza del oyente.

Málaga, Febrero de 2013

Agradecimientos

En primer lugar quisiera dar las gracias a las dos fuentes principales de recursos de este proyecto: Olivier Warusfel como responsable del equipo de acústica de salas del IRCAM, y a Vicktor Budaházi por su proyecto de código libre MoveFramework.

Gracias a Salvador Luna, tutor de este proyecto, por su apoyo.

Gracias a todas las personas que han participado en las pruebas subjetivas.

Gracias a mis amigos por compartir tantas cosas conmigo.

Gracias a mi profesor Jesús de Tecnología por plantar en mí la semilla de un ingeniero, y a todos los demás profesores que creyeron en mí y me soportaron.

Gracias a mis hermanos por estar ahí y por servirme de ejemplo. A Antonio por enseñarme tantas cosas.

Gracias a mis padres por... todo. Por ser los mejores padres que se pueden tener. Sin vuestro apoyo no estaría donde estoy hoy, y sólo espero llegar hacerlo la mitad de bien con mis hijos. ¡Ya tenéis cuatro titulados! 😊

Gracias a Elena por soportar mis fricadas, por enseñarme otras nuevas, por cogerme de la mano en tantos sentidos.

Gracias.

Índice de contenidos

Lista de figuras	v
Lista de tablas	vii
CAPÍTULO 1: Introducción.....	1
1.1. Marco histórico	2
1.2. El proyecto	3
1.3. Objetivos	4
1.4. Estructura de la memoria	5
CAPÍTULO 2: Conceptos básicos.....	7
2.1. Acústica binaural	8
2.1.1. Introducción a la acústica binaural.....	8
2.1.2. Localización del sonido.....	8
2.1.3. Emulación de localización. Marco histórico.	9
2.1.4. Técnicas de emulación	11
2.2. Seguimiento de movimiento	12
2.2.1. Introducción	12
2.2.2. Tracking óptico	13
2.2.3. Tracking con sensores	17
2.2.4. Conclusión.....	20
2.3. Procesado de señal digital.....	21
2.3.1. Introducción	21
2.3.2. Filtrado digital	22
2.3.3. Fast Fourier Transform.....	24
CAPÍTULO 3: Diseño e implementación	27
3.1. Subsistema de acústica binaural.....	28
3.1.1. Cálculo de azimuth y elevación	29
3.1.2. Bases de datos de HRTF	30
3.1.3. Prototipo MATLAB	32

3.1.4. Procesado de la señal	33
3.1.5. Método overlap-add	34
3.2. Seguimiento de movimiento	35
3.2.1. Elección del dispositivo	35
3.2.2. Dispositivo PS Move	41
3.3. Implementación	43
3.3.1. Elección del lenguaje de programación	43
3.3.2. Librería FMOD	43
3.3.3. Librería MOVEFRAMEWORK	45
3.3.4. Librería FFTW	45
3.3.5. Estructura de clases	46
3.3.6. Manejador HRTF	47
3.3.7. HRTF Collection	47
3.3.8. HRTF Provider	47
3.3.9. Archivos XML	49
3.3.10. Manejador Tracking	49
3.3.11. HRTF Unit	50
3.3.12. Filtrado HRTF	51
3.4. Diseño de la aplicación visual	52
3.5. Diagramas de código fuente	53
CAPÍTULO 4: Pruebas y resultados	57
4.1. Pruebas Objetivas	58
4.1.1. Posicionamiento	58
4.1.2. Equipo de pruebas	60
4.1.3. Tiempo de adquisición y disponibilidad de los datos	60
4.1.4. Tiempo de proceso del buffer	62
4.1.5. Carga del procesador	62
4.1.6. Conclusiones	63
4.2. Pruebas subjetivas	64
4.2.1. Descripción	64
4.2.2. Resultados	66
4.2.3. Conclusiones	68

CAPÍTULO 5: Conclusiones y líneas futuras	69
5.1. Repaso de los objetivos propuestos	70
5.2. Posibles mejoras y líneas futuras	71
5.2.1. Interpolación de HRTF.....	71
5.2.2. Mejora de HRTF	72
5.2.3. Recreación condiciones del entorno.....	72
5.2.4. Plugin VST.....	72
5.2.5. Sensores propios.....	73
5.2.6. Hardware autónomo	73
Bibliografía.....	75
APÉNDICE A: Especificaciones técnicas dispositivo PS Move	81
APÉNDICE B: Cuaterniones y operaciones geométricas.....	83
APÉNDICE C: Formato WAV.....	85
APÉNDICE D: Análisis FiltroHRTF	87
APÉNDICE E: Guía de la aplicación.....	91
APÉNDICE F: Pruebas subjetivas	95

Lista de figuras

Figura 1.1. Esquema gráfico general del proyecto	6
Figura 2.1. Maniquí KEMAR.....	10
Figura 2.2. Tracking visual corporal	14
Figura 2.3. Captura facial en “El origen del planeta de los simios”, Rupert Wyatt	15
Figura 2.4. Captura facial en “Avatar”, James Cameron.....	15
Figura 2.5. Tecnología MotionScan	16
Figura 2.6. Actor John Noble en L.A. Noire	17
Figura 2.7. Helicóptero de radiocontrol con giroscopio	18
Figura 2.8. Aplicación de Realidad Aumentada.....	19
Figura 2.9. Dispositivo Wiimote	20
Figura 2.10. Procesado digital por superposición.....	22
Figura 2.11. Comparativa DFT vs FFT [24]	25
Figura 3.1. Esquema general del sistema	28
Figura 3.2. Ejemplo de cálculo del azimuth relativo de dos sonidos.	29
Figura 3.3. Composición sonora de ambiente de guerra	30
Figura 3.4. Tiempo de ejecución vs longitud de la respuesta.....	33
Figura 3.5. Convolución por FFT	34
Figura 3.6. Método Overlap-Add	35
Figura 3.7. Nintendo Wiimote.....	37
Figura 3.8. Microsoft Kinect	38
Figura 3.9. Sony PS3 Move.....	39
Figura 3.10. Dispositivos PS Move y PS3 Eye	41
Figura 3.11. Esfera iluminada del dispositivo PS Move	42
Figura 3.12. Diseño del acoplamiento del dispositivo PS Move en la cabeza del oyente... ..	42
Figura 3.13. Esquema general de implementación.....	46
Figura 3.14. Referencias angulares de azimuth y elevación.....	48
Figura 3.15. Disposición de muestras multicanal en array.....	51
Figura 3.16. División de muestras para filtrado y recombinación.....	51
Figura 3.17. Estructura de formularios de la aplicación.....	52
Figura 3.18. Diagrama de secuencia de carga de colección	53

Figura 3.19. Diagrama de secuencia de registro de dispositivo	54
Figura 3.20. Diagrama de secuencia de unidad HRTF	55
Figura 3.21. Diagrama de secuencia de actualización de rotación.....	56
Figura 4.1. Cobertura de la cámara PS Eye.....	59
Figura 4.2. Captura de diagnóstico FMOD Profiler	63
Figura 4.3. Diagrama de puntuación de azimuth.....	66
Figura 4.4. Diagrama de puntuación en elevación	67
Figura B.1. Representación gráfica con Quaternion	83
Figura B.2. Cálculo de ángulos desde Quaternion	84
Figura C.1. Ejemplo de archivo WAV [50]	86
Figura E.1. Ventana principal.....	91
Figura E.2. Ventana de carga de colección	92
Figura E.3. Ventana de visualización de PS Move	92
Figura E.4. Ventana de unidad de HRTF	93

Lista de tablas

Tabla 3.1. Ejemplo de estructura de colección de HRTF	31
Tabla 3.2. Nomenclatura archivos WAV IRCAM	32
Tabla 3.3. Comparativa de controles de videoconsolas.....	40
Tabla 3.4. Formato de archivo de Colección HRTF XML propio	49
Tabla 4.1. Comparativa disposición de respuestas HRTF	58
Tabla 4.2. Características equipo de pruebas	60
Tabla A.1. Características Técnicas PS Move.....	81
Tabla A.2. Desglose hardware PS Move	82
Tabla A.3. Características Técnicas PS Eye.....	82
Tabla C.1. Estructura de archivo WAV.....	85
Tabla F.1. Resultados Sujeto 1	95
Tabla F.2. Resultados Sujeto 2	95
Tabla F.3. Resultados Sujeto 3	96
Tabla F.4. Resultados Sujeto 4	96
Tabla F.5. Resultados Sujeto 5	96
Tabla F.6. Resultados Sujeto 6	96
Tabla F.7. Resultados Sujeto 7	97
Tabla F.8. Resultados Sujeto 8	97
Tabla F.9. Resultados Sujeto 9	97

CAPÍTULO 1: Introducción

En este capítulo se hace una introducción del proyecto analizando su marco histórico y la necesidad de la que surge. Se presenta un resumen de los objetivos del mismo y se adelantan algunas de las técnicas que se van a usar para su implementación. Se comenta la estructura de esta memoria y se presenta un esquema básico general del sistema.

1.1. MARCO HISTÓRICO

Una de las misiones principales de la tecnología es la de estar al servicio de las personas. En el ámbito audiovisual, el desarrollo tecnológico ha perseguido crear experiencias cada vez más reales, llegar cada vez más al espectador, provocar más sensaciones. Ese desarrollo ha tenido como protagonista y abanderada a la industria cinematográfica.

Desde las primeras proyecciones en blanco y negro hasta las modernas películas en tres dimensiones, el reto siempre ha sido proporcionar al usuario una experiencia para sus sentidos.

A finales del siglo XIX, en los orígenes del cine[1], una simple secuencia de imágenes de un tren que parecía abalanzarse sobre los espectadores provocó reacciones de pavor. Una experiencia visual hasta entonces desconocida creó la falsa sensación de estar realmente delante de ese tren. Poco después surgieron los primeros efectos especiales, artimañas rudimentarias para crear experiencias mágicas.

No fue hasta unas décadas después cuando se empezó a grabar e incluir el sonido en las películas, dando comienzo a lo que se conoce como *cine sonoro*. Hasta entonces, las proyecciones se acompañaban mayormente de música en directo. Por aquel entonces el cine se estaba convirtiendo ya en una de las industrias más potentes del siglo XX.

Desde sus orígenes, el cine ha ido de la mano (y muchas veces a la cabeza) del desarrollo tecnológico, atendiendo a nuevas demandas del público, creando nuevos efectos cada vez más sorprendentes. En el ámbito visual, los gráficos por ordenador han permitido recrear escenas imposibles y personajes extraordinarios, cada vez con más detalle. Uno de los saltos cualitativos más recientes es la implantación de las proyecciones tridimensionales en los cines comerciales. La inmersión del espectador es mucho mayor gracias a esta tecnología.

En el ámbito sonoro, la evolución ha sido más discreta. Los sistemas de audio que se usan en los cines han ido creciendo en complejidad y sobretodo en número de canales (altavoces en la sala). La inmersión del espectador en este aspecto depende de lo real que parezcan los sonidos que escucha, y gracias a que el ser humano es capaz de localizar los

sonidos en el espacio, se hace indispensable emular esa sensación reproduciendo los sonidos desde distintos altavoces situados en la sala.

1.2. EL PROYECTO

Es precisamente en ese intento de mejorar la inmersión auditiva en el que se centra este proyecto. La misión del mismo es la de proporcionar una nueva experiencia sonora, una sensación más real. Del mismo modo que las proyecciones tridimensionales intentan recrear la visualización de imágenes reales, este proyecto se basa en los métodos de la acústica binaural para emular un escenario tridimensional virtual que consiga provocar la sensación de ser real en el oyente.

El avance tecnológico en el que se basa este proyecto supone una gran evolución en la forma de reproducir sonidos, con múltiples posibles aplicaciones. Desde la mencionada potenciación de la inmersión de un espectador de un contenido audiovisual, hasta la mejora de la calidad de vida de las personas dependientes; como los invidentes, o personas con deficiencias auditivas.

El cerebro humano es capaz de analizar el sonido captado por el sistema auditivo y estimar la posición del mismo en el espacio. Hay muchos factores que influyen en este proceso. El entorno y el cuerpo (en especial la cabeza) actúan como un filtro para el sonido, siendo el cerebro capaz no solo de captar el sonido en sí, sino de interpretar esos cambios y estimar desde qué dirección le llega. Utilizando las técnicas descritas con detalle en los capítulos siguientes, se persigue recrear esas condiciones para simular la sensación de la fuente sonora en una posición concreta del espacio que rodea al oyente.

Actualmente existen sistemas de audición binaural con auriculares y utilizando las mismas técnicas de procesado que este proyecto, sin embargo, todas tienen la misma limitación: si el oyente se gira, se pierde por completo la sensación, ya que el sonido se moverá con él. Por tanto, un paso definitivo en la recreación del entorno virtual es que el sistema tenga en cuenta el movimiento de la persona con respecto a la fuente sonora. Esto es, la posición y rotación relativas al origen del sonido.

Este proyecto añade esa información al sistema utilizando un dispositivo de seguimiento de movimiento, permitiendo así que éste sea capaz de mantener la posición virtual de la fuente sonora con respecto al oyente, mientras éste se mueve.

1.3. OBJETIVOS

La misión principal del proyecto es conseguir provocar en el oyente una sensación de inmersión lo más realista posible mediante la creación de un entorno sonoro virtual tridimensional. Este objetivo se consigue aplicando dos técnicas:

Procesado binaural: Simula la transformación que sufre el sonido hasta llegar a cada oído del oyente. Este proceso se reproduce usando auriculares. El sistema debe ser capaz de leer un archivo de audio, procesarlo utilizando un filtro basado en la respuesta en frecuencia de la cabeza humana[2] (HRTF, *Head Related Transfer Function*), y reproducirlo. Además, se persigue realizar el proceso en tiempo real. Para ello, se desarrollará un sistema de procesamiento de señal basado en el filtrado digital, que dividirá la señal de entrada en bloques pequeños y los filtrará haciendo uso de convolución, para luego recomponer la señal de salida de forma que ésta se reproduzca de forma continua. Se contará con una librería externa que se hará cargo de la lectura de los ficheros y la gestión del driver de la tarjeta de sonido.

Seguimiento de movimiento: Mantiene el control de la posición y rotación del oyente con respecto a la fuente, con el objeto de que el sonido se mantenga fijo en el espacio. El sistema debe ser capaz de realizar un seguimiento de la rotación de la cabeza y entregar esos datos al sistema de procesamiento binaural con suficiente velocidad como para garantizar una experiencia realista. Para lograr este objetivo, se utilizará como sensor de movimiento el controlador de una videoconsola, aprovechando su arquitectura ya cerrada y el fácil acceso a documentación debido a su uso extendido.

Una de las limitaciones a tener en cuenta del procesamiento binaural es que depende en gran medida de las proporciones físicas de la persona . Por ello se contará con una colección de respuestas en frecuencia de la cabeza para elegir la que más se adecue al oyente, con el objetivo de proporcionar la sensación más real posible. Dicha colección se compone de un

número de respuestas correspondientes a distintos ángulos de elevación y azimut. Debe haber suficientes muestras como para garantizar una experiencia realista.

1.4. ESTRUCTURA DE LA MEMORIA

Este proyecto se estructura en cinco capítulos en los que se analizarán y comentarán distintos aspectos del mismo.

- **Introducción**

En este capítulo se ha visto un breve marco histórico y tecnológico del proyecto con el fin de ubicar al lector y dar una primera pincelada sobre la motivación y los objetivos del sistema desarrollado.

- **Conceptos básicos**

Una vez entendida la misión del proyecto, en el segundo capítulo se dan las bases teóricas del mismo. Se repasan los conceptos básicos de la acústica binaural y el seguimiento de movimiento, y se presentan los cimientos sobre los que se ha construido este sistema.

- **Diseño e implementación**

En este capítulo se detalla cómo se ha diseñado el sistema para llegar a esos objetivos marcados en la introducción. Se presenta primero el sistema de procesado binaural. Luego, el sistema de seguimiento de movimiento, y finalmente, se detalla la integración de los mismos.

- **Pruebas y resultados**

Tras la implementación, el siguiente paso es verificar los objetivos requeridos para el proyecto. En este capítulo se presenta un conjunto de pruebas objetivas y subjetivas que establecen un criterio para comprobar si se han alcanzado las metas previstas.

- **Conclusión**

A modo de conclusión, en este capítulo se ha realizado una reflexión final valorando el cumplimiento de los objetivos marcados al inicio del proyecto, comentando las dificultades que se han presentado en su implementación. También se presentan las posibles aplicaciones y líneas de ampliación futuras del sistema desarrollado.

En la Figura 1.1 se presenta un esquema gráfico general del proyecto en el que aparecen los distintos bloques que lo componen y que serán descritos en detalle en los próximos capítulos.

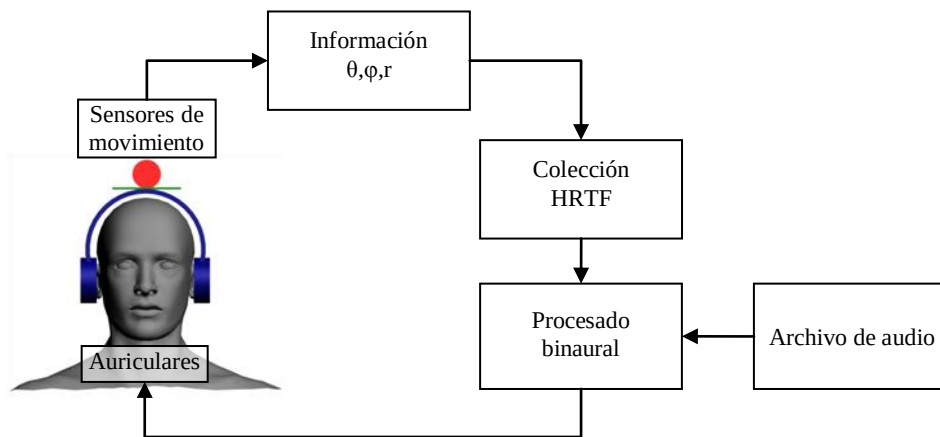


Figura 1.1. Esquema gráfico general del proyecto

CAPÍTULO 2: Conceptos básicos

En este capítulo se hace un repaso general a los conceptos básicos que forman los pilares sobre los que se construye este proyecto. Se hace una introducción a la acústica binaural y al seguimiento de movimiento, además de presentar un marco histórico y tecnológico de su evolución hasta el momento actual. En el último apartado se hace un repaso teórico al procesado digital de señal, una parte fundamental de este sistema.

2.1. ACÚSTICA BINAURAL

2.1.1. INTRODUCCIÓN A LA ACÚSTICA BINAURAL

Para hacer una introducción a la acústica binaural, primero es necesario entender que la audición en el ser humano es un proceso complejo. En una situación en la que una fuente sonora emite un sonido y una persona lo oye, lo que el cerebro de la persona recibe no es exactamente el sonido que la fuente ha generado. El sonido sufre multitud de cambios hasta llegar al cerebro. Algunos de esos cambios los produce el entorno que rodea a ambos elementos (distancia, geometría y materiales del recinto, otros sonidos existentes, etc.), y otros cambios los produce la propia persona (morfología propia de la persona, capacidad auditiva, etc.). Todos esos elementos modifican el sonido, y algunas de esas modificaciones son analizadas por el cerebro para obtener información complementaria sobre el sonido que le ha llegado.

La psicoacústica[3] estudia la relación que existe entre el estímulo físico que se produce en el sistema auditivo y la respuesta psicológica que éste provoca. De entre todas esas interpretaciones que el cerebro hace a partir de los estímulos físicos y las correspondientes áreas de estudio de la psicoacústica, la acústica binaural, en el sentido que engloba este proyecto, se centra en la capacidad del cerebro de localizar el punto desde donde se está emitiendo el sonido.

2.1.2. LOCALIZACIÓN DEL SONIDO

El cerebro interpreta los sonidos usando la información que llega de los dos oídos[4]. Esa información es distinta, ya que los oídos están situados en posiciones distintas con respecto a la fuente que origina el sonido. Así, la onda sonora alcanzará a cada uno a un tiempo distinto. Por lo tanto, habrá una diferencia de fase y de amplitud entre las dos señales. Esas diferencias son analizadas por el cerebro para obtener la información en el plano lateral, correspondiente a la izquierda y la derecha.

El oído externo (pabellón auricular) tiene una forma concreta[5] que permite una modificación del sonido que depende de la dirección de la que éste proviene. Según esa

dirección, distintos filtros resonantes se activan, añadiendo ciertas resonancias en el espectro de la señal que el cerebro detecta para interpretar la dirección de la que proviene. Además, la forma de la cabeza humana, al no ser simétrica, también realiza un filtrado de la señal que depende de la dirección. Estos efectos combinados ayudan a que el cerebro localice la dirección del sonido en el resto de planos (delante, detrás, arriba, abajo).

2.1.3. EMULACIÓN DE LOCALIZACIÓN. MARCO HISTÓRICO.

Una vez se ha entendido cómo el cerebro funciona conjuntamente con el sistema auditivo para localizar el sonido en el espacio, se pueden caracterizar todos esos procesos y realizar un sistema que modele la transformación que sufre el sonido. Ese sistema podría generar un sonido y emular la transformación sufrida en una dirección concreta, consiguiendo que aunque el sonido no se encuentre en esa dirección, el cerebro lo interprete de esa forma.

A lo largo de la historia ha habido interés en la grabación y posterior recreación de diversas obras sonoras. En 1881[6] se usó el primer sistema rudimentario de grabación binaural, el *Téâtrophone*, el cual consistía en una matriz de micrófonos situados en teatro de ópera *Palais Garnier*, que grababan el sonido y era enviado a los oyentes por medio de la línea telefónica y escuchado usando unos auriculares especiales.

En 1931[7], un ingeniero llamado Alan Blumlein desarrolló lo que él llamó “sonido binaural” sentando las bases de lo que ahora se conoce como sonido estéreo. Durante casi toda su historia, la grabación y escucha binaural han sido una práctica poco extendida debido a su complejidad y coste. Hasta que la evolución tecnológica no ha permitido implementarlas de una forma accesible, ha estado al alcance sólo de unos pocos. No fue hasta la década de los 70 cuando se empezaron a utilizar estas técnicas, gracias a la aparición de los métodos de procesado digital de señal explicados en detalle en los siguientes apartados. Se empezaron entonces a distribuir las primeras grabaciones binaurales de discos de música y de conciertos.

En 1972[8] la compañía Knowles Electronics desarrolló un modelo físico de una cabeza humana destinado a grabaciones binaurales, llamado KEMAR (*Knowles Electronics Manekin for Acoustic Research*). El modelo se ha convertido en un estándar en este ámbito y hoy en día se sigue usando. En la Figura 2.1 se visualiza el maniquí KEMAR.



Figura 2.1. Maniquí KEMAR

Los sonidos binaurales generados (no grabados) se consiguieron mediante la llamada HRTF (*Head Related Transfer Function*). La HRTF modela la respuesta en frecuencia de la cabeza humana, incluyendo todos los efectos combinados mencionados en el anterior apartado sobre localización de sonido. Para crear una HRTF, se mide la respuesta al impulso con micrófonos ubicados o apuntando hacia el conducto auditivo, ya sea de una persona real o de un modelo artificial de una cabeza humana. Se mide la respuesta para cada oído y para cada dirección, obteniendo finalmente una base de datos de HRTF. Cualquier sonido puede ser filtrado con esa respuesta y por lo tanto emularse su posicionamiento.

En 1985 se empezó a emitir un programa de radio llamado “Audiophile Audition” en el que se retransmitía sonido binaural, que pasó tras 13 años en antena a ser un portal de

internet dedicado a la acústica binaural hoy día todavía accesible[9]. En internet existen multitud de portales dedicados a este ámbito, que recopilan grandes cantidades de sonidos simulados, grabaciones y experimentos.

Gracias a los grandes progresos que se han hecho en el procesamiento de audio en los últimos años, han surgido numerosas aplicaciones de estas técnicas. Cabe destacar su uso en los videojuegos. Títulos como *Zen Bound (Secret Exit)*, *Aves (Action=Reaction Games)* o *Papa Sangre (Somethin' Else)* utilizan estas técnicas en sus juegos como mecánica principal.

2.1.4. TÉCNICAS DE EMULACIÓN

Como se ha visto en el apartado anterior, existen básicamente dos formas de recrear la experiencia de la localización al reproducir un sonido, que también componen los dos ámbitos principales de aplicación de la acústica binaural: La grabación y la síntesis.

La grabación consiste en recoger el sonido en directo de una forma en la que luego se pueda reproducir en las condiciones más parecidas a la situación de estar realmente delante de ese sonido en el momento en el que se originó. Para ello, basándose en los conocimientos de acústica binaural, se intenta grabar el sonido justo desde donde se va a reproducir, normalmente en unos auriculares. Para ello, se recrean las condiciones usando un maniquí con la forma de una cabeza humana, y se graba el sonido que incide sobre su conducto auditivo. Ese sonido ya tendrá incorporadas todas las modificaciones referentes a la forma de la cabeza y a los filtros selectivos del pabellón auditivo. Así, al reproducir el sonido en unos auriculares, se intenta conseguir que el sonido continúe su camino por donde lo dejó, simulando las condiciones exactas en el oído de una persona. El único problema existente con esta técnica es que la forma de la cabeza del maniquí no será la misma que la del oyente, ni tampoco sus pabellones auditivos, por lo tanto la sensación no será tan real como si el oyente estuviera allí mismo, ya que esas modificaciones son extremadamente precisas.

Por otro lado, la síntesis persigue el mismo objetivo, pero generando el sonido, o partiendo de un sonido que no ha sido grabado con ninguna técnica binaural. Se parte de

un sonido, en principio, tal y como lo reproduciría una fuente sonora. Por lo tanto, hay que añadirle esas modificaciones que sufriría hasta llegar al conducto auditivo. Para ello, se hace uso de la anteriormente citada HRTF, la respuesta en frecuencia de la cabeza humana. Esa respuesta pregrabada se modela como filtro y se aplica al sonido, resultando el mismo efecto que como si se hubiera grabado con el maniquí. La única diferencia existente es que la grabación en directo contiene también los efectos de la sala en la que se está emitiendo el sonido, y la síntesis no. Esto, aunque no influye demasiado en la localización, es un componente importante para dotar de realismo el sonido que se está escuchando.

Ambas técnicas se basan en el mismo principio, pero se han comentado por separado para ilustrar las principales diferencias entre ellas.

2.2. SEGUIMIENTO DE MOVIMIENTO

2.2.1. INTRODUCCIÓN

El seguimiento de movimiento[10] o captura de movimiento es la técnica de reconocer y grabar los movimientos de objetos o personas para su uso en multitud de aplicaciones, como pueden ser la investigación científica, la robótica, el rodaje de películas o videojuegos.

Una de las primeras formas rudimentarias de capturar el movimiento fue la técnica denominada rotoscopia[11], patentada por Max Fleischer en 1917, el cual la usó en una serie de dibujos animados llamada “*Out of the Inkwell*”. La técnica consiste en proyectar los fotogramas de una grabación sobre el papel o lienzo, para copiar así los movimientos naturales. Poco después, estudios como Warner Bros o Disney empezaron a usarla produciendo sus más famosas películas de animación usando esta técnica, como por ejemplo “Blancanieves y los Siete Enanitos” y posteriores. Otro ejemplo del uso de esta técnica es la famosa saga “La Guerra de las Galaxias”, donde entre otros elementos, la rotoscopia se usó para recrear los famosos sables de luz.

No fue hasta la década de los 80[12][13] cuando la tecnología empezó a permitir la captura real del movimiento con los primeros sensores, que no eran más que

potenciómetros que se fijaban en las partes móviles y mandaban una señal analógica que era capturada por microcontroladores. Muy poco después se desarrollaron las primeras técnicas de tracking con marcas visuales, utilizando software de análisis de imágenes.

Las técnicas de captura de movimiento evolucionaron muy deprisa y se empezaron a utilizar pronto en animación por ordenador. En 1992, en la producción de la película “Jurassic Park”, un equipo de ingenieros construyó pequeñas marionetas de un dinosaurio conectadas a un ordenador con la que se capturaban movimientos, para posteriormente reproducirlos en robots de 6m de altura para grabar las escenas. En 1997 se produjo la primera película CGI (*Computer Generated Imagery*, o Imagen Generada por Ordenador) usando exclusivamente captura de movimiento, “Simbad: Beyond the Veil of Mists”. La mayoría de películas con imagen generada por ordenador desde entonces usa algún tipo de captura de movimiento para sus efectos.

Sin embargo, algunos de los primeros intentos de recrear personajes virtuales usando captura de movimiento no funcionaron debido a que la tecnología aún no estaba lo suficientemente desarrollada. El seguimiento de movimiento con sensores no era capaz de capturar movimientos con suficiente naturalidad. Un ejemplo de ello es la anécdota sucedida durante el rodaje de la trilogía de “El Señor de los Anillos”, en la que, a pesar de haber usado captura de movimiento con sensores grabando a un actor para recrear al personaje *Gollum*, la información se descartó por no resultar suficientemente natural, y los animadores tuvieron que recurrir al método de la rotoscopia para producir las animaciones del personaje en la película.

Con la evolución tecnológica de la informática y el crecimiento exponencial en la potencia gráfica de los ordenadores, las técnicas de captura de movimiento se han diversificado y especializado para cada tipo de aplicación. A continuación se comentan las principales técnicas de seguimiento de movimiento que se utilizan en la actualidad.

2.2.2. TRACKING ÓPTICO

Este tipo de técnica se basa en la captura por medio de cámaras de vídeo de marcas visuales implantadas en el objeto que se está moviendo. Requiere de un potente software

de análisis de imagen, ya que para obtener una captura precisa es necesario el uso de varias cámaras al mismo tiempo, que obtienen imágenes desde distinto ángulo y el software triangula la posición de las marcas.

Este método es el más usado hoy en día dada la actual potencia gráfica de los ordenadores, ya que todo el trabajo lo hace el software, siendo los sensores normalmente simples pegatinas o esferas de plástico pegadas al cuerpo del actor (Figura 2.2).

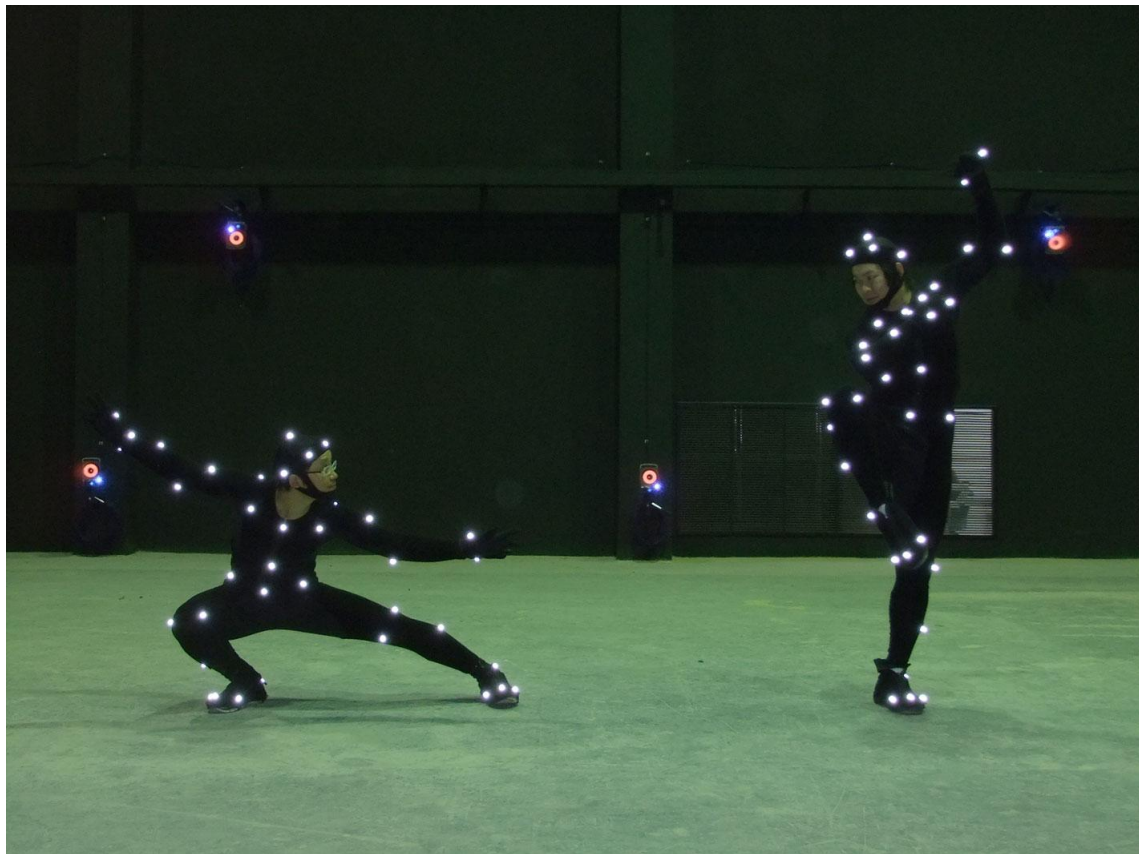


Figura 2.2. Tracking visual corporal

Esta técnica tiene su mayor avance en la captura de las expresiones faciales, las cuales se producen en un rango de movimiento mucho menor, por lo que se necesita más precisión. Para ello se colocan pequeñas marcas por la cara del actor con el fin de captar cada detalle de su interpretación. Películas como “El origen del planeta de los simios” (Figura 2.3) o “Avatar” (Figura 2.4) se basaron en esta tecnología para recrear sus personajes virtuales.



Figura 2.3. Captura facial en “El origen del planeta de los simios”, Rupert Wyatt



Figura 2.4. Captura facial en “Avatar”, James Cameron

El tracking visual facial ha alcanzado uno de sus mayores logros en la producción del videojuego L.A. Noire[14]. En el videojuego, el jugador encarna a un detective el cual debe investigar casos e interrogar a sospechosos, analizando sus expresiones faciales para descubrir si mienten o dicen la verdad. Para lograr el objetivo de crear esas expresiones tan

detalladas, el estudio RockStar hizo uso de una tecnología llamada MotionScan (Figura 2.5) desarrollada por la compañía australiana Depth Analysis.

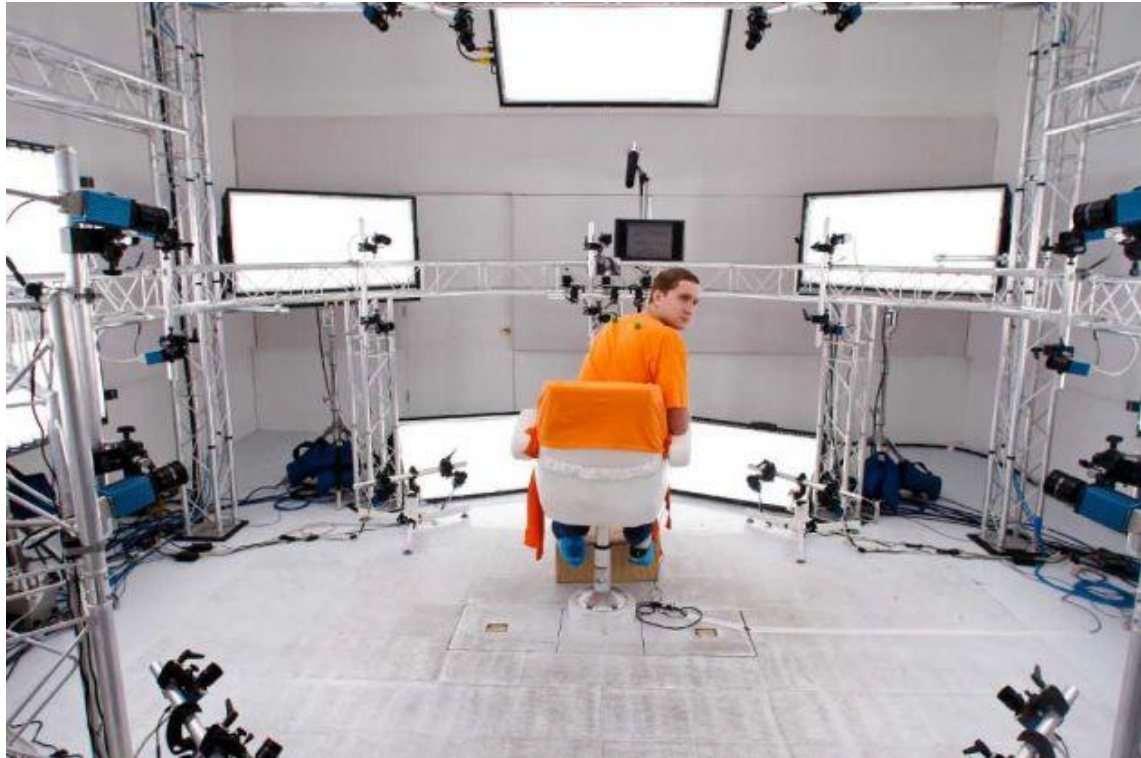


Figura 2.5. Tecnología MotionScan

Esta tecnología emplea 32 cámaras de alta definición para capturar hasta el más mínimo detalle de la interpretación del actor. En el desarrollo de L.A. Noire participaron más de 50 actores[15], entre ellos, Aaron Staton (Figura 2.5) y John Noble (Figura 2.6).

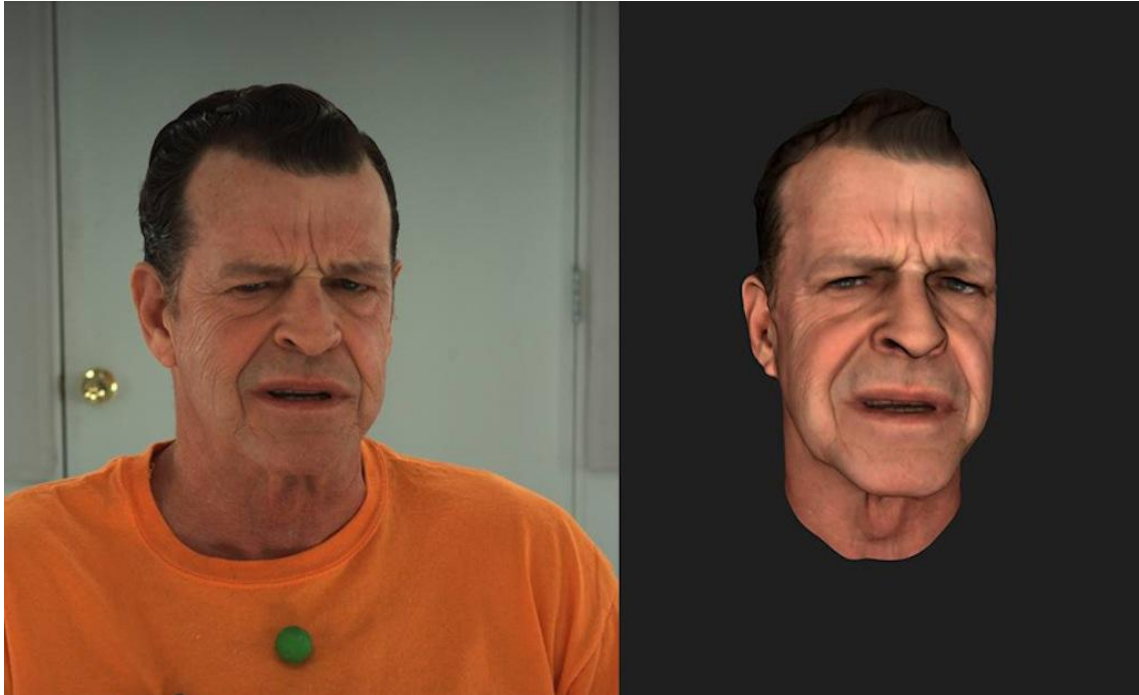


Figura 2.6. Actor John Noble en L.A. Noire

2.2.3. TRACKING CON SENSORES

Este conjunto de técnicas se basa en colocar sensores en los puntos clave del objeto que se va a capturar. Dichos sensores envían la información de la posición al servidor, que recoge los datos y los procesa. No hacen falta cámaras por tanto. Los sensores que se suelen utilizar son acelerómetros (miden aceleración), giroscopios (miden rotación), magnetómetros (miden orientación magnética).

Este tipo de tracking se suele usar en situaciones que requieren gran precisión, y no necesitan medir muchos puntos distintos, como por ejemplo en vehículos, aeronaves y en robótica.

En la Figura 2.7 se visualiza un helicóptero de radiocontrol que usa un sensor de giroscopio para mantenerse estable en el aire.



Figura 2.7. Helicóptero de radiocontrol con giroscopio

Estos sensores también se han extendido entre los teléfonos móviles, ya que gracias a ellos se puede tener acceso a la rotación, aceleración y orientación del dispositivo, dando lugar a un sinfín de aplicaciones y videojuegos muy populares. Cabe destacar su uso en aplicaciones de realidad aumentada, en la que se combinan todos los sensores además de la información de GPS y el análisis de la imagen captada por la cámara para proporcionar al usuario información adicional sobre lo que tiene en su pantalla. En la Figura 2.8 se visualiza la pantalla de un teléfono móvil en la que se ha combinado la imagen procedente de la cámara con información sobre los restaurantes que aparecen frente al usuario.

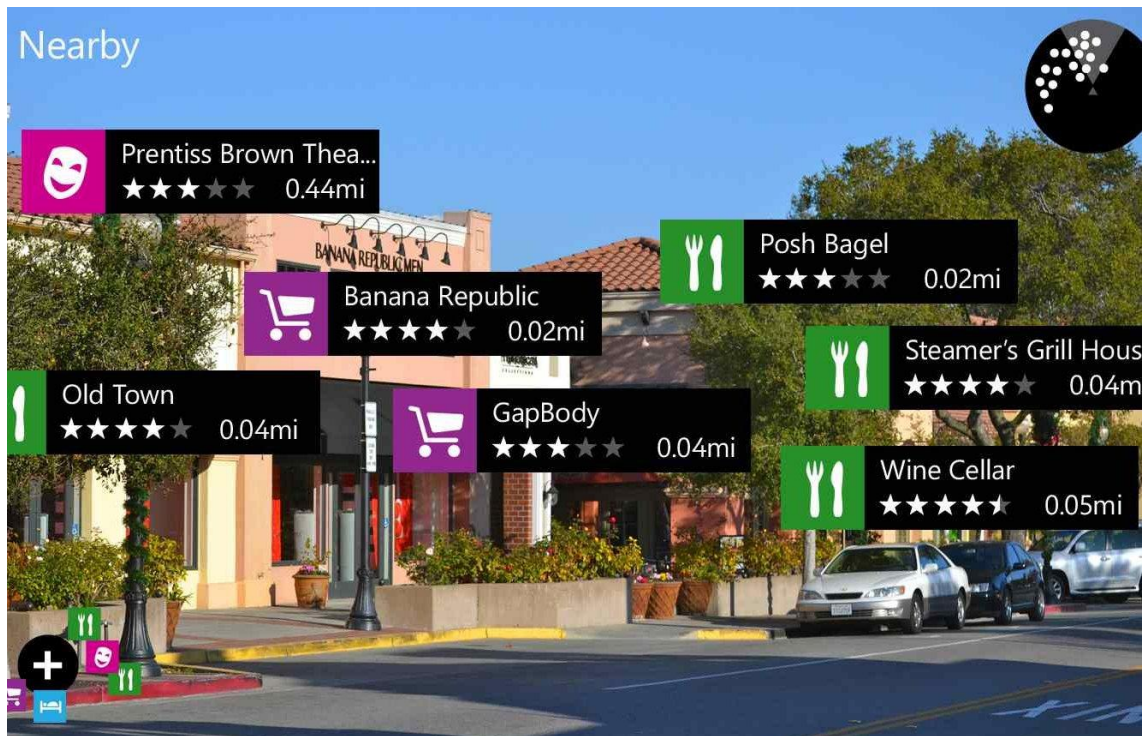


Figura 2.8. Aplicación de Realidad Aumentada

Por último, cabe destacar el uso de este tipo de sensores en los controles remotos de videoconsolas. La videoconsola de Nintendo, Wii[16], fue la pionera en incorporar este tipo de sensores y desarrollar videojuegos específicos para su uso. Su controlador Wii Remote (Figura 2.9) puede seguir movimientos en un plano tridimensional, además de actuar como puntero apuntando a la pantalla. Uno de los primeros títulos que salieron al mercado fue Wii Sports, con más de 40 millones de copias vendidas en apenas dos años. En el videojuego, se podía usar el mando para practicar golf, tenis o boxeo, con una sorprendente precisión.

Poco después, los competidores de Nintendo empezaron a apostar por esta nueva práctica de seguir el movimiento del usuario mediante el control remoto. Sony incorporó estos sensores a su control remoto Dualshock 3[17] y posteriormente a su dispositivo PS Move, un controlador parecido al Wii Remote.



Figura 2.9. Dispositivo Wiimote

2.2.4. CONCLUSIÓN

Como se ha visto en este apartado, el seguimiento de movimiento ha jugado un papel muy importante en muchos campos que van desde la investigación médica hasta el ocio. Ha supuesto un ámbito de investigación y desarrollo para muchas empresas que han innovado en la tecnología, y sus aplicaciones siguen expandiéndose hacia nuevos estudios y productos.

2.3. PROCESADO DE SEÑAL DIGITAL

2.3.1. INTRODUCCIÓN

El procesamiento de señales digitales[18][19][20] es la manipulación matemática de una señal para modificarla o mejorarla en algún sentido. Se basa en operaciones con señales discretas ya sea en el dominio temporal o en el dominio frecuencial.

El procesamiento digital se lleva a cabo en sistemas basados en microprocesadores, que mediante un hardware concreto y un juego de instrucciones en su software realizan dichas operaciones.

El objetivo más habitual del DSP es el de medir, filtrar o comprimir señales continuas analógicas reales. Para ello uno de los pasos más importantes es la conversión analógico-digital de dichas señales haciendo uso de técnicas de muestreo.

Uno de los mayores beneficios del DSP es la facilidad de uso de las transformadas de señales. Al realizar, por ejemplo, la Transformada de Fourier Discreta, se obtiene la señal en un dominio en el cual el análisis es más sencillo y eficaz. Los procesados más habituales son los filtros (paso bajo, paso alto, paso banda, eliminación de banda, eliminación de ruido, etc.).

El DSP tiene un ámbito de aplicación muy extendido, ya que engloba el procesamiento de cualquier tipo de señal digital. De entre sus aplicaciones más extendidas se encuentran el procesamiento de imágenes y el de sonido. En el ámbito de este proyecto, se trata únicamente el procesamiento de señales de audio.

Aunque los primeros sistemas DSP se empezaron a utilizar en los años 70, no fue hasta unos años más tarde cuando se empezó a aplicar al procesamiento del audio, con dos fines principales: la música y la voz. Uno de los primeros procesamientos que se llevaron a cabo con la señal de música fue la reverberación. La reverberación tiene una gran influencia en el oyente cuando se trata de la música. Las pistas de audio se grababan en estudio y posteriormente, en el proceso de mezcla, se añadía reverberación para simular algunos ambientes de escucha ideales.

La mayoría de las técnicas de procesamiento digital de señal se basan en la estrategia de la superposición, que consiste en dividir un problema en problemas más pequeños. Esto da pie a los llamados sistemas lineales, un conjunto de métodos y reglas para procesar señales que permiten dividirlos en componentes más simples, procesar cada componente por separado y volver a recomponer la señal, siendo el mismo resultado que si se procesara la señal de origen completa. Esta potente metodología es uno de los pilares fundamentales del DSP. En la Figura 2.10 se visualiza el proceso descrito anteriormente.

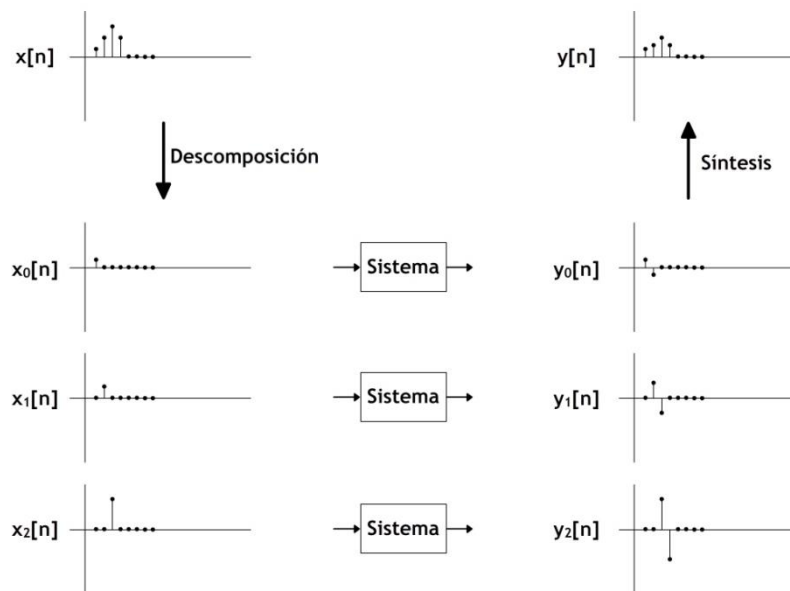


Figura 2.10. Procesado digital por superposición

2.3.2. FILTRADO DIGITAL

Para realizar el mencionado proceso de filtrado digital, se hace uso de un conjunto de algoritmos matemáticos que se ejecutan en la unidad de procesamiento del microprocesador. Uno de los más básicos es la convolución.

Los sistemas lineales[21] se caracterizan por su respuesta al impulso. Es la respuesta del sistema ante una entrada que es un impulso unitario. Teniendo en cuenta lo visto anteriormente de los sistemas lineales, esta respuesta define cómo se comporta el sistema ante cualquier entrada. Para obtener la respuesta del sistema ante una señal completa de

entrada, se realiza el proceso de convolución[22], cuya definición se presenta en la Ecuación 2.1 a continuación:

$$y[i] = \sum_{j=0}^{M-1} h[j]x[i-j] \quad (2.1)$$

En la ecuación, $h[n]$ es la respuesta al impulso del sistema de tamaño M .

Otra de las herramientas más potentes en el procesamiento de señal digital es la Transformada de Fourier Discreta, una técnica matemática que descompone una señal en otras señales más sencillas: sinusoides. La razón y la gran utilidad de descomponer la señal en sinusoides radican en que, en un sistema lineal, una entrada sinusoidal produce siempre una salida sinusoidal, en la que la frecuencia se mantiene constante. Únicamente cambian la amplitud y la fase. Estos cambios de amplitud y fase son los llamados coeficientes. Esta poderosa herramienta permite la representación de las señales en un dominio distinto al temporal: El dominio frecuencial.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N}kn} \quad k = 0, \dots, N-1 \quad (2.2)$$

La Transformada Discreta de Fourier (Ecuación 2.2.) es una de las herramientas más importantes en el procesamiento de señal digital. Permite realizar análisis espectrales tanto de señales como de sistemas. Esto resulta de gran utilidad en el ámbito de las señales de audio, ya que es muy común caracterizar las señales por su espectro. Señales graves tienen más componentes de baja frecuencia, y señales agudas tienen más componentes de alta frecuencia.

Otra de las aplicaciones más beneficiosas de la transformada es la posibilidad de realizar la convolución en el dominio frecuencial. En el dominio de la frecuencia, la operación de convolución se transforma en una simple multiplicación de las señales. Esto simplifica mucho los algoritmos de filtrado, ya que, en un microprocesador, resulta mucho más sencillo realizar el proceso multiplicando las señales que realizando la convolución, como se analizará en el siguiente capítulo.

2.3.3. FAST FOURIER TRANSFORM

La FFT (Fast Fourier Transform)[23] es uno de los algoritmos más complejos realizados en el ámbito del procesamiento digital de señal. Es un método para calcular la Transformada Discreta de Fourier que reduce drásticamente el tiempo de proceso en computación. Se basa en la descomposición de una señal de N puntos en N señales de un punto, la obtención de los N espectros, y un complejo proceso de síntesis del espectro final a partir de los N espectros diferentes.

La eficiencia del algoritmo de cálculo de la FFT hace que el tiempo computacional del cálculo se reduzca. Esto se hace imprescindible para el procesamiento de señales, sobretodo en señales largas. Mientras que el cálculo de una DFT de una señal de N muestras por el proceso tradicional requiere N^2 operaciones, el cálculo utilizando un algoritmo de FFT requiere $N \cdot \log_2(N)$. Por ejemplo, para una señal de 1024 muestras, la DFT normal realizaría 1048576 operaciones, mientras que la FFT realizaría 10240. La Figura 2.11. muestra una gráfica que compara el tiempo de procesamiento de los dos algoritmos para los correspondientes tamaños de señal.

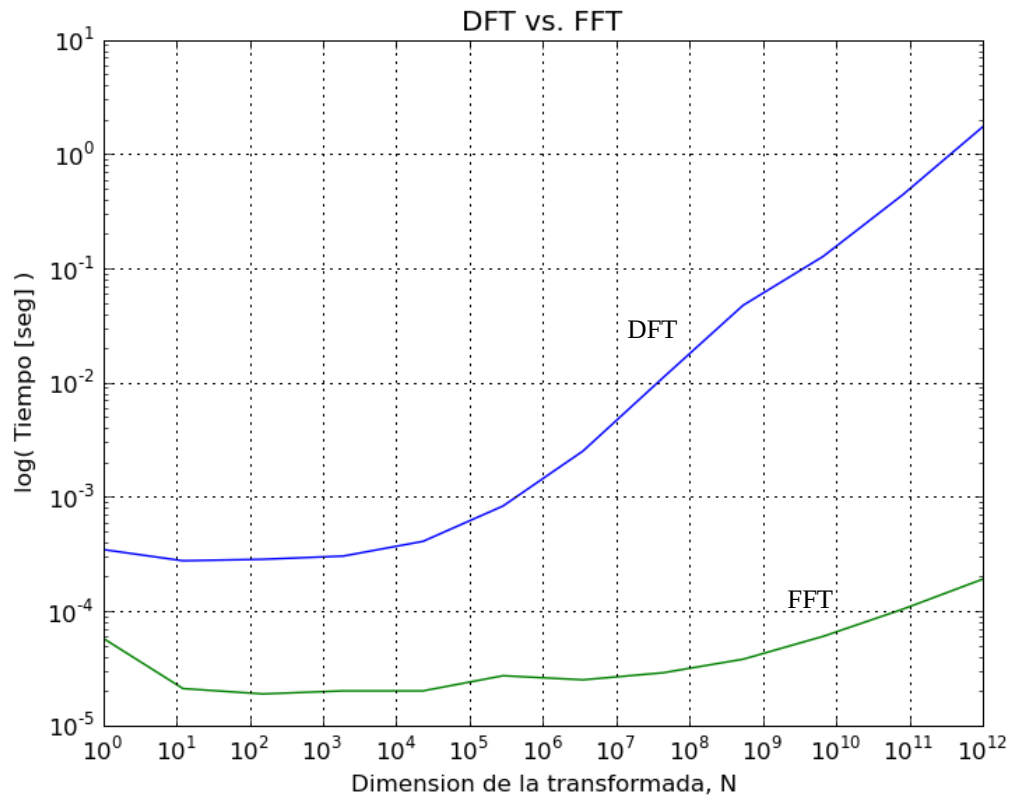


Figura 2.11. Comparativa DFT vs FFT [24]

Esta gran diferencia en tiempo de cálculo hace que el algoritmo de FFT sea utilizado en infinidad de aplicaciones. En el ámbito del procesamiento de audio, uno de los usos directamente aplicables a este proyecto es el método de convolución mediante FFT, que se verá en detalle en el siguiente capítulo.

CAPÍTULO 3: Diseño e implementación

En este capítulo se detalla el proceso de diseño de los subsistemas que componen el proyecto, el subsistema de acústica binaural y el subsistema de seguimiento de movimiento. Seguidamente se detalla el proceso de implementación de ambos sistemas y su unión.

3.1. SUBSISTEMA DE ACÚSTICA BINAURAL

La Figura 3.1. presenta un esquema visual general del subsistema de acústica binaural.

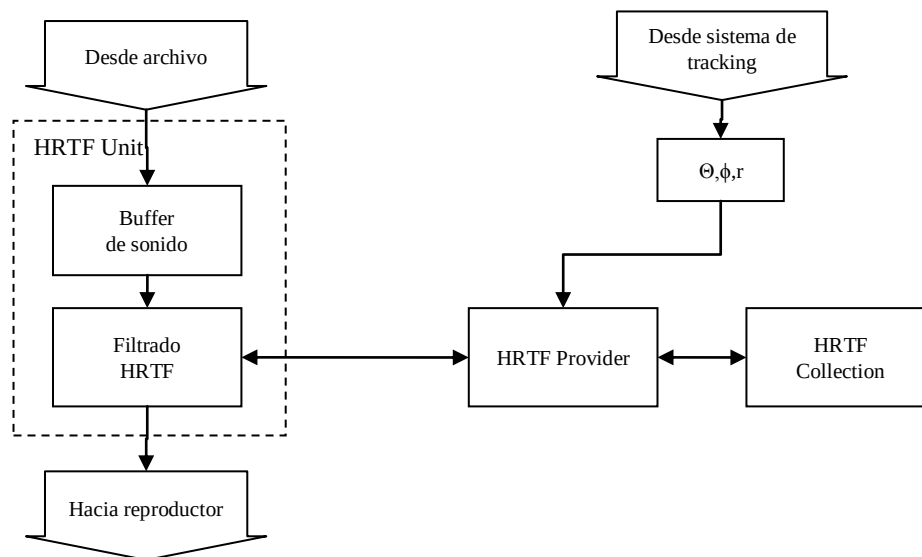


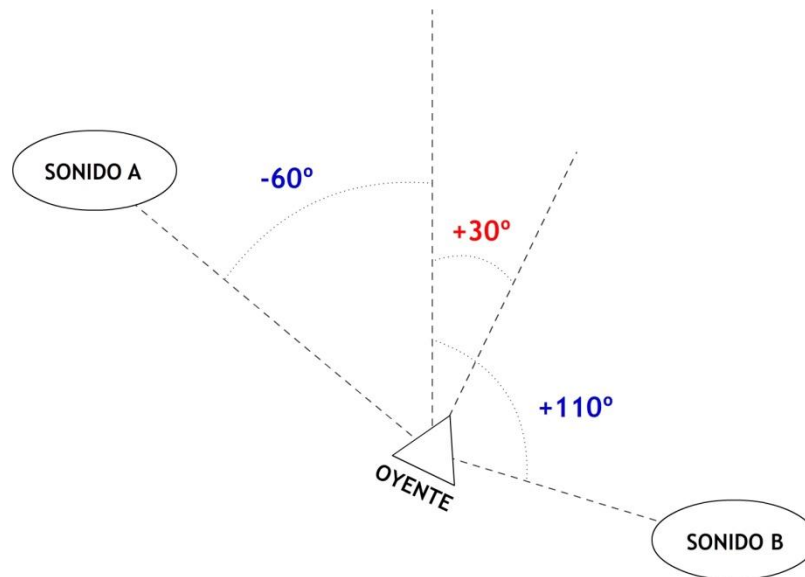
Figura 3.1. Esquema general del sistema

El subsistema de procesamiento binaural se compone de dos bloques principales. Por un lado, el gestor de HRTF recibe los datos de azimuth y elevación provenientes del sistema de seguimiento de movimiento (tracking). Usa esos datos para buscar en la colección de respuestas la HRTF correspondiente, o la más cercana en caso de no haber una concreta para dichos valores.

Por otro lado, se tiene el HRTF Unit, una unidad de procesamiento que funciona como un módulo independiente, que lee el archivo de audio y lo procesa según la HRTF seleccionada por el gestor de HRTF. Se ha desarrollado como un módulo independiente para poder ejecutar varias unidades simultáneamente, siendo el sistema capaz de procesar varios sonidos al mismo tiempo.

3.1.1. CÁLCULO DE AZIMUT Y ELEVACIÓN

Se ha implementado la posibilidad de que, opcionalmente, cada unidad de procesamiento aplique un offset propio a los valores entregados por el sistema de seguimiento de movimiento. De esta forma, cada unidad puede posicionar un sonido en una posición virtual propia. Cada uno de ellos se filtrará con una HRTF distinta, dando lugar a dos sonidos diferentes simultáneos en dos posiciones virtuales separadas.



$$\text{Azimut Sonido A} = -60 - 30 = -90^\circ$$

$$\text{Azimut Sonido B} = 110 - 30 = 80^\circ$$

Figura 3.2. Ejemplo de cálculo del azimut relativo de dos sonidos.

Tal y como se aprecia en la Figura 3.2, teniéndose en cuenta la posición relativa de cada sonido con respecto a la rotación dada por el sistema de seguimiento de movimiento, se obtiene la posición final virtual del sonido que debe percibir el oyente. Siendo así, cada unidad de procesamiento utiliza una HRTF distinta para procesar el sonido, dando lugar al mencionado efecto de dos sonidos situados en dos posiciones diferentes.

Este efecto se puede aplicar en dos o más unidades, tantas como el sistema sea capaz de ejecutar al mismo tiempo. En el capítulo sobre pruebas de rendimiento se analiza esta limitación.

Este método resulta de gran utilidad para recrear un espacio virtual completo, con varios sonidos colocados en distintas posiciones, generando un ambiente virtual muy realista. En la Figura 3.3 se representa a modo de ejemplo un ambiente recreado gracias a este sistema.



Figura 3.3. Composición sonora de ambiente de guerra

3.1.2. BASES DE DATOS DE HRTF

Para conseguir la sensación en el oyente del posicionamiento virtual del sonido en un espacio tridimensional, es necesario contar con modelos de respuesta de la cabeza humana. Desde el principio se eligió acudir a alguna biblioteca existente, dada la complejidad de grabar una propia. Tras realizar una búsqueda y valoración de distintas opciones, se eligió la base de datos realizada por el IRCAM (*Institut de Recherche et Coordination Acoustique/Musique*), en el ámbito de su proyecto *Listen*[25].

La base de datos contiene grabaciones de las respuestas de 51 sujetos tomadas en 187 direcciones distintas, con una separación de 15° entre muestras de azimut y de elevación. Ofrecen los datos en formato MAT (MATLAB) y WAV. También ofrecen las grabaciones compensadas en las que se han eliminado las respuestas de las cadenas de audio tanto en

grabación como en reproducción. Las respuestas tienen una longitud de 512 muestras tomadas a 44100 Hz y están organizadas como se aprecia en la Tabla 3.1.

Elevación	Azimut
-45	0
-45	15
...	...
-15	345
-30	0
-30	15
...	...
-30	345
...	...
...	...
75	345
90	0

Tabla 3.1. Ejemplo de estructura de colección de HRTF

En los archivos MATLAB, la información se organiza en forma de estructura, englobando todos los datos en una sola variable. La estructura contiene:

- `elev_v`: Vector de elevaciones.
- `azim_v`: Vector de azimuts.
- `content_m`: Matriz de datos
- `type_s`: Tipo de contenido (FIR)
- `samplin_hz`: Frecuencia de muestreo

Para el formato de archivo WAV, se ha seguido una nomenclatura en la cual cada archivo tiene la estructura genérica:

IRC_<sujeito>_<estado>_R<radio>_T<azimut>_P<elevación>

La Tabla 3.2 muestra en detalle cada campo del nombre:

Campo	Descripción
<sujeito>	Identificación del sujeto, desde 1000 a 1999.
<estado>	R para datos en bruto, C para datos compensados.
<radio>	Radio en centímetros.
<azimut>	Azimut en grados (0-359).
<elevación>	Elevación en grados (0-90 y 315-345).

Tabla 3.2. Nomenclatura archivos WAV IRCAM

Tanto el azimut como la elevación van en dirección CW (*clockwise*, agujas del reloj) siendo el 0 frente al sujeto en ambos casos. La elevación parte desde el frontal del sujeto y aumenta hacia la parte superior. Cabe mencionar que, aunque en todo el proyecto se habla de HRTF, lo que la base de datos del IRCAM proporciona en realidad son respuestas al impulso en el dominio del tiempo, por lo tanto, HRIR (*Head Related Impulse Response*). Para seguir una nomenclatura única, en adelante se empleará HRTF, ya que ambas son equivalentes al ser HRTF la transformada de HRIR.

3.1.3. PROTOTIPO MATLAB

En una primera fase se desarrolló un prototipo de pruebas en MATLAB para realizar un primer acercamiento al funcionamiento de la base de datos de HRTF y del filtrado binaural. El script desarrollado en primer lugar leía un archivo WAV, lo filtraba con una HRTF seleccionada y lo guardaba en otro archivo.

En una segunda fase, se añadió la funcionalidad del filtrado por bloques, el cambio dinámico de la respuesta HRTF y la reproducción en tiempo real. El azimut se variaba lentamente haciendo un recorrido en círculos alrededor de la cabeza del oyente. Sin embargo, dado el limitado control de la tarjeta de sonido que ofrece MATLAB, no se consiguió realizar una reproducción continua de la señal de salida.

Se planteó entonces el uso de una librería de MATLAB específica para reproducción de sonidos, pero este y otros motivos que se comentan más adelante en este mismo capítulo provocaron en este punto la decisión de utilizar otro entorno de desarrollo.

3.1.4. PROCESADO DE LA SEÑAL

El bloque de procesado es uno de los puntos más importantes del sistema, como se comentó en el capítulo de introducción. De su eficiencia depende el cumplimiento de varios de los objetivos propuestos para este proyecto. Por lo tanto, su diseño debe orientarse hacia la máxima eficiencia.

Dado que se cuenta con bloques de muestras de la señal de entrada por un lado y las muestras pertenecientes a la HRTF por otro, ambas en el dominio del tiempo, en el primer prototipo se usó la convolución.

Sin embargo, buscando la máxima eficiencia se optó por implementar el proceso realizando la *convolución por FFT*[26], un método más eficiente diseñado para el procesamiento digital, que permite realizar este tipo de filtrado en un tiempo menor. Tal y como se ha visto en el capítulo anterior, FFT es un algoritmo para el cálculo de la Transformada Discreta de Fourier que reduce drásticamente el tiempo de computación.

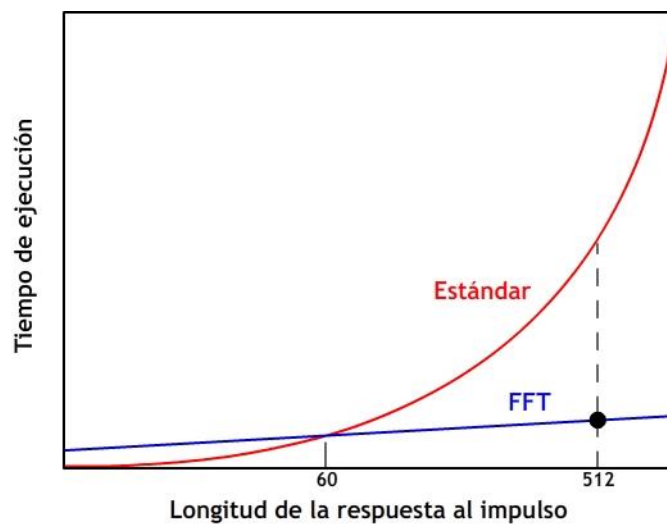


Figura 3.4. Tiempo de ejecución vs longitud de la respuesta

Como se puede apreciar en la Figura 3.4, para respuestas más largas de 60 muestras la convolución por FFT empieza a ser más eficiente, siendo, en el caso de las HRTF

utilizadas en este proyecto de 512 muestras de longitud, entre 4 y 5 veces más rápida que la convolución estándar.

En la Figura 3.5 se visualiza el funcionamiento de la convolución por FFT.

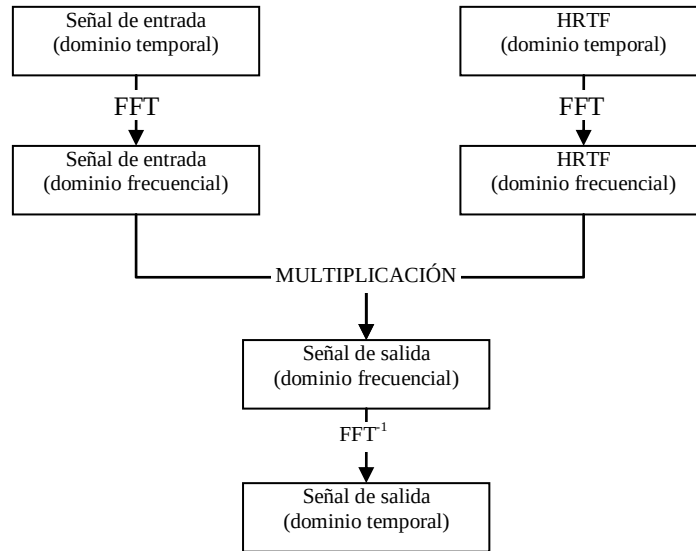


Figura 3.5. Convolución por FFT

3.1.5. MÉTODO OVERLAP-ADD

Para cumplir el objetivo de ejecutar el sistema en tiempo real, es necesario procesar la señal en bloques, los cuales deben tener un tamaño adecuado que permita procesar el bloque $n+1$ mientras se reproduce el bloque n . De esa forma se podrá reproducir la señal procesada de manera continua.

El método Overlap-Add[27][28][29][20] es una forma eficiente de obtener la convolución discreta de una señal larga (entrada) con una respuesta al impulso finita. El método consiste en dividir la operación en múltiples convoluciones de menor tamaño.

La clave de este método es entender cómo la convolución afecta a la longitud de las señales involucradas. Una señal de N muestras convolucionada con otra de M da lugar a una señal de salida de $N+M-1$ muestras.

Por lo tanto, si se filtra un bloque de 1024 muestras con un filtro cuya respuesta tiene 512, la señal de salida tendrá $1024+512-1 = 1535$ muestras. Esas 511 muestras sobrantes formarían parte del siguiente bloque, por lo que han de ser guardadas en un registro y sumadas en la siguiente iteración. La Figura 3.6 muestra este proceso gráficamente.

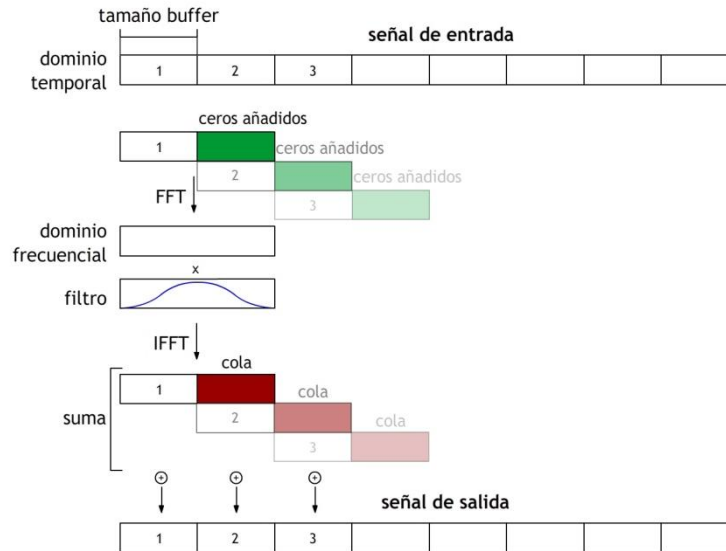


Figura 3.6. Método Overlap-Add

3.2. SEGUIMIENTO DE MOVIMIENTO

El subsistema de seguimiento de movimiento tiene como principal misión mantener la información de la rotación y distancia del oyente actualizada. Para ello, se requiere de un dispositivo que sea capaz de medir esa rotación. Como se comentó en el capítulo de introducción, se ha elegido un dispositivo controlador remoto de videoconsola.

3.2.1. ELECCIÓN DEL DISPOSITIVO

Partiendo de la idea de que no se pretendía desarrollar hardware ni software desde cero para la captura de movimiento, se valoraron algunas soluciones existentes en el mercado buscando que fuesen fácilmente integrables en el sistema. Las soluciones a priori

seleccionadas fueron algunos modelos de controles remotos de videoconsolas. El motivo principal fue la experiencia previa con la que contaba el proyectando usando los sensores de este tipo de dispositivos en varios proyectos personales. Además, son sistemas muy extendidos, con lo que es fácil encontrar documentación y proyectos que usan sus sensores.

Dado que los datos necesarios a obtener para la simulación son la rotación de la cabeza del oyente (azimut, elevación) y la distancia al punto, el primer requisito es que el dispositivo sea capaz de entregar al sistema los mencionados datos.

Se seleccionaron y valoraron dispositivos según los siguientes requisitos y criterios:

- Disponibilidad de obtener datos de rotación (azimut, elevación).
- Posibilidad de obtener datos de distancia.
- Precisión en la medida.
- Velocidad de adquisición.
- Método de comunicación.
- Existencia de librerías desarrolladas, preferiblemente en el lenguaje escogido.
- Dificultad de implementación.
- Necesidad de marcas visuales o sensores en el oyente.
- Coste del dispositivo.

Dispositivos analizados:

- Wiimote (Nintendo Wii)[16][30][31]
- Kinect (Microsoft Xbox)[32][33]
- Playstation Move (Sony PS3)[34][35]

3.2.1.1. *Wiimote (2006)*



Figura 3.7. Nintendo Wiimote

El dispositivo Wiimote (2006) representado en la Figura 3.7 destacaba por ser el primer control remoto de videoconsola destinado a capturar el movimiento del usuario. Fue un gran éxito: Actualmente la videoconsola Wii es la más vendida de la historia. Es un dispositivo novedoso, barato y muy extendido. Al llevar tiempo en el mercado, existe mucha información técnica y muchos proyectos de desarrollo para comunicarse con él y aplicar su tecnología a distintos fines.

La principal desventaja del Wiimote es que como combinación con sus acelerómetros, necesita la presencia de una barra de leds para calcular su posición, ya que el controlador incluye una cámara que debe captar los leds frente a ella. Por lo tanto, si ubicamos el sensor en la cabeza del usuario y los leds frente a él, al girarse el dispositivo perderá contacto visual, perdiendo así la información de distancia.

3.2.1.2. Microsoft Kinect (2010)



Figura 3.8. Microsoft Kinect

El controlador remoto de Microsoft (Figura 3.8) hace su aparición en 2010 con el eslogan de que el propio usuario es el controlador. Simplemente poniéndose frente al dispositivo, el usuario es capaz de interactuar con la consola mediante gestos y voz. El dispositivo incorpora una cámara junto con un sensor de profundidad que, gracias a un potente software de análisis de imagen, es capaz de reconocer gestos y rostros con gran precisión.

La gran ventaja de no necesitar marcas o sensores en el usuario es también su mayor inconveniente. Sería necesario para este proyecto implementar los algoritmos de reconocimiento, ya que este procesamiento lo hace la consola, no el controlador.

Por otro lado, aunque la precisión en el reconocimiento de gestos es muy buena, el sistema no reconoce pequeños cambios en la rotación de la cabeza del usuario.

3.2.1.3. Sony PS3 Move (2010)



Figura 3.9. Sony PS3 Move

El dispositivo Move (Figura 3.9) llegó al mercado en 2010 como apuesta de Sony por ofrecer lo mismo que ofrecía Nintendo con Wii. Dada la considerable diferencia en potencia gráfica entre las dos consolas, Sony esperaba que su dispositivo constituyera un paso más con respecto a Nintendo, pero no supieron ofrecer ninguna novedad, ni videojuegos más divertidos que los de su competidor, y el producto fracasó.

En el ámbito de este proyecto, el dispositivo Move presenta todas las ventajas que el Wiimote de Nintendo, y además soluciona sus inconvenientes.

No incluye la cámara integrada en el controlador, lo permite situar la cámara fija y el dispositivo móvil, solucionando el problema de la pérdida de contacto visual. Además, el objeto que la cámara reconoce es una esfera luminosa, lo que facilita aún más la visibilidad cuando la cabeza del oyente gira.

El dispositivo lleva menos tiempo en el mercado que el Wiimote, sin embargo existen proyectos en marcha en pequeñas comunidades de desarrollo que permiten comunicarse con el dispositivo de una manera sencilla.

3.2.1.4. Conclusiones

Se presenta a modo de comparativa la Tabla 3.3 que ilustra las diferencias entre los dispositivos analizados en cuanto a los requisitos y criterios necesarios para este proyecto.

Criterio	Nintendo Wiimote	Microsoft Kinect	Sony PS3 Move
Datos de rotación	Sí	No (Necesidad de algoritmos)	Sí
Datos de distancia	Sí	No (Necesidad de algoritmos)	Sí
Precisión	Buena pero limitada	Mala	Muy buena
Velocidad de adquisición	¿?	Depende del algoritmo	22 ms
Comunicación	Bluetooth	USB	Bluetooth
Librerías	Sí (C#)	No	Sí (C#)
Dificultad	Baja	Muy alta	Baja
Marcas o sensores	Sí	No	Sí
Coste	Bajo	Alto	Medio

Tabla 3.3. Comparativa de controles de videoconsolas

Atendiendo al anterior análisis, finalmente se optó por implementar esta parte del proyecto con el controlador de Sony.

3.2.2. DISPOSITIVO PS MOVE

El dispositivo[36], representado en la Figura 3.10, se compone de dos partes fundamentales: El controlador Move y la cámara PS3 Eye.



Dispositivo PS Move



Dispositivo PS3 Eye

Figura 3.10. Dispositivos PS Move y PS3 Eye

El controlador es un dispositivo inalámbrico dotado con sensores que miden la rotación del mismo, enviando los datos a la consola mediante Bluetooth. También consta de botones de control. Se puede encontrar un análisis de sus especificaciones técnicas en el Apéndice A.



Figura 3.11. Esfera iluminada del dispositivo PS Move

En su parte superior, como se aprecia en la Figura 3.11, tiene una esfera luminosa que, siendo captada por la cámara (situada junto a la consola), permite calcular la posición relativa a ésta.

Su radio de acción está limitado por la tecnología de comunicación inalámbrica que utiliza (Bluetooth), que en este dispositivo es de 10 metros. Esta distancia estará también limitada por la capacidad de captación visual de la cámara, la cual debe reconocer la esfera luminosa. En el capítulo de pruebas se analiza esta limitación.

El sensor de movimiento (controlador de videoconsola) debe estar fijado en la cabeza del oyente. Para ello, se acoplará el controlador a los auriculares tal y como se aprecia en la Figura 3.12 a continuación.

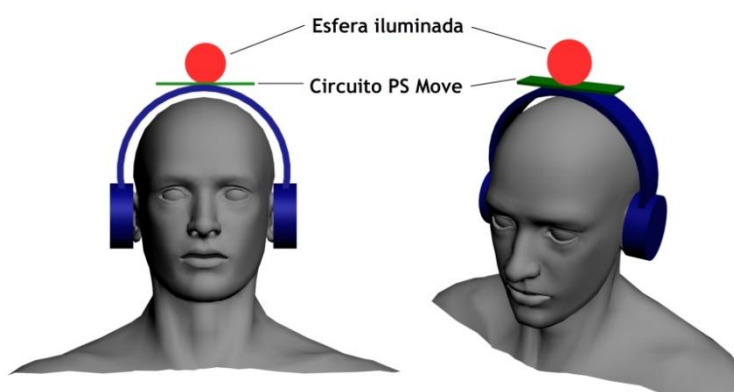


Figura 3.12. Diseño del acoplamiento del dispositivo PS Move en la cabeza del oyente

3.3. IMPLEMENTACIÓN

Se detallan a continuación los detalles de implementación de los subsistemas anteriormente diseñados.

3.3.1. ELECCIÓN DEL LENGUAJE DE PROGRAMACIÓN

Como se comentó en el apartado referente al primer prototipo desarrollado en MATLAB, hubo un punto en el que se cambió de entorno de desarrollo y de lenguaje. Tras una valoración inicial finalmente se optó por desarrollarlo en el lenguaje C#. Los motivos son variados:

- MATLAB tiene serias limitaciones en cuanto a la gestión de la tarjeta de sonido sin hacer uso de librerías externas.
- El proyectando cuenta con experiencia en el lenguaje C#.
- El proyectando ha realizado en el pasado un proyecto personal relacionado con un controlador de videoconsola (Nintendo Wii), en C#.
- C# es el único lenguaje en el que todas las librerías externas usadas en este proyecto están portadas.

Teniendo en cuenta todos los puntos anteriormente mencionados, finalmente se eligió el lenguaje C#[37] junto con el entorno de desarrollo Visual Studio de Microsoft para el desarrollo de la aplicación.

3.3.2. LIBRERÍA FMOD

La librería FMOD[38] está desarrollada por la empresa *Firelight Technologies*, en Australia. FMOD es su producto estrella, un motor de audio líder en la industria de los videojuegos, utilizada por la mayoría de las empresas más potentes. También es utilizado por ingenieros de sonido y productores musicales. La licencia de la librería FMOD se concede gratuitamente para uso no comercial.

La ventaja del éxito de este motor es que dispone de versiones adaptadas a todo tipo de dispositivos, desde ordenadores (Windows, Mac, Linux) hasta dispositivos móviles (iOS, Android, Blackberry, Windows Phone) y videoconsolas (Xbox, Wii, PS3, PS Vita, etc). Por tanto, de desear portar este proyecto a cualquiera de esos dispositivos, el proceso requerido sería mucho más sencillo.

FMOD Tiene su propio sistema de posicionamiento virtual en 3D. Sin embargo, su sistema se basa en un filtrado básico (paso bajo para sonidos posicionados detrás del oyente y panoramización para los laterales) para conseguir el efecto. Por ello, se descarta cualquiera de sus funciones orientadas al procesamiento tridimensional.

En este proyecto, la librería se encarga de las siguientes funciones:

- Comunicación con la tarjeta de sonido para reproducir el sonido.
- Lectura de los archivos. Soporta la mayoría de formatos de audio existentes, entre ellos: WAV, MP3, MIDI, XMA, OGG Y MOD. Incluso se pueden definir nuevos *codecs* para formatos propios.
- División del flujo de muestras en bloques. Se ha utilizado un tamaño de buffer de 1024 muestras.
- Análisis de carga del procesador. La librería incluye una herramienta de visualización y diagnóstico.

La librería tiene un conjunto sencillo de filtros básicos (ecualización, reverberación, etc) y además permite la implementación de filtros personalizados.

La parte de filtrado del proyecto se implementa precisamente como uno de esos filtros personalizados. El funcionamiento se basa en un método en el que la librería FMOD entrega un buffer de muestras, y recoge ese mismo buffer procesado. El proceso se detalla más adelante en este capítulo.

3.3.3. LIBRERÍA MOVEFRAMEWORK

La librería MoveFramework[39] es un proyecto de código libre de una comunidad de desarrollo, creado por Viktor Budaházi en Google Code. Permite la comunicación entre el PC y el dispositivo PS Move de Sony. Además de comunicarse con el dispositivo y obtener los datos necesarios de él, dispone de un algoritmo de reconocimiento de imágenes que realiza el seguimiento visual de la esfera iluminada con la cámara del dispositivo PS Move, proporcionando también su posición relativa en tres dimensiones.

Su implementación en C# dispone de un sencillo método que busca y se conecta automáticamente con el dispositivo.

3.3.4. LIBRERÍA FFTW

Tal y como comentó en el capítulo anterior, la FFT es uno de los algoritmos más complejos jamás desarrollados en el ámbito del procesamiento digital de señal. En ese sentido, para realizar este proyecto se ha utilizado una implementación ya existente. Se ha recurrido a la librería llamada FFTW (*Fastest Fourier Transform in the West*)[40], una librería de métodos para el cálculo de FFT muy extendida entre el desarrollo de software debido a su eficiencia y su versatilidad, ya que ofrece la posibilidad de calcular la FFT por diversos métodos, en múltiples dimensiones y para señales de tamaño arbitrario, lo cual facilita mucho su uso. La librería es de código libre, y está desarrollada en C++. Se ha utilizado una encapsulación en C# desarrollada por Tamas Szalay[41].

3.3.5. ESTRUCTURA DE CLASES

Se presenta en la Figura 3.13 el esquema general de implementación que se ha seguido para desarrollar el código de la aplicación.

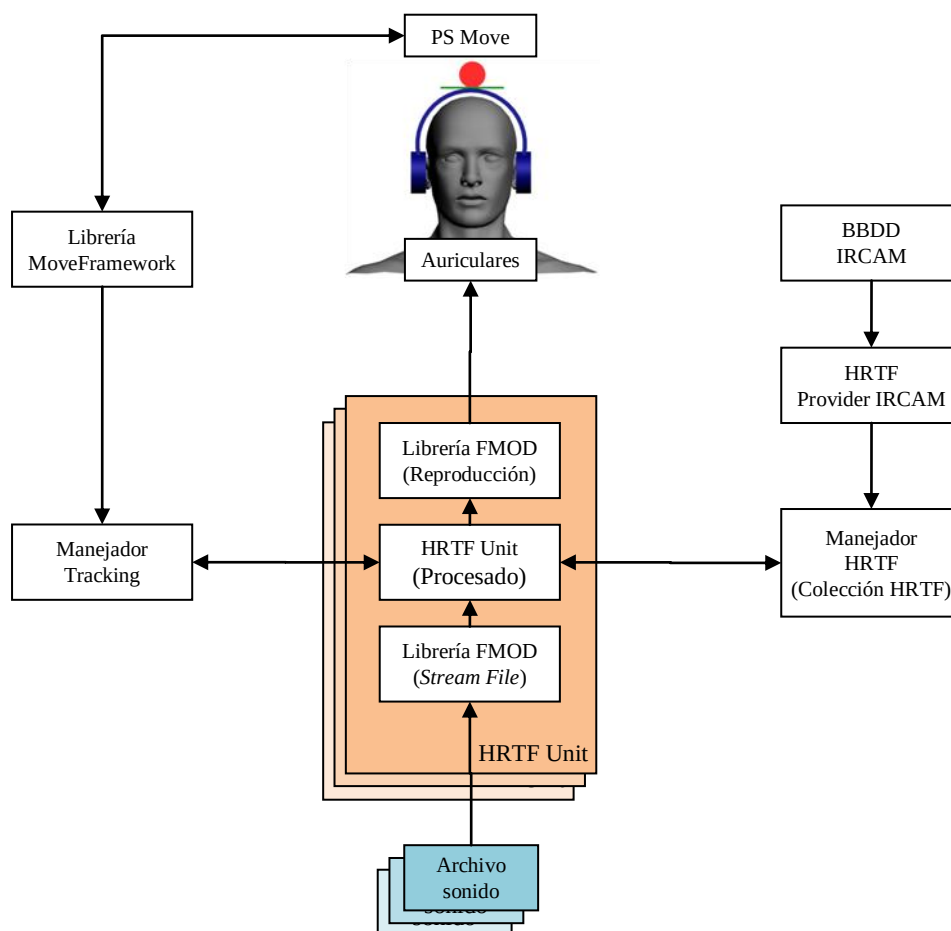


Figura 3.13. Esquema general de implementación

En los siguientes apartados se comentan los distintos bloques que componen este esquema general detallando su función.

3.3.6. MANEJADOR HRTF

Esta clase tiene como misión gestionar todo lo relacionado con las respuestas HRTF. En primer lugar, se le ha de indicar la colección de respuestas a cargar. Una vez cargada en memoria, se podrá acceder a la HRTF correspondiente a cualquier azimuth y elevación. Esta clase proporciona acceso a otras subclases como *HRTFCollection* y *HRTFProvider*.

3.3.7. HRTF COLLECTION

Con el fin de independizar lo máximo este proyecto de los recursos utilizados, se ha creado una clase propia que almacena las respuestas HRTF pertenecientes a todos los azimuths y elevaciones disponibles. Además, para optimizar el proceso de filtrado, la colección almacena directamente las HRTF en el dominio de la frecuencia.

Esta clase permite buscar entre sus registros la HRTF correspondiente a los ángulos de azimuth y elevación proporcionados, devolviendo el registro más cercano en caso de que no exista respuesta para los datos proporcionados.

3.3.8. HRTF PROVIDER

Esta clase se encarga de gestionar el enlace entre la base de datos de HRTF y la colección de HRTF. Lee los ficheros del disco duro y los transforma en una colección HRTF.

Con la intención de no limitar el desarrollo de este proyecto únicamente a las bases de datos de respuestas del IRCAM, se ha implementado la clase *HRTFProviderBase*, la cual es una interfaz genérica entre el Manejador y la base de datos. La clase *HRTFProviderBase* se debe extender para el tipo de recurso que se vaya a usar.

Se ha visto en este mismo capítulo la estructura de la información proporcionada por el IRCAM. Esta información viene dada en archivos WAV, con una nomenclatura de archivos muy concreta, y una referencia específica para los ángulos. Para otra base de

datos proveniente de otro estudio, los datos pueden organizarse de forma totalmente distinta.

En este caso, se ha implementado una clase extendida *HRTFProvider_IRCAM* que desarrolla los métodos de carga de colección y corrección de ángulos específicamente para las HRTF proporcionadas por el IRCAM.

En este proyecto se usa el formato y referencia mostrados en la Figura 3.14 para los ángulos de azimuth y elevación, los cuales coinciden con la referencia usada por el IRCAM.

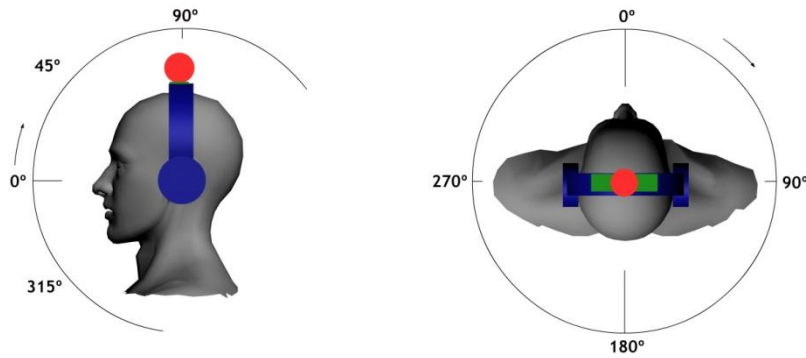


Figura 3.14. Referencias angulares de azimuth y elevación.

Aunque la información de las HRTF están en formato de archivo WAV y la librería FMOD es capaz de leer archivos de ese formato, con el objetivo de mantener independientes los dos subsistemas se ha desarrollado un método que lee y decodifica este tipo de archivos, obteniendo la información de las muestras y almacenando su transformada FFT en la colección HRTF. Se puede encontrar información más detallada sobre el formato WAV en el Apéndice C.

3.3.9. ARCHIVOS XML

Persiguiendo el objetivo de independizar el sistema del formato de las bases de datos de HRTF externas, se ha desarrollado un formato de archivo propio para almacenar las muestras pertenecientes a las HRTF de las colecciones. Se han implementado los métodos *GuardarXML* y *LeerXML*, que escriben y leen respectivamente los archivos XML en el formato propio anteriormente mencionado. Este archivo tiene la estructura mostrada en la Tabla 3.4.

```
<datos de cabecera>
<HRTF_data>
  <HRTF az="0" el="0"> (muestras codificadas) </HRTF>
  <HRTF az="15" el="0"> (muestras codificadas) </HRTF>
  ...
  ...
</HRTF_data>
```

Tabla 3.4. Formato de archivo de Colección HRTF XML propio

Este formato permite realizar una base de datos de HRTF propia con las muestras codificadas en punto flotante de 64 bits.

3.3.10. MANEJADOR TRACKING

Esta clase se ocupa de mantener un seguimiento del movimiento de la cabeza del oyente gracias a la librería *MoveFramework*. La librería proporciona un método de *callback* que actualiza los valores tanto de orientación como de posición, así como los botones pulsados.

La clase adapta los datos proporcionados por la librería *MoveFramework* para su correcto uso, ya que ésta proporciona la rotación en modo *Quaternion*. Para ello hace uso de otra librería externa de código libre llamada *YLSDrawing*, un conjunto de métodos y clases diseñados para representar gráficamente objetos en 3D de una forma sencilla que

ayuda a realizar los cálculos de rotación. Se puede ampliar la información sobre este proceso en el Apéndice B.

Una vez obtenidos los datos de azimut, elevación y distancia, se almacenan en la clase de manera pública para que los demás subsistemas tengan acceso a ellos.

3.3.11. HRTF UNIT

Esta clase es una de las principales de todo el sistema. Se encarga de abrir un archivo de audio, procesarlo y reproducirlo. Funciona de manera independiente, de tal forma que se pueden lanzar varias instancias simultáneas. Además, permite que cada instancia tenga unos valores de offset propios para los valores de posición y orientación, haciendo posible que cada instancia posicione un sonido en una posición distinta del espacio tridimensional virtual alrededor del oyente.

La unidad hace uso de la librería externa FMOD, un motor de sonido encargado de gestionar la decodificación de los archivos de entrada, así como de la reproducción de las muestras procesadas. La unidad HRTF crea un canal dentro del subsistema FMOD y lo asigna al archivo designado. Así mismo, también crea un filtro personalizado. La librería requiere la implementación de un método para gestionar ese filtro personalizado, en el cual hace entrega del buffer de muestras y recoge el mismo buffer procesado. Es en el método de *callback* llamado *BasicFilterCallback* de cada HRTF Unit en el que se realiza el procesamiento de la señal.

FMOD entrega las muestras de una forma concreta:

- Si la señal es monocanal, entrega un array de muestras, una tras otra.
- Si la señal es multicanal, entrega un array de muestras en las que se van intercalando las muestras de los canales, tal y como se visualiza en la Figura 3.15.



Figura 3.15. Disposición de muestras multicanal en array

Debido a esta especial configuración, en el caso de sonidos en estéreo es necesario realizar un pre-procesado de las muestras, separándolas en dos arrays distintos y procesándolas por separado para finalmente volverlas a unir. El proceso se describe visualmente en la Figura 3.16 mostrada a continuación.

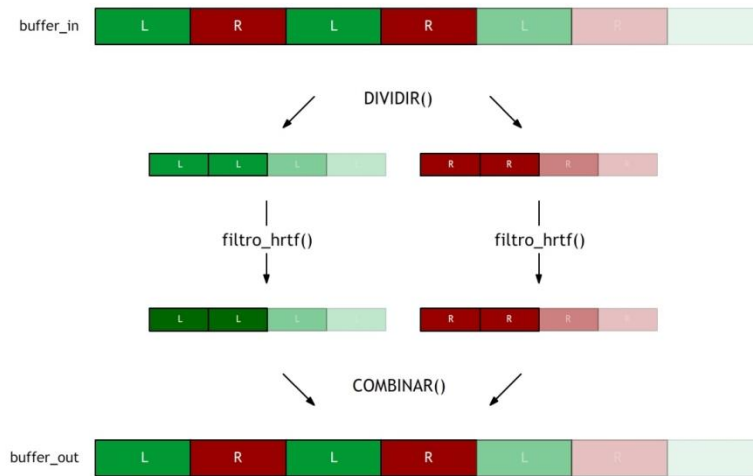


Figura 3.16. División de muestras para filtrado y recombinación

3.3.12. FILTRADO HRTF

Para realizar el filtrado HRTF se ha seguido el proceso tal y como se diseñó en el correspondiente apartado anterior en este mismo capítulo. Se ha implementado el método de Overlap-Add anteriormente descrito dando muy buenos resultados en cuanto a eficiencia, tal y como se puede comprobar en el capítulo sobre pruebas y resultados. Se puede encontrar un análisis del código fuente de este método en el Apéndice D.

3.4. DISEÑO DE LA APLICACIÓN VISUAL

Aprovechando la potencia que ofrece el entorno de desarrollo Visual Studio de Microsoft, se ha desarrollado una interfaz visual para facilitar el manejo del sistema. Se presenta un esquema visual de las ventanas y sus funciones en la Figura 3.17.

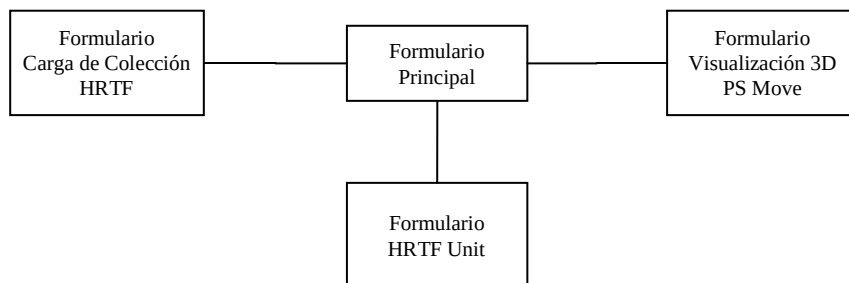


Figura 3.17. Estructura de formularios de la aplicación

Al iniciar la aplicación se muestra el formulario principal, que permite acceder a los demás, así como elegir qué variables de seguimiento se quieren activar.

El primer paso para la puesta en marcha del sistema es la calibración del dispositivo de seguimiento. Para ello, el oyente debe colocarse los auriculares y posicionarse frente a la cámara PS Eye. Una vez en esa posición, debe pulsar el botón PS situado en el dispositivo PS Move, lo cual reiniciará los valores de rotación a su punto de origen, asignando el valor 0° al azimut y a la elevación. Este ajuste puede hacerse en cualquier momento, y se puede visualizar gráficamente en la ventana de visualización 3D.

Para iniciar el proceso de simulación, se debe primero cargar una colección de HRTF de la base de datos desde la ventana de carga de colección. Una vez cargada en memoria, se podrán lanzar una o más Unidades HRTF con los archivos de audio que se deseen.

La ventana HRTF Unit permite reproducir y pausar la reproducción, así como situar visualmente la fuente sonora con respecto al oyente. También permite activar y desactivar el filtrado HRTF y el seguimiento de movimiento para observar los efectos que estos producen.

Se puede encontrar una guía de uso más detallada de la aplicación en el Apéndice E.

3.5. DIAGRAMAS DE CÓDIGO FUENTE

En este apartado se detalla la implementación del sistema desde el punto de vista del código fuente desarrollado. Cabe destacar que se ha usado una programación dirigida por eventos, con lo que no es factible la visualización por medio de diagramas de flujo. En su lugar, se han realizado diagramas de secuencia de algunos de los procesos más destacables del sistema, presentados y comentados a continuación.

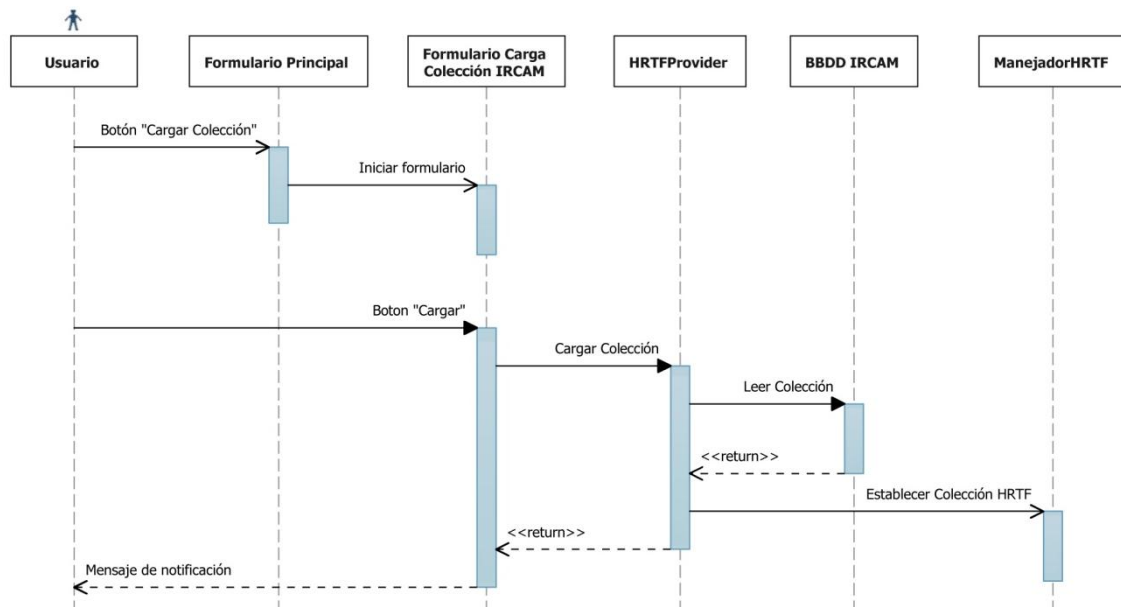


Figura 3.18. Diagrama de secuencia de carga de colección

La Figura 3.18 muestra el proceso inicial de cargar colección, un paso necesario para poder realizar la simulación. El usuario inicia la secuencia haciendo clic en el botón de carga de colección. El formulario principal inicializa y muestra el formulario secundario de carga de colección, específicamente diseñado para las colecciones del IRCAM. Al hacer clic en el botón de cargar, el formulario envía la orden a la clase HRTFProvider (HRTFProvider_IRCAM en este caso) que se encarga de leer la colección desde los archivos WAV, y una vez ha compuesto el objeto HRTFCollection, lo enlaza en ManejadorHRTF y notifica al usuario de la finalización del proceso mediante un mensaje en pantalla.

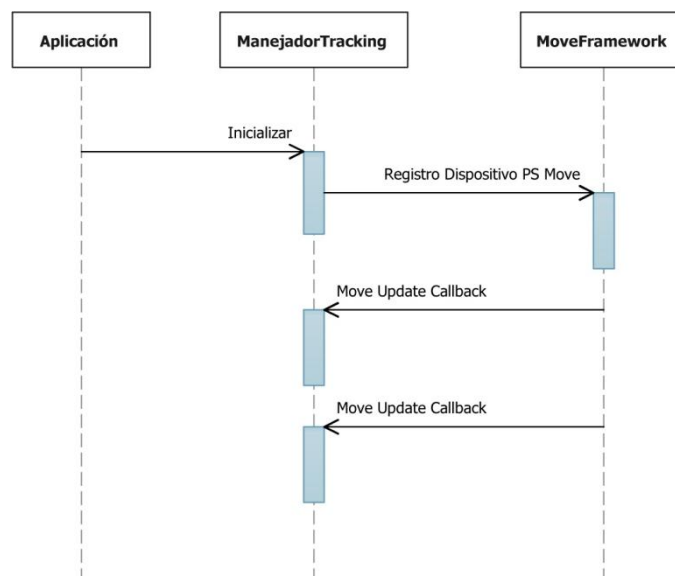


Figura 3.19. Diagrama de secuencia de registro de dispositivo

La Figura 3.19 muestra el proceso que sigue la aplicación una vez se ha ejecutado, en cuanto al seguimiento de movimiento. Vemos cómo ordena a la librería que gestiona el dispositivo de seguimiento que registre un método de callback para que ésta le informe de cada cambio en los datos de rotación y posición. Al recibir esos datos, se almacenan en las variables públicas correspondientes para que todo el sistema pueda tener acceso a ellas.

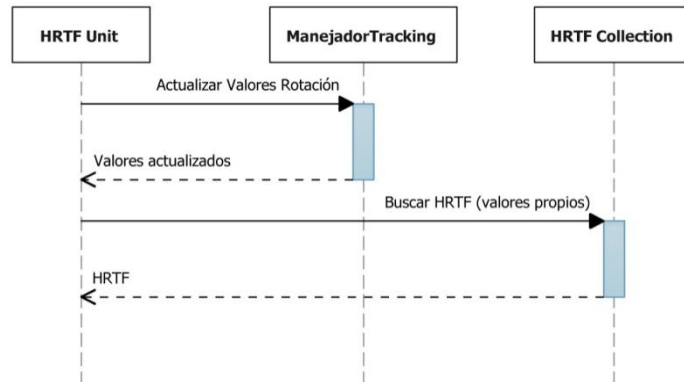


Figura 3.21. Diagrama de secuencia de actualización de rotación

En la Figura 3.21 se muestra un resumen del proceso necesario para actualizar los valores de rotación provenientes del dispositivo de seguimiento. Anteriormente se vio cómo ManejadorTracking registraba un callback en el que se actualizaban sus datos. En este proceso, se visualiza cómo cada unidad HRTF actualiza sus propios datos y con ellos selecciona la HRTF adecuada. Nótese que cada unidad tiene sus propios valores porque cada una de ellas tiene la posibilidad de añadir un offset que le permite posicionar el sonido en un azimut y elevación distintos de los que el ManejadorTracking está reportándoles.

CAPÍTULO 4: Pruebas y resultados

En este capítulo se presentan las pruebas realizadas al sistema y los resultados obtenidos, los cuales proporcionan información sobre el comportamiento del mismo en distintos aspectos.

Se presentan principalmente dos tipos de pruebas. Las pruebas objetivas valoran el nivel de eficiencia a nivel técnico del sistema. Las pruebas subjetivas, por el contrario, ofrecen un análisis de las respuestas de múltiples sujetos al sistema. Al ser un proyecto orientado a mejorar la experiencia sonora de las personas, se hace imprescindible el estudio del mismo en un grupo heterogéneo de individuos.

4.1. PRUEBAS OBJETIVAS

4.1.1. POSICIONAMIENTO

Las pruebas de posicionamiento permiten comprobar las especificaciones técnicas del sistema con los resultados reales, y compararlos con los objetivos proporcionados desde la teoría acústica. Se muestra el análisis de la resolución ofrecida por el subsistema de seguimiento.

4.1.1.1. *Azimut y Elevación*

Teniendo en cuenta que la base de datos de respuestas HRTF del IRCAM tiene sus grabaciones con una separación de 15°, se ha usado el tipo de dato entero para la representación del azimut y la elevación, una resolución que se ha considerado suficiente, tal y como se puede observar en la Tabla 4.1.

Precisión PS3 Move	Precisión Sistema	Separación muestras de HRTF	Precisión oído humano
<< 1°	1°	15°	1°-15°

Tabla 4.1. Comparativa disposición de respuestas HRTF

Se concluye que el sistema cumple holgadamente las especificaciones propuestas en este aspecto, ya que su resolución angular es mucho mayor que la que se necesita.

4.1.1.2. *Distancia*

La distancia se obtiene mediante un algoritmo que analiza la imagen captada por la cámara PS3 Eye. El algoritmo detecta la esfera iluminada del control remoto y estima la posición del mismo. La prueba realizada consiste en la medición de la posición en distintos puntos del espacio para determinar la cobertura y la precisión de la detección del algoritmo incluido en la librería.

Acudiendo de nuevo a las especificaciones técnicas de la cámara (disponibles en el Apéndice A), ésta posee un conmutador que cambia el ángulo de visión entre 56° y 75°, situado en la segunda opción (75°) para obtener una cobertura lo más amplia posible. Cabe destacar el hecho de que la información de rotación es totalmente independiente de la de la

distancia. Aunque el sistema no detecte la esfera luminosa, la información de rotación sigue siendo exacta ya que se envía mediante bluetooth.

La Figura 4.1 muestra un mapa visual de la cobertura obtenida. La zona verde marca los puntos en los que la cámara detecta la esfera luminosa.

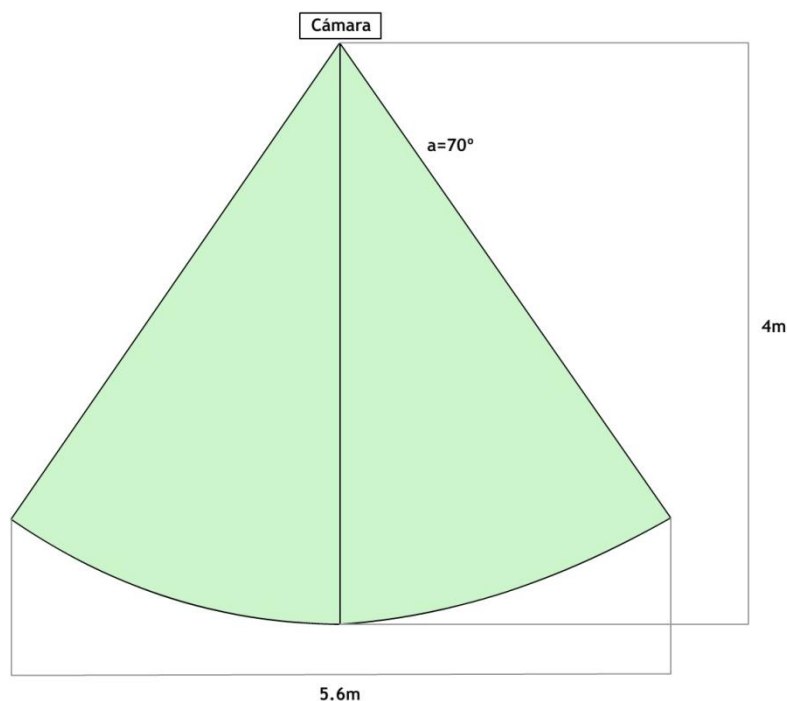


Figura 4.1. Cobertura de la cámara PS Eye

Se ha comprobado que la cámara capta la esfera luminosa hasta una distancia aproximada de 4m, con un ángulo de visión de casi 70° , llegando a abarcar 5.6m de amplitud horizontal.

4.1.2. EQUIPO DE PRUEBAS

Para el resto de pruebas objetivas, se ha contado con un equipo con la configuración descrita en la Tabla 4.2 a continuación:

Procesador	Intel Core i5 @ 3.2GHz (64 bits)
Disco duro	320 GB
RAM	4 GB @ 1333 MHz
Pantalla	13.3 Pulgadas 16:9
Sistema Operativo	Windows 7 Professional
Tarjeta Gráfica	Intel HD Graphics 3000 Series
Bluetooth	Broadcomm Integrado 2.0

Tabla 4.2. Características equipo de pruebas

4.1.3. TIEMPO DE ADQUISICIÓN Y DISPONIBILIDAD DE LOS DATOS

El sistema de seguimiento de movimiento debe ser capaz de capturar los datos de orientación y distancia con cierta precisión, pero también es importante que los datos estén disponibles y actualizados con la suficiente velocidad como para que el sistema reaccione deprisa a los movimientos del oyente.

En primer lugar, se parte del dato del tiempo de adquisición del dispositivo PS Move, que según sus especificaciones técnicas (disponibles en el Apéndice A) es de 22ms. Pero este tiempo se verá aumentado por los distintos procesos de la librería MoveFramework, responsable de recopilar todos los datos: no sólo la orientación, sino también la posición, y la posición se obtiene aplicando un algoritmo de procesamiento de imágenes en el que el sistema detecta la esfera luminosa y calcula su posición relativa.

Para medir el tiempo real de adquisición de la librería, esto es, el intervalo que tiempo que transcurre entre cada llamada al método *MoveUpdateCallback* de la clase *ManejadorTracking*, se ha recurrido a un temporizador de C# específico para este tipo de pruebas. La clase *StopWatch* se encarga de esta tarea con gran precisión, pues es capaz de entregar tiempos medidos en ciclos de reloj.

Este intervalo se encuentra entre 10 y 30 ms.

Sin embargo, hay otro factor a tener en cuenta en este tiempo de adquisición. Las unidades de proceso HRTF Unit se han diseñado para ser independientes, y por tanto, para no depender del tiempo de actualización de los datos en el *ManejadorTracking*. Cada unidad accede a los datos disponibles en la clase en su propio momento, y ese momento se rige por un temporizador interno de cada unidad; con lo que, en última instancia, es el tiempo de adquisición efectivo. Ese intervalo de tiempo del temporizador se podría bajar al mínimo, pero hay que tener en cuenta también los procesos que se siguen en su llamada, ya que puede afectar al rendimiento del sistema debido a la carga del procesador. En su llamada o *tick*, la unidad HRTF se copia los valores de posición y orientación existentes en *ManejadorTracking*, pero también actualiza su HRTF. Ese proceso implica que la colección de HRTF actual busque entre sus registros y devuelva la HRTF más cercana. Y, aunque los procesos de búsqueda en listas suelen intentar evitarse en lo posible en las aplicaciones destinadas al procesamiento de audio, en este caso, la medida de *StopWatch* indica que el proceso de búsqueda y copia de la HRTF desde la colección a la unidad tarda entre 12 y 30 μ s, lo cual es casi despreciable para el resto de tiempos que maneja el sistema en cada iteración, como se verá en los siguientes apartados.

Por tanto, como conclusión a este apartado, se ha visto que el máximo responsable del tiempo de adquisición es el temporizador de la unidad HRTF. Su valor de intervalo se puede establecer entre 10 y 30 ms garantizando así la máxima rapidez en la adquisición de los datos provenientes del *ManejadorTracking*, sin riesgo de sobrecargar el procesador. Desde un punto subjetivo, valores en torno a los 100ms siguen siendo válidos, pues dado que la resolución angular de la base de datos del IRCAM es de 15°, en movimientos normales de la cabeza no se aprecia ningún tipo de salto destacable.

4.1.4. TIEMPO DE PROCESO DEL BUFFER

Se ha buscado la optimización del código al máximo para garantizar que el procesado se haga en tiempo real. Tal y como se ha visto en el capítulo sobre diseño e implementación, en el marco de este proyecto se establece que el sistema funciona en tiempo real si consigue procesar el bloque $n+1$ antes de que termine la reproducción del bloque n . Con este objetivo en mente, se presentan los resultados de la medición del tiempo de proceso.

Se ha medido el tiempo usando la clase `StopWatch` de C#. Se ha usado en la función crítica del sistema, la función que procesa el buffer de audio: *BasicFilterCallback* de `HRTFUnit`.

En el peor caso, el filtro procesa 1024 muestras de audio a 44.100 Hz.

$$T_{max} = 1024 \text{ muestras} \times \frac{1 \text{ segundo}}{44.100 \text{ muestras}} = 23.22 \text{ ms} \quad (4.1)$$

El proceso, tal y como se muestra en la ecuación 4.1, no se puede extender más de ese tiempo. En el equipo descrito el tiempo de proceso medido oscila entre 1 y 2 ms, cumpliéndose también holgadamente este requisito.

4.1.5. CARGA DEL PROCESADOR

Usando el software de diagnóstico incluido con la librería `FMOD`, `FMOD Profiler`, se ha medido la carga del procesador resultando oscilar entre el 5% y el 7% aproximadamente por cada unidad de `HRTF`. En la Figura 4.2 se visualiza una captura del informe en tiempo real que proporciona el software `FMOD Profiler`.

En el equipo de pruebas se han conseguido procesar múltiples unidades simultáneas sin percibir ningún tipo de distorsión o retraso, siendo el límite para este equipo en concreto, de hasta 11 unidades como se aprecia en la figura.

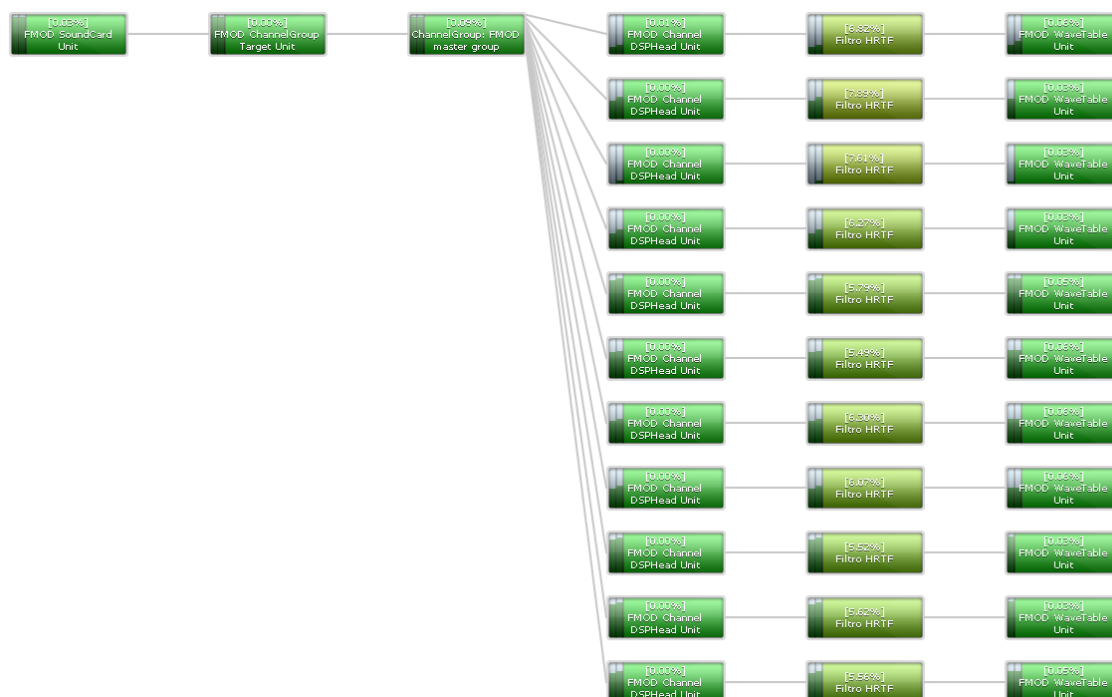


Figura 4.2. Captura de diagnóstico FMOD Profiler

4.1.6. CONCLUSIONES

En una primera fase del diseño conceptual del proyecto, se ajustaron los requisitos técnicos a unos mínimos para garantizar el éxito en el intento de realizarlo. Con el tiempo y esfuerzo invertidos, se han ido consiguiendo aquellos objetivos y marcando algunos nuevos, hasta llegar al sistema completo que se presenta en esta memoria.

Ejemplos de ello son objetivos como el procesado en tiempo real, el procesado de varias señales al mismo tiempo, o el posicionamiento de dichas señales en posiciones virtuales distintas.

Cabe destacar el compromiso con la eficiencia adquirido desde el primer momento del desarrollo. Gracias a ese esfuerzo se han cumplido los objetivos marcados con suficiente margen como para seguir dando pasos hacia un sistema más completo. Actualmente, como se ha podido comprobar en este primer apartado sobre pruebas objetivas, el sistema cumple todos los objetivos técnicos holgadamente, lo cual permite seguir añadiéndole funcionalidades, algunas de ellas comentadas en el capítulo sobre conclusiones y líneas futuras.

4.2. PRUEBAS SUBJETIVAS

4.2.1. DESCRIPCIÓN

El objetivo de las pruebas subjetivas es el de comprobar que el sistema consigue crear la ilusión de la posición virtual del sonido con respecto al oyente. La principal limitación del método de la HRTF es que depende en gran medida de la forma física de la persona. Es por eso por lo que es necesario realizar la prueba en varios sujetos distintos para tener un resultado concluyente.

Es necesario recordar que aunque la base de datos de HRTF del IRCAM está tomada con 51 sujetos distintos, eso no garantiza que los sujetos de estas pruebas tengan una experiencia totalmente real, ya que no son los mismos sujetos. Como se verá en la descripción de las pruebas, el primer paso es hacer un proceso de *matching* para buscar entre los 51 sujetos de la base de datos del IRCAM aquel cuya grabación provoque la sensación más realista en el sujeto protagonista de estas pruebas.

Al ser un proceso subjetivo, todas las pruebas se basan en las respuestas del sujeto a las preguntas formuladas.

La prueba se compone de los siguientes pasos:

- Buscar la colección de HRTF más adecuada para el sujeto
 - El sujeto escuchará sonidos de demostración[25] de cada colección consistentes en un sonido que se mueve en una circunferencia alrededor de la cabeza. Elegirá la que le produzca un efecto global más realista.
- Realizar prueba estática del posicionamiento (sin tracking) consistente en:
 - Situar un sonido concreto[42][43] en una posición determinada y pedir al sujeto que señale en qué dirección está. Las posiciones son 0, 45, 90, 135, 180, 225, 270 y 315° de azimut y 0° de elevación, junto con dos pruebas más con elevaciones positivas y negativas en el ángulo de azimut que mejor puntuación haya tenido en el anterior ejercicio. Se anotará si el sujeto ha sabido localizar el sonido correctamente, puntuando la medida con una nota del 1 al 5 siendo 5 el máximo.

- Realizar prueba dinámica del posicionamiento(con seguimiento de movimiento):
 - Situar varios sonidos en diferentes posiciones y pedir al sujeto que se gire hasta que le parezca que está situado frente a uno concreto de entre ellos. Las posiciones se elegirán aleatoriamente entre los azimuts que mayor puntuación hayan alcanzado, dado que lo que se busca con esta prueba es comprobar el funcionamiento del seguimiento de movimiento. Se puntuará también del 1 al 5. Se realizarán tres pruebas.

Adicionalmente se pedirá al sujeto que valore la sensación experimentada con una puntuación del 1 al 10 y se recogerán sus comentarios subjetivos al respecto.

4.2.2. RESULTADOS

A continuación se presenta una media de los resultados obtenidos en todas las pruebas subjetivas realizadas. En el Apéndice F se pueden encontrar los detalles de cada prueba concreta.

La Figura 4.3 muestra una representación gráfica de la respuesta a la primera de las pruebas subjetivas, en forma de diagrama polar, en el que se ha representado la puntuación del 1 al 5 en cada dirección escogida.

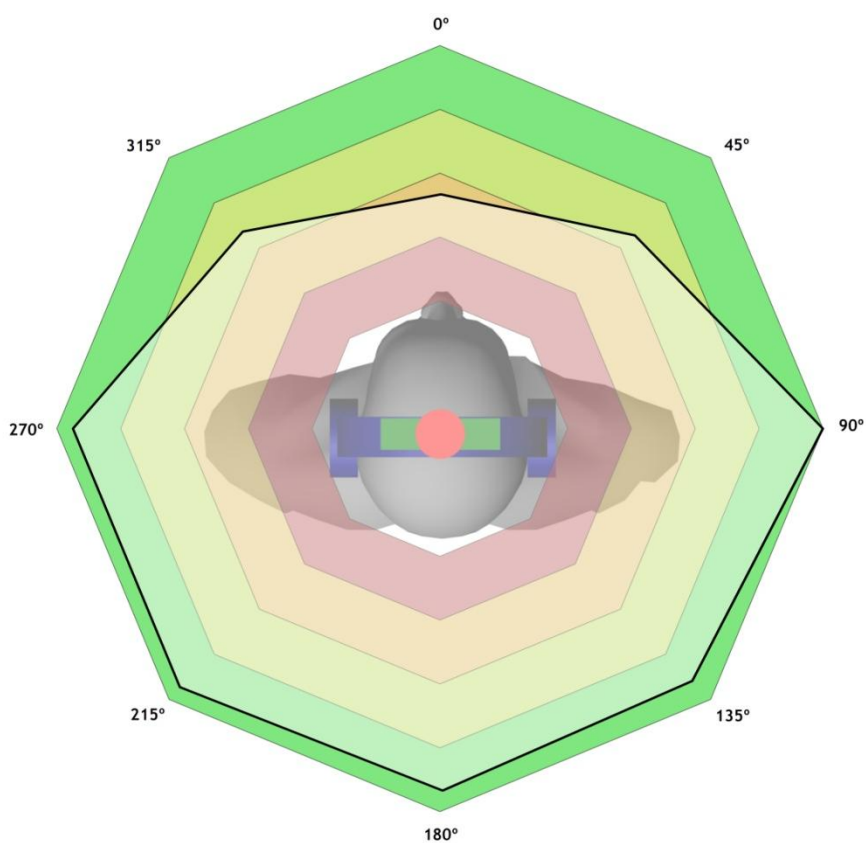


Figura 4.3. Diagrama de puntuación de azimut

La Figura 4.4 muestra una representación gráfica de la media en las pruebas de elevación.

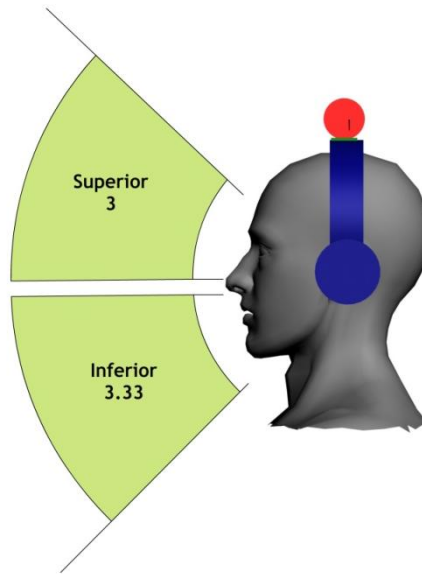


Figura 4.4. Diagrama de puntuación en elevación

Las pruebas de localización con seguimiento de movimiento han dado como resultado un 100% de la puntuación más alta. Todos los sujetos han sabido localizar el sonido en el espacio virtual con gran precisión.

En cuanto a la puntuación personal de los sujetos, la media se encuentra en un 8.7 sobre 10.

Por último, de entre los comentarios recogidos, se destaca la gran diferencia que comentan los sujetos que existe entre las pruebas con y sin seguimiento de movimiento. En algunos sujetos las pruebas estáticas no han funcionado bien en la parte frontal (azimuts entre 315 y 45).

4.2.3. CONCLUSIONES

El primer dato interesante que se desprende del análisis de los resultados de las pruebas objetivas es la confirmación de que la efectividad de la recreación virtual depende en gran medida del individuo, debido a la personalización de las HRTF escogidas. Algunos sujetos han encontrado una colección adecuada en todos los sentidos. Otros han tenido dificultades, sobre todo en los azimuts entre 270 y 90, correspondientes a la parte delantera. Esto se debe probablemente a la influencia de la forma de la cara en la respuesta HRTF.

La segunda conclusión a la que se ha llegado es que la incorporación del seguimiento de movimiento en la simulación potencia la inmersión en una gran medida. Incluso los sujetos que no supieron localizar sonidos en algunas posiciones en la primera prueba estática, luego fueron capaces de hacerlo con gran precisión en la segunda, en la que contaban con el seguimiento de movimiento. Al girarse y oír cómo cambiaba la posición del sonido, les fue mucho más sencillo localizarlo. Por lo tanto se deduce que la inclusión del seguimiento de movimiento ha sido crucial para conseguir una experiencia realista.

También es necesario añadir que en el proceso de *matching*, en el cual se intenta buscar una colección HRTF adecuada para el sujeto, se reproduce un archivo de audio consistente en un sonido que realiza un movimiento circular alrededor de la cabeza, pero lo hace a elevación 0°. Por lo tanto, aunque el sujeto perciba ese movimiento circular con gran realismo, ello no garantiza que vaya a suceder lo mismo para distintas elevaciones, lo cual puede afectar (y afecta, como se puede observar en las pruebas subjetivas estáticas) a la localización de sonidos situados por encima o por debajo de la línea visual del oyente.

En conclusión a este apartado sobre pruebas subjetivas, se ha comprobado que los sujetos han reaccionado bien a la localización estática en un entorno tridimensional virtual, pero han reaccionado notablemente mejor con la inclusión del seguimiento de movimiento, lo cual confirma el acierto de combinar los dos subsistemas para conseguir una sensación realista en el oyente.

CAPÍTULO 5:

Conclusiones y líneas futuras

Tal y como se comentó en el capítulo de introducción, este proyecto ha intentado dar un paso adelante en cuanto a la capacidad de recrear una sensación sonora en las personas. Supone un avance tecnológico y un salto cualitativo que puede formar parte del futuro del ámbito audiovisual. Sus muchas posibles aplicaciones lo hacen un campo de estudio e investigación muy interesante tal y como se verá en el apartado dedicado a las posibles líneas futuras de este mismo capítulo.

En el transcurso de esta memoria se han analizado todos los aspectos del proyecto:

- Su marco histórico, antecedentes, y el análisis de la necesidad de dotar a la experiencia auditiva de más realismo. Se han marcado los objetivos que se deseaban conseguir.
- Los conceptos básicos como pilares que sustentan este proyecto.
- El diseño de las necesidades del proyecto, y cómo se han llevado a cabo.
- Las pruebas objetivas y subjetivas que arrojan luz sobre la eficacia real del sistema.

En este último capítulo, se realiza un análisis final en el que se comentan los objetivos propuestos y se valora su cumplimiento. Se plantean posibles líneas futuras de ampliación y se comentan algunos problemas sufridos durante el desarrollo.

5.1. REPASO DE LOS OBJETIVOS PROPUESTOS

Se ha comentado en la introducción que la misión principal de este proyecto es la de provocar una nueva sensación en el oyente. Una experiencia mejor que la que existe actualmente. El objetivo más importante es mejorar la experiencia auditiva en las personas. Que el sistema consiga emular el posicionamiento de un sonido en un espacio tridimensional virtual en torno al oyente.

Tal y como se puede repasar en el capítulo anterior sobre pruebas subjetivas, este objetivo se ha conseguido. Todos y cada uno de los sujetos han respondido de manera satisfactoria a las pruebas, reconociendo su potencial y sus posibles y múltiples aplicaciones. Todos los sujetos han percibido la sensación que persigue el proyecto, en mayor o menor medida, como se puede apreciar en las puntuaciones.

El sistema, tal y como se ha implementado, es demasiado dependiente de la morfología de la cabeza del oyente tal y como se adelantó en el capítulo de introducción. Este efecto se agrava en especial en posiciones ubicadas frente al oyente, debido probablemente a la influencia de la forma del rostro, y a que el sistema auditivo es mucho más sensible localizando sonidos en la parte frontal. Este es un punto susceptible de mejorar, tal y como se comenta más adelante en el apartado de líneas futuras.

En el ámbito del procesado binaural, el objetivo principal era el de desarrollar un sistema de procesado que fuese capaz de, partiendo de un archivo de sonido, generar una señal de audio continua con ese sonido filtrado mediante un filtro basado en la respuesta de la cabeza humana (HRTF). Dicho objetivo se ha cumplido gracias a los métodos diseñados e implementados en el sistema. Adicionalmente se perseguía el objetivo de realizar el proceso en tiempo real, lo cual se ha conseguido, llegando a ser capaz de procesar hasta once canales de manera simultánea, en un ordenador personal de características normales.

En el ámbito del seguimiento de movimiento, gracias al uso de un dispositivo comercial orientado al mercado de las videoconsolas y por tanto al uso intensivo, se ha conseguido realizar el seguimiento de la cabeza del oyente con gran precisión. Esto ha sido posible también gracias al uso de una librería externa que ha simplificado dicha tarea. Este

seguimiento ha sido fundamental para crear el escenario virtual tridimensional alrededor del oyente, cumpliendo de manera holgada todas las expectativas.

Se puede concluir este apartado, en el que se han repasado los objetivos propuestos, con que el sistema cumple todos ellos, con las ligeras limitaciones que ya se adelantaron en su diseño.

5.2. POSIBLES MEJORAS Y LÍNEAS FUTURAS

En este apartado se comentan algunas mejoras aplicables al proyecto, así como posibles aplicaciones del mismo en distintos ámbitos.

5.2.1. INTERPOLACIÓN DE HRTF

Tal y como se ha visto en el capítulo de diseño e implementación del sistema, aunque la base de datos del IRCAM tiene una separación de 15° entre sus grabaciones, la clase implementada `HRTFCollection` realiza una búsqueda entre sus registros y devuelve la HRTF más cercana si no cuenta con un registro para el ángulo exacto que se busca.

Una posible ampliación de funcionalidad sería la de que el sistema, en lugar de buscar el registro más cercano, ofreciera una respuesta interpolada entre las dos más cercanas. Esto incrementaría el coste computacional de la selección y por tanto afectaría al rendimiento del sistema, pero daría lugar a un efecto mucho más suave que contribuiría a dotar al sistema de mayor realismo. Sería factible realizar un análisis del impacto de la implementación de esta funcionalidad en el rendimiento.

5.2.2. MEJORA DE HRTF

Aunque la base de datos de HRTF del IRCAM es muy completa y perfectamente adecuada para este proyecto, sería factible la búsqueda de una nueva base de datos más reciente (La base de datos del IRCAM es de 2003), o una sola colección de un modelo de cabeza *dummy*, lo cual evitaría el proceso preliminar de selección de la colección adecuada para el oyente.

5.2.3. RECREACIÓN CONDICIONES DEL ENTORNO

El entorno que rodea al oyente juega un papel fundamental en la forma en la que éste percibe los sonidos. La acústica de la sala en la que se encuentra éste actúa también como un filtro para el sonido. Por tanto, un avance en el intento de dotar de más realismo a la simulación sería el de incluir también ese efecto en ella. Datos como la respuesta en frecuencia o el tiempo de reverberación son fácilmente medibles, y se pueden implementar en el sistema como filtros parametrizables.

5.2.4. PLUGIN VST

VST (*Virtual Studio Technology*)[44] es una interfaz desarrollada por Steinberg para la realización de plugins de procesamiento de audio. Es un estándar en la industria del sonido y la música.

Una posible e interesante línea futura sería la de desarrollar un plugin VST para procesar en tiempo real las pistas en programas de producción musical. Se necesitaría adoptar el SDK (*Source Development Kit*) de Steinberg y probablemente reescribir todo el código. Sin embargo, se conseguiría un plugin que funcionaría en una amplia variedad de programas de edición de sonido y producción musical.

5.2.5. SENSORES PROPIOS

Aunque, tal y como se ha comentado en varios puntos de la memoria, el uso de un dispositivo controlador de una videoconsola existente tiene muchas ventajas, sigue siendo un elemento externo del cual el sistema depende en cierta medida. Al no haber un SDK oficial liberado por Sony, la adquisición de datos requiere de cierta ingeniería inversa y del uso de unos drivers concretos que también hacen al sistema demasiado dependiente de la plataforma en la que se ejecuta.

Por tanto, desarrollar un sistema de sensores propios facilitaría la tarea de hacer el sistema multiplataforma. Se diseñaría un hardware específicamente diseñado para este proyecto, eliminando otras funciones no necesarias. Los protocolos de adquisición se implementarían mediante unos drivers HID (Dispositivo de Interfaz Humana) los cuales son un estándar en cualquier plataforma.

5.2.6. HARDWARE AUTÓNOMO

Una posible aplicación comercial sería la de implementar todo el sistema en un hardware autónomo, independiente de la ejecución en alguna plataforma.

El dispositivo podría aceptar señales estéreo o multicanal (*surround*) y generar un espacio tridimensional alrededor del oyente recreando las condiciones de un equipo con varios altavoces.

Bibliografía

- [1] Historia del cine, Wikipedia. http://es.wikipedia.org/wiki/Historia_del_cine
- [2] Head Related Transfer Function, Wikipedia. http://en.wikipedia.org/wiki/Head-related_transfer_function
- [3] Daniel Maggiolo, "Apuntes de Psicoacústica", Diciembre 2012, Documento en formato PDF accesible por Internet en la dirección:
<http://www.eumus.edu.uy/docentes/maggiolo/acuapu/int.html>
- [4] Sound Localization, Wikipedia. http://en.wikipedia.org/wiki/Sound_localization
- [5] Vertical Sound Localization, Wikipedia.
http://en.wikipedia.org/wiki/Vertical_sound_localization
- [6] Binaural Recording, Wikipedia. http://en.wikipedia.org/wiki/Binaural_recording
- [7] Bowers & Wilkins. A history of binaural recording. <http://blog.bowers-wilkins.com/speakers/definitive-guides/a-history-of-binaural-recording/>
- [8] Sitio web de Maniquí Kemar (G.R.A.S.). <http://kemar.us/>
- [9] Audiophile Audition. Sitio oficial. <http://audaud.com/>
- [10] Motion Capture, Wikipedia. http://en.wikipedia.org/wiki/Motion_capture
- [11] Rotoscoping, Wikipedia. <http://en.wikipedia.org/wiki/Rotoscoping>
- [12] David J. Sturman. A Brief History of Motion Capture for Computer CharacterAnimation.
http://cose.math.bas.bg/Sci_Visualization/compAnim/animation/character_animation/motion_capture/history1.htm

- [13] Mike Seymour, "Art of Tracking", Agosto 2004, Documento en formato HTML accesible por Internet en la dirección:
http://www.fxguide.com/featured/art_of_tracking_part_1_history_of_tracking/
- [14] L.A. Noire, Wikipedia. http://en.wikipedia.org/wiki/La_Noire
- [15] Internet Movie Database. <http://www.imdb.com/>
- [16] Nintendo Wii, Wikipedia. <http://es.wikipedia.org/wiki/Wii>
- [17] DualShock, Wikipedia. <http://es.wikipedia.org/wiki/DualShock>
- [18] Digital Signal Processing, Wikipedia.
http://en.wikipedia.org/wiki/Digital_signal_processing
- [19] Procesamiento Digital de Señales, Wikipedia.
http://es.wikipedia.org/wiki/Procesamiento_digital_de_se%C3%B1ales
- [20] Steven W. Smith, "The Scientist and Engineer's Guide to Digital Signal Processing", 2002, Documento en formato PDF accesible por Internet en la dirección: <http://www.dspguide.com/>
- [21] Alan S. Willsky, S. Hamid Nawab Alan V. Oppenheim, *Señales y Sistemas*, Segunda Edición ed. España: Prentice Hall, 1998.
- [22] Convolution, Wikipedia. <http://en.wikipedia.org/wiki/Convolution>
- [23] Fast Fourier Transform, Wikipedia.
http://en.wikipedia.org/wiki/Fast_Fourier_transform
- [24] Blog de Procesamiento de Señales Digitales. Universidad del Valle de Guatemala.
<http://uvgie3016.wordpress.com/>

- [25] Proyecto Listen, IRCAM.
<http://recherche.ircam.fr/equipes/salles/listen/index.html>
- [26] Fast Convolution Using the FFT, Connexions.
<http://cnx.org/content/m10963/latest/>
- [27] Jan Berg Frank Wefers, "High-Performance Real-Time FIR-Filtering using fast convolution on graphics hardware", Septiembre 2010, Documento en formato PDF accesible por Internet en la dirección:
http://dafx10.iem.at/proceedings/papers/WefersBerg_DAFx10_P76.pdf
- [28] Steven W. Smith, "FFT Convolution and the Overlap-Add method, Artículo EETimes", Julio 2007, Documento en formato HTML accesible por Internet en la dirección: <http://eetimes.com/design/signal-processing-dsp/4017492/FFT-convolution-and-the-overlap-add-method>
- [29] Understanding Overlap and Add for Filtering, Stackoverflow.
<http://stackoverflow.com/questions/5117839/understanding-overlap-and-add-for-filtering>
- [30] Wiimote, Wii Brew. <http://wiibrew.org/wiki/Wiimote>
- [31] Michel Marriot, "At the Heart of the Wii, Micron-Size Machines", Diciembre 2006, Documento en formato HTML accesible por Internet en la dirección:
<http://www.nytimes.com/2006/12/21/technology/21howw.html>
- [32] LilysZajin, "Microsoft Kinect Specifications & Sensor Breakdown", Febrero 2012, Documento en formato HTML accesible por Internet en la dirección:
<http://lilyszajnberg.com/blog/?p=688>
- [33] Microsoft Kinect, Wikipedia. <http://es.wikipedia.org/wiki/Kinect>
- [34] PlayStation Move, Wikipedia. http://en.wikipedia.org/wiki/PlayStation_Move

- [35] PlayStation Move VS WiiMote, Blog Findelapartida.
<http://findelapartida.blogspot.com.es/2010/03/playstation-move-vs-wiimote.html>
- [36] Move Motion Controller. PS3 Developent Wiki.
http://www.ps3devwiki.com/wiki/Move_Motion_Controller
- [37] Microsoft Developer Network. <http://msdn.microsoft.com>
- [38] Proyecto FMOD, Firelight Technologies. <http://www.fmod.org/>
- [39] Proyecto MoveFramework, Google Code.
<http://code.google.com/p/moveframework/>
- [40] Fastest FourierTransform in the West. <http://www.fftw.org/>
- [41] Using FFTW in C#. <http://www.sdss.jhu.edu/~tamas/bytes/fftwcsharp.html>
- [42] Freesound. <http://www.freesound.org>
- [43] Mixing Secrets For The Small Studio. <http://www.cambridge-mt.com/ms-mtk.htm>
- [44] Steinberg VST Developer Support. <http://ygrabit.steinberg.de/>
- [45] "Full Tech Specs: PlayStation Move", Marzo 2010, Documento en formato HTML accesible por Internet en la dirección: <http://www.develop-online.net/features/819/Full-Tech-Specs-Playstation-Move>
- [46] Repurposing the PS3 Move. http://www.eissq.com/ps3_move/
- [47] Quaternion, Wikipedia. <http://en.wikipedia.org/wiki/Quaternion>
- [48] Proyecto YLS_CS_Drawing, CodeProject.
<http://www.codeproject.com/Articles/36868/Quaternion-Mathematics-and-3D-Library-with-C-and-G>

- [49] Peter Kabal, "WAVE Format Specifications", Enero 2011, Documento en formato HTML accesible por Internet en la dirección: <http://www-mmmsp.ece.mcgill.ca/Documents/AudioFormats/WAVE/WAVE.html>

- [50] Scott Wilson, "WAVE PCM Soundfile format", Enero 2003, Documento en formato PDF accesible por Internet en la dirección: <https://ccrma.stanford.edu/courses/422/projects/WaveFormat/>

APÉNDICE A: Especificaciones técnicas

dispositivo PS Move

En este apéndice se muestran en detalle las especificaciones técnicas del dispositivo de seguimiento de movimiento[45] que se ha usado en este proyecto. Sus principales características son:

- Tres giroscopios (Ejes X, Y, Z)
- Tres acelerómetros (Ejes X, Y, Z)
- Sensor de campo magnético terrestre
- Esfera luminosa multicolor para obtención de posición
- Tecnología Bluetooth
- Dispositivo de vibración
- Cámara PS Eye con micrófono incorporado y reducción de ruido de fondo

La Tabla A.1 muestra las principales características técnicas del dispositivo PS Move.

Característica	Detalle
Nombre de producto	PlayStation Move Controller
Código de producto	CECH-ZCM1
Fecha de lanzamiento	Otoño 2010
Color	Negro
Peso	Aprox. 145gr.
Dimensiones	200mm x 46mm (Altura x Diámetro)
Tipo de batería	Interna, litio-ion recargable
Voltaje	DC 3.7V
Temperatura de funcionamiento	5°C – 35°C

Tabla A.1. Características Técnicas PS Move

La siguiente Tabla A.2 muestra un desglose del hardware más destacable[46] incluido en el dispositivo PS Move.

Modelo	Tipo	Descripción
STM32F103VBT6	Microcontrolador	ARM Cortex M3 @ 72Mhz
AKM AK8974	Magnetómetro	Proporciona un valor en 3 dimensiones de la orientación.
STM LPR425AL	Giroscópio	Giroscopio de dos ejes (X-Y)
Y5250H 2029 K8QEZ	Giroscopio	Giroscopio de un eje (Z)
KIONIX KXSC4 10227 2410	Acelerómetro	Acelerómetro de tres ejes
BC4RE	Módulo Bluetooth	Módulo CSR BlueCore4

Tabla A.2. Desglose hardware PS Move

En cuanto a la cámara PS Eye, la Tabla A.3 resume sus características más importantes.

Característica	Detalle
Resolución	60 Hz: 640x480 px. 120 Hz: 320:240 px.
Frecuencia de muestreo	48 Khz.
SNR	-90 dB
Color	Negro
Fuente de alimentación	USB
Opciones de conectividad	USB 2.0

Tabla A.3. Características Técnicas PS Eye

APÉNDICE B: Cuaterniones y operaciones geométricas

En este apéndice se detalla proceso de cálculo de los valores de azimut y elevación partiendo del valor proporcionado por la librería MoveFramework en formato de Quaternion.

Los Cuaterniones[47] son una extensión de los números reales similar a los números complejos. Fueron creados por William Rowan Hamilton en 1843, en un intento por extender los números complejos a más dimensiones, en este caso, 4 dimensiones. De entre todas sus aplicaciones, cabe destacar su uso en gráficos generados por ordenador para representar la orientación de un objeto en tres dimensiones. Es una forma de representación más compacta, sencilla y rápida que usar la notación angular de Euler y además evita algunos problemas asociados a ella al usarlos en rotaciones.

Una sencilla forma de interpretar un vector de Quaternion es considerarlo como la rotación existente entre un punto A y un punto B en tres dimensiones. Concretamente el Quaternion hace referencia a la rotación de un punto alrededor de un eje concreto, como se puede apreciar en la Figura B.1.

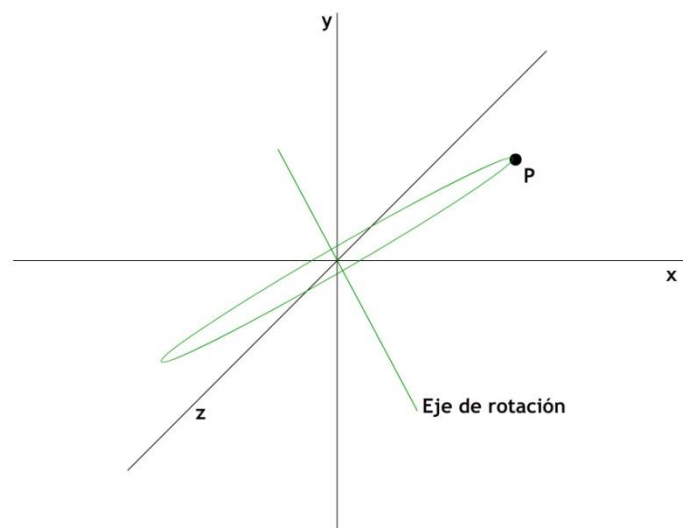


Figura B.1. Representación gráfica con Quaternion

Sin embargo, en este proyecto se realiza todo el cálculo geométrico y selección de HRTF usando los valores de azimut y elevación, por tanto, es necesario convertir el vector proporcionado por la librería MoveFramework en sus valores correspondientes de azimut y elevación. Para ello se hace uso de la librería YLSDrawing[48]. Dicha librería es un sencillo conjunto de métodos para representar puntos en tres dimensiones, creada para la representación de figuras tridimensionales sencillas.

Para obtener los valores de azimut y elevación a partir del Quaternion de orientación del dispositivo, primero se ha hallado la disposición de los ejes que usa la librería MoveFramework para componer su Quaternion. Una vez establecida la orientación base, se han usado dos Quaterniones que hacen referencia a los ejes que componen los movimientos de azimut y elevación, y finalmente se calcula el ángulo entre cada par de Quaterniones. La Figura B.2. muestra gráficamente el proceso de cálculo.

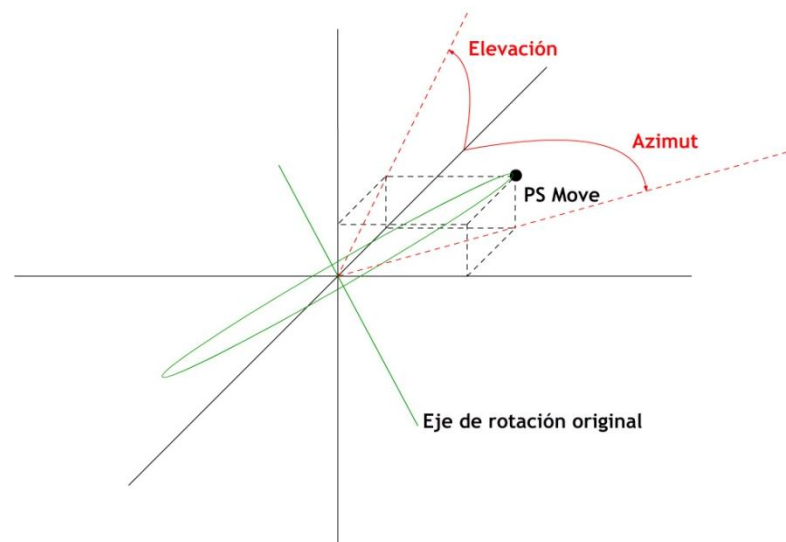


Figura B.2. Cálculo de ángulos desde Quaternion

APÉNDICE C: Formato WAV

En este apéndice se detalla la estructura de un archivo WAV. El formato WAV (Wave Table)[49][50] fue desarrollado por Microsoft para el almacenamiento de señal de audio. Se compone de una cabecera en la que se recogen las principales características de los datos almacenados, y de los datos en sí.

La Tabla C.1 muestra la estructura de campos del archivo WAV.

Endian	Offset	Nombre	Tamaño	Descripción	Bloque
Big	0	ChunkID	4	Contiene “RIFF” en modo ASCII	Bloque de cabecera
Little	4	ChunkSize	4	Tamaño del contenido	
Big	8	Format	4	Contiene “WAVE”	
Big	12	Subchunk1ID	4	Contiene “fmt “	Bloque de Formato (fmt)
Little	16	Subchunk1Size	4	Tamaño del bloque de formato	
Little	20	AudioFormat	2	PCM = 1, resto para algún tipo de compresión	
Little	22	NumChannels	2	Mono=1, Estéreo=2, etc.	
Little	24	SampleRate	4	Frec. Muestreo.	
Little	28	ByteRate	4	Tasa de bits	
Little	32	BlockAlign	2	Numero de bytes por muestra	
Little	34	BitsPerSample	2	Bits por muestra	
Big	36	Subchunk2ID	4	Contiene “data”	Bloque de Datos (data)
Little	40	Subchunk2Size	4	Numero de bytes del bloque de datos	
Little	44	Data	*	Muestras de sonido	

Tabla C.1. Estructura de archivo WAV

La Figura C.1 muestra un ejemplo de estructura de archivo WAV.

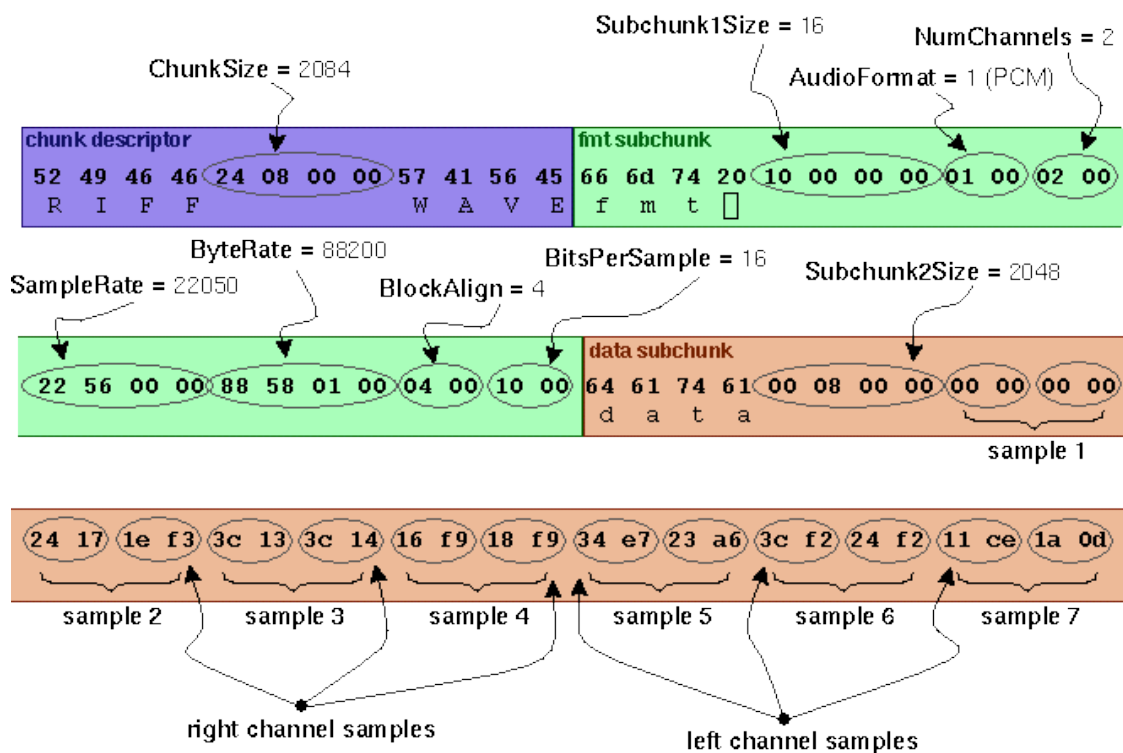


Figura C.1. Ejemplo de archivo WAV [50]

APÉNDICE D: Análisis FiltroHRTF

En este apéndice se presenta un análisis del código fuente del método FiltroHRTF de la clase HRTFUnit, el que se ha considerado el punto crítico del sistema, dada la importancia que tiene que sea lo más eficiente posible. Se presenta a continuación el código fuente completo del método, para posteriormente pasar a su análisis bloque a bloque.

```
private void FiltroHRTF(IntPtr inbuf, IntPtr outbuf, uint length, int canal)
{
    uint len_x = 1024;
    uint len_h = 512;
    uint len_blk = len_x + len_h - 1;

    IntPtr filter_data = canal == 0 ? filter_i_data : filter_d_data;
    ZeroPad(inbuf, len_x, 2 * (len_blk));

    fftwf.execute(fftwf.dft_r2c_1d((int)len_blk, inbuf, X_in, fftw_flags.Estimate));

    unsafe
    {
        float x, y, u, v;
        float* pX_in = (float*)X_in.ToPointer();
        float* pX_out = (float*)X_out.ToPointer();
        float* currentHRTF = (float*)filter_data.ToPointer();
        float* ib = (float*)inbuf.ToPointer();

        for (int i = 0; i < (2*len_blk)-1; i += 2)
        {
            x = pX_in[i];
            y = pX_in[i + 1];
            u = currentHRTF[i];
            v = currentHRTF[i + 1];
            pX_out[i] = (x * u - y * v);
            pX_out[i + 1] = (x * v + y * u);
        }

        fftwf.execute(fftwf.dft_c2r_1d((int)len_blk, X_out, outbuf, fftw_flags.Estimate));

        float[] tail = canal == 0 ? tail_i : tail_d;
        unsafe
        {
            float* outbuffer = (float*)outbuf.ToPointer();

            float n = 0;
            for (var i = 0; i < len_x; i++)
            {
                n = outbuffer[i] / (int)(len_blk);
                outbuffer[i] = n;
                if (i <= len_h-1) outbuffer[i] += tail[i];
            }

            for (var i = 0; i < len_h; i++)
                tail[i] = outbuffer[len_x + i] / (int)(len_blk);
        }
    }
}
```

Como ya se adelantó en el capítulo sobre diseño e implementación, este método incorpora las técnicas de Overlap-Add y Convolución por FFT para realizar el filtrado HRTF a un bloque de muestras de la señal original. A continuación se analiza el método en detalle comentando cada bloque por separado.

```
private void FiltroHRTF(IntPtr inbuf, IntPtr outbuf, uint length, int canal)
```

El método FiltroHRTF tiene como parámetros el buffer de entrada (inbuf), el de salida (outbuf), el tamaño de buffer (length) y el canal que se va a filtrar (izquierdo/derecho).

```
uint len_x = 1024;
uint len_h = 512;
uint len_blk = len_x + len_h - 1;

IntPtr filter_data = canal == 0 ? filter_i_data : filter_d_data;
```

Las variables len_x y len_h hacen referencia al tamaño de buffer y al tamaño de la respuesta HRTF respectivamente. La variable filter_data almacena la respuesta en sí, que es seleccionada entre la izquierda y la derecha según el canal que se ha entregado como parámetro (0=izquierda, 1=derecha).

```
ZeroPad(inbuf, len_x, 2 * (len_blk));
```

ZeroPad es un método que rellena un array con ceros según las posiciones que se le indiquen. En este caso, el buffer de entrada se rellena con ceros hasta completar el tamaño del bloque.

```
fftwf.execute(fftwf.dft_r2c_1d((int)len_blk, inbuf, X_in, fftw_flags.Estimate));
```

En esta línea se realiza la FFT del buffer de entrada y se almacena en la variable X_in, haciendo uso de la librería FFTW. Nótese que el método al que se llama es fftwf.dft_r2c_1d(), el cual hace referencia a realizar la DFT con entrada real y salida

compleja, en una dimensión, y en tipo de datos float. La etiqueta `fftw_flags.Estimate` se usa para que la librería `fftwf` realice la transformada de la forma que estime más eficiente. La librería tiene la opción de ejecutar primero unas pruebas de diagnóstico que optimizan el cálculo las transformadas, pero dado que el sistema ya resulta suficientemente rápido de por sí, esta opción no es necesaria.

```
unsafe
{
    float x, y, u, v;
    float* pX_in = (float*)X_in.ToPointer();
    float* pX_out = (float*)X_out.ToPointer();
    float* currentHRTF = (float*)filter_data.ToPointer();
    float* ib = (float*)inbuf.ToPointer();

    for (int i = 0; i < (2*len_blk)-1; i += 2)
    {
        x = pX_in[i];
        y = pX_in[i + 1];
        u = currentHRTF[i];
        v = currentHRTF[i + 1];
        pX_out[i] = (x * u - y * v);
        pX_out[i + 1] = (x * v + y * u);
    }
}
```

En primer lugar, la instrucción `unsafe` hace referencia a que el código situado entre sus llaves se considera “no seguro”. Un ámbito especial de `c#` necesario para trabajar con punteros, lo cual resulta especialmente rápido para operaciones con arrays de datos. En este bloque se realiza la multiplicación de la señales en el dominio de la frecuencia. Se toman los datos de `X_in` y de la respuesta `HRTF`, se multiplican (multiplicación de números complejos) y se almacenan en la variable de salida `X_out`, que será la TFD de la señal filtrada.

```
fftwf.execute(fftwf.dft_c2r_1d((int)len_blk,X_out,outbuf,fftw_flags.Estimate));
```

La señal de salida se devuelve al dominio del tiempo. Nótese que esta vez el método llamado hace referencia a la transformada con entrada compleja y salida real.

```
float[] tail = canal == 0 ? tail_i : tail_d;
unsafe
{
    float* outbuffer = (float*)outbuf.ToPointer();

    float n = 0;
    for (var i = 0; i < len_x; i++)
    {
        n = outbuffer[i] / (int)(len_blk);
        outbuffer[i] = n;
        if (i <= len_h-1 ) outbuffer[i] += tail[i];
    }

    for (var i = 0; i < len_h; i++)
        tail[i] = outbuffer[len_x + i] / (int)(len_blk);
}
```

Por último, se añade a la señal de salida la información perteneciente a la cola del bloque anterior, y así mismo se almacena la cola de éste bloque de muestras procesado para añadirse al bloque siguiente.

APÉNDICE E: Guía de la aplicación

En este apéndice se presenta una guía de uso de la aplicación desarrollada y una descripción de los requisitos y pasos a seguir para su funcionamiento. La aplicación lanza al ejecutarse la ventana principal, mostrada en la Figura E.1.

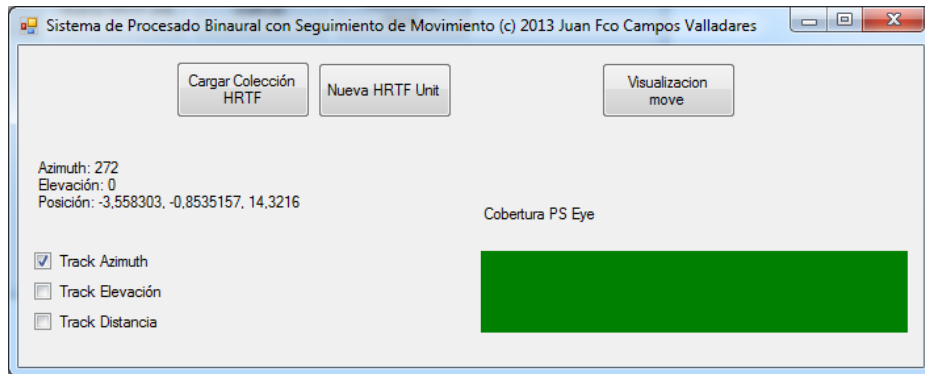


Figura E.1. Ventana principal

La ventana principal tiene información sobre el estado del seguimiento de movimiento, con el propósito de calibrarlo antes de iniciar ninguna unidad. También se puede omitir el seguimiento de azimuth, elevación o distancia, respectivamente. A la derecha se encuentra el indicador que refleja si el dispositivo PS Move está siendo detectado por la cámara PS Eye, cambiando el color del recuadro a rojo si no lo está detectando. Los botones superiores abren el resto de ventanas de la aplicación.

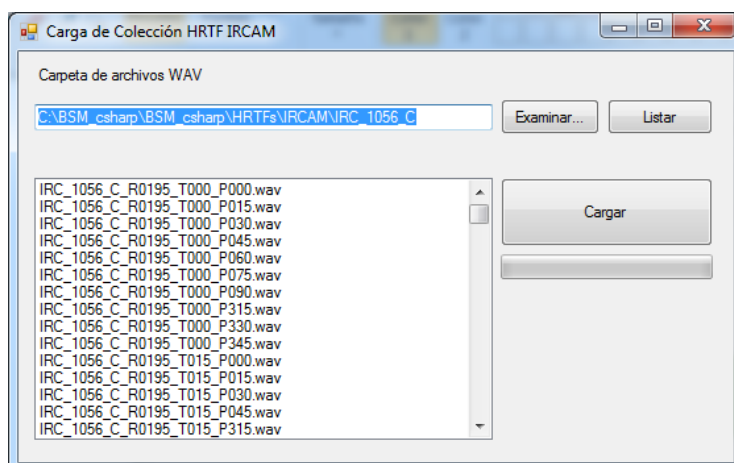


Figura E.2. Ventana de carga de colección

La ventana de carga de colección (Figura E.2) selecciona la colección de HRTF y la carga en memoria. Es el primer paso necesario antes de realizar la simulación. Está específicamente diseñada para cargar las colecciones del IRCAM.

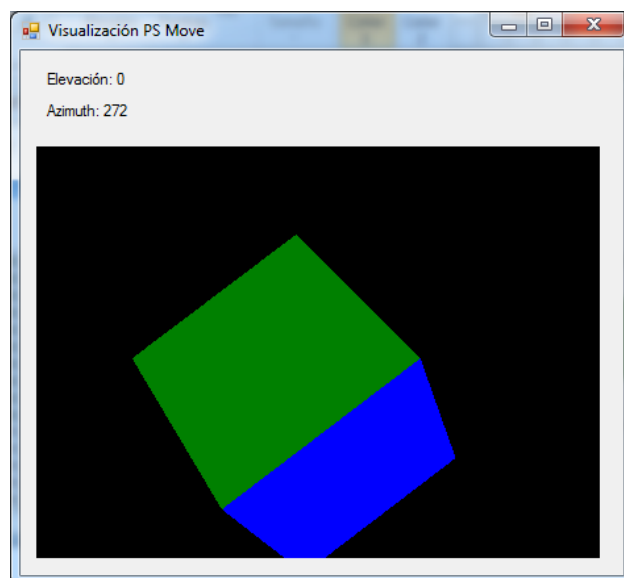


Figura E.3. Ventana de visualización de PS Move

La ventana de visualización (Figura E.3) representa gráficamente la orientación del dispositivo de seguimiento mediante la presentación en pantalla de un cubo tridimensional cuya rotación coincide con la del dispositivo.

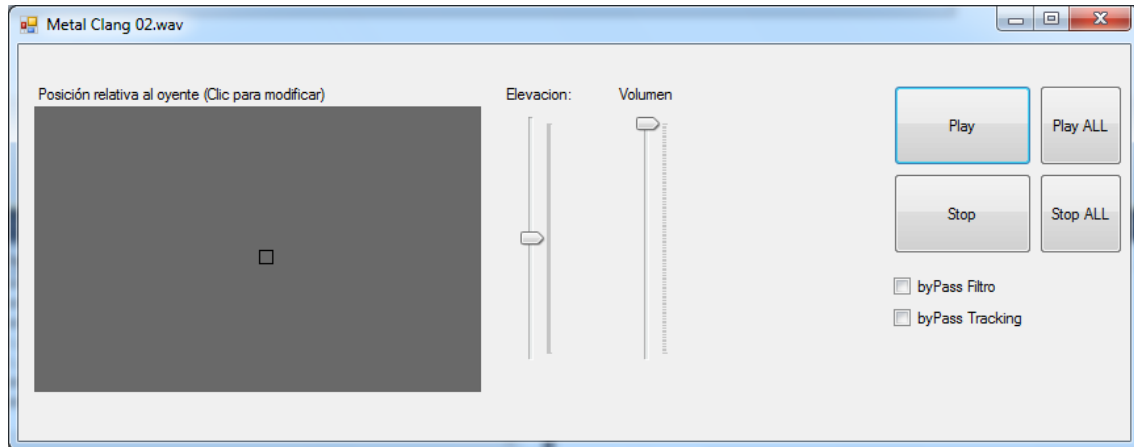


Figura E.4. Ventana de unidad de HRTF

Por último, la ventana de unidad de HRTF (Figura E.4) maneja la clase `HRTFUnit`, responsable de la reproducción y filtrado de los sonidos. Cada ventana corresponde a uno de los sonidos. En la parte izquierda se puede colocar el sonido relativo al oyente en el plano superior (sólo azimut) haciendo clic en el panel. Se puede establecer el valor de la elevación así como del volumen del sonido mediante los controles deslizantes centrales. El control de bypass de filtro desactiva el filtrado HRTF, y el control de bypass de tracking desactiva el seguimiento de movimiento. Los botones Play y Stop comienzan y detienen respectivamente la reproducción del sonido. Los botones Play ALL y Stop ALL actúan sobre todas las ventanas de unidades HRTF abiertas en el momento.

Se comentan a continuación algunos requisitos necesarios para poder ejecutar la aplicación.

El dispositivo PS Move debe estar encendido y emparejado antes de iniciar la aplicación. Para emparejar el dispositivo es necesario seguir los pasos descritos en la documentación del proyecto MoveFramework.

La cámara PS Eye debe estar conectada. Para usar la cámara es necesario instalar los drivers CL Eye Driver disponibles también en la página del proyecto MoveFramework.

APÉNDICE F: Pruebas subjetivas

Se presentan a continuación las tablas de resultados pertenecientes a las pruebas de todos los sujetos que han participado en ellas.

En cada tabla se pueden encontrar los siguientes datos:

- Número de sujeto, características del sujeto (sexo y edad)
- Colección seleccionada
- Puntuaciones de localización en las 8 direcciones de azimut a 0° de elevación.
- Puntuaciones de localización en elevaciones superior e inferior a 60° y 345°.
- Valoración global del sujeto y comentarios.

Sujeto			Características			Colección	
Sujeto 1			Hombre, 28			1056	
0	45	90	135	180	225	270	315
4	5	5	5	5	5	5	5
Superior				Inferior			
4				2			
Prueba 1			Prueba 2			Prueba 3	
5			5			5	
Valoración global		Comentarios					
9		Existe cierta dificultad en localizar sonidos por debajo de la línea de visión, sobretodo en posiciones frontales y traseras.					

Tabla F.1. Resultados Sujeto 1

Sujeto			Características			Colección		
Sujeto 2			Mujer, 28			1025		
0	45	90	135	180	225	270	315	
1	2	5	5	4	5	5	2	
Superior				Inferior				
5				4				
Prueba 1			Prueba 2			Prueba 3		
5			5			5		
Valoración global		Comentarios						
8		La colección que más real parecía al sujeto no funcionaba bien en la zona frontal.						

Tabla F.2. Resultados Sujeto 2

Sujeto			Características			Colección	
Sujeto 3			Mujer, 64			1053	
0	45	90	135	180	225	270	315
1	3	5	4	5	5	5	4
Superior				Inferior			
4				4			
Prueba 1		Prueba 2			Prueba 3		
5		5			5		
Valoración global		Comentarios					
10		La simulación ha funcionado mucho mejor con el seguimiento de movimiento					

Tabla F.3. Resultados Sujeto 3

Sujeto			Características			Colección	
Sujeto 4			Hombre, 67			1009	
0	45	90	135	180	225	270	315
5	2	5	3	5	4	5	2
Superior				Inferior			
3				3			
Prueba 1		Prueba 2			Prueba 3		
5		5			5		
Valoración global		Comentarios					
10		La simulación ha funcionado mucho mejor con el seguimiento de movimiento					

Tabla F.4. Resultados Sujeto 4

Sujeto			Características			Colección	
Sujeto 5			Hombre, 27			1028	
0	45	90	135	180	225	270	315
3	4	5	5	5	5	3	3
Superior				Inferior			
3				4			
Prueba 1		Prueba 2			Prueba 3		
5		5			5		
Valoración global		Comentarios					
8							

Tabla F.5. Resultados Sujeto 5

Sujeto			Características			Colección		
Sujeto 6			Mujer, 21			1056		
0	45	90	135	180	225	270	315	
2	3	5	5	5	3	5	5	
Superior				Inferior				
5				3				
Prueba 1		Prueba 2			Prueba 3			
5		4			5			
Valoración global		Comentarios						
8		El sujeto no supo encontrar ninguna colección apropiada, pero aun así en la prueba con seguimiento de movimiento fue capaz de localizar el sonido.						

Tabla F.6. Resultados Sujeto 6

Sujeto			Características			Colección		
Sujeto 7			Hombre, 31			1021		
0	45	90	135	180	225	270	315	
1	3	5	5	5	5	5	3	
Superior				Inferior				
3				2				
Prueba 1		Prueba 2			Prueba 3			
5		5			5			
Valoración global		Comentarios						
8								

Tabla F.7. Resultados Sujeto 7

Sujeto			Características			Colección	
Sujeto 8			Hombre, 29			1009/1026	
0	45	90	135	180	225	270	315
5	3	5	5	5	5	4	3
Superior				Inferior			
1				5			
Prueba 1		Prueba 2			Prueba 3		
5		4			5		
Valoración global		Comentarios					
9							

Tabla F.8. Resultados Sujeto 8

Sujeto			Características			Colección		
Sujeto 9			Hombre, 33			1033		
0	45	90	135	180	225	270	315	
3	5	5	4	3	5	5	4	
Superior				Inferior				
4				3				
Prueba 1		Prueba 2			Prueba 3			
5		5			5			
Valoración global		Comentarios						
8								

Tabla F.9. Resultados Sujeto 9

