AURALIZACIÓN EN TIEMPO REAL: IMPLEMENTACIÓN DEL MODELO DEL OYENTE

FABIÁN C. TOMMASINI, MARIANO ARANEDA Y OSCAR A. RAMOS

Centro de Investigación y Transferencia en Acústica. Unidad Asociada del CONICET. Universidad Tecnológica Nacional, Facultad Regional Córdoba. Córdoba, Argentina.
ftommasini@gmail.com

Resumen – *Se está desarrollando un sistema experimental dinámico e interactivo de realidad acústica virtual (AVR en inglés) basado en una arquitectura de computadora de propósito general. Este trabajo introduce dicho sistema que combina flexibilidad, escalabilidad, modularidad y un diseño orientado a objetos, con la estabilidad proporcionada por un sistema operativo en tiempo real. Se describe la estrategia usada para lograr que un kernel GNU/Linux permita ejecutar tareas en hard real-time.*
En esta primera etapa, se presenta la implementación del modelo del oyente, basado en respuestas impulsivas de cabeza (HRIRs en inglés), utilizando tres técnicas diferentes: convolución discreta, convolución por medio de FFT y filtros IIR binaurales obtenidos por el método iterativo de Steiglitz-McBride. Se discuten las ventajas y limitaciones de cada una y se realiza un análisis comparativo para determinar cuál logra el mejor balance entre precisión y eficiencia computacional. Además, como las HRIRs que se disponen están medidas con menos resolución espacial que la que un ser humano es capaz de discriminar (la diferencia mínima notable en el hemisferio frontal es del orden de un grado) se hace necesario calcular posiciones intermedias por medio de una interpolación bilineal. Esto permite al oyente percibir cambios suaves mientras la fuente o la cabeza se mueven.

Abstract – *A dynamic and interactive experimental acoustic virtual reality (AVR) system is being developed based on a general-purpose computer architecture. This paper introduces this system that combines flexibility, scalability, modularity and an object-oriented design, with stability provided by a real-time operating system. The strategy used to achieve hard real time tasks in a GNU/Linux kernel is described.*
At this first stage, the listener model implementation, based on head-related impulse responses (HRIRs), using three different techniques is presented: discrete convolution, FFT convolution and binaural IIR filters designed using the Steiglitz-McBride iterative method. Advantages and limitations of each one are discussed and a comparative analysis to determine which achieves the best balance between accuracy and computational efficiency is presented. Moreover, as HRIRs are measured with a less spatial resolution than a human being can discriminate (the just-noticeable difference in the frontal hemisphere is about one degree), it is necessary to calculate intermediate positions with a bilinear interpolation. This allows to the listener to perceive smooth changes while the sound source or head are moved.

1. INTRODUCCIÓN

El principio que sustenta la simulación acústica fue enunciado por Kleiner, Dalenbäck y Svensson [1], en 1993, introduciendo por primera vez el concepto de *auralización*. Este concepto indica que si, mediante algún medio de reproducción de sonidos, se aplican a los tímpanos de un oyente las señales biológicamente correctas, será posible estimular en el mismo la sensación de presencia o inmersión en el entorno modelado. El término auralización se refiere a una *representación auditiva* del espacio.

Los campos de investigación relacionados con la auralización son multidisciplinarios y requieren conocimientos sobre física acústica, acústica de recintos, procesamiento digital de señales, sistemas de tiempo real y psicoacústica. Los diversos abordajes para materializarla difieren según sea su aplicación, pudiéndose señalar en general tres tipos: a) el físico-acústico [2]; b) el perceptual [3][4] y c) el físico-perceptual [5][6].

En los primeros trabajos se buscaba una representación auténtica y totalmente fiel de la realidad en un entorno virtual, pero en contraposición requerían un alto costo computacional. En lugar de ello, actualmente las investigaciones se han inclinado hacia el desarrollo de entornos *plausibles*, donde se persigue lograr una percepción semejante más que las mismas características físicas-acústicas del ambiente

real. Entonces, el objetivo es sintetizar sonidos mediante métodos físico-matemáticos que, al ser escuchados por un oyente, produzcan en él la sensación de inmersión en el espacio simulado. Pellegrini sugiere que no se puede alcanzar una "auténtica auralización" y que su calidad debe ser evaluada desde la perspectiva de su aplicación para realizar una tarea en particular (plausibilidad) [4].

Hablamos de realidad acústica virtual (acoustic virtual reality, AVR) si este proceso se realiza en *tiempo real* (real-time), de manera que el usuario pueda *interactuar* con el entorno modelado. Así surgen los sistemas de realidad acústica virtual (AVR system, AVRS) con tres premisas fundamentales: plausibilidad, ejecución en tiempo real e interactividad. La interactividad se refiere al hecho de que el sistema tenga la capacidad de responder a los eventos generados por el usuario (como un movimiento de su cabeza) y que, de esta manera, éste intuitivamente se comporte de una forma muy similar a como lo haría en un ambiente real.

El propósito del equipo de trabajo es el de desarrollar un AVRS que se ejecute sobre una arquitectura de computadora de propósito general (hardware y software estándares). Para alcanzar este objetivo tres subsistemas deben ser modelados: a) la fuente sonora; b) el recinto y el medio de propagación; y c) el oyente o receptor. El abordaje que se sigue es el de lograr modelos simplificados basados en evidencias perceptuales factibles de operar en tiempo real, o sea, se apela a simplificaciones físico-acústicas *perceptualmente aceptables*. En el presente trabajo, se presenta la implementación del modelo del oyente.

Si bien se han desarrollado sistemas similares [7][8][9], cada abordaje es diferente. No se conocen antecedentes de sistemas que se diseñaron a partir de las implicancias perceptuales auditivas de un oyente.

Para poder llevar a cabo esta primera etapa de implementación en tiempo real del AVRS, se han desarrollado con anterioridad una serie programas basados en scripts de MATLAB: a) un simulador de recintos, con el cual es posible calcular la RIR (room impulse response) y visualizar la trayectoria de los rayos del método de la fuente-imagen [10]; b) un módulo para calcular los parámetros acústicos especificados en la norma ISO 3382:1997 [11] que permite comparar RIRs medidas y sintetizadas; y c) un software que incorpora varios modelos del oyente de diversos niveles de complejidad junto a la administración de pruebas psicoacústicas para poder evaluarlos perceptualmente [12][13].

El diseño del AVRS está basado en las ventajas del diseño orientado a objetos, como la flexibilidad, la escalabilidad y la modularidad. Se han extraído algunas ideas de implementación generales introducidas por Scarpaci [14].

A continuación se presentan los principales aspectos teóricos involucrados, tanto del modelo del oyente como de los sistemas en tiempo real. Luego se describe una etapa preliminar de procesamiento de datos, para después detallar aspectos de la implementación de tres técnicas para el modelo del oyente: convolución discreta, convolución por medio de FFT (fast Fourier transform) y filtros IIR (infinite impulse response) binaurales obtenidos por el método iterativo de Steiglitz-McBride. Para finalizar se realiza una comparativa del rendimiento de las mismas y una discusión acerca de cuál logra una mejor relación entre precisión y eficiencia computacional.

2. EL MODELO DEL OYENTE

Las ondas producidas por una fuente sonora son modificadas, antes de alcanzar los tímpanos de un oyente, por la cabeza, el torso y el oído externo. Las respuestas impulsivas de cabeza (head-related impulse response, HRIR) o las funciones de transferencia de cabeza (head-related transfer function, HRTF)[1] en el dominio de la frecuencia, caracterizan dichas transformaciones. Estas son diferentes para cada oído y varían sistemáticamente con la ubicación de la fuente sonora en el espacio. Por esto las HRIRs son medidas entre posiciones diferentes de una fuente sonora y con micrófonos miniatura colocados en la proximidad de los tímpanos. Las HRIRs contienen las claves fundamentales que un ser humano utiliza para localizar una fuente sonora ubicada en el espacio que lo circunda, como son: a) la diferencia de tiempo interaural (interaural time difference, ITD), que tiene un rol dominante en bajas frecuencias; b) la diferencia interaural de niveles (interaural level difference, ILD), que hace lo propio para altas frecuencias [3]; y c) los perfiles espectrales.

Si se considera a las HRIRs como las salidas de un sistema lineal e invariante en el tiempo (linear time-invariant, LTI), entonces, en teoría, pueden ser representadas por filtros no recursivos de respuestas finitas (finite impulse response, FIR). Aunque se debe notar que el problema no es trivial. La idea de modelar las HRTFs utilizando filtros digitales data del año 1989 [15][16]. En estos estudios pioneros, los autores comprobaron que los sujetos podían localizar fuentes sonoras virtuales (sintetizadas mediante filtros FIR) usando auriculares con la misma precisión con que localizaban fuentes sonoras reales.

Con anterioridad, Mehrgardt y Mellert [17], en 1977, habían demostrado que las HRTFs pueden considerarse secuencias de fase-mínima y que el *resto de fase* es casi lineal según la posición e independiente de la frecuencia, y está asociado al ITD. Por lo tanto, las HRTFs pueden ser descompuestas en un sistema de fase-mínima y en un

[1] De aquí en adelante se las nombrará de forma indistinta, salvo que se aclare lo contrario.

sistema "all-pass" [18]. Esto permitió desarrollar un modelo simplificado conocido como: fase-mínima-más-retardo (minimum-phase-plus-delay).

Algunos estudios psicoacústicos posteriores han corroborado que la habilidad de los sujetos para localizar una fuente sonora no se degrada si se utilizan HRTFs generadas por este modelo [19][20]. Sin embargo, otros estudios más recientes demuestran que la fase de las HRTFs para bajas frecuencias (hasta 1500 Hz) cumple un rol dominante y que asumir que es lineal, sobre todo para el oído contralateral, conduce a errores insalvables en aplicaciones críticas [21][22].

Por otra parte, en un estudio que se ha realizado con anterioridad [13], se construyeron filtros IIR de orden 20 para un conjunto de HRIRs. Los coeficientes del filtro se obtuvieron por el método iterativo de Steiglitz-McBride [23], y con tan sólo 5 iteraciones del método se obtuvieron coeficientes que lograban muy buenos resultados. Existían errores mínimos tanto de magnitud como de fase en el plano vertical, y las diferencias de ITD e ILD en el rango de frecuencias preponderantes (172-15000 Hz), están por debajo de las mínimas diferencias notables (just-noticeable difference, jnd).

2.1 Interpolación de HRIRs

Se ha demostrado que la gente adulta puede distinguir entre dos fuentes sonoras separadas por 1º en el hemisferio frontal [3][24]. Sin embargo, es prácticamente imposible medir las HRIRs con esta densidad espacial. Debido al tiempo que toma realizar la medición en cada posición, es muy difícil lograr que un sujeto experimental se encuentre sentado en la misma posición con la cabeza quieta por un largo período de tiempo. Por esto, la mayoría de los conjuntos de datos de HRIRs no son medidas con la suficiente resolución espacial para crear una percepción suave cuando se cambia de una HRIR a otra. Al momento de reproducir un sonido continuo y realizar un cambio de HRIRs, generalmente se escuchan clics audibles en ciertos estímulos.

Para incrementar la densidad espacial se realiza un post-procesamiento de los datos. Se calculan posiciones intermedias no medidas en el conjunto original de las HRIRs, mediante algún tipo de interpolación.

Estudios psicoacústicos realizados por Langendijk y Bronkhorst [25], en 2000, sobre la influencia de la resolución de las HRTFs medidas y la dirección de la interpolación (tanto en el plano horizontal, vertical, como diagonal) aplicando interpolación lineal, encontraron que una resolución espacial en las HRTFs medidas de alrededor 6º no introduce claves audibles, y una resolución entre 10º y 15º introduce sólo algunas claves espectrales. Entonces, las respuestas intermedias pueden ser calculadas por interpolación.

3. SISTEMAS EN TIEMPO REAL

Cuando se discute acerca de la ejecución en tiempo real de un sistema, la idea subyacente es que un evento dado tenga una respuesta *a tiempo*. Esto no significa que la respuesta sea rápida, sino simplemente que sea *lo suficientemente rápida* en el contexto en el cual opera el sistema. Por esto, la principal diferencia existente entre una tarea que se ejecuta en tiempo real y una que no, es que la primera tiene un tiempo límite o máximo (deadline) en el cual debe completar su ejecución. Cuando el tiempo de respuesta es crítico, se utiliza lo que se llama "hard real-time" (HRT), donde el resultado, vencido el deadline, es erróneo. Pero existe también el "soft real-time" (SRT), que admite, ocasionalmente, superar el deadline. Por lo que suele utilizarse en tareas no críticas. Un sistema operativo de tiempo real (real-time operating system, RTOS) debe garantizar que se cumplan todas estas restricciones críticas de tiempo.

En los últimos años, el crecimiento de Linux (y del software libre) ha hecho que se convierta casi en un estándar de facto en la experimentación científica computacional, lo que generó mucho interés la posibilidad de adaptar este sistema operativo de propósito general en un RTOS.

En general, existen dos abordajes para lograrlo. El primero implica *parchar* el kernel para hacerlo más "preemtible"[2], lograr menores latencias en las interrupciones, mejores tiempos en los cambios de contexto, menor latencia en la planificación y tener mayor precisión en el temporizador (timer). El segundo abordaje implica usar un nano-kernel de tiempo real (en un enfoque de kernel dual) que ejecuta el kernel de Linux como una tarea de baja prioridad, mientras administra las interrupciones del hardware subyacente y sólo redirecciona hacia el kernel Linux las que son relevantes para el mismo [26].

Si bien existen ventajas y desventajas en cada uno de estos abordajes, dentro del software libre el más popular es el del nano-kernel. Permite mayor control de las latencias y, por lo tanto, del determinismo. Dentro de este abordaje existen dos extensiones "open source" para Linux: RTAI[3] (Real-Time Application Interface) [27] y Xenomai[4]. Estudios comparativos realizados muestran que RTAI logra un mejor rendimiento (menor "delay" y menor "jitter") que Xenomai, e inclusive que otras alternativas privativas-comerciales [28].

Tanto RTAI como Xenomai están basados en el nano-kernel Adeos[5] (Adaptive Domain Environment

[2] Permitir que determinados procesos se ejecuten con máxima prioridad de forma que puedan interrumpir a cualquier otro proceso de menor prioridad en el acceso a los recursos que necesiten.

[3] https://www.rtai.org

[4] http://www.xenomai.org

[5] http://home.gna.org/adeos/

for Operating Systems). Este nano-kernel se sitúa bajo el sistema operativo Linux en ejecución (llamado dominio) y permite que múltiples sistemas operativos compartan el mismo entorno de hardware [29][30].

Entonces, cuando se cargan los módulos de tiempo real de RTAI, se registra un nuevo dominio y comienza a administrar las interrupciones de hardware. Propaga, mediante el nano-kernel Adeos hacia el kernel de Linux, sólo las interrupciones que no son de interés para las tareas de tiempo real (Figura 1).

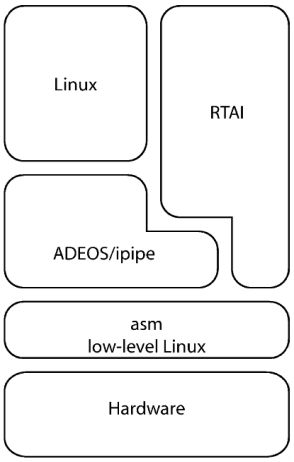

Figura 1: Estructura de capas en RTAI (extraído de [28])

Actualmente, en el desarrollo del AVRS se está usando LXRT. LXRT (LinuX Real-Time) es un módulo de RTAI que extiende su funcionalidad al espacio de usuario[6]. Permite ejecutar tareas en HRT en el espacio de usuario usando la misma API (application programming interface) provista por RTAI para el espacio de kernel. Las ventajas son significativas: permite definir en el mismo programa algunas tareas que se ejecutan en tiempo real y otras que no se ejecutan en tiempo real; y permite comunicar a las tareas en tiempo real con procesos de Linux estándares.

En el contexto del AVRS, es necesario utilizar un RTOS con la capacidad de resolver los modelos dentro de lo que Wenzel [31] denomina una latencia perceptualmente tolerable, menor a los 100 ms. En este tiempo el sistema debe ser capaz de actualizar el *escenario acústico*. Esto es, producir las señales binaurales correctas según la ubicación y orientación de la cabeza del usuario y la posición y ubicación de la fuente sonora en el espacio modelado. Todo esto teniendo en cuenta que los dispositivos externos conectados a la computadora (como un sensor de movimientos de cabeza) contribuyen con sus propias latencias y velocidades de actualización.

El RTOS utilizado está compuesto por un sistema operativo Linux con un kernel original (vanilla kernel) 2.6.28.9, modificado con la extensión RTAI 3.7.1. Esta extensión incluye el módulo de LXRT.

4. HARDWARE DEL SISTEMA

Actualmente el AVRS se ejecuta sobre una computadora que se utiliza como servidor dedicado (servidor de auralización). Posee un procesador Intel Core 2 Quad Q8200 (arquitectura x86-64 de 4 núcleos) que permite ejecutar tareas en paralelo, 4 GB de memoria RAM DDR2 800 MHz, una interfaz de red Gigabit Ethernet para comunicarse con la computadora de desarrollo, y una placa de sonido Intel de hasta 24 bits de resolución.

Se cuenta, además, con un sensor de movimientos Polhemus Patriot[7] de 6 grados de libertad (X, Y, Z, azimut, elevación, rol) con una latencia media de 17 ms enviando 60 muestras por segundo. Para la reproducción de los sonidos sintetizados se utilizan auriculares Sennheiser HD 570.

Debido a que el hardware utilizado es estándar, cualquier componente puede ser reemplazado y actualizado sin afectar al resto de los componentes. Esto le permite al sistema ser completamente escalable.

5. ETAPA PRELIMINAR

5.1 HRIRs utilizadas

La base de datos de HRIRs que se ha utilizado para realizar las pruebas fue la del CIPIC (Center for Image Processing and Integrated Computing) [32] de la Universidad de California. La cual consiste en mediciones realizadas a 45 sujetos (incluidas dos mediciones a un maniquí KEMAR) en 1250 posiciones diferentes para cada oído. La ubicación de la fuente sonora está dada por el ángulo de azimut θ (25 posiciones) y el ángulo de elevación Φ (50 posiciones) referenciados a un eje que pasa por ambos oídos denominado eje inter-aural (Figura 2). Las HRIRs fueron medidas entre la fuente sonora y micrófonos miniaturas colocados a las entradas de los conductos auditivos bloqueados. La longitud de una HRIR es de 200 muestras en el dominio del tiempo, con una frecuencia de muestreo de 44.1 kHz.

[6] El espacio de usuario es el área de memoria donde se ejecutan las aplicaciones del usuario, a diferencia del espacio de kernel, que es el área de memoria reservado para ejecutar el kernel y los módulos de kernel que hacen al funcionamiento interno de un sistema operativo.

[7] http://www.polhemus.com/?page=Motion_Patriot

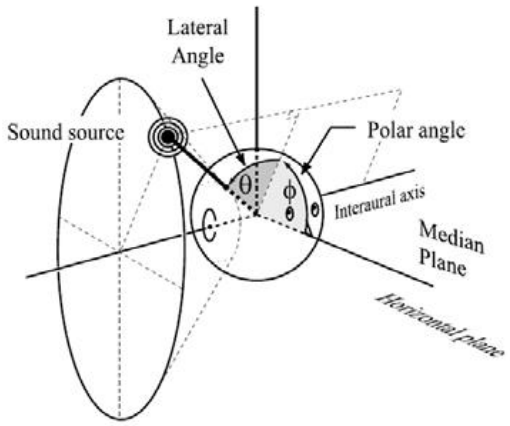

Figura 2: Sistema de coordenadas utilizado por el conjunto de HRIRs del CIPIC (extraído de [32])

Las mediciones se realizaron para ángulos de elevación que varían uniformemente en pasos de 5.625° entre -45° y +230.625°. Mientras que las mediciones para ángulos de azimut se realizaron para: -80°, -65°, -55°, desde -45° a +45° en pasos de 5°, +55°, +65° y +80°. Los valores de azimut 0° y elevación 0°, corresponde al frente; azimut 0° y elevación 180° detrás del sujeto; ángulos de azimut negativos a la izquierda y positivos a la derecha del sujeto; los valores de elevación negativos y mayores a 180° corresponden a posiciones debajo del eje inter-aural, al frente y detrás respectivamente.

Debido que para esta etapa de implementación no es relevante la realización de estudios psicoacústicos, sino, en su lugar, el correcto funcionamiento del sistema, se ha utilizado solamente el conjunto de HRIRs medidas a un sujeto tomado al azar.

5.2 Pre-procesamiento de las HRIRs

Como se dijo, al conjunto de las HRIRs se le debe incrementar la densidad espacial, calculando posiciones intermedias no medidas. Primero se las prepara eliminando, por una parte, el retardo inicial (común a ambos oídos) dado por la distancia entre la fuente y los micrófonos; y por la otra, el retardo de cada par izquierda-derecha de HRIRs en particular, dado por las ITDs. Este último retardo se guarda para su posterior utilización.

Para lograr una mejor resolución en todo el espacio que circunda al sujeto, se decidió estimar las HRIRs en posiciones igualmente espaciadas. Los valores de azimut fueron distanciados 1.25º uno de otro, entre -80º a +80º. En cambio los valores de elevación fueron distanciados 2.8125º uno de otro, entre -45º y +230.625º. Dando como resultado un total de 12771 posiciones (previamente se contaba con 1250) para cada oído, con 129 posiciones en azimut y 99 posiciones en elevación. Estas estimaciones corresponden a valores intermedios de azimut y elevación a los medidos por el CIPIC.

Es necesario interpolar dos conjuntos de datos. Por un lado, las HRIRs y, por el otro, las ITDs extraídas. El método de interpolación que se ha utilizado es el de interpolación bilineal, también llamada escalado por inverso de la distancia, y se la ha realizado en el dominio del tiempo.

Suponiendo que se quiere encontrar un valor desconocido de una *HRIR* para la posición $P = (\Phi, \theta)$ para un tiempo determinado t (ya sea oído izquierdo o derecho indistintamente) (Figura 3), y conociendo los valores para las posiciones $Q_{11} = (\Phi_1, \theta_1)$, $Q_{12} = (\Phi_1, \theta_2)$, $Q_{21} = (\Phi_2, \theta_1)$, y $Q_{22} = (\Phi_2, \theta_2)$, se procede de la siguiente manera.

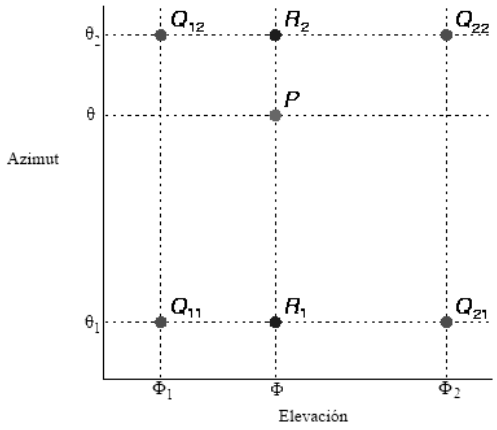

Figura 3: Puntos a calcular en una interpolación bilineal (adaptado de [33])

Se realizan dos interpolaciones lineales en una misma dirección, para los puntos $R_1 = (\Phi, \theta_1)$ y para $R_2 = (\Phi, \theta_2)$

$$HRIR(R_1,t) \approx \frac{\Phi_2 - \Phi}{\Phi_2 - \Phi_1} HRIR(Q_{11},t) + \frac{\Phi - \Phi_1}{\Phi_2 - \Phi_1} HRIR(Q_{21},t) \quad (1)$$

$$HRIR(R_2,t) \approx \frac{\Phi_2 - \Phi}{\Phi_2 - \Phi_1} HRIR(Q_{12},t) + \frac{\Phi - \Phi_1}{\Phi_2 - \Phi_1} HRIR(Q_{22},t) \quad (2)$$

Luego se procede a realizar otra interpolación lineal, en la otra dirección, para el punto P

$$HRIR(P,t) \approx \frac{\theta_2 - \theta}{\theta_2 - \theta_1} HRIR(R_1,t) + \frac{\theta - \theta_1}{\theta_2 - \theta_1} HRIR(R_2,t) \quad (3)$$

Obteniéndose así el valor de la *HRIR* para un instante de tiempo *t*. Se debe proceder de la misma

manera para el resto de las muestras y para cada uno de los oídos.

De una forma similar se interpolan los valores de las ITDs. El cálculo se realiza para las mismas posiciones de azimut y elevación descriptas. En este caso, el resultado de la interpolación es independiente del tiempo *t* y se realiza sólo una vez por cada par izquierda-derecha de HRIRs.

Todos los cálculos han sido realizados con MATLAB. Este conjunto de datos, el cual se lo ha llamado "HRIRS procesadas", se lo utiliza posteriormente en la implementación.

6. IMPLEMENTACIÓN

En la Figura 4 se encuentra un diagrama esquemático de los módulos principales del AVRS, así como sus datos de entrada. La programación en tiempo real se ha realizado con C/C++, y hasta el momento se han implementado los módulos sombreados en la figura: *captura de movimiento* (ejecutándose en SRT), *modelo del oyente* (HRT) y *convolución en tiempo real* (HRT).

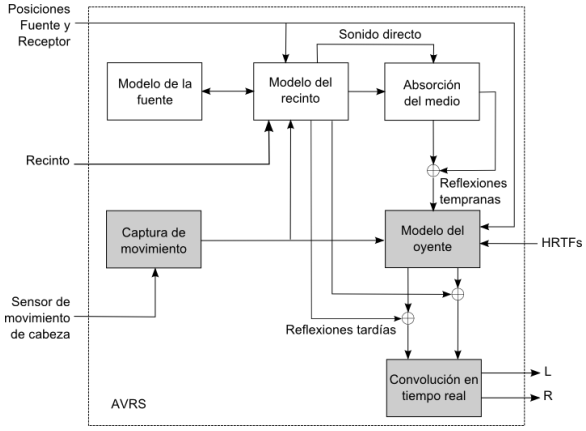

Figura 4: Diagrama esquemático de los principales módulos del AVRS

Explicado brevemente, el módulo *captura de movimiento* mantiene en un "buffer" los datos obtenidos a partir del sensor de movimiento. En esta etapa del desarrollo se han simulado los movimientos de cabeza mediante datos previamente grabados en un archivo, lo que permite independizarse del sensor. Mientras que el módulo *convolución en tiempo real* se encarga de generar la auralización (mediante convolución por FFT) y de enviar los datos para su reproducción en tiempo real.

A continuación, se detalla la implementación del *modelo del oyente* con tres técnicas diferentes: convolución discreta (filtro FIR), convolución por FFT y filtrado con filtros IIR binaurales obtenidos por el método iterativo de Steiglitz-McBride.

6.1 Método de solapamiento-suma

El método de "overlap-add" provee una manera eficiente de calcular la convolución de una señal muy larga (o infinita) $x[n]$ con una respuesta impulsiva $h[n]$ (en este caso una HRIR) de M cantidad de muestras. Se basa en el concepto de dividir el problema en múltiples convoluciones de $h[n]$ con segmentos cortos de la señal $x[n]$. El nombre está dado porque, para cada segmento de salida, $y_i[n]$ e $y_{i+1}[n]$ se solapan $(M - 1)$ muestras, para luego sumarse [18][34].

Se ha implementado este método para ambas técnicas de convolución, lo que permite la utilización de señales infinitas de entrada.

6.2 Técnica de convolución discreta

La convolución discreta de dos señales $x[n]$ y $h[n]$ se define de la siguiente manera

$$y[n] = x[n] * h[n] = \sum_{k=0}^{N-1} h[k]x[n-k] \qquad (4)$$

donde N es la cantidad de muestras de la señal $x[n]$.

La suma de convolución (Ecuación 4) se puede implementar fácilmente de una manera similar a la de un filtro FIR. Esta implementación requiere N multiplicaciones por cada punto de salida.

Para la implementación del módulo *modelo del oyente* con esta técnica se ha utilizado este algoritmo:

i. Obtener la posición de la cabeza, por medio del módulo *captura de movimiento*.
ii. Buscar la posición pre-calculada (*ajustada*) más próxima a la posición de la cabeza (*original*). Este algoritmo busca, a partir de la posición original, una posición ajustada (en azimut y elevación). Por ejemplo, si se debe filtrar una señal con la HRIR de la posición original 7º en azimut y -28º en elevación, el filtrado se realizará con las HRIRs calculadas para la posición ajustada de 7.5º en azimut y -28.125º en elevación.
iii. Recuperar de memoria el par izquierda-derecha de HRIRs en el dominio del tiempo para dicha posición ajustada, y colocar los valores en un buffer de cálculo.
iv. Obtener los valores del buffer de entrada (dominio del tiempo).
v. Realizar la convolución discreta utilizando el método de overlap-add, obteniéndose así 512 muestras para cada oído.
vi. Enviar los datos al buffer de salida.

Para esta técnica es necesario contar con las HRIRs completas en el dominio del tiempo. Entonces, se agregaron nuevamente los retardos de las ITDs previamente guardados, ya sean los medidos o los interpolados. No se agregaron los retardos comunes iniciales a todo el conjunto.

También se quitaron las muestras con valor nulo comunes al final del conjunto de HRIRs, quedando reducida la cantidad de muestras a 183 (en lugar de las 200 originales). Esto permite un ahorro de memoria y de procesamiento.

Al conjunto resultante se lo ha llamado "HRIRs para convolución" (utilizado como entrada en el paso iii del algoritmo) y se lo ha guardado en un archivo binario. El módulo *modelo del oyente* es el encargado de interpretarlo y cargarlo en memoria antes del comienzo de la simulación.

Como ya se verá, la convolución discreta es menos eficiente computacionalmente que la convolución por FFT. Por esto, sólo se la ha mencionado a modo comparativo. Su uso en la práctica está muy condicionado.

6.3 Técnica de convolución por FFT

Una técnica más eficiente es la de la transformada discreta de Fourier (discrete Fourier transform, DFT) para realizar la convolución en el dominio de la frecuencia. Procesando segmentos de N muestras, se calcula la DFT de $x[n]$, se realiza la multiplicación compleja de su espectro con el espectro de $h[n]$, y se calcula la DFT inversa. Esta forma de implementación es más eficiente debido al uso del algoritmo de FFT para implementar la DFT. La FFT requiere $O(\log N)$ multiplicaciones por cada punto de salida.

Para calcular la convolución por FFT, se ha usado la librería open source FFTW (Fastest Fourier Transform in the West)[8] [35] para calcular la DFT. Esta librería incorpora diversas optimizaciones del algoritmo de la FFT, y es conocida por ser una de las implementaciones más rápidas que existen.

Se ha utilizado la DFT con entrada real y salida compleja, que implementa el siguiente cálculo

$$X_k = \sum_{n=0}^{N-1} x_n e^{\frac{-2\pi i}{N} kn} \qquad k = 0, \ldots, N/2 \tag{5}$$

donde i es la unidad imaginaria, x es un vector real de tamaño N, y X es un vector complejo de tamaño $(N/2+1)$. Este vector de salida aprovecha la simetría Hermitiana donde $X_k = X_{n-k}^*$. De esta manera la mitad de la salida es redundante (una mitad es compleja conjugada de la otra mitad). Esto agiliza la multiplicación en el dominio de la frecuencia.

Al momento de realizar el proceso inverso, se ha utilizado la DFT inversa con entrada compleja y salida real. En este caso se supone que X es un vector complejo de tamaño $(N/2+1)$, o sea de sólo la mitad de los valores debido a la simetría Hermitiana. Entonces, la transformada se define como

$$x_n = \sum_{k=0}^{N/2} X_k e^{\frac{2\pi i}{N} kn} \qquad n = 0, \ldots, N-1 \tag{6}$$

Como se puede apreciar, la DFT inversa está desnormalizada. Se debe realizar a posteriori una normalización de los datos.

Para la implementación del módulo *modelo del oyente* con esta técnica se ha utilizado este algoritmo:

i. Obtener la posición de la cabeza, por medio del módulo *captura de movimiento*.
ii. Buscar la posición pre-calculada (ajustada) más próxima a la posición de la cabeza (original).
iii. Recuperar de memoria el par izquierda-derecha de HRTFs en el dominio de la frecuencia para dicha posición ajustada, y colocar los valores en un buffer de cálculo.
iv. Obtener los valores del buffer de entrada (dominio del tiempo).
v. Calcular la DFT de los valores de entrada, para pasarlos al dominio de la frecuencia.
vi. Realizar la convolución por FFT (multiplicación compleja y normalización) utilizando el método de overlap-add, obteniéndose así 512 muestras para cada oído.
vii. Enviar los datos al buffer de salida.

Para esta técnica es necesario contar con las HRTFs en el dominio de la frecuencia. Para esto se ha utilizado el conjunto de "HRIRs para convolución" y se ha calculado "off-line" la DFT a cada una. El resultado de esta operación son 348 valores complejos (almacenados como 348 × 2 valores reales) para cada HRTF individual.

A este conjunto de datos se lo ha llamado "HRTFs para convolución" (utilizado como entrada en el paso iii del algoritmo), el cual se ha guardado en un archivo binario. El módulo *modelo del oyente* es el encargado de interpretarlo y cargarlo en memoria antes del comienzo de la simulación.

6.4 Técnica de filtros IIR binaurales

Como se dijo, en un estudio anterior [13], se lograron resultados perceptualmente aceptables utilizando esta técnica, tanto en fase como en magnitud. Se utiliza el método iterativo de Steiglitz-McBride para encontrar los coeficientes de un filtro IIR a partir de una respuesta impulsiva. Más detalles se pueden encontrar en [23].

En la implementación que se ha realizado, los filtros IIR son calculados por la librería open source

[8] http://www.fftw.org

STK (Synthesis ToolKit)[9] [36], optimizada para su utilización en aplicaciones de audio en tiempo real.

Para la implementación del módulo modelo del oyente con esta técnica se ha utilizado este algoritmo:

i. Obtener la posición de la cabeza, por medio del módulo captura de movimiento.
ii. Buscar la posición pre-calculada (ajustada) más próxima a la posición de la cabeza (original).
iii. Recuperar de memoria el par izquierda-derecha de los coeficientes del filtro que representan las HRIRs, para dicha posición ajustada, y colocar los valores en un buffer de cálculo.
iv. Recuperar de memoria el valor de ITD correspondiente a la posición ajustada (si es positivo corresponde a un retraso de la señal en el oído izquierdo, si es negativo un retraso en la señal en el oído derecho).
v. Obtener los valores del buffer de entrada (dominio del tiempo).
vi. Realizar el filtrado por IIR de la señal, obteniéndose así 512 muestras para cada oído.
vii. Agregar el retardo al oído que corresponda.
viii. Enviar los datos al buffer de salida.

Para calcular los coeficientes del filtro a partir de las HRIRs utilizando el método iterativo de Steiglitz-McBride, se ha utilizado el conjunto de "HRIRs procesadas". Se han realizado 10 iteraciones del método (5 más que las del estudio de referencia, aumentando la precisión), y se obtuvieron 21 coeficientes para el numerador y 21 coeficientes para para el denominador, por cada HRIR individual. A estos datos se agregaron las ITDs interpolados calculados con anterioridad, dando como resultado al conjunto "Coeficientes-ITDs para filtro" (utilizado como entrada en los pasos iii y iv del algoritmo). A este conjunto se lo guardó en un archivo binario. El módulo modelo del oyente es el encargado de interpretarlo y cargarlo en memoria antes del comienzo de la simulación

7. COMPARACIÓN DE RENDIMIENTO

A modo de comparar el rendimiento, en la Tabla 1 se presentan los tiempos de ejecución del módulo *modelo del oyente* del AVRS, utilizando las técnicas presentadas.

Estos valores corresponden al tiempo promedio que se demora en calcular 512 muestras para cada oído (512 × 2) el *módulo del oyente*. Incluyen la ejecución de los pasos previamente descriptos.

También, a modo de referencia, se muestra la cantidad de valores (correspondientes a los diferentes conjuntos de entrada) que debe procesar cada técnica en el buffer de cálculo. Se puede apreciar que esta cantidad no es directamente proporcional al tiempo de ejecución de cada técnica.

Técnica	Cantidad de valores de entrada a procesar	Tiempo de ejecución [ms]
Convolución discreta	183 × 2 = 366	1.0696
Convolución por FFT	348 × 2 × 2 = 1392	0.2943
Filtros IIR binaurales	21 × 2 × 2 = 84	0.0556

Tabla 1: Comparativa de rendimiento de las técnicas implementadas

8. DISCUSIÓN

En el presente trabajo se ha presentado una primera etapa de implementación del modelo del oyente en el contexto del AVRS. Se compararon tres técnicas diferentes para dicho modelo: convolución discreta, convolución por medio de FFT y filtros IIR binaurales obtenidos por el método iterativo de Steiglitz-McBride.

Los resultados permitieron reafirmar que la convolución discreta no es una implementación viable cuando se requieren resultados en tiempo real. La implementación de la convolución por FFT es un 72.5% más rápida, produciendo exactamente los mismo resultados. En cuanto a la implementación usando los filtros IIR binaurales, los tiempos son muy buenos, es un 94.8% más rápida que la convolución discreta, e inclusive un 81.1% más rápida que la convolución por FFT.

Entonces, si se desea tener la máxima precisión a un costo computacional relativamente bajo, la convolución por FFT presentada es una buena alternativa. En cambio, si el propósito es lograr resultados perceptualmente aceptables con alta eficiencia computacional, la técnica de filtros IIR se convierte en la mejor opción. Los errores registrados por los estudios anteriores arrojan valores por debajo de las jnd en ITD e ILD en frecuencias preponderantes, y errores mínimos tanto en magnitud como en fase. En definitiva, logra el mejor balance entre precisión y eficiencia computacional.

Las mediciones presentadas arrojaron que el tiempo que le demanda al módulo *modelo del oyente*, usando la técnica de filtros IIR, calcular 512 muestras para un par izquierda-derecha de HRIRs, es de 0.0556 ms. Este valor representa tan sólo el 0.47% del tiempo que tarda en reproducirse dicha cantidad de muestras mediante auriculares (11.6099 ms a una frecuencia de muestreo de 44.1 kHz).

Este escaso tiempo de cálculo resulta alentador, por lo que los futuros trabajos deben seguir indagando en la implementación de modelos que tengan un bajo costo computacional y que brinden resultados perceptualmente aceptables.

[9] https://ccrma.stanford.edu/software/stk/

9. REFERENCIAS

[1] M. Kleiner, B.I. Dalenbäck, and P. Svensson, *"Auralization-an overview,"* Journal of the Audio Engineering Society, vol. 41, pp. 861–875. 1993.

[2] J.H. Rindel, *"Evaluation of room acoustic qualities and defects by use of auralization,"* Journal of the Acoustical Society of America, vol. 116, p. 2483. 2004.

[3] J. Blauert, *Spatial hearing: the psychophysics of human sound localization*. MIT Press, Cambridge, MA, USA. 1997.

[4] R.S. Pellegrini, *"A virtual reference listening room as an application of auditory virtual environments,"* PhD thesis, Ruhr-University. 2002.

[5] E.M. Wenzel, J.D. Miller, and J.S. Abel, *"Sound Lab: A real-time, software-based system for the study of spatial hearing,"* Proceedings of the 108th Convention of the Audio Engineering Society. Paris, Francia. 2000.

[6] L. Savioja, T. Lokki, and J. Huopaniemi, *"Auralization applying the parametric room acoustic modeling technique-the DIVA auralization system,"* Proceedings of the 8th International Conference on Auditory Display. 2002.

[7] L. Savioja, J. Huopaniemi, T. Lokki, and R. Vaananen, *"Creating interactive virtual acoustic environments,"* Journal of the Audio Engineering Society, vol. 47, pp. 675–705. 1999.

[8] T. Lentz, D. Schröder, M. Vorländer, and I. Assenmacher, *"Virtual reality system with integrated sound field simulation and reproduction,"* EURASIP Journal on Advances in Signal Processing, vol. 2007, pp. 1-17. 2007.

[9] M. Noisternig, B.F. Katz, S. Siltanen, and L. Savioja, *"Framework for Real-Time Auralization in Architectural Acoustics,"* Acta Acustica united with Acustica, vol. 94, pp. 1000-1015. 2008.

[10] F.C. Tommasini and O.A. Ramos, *"Modelo basado en evidencias perceptuales para simular respuestas impulsivas de recintos,"* Conferencia Latinoamericana de Informática (CLEI2008). Santa Fe, Argentina. 2008.

[11] F.C. Tommasini, O.A. Ramos, and S. Ferreyra, *"Comparación objetiva de respuestas impulsivas de recintos medidas y simuladas,"* VI Congreso Iberoamericano de Acústica (FIA2008). Buenos Aires, Argentina: 2008.

[12] O.A. Ramos, G. Calvo, and F.C. Tommasini, *"Modelo Acústico de Cabeza y Torso Mediante Análisis de Componentes Principales,"* XVI Congreso sobre Métodos Numéricos y sus Aplicaciones. ENIEF2007, pp. 46-58. Asociación Argentina de Mecánica Computacional. Córdoba, Argentina. 2007.

[13] O.A. Ramos, M. Araneda, and F.C. Tommasini, *"Diseño y evaluación de filtros binaurales,"* XVIII Congreso sobre Métodos Numéricos y sus Aplicaciones. ENIEF2009, pp. 137-148. Asociación Argentina de Mecánica Computacional . Tandil, Argentina. 2009.

[14] J.W. Scarpaci, *"Creation of a system for real time virtual auditory space and its application to dynamic sound localization,"* PhD thesis, Boston University, 2006.

[15] F.L. Wightman and D.J. Kistler, *"Headphone simulation of free-field listening. I: Stimulus synthesis,"* The Journal of the Acoustical Society of America, vol. 85, pp. 858-867. 1989.

[16] F.L. Wightman and D.J. Kistler, *"Headphone simulation of free-field listening. II: Psychophysical validation,"* The Journal of the Acoustical Society of America, vol. 85, pp. 868-878. 1989.

[17] S. Mehrgardt and V. Mellert, *"Transformation characteristics of the external human ear,"* The Journal of the Acoustical Society of America, vol. 61, pp. 1567-1576. 1977.

[18] A.V. Oppenheim and R.W. Schafer, *Digital signal processing*, Prentice-Hall, 1975.

[19] D.J. Kistler and F.L. Wightman, *"A model of head-related transfer functions based on principal components analysis and minimum-phase reconstruction,"* The Journal of the Acoustical Society of America, vol. 91, pp. 1637-1647. 1992.

[20] A. Kulkarni, S.K. Isabelle, and H.S. Colburn, *"Sensitivity of human subjects to head-related transfer-function phase spectra,"* The Journal of the Acoustical Society of America, vol. 105, pp. 2821-2840. 1999.

[21] J.W. Scarpaci and H.S. Colburn, *"Principal components analysis interpolation of head related transfer functions using locally-chosen basis functions,"* Proceedings of 11th Meeting of the International Conference on Auditory Display. Limerick, Irlanda. 2005.

[22] P. Zahorik, P. Bangayan, V. Sundareswaran, K. Wang, and C. Tam, *"Perceptual recalibration in human sound localization: Learning to remediate front-back reversals,"* The Journal of the Acoustical Society of America, vol. 120, pp. 343-359. 2006.

[23] K. Steiglitz and L. McBride, *"A technique for the identification of linear systems,"* IEEE Transactions on Automatic Control, vol. 10, pp. 461–464. 1965.

[24] A.W. Mills, *"On the Minimum Audible Angle,"* The Journal of the Acoustical Society of America, vol. 30, pp. 237-246. 1958.

[25] E.H.A. Langendijk and A.W. Bronkhorst, *"Fidelity of three-dimensional-sound reproduction using a virtual auditory display,"* The Journal of the Acoustical Society of America, vol. 107, pp. 528-537. 2000.

[26] N. Vun, H.F. Hor, and J.W. Chao, *"Real-Time Enhancements for Embedded Linux,"* 14th IEEE International Conference on Parallel and Distributed Systems, 2008. ICPADS'08, pp. 737–740. 2008.

[27] P. Mantegazza, E.L. Dozio, and S. Papacharalambous, *"RTAI: Real Time Application Interface,"* Linux J., vol. 2000, p. 10. 2000.

[28] A. Barbalace, A. Luchetta, G. Manduchi, M. Moro, A. Soppelsa, and C. Taliercio, *"Performance comparison of VxWorks, Linux, RTAI, and Xenomai in a hard real-time application,"* IEEE Transactions on Nuclear Science, vol. 55, pp. 435–439. 2008.

[29] R. Bucher and S. Balemi, *"Scilab/Scicos and Linux RTAI - a unified approach,"* Proceedings of 2005 IEEE Conference on Control Applications, 2005. CCA 2005, pp. 1121–1126. 2005.

[30] K. Yaghmour, *"Adaptive domain environment for operating systems,"* Opersys inc. 2001.

[31] E.M. Wenzel, *"The impact of system latency on dynamic performance in virtual acoustic environments,"* Proceedings of the 16th I International Congress of Acoustics and 135th Meeting of the Acoustical Society of America, p. 180. Seattle, WA, USA. 1998.

[32] V.R. Algazi, R.O. Duda, D.M. Thompson, and C. Avendano, *"The CIPIC HRTF database,"* Proceedings of 2001 IEEE Workshop on Applications of Signal Processing to Audio and Electroacoustics, pp. 99–102. New Paltz, NY, USA. 2001.

[33] Wikipedia, the free encyclopedia, *"Bilinear interpolation"*. Jun, 2010.

[34] M.H. Hayes, *Schaum's outline of theory and problems of digital signal processing*, McGraw-Hill Professional, 1999.

[35] M. Frigo and S.G. Johnson, *"The design and implementation of FFTW3,"* Proceedings of the IEEE, pp. 216–231. 2005.

[36] G.P. Scavone and P.R. Cook, *"RtMidi, RtAudio, and a Synthesis Toolkit (STK) update," Proceedings of the 2005 International Computer Music Conference*, pp. 1-4. Barcelona, España. 2005.