XXX ENCONTRO DA SOBRAC 2023
SOCIEDADE BRASILEIRA DE ACÚSTICA
19 a 22 de novembro de 2023
Natal - RN

Máquina de auralização dinâmica aplicada à comparação de HRTFs em tempo real

Carvalho, D. R.[1]; **Fonseca, W. D'A.**[2]; **Mello, F. R.**[2]

[1] Audioscenic Limited, Southampton, Hampshire, Inglaterra, davi.carvalho@audioscenic.com

[2] Engenharia Acústica (EAC) & Programa de Pós-Graduação Arquitetura, Urbanismo e Paisagismo (PPGAUP), Universidade Federal de Santa Maria, Santa Maria, RS, Brasil, {will.fonseca, felipe. mello}@eac.ufsm.br

Resumo

Auralização é o processo de criação ou captura de um cenário auditivo e é comumente associada à audição biauricular, que permite a localização de fontes sonoras no espaço. Este artigo apresenta ferramentas projetadas para facilitar a geração de auralizações adaptativas com base na posição do ouvinte, permitindo comparações em tempo real entre Funções de Transferência Relacionadas à Cabeça (HRTFs). São abordados os principais algoritmos para convolução por partes, como o *Overlap-Add* (OLA), *Overlap-Save* (OLS) e *Uniformly Partitioned Overlap-Save* (UPOLS), discutindo suas vantagens e desvantagens em termos de eficiência computacional e qualidade de áudio resultante. Suas implementações computacionais são também aclaradas, visto que oferecem recursos de convolução em tempo real computacionalmente competentes. O trabalho também destaca a técnica de transição suave entre filtros convoluídos (*crossfading*), para garantir uma reprodução contínua e natural do som. Além disso, é discutido um sistema de rastreamento da posição da cabeça do ouvinte (*wireless*) utilizando imagens de vídeo e inteligência artificial, que permite a adaptação da auralização à posição do ouvinte em tempo real. Por fim, é apresentado um sistema que integra ambas as ferramentas — chamado de *máquina de auralização* —, destinado a estudos de validação subjetiva, como a análise de métodos de individualização HRTF ou a comparação de projetos de acústica de salas. Neste estudo, o funcionamento adequado do projeto foi revelado por meio de um teste-piloto. Cabe ainda salientar que as ferramentas são distribuídas gratuitamente e podem ser melhoradas de forma colaborativa pela comunidade científica. Observa-se ainda que as ferramentas e técnicas apresentadas neste trabalho são fundamentais para a criação de ambientes sonoros verossímeis em aplicações como jogos, realidade virtual, cinema e produção musical.

Palavras-chave: auralização, convolução, filtros FIR, overlap, rastreador de cabeça, teste subjetivo.

Dynamic auralization machine applied to real-time HRTF comparison

Abstract

Auralization is the process of creating or capturing an auditory scene and is commonly associated with binaural hearing, which allows the localization of spatial sound sources. This paper presents tools designed to facilitate the generation of adaptive auralizations based upon listener position, allowing real-time comparisons between Head-Related Transfer Functions (HRTFs). The main algorithms for partitioned convolution, such as Overlap-Add (OLA), Overlap-Save (OLS), and Uniformly Partitioned Overlap-Save (UPOLS) are discussed. Their advantages and disadvantages in terms of computational efficiency and resulting audio quality are also elaborated. Moreover, their computational implementations are also elucidated, as they offer computationally competent real-time convolution features. The study also highlights the technique of smooth transition between convoluted filters (*crossfading*) to ensure continuous and natural sound reproduction. Furthermore, a (*wireless*) system for tracking the listener's head position using video images and artificial intelligence is discussed, which allows the adaptation of the auralization to the listener's position in real-time. Finally, a system that integrates both tools, referred to as the *auralization engine*, is presented for subjective validation studies, such as the analysis of HRTF individualization methods or the comparison of room acoustics designs. In this study, the proper functioning of the project was revealed through a pilot test. It should also be noted that the tools are distributed for free and can be collaboratively improved by the scientific community. Furthermore, it is observed that the tools and techniques presented in this work play a fundamental role in the creation of realistic sound environments in applications such as gaming, virtual reality, cinema, and music production.

Keywords: auralization, convoltion, FIR filters, overlap, head tracker, subjective testing.

1. Introdução

Auralização[1] é o processo de criar ou capturar um cenário auditivo, seja por meio de sinais sintéticos ou gravados. Ela está comumente associada à audição biauricular, a qual permite a localização de fontes sonoras posicionadas no espaço tridimensional.

A tecnologia biauricular (*coração da auralização*) funciona sobre o princípio de que humanos possuem dois receptores (um em cada canal auditivo). Logo, a reprodução de qualquer *evento sonoro* que ocorre naturalmente no mundo real pode ser recriada posteriormente ao entregar a cada orelha de um ouvinte os sinais aproximadamente exatos e distintos que caracterizam o fenômeno que se deseja reproduzir.

A primeira etapa para a geração de um sinal biauricular a partir de uma fonte sonora (de um canal, por exemplo) é a atribuição de sua localização espacial. Tal processo é possível por meio da convolução entre o sinal mono e um par de funções de transferência que caracterizem o *caminho* entre uma fonte sonora no espaço e a entrada dos canais auditivos de um ouvinte. Tais funções de transferência são tradicionalmente denominadas por *Head-Related Transfer Functions* (HRTFs ou ainda Funções de Transferência Relacionadas à Cabeça) [1]. Ao aplicar HRTFs a um sinal qualquer e reproduzir posteriormente por um método que garanta a entrega dos sinais diretamente à entrada dos canais auditivos de um ouvinte (seja por meio de fones de ouvido ou por sistemas de cancelamento de *crosstalk* [2]), o receptor irá perceber a fonte sonora na *mesma orientação espacial* em que a HRTF foi originalmente gravada.

É necessário ainda observar que esse procedimento, quando ainda convoluído com a *Resposta Impulsiva* (RI) mono de uma sala, gera uma *primeira aproximação* para o cenário auditivo, se comparado com a mesma cena sendo medida de forma biauricular (na sala). A medição irá conter tanto a interação do ouvinte com os objetos que o cercam (cadeira e mesas, por exemplo, mudando a coloração espectral) quanto partes de Campo Reverberante proveniente de todas as direções — diferentemente dessa *primeira aproximação* supramencionada. Outrossim, ainda que incompleta, consegue gerar a sensação de posicionamento de uma fonte sonora na direção pretendida e dentro de um ambiente.

A convolução é uma operação matemática que expressa como uma função $f(t)$ modifica uma segunda função $g(t)$ [3]. Na prática essa é a operação que permite a caracterização de como um sinal é modificado por um sistema (linear e invariante no tempo) qualquer. A convolução entre dois sinais no domínio do tempo $f(t)$ e $g(t)$ é definida pela integral do produto entre os sinais após um deles ser espelhado em relação a um dos eixos (vertical, por exemplo) e deslocado sobre o segundo sinal tal que,

$$(f * g)(t) := \int_{-\infty}^{\infty} f(\tau) g(t-\tau)\, \mathrm{d}\tau\,. \tag{1}$$

em que $*$ representa a operação de convolução linear.

A partir desses fundamentos um possível próximo passo é a *auralização dinâmica*. Nessa modalidade o ouvinte pode interagir com o cenário acústico que está envolvido, tal que ao movimentar-se os sinais biauriculares são também modificados, de forma que as funções de transferência referentes à nova posição relativa entre a fonte sonora virtual e a entrada dos canais auditivos do ouvinte sejam correspondentes ao que seria observado em um ambiente real.

Para garantir os pré-requisitos de interação em uma auralização em *tempo real*, o processo de convolução definido na Equação (1) é representado por algoritmos de convolução rápida por partes, que realizam a operação com sinais discretos, finitos e causais, além de permitir a adaptação das funções de transferência uma vez que o ouvinte se move no cenário virtual.

Nesse contexto, este trabalho apresenta o desenvolvimento de uma *máquina de auralização (dinâmica) em tempo real* (via convolução) que permite de forma simples alternar entre bancos de dados de HRTFs diferentes. Com isso, é possível comparar o desempenho de ouvintes na tarefa de localização sonora 3D, considerando os detalhes intrínsecos de cada conjunto de HRTFs.

[1] A *auralização* compreende um conjunto de técnicas utilizadas para *tornar algo audível*, é análoga à *visualização*, para *tornar algo visível*. A nomenclaturada usada neste artigo segue o neologismo da palavra inglesa *auralization*. Na literatura em português ainda é encontrada como *aurilização* ou *audibilização*.

Na Seção 2 deste artigo são apresentados os três algoritmos para o processo de *convolução por partes*[2] utilizados neste desenvolvimento. São discutidas algumas vantagens e desvantagens de cada um dos métodos, bem como seus desempenhos. Ademais, na Seção 2.4 apresenta-se um procedimento para a transição suave entre os filtros (*crossfading*).

O processo de adaptação à posição do ouvinte é realizado por um rastreador de cabeça *não tátil* (*wireless*) [4]. Faz-se uso de visão computacional (utilizando *inteligência artificial*) para estimar a orientação do ouvinte no mundo real e, consequentemente, selecionar as HRTFs correspondentes à posição relativa entre ouvinte e fonte sonora virtual — demais detalhes estão delineados na Seção 3.

Na Seção 4 a arquitetura e aplicação dessas ferramentas desenvolvidas são pormenorizadas. O seu funcionamento conjunto concretiza o *software* chamado de *máquina para auralização dinâmica*, batizado de `multiHRTFrenderer`. Essa ferramenta tem por objetivo ser utilizada no processo de validação subjetiva (espacial) de um algoritmo de individualização de HRTFs.

Por fim, a Seção 5 concluiu com as considerações finais do trabalho. Aclara-se ainda que no repositório do projeto no GitHub encontram-se informações adicionais que apoiam usuários iniciantes (ou mestres, no contexto de ensino), indicando os passos da instalação e códigos simples (em Matlab e Python) para aprendizado.

2. Convolução por partes

A *convolução por partes* descreve o processo de convolução em que o áudio a ser convoluído com a resposta ao impulso (desejada) é dividido em múltiplas partes, as quais são convoluídas individualmente com a resposta ao impulso. Esse tipo de particionamento é comum no processamento de áudio em tempo real dado que a interface de áudio lê e reproduz os sinais em blocos. Quando esse processo de captura, processamento e reprodução ocorrem rápidos o suficiente (e sem que haja percepção do receptor), diz-se que o processamento ocorre em *tempo real*.

Nesta seção são descritos três métodos para a convolução por partes com *filtros de resposta finita* (FIR) [3,5, 6]: *Overlap-Add* (OLA) [7], *Overlap-Save* (OLS) [8] e *Uniformly Partitioned Overlap-Save* (UPOLS) [9]. Esses métodos são matematicamente equivalentes[3], logo quaisquer diferenças entre os resultados são apenas erros numéricos ou de arredondamento de máquina. Ao fim, as principais diferenças encontram-se na eficiência computacional de cada algoritmo.

2.1 *Overlap-Add* (OLA)

Dado um bloco de áudio de tamanho[4] B em amostras (ou *samples*) — representando[5] $x_1(n)$ —, faz-se a *convolução circular* (comumente denotada pelo símbolo $\circledast$) com a resposta ao impulso ($h(n)$) de tamanho N em amostras. Considerando que o resultado da convolução linear deve possuir um tamanho K de pelo menos $\{B+N-1\}$ amostras (a fim de evitar *aliasing* temporal) e que o *tamanho do bloco* de entrada e saída de um dispositivo de áudio é normalmente fixo e de mesmo tamanho[6], o resultado da convolução de um bloco de entrada com a resposta ao impulso sempre será maior que o tamanho esperado pelo dispositivo de áudio quando bloco de saída e entrada têm o mesmo tamanho. Com isso, ao usar o algoritmo de OLA as amostras restantes são guardadas, e somadas ao resultado da convolução do bloco seguinte com a resposta ao impulso, que por sua vez também terá amostras em excesso que serão adicionadas ao

[2]Em português encontra-se ainda esse procedimento como *convolução em blocos* (*block convolution*), *convolução em seções* (*convolution by sectioning*) ou ainda *método de convolução por sobreposição* (ou *superposição*) — em inglês seria *Overlap-*{*Add*, *Save*} ou *Overlap and* {*Add*, *Save*}. Neste artigo são apresentados os métodos: *Overlap-Add* (sobreposição e adição), *Overlap-Save* (sobreposição e armazenamento) e *Uniformly Partitioned Overlap-Save* (partição uniforme em sobreposição e armazenamento).

[3]Resultados comparativos do desempenho estão apresentados na Seção 2.5.

[4]Por exemplo, um *áudio completo* teria um tamanho $C = 20B$ amostras, se foi seccionado em $M = 20$ partes de tamanho B.

[5]A versão amostrada e quantizada de $x(t)$ é representada por $x(n)$.

[6]Esse valor é normalmente descrito como uma potência de 2 (dois), que de acordo com a taxa de amostragem utilizada pode ser maior ou menor desde que a latência causada pelo tamanho do bloco não cause percepção de atraso. Para uma amostragem de 48 kHz geralmente esse valor é menor que 1024.

bloco posterior e assim sucessivamente. O diagrama na Figura 1 descreve o processo de convolução com *Overlap-Add* para aplicação de tempo real — a RI é representada pelo filtro de N amostras (ou *taps*).

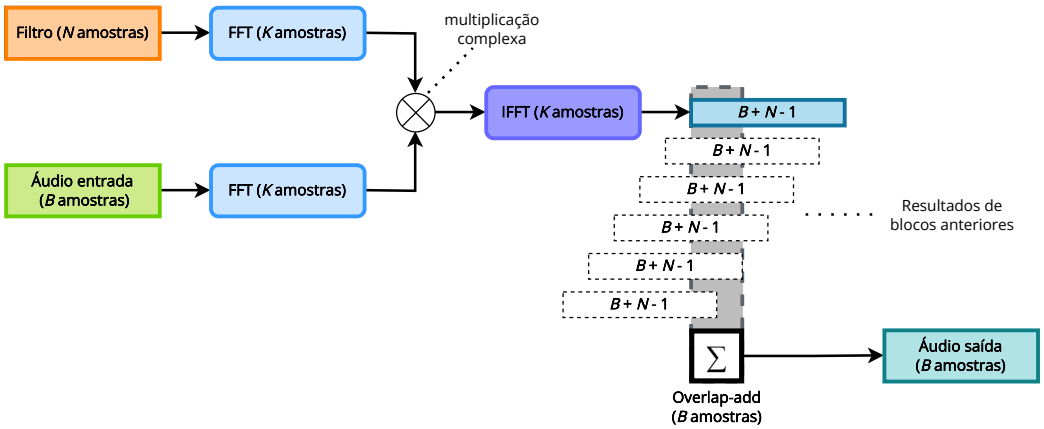


Figura 1: Diagrama de convolução por meio de *Overlap-Add* (OLA) para saída *online* — adaptado de Wefers [9].

O algoritmo de OLA inicia-se (1) completando ambos os vetores com zeros de modo que tenham K amostras, com isso, (2) aplica-se o resultado na Transformada Rápida de Fourier[7] (FFT), transformando o bloco de áudio e da resposta ao impulso para o domínio da frequência. Faz-se (3) a multiplicação complexa entre os vetores, que corresponde a *convolução linear* no domínio do tempo. O resultado (4) é transformado de volta ao domínio do tempo (via IFFT[8]), correspondendo à saída $y_1(n)$.

Depois dessas etapas, existem dois caminhos: (i) escrever os resultados em um arquivo (para reprodução *offline*) ou (ii) entregar os resultados de forma *online* para um receptor, escrevendo-os em um *buffer*[9], com reprodução em tempo real — sendo este último o objetivo da ferramenta deste artigo.

Para o caso (i), as saídas consecutivas $y_m(n)$, sendo $m = \{1,\ldots,M\}$, são somadas tal que as últimas $(N-1)$ $y_m(n)$ são somadas com as primeiras $(N-1)$ amostras de $y_{m+1}(n)$, veja a Figura 2. Por exemplo, as saídas $y_1(n)$ e $y_2(n)$ são empilhadas, tal que o início de $y_2(n)$ é alinhado com as $(N-1)$ amostras finais de $y_1(n)$. Logo, as $(N-1)$ finais de $y_1(n)$ são somadas com as $(N-1)$ iniciais de $y_2(n)$ — esse processo continua até o término dos M blocos.

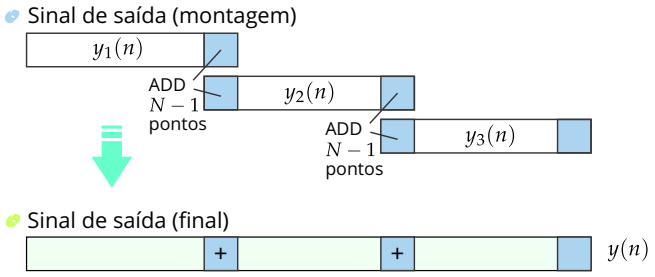


Figura 2: Convolução por *Overlap-Add* (OLA) para saída *offline* — adaptado de Fonseca [10].

Para a situação de tempo real (*online*, ii), B das K amostras resultantes são escritas no *buffer* de áudio de saída, as $(N-1)$ amostras restantes são salvas. Na próxima iteração, quando um novo bloco de áudio é recebido, repete-se o processo, porém as amostras iniciais do novo resultado B' são somadas às amostras restantes da iteração anterior B — seguindo a lógica da Figura 2.

2.2 *Overlap-Save* (OLS)

O algoritmo de *Overlap-Save* é similar ao *Overlap-Add*, porém em vez de acumular os resultados da convolução corrente com os resultados de convoluções anteriores, o bloco de entrada é salvo em um *buffer* circular de tamanho K com pelo menos $\{B+N-1\}$ amostras e a convolução a cada iteração se dá entre

[7] Do inglês, *Fast Fourier Transform* (FFT).
[8] Do inglês, *Inverse Fast Fourier Transform* (IFFT).
[9] O *buffer* se trata de um local de troca rápida e temporária de dados.

o *buffer* circular e a resposta ao impulso. Nesse *buffer* circular, as amostras anteriores são movidas B amostras para a esquerda e o novo bloco é alocado no final do *buffer*. Aqui, no entanto, o resultado da Transformada Inversa de Fourier (IFFT) contém componentes com *amostras temporais espúrias* (*time aliasing*), tal que apenas as B amostras no final do bloco são válidas, logo, o resto é descartado.

O diagrama na Figura 3 ilustra o processo de convolução por meio de *Overlap-Save*. Na prática, é necessário o mesmo número de operações de FFTs e IFFTs que no *Overlap-Add*, mas não é necessária a adição de blocos anteriores, o que torna sua eficiência computacional marginalmente melhor e mantém uma complexidade semelhante na implementação.

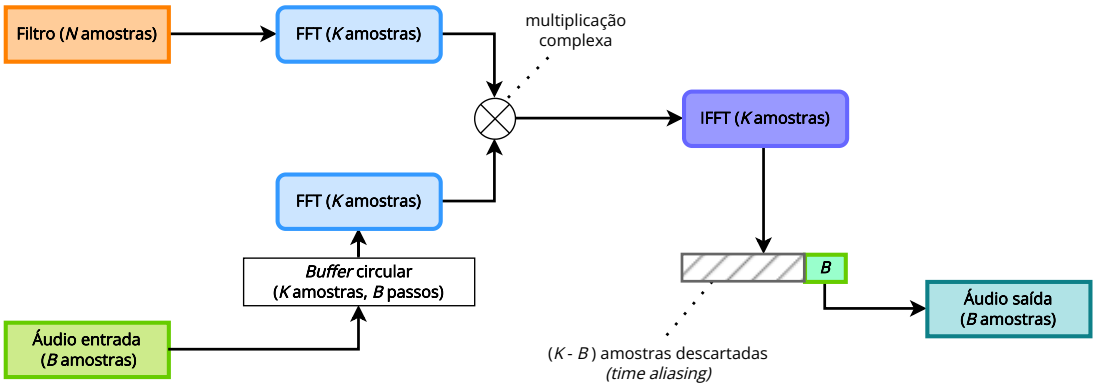


Figura 3: Diagrama de convolução por meio de *Overlap-Save* (OLA) — adaptado de Wefers [9].

2.3 *Uniformly Partitioned Overlap-Save* (UPOLS)

Até aqui foram apresentados casos em que apenas o áudio de entrada é dividido em partes, sendo essa uma abordagem padrão no universo do processamento em tempo real. Tal abordagem atende aos requisitos para reprodução em tempo real no contexto de auralização e HRTFs, uma vez que estas são funções de transferência que necessitam de um número relativamente pequeno de amostras para serem representadas adequadamente. Por outro lado, funções de transferência biauriculares em uma sala (BRIRs, *Binaural Room Impulse Responses*), que contemplam também o tempo de reverberação do ambiente, podem ter alguns segundos de duração. Nesse caso, a latência necessária para a convolução dessas respostas é da mesma ordem do tamanho das BRIRs, tornando ineficientes os métodos considerados previamente.

A fim de solucionar tal problema, pode-se particionar também a Resposta ao Impulso (RI) em partes iguais [11]. Essa abordagem substitui o uso de uma única FFT longa por várias FFTs mais curtas.

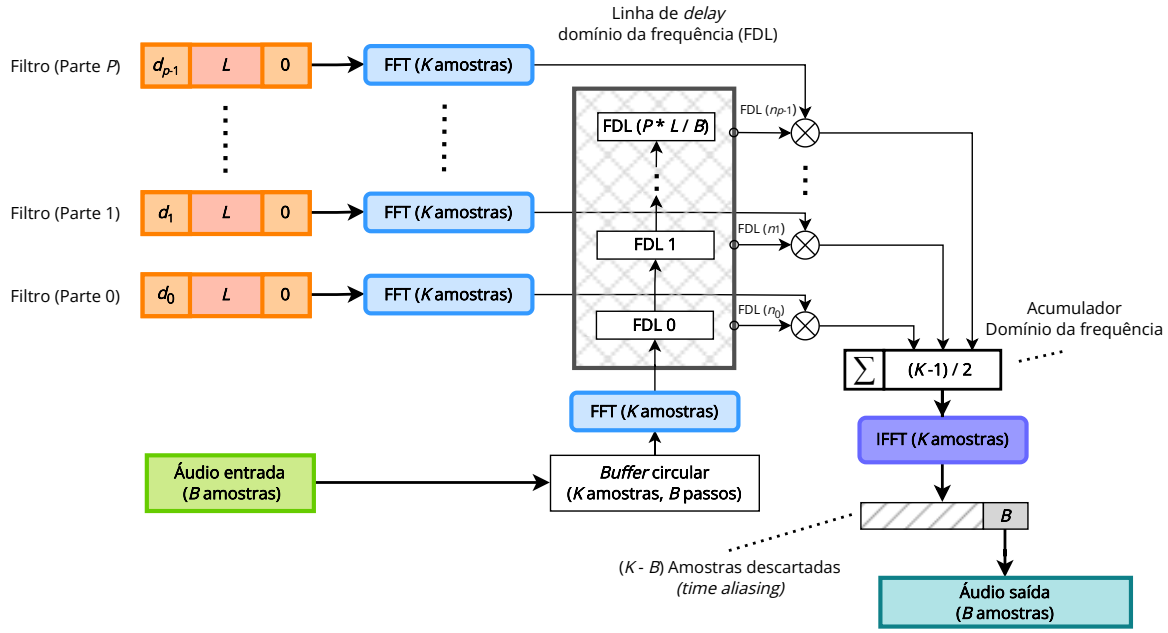


Figura 4: Diagrama de convolução por meio de *Uniformly Partitioned Overlap-Save* (UPOLS) — adaptado de Wefers [9].

A Figura 4 ilustra o processo de convolução por UPOLS (*Uniformly Partitioned Overlap-Save*). O filtro de tamanho N é dividido em P sub-partes de tamanho L. A cada uma dessas partes são adicionados zeros no início com o intuito de manter a relação intratemporal da resposta ao impulso. O número $d(p)$ de zeros para cada partição p é definido por $d(p) = \texttt{mod}(p \cdot L, B)$, em que a operação `mod` representa o resto da divisão do primeiro termo pelo segundo. Posteriormente, $\{K - L - d(p)\}$ zeros são adicionados ao final de cada partição a fim de garantir que todas tenham tamanho K. Em seguida, os valores são transformados para o domínio da frequência.

Assim como apresentado para o OLS, o bloco de áudio é avaliado dentro de um *buffer* circular, em que a cada novo bloco o *buffer* é *girado* para a esquerda e as novas amostras são adicionadas ao final. O resultado é então inserido em uma linha de *delay* com elementos no domínio da frequência (*Frequency-Domain Delay Line* ou FDL) [12] após os valores lá presentes serem movidos 1 (um) índice para cima. Note que, para a convolução, nem todas as partições são necessárias ao mesmo tempo, e a escolha de qual partição será utilizada depende da combinação entre o tamanho do bloco de áudio B, do tamanho da partição L e do tamanho necessário para convolução K. Assim, utiliza-se a FDL como mecanismo de gerenciamento de qual partição deve ser utilizada em determinado momento.

Todos os subfiltros são convoluídos com um bloco de entrada correspondente na FDL e os resultados são somados no domínio da frequência. O subfiltro p é multiplicado com o bloco q na FDL, de forma que $q(p) = \texttt{floor}(p \cdot L/B)$, em que a operação `floor` resulta no menor e mais próximo número inteiro. Por fim, o resultado acumulado é transformado de volta ao domínio do tempo e, assim como no OLS, apenas as B amostras ao final do vetor são mantidas e o resto é descartado por constituírem em um *aliasing* temporal.

Note que o tamanho das partições P e, consequentemente, o número de elementos K para convolução têm grande influência no desempenho do algoritmo. Neste estudo, adota-se como padrão o processo de otimização descrito por Wefers [13] como diretriz para seleção desses parâmetros. Essa otimização garante que o UPOLS será sempre a abordagem mais eficiente computacionalmente, mesmo para filtros curtos ($N < B$) uma vez que o UPOLS convergirá em um OLS.

2.4 *Crossfading*

A convolução dinâmica depende de uma atualização constante dos filtros que são convoluídos com o áudio de entrada. Aqui, especificamente, a troca de filtros corresponde à variação espacial das HRTFs. Desse modo, a medida em que o ouvinte se move, é possível manter o objeto virtual com uma *posição estática no espaço*, ou seja, fixar uma posição aparente entre a fonte (virtual) e o receptor, sem que existam falhas audíveis.

Em casos de testes de audição crítica, existem recomendações para tempos de transição adequada entre os objetos de análise. Por outro lado, no caso da auralização adaptativa, a meta é atualizar os filtros a uma velocidade em que o ouvinte não consiga *perceber* um atraso entre seu movimento e a reação do sistema de auralização, o que provoca instabilidade na posição relativa entre fonte e receptor. Geralmente tal instabilidade é uma limitação da velocidade de atualização do sistema de rastreamento da posição do ouvinte, e não uma limitação do processo de troca dos filtros em si.

O procedimento de atualização dos filtros utilizados pelos algoritmos de convolução por partes é denominado de *crossfading* (ou *transição cruzada*), dado que o procedimento de troca de filtros acontece por meio de um decaimento e ascensão simultâneos entre o *filtro corrente* e o *filtro futuro* [14], respectivamente tem-se

$$y_0(n) = x(n) * h_0(n) \qquad (2) \qquad \text{e} \qquad y_1(n) = x(n) * h_1(n)\,. \qquad (3)$$

O bloco de áudio de saída $y(n)$ é calculado pela soma ponderada entre os dois resultados. A função de ponderação $f(n)$ pode ter tamanho arbitrário L. No caso em que $L \leq B$ tem-se,

$$y(n) = \begin{cases} y_0(n) \cdot f_0(n) + y_1(n) \cdot f_1(n), & \text{caso } n \leq L, \\ y_1(n), & \text{caso } n > L\,. \end{cases} \qquad (4)$$

Caso $L > B$, a função f é particionada em L/B partes, tal que a última parte sempre será $L \leq B$.

Essa metodologia foi utilizada tanto no contexto da troca de filtros ao variar a posição relativa entre fonte e ouvinte, tal como para a troca de filtros quando o usuário solicita a auralização com diferentes perfis de HRTF, como detalhado na Seção 4.

2.5 Desempenho dos métodos de convolução

Burrus [8] demonstra e depois Wefers [9] sintetiza que algoritmos de convolução por partes são procedimentos de *baixa latência* para calcular a convolução linear por meio da convolução circular, enquanto se retorna resultados parciais à medida que o sinal é processado. Dessa forma, espera-se que qualquer diferença observada entre os métodos seja caracterizada apenas na ordem de precisão de máquina, ou seja, qualquer erro observado deve ter a mesma ordem da precisão dos números que calculou o erro, por serem uma consequência de imprecisão devido a arredondamentos na aritmética de ponto flutuante.

As diferenças entre a convolução linear no domínio do tempo e os métodos de convolução por partes descritos nas seções anteriores são apresentadas na Figura 5. A partir do resultado y_{L} da convolução entre dois sinais de *ruído branco* (por meio da convolução linear discreta), e o resultado y_{P} via convolução por partes, calculou-se o erro absoluto ao longo do tempo entre cada método de convolução por partes e a convolução linear discreta, tal que $\texttt{erro} = |y_{\mathrm{L}} - y_{\mathrm{P}}|$, em que $|\cdot|$ representada o valor absoluto.

Nesse artigo, foram utilizados números no formato `float64`, que possuem *resolução* na ordem de $1 \cdot 10^{-15}$ [15]. As diferenças ilustradas na Figura 5 estão na ordem de -200 dB (ref.: $2 \cdot 10^{-5}$ Pa) e, desse modo, podem ser consideradas inaudíveis.

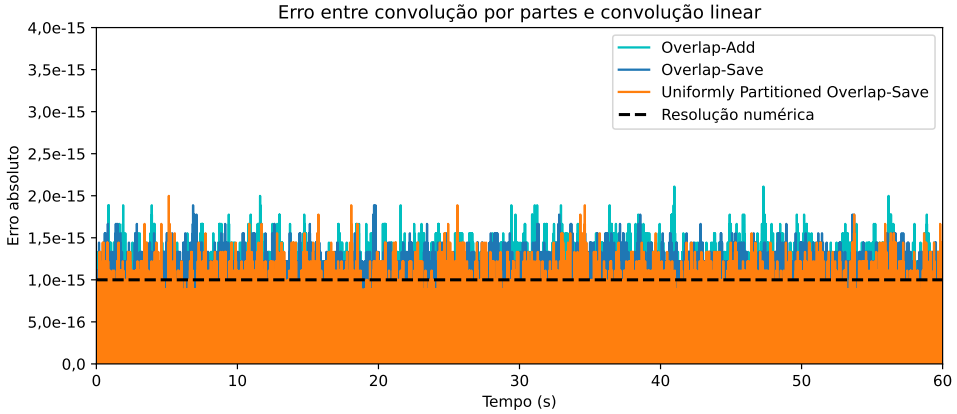


Figura 5: Erro entre métodos de convolução por partes e convolução linear.

3. Rastreador de cabeça

Sistemas de rastreamento de cabeça são essenciais para determinação da posição do ouvinte quando se tem por objetivo manter uma reprodução adaptativa em função da orientação relativa entre fontes virtuais e posição real do ouvinte.

O sistema apresentado neste trabalho é flexível para receber dados de posição de quaisquer tipos de rastreadores de cabeça[10] [4]. Neste estudo, optou-se pelo uso de um sistema de rastreamento por meio de imagens, o qual utiliza-se de redes neurais artificiais para a detecção de pontos no rosto do ouvinte, por meio de uma transformação *Perspective-n-Point* (PnP) [16]. Esse algoritmo faz o mapeamento entre pontos no rosto do ouvinte na imagem 2D e uma malha de pontos 3D de uma cabeça genérica, assim permite-se inferir a posição do ouvinte (e de sua orelha) em relação a um sistema de coordenadas com origem na câmera, veja a Figura 6.

Esse tipo de sistema tem as grandes vantagens de ser não invasivo, ter baixo custo computacional e monetário, além de permitir precisão comparável a sistemas de monitoramento com *hardware* avançados [4]. Por outro lado, possui a desvantagem de requerer que o rosto do ouvinte seja visível a todos os momentos, o que pode limitar a amplitude de movimentos do ouvinte ou as condições de iluminação do ambiente. No entanto, essas limitações podem ser facilmente contornadas com o uso de múltiplas câmeras ao redor do

[10] Desde de que o sistema de coordenadas e formato de mensagem UDP (*User Datagram Protocol*) sejam equivalentes.

(a) Pontos de referência da face calculados com a malha.

(b) Pontos de referência do rosto selecionados para estimar a posição da cabeça.

Figura 6: Estratégia do rastreador da posição da cabeça — adaptado de Kartynnik *et al.* [17].

ouvinte[11] e/ou o uso de câmeras infravermelho, ou que operem em baixa luminosidade — câmeras com FPS (*Frames Per Second*) aumentado (60 FPS, por exemplo) podem também ajudar no desempenho.

4. Arquitetura e aplicação da máquina de auralização

A máquina de auralização no contexto deste trabalho caracteriza um sistema que faz a convolução em tempo real de um áudio com um par de Respostas ao Impulso Biauriculares (HRIR[12], *Head-Related Impulse Response*) em formato SOFA[13] [18], as quais são adaptadas com o objetivo de manter as fontes sonoras estáticas no espaço virtual com o auxílio de um sistema de rastreamento de cabeça.

Com o objetivo de comparar HRTFs em tempo real, esse sistema permite carregar múltiplos arquivos SOFA, conferindo ao ouvinte a capacidade de realizar a seleção do conjunto de dados (*dataset*) que deseja ouvir. Essa troca pode acontecer tanto por meio de comandos remotos com mensagens OSC[14] [19] quanto por atalhos no teclado (de um computador) de forma local. Permitindo que o ouvinte (ou uma terceira pessoa) possa acionar tal controle.

Na Figura 7 tem-se um diagrama geral da organização do sistema. O usuário carrega o áudio a ser auralizado e os perfis SOFA que deseja comparar. O gerenciador de posições seleciona qual a posição do *dataset* deve ser utilizada, baseado na posição da fonte especificada e na orientação do ouvinte. A transição suave entre filtros é gerenciada pela biblioteca de *convolução por partes* tal qual apresentada na Seção 2.4.

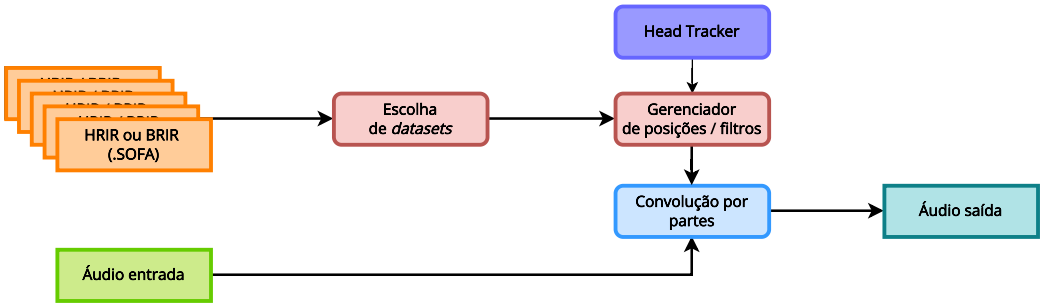


Figura 7: Diagrama da organização da máquina de auralização adaptativa (multiHRTFrenderer).

Esse tipo de arquitetura foi desenvolvido com o objetivo de gerar dados para análises de testes subjetivos na validação de HRTFs individualizadas. No entanto, pode ser facilmente adaptada à comparação de simulações de projetos de acústica de salas, ou de qualquer análise subjetiva que requeira a convolução com múltiplos

[11] Note que para utilizar múltiplas câmeras necessita-se o uso de mais de uma instância do *head tracker* e, consequentemente, é necessário implementar um sistema de transformação de coordenadas locais de cada câmera para um sistema de coordenas global.

[12] A HRIR é a versão temporal da HRTF, isto é, $\mathfrak{F}\{\text{HRIR}\} = \text{HRTF}$, sendo $\mathfrak{F}\{\cdot\}$ a Transformada de Fourier [3].

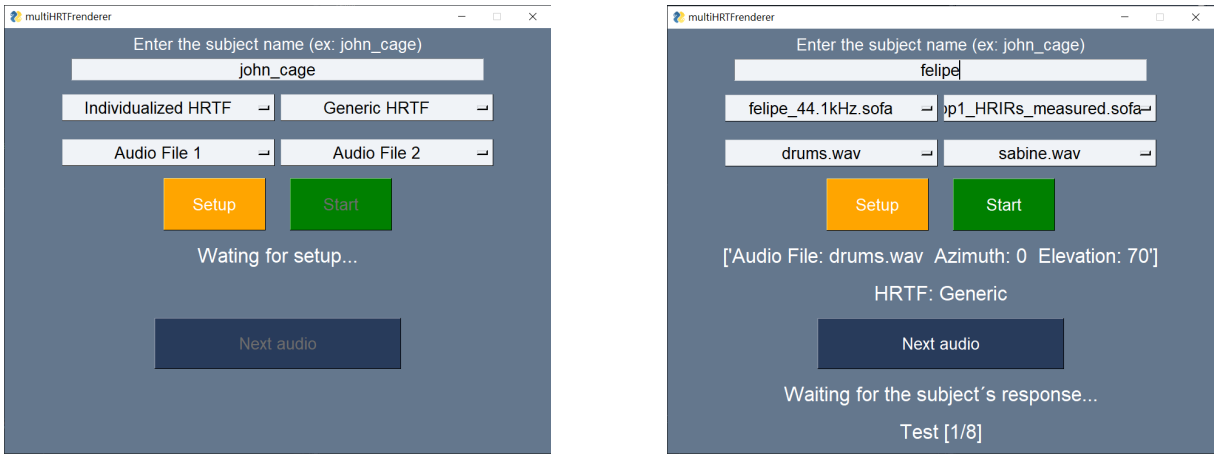
[13] Spatially Oriented Format for Acoustics (SOFA), veja demais informações em https://www.sofaconventions.org e também na norma AES69-2022: AES standard for file exchange – Spatial acoustic data file format [18].

[14] Open Sound Control (OSC) é um protocolo aberto, independente de transporte e baseado em mensagens, desenvolvido para a comunicação entre computadores, sintetizadores de som e outros dispositivos multimídia.

filtros de resposta ao impulso finita (FIR). Outro tipo de ensaio possível com essa *máquina* seria a troca dinâmica de conjuntos de HRTFs livremente para o usuário, ou seja, permitindo a alternância ilimitada entre eles para um mesmo ângulo e áudio. Nesse caso, a pergunta para o ouvinte poderia ser: "*Qual dos casos (A ou B) soa mais natural?*" — o voluntário poderia alternar até se sentir confiante para responder.

4.1 Interface gráfica (GUI)

Para a realização de testes de comparação entre HRTFs individualizadas[15] e genéricas, construiu-se também uma interface gráfica (GUI[16]), ilustrada na Figura 8. A GUI foi desenvolvida especificamente para um teste de comparação no qual um sujeito é exposto a sinais sonoros localizados em pontos específicos no espaço (pré-definidos pelo operador do teste no código-fonte) e sua tarefa é simplesmente "olhar para a fonte" (ou seja, *ficar de frente para ela*) e apertar um botão de ação. Ao longo do experimento, os sinais de áudio são convoluídos ora com a HRTF individualizada, ora com a HRTF genérica. Ademais, o rastreador de cabeça é responsável por verificar a direção (azimute e elevação) para a qual o sujeito está olhando, permitindo comparar a posição real da fonte com a posição percebida pelo ouvinte e, assim, averiguar os desempenhos em cada caso.



(a) Tela inicial de configuração da GUI.

(b) Tela da GUI durante um teste.

Figura 8: Interface gráfica (GUI) desenvolvida para a avaliação de desempenho de HRTFs personalizadas.

A utilização da GUI é simples e direta. Primeiro, o operador deve escrever o nome do sujeito e selecionar os arquivos SOFA com a HRTF individualizada (*Individualized HRTF*) e genérica (*Generic HRTF*), respectivamente. Na sequência, o operador seleciona o(s) arquivo(s) de áudio que serão posicionados no espaço ao longo do teste (os sinais são tocados individualmente, e nunca em conjunto). As posições são previamente selecionadas e precisam ser escritas em código, não havendo número limite de posicionamentos. Com as configurações prontas, o operador deve clicar em *Setup*, aguardar o carregamento do programa e clicar em *Start* para iniciar o teste.

Ao longo do experimento, cada sujeito ouvirá os áudios em sequência randomizada e deverá *olhar para a fonte sonora* posicionada virtualmente no espaço. Ao acreditar estar de frente para a fonte, basta clicar na tecla "seta direita" do teclado (ou de um controle para apresentação de slides, por exemplo). O programa irá capturar a posição da cabeça do ouvinte, o arquivo de áudio reproduzido, a HRTF utilizada e a posição real da fonte, salvando tudo em um arquivo do tipo `.npz` para análise posterior. Uma mensagem de validação aparece na tela, permitindo ao operador passar para o próximo áudio.

4.1.1 Extração de resultados e teste-piloto

Ao longo do teste proposto, o programa salva um arquivo `.npz` para cada posição capturada (ou seja, cada vez que o sujeito aperta o botão de ação). De modo a possibilitar a leitura desses dados em Matlab

[15] É comum a utilização do termo "HRTFs individualizadas", no entanto, o que está geralmente contido no arquivo SOFA é a Função de Transferência Direcional (*Directional Transfer Function*, DTF), que contém apenas as informações (das HRTFs) com dependência direcional [20].

[16] Do inglês, *Graphical User Interface*.

(para pós-processamento e análises), escreveu-se uma rotina em Python que converte os arquivos `.npz` para `.mat`. Ademais, como o teste está configurado para randomizar a sequência de localizações da fonte sonora a cada execução, fez-se necessário uma etapa de padronização dos dados obtidos para cada sujeito antes da realização de qualquer análise. Para esse fim, elaborou-se um código em Matlab responsável por ler os dados obtidos com cada sujeito (já convertidos em `.mat`), organizá-los em uma sequência padrão e salvá-los em uma matriz única contendo todas as informações capturadas durante o teste. A Figura 9 contém um diagrama que resume o procedimento, bem como explicita as informações obtidas em cada etapa.

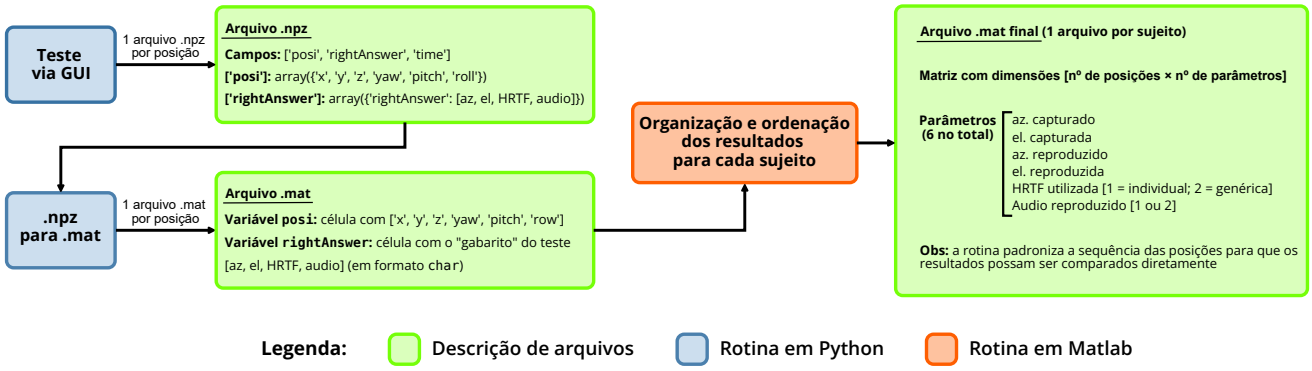
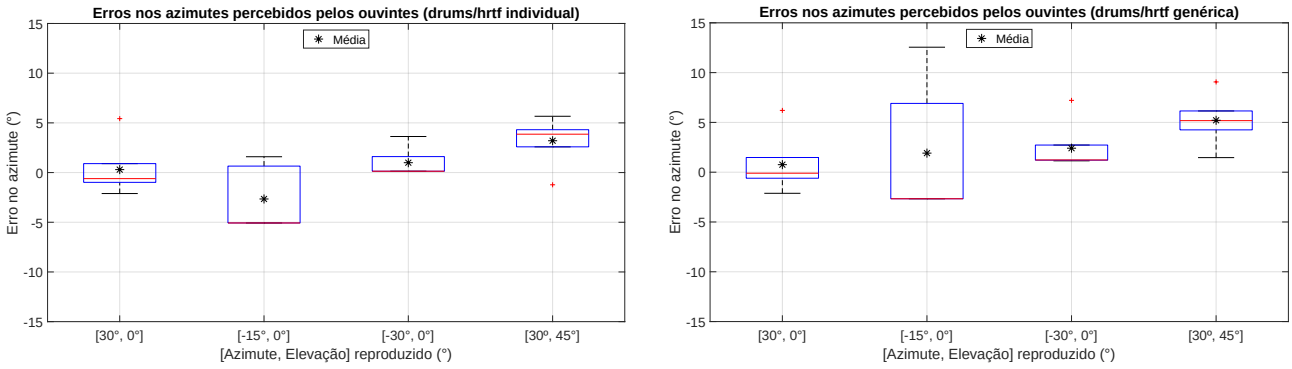


Figura 9: Diagrama explicativo do processo de extração e organização dos resultados.

Para averiguar o funcionamento do teste proposto, uma rodada piloto foi realizada com cinco sujeitos. Na ocasião, os sujeitos foram expostos a dois áudios (gravação de uma bateria acústica e um sinal de fala), considerando quatro posições de fonte e o uso de conjuntos de HRFTs (individualizada e genérica). Além disso, um sinal de calibração consistindo no sinal de bateria ao centro (azimute e elevação em 0°) foi reproduzido ao início de cada teste, para que os sujeitos pudessem compreender a proposta. Desse modo, um total de 17 sinais foram reproduzidos para cada sujeito.

Ao fim, para ilustração, avaliou-se a média geral dos erros em azimute. Os resultados para as HRTFs individualizada e genérica podem ser verificados nas Figuras 10 (a) e 10 (b), respectivamente. É importante mencionar que a rodada-piloto teve como objetivo averiguar o funcionamento do teste desenvolvido, bem como da interface gráfica. Desse modo, e dada a pequena população considerada, não é possível afirmar qual dos conjuntos de HRTFs foi superior. Todavia, notou-se que os erros considerando os bancos de HRTFs individualizadas foram ligeiramente menores e menos dispersos em comparação aos erros para o banco de HRTFs genérico. Por fim, espera-se, para trabalhos futuros, realizar testes oficiais com uma população maior.



(a) HRTFs individualizadas (DTFs). **(b)** HRTFs genéricas.

Figura 10: Erros nos azimutes percebidos por cinco sujeitos em teste-piloto, considerando o uso de HRTFs individualizadas e genéricas — os asteriscos em preto representam o erro médio; as linhas vermelhas a mediana; as arestas inferior e superior dos retângulos azuis representam os 1º e 3º quartis, respectivamente; as linhas pretas descrevem os limites inferior e superior; e, finalmente, as cruzes vermelhas representam valores discrepantes (*outliers*).

4.2 Distribuição do *software*

A máquina de auralização (multiHRTFrenderer) apresentada neste artigo foi implementada em Python. Ela é distribuída sob licença MIT [21] e pode ser encontrada em github.com/davircarvalho/multiHRTFrenderer. O módulo de convolução por partes pode ser também utilizado como uma biblioteca individual, consulte pypi.org/project/FIRconv. Para este artigo, foi criado também um *repositório congelado*[17] com as versões desses módulos atualizadas na data de publicação, ele pode ser acessado em github.com/eac-ufsm/sobrac2023-convolution. Por fim a biblioteca de *head tracker* também é distribuída livremente e pode ser utilizada como uma biblioteca individual. Ela pode ser encontrada em pypi.org/project/EACheadtracker.

5. Considerações finais

Nesse artigo, foram apresentadas algumas ferramentas para a geração de cenários auditivos virtuais, construídas para uso em testes nos quais busca-se comparar subjetivamente diferentes bancos de HRIRs (ou BRIRs). Com essa *caixa de ferramentas*, é possível realizar a *auralização dinâmica* (em tempo real) com o uso de um rastreador de cabeça sem contato, com a opção de selecionar entre três algoritmos de convolução por partes: *Overlap-Add* (OLA, sobreposição e adição), *Overlap-Save* (OLS, sobreposição e armazenamento) e *Uniformly Partitioned Overlap-Save* (UPOLS, partição uniforme em sobreposição e armazenamento). A inclusão de diferentes algorítimos para convolução oferecem ao usuário flexibilidade no processo de *auralização baseada em objetos* (*object-based auralization*), assim como permite a convolução em tempo real mesmo com BRIRs longas (com o uso de UPOLS).

A *máquina de auralização* desenvolvida permite a troca entre perfis de HRTFs em tempo real por meio de atalhos no teclado ou, ainda, remotamente por meio de comandos OSC (sem falhas audíveis). Essa permutação possibilita testes subjetivos que requeiram uma alternância instantânea entre a auralização de diferentes funções de transferência. Ademais, uma interface gráfica foi construída para auxílio na realização de testes de comparação entre HRTFs individualizadas e genéricas. O teste proposto foi avaliado em uma rodada-piloto com cinco sujeitos, com a qual foi possível verificar o funcionamento do algoritmo e pontos de melhoria.

No geral, as ferramentas apresentadas neste artigo mostram-se promissoras para a criação de cenários auditivos virtuais e a realização de experimentos subjetivos relacionados à sensação espacial do som. A auralização biauricular em tempo real, aliada à capacidade de trocar os perfis de HRTFs de forma dinâmica, oferece um ambiente flexível para experimentos e escuta, permitindo investigações detalhadas e personalizadas. A associação entre os algoritmos de convolução por partes, o rastreador de cabeça, e uma interface guia para o usuário permitiram a criação de um sistema para a avaliação de desempenho de localização espacial no uso de HRTFs, evidenciando uma das formas de utilizar todas essas ferramentas de forma prática, no contexto da pesquisa de áudio biauricular.

6. Agradecimentos

Os autores gostariam de agradecer o apoio e infraestrutura da Engenharia Acústica (EAC) [22] e do Programa de Pós-Graduação Arquitetura, Urbanismo e Paisagismo (PPGAUP) da Universidade Federal de Santa Maria (UFSM), bem como os seus editais internos de bolsa FIPE-CT e FIEX-CT que apoiaram parte do desenvolvimento abordado neste trabalho.

Referências

1. XIE, Bo-sun. *Head-Related Transfer Function and Virtual Auditory Display*. 2. ed. Plantation, FL, EUA: J. Ross Publishing, 2013. ISBN 978-1604270709.

[17] Detalhes de instalação (passo a passo) podem ser consultados no próprio GitHub. Além disso, com objetivo de apoiar mestres e aprendizes, o repositório https://github.com/eac-ufsm/sobrac2023-convolution contém uma pasta chamada "Teaching" com códigos simples de *Overlap-Add* e *Overlap-Save* (em Matlab e Python) para que usuários possam trabalhar com sinais simples e compreender os conceitos desenvolvidos neste artigo. Outrossim, é disponibilizada uma videoaula (neste link https://youtu.be/UavaCUBvRP8) sobre o tema de *convolução por partes*, que é comumente ministrada pelo professor William D'Andrea Fonseca, pertencente ao curso de Engenharia Acústica (EAC) [22], da Universidade Federal de Santa Maria (UFSM), para a disciplina de Processamento Digital de Sinais 2 [10].

2. GÁLVEZ, Marcos F. Simón; MENZIES, Dylan; FAZI, Filippo Maria. Dynamic audio reproduction with linear loudspeaker arrays. *Journal of the Audio Engineering Society*, Audio Engineering Society, v. 67, n. 4, p. 190–200, abr. 2019. ISSN 0004-7554, 1549-4950. doi: 10.17743/jaes.2019.0007.

3. OPPENHEIM, Alan; WILLSKY, A. Simon. *Sinais e Sistemas*. 2. ed. São Paulo: Pearson, 2010. ISBN 978-8576055044.

4. CARVALHO, Davi Rocha; FONSECA, William D'Andrea; HOLLEBON, Jacob; MAREZE, Paulo Henrique; FAZI, Filippo Maria. Head tracker using webcam for auralization. In: *50th International Congress and Exposition on Noise Control Engineering – Internoise 2021*. Washington, DC, USA: [s.n.], 2021. p. 1–10. doi: 10.3397/IN-2021-2956. Disponível em: https://bit.ly/headtracker-int2021.

5. HARRIS, Fredric J. *Multirate Signal Processing for Communication Systems*. USA: Prentice Hall PTR, 2004. ISBN 978-0131465114.

6. KUO, Sen M.; LEE, Bob H.; TIAN, Wenshun. *Real-time digital signal processing: implementations and applications*. 3. ed. United Kingdom: John Willey & Sons, 2013. ISBN 978-1118414323.

7. HURCHALLA, Jeffrey. Low Latency Convolution in One Dimension Via Two Dimensional Convolutions: An Intuitive Approach. In: *125th Convention of the Audio Engineering Society*. San Francisco, CA, EUA: Audio Engineering Society (AES), 2008. p. 1–11. Paper n. 7634. Disponível em: http://www.aes.org/e-lib/browse.cfm?elib=14785.

8. BURRUS, Charles Sidney; PARKS, Thomas W. *DFT/FFT and convolution algorithms: theory and implementation*. New York, EUA: John Wiley & Sons, 1984. ISBN 978-0824714994.

9. WEFERS, Frank. *Partitioned convolution algorithms for real-time auralization*. Tese (Doutorado) — Instituto de Acústica Técnica, Universidade RWTH Aachen, Aachen, Alemanha, setembro 2014. ISBN 978-3832539436. Disponível em: https://publications.rwth-aachen.de/record/466561.

10. FONSECA, William D'Andrea. *Notas de aula da disciplina EAC 1007 - Processamento Digital de Sinais II (semestre 2023-1)*. 2023. Engenharia Acústica (EAC), Universidade Federal de Santa Maria (UFSM), Brasil.

11. TORGER, Anders; FARINA, Angelo. Real-time partitioned convolution for Ambiophonics surround sound. In: *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No.01TH8575)*. New Paltz, New York, EUA: IEEE. p. 195–198. doi: 10.1109/aspaa.2001.969576.

12. KULP, Barry D. Digital Equalization Using Fourier Transform Techniques. In: *85th Audio Engineering Society Convention*. Los Angeles, EUA: Audio Engineering Society (AES), 1988. p. 1–32. Disponível em: http://www.aes.org/e-lib/browse.cfm?elib=4750.

13. WEFERS, Frank; VORLÄNDER, Michael. Optimal filter partitions for real-time FIR filtering using uniformly-partitioned FFT-based convolution in the frequency-domain. In: *14th International Conference on Digital Audio Effects (DAFx-11)*. Paris, França: [s.n.], 2011. p. 19–23. Disponível em: https://publications.rwth-aachen.de/record/117660.

14. WEFERS, Frank; VORLÄNDER, Michael. Efficient time-varying FIR filtering using crossfading implemented in the DFT domain. In: *Forum Acusticum*. Cracóvia, Polônia: EAA, 2014. p. 1–6. Disponível em: https://bit.ly/FA2014-Wefers.

15. IEEE Standards Association. *IEEE Standard for Floating-Point Arithmetic*. 2019. Standard link.

16. LUGARESI, Camillo; TANG, Jiuqiang; NASH, Hadon; MCCLANAHAN, Chris; UBOWEJA, Esha; HAYS, Michael; ZHANG, Fan; CHANG, Chuo-Ling; YONG, Ming Guang; LEE, Juhyun; CHANG, Wan-Teh; HUA, Wei; GEORG, Manfred; GRUNDMANN, Matthias. MediaPipe: A Framework for Building Perception Pipelines. *CoRR*, abs/1906.08172, 2019. Disponível em: http://arxiv.org/abs/1906.08172 ou https://github.com/google/mediapipe.

17. KARTYNNIK, Yury; ABLAVATSKI, Artsiom; GRISHCHENKO, Ivan; GRUNDMANN, Matthias. Real-time Facial Surface Geometry from Monocular Video on Mobile GPUs. *CoRR*, abs/1907.06724, p. 1–4, 2019. Disponível em: http://arxiv.org/abs/1907.06724.

18. Audio Engineering Society (AES). *(Standard) AES69-2022: AES standard for file exchange – Spatial acoustic data file format*. 2022. Standard link.

19. FREED, Adrian; SCHMEDER, Andrew. Features and Future of Open Sound Control version 1.1 for NIME. Zenodo, p. 116–120, 2009. doi: 10.5281/ZENODO.1177516.

20. FONSECA, William D'Andrea; CARVALHO, Davi Rocha; MELLO, Felipe Ramos de. Binaural Parameters: código computacional para ensino da tecnologia biauricular. In: *XXX Encontro da Sociedade Brasileira de Acústica (Sobrac 2023)*. Natal, RN, Brasil: Sociedade Brasileira de Acústica, 2023. p. 1–16. Disponível em: https://bit.ly/sobrac2023-binaural.

21. Massachusetts Institute of Technology. *The MIT License (MIT)*. 2023. Acesso em 24/06/2023. Disponível em: https://opensource.org/license/mit.

22. FONSECA, William D'Andrea; BRANDÃO, Eric; MAREZE, Paulo Henrique; MELO, Viviane Suzey; TENENBAUM, Roberto A; SANTOS, Christian dos; PAIXÃO, Dinara Xavier da. Acoustical engineering: a complete academic undergraduate program in Brazil. *The Journal of the Acoustical Society of America*, v. 152, n. 2, p. 1180–1191, 2022. ISSN 0001-4966, 1520-8524. doi: 10.1121/10.0013570.