UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
ENGENHARIA DE COMPUTAÇÃO

GIOVANNI LUIZ ZANETTI

SIMULAÇÃO DA RESPOSTA ACÚSTICA AO IMPULSO EM AMBIENTES TRIDIMENSIONAIS UTILIZANDO COMPUTAÇÃO PARALELA

CURITIBA
2021

GIOVANNI LUIZ ZANETTI

SIMULAÇÃO DA RESPOSTA ACÚSTICA AO IMPULSO EM AMBIENTES TRIDIMENSIONAIS UTILIZANDO COMPUTAÇÃO PARALELA

Acoustic impulse response simulation in three-dimensional environments using parallel computing

Trabalho de Conclusão de Curso de Graduação do curso de Engenharia de Computação do Departamento Acadêmico de Informática e Departamento Acadêmico de Eletrônica da Universidade Tecnológica Federal do Paraná (UTFPR), como requisito parcial para a obtenção do título de Engenheiro de Computação.

Orientador: Prof. Dr. Marcelo de Oliveira Rosa
DAELT - Departamento Acadêmico de Eletrotécnica - UTFPR

CURITIBA
2021

GIOVANNI LUIZ ZANETTI

SIMULAÇÃO DA RESPOSTA ACÚSTICA AO IMPULSO EM AMBIENTES TRIDIMENSIONAIS UTILIZANDO COMPUTAÇÃO PARALELA

Trabalho de Conclusão de Curso de Graduação do curso de Engenharia de Computação do Departamento Acadêmico de Informática e Departamento Acadêmico de Eletrônica da Universidade Tecnológica Federal do Paraná (UTFPR), como requisito parcial para a obtenção do título de Engenheiro de Computação.

Data de aprovação: 6 de outubro de 2021

Marcio Henrique de Avelar Gomes
Doutorado
Universidade Tecnológica Federal do Paraná

Glauber Gomes de Oliveira Brante
Doutorado
Universidade Tecnológica Federal do Paraná

Marcelo de Oliveira Rosa
Doutorado
Universidade Tecnológica Federal do Paraná

CURITIBA
2021

AGRADECIMENTOS

Gostaria de agradecer ao meu orientador, professor Dr. Marcelo de Oliveira Rosa, pelo apoio e paciência, pelo acompanhamento, sugestões e dedicação durante toda a formulação e execução desse trabalho.

Ao professor Marcio Henrique de Avelar Gomes por fornecer as respostas analíticas que serviram como base para a validação de resultados obtidos nesse trabalho.

A Deus, aos meus pais, meu irmão, meus amigos, todos os professores envolvidos na minha formação, a todos que sempre estiveram de alguma forma envolvidos no meu crescimento dentro da universidade durante esses anos, e que sem eles não teria chegado nessa etapa da minha formação.

À Bruna pela colaboração nas revisões da escrita desse trabalho, e por toda a sua amizade, companheirismo, apoio e convivência.

RESUMO

ZANETTI, G. L. Simulação da resposta acústica ao impulso em ambientes tridimensionais utilizando computação paralela. 2021. 50 f. – Engenharia de Computação, Universidade Tecnológica Federal do Paraná. Curitiba, 2021.

A caracterização acústica de ambientes é um fator importante para a projeção e construção de ambientes acusticamente melhores, como salas de concerto, salas de aula, teatros e até mesmo em casas. Os parâmetros acústicos obtidos podem ser valores como o tempo de reverberação, ou até mesmo a resposta ao impulso da sala (RIR - *Room Impulse Response*), que descreve como a energia sonora se atenua nesse ambiente, em função do tempo. Esses parâmetros obtidos estão relacionados com fatores como a inteligibilidade da fala no ambiente. O objetivo desse trabalho é fornecer um programa capaz de realizar as computações de cálculo dessa resposta ao impulso, considerando absorção de energia dependente da frequência sonora, de forma paralelizada e em linguagem de programação de alto nível. Para isso, foi desenvolvido um programa na linguagem *Python* com o uso de bibliotecas auxiliares como *NumPy* e *SciPy* para realização de operações matemáticas gerais, e do *framework Apache Spark* para realizar a paralelização do algoritmo. Foram utilizadas propriedades da transformada de Fourier para cálculos de absorção de energia e resposta final. A geometria e os parâmetros do ambiente, bem como as posições dos objetos, são indicadas ao algoritmo através de arquivos de entrada, tornando simples a customização de um ambiente. O algoritmo desenvolvido possibilita a obtenção da resposta ao impulso de um ambiente tridimensional qualquer, de forma paralelizada, por consequência mais rápida do que a de métodos não paralelizados, além de demonstrar que essa resposta é similar à resposta analítica de base.
Palavras-chave: Acústica. Resposta ao impulso. RIR. Frequência. Computação paralela

ABSTRACT

ZANETTI, G. L. Acoustic impulse response simulation in three-dimensional environments using parallel computing. 2021. 50 f. – Engenharia de Computação, Universidade Tecnológica Federal do Paraná. Curitiba, 2021.

The acoustic caracterization of environments is a important factor for project development and construction of acoustically better environments, as concerts, classrooms, theaters and even at houses. The obtained acoustic parameters can be values as reverberation time or even the room impulse response (RIR), which describes how the energy decays at the environment, as a function of time. These obtained parameters are related with elements like the speech intelligibility at the environment. The goal of this work is to provide a program able to perform computations to calculate this impulse response, considering that the energy absorption is dependent on sound frequency, using parallel computing and a high level programming language. For this, it was developed a program with *Python* language, using libraries like *NumPy* and *SciPy* for performing general mathematical operations, and the framework *Apache Spark* for performing the algorithm paralelization. It was used Fourier transform properties for energy absorption and final response calculation. The environment geometry and parameters, as well as objects positions, are given to the algorithm through input files, making simple the environment customization. The developed algorithm enables the obtainment of the impulse response of any three-dimensional environment, in parallel, consequently faster than non-parallel methods, besides demonstrating that this response is similar to the analytical base response.

Keywords: Acoustics. Impulse response. RIR. Frequency. Parallel computing

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 CONTEXTO

O principal objetivo do modelamento acústico de um ambiente é prover suas características de tempo e energia e atributos como tempo de reverberação, curva de decaimento, etc, além da sua resposta ao impulso, que é definida como a resposta acústica dessa sala quando um impulso sonoro é emitido, e descreve como a energia sonora irá atenuar nesse ambiente, com base no tempo (SAVIOJA; SVENSSON, 2015). Isso também possibilita a auralização da sala (KLEINER; DALENBÄCK; SVENSSON, 1993), que simula dinamicamente, a partir de um som de entrada, o som resultante que será ouvido na sala em questão. A medição ou cálculo da resposta ao impulso de um ambiente é um estudo importante na área de processamento de sinais e áudio, pois reflete as características sonoras de um ambiente, as quais descrevem a forma como o som será propagado e absorvido pelo ambiente e como será percebido por um ouvinte em seu interior. Estudos feitos por Sabine (SABINE, 1922) para investigar a propagação do som podem ser considerados como um primeiro passo no estudo do método de traçado de raios e na acústica geométrica. O conceito de livre caminho médio também está relacionado com a acústica geométrica, tendo relação com a fórmula de tempo de reverberação descrita previamente por Sabine em seu trabalho. A validade dos algoritmos baseados em acústica geométrica para pequenos comprimentos de onda foi comprovada em 1929, e foram publicados nos anos 1970 estudos feitos por Schroeder demonstrando a auralização de uma sala de concerto (SCHROEDER, 1973). Um salto nas pesquisas e estudos sobre modelamento acústico de salas ocorreu no início de 1990 e, desde então, novos trabalhos estudaram os métodos até então existentes e introduziram novas melhorias. Esses métodos vêm ganhando mais precisão e performance até hoje, com algoritmos mais robustos que podem ser executados graças a computadores com maior capacidade de processamento.

1.2 OBJETIVO

1.2.1 Objetivo geral

Realizar o estudo e implementação de um algoritmo para obtenção da resposta acústica ao impulso (RIR - *Room Impulse Response*) de um ambiente tridimensional qualquer estipulado, baseado no método geométrico de traçado de raios. Além disso, implementar a paralelização do processamento do algoritmo, visando melhorar a sua performance computacional, bem como o processamento com base no domínio da frequência.

1.2.2 Objetivos específicos

- Estudar os métodos já existentes de caracterização de resposta acústica;
- Implementar na linguagem Python o algoritmo baseado no método de traçado de raios;
- Obter uma resposta ao impulso do ambiente similar à resposta analítica;
- Possibilitar o cálculo da resposta ao impulso para ambientes tridimensionais quaisquer (no formato .obj), inclusive ambientes não-convexos;
- Tornar o algoritmo capaz de realizar o cálculo para vários emissores/receptores em uma única execução;

- Adaptar o algoritmo com o uso do *framework* Apache Spark para fazer a paralelização da sua execução, de forma que cada raio é processado independentemente e paralelamente até a sua extinção, assim aumentando a performance em termos de redução do tempo de processamento total do algoritmo;
- Implementar na resposta ao impulso efeitos causados no domínio da frequência, de acordo com as propriedades acústicas espectrais do ambiente e seus obstáculos, onde existe uma absorção maior para determinadas bandas de frequência e menor para outras;
- Possibilitar a auralização da sala, ou seja, a obtenção do som percebido pelo receptor em determinada sala, ao ser inserido um som arbitrário e convoluído com a resposta ao impulso obtida pelo algoritmo.

Nesse trabalho, por se tratar da implementação de um método que se baseia na propagação de raios, não serão consideradas características e efeitos ondulatórios da natureza do som, como a interferência e a difração. Para a propagação do som serão apenas consideradas as suas características geométricas.

1.3 JUSTIFICATIVA

A motivação geral do cálculo da resposta acústica de uma sala é possibilitar a construção e projeção de ambientes acusticamante melhores, principalmente quando se trata de salas de concerto, teatros e estúdios, salas de aula, estações de trem e metrô, e até mesmo em casas, pois fornece importantes informações sobre a acústica do ambiente, da qualidade e características da inteligibilidade da fala nesse ambiente (EATON et al., 2016). Nesse trabalho será elaborado um programa na linguagem *Python*, uma linguagem de código aberto, de alto nível e fácil compreensão, assim possibilitará que o programa desenvolvido seja flexível e de fácil modificação. Isso torna possível a implementação de futuras melhorias em cima do algoritmo, de forma a se obter resultados ainda mais próximos da realidade, além de novos trabalhos de pesquisa na área de acústica que podem se basear nos estudos feitos. A paralelização do algoritmo implementada nesse trabalho, com o uso do *framework Apache Spark*, possibilita uma execução mais rápida, o que é de grande interesse para execução em larga escala e para possíveis usuários, além de facilitar a execução de testes durante o seu desenvolvimento. O estudo apresentado sobre a adaptação do algoritmo para processamentos no domínio da frequência pode ser um diferencial importante para resultados mais próximos da realidade.

1.4 ESTRUTURA DO TRABALHO

Este trabalho está organizado em cinco capítulos, de forma a abordar os temas estudados e desenvolvidos, juntamente com os resultados obtidos, pontuados da seguinte forma:

- Introdução: Capítulo introdutório, que apresenta o contexto do problema, os objetivos do trabalho e sua justificativa.
- Revisão de literatura: Apresenta uma visão geral dos métodos já existentes, as vantagens e problemas de cada um, de acordo com teóricos e artigos.
- Metodologia: Discorre sobre o desenvolvimento de todo o trabalho, desde a introdução sobre assuntos teóricos importantes e revisão matemática, bem como o desenvolvimento e características de implementação de cada etapa do trabalho.
- Resultados: Apresenta os resultados obtidos pelo programa desenvolvido, como a similaridade com o resultado analítico, performance e efeitos da frequência.

- Conclusão: Considerações finais sobre o trabalho, observações acerca dos resultados e melhorias que podem ser implementadas em possíveis trabalhos futuros.

2 REVISÃO DE LITERATURA

2.1 VISÃO GERAL

Uma forma de obter os parâmetros acústicos característicos de uma sala ou ambiente, como tempo de reverberação ou a sua resposta ao impulso, é emitir um sinal e avaliar o sinal capturado com um microfone posicionado em um local da sala. Diferentes abordagens foram utilizadas para a medição de resposta ao impulso, sendo uma das primeiras baseada na emissão de um impulso como sinal de entrada. Desconsiderando o ruído, o sinal captado pode fornecer diretamente a resposta ao impulso da sala se esta for considerada como um sistema linear e invariante no tempo, sendo possível também realizar uma média de várias respostas ao impulso medidas para minimizar o efeito do ruído. Um dos problemas dessa abordagem é a impossibilidade de se emitir um impulso ideal, o qual na prática será uma aproximação (CARINI; CECCHI; ROMOLI, 2016). Outra técnica de medição é realizar a emissão de sinais com diferentes frequências, como uma varredura de senos, medindo a atenuação de cada frequência (NOVAK; RUND; HONZÍK, 2016).

Além dos métodos de medição, existem também modelos que possibilitam a simulação da propagação do som dentro da sala, os quais obtêm uma resposta acústica estimada para aquele ambiente. As duas formas mais comuns são as que utilizam modelos numéricos baseados na natureza ondulatória do som, e as que baseiam-se em aproximações geométricas. Os modelos resultam em uma resposta ao impulso percebida por um receptor naquela sala, o que possibilita também que sejam calculadas características a partir dessa resposta ao impulso, como a curva de decaimento, o tempo de reverberação, etc.

O modelamento baseado no comportamento ondulatório é capaz de prover melhores resultados, mas é computacionalmente bastante custoso e algumas vezes limitado a determinados formatos de sala. Já na acústica geométrica, as propriedades ondulatórias do som são ignoradas, e é assumido que o som se propaga em forma de raios. Essa aproximação é válida principalmente em frequências altas. Para frequências baixas, o comprimento de onda é mais próximo do tamanho dos objetos, espaços e dimensões da sala, nesse caso as características ondulatórias como interferência e difração são mais visíveis. Essa frequência de transição é conhecida como frequência de Schroeder (SCHROEDER, 1973), e leva em consideração o volume da sala em questão.

Segundo (SAVIOJA; SVENSSON, 2015), as reflexões que ocorrem dentro da sala podem ser consideradas especulares ou difusas. As reflexões especulares ocorrem em apenas uma direção, como acontece com a luz em um espelho. Já a reflexão difusa ocorre em várias direções. Uma implementação que contém os dois tipos de reflexão pode prover bons resultados de simulação. Os princípios de modelamento geométrico são bem gerais, principalmente o de traçado de raios, que é muito utilizado na computação gráfica, no qual o problema é a propagação de luz, sendo um problema análogo ao do som. As computações da propagação de raios são bem custosas pela necessidade de se encontrar interseções entre raios e superfícies. Uma forma de otimizar esse processo é fazer o uso de estruturas espaciais como *bounding volumes* e *kd-trees*. Além disso, as superfícies não são totalmente planas: elas possuem várias irregularidades, as quais podem ser consideradas pelos modelos, apesar de não ser uma implementação simples. Irregularidades que são muito menores que o comprimento de onda podem ser desprezadas, ou se forem muito grandes comparadas ao comprimento de onda, já estarão "embutidas" na geometria da superfícies. Irregularidades com tamanho da mesma ordem do comprimento de onda podem ter um grande impacto

na característica da onda refletida, afetando o resultado final.

O trabalho de (SAVIOJA; SVENSSON, 2015) também faz um apanhado geral de vários métodos de caracterização acústica de uma sala, dentre eles os baseados em geometria acústica. A seguir são descritos brevemente alguns desses métodos.

2.2 MÉTODOS DE CARACTERIZAÇÃO ACÚSTICA

2.2.1 Método das fontes virtuais

O método das fontes virtuais é um método cuja principal característica é "refletir" a fonte sonora contra as superfícies existentes na sala, resultando em várias fontes virtuais. As fontes virtuais geradas são refletidas novamente, ou seja, é um processo recursivo, que é repetido até que uma determinada ordem de reflexão seja alcançada, ou que a resposta ao impulso chegue a um tamanho desejado. Cada uma das fontes resultantes representa uma reflexão, tal que a distância entre uma fonte virtual até um receptor corresponde à distância total percorrida pelo som por aquele caminho dentro da sala. A resposta ao impulso final é construída a partir da soma da contribuição de cada fonte.

No caso de geometrias poligonais arbitrárias, algumas verificações adicionais devem ser aplicadas ao algoritmo: as fontes devem sofrer reflexão apenas no lado "frontal" das superfícies. Ou seja, não refletem do lado oculto de uma superfície, portanto essas reflexões devem ser desconsideradas. De acordo com (MENEZES, 2018), para uma fonte ser válida, ela deve respeitar os critérios de validade, proximidade e visibilidade. Em linhas gerais, a fonte só é válida se não foi gerada por uma reflexão na mesma superfície da qual ela se originou, e a fonte deve necessariamente estar próxima o suficiente do receptor para ser considerada. Além disso a fonte deve ser visível pelo receptor, sem obstáculos entre eles.

2.2.2 Método de traçado de raios

No método de traçado de raios, o comportamento do som é modelado a partir de vários raios, que representam a propagação da onda sonora. Não só a reflexão especular é considerada, mas qualquer tipo de reflexão pode ser implementada, tais como reflexões difusas, ou um método estocástico por exemplo, em que os raios podem sofrer um pequeno desvio aleatório ao serem refletidos. Cada raio emitido pela fonte é propagado até que este colida com uma superfície: assim o raio é refletido em uma nova direção. Cada raio carrega informação sobre tempo e energia, e sempre que ele colide com uma superfície, perde energia de acordo com propriedades da parede. A interseção entre o raio e os receptores também é calculada, de forma que, quando acontece esse tipo de colisão, é calculada a contribuição desse raio na resposta ao impulso percebida por esse receptor.

Toda a operação pode ser feita separadamente para cada banda de frequência, cada uma contendo sua energia. Ao mesmo tempo, a absorção pelas paredes também é específica para cada banda. O raio é extinto quando a sua energia total cai abaixo de um certo valor limite.

A convergência e a precisão do método aumenta quanto maior o número de raios emitidos pelo emissor, tendo em contrapartida um custo computacional maior. O número de raios para se conseguir uma resposta satisfatória depende da geometria da sala e das características dos materiais, sendo difícil estipular um número de raios adequado antes de executar a simulação.

2.2.3 Método de traçado de feixes

O primeiro trabalho que introduz a ideia de raios volumétricos (feixes) foi publicado no início da década de 1970 (HAVILAND; THANEDAR, 1973). O método é similar ao de traçados de raios, entretanto é emitido um objeto volumétrico ao invés de um raio infinitamente estreito. Dessa forma esse feixe pode ser facilmente detectado por um receptor cuja geometria é apenas um ponto no espaço, diferente do que acontece no método de raios, em que são necessários receptores volumétricos para que seja possível detectá-los.

2.2.4 Difração nos métodos geométricos

Ainda segundo (SAVIOJA; SVENSSON, 2015), a inclusão do modelamento de difração nos métodos de acústica geométrica tem papel importante para a obtenção de resultados mais precisos em casos específicos. A propagação do som em ambiente abertos, barreiras de som e geometrias de cidades podem certamente tirar vantagem desses aperfeiçoamentos que consideram a difração. Em salas, o maior proveito desse método é quando existem, por exemplo, presença de pilares e aberturas entre subvolumes.

Uma forma de combinar o método de traçado de raios com a difração, é utilizar "portais" transparentes nas bordas mais importantes, detectar a distância entre a borda e o ponto de interseção entre raio e portal, e alterar o caminho do raio com base nessa distância.

Para o método de fontes virtuais, a inclusão do modelamento de difração é geralmente baseado na detecção de bordas difratoras. Quando essas bordas são detectadas, elas podem ser consideradas como uma nova fonte virtual.

2.2.5 Métodos baseados em superfície

É citado ainda em (SAVIOJA; SVENSSON, 2015) sobre os métodos baseados em superfície, que baseiam-se na ideia de que as superfícies se comportam como meios intermediários de armazenamento de energia. Primeiramente, a energia sonora é propagada pela fonte até as superfícies, e após isso, propagada entre as superfícies. No final, a energia é propagada até o receptor. Um grande benefício desse método é que boa parte do processamento pode ser feito independente da posição do receptor, sendo muito útil em aplicações dinâmicas e interativas, pois apenas o último passo, o qual envolve a propagação da energia até o receptor, precisa ser recalculado.

Um dos métodos que aplicam essa ideia é o método da radiosidade, que foi primariamente utilizado em aplicações de termodinâmica nos anos 1950, apesar de que as principais equações desse método foram mostradas em um trabalho sobre ótica em (YAMAUTI, 1926). Um primeiro estudo feito em (MOORE, 1984) discute os princípios da radiosidade para a determinação da resposta ao impulso acústica.

3 METODOLOGIA

3.1 PROCEDIMENTO METODOLÓGICO

O desenvolvimento desse trabalho foi realizado em cinco maiores etapas. A primeira foi o estudo dos métodos de caracterização acústica já existentes e desenvolvidos no passado. A segunda foi o início do desenvolvimento da base do algoritmo em Python, como o cálculo da interseção entre reta e triângulo, bem como o desenvolvimento de todo o algoritmo de traçado de raios com as reflexões/refrações, e cálculo da resposta ao impulso. A terceira etapa foi referente a adaptação do algoritmo para o domínio da frequência. Na quarta etapa o algoritmo foi adaptado para o processamento com o *framework* Spark, possibilitando a sua paralelização. A quinta etapa foi referente à análise dos resultados.

3.2 REVISÃO MATEMÁTICA

Nessa seção serão abordados alguns tópicos matemáticos importantes para o desenvolvimento do restante do trabalho, onde serão explicados e detalhados conceitos que são bases fundamentais do trabalho em geral, compreendendo parte da geometria analítica abordada, além do estudo sobre a implementação em frequência, que irão facilitar a compreensão das seções seguintes.

3.2.1 Interseção entre reta e triângulo

As seguintes equações foram deduzidas como parte do desenvolvimento desse trabalho para cálculo da interseção entre uma reta e um triângulo.

Considerando uma reta em três dimensões definida vetorialmente pela equação:

$$\vec{P_r} = \vec{Po_r} + w\vec{W} \tag{1}$$

na qual $\vec{Po_r}$ é o ponto inicial, $\vec{W}$ é um vetor diretor que aponta na direção da reta e w é um escalar que ao variá-lo, percorre-se todos os pontos pertencentes à reta descritos por $\vec{P_r}$.

Da mesma forma, considerando um plano infinito em três dimensões descrito por:

$$\vec{P_t} = \vec{Po_t} + u\vec{U} + v\vec{V} \tag{2}$$

na qual $\vec{Po_t}$ é um ponto inicial, $\vec{U}$ e $\vec{V}$ são vetores diretores que não precisam necessariamente ser perpendiculares. Variando o escalares u e v, percorre-se todos os pontos pertencentes ao plano descritos por $\vec{P_t}$.

O ponto de interseção entre a reta e o plano são obtidos quando $\vec{Pr} = \vec{Pt}$, ou seja:

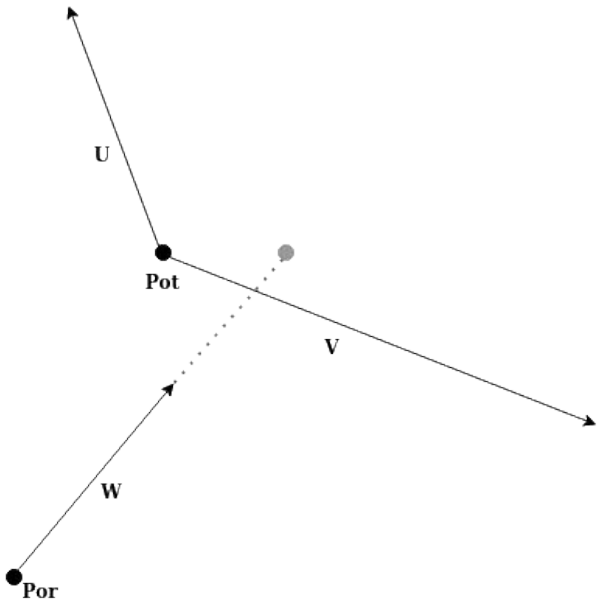
$$\vec{Po_r} + w\vec{W} = \vec{Po_t} + u\vec{U} + v\vec{V} \tag{3}$$

ou considerando a condição em três dimensões:

$$\begin{aligned} Po_{rx}\hat{i} + Po_{ry}\hat{j} + Po_{rz}\hat{k} + wW_x\hat{i} + wW_y\hat{j} + wW_z\hat{k} = \\ Po_{tx}\hat{i} + Po_{ty}\hat{j} + Po_{tz}\hat{k} + uU_x\hat{i} + uU_y\hat{j} + uU_z\hat{k} + vV_x\hat{i} + vV_y\hat{j} + vV_z\hat{k} \end{aligned} \tag{4}$$

Quer-se encontrar os valores de u, v e w que satisfaçam a equação acima

Figura 1 – Representação da reta e do plano.



Fonte: Autoria própria

Agrupando as variáveis por dimensão do vetor

$$\begin{cases} (Po_{rx} + wW_x)\hat{i} = (Po_{tx} + uU_x + vV_x)\hat{i} \\ (Po_{ry} + wW_y)\hat{j} = (Po_{ty} + uU_y + vV_y)\hat{j} \\ (Po_{rz} + wW_z)\hat{k} = (Po_{tz} + uU_z + vV_z)\hat{k} \end{cases} . \tag{5}$$

Eliminando os termos $\hat{i}$, $\hat{j}$ e $\hat{k}$ e rearranjando os termos:

$$\begin{cases} uU_x + vV_x - wW_x = Po_{rx} - Po_{tx} \\ uU_y + vV_y - wW_y = Po_{ry} - Po_{ty} \\ uU_z + vV_z - wW_z = Po_{rz} - Po_{tz} \end{cases} . \tag{6}$$

Colocando o sistema de equação na forma matricial, tem-se então que:

$$\begin{bmatrix} U_x & V_x & -W_x \\ U_y & V_y & -W_y \\ U_z & V_z & -W_z \end{bmatrix} \cdot \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} Po_{rx} - Po_{tx} \\ Po_{ry} - Po_{ty} \\ Po_{rz} - Po_{tz} \end{bmatrix} \tag{7}$$

que pode ser escrito como um sistema na forma matricial:

$$V \cdot K = P \tag{8}$$

Isolando a matriz de coeficientes K:

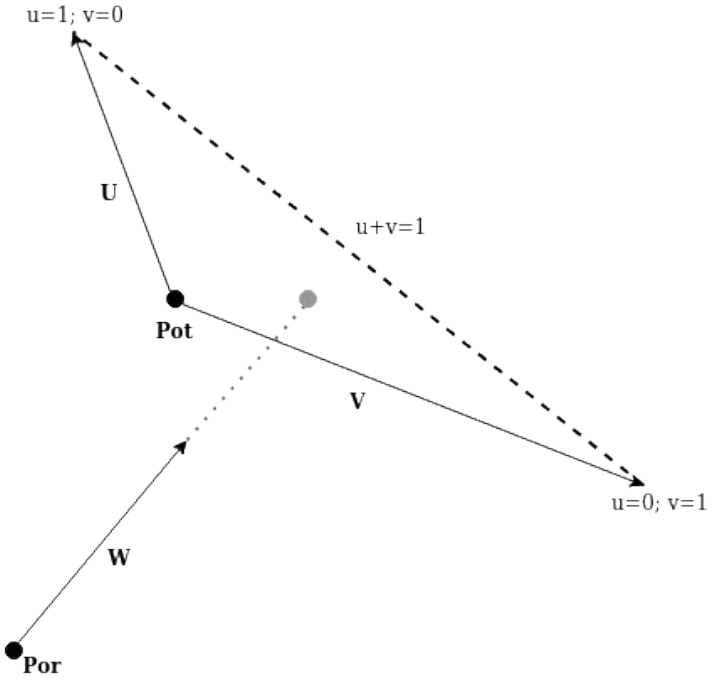
$$K = V^{-1} \cdot P \tag{9}$$

Obtendo-se K a partir da matriz inversa de V, multiplicada pela matriz P, temos os valores de u, v e w em que ocorre a interseção entre a reta e o plano em questão.

Para esse trabalho, entretanto, necessita-se encontrar a interseção da reta com um triângulo arbitrário, ao invés de um plano infinito. É possível particularizar o caso do

plano em questão para um triângulo, limitando-se os valores permitidos para os escalares u e v. Considerando que $\vec{U}$ e $\vec{V}$ não estão normalizados, eles contém implicitamente o tamanho da aresta do triângulo. Quando $u = 1$ e $v = 0$, o ponto corresponde ao vértice superior do triângulo, da mesma forma, quando $u = 0$ e $v = 1$, o ponto corresponde ao vértice direito do triângulo. O triângulo é delimitado por uma reta que liga esses dois pontos, que corresponde a $u + v = 1$. A figura 2 mostra essa reta representada por uma linha tracejada. Além disso, a interseção só é válida se ocorre na direção positiva de $\vec{W}$, de forma que a só é detectada uma interseção que ocorre na frente do raio.

Figura 2 – Reta e delimitação do plano de forma a representar um triângulo.



Fonte: Autoria própria

Dessa forma a interseção calculada só é válida para o triângulo quando as variáveis obedecem as seguintes restrições:

- $u \geq 0$, de modo que não é permitido nenhum ponto na direção "contrária" de $\vec{U}$;
- $v \geq 0$, de modo que não é permitido nenhum ponto na direção "contrária" de $\vec{V}$;
- $u + v \leq 1$, de modo que não é permitido nenhum ponto acima da reta que liga os pontos limites quando $u + v = 1$;
- $w \geq 0$, de modo que só é detectada uma interseção que ocorre na direção positiva do vetor $\vec{W}$, ou seja, na frente do raio.

3.2.2 Sistemas, convolução e resposta ao impulso

De acordo com (OPPENHEIM; WILLSKY, 2010) um sistema é uma interconexão de componentes, que ao se aplicar um sinal de entrada $x(t)$ é obtido na saída um sinal $y(t)$

$$x(t) \rightarrow y(t) \tag{10}$$

ou no caso discreto:

$$x[n] \rightarrow y[n] \tag{11}$$

A resposta ao impulso $h(t)$ de um sistema descreve o seu comportamento, e é obtida na saída quando um impulso $\delta(t)$ é aplicado na entrada do sistema. O mesmo vale para o caso discreto, no qual tem-se a resposta $h[n]$ quando se aplica um impulso $\delta[n]$ na entrada do sistema.

No caso discreto, um sinal descrevendo um impulso unitário é descrito como na equação 12:

$$\delta[n] = \begin{cases} 0, n \neq 0 \\ 1, n = 0 \end{cases} . \tag{12}$$

Um sinal discreto no tempo pode ser decomposto em uma combinação de impulsos atrasados, como mostra a equação 13:

$$x[n] = \sum_{k=-\infty}^{+\infty} x[k]\delta[n-k] \tag{13}$$

Além disso, tendo a resposta ao impulso $h[n]$ de um sistema linear invariante no tempo juntamente com um sinal qualquer no tempo $x[n]$ a ser colocado como entrada no sistema, pode-se encontrar o sinal de saída produzido.

$$y[n] = x[n] * h[n] \tag{14}$$

que é a chamada convolução entre $x[n]$ e $h[n]$, calculada pela soma de convolução descrita pela equação 15

$$y[n] = \sum_{k=-\infty}^{+\infty} x[k]h[n-k] \tag{15}$$

Portanto tem-se que a resposta ao impulso é uma característica muito importante de um sistema, a qual descreve seu comportamento quando um sinal qualquer é aplicado em sua entrada.

É possivel, através da transformada de Fourier, transformar os sinais no tempo $x[n]$, $y[n]$ e $h[n]$ para o domínio da frequência, a partir da equação 16, ou seja:

$$X(e^{j\omega}) = \sum_{n=-\infty}^{+\infty} x[n]e^{-j\omega n} \tag{16}$$

Tendo assim $X(e^{j\omega})$, $Y(e^{j\omega})$, que são respectivamente a entrada e saída do sistema no domínio da frequência, e $H(e^{j\omega})$, que é chamada de função de transferência do sistema, que nada mais é do que a resposta ao impulso do sistema no domínio da frequência.

Tem-se das propriedades da transformada de Fourier que uma convolução entre dois sinais no domínio do tempo é equivalente à uma multiplicação desses mesmos sinais no domínio da frequência, e vice-versa, como mostra a equação 17.

$$y[n] = x[n] * h[n] \rightarrow Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega}) \tag{17}$$

A operação inversa da transformada de Fourier pode ser feita através da equação 18, reconstituindo o sinal original no tempo a partir do sinal transformado:

$$x[n] = \frac{1}{2\pi}\int_{2\pi} X(e^{j\omega})e^{j\omega n}d\omega \tag{18}$$

Assim, para descobrir a resposta do sistema $y[n]$ à uma determinada entrada $x[n]$, um caminho é, ao invés de calcular a convolução entre $x[n]$ e $h[n]$, calcular a transformada

de Fourier dos sinais através da equação 16, obtendo $X(e^{j\omega})$ e $H(e^{j\omega})$, multiplicar $X(e^{j\omega})$ e $H(e^{j\omega})$, tendo $Y(e^{j\omega})$, e após isso, calcular a transformada inversa de $Y(e^{j\omega})$ descrita pela equação 18, obtendo $y[n]$. Esse procedimento será usado em uma etapa do trabalho, como será mostrado na seção 3.3.3.

Outro conceito importante utilizado nesse trabalho, é que para um impulso unitário descrito pela equacão 12, sabe-se que seu valor é diferente de zero apenas quando $n = 0$. Portanto tem-se pela equação 16 que a transformada de Fourier do impulso unitário é igual a 1 para todo ω.

$$\delta[n] \xrightarrow{F} 1 \tag{19}$$

Portanto, o impulso $\delta[n]$ possui todo o espectro de frequência.

3.3 DESENVOLVIMENTO

Foi implementado na linguagem *Python* o algoritmo de traçado de raios para a obtenção da resposta ao impulso, com a utilização de algumas bibliotecas nativas, como o *Numpy* para cálculos de operações vetoriais e matriciais, como o produto entre vetores, cálculo de matriz inversa. Essa biblioteca também foi utilizada ao adaptar o algoritmo para o domínio da frequência, para auxiliar em operações com números complexos e no cálculo da transformada inversa de Fourier, em que foi utilizada a sua função de *Fast Inverse Fourier Transform*. Também foi utilizada a biblioteca *SciPy* para a obtenção de filtros FIR *(Finite Impulse Response)*, e a biblioteca *Matplotlib* para a plotagem e visualização de gráficos e raios gerados, de forma a facilitar a depuração do algoritmo no decorrer do seu desenvolvimento e gerar figuras para este trabalho.

A seguir será abordado o desenvolvimento de cada etapa do trabalho e seu detalhamento. O código completo desenvolvido está disponível em: https://gitlab.com/GiovanniZanetti/tcc2-roomimpulseresponse.

3.3.1 Processamento individual de cada raio

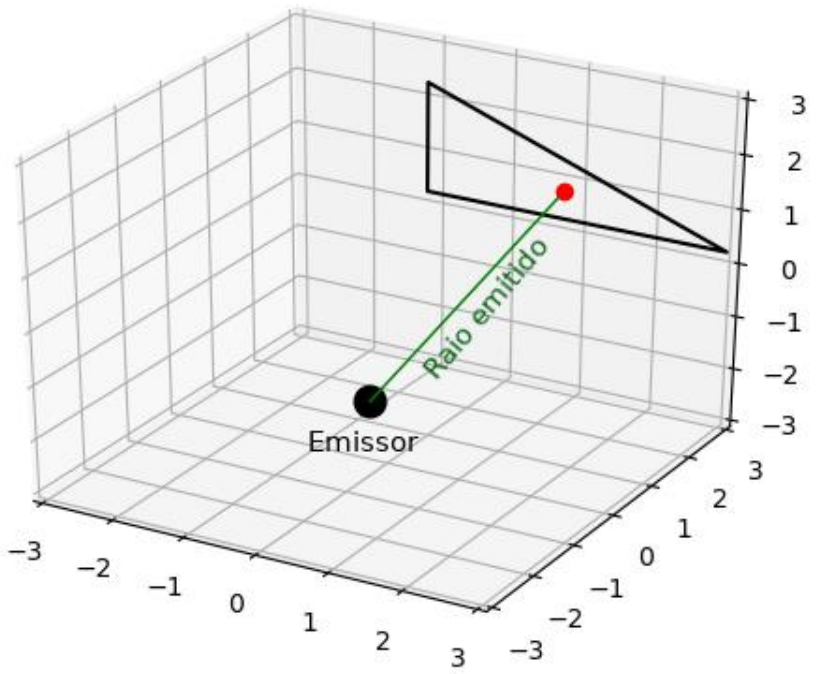
O algoritmo se baseia no processamento individual e independente de cada raio gerado pelo emissor e pelas reflexões/refrações causadas pelo ambiente. Cada raio é processado separadamente, desde a sua criação, até que seja extinto ao acontecer um dos seguintes casos:

- atenuação considerável da energia total do raio, sendo possível desprezá-lo;
- não existir uma interseção definida do raio com qualquer objeto (matriz V inversa indefinida para todos os objetos do ambiente).

A primeira etapa se refere à implementação do cálculo de interseção entre uma reta tridimensional (um raio individual) e um triângulo arbitrário, através da equação 9 mostrada na revisão de literatura, obtendo-se a matriz inversa em questão e os coeficientes u, v e w relacionados. O motivo pelo qual foram utilizados triângulos ao invés de retângulos, é que o cálculo da interseção da reta é matematicamente mais simples e computacionalmente menos custoso para o caso do triângulo, além de que qualquer geometria tridimensional pode ser decomposta em um conjunto de triângulos, possibilitando qualquer formato de ambiente e obstáculos.

A figura 3 mostra os conceitos e objetos fundamentais desse projeto. Uma esfera preta representando um emissor, uma reta verde representando um raio emitido por esse emissor, um triângulo preto representando uma parede e um círculo vermelho representando o ponto de interseção entre o raio e a parede. Esse ponto de interseção é obtido a partir

Figura 3 – Interseção entre raio e triângulo.



Fonte: Autoria própria

do coeficiente w obtido pela equação 9, juntamente com o ponto inicial do raio $\vec{Po_r}$ e seu vetor diretor $\vec{W}$, como descreve a equação 1. O valor de w no exemplo da figura 3 é de 6,88m, que representa a distância entre o ponto inicial do raio e o ponto de interseção com a parede. Com esse valor de w e a velocidade do som c especificada, é possível também calcular o tempo que o som demora para percorrer esses dois pontos, como será mostrado mais adiante na seção 3.3.2.

Após isso foi implementada uma função que faz esse cálculo para cada triângulo contido no ambiente. Dentre os triângulos em que houve interseção com este raio, o triângulo mais próximo do ponto inicial do raio (menor valor de w absoluto) é o que a colisão acontece primeiro. Portanto a interseção mais próxima é onde realmente ocorre a colisão do raio e do triângulo em questão, e as demais interseções são ignoradas.

Quando é encontrada essa interseção entre o raio e o triângulo, uma interação entre esses dois elementos (raio e parede) acontece. A energia desse raio é dividida em três partes:

- parte dela é refletida;
- parte dela é refratada (“atravessa” a parede);
- parte dela é absorvida pela parede e é perdida.

Cada triângulo possui um coeficiente de reflexão C_{Rfl} e outro de refração C_{Rfr} (valores no intervalo [0, 1]), e o raio incidente possui uma energia E_i. Portanto, ao interagir com o triângulo, é gerado um raio refletido com energia $E_{Rfl} = E_i \cdot C_{Rfl}$, e um raio

refratado com energia $E_{Rfr} = E_i \cdot C_{Rfr}$. A energia absorvida é a que não foi nem refletida e nem refratada, portanto é dada por $E_i - E_{Rfr} - E_{Rfl}$, mas não é utilizada nesse algoritmo.

Além da energia que é perdida pela absorção da parede, também é perdida energia pelo contato com o ar. Segundo (KULOWSKI, 1985), a energia atenuada no ar é calculada pela seguinte equação:

$$E_i = E_{i1} \cdot e^{-mw} \tag{20}$$

na qual E_{i1} é a energia que o raio tinha ao partir do seu ponto inicial, E_i a energia do raio após percorrer a distância w, e m um coeficiente representando o quão grande é a atenuação da energia pelo contato com o ar. A energia E_i é então a energia do raio incidente ao colidir com a parede.

A partir disso é gerado o raio refletido com energia E_{Rfl}, tendo como ponto de início o ponto de colisão, e a direção refletida. A direção do raio refletido é calculada a partir da direção do raio incidente $\vec{V_i}$ e do vetor normal do triângulo $\vec{N}$, como mostrado por (COMNINOS, 2006):

$$\vec{V_{Rfl}} = norm[\vec{V_i} - 2(\vec{V_i} \cdot \vec{N})\vec{N}] \tag{21}$$

na qual *norm* representa o vetor normalizado.

Além do raio refletido, é gerado também um raio refratado com energia E_{Rfr}, o qual simula a energia que atravessa a parede. Esse raio tem o ponto de início no ponto de colisão, e direção igual a direção V_i do raio incidente.

A figura 4 mostra, como na figura 3, um raio A partindo do emissor e o ponto de colisão com o triângulo, e além disso, um raio $B1$ mostrado em vermelho, que é refletido na direção de reflexão dada pela equação 21, e um raio $B2$ que é refratado na mesma direção do raio A. Ambos os raios refletido e refratado partem do ponto de interseção entre o raio A e a parede. As energias dos raios $B1$ e $B2$ são calculadas como explicado acima. Esse processo é feito para cada novo raio, até que todos os raios sejam eliminados de acordo com os critérios apresentados anteriormente.

3.3.2 Interseção do raio com o receptor

Quando um raio é interceptado por um receptor, ele tem uma energia final E_i. A direção em que o raio colide com o receptor não é considerada. Além disso, é calculada a distância total Δx que foi percorrida pelo raio, tendo assim diretamente o tempo Δt dado pela equação 22, a partir da velocidade do som c configurada.

$$\Delta t = \Delta x / c \tag{22}$$

Assim, cada raio que colide com um receptor, gera um pequeno impulso descrito por:

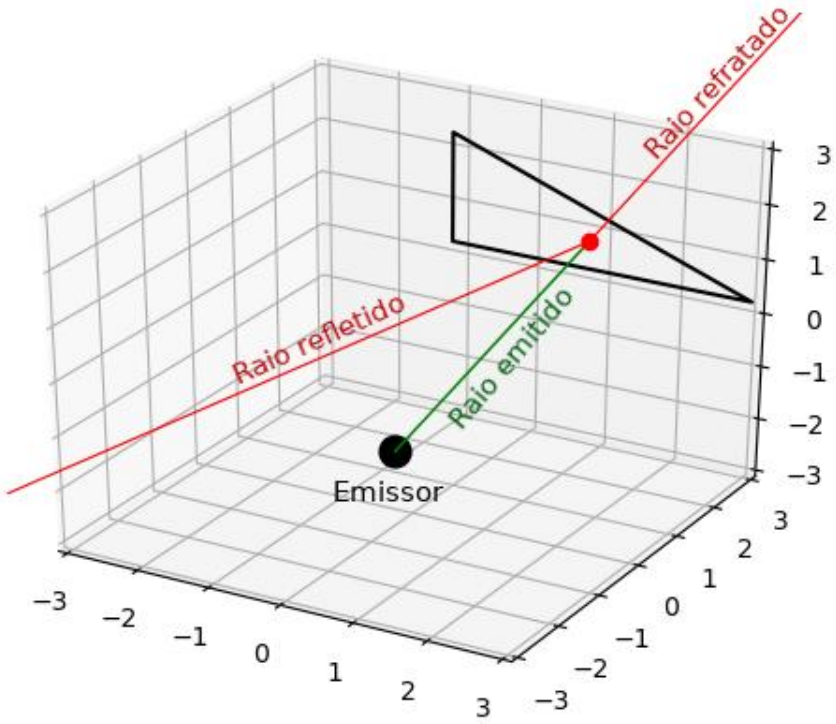
$$E_i \cdot \delta(t - \Delta t) \tag{23}$$

A resposta ao impulso total percebida pelo receptor em questão, é a soma de todos esses pequenos impulsos, pelo princípio da superposição.

Os receptores do ambiente podem ter qualquer estrutura geométrica, composta por triângulos, assim como as paredes e qualquer outra estrutura do ambiente.

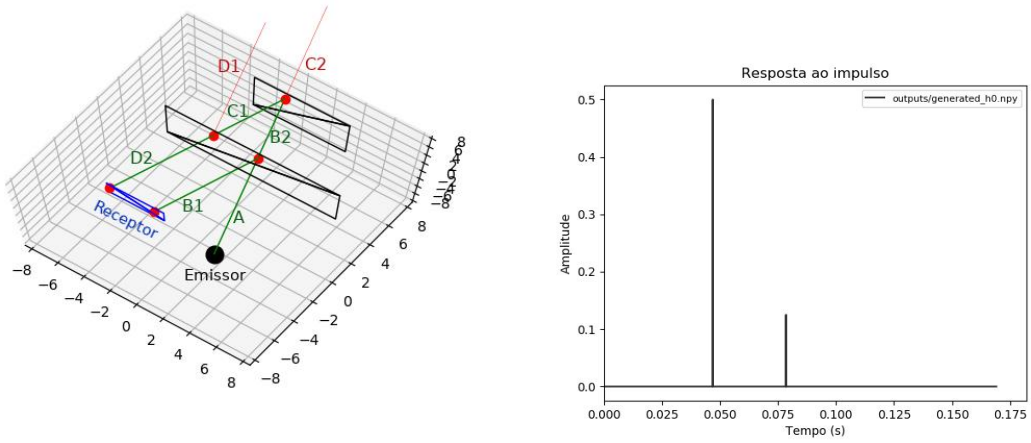
No exemplo da figura 5, foi considerado um coeficiente de reflexão de 0,5 e um coeficiente de refração 0,5, ou seja, nenhuma energia é absorvida, apenas de forma a ilustrar

Figura 4 – Interseção entre raio e triângulo: raio refletido e refratado.



Fonte: Autoria própria

Figura 5 – Interação do raio com o ambiente e resposta ao impulso gerada.



(a) (b)

Fonte: Autoria própria

o processo. A figura 5 mostra o raio A partindo do emissor e colidindo na primeira parede. Essa colisão gera os raios $B1$ e $B2$, ambos com energia 0,5. O raio $B1$ colide diretamente com o receptor (em azul). O raio $B2$ colide com a segunda parede, gerando dois novos raios $C1$ e $C2$, ambos com energia $0{,}5 \times 0{,}5 = 0{,}25$. O raio $C2$ refratado, vai para o infinito, o raio $C1$ refletido volta em direção a primeira parede, colidindo e gerando novos dois raios $D1$ e $D2$, ambos com energia $0{,}25 \times 0{,}5 = 0{,}125$. $D1$ vai para o infinito e $D2$ colide com o receptor.

Assim, temos o raio $B1$ chegando no receptor aproximadamente no instante $t = 0{,}04s$, e energia 0,5, e o raio $D2$ chegando no receptor aproximadamente no instante $t = 0{,}08s$, com energia 0,125. Isso reflete o que é mostrado na resposta ao impulso obtida, na figura 5b.

3.3.3 Adaptação para processamento com base na frequência

Para se obter a resposta ao impulso do ambiente, o emissor deve emitir um impulso $\delta(t)$, representado por vários raios que partem do emissor no mesmo instante $t = 0$, ou seja, cada raio emitido por este emissor representa uma parcela de energia desse impulso no tempo.

Para incorporar atenuações dependentes de frequência, foi feita uma adaptação no algoritmo de traçado de raios na qual considera-se para cada raio existente, ao invés de um único escalar descrevendo sua energia, uma série de valores contendo a energia contida em cada banda de frequência. Cada raio parte do emissor contendo um valor inicial de energia total E_i, que é igual para todo espectro de frequência, sendo descrita por $X(e^{j\omega}) = E_i$, equivalente a um impulso no tempo $E_i\delta(t)$.

Da mesma forma, as paredes contidas na sala podem ser configuradas de forma a se comportar como um filtro $F(e^{j\omega})$, atenuando a energia do raio de maneira diferente para cada banda de frequência sonora. Nesse caso, cada parede contida no ambiente possui não um único par de coeficientes de reflexão C_{Rfl} e de refração C_{Rfr}, mas sim uma série de coeficientes de reflexão e refração, sendo um para cada banda de frequência, formando estes um filtro de reflexão $F_{Rfl}(e^{j\omega})$ e outro de refração $F_{Rfr}(e^{j\omega})$.

Como os filtros utilizados no algoritmo devem ter um tamanho finito, a frequência não é contínua, e sim discreta. Sendo assim, definiu-se que o algoritmo considera, para todas as paredes, filtros de tamanho 512. Ou seja, cada filtro possui 512 bandas de frequência, todas de mesmo tamanho, cada uma compreendendo 1/512 de toda a faixa de frequência considerada, de $0Hz$ até $11025Hz$, que é a frequência de Nyquist (ZAWISTOWSKI; SHAH, 2021) para a taxa de amostragem de $22050Hz$ utilizada. Da mesma forma, a energia em frequência do raio possui o mesmo tamanho e número de bandas, compreendendo a mesma faixa de frequências. O número de bandas escolhido foi um meio termo encontrado, de forma a balancear o tempo de processamento e precisão.

Sendo assim, a energia contida no raio em $X(e^{j\omega})$ pode ser atenuada por um filtro em frequência de uma parede, descrito por $F(e^{j\omega})$.

$$X'(e^{j\omega}) = X(e^{j\omega}).F(e^{j\omega}) \tag{24}$$

Portanto, ao atenuar a energia de $X(e^{j\omega})$, temos um sinal resultante $X'(e^{j\omega})$, que pode ser novamente filtrado ao sofrer novas reflexões/refrações, e que ao chegar no receptor como um sinal $Y(e^{j\omega})$, pode ser transformado para o domínio do tempo pela transformada inversa discreta de Fourier descrita pela equação 18, resultando em um sinal no tempo $y[n]$ que corresponde à contribuição causada por aquele raio na resposta ao impulso. O

valor de n pode ser então efetivamente transformado para a unidade de segundos, com base na taxa de amostragem utilizada.

Entretanto, em alguns casos específicos, o valor da transformada inversa de Fourier de $Y(e^{j\omega})$ pode resultar em uma resposta não-causal, ou seja, uma resposta que possui valores não nulos para $n < 0$ no domínio do tempo. Por essa característica, uma resposta não-causal não representa bem o mundo real. Isso pode acontecer, por exemplo, no caso em que são utilizados filtros ideais retangulares nas paredes, os quais tem como característica uma resposta ao impulso não-causal. Isso se propagaria para os raios até o receptor, resultando em uma resposta final não-causal, sendo incoerente com a realidade.

Para contornar esse problema, uma alternativa seria, ao invés de utilizar filtros retangulares nas paredes, utilizar filtros com resposta ao impulso finita (*FIR - Finite Impulse Response*), que têm a característica de possuir uma resposta ao impulso causal, sendo mais condizentes com a realidade. Uma característica desse tipo de filtro é que eles possuem um deslocamento em fase: no decorrer das reflexões nas paredes, esse deslocamento em fase poderia se acumular indeterminadamente, o que se somaria ao deslocamento já existente decorrente do atraso do raio para chegar no receptor, resultando em um atraso total diferente do desejado.

Uma segunda alternativa - que foi utilizada como solução nesse trabalho - foi aplicar o filtro FIR apenas na última etapa de processamento do raio, ao chegar no receptor. Nesse caso são utilizados filtros ideais retangulares nas paredes, sem deslocamento de fase. Assim, em cada reflexão a energia do raio sofre uma filtragem em frequência apenas na sua amplitude, sem alteração de fase, eliminando o problema da acumulação de fase. Quando o raio chega no receptor, após sofrer várias reflexões e atenuações, ele possui uma energia $Y(e^{j\omega})$. Esse sinal é então transformado, ainda no domínio da frequência, em um sinal cuja resposta no tempo tem uma característica finita e causal. Essa transformação é feita através da função `firls()` da biblioteca *SciPy*. Ao ser transformado, a sua resposta ao impulso $y[n]$, que poderia ou não ser não-causal, irá se transformar em uma resposta causal e finita, com valores nulos para todo $n < 0$.

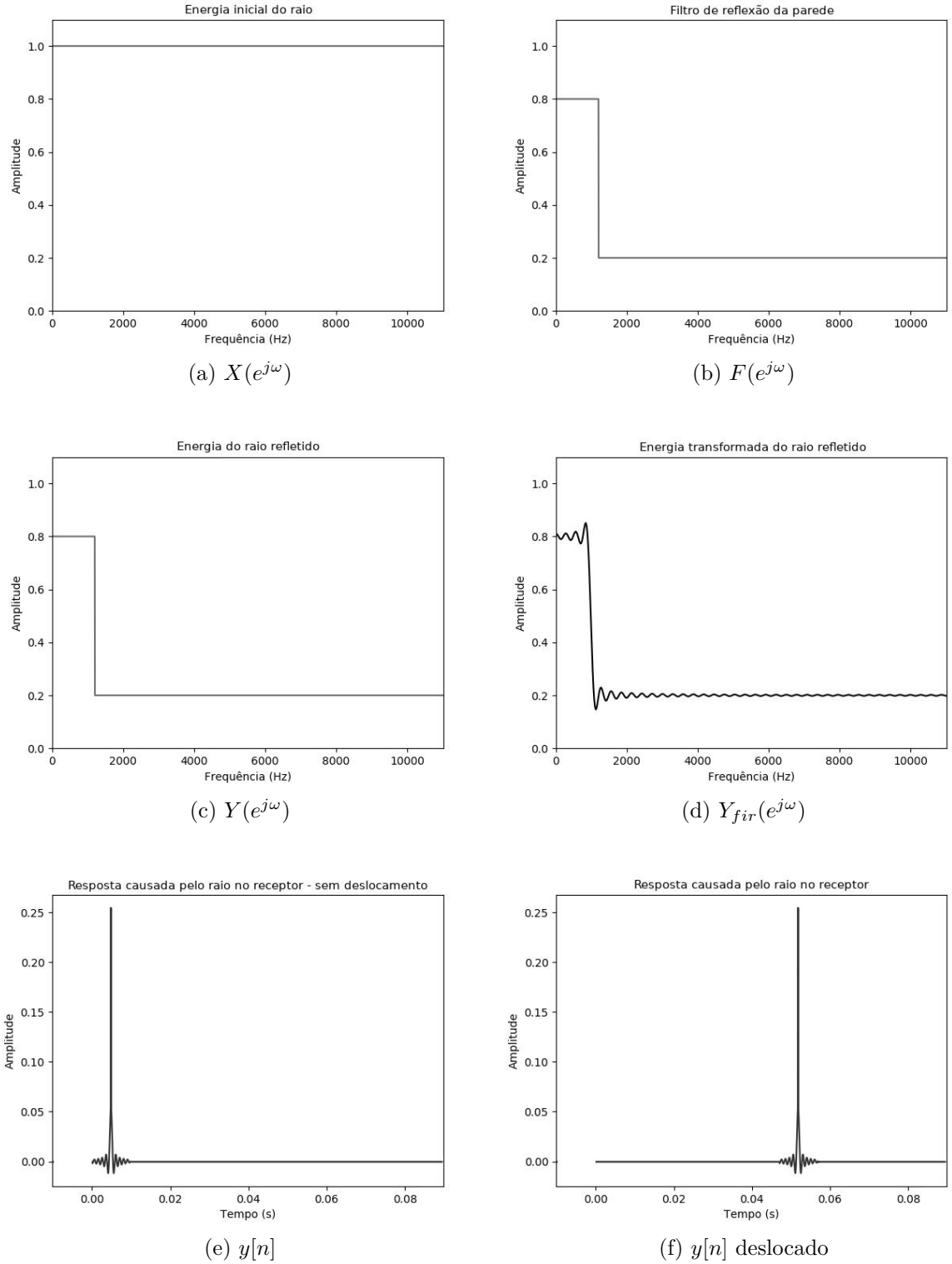
A função `firls()` utilizada, calcula os coeficientes de um filtro FIR, ou seja, um filtro que tem uma resposta no tempo finita e causal, cuja resposta em frequência tem a melhor aproximação com a resposta em frequência desejada. No caso utilizado nesse trabalho, a função não está sendo utilizada para se encontrar um filtro, mas sim para transformar um sinal qualquer na frequência (a energia do raio $Y(e^{j\omega})$), em um sinal aproximado $Y_{fir}(e^{j\omega})$ cuja resposta no tempo terá um característica finita e causal.

Portanto, para exemplificar todo este processo de forma mais simples, é ilustrado o caso em que é considerado o seguinte cenário: um único raio parte do emissor como um impulso (contendo todo o espectro de frequência como mostrado pela equação 19), reflete na parede, e atinge o receptor. Considera-se que a parede se comporta como um filtro em frequência, denotado por $F(e^{j\omega})$. Nesse caso a energia do raio $X(e^{j\omega})$ é atenuada conforme o filtro da parede, resultando num raio com energia $Y(e^{j\omega})$ que após isso colide com o receptor. Ao chegar no receptor, o sinal é transformado em um sinal finito e causal através de seus pontos de inflexão, resultando em $Y_{fir}(e^{j\omega})$. Então é calculada a transformada discreta inversa de Fourier da energia em frequência do raio $Y_{fir}(e^{j\omega})$, obtendo-se a resposta no tempo $y[n]$. Essa resposta é então deslocada no tempo conforme o tempo que o raio em questão demorou para chegar no receptor desde sua partida, da mesma forma que anteriormente, de acordo com o Δt do raio conforme a equação 22.

A figura 6 ilustra esse procedimento que foi exemplificado, da seguinte forma:

- Figura 6a: energia em frequência $X(e^{j\omega})$ de um raio que partiu do emissor, antes de colidir com a parede, contendo todo o espectro de frequência

Figura 6 – Etapas de uma reflexão e filtragem em frequência.



(a) $X(e^{j\omega})$ (b) $F(e^{j\omega})$

(c) $Y(e^{j\omega})$ (d) $Y_{fir}(e^{j\omega})$

(e) $y[n]$ (f) $y[n]$ deslocado

Fonte: Autoria própria

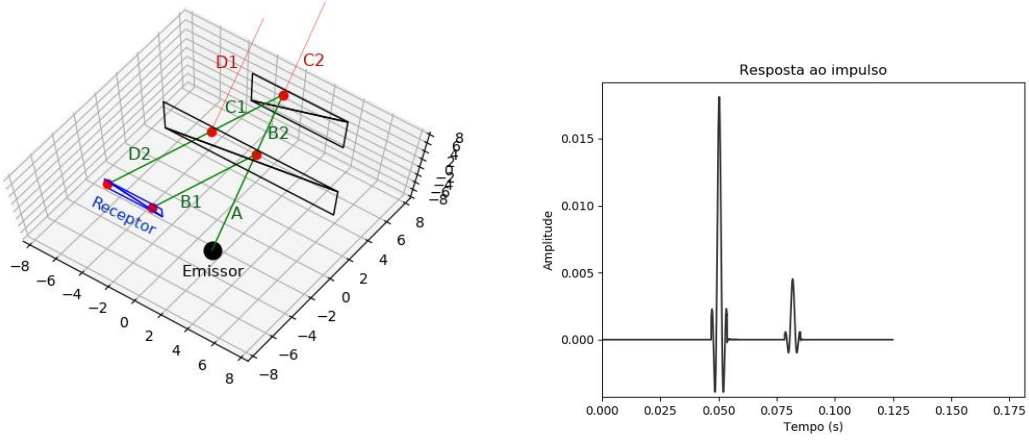
- Figura 6b: filtro da parede $F(e^{j\omega})$, que é um filtro ideal passa-baixa retangular, com frequência de corte em 1000Hz, em que 80% da energia das frequências menores que 1000Hz é refletida, e 20% da energia das frequências maiores que 1000Hz é refletida

- Figura 6c: energia do raio refletido após ser filtrada pela parede, denotada por $Y(e^{j\omega})$, resultado da multiplicação entre $X(e^{j\omega})$ e $F(e^{j\omega})$.
- Figura 6d: a mesma energia, mas transformada para um sinal $Y_{fir}(e^{j\omega})$ que no tempo tem comportamento causal e finito.
- Figura 6e: energia do raio refletido no domínio do tempo $y[n]$, obtida pela transformada inversa de Fourier de $Y_{fir}(e^{j\omega})$.
- Figura 6f: deslocamento no tempo feito em cima de $y[n]$ com base no valor de Δt, obtido pela equação 22, que representa o deslocamento provocado pelo tempo que o raio demorou para chegar no receptor depois de partir do emissor. Esse sinal deslocado representa então a resposta que este raio em específico provocou no receptor.

Esse processo de filtragem é feito para cada raio, percorrendo todas as reflexões e refrações, até que este raio chegue ao receptor. As respostas de cada raio quando chegam ao receptor, após a transformada inversa de Fourier e o deslocamento, são então somadas formando a resposta ao impulso final percebida pelo receptor. Vale ressaltar que durante as reflexões e refrações, a filtragem de energia é feita toda no domínio da frequência, através da multiplicação entre a energia do raio e do filtro da parede em questão. O único momento que essa energia é transformada para o domínio do tempo é quando o raio chega no receptor.

A figura 7 ilustra dois raios chegando em momentos diferentes após passarem por caminhos e reflexões/refrações diferentes, da mesma forma que foi mostrado na figura 5, mas considerando a filtragem em frequência, com as duas paredes se comportando como um filtro passa baixa ideal com frequência de corte de $400Hz$, tanto para reflexão quanto para refração. Esse valor de $400Hz$ é um valor qualquer escolhido, apenas de forma a exemplificar o processo.

Figura 7 – Interação do raio com o ambiente e filtragem por frequência, e resposta ao impulso gerada.



(a) (b)

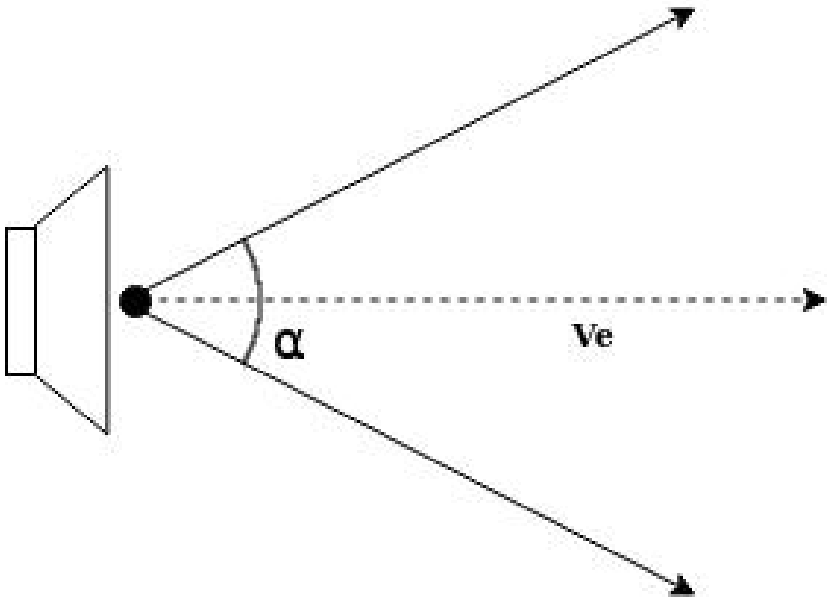
Fonte: Autoria própria

Em complemento a isso, o coeficiente de atenuação no ar m, descrito pela equação 20, também foi adaptado de forma a ser dependente da frequência, sendo da forma $m(e^{j\omega})$. Como a energia do raio também é dependente da frequência ω, a mesma equação pode ser utilizada diretamente.

3.3.4 Geração de raios a partir do emissor

Para que a execução do algoritmo produza uma resposta ao impulso mais próxima da real, é necessário que sejam gerados não só um, mas vários raios pelo emissor sonoro, partindo de um ponto em comum no centro desse emissor. O emissor tem uma característica de diretividade, ou seja, os raios não são necessariamente emitidos para todas as direções, mas podem ser emitidos em direções mais próximas ao que aponta a "frente" do emissor, de acordo com um certo ângulo de abertura α especificado antes de executar algoritmo.

Figura 8 – Representação do emissor e o ângulo de abertura.



Fonte: Autoria própria

A figura 8 mostra esse ângulo de abertura, representado por α. O emissor transmite raios de acordo com sua direção $\vec{V_e}$, dentro da região delimitada pelo ângulo de abertura. Estipulando um ângulo de abertura de 360 graus, a emissão é igual para todas as direções envolta do emissor.

Cada raio emitido tem um vetor diretor $\vec{W}$, como mostrado na equação 1. Para calcular o vetor $\vec{W}$ que obedeça as condições de direção do emissor e ângulo de abertura, são utilizadas coordenadas esféricas.

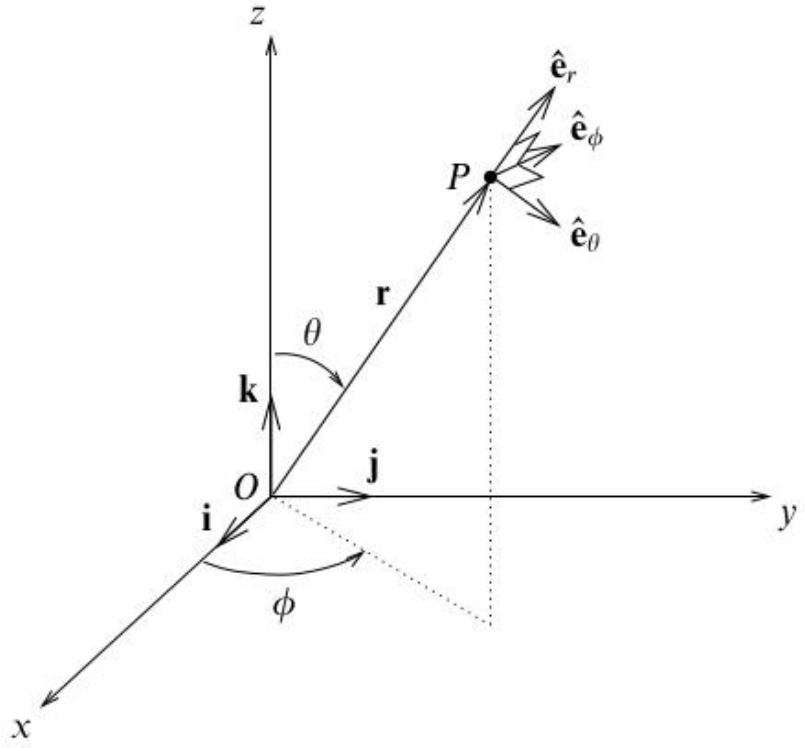
De acordo com (RILEY; HOBSON; BENCE, 2006), a conversão de coordenadas esféricas para coordenadas cartesianas é feita pelas seguintes equações:

$$\begin{aligned} x &= r\ \sin\theta\ \cos\phi, \\ y &= r\ \sin\theta\ \sin\phi, \\ z &= r\ \cos\theta \end{aligned} \tag{25}$$

na qual os valores de r, θ e ϕ são representados na figura 9

A partir dos ângulos iniciais que descrevem a direção do emissor θ_e e ϕ_e, e um ângulo de abertura α, os raios gerados devem ser limitados de forma que seus ângulos de

Figura 9 – Representação das coordenadas esféricas.



Fonte: (RILEY; HOBSON; BENCE, 2006)

direção θ_r e ϕ_r obedeçam as seguintes restrições:

$$\begin{aligned} \theta_e - \alpha < \theta_r < \theta_e + \alpha \\ \phi_e - \alpha < \phi_r < \phi_e + \alpha \end{aligned} \tag{26}$$

O algoritmo recebe como entrada o número de raios a serem gerados pelo emissor. Esses raios são então gerados em direções aleatórias que obedeçam os limites de ângulo impostos pela equação 26. Os ângulos aleatoriamente gerados são convertidos para coordenadas cartesianas pela equação 25, sendo a partir disso calculado o vetor diretor $\vec{W}$ do raio. Como o vetor $\vec{W}$ tem módulo unitário, o valor de r em coordenadas esféricas é igual a 1, podendo ser suprimido da equação. A partir desse vetor diretor e o ponto que representa a posição do emissor, é possível gerar a equação da reta que o representa o raio.

3.3.5 Geometria do ambiente tridimensional

O ambiente tridimensional que representa a sala, cuja acústica é simulada pelo algoritmo, é descrito por um arquivo no formato `.obj` (*Wavefront obj*) (Library of Congress, 2020), que é um formato de arquivo de fácil interpretação humana e suportado por vários programas de edição 3D, como o Blender (Blender Foundation, 2021) - programa *open-source* que suporta edição de objetos tridimensionais, e foi utilizado no modelamento geométrico dos ambientes de teste para o desenvolvimento desse projeto. O arquivo descreve a posição dos vértices de cada triângulo contido no ambiente, e tem a seguinte estrutura básica:

```
o Wall_001
v 0.000000 2.000000 2.000000
v 0.000000 0.000000 2.000000
v 2.000000 0.000000 2.000000
v 2.000000 2.000000 2.000000
v 0.000000 2.000000 0.000000
v 0.000000 0.000000 0.000000
v 2.000000 0.000000 0.000000
v 2.000000 2.000000 0.000000
f 1 2 3 4
f 8 7 6 5
f 4 3 7 8
f 5 1 4 8
f 5 6 2 1
f 2 6 7 3
```

Cada linha precedida por "o" descreve o nome de um objeto e indica que as linhas que o prosseguem se referem à esse objeto. Cada linha precedida por "v" descreve a posição tridimensional de um vértice no ambiente, e cada linha descrita por "f" representa uma face do objeto, composta por vários vértices, na qual cada número representa o índice de cada vértice.

Todo objeto e toda geometria do ambiente pode ser descrita por triângulos, inclusive as paredes, receptores e emissores. Para diferenciar neste trabalho cada tipo desses três objetos, foi definido um padrão para seus nomes de acordo com seu início, dessa forma o programa interpreta esse nomes criando os objetos de acordo com o esperado.

- "Wall_" (parede)
- "Receiver_" (receptor)
- "Transmitter_" (emissor)

Todos os triângulos são descritos por três pontos $\vec{P_0}$, $\vec{P_1}$ e $\vec{P_2}$ no espaço tridimensional. A partir desses três pontos é possível calcular o vetor normal desse triângulo, que é utilizado durante a execução do algoritmo para calcular as direções de reflexão dos raios, conforme a equação 21 mostrada anteriormente. O cálculo do vetor normal $\vec{N}$ do triângulo é feito através do produto vetorial entre duas arestas do triângulo, como descrito pela equação 27.

$$\vec{N} = norm[(\vec{P_1} - \vec{P_0}) \times (\vec{P_2} - \vec{P_0})] \tag{27}$$

em que *norm* representa o vetor normalizado.

3.3.6 Paralelização do algoritmo com o *framework* Apache Spark

De acordo com o website da documentação do Apache Spark (Apache Software Foundation, 2021), o *Apache Spark* é um *engine* unificado para processamento de dados em larga escala, que provê uma API de alto nível disponível nas linguagens *Java*, *Scala*, *Python* e *R*. Em um nível mais alto, cada aplicação *Spark* possui um programa *driver* que executa a função principal e dispara as tarefas para um *cluster* de nós escravos que executam as tarefas paralelamente. A principal abstração do *framework Spark* é o *Resilient Distributed Dataset* (RDD), que é uma coleção (lista) de elementos particionados entre os nós do *cluster*, cujo processamento pode ser feito em paralelo. O RDD pode ser criado na aplicação

principal executada no *driver*, que se encarrega de distribuir os elementos do RDD para os nós e processá-los. Esses nós podem ser diferentes núcleos de um mesmo processador, ou vários computadores distribuídos. Essa forma de distribuição fica transparente ao desenvolvedor, que constrói o código sem precisar tratar desses detalhes. Além disso, o *Spark* possibilita os RDDs serem persistidos em memória, permitindo reusá-los de forma eficiente, além de recuperar automaticamente os RDDs quando ocorre alguma falha nos nós.

Uma segunda abstração do *Spark* utilizada nesse projeto são as variáveis compartilhadas *(broadcast variables)*. Por padrão, quando o *Spark* executa tarefas paralelas em diferentes nós, ele envia as variáveis utilizadas para cada nó. Algumas vezes essa variável precisa ser compartilhada entre os nós, e não existe a necessidade de refazer a sua cópia para cada nó toda vez que uma tarefa é executada. Nesse caso é possivel usar uma variável compartilhada, que serve como um *cache* em memória de uma variável, podendo esta ser acessada e reutilizada por todos os nós.

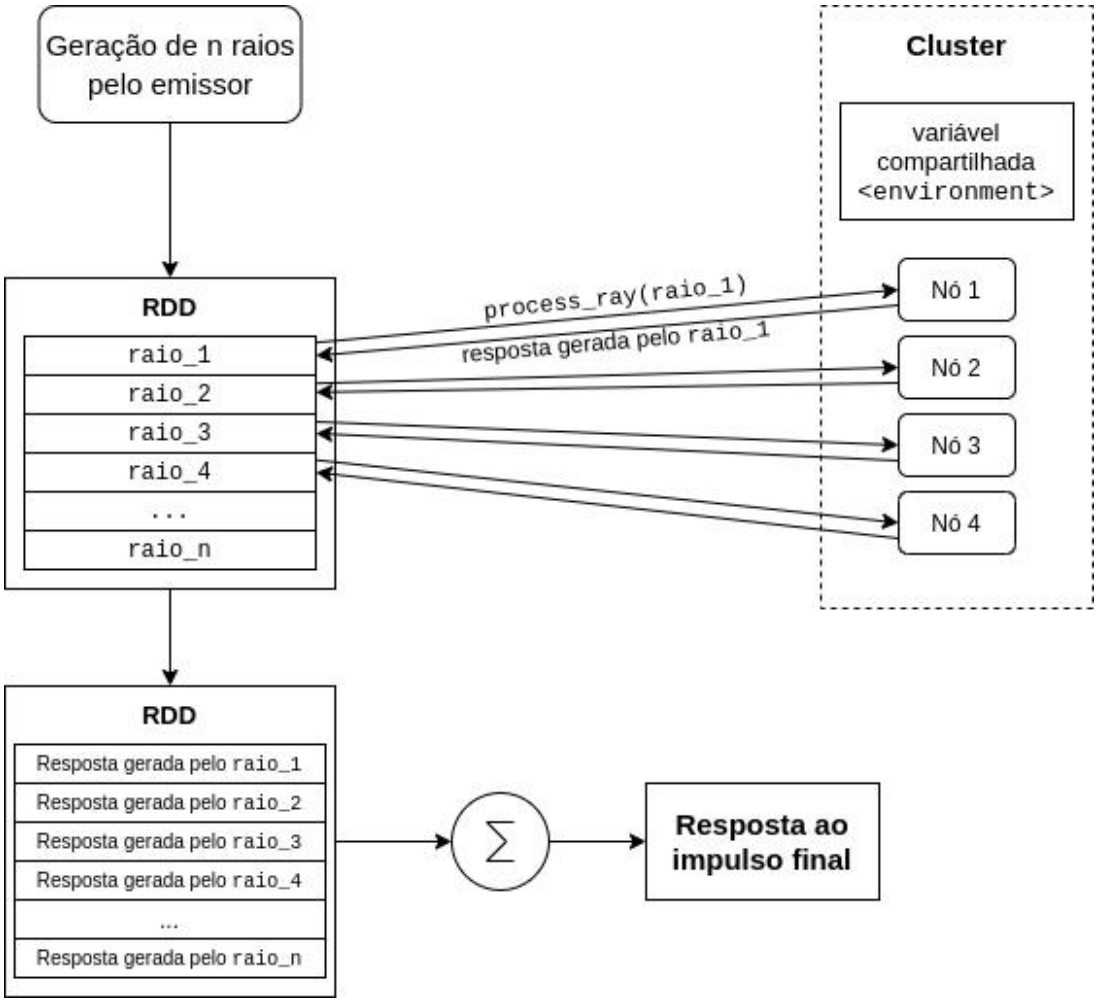
A paralelização do programa foi feita através de uma função `process_ray(ray, environment)` que se encarrega de processar cada raio que parte do emissor até sua extinção, ou seja, até que este sofra uma grande atenuação, ou até que não haja interseção com qualquer objeto no ambiente. Essa função retorna, após a finalização desse processamento, as respostas no tempo vistas pelo receptor, geradas por esse raio e seus filhos.

Cada raio inicial que parte do emissor é posto no RDD. Para cada raio contido nesse RDD é aplicada a função `process_ray(ray, environment)`, o processamento da função para cada elemento do RDD é direcionado a um nó do *cluster*, de acordo com a disponibilidade de cada nó. Após o processamento de cada raio e suas respostas individuais, tais respostas são somadas para produzir uma resposta total para o receptor.

O ambiente e seus parâmetros são fatores constantes durante todo o processamento, e são compartilhados entre todos os nós. Dessa forma foi criada uma variável compartilhada para armazenar os parâmetros de forma comum aos nós.

A lógica de processamento do código em Spark é exemplificada no diagrama mostrado na figura 10, de forma simplificada.

Figura 10 – Fluxo simplificado do programa em Spark.



Fonte: Autoria própria

4 RESULTADOS

4.1 COMPARAÇÃO COM RESPOSTA BASE

Para validação dos resultados obtidos pelo método implementado, foi feita a comparação com um resultado de uma resposta ao impulso calculada analiticamente em uma sala com formato e dimensões pré-estabelecidas. Essa resposta que serve de base comparativa foi obtida através de um método analítico para resolver a equação da onda acústica, conforme mostrado por (GOMES; BERTOLI; DEDECA, 2007), e só opera sobre modelos geométricos com o formato "caixa de sapato".

A sala em questão é uma sala com formato de paralelepípedo ou "caixa de sapato", com as dimensões $\vec{L} = (4{,}80m; 8{,}40m; 2{,}70m)$, com um emissor na posição $\vec{P_e} = (1{,}70m; 2{,}00m; 1{,}20m)$ e um receptor na posição $\vec{P_r} = (3{,}44m; 7{,}06m; 1{,}74m)$. A vista superior dessa sala juntamente com as posições do emissor (em cinza) e receptor (em azul) são mostradas na figura 11.

Considerou-se a velocidade do som $c = 330m/s$ e o coeficiente atenuação "do ar" nulo ($C_{ar} = 0$), ou seja, não há atenuação no ar. O filtro de reflexão de todas as paredes é $F_{Rfl}(e^{j\omega}) = 0{,}9423$ para todo ω e o filtro de refração de todas as paredes é $F_{Rfr}(e^{j\omega}) = 0$ para todo ω, o que significa que o coeficiente de reflexão é independente de frequência e igual a 0,9423, e o coeficiente de refração é independente de frequência e igual a 0; portanto a refração é desconsiderada. Toda a energia não refletida é absorvida pela parede, ou seja, o coeficiente de absorção é propositalmente de $1 - 0{,}9423 = 0{,}0577$, de forma a ser semelhante ao coeficiente teórico das paredes da sala, com base nas suas dimensões e seu tempo de reverberação real, através da Fórmula de Sabine (NEUBAUER R.; KOSTEK, 2021). O ângulo de abertura do emissor é de 360 graus, o que significa que ele emite raios uniformemente para todas as direções. O receptor utilizado tem formato quadrado, com as dimensões de 1m x 1m, de forma que mais raios o interceptem, e é direcionado conforme mostra a figura 11.

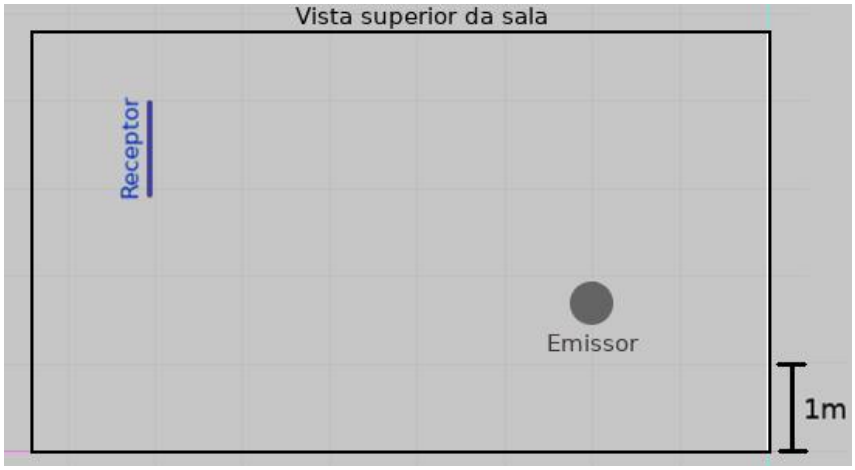
A resposta ao impulso teórica obtida analiticamente, juntamente com sua curva de decaimento, são mostradas na figura 12. A resposta ao impulso foi normalizada de forma que o pico tenha um valor de 1,0 e os outros valores alterados proporcionalmente. Essa resposta ao impulso será considerada abaixo como base de comparação para os resultados obtidos pelo algoritmo de traçado de raios implementado neste trabalho.

A curva de decaimento é calculada com base na resposta ao impulso, e mostra para cada instante de tempo t, quanto da energia total ainda está disponível após o tempo t na resposta ao impulso $h(t)$. De acordo com (SMITH, 2021), a curva de decaimento no instante t é calculada pela seguinte equação 28, denominada integral de Schroeder. No caso ilustrado a energia total foi normalizada para o valor de 1, que é o valor da energia restante no instante $t = 0$ para esse modelo teórico.

$$EDC(t) = \int_t^\infty h^2(\tau) d\tau \tag{28}$$

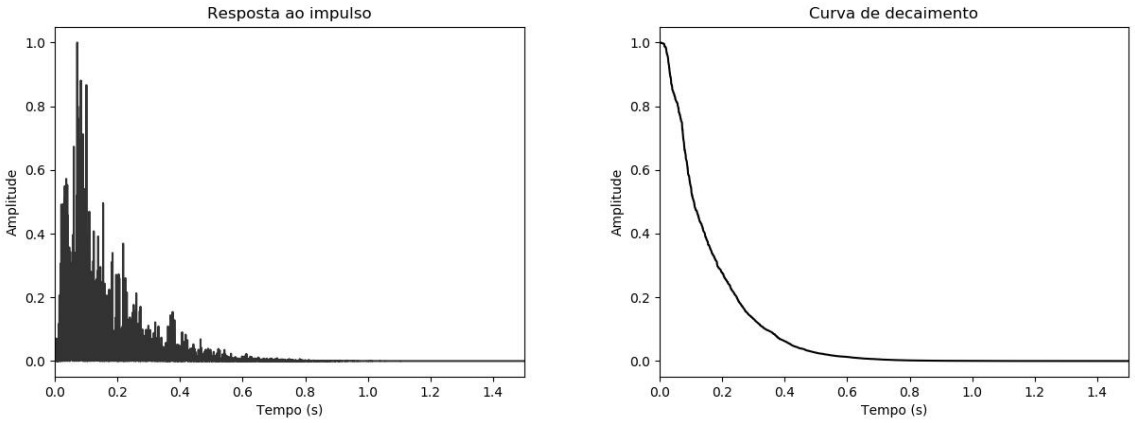
Para comparação e medição de semelhança entre as respostas encontradas pelo algoritmo e a resposta analítica teórica, foi usada uma métrica chamada erro médio quadrático (MSE ou *Mean Squared Error*), que compara ponto a ponto o erro quadrado, ou seja, a diferença ao quadrado entre cada elemento de duas séries temporais, e calcula uma média desses valores. O MSE pode ser formalmente descrito pela seguinte equação,

Figura 11 – Vista superior da sala utilizada nos testes, mostrando o receptor (em azul) e o emissor (em cinza).



Fonte: Autoria própria

Figura 12 – Resposta ao impulso e curva de decaimento obtidas analiticamente.



(a) (b)

Fonte: Autoria própria

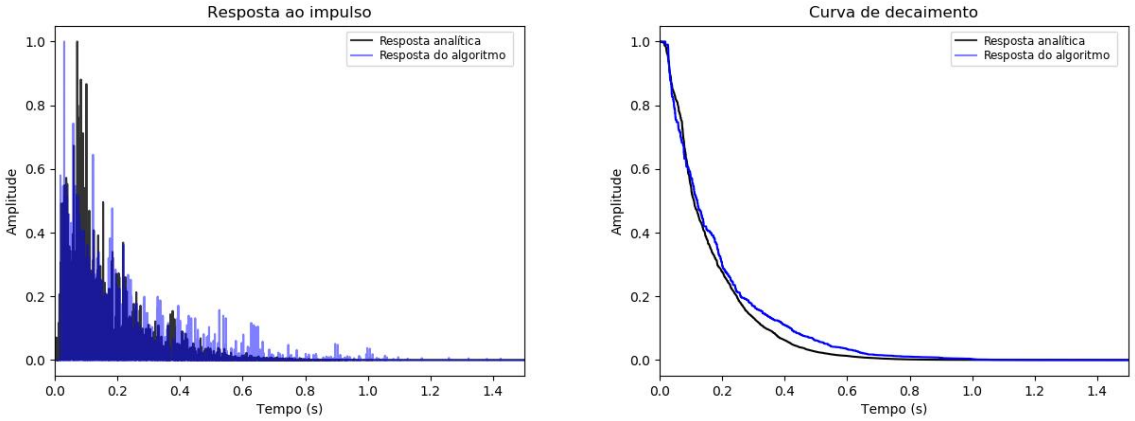
considerando duas séries de valores x e y a serem comparadas:

$$mse = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \tag{29}$$

Para um primeiro teste de comparação entre a resposta encontrada pelo algoritmo e a resposta analítica, o algoritmo foi executado com os parâmetros mencionados anteriormente, sendo emitidos 500 raios pelo emissor. Isso resultou na seguinte resposta ao impulso e curva de decaimento, mostradas em azul na figura 13, e sobrepostas com a resposta analítica de base, em preto. A execução do algoritmo nessas condições resultou em um MSE para a resposta ao impulso de 0,0152, e um MSE para a curva de decaimento de 0,0012.

Aumentando o número de raios emitidos para 6 mil, obteve-se as seguintes respostas

Figura 13 – Respostas ao impulso e curva de decaimento da resposta analítica (em preto) e da resposta obtida pelo algoritmo (em azul) - 500 raios emitidos.

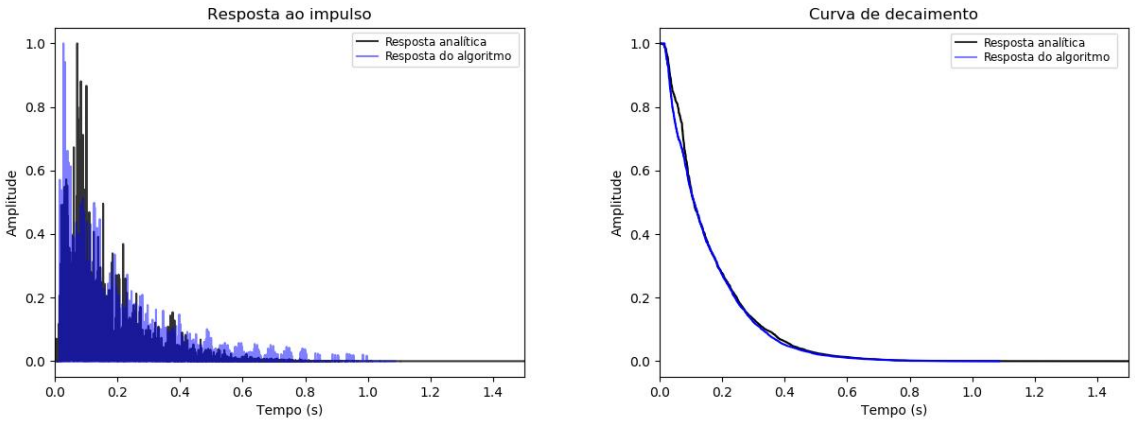


(a) (b)

Fonte: Autoria própria

mostradas na figura 14, novamente em azul e sobrepostas com a resposta base em preto. O resultado foi um MSE para a resposta ao impulso de 0,0078, e um MSE para a curva de decaimento de 0,0004.

Figura 14 – Respostas ao impulso e curva de decaimento da resposta analítica (em preto) e da resposta obtida pelo algoritmo (em azul) - 6 mil raios emitidos.



(a) (b)

Fonte: Autoria própria

É possível perceber que os erros médios quadráticos diminuíram ao se aumentar o número de raios que foram emitidos no início do algoritmo. Outro detalhe percebido é que a resposta ao impulso obtida pelo algoritmo tem um pico que ocorre em um tempo anterior do pico da resposta ao impulso obtida analiticamente. Essa é uma característica dos métodos geométricos de cálculo de RIRs, e isso também é refletido na curva de decaimento. Apesar disso, a resposta ao impulso obtida tem um comportamento bem parecido com a analítica, e também tem um tempo de decaimento muito próximo, que pode ser percebido

pela grande similaridade entre as curvas de decaimento, principalmente no caso em que foi utilizado 6 mil raios.

Outro detalhe percebido é que, pelo fato do receptor ter visada direta do emissor nessa sala, o menor tempo para um raio chegar no receptor deve ser igual ao tempo que se leva para percorrer a distância entre o emissor e o receptor, que pode ser obtida a partir dos valores de $\vec{P}_e$ e $\vec{P}_r$ descritos, resultando em um valor de 5,35m, por consequência um tempo de propagação de aproximadamente 16,3ms. Esse valor foi exatamente o instante de tempo em que a resposta ao impulso obtida pelo algoritmo se inicia.

Para uma melhor comparação entre os resultados de erro médio, o algoritmo foi executado diversas vezes com diferentes números de raios emitidos. Para cada número diferente de raios emitidos, o algoritmo foi executado 5 vezes, de forma que foi feita uma média entre os erros a serem apresentados. A tabela 1 resume esses dados em conjunto com a figura 15, mostrando a melhora da precisão dos resultados do algoritmo conforme o aumento do número de raios emitidos no início do algoritmo, percebida pela redução do erro médio quadrático. É mostrado tanto a média do MSE da resposta ao impulso (MSE), quanto a média do MSE da curva de decaimento (MSE decaimento). Na figura foi usada uma escala logarítmica no eixo x do gráfico, de forma a facilitar a visualização dos resultados.

Tabela 1 – Erro do algoritmo para diferentes números de raios emitidos.

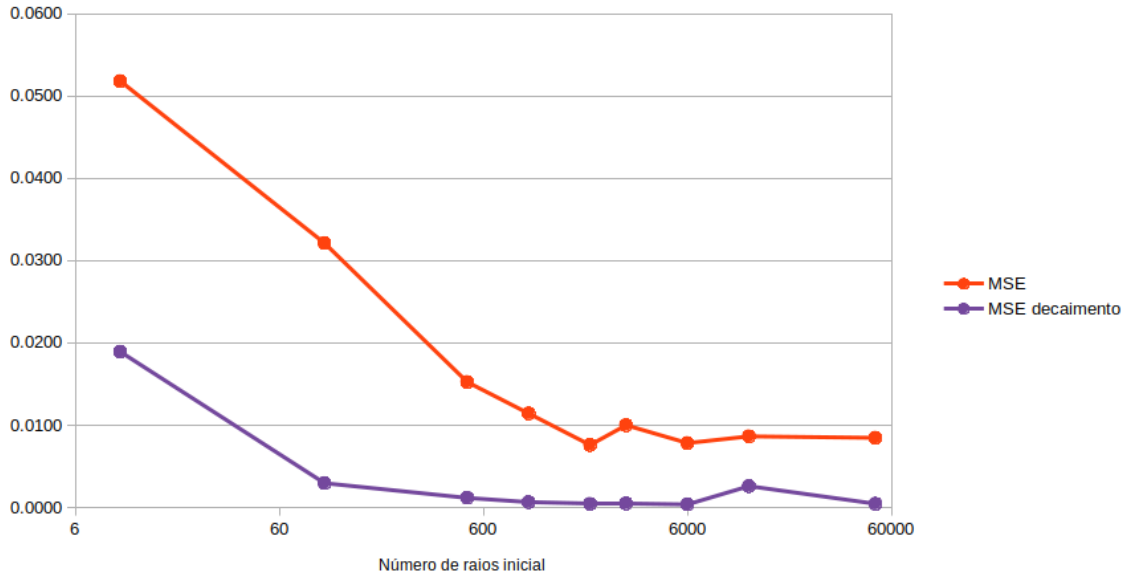
Núm. raios	**MSE**	**MSE curva decaimento**
10	0,0518	0,0189
100	0,0321	0,0030
500	0,0152	0,0012
1000	0,0114	0,0007
2000	0,0076	0,0004
3000	0,0100	0,0005
6000	0,0078	0,0004
12000	0,0086	0,0026
50000	0,0085	0,0005

4.2 PERFORMANCE

Além da comparação entre os erros médios obtidos, foi feita uma comparação para analisar o comportamento do programa quanto ao tempo de processamento do algoritmo, conforme a variação do número de raios emitidos, além de efeitos da paralelização, percebido ao executar o algoritmo com e sem a utilização do Spark. A tabela 2 mostra esses resultados. Para o modo utilizando a paralelização com o Spark foram usados diferentes números de núcleos do processador, de 1 até 4 (o máximo disponível na máquina utilizada para os testes), de forma a visualizar como essa variável impacta no tempo de processamento.

Não foi possível executar o algoritmo para mais que 50 mil raios, por limitações de memória da máquina utilizada, da mesma forma que para 6 mil e 12 mil raios no modo sem Spark. O gráfico da figura 16 mostra os mesmos resultados da tabela 2, novamente com escala logarítmica no eixo x, de modo a facilitar a visualização da diferença entre os tempos obtidos. Como esperado, o tempo de processamento aumenta conforme aumenta o número de raios emitidos.

Figura 15 – Erro do algoritmo para diferentes números de raios emitidos.



Fonte: Autoria própria

Tabela 2 – Tempo de processamento do algoritmo para diferentes números de raios emitidos.

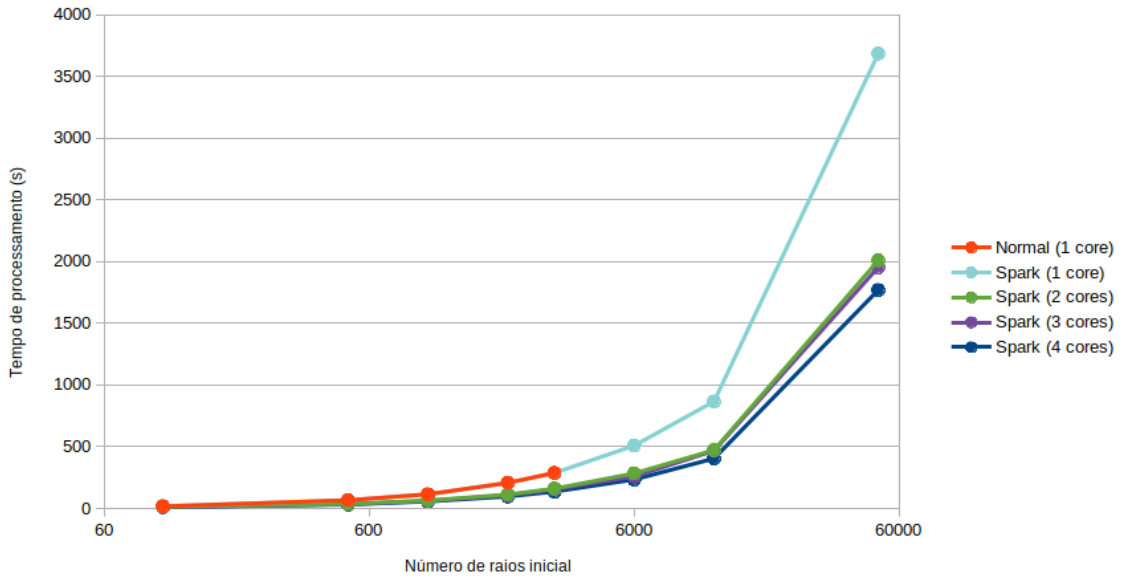
Núm. raios	**Tempo de processamento (s)**				
	Sem Spark	Com Spark			
	1 núcleo	1 núcleo	2 núcleos	3 núcleos	4 núcleos
100	17,2	19,7	13,8	11,9	10,7
500	67,3	67,4	37,1	34,8	32,1
1000	115,4	115,5	64,9	61,8	57,3
2000	207,2	209,4	113,0	107,8	97,4
3000	288,4	287,9	161,0	156,0	136,8
6000	-	509,7	284,5	259,7	234,4
12000	-	866,9	472,6	468,1	405,0
50000	-	3684,8	2009,9	1954,6	1770,1

O tempo de processamento obtido no caso sem Spark e no caso com Spark utilizando apenas 1 núcleo são bem similares, como era de se esperar. Uma mínima diferença pode acontecer devido ao gargalo na transferência de dados entre os nós do Spark e o seu driver, resultando em um tempo levemente maior para o modo com Spark nesse caso.

Apesar disso, é possível visualizar que conforme é aumentado o número de núcleos utilizados pelo algoritmo com o Spark, sua execução fica consideravelmente mais rápida em comparação ao modo sem Spark. Sendo assim, os resultados obtidos foram bastante satisfatórios, pois o tempo de processamento com paralelização com 4 núcleos foi em média aproximadamente metade do tempo de processamento sem paralelização.

O algoritmo tem duas etapas mais significantes em termos de tempo de processamento. Na etapa 1 acontecem os cálculos de interseção dos raios com os triângulos que representam as paredes do ambiente, a filtragem em frequência da energia dos raios

Figura 16 – Tempo de processamento do algoritmo para diferentes números de raios emitidos.



Fonte: Autoria própria

nas colisões, e o cálculo do Δt do raio ao chegar no receptor. Já na etapa 2 acontece a transformação da energia dos raios para um sinal causal e finito a partir da função `firls()` conforme explicado anteriormente, o cálculo da transformada inversa de Fourier desse sinal, e a somas das respostas de cada raio de forma a compor a resposta ao impulso total percebida pelo receptor. Nas execuções do algoritmo feitas no teste anterior, foi medido o tempo de cada uma dessas duas etapas, e calculada uma média de forma a comparar percentualmente qual o tempo de processamento de cada etapa. O resultado foi o seguinte:

- Etapa 1: 98,9%
- Etapa 2: 1,1%

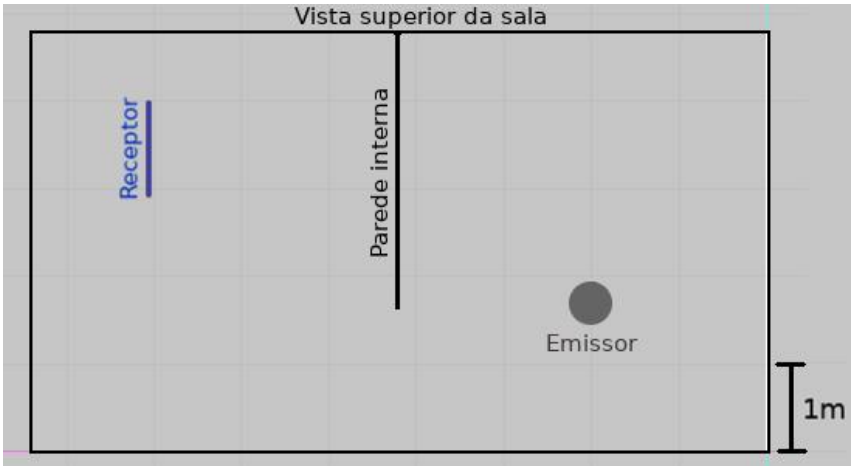
Portanto percebe-se que o tempo de processamento da primeira etapa é significativamente maior que o da segunda etapa. Isso se deve ao fato que a etapa 1 inclui o cálculo da interseção de cada raio existente com cada triângulo do ambiente, o que exige várias vezes o cálculo da matriz inversa de V, descrita na equação 9, que é uma operação bastante custosa computacionalmente.

4.3 AMBIENTES NÃO-CONVEXOS

O algoritmo implementado também lida com ambientes tridimensionais não-convexos. Para demonstrar essa possibilidade e validar os resultados, o algoritmo foi executado na sala mostrada na figura 17, que é idêntica à descrita nos testes anteriores com relação às suas dimensões e posições de emissor e receptor, com exceção apenas de que foi inserida uma parede interior na metade da sala, entre o emissor e o receptor, com uma "porta" de acesso entre as duas regiões, localizada no último terço dessa parede, representando uma porção não-convexa.

O algoritmo foi executado com um número inicial de raios igual a 1000, para duas situações: uma sem e outra com a parede central. Todas as paredes possuem as mesmas

Figura 17 – Vista superior da sala não-convexa utilizada nos testes, mostrando o receptor (em azul) e o emissor (em cinza), e uma parede interna.



Fonte: Autoria própria

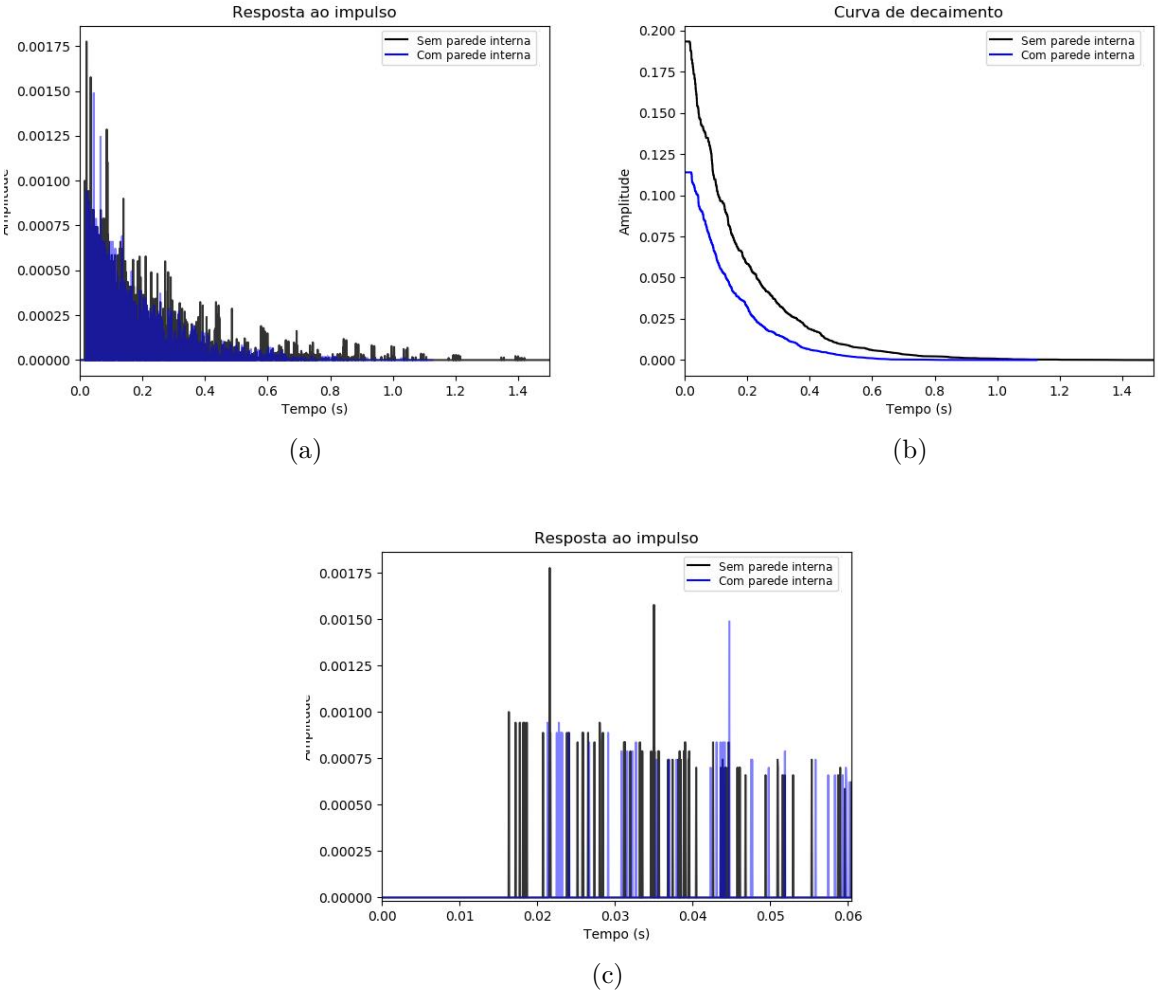
características de reflexão e refração descritas no teste anterior, ou seja, $C_{Rfl} = 0{,}9423$ e $C_{Rfr} = 0$. A sobreposição das respostas obtidas estão na figura 18 abaixo, em preto a resposta da sala sem a parede interior (sala convexa), e em azul a resposta da sala com a parede interior (sala não-convexa).

É possível perceber pela curva de decaimento na figura 18b, que para uma mesma energia inicial emitida, na sala com a parede interior, a energia total que chega no receptor é aproximadamente metade da que chega na sala sem a parede. Isso se deve ao fato de que o som tem um espaço menor para passar entre as duas regiões da sala separada por essa parede interna, fazendo com que menos raios cheguem ao receptor, ou sofram mais reflexões antes de chegar no receptor. A figura 18c mostra o início da resposta ao impulso mostrada na figura 18a, na qual se pode perceber que o tempo em que se inicia a resposta ao impulso na sala com a parede interior (em torno de $21ms$) é maior do que na sala sem a parede interior (em torno de $16ms$), devido ao fato que a parede interna bloqueia a passagem de raios que chegariam diretamente em linha reta entre o emissor e o receptor. Assim esses raios precisam passar por um caminho maior antes de chegar ao receptor.

Outro teste realizado no mesmo ambiente considerou novamente mil raios, mas alterando as características da parede interna, mantendo o coeficiente de absorção e dividindo a energia que antes era toda refletida, entre reflexão e refração, sendo os coeficientes de reflexão e refração $C_{Rfl} = C_{Rfr} = 0{,}47115$. A figura 19 mostra os resultados obtidos para esse caso. Novamente, a resposta em preto mostra o resultado obtido anteriormente sem a parede interna, e a resposta azul mostra o resultado obtido para essa execução do algoritmo.

A curva de decaimento mostrada na figura 19b mostra que a energia total que chega no receptor é menor que a metade do que na sala sem a parede interna, e a energia decai bem mais rapidamente, inclusive mais rápido do que no caso com a parede interna mas sem refração. Isso provavelmente se deve ao fato de que, apesar de agora parte dos raios passarem para o outro lado da sala "atravessando" a parede pela refração, o coeficiente de reflexão é bem menor que o do caso anterior. Portanto em uma única reflexão/refração, boa parte da energia já é perdida, enquanto no caso anterior, necessitava-se entre 12 e 13 reflexões para o raio perder essa mesma quantidade de energia. É possível observar

Figura 18 – Resposta ao impulso e curva de decaimento obtidas pelo algoritmo na sala sem a parede interna (em preto) e com parede interna (em azul) - sem refração.



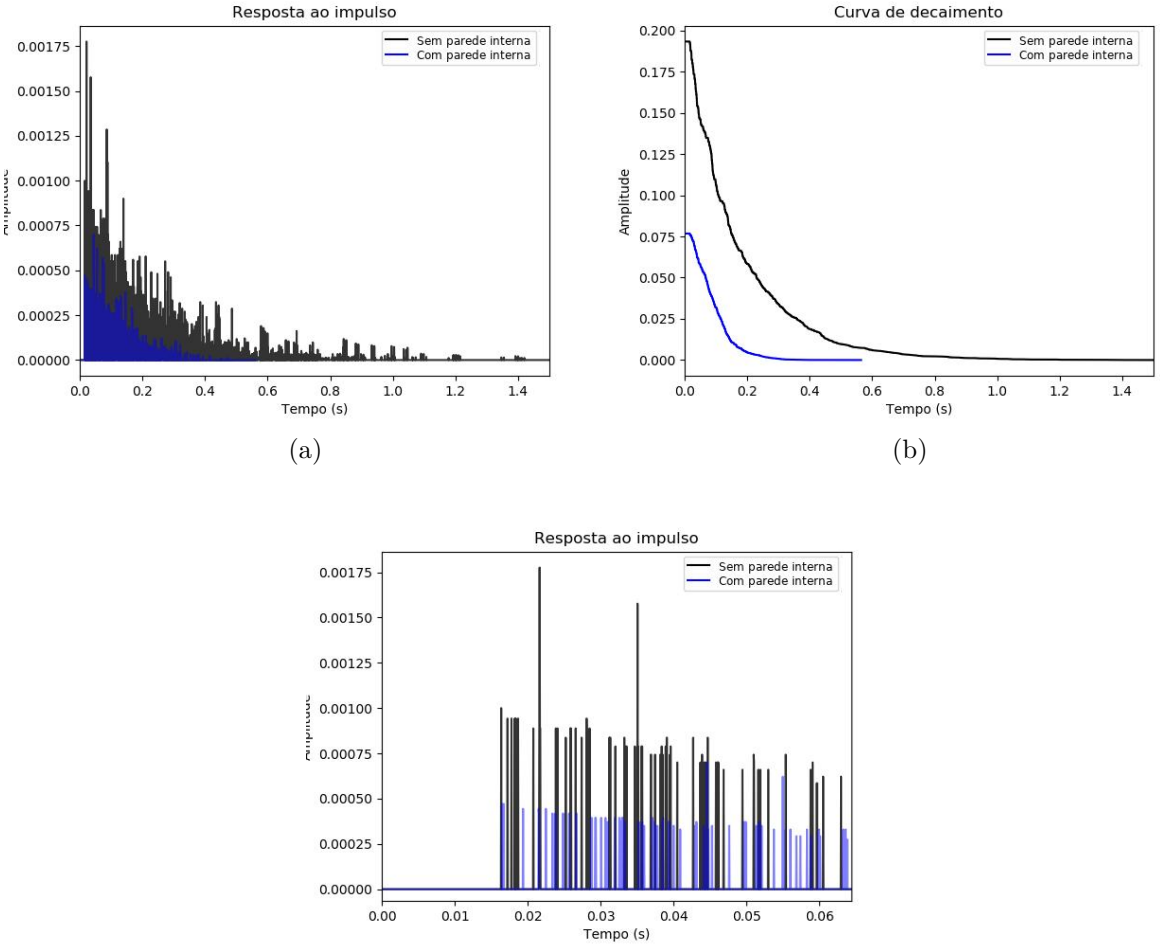
Fonte: Autoria própria

também pela figura 19c que as respostas ao impulso da sala sem a parede interna e com a parede interna com refração, iniciam no mesmo instante de tempo, em torno de $16ms$. Isso é pelo fato de que, diferentemente do caso sem refração, é possível um raio partir do emissor e chegar diretamente no receptor em linha reta, mesmo que parte de sua energia seja perdida na interseção com a parede interna.

4.4 EFEITOS DE FREQUÊNCIA

Para análise dos efeitos causados pela absorção do som dependente da frequência, o algoritmo foi executado no mesmo ambiente utilizado nos primeiros testes (sem parede interna, da figura 11), e considerando que todas as paredes possuem um mesmo filtro retangular, com frequência de corte de $2kHz$, e amplitude de 0,9423 na banda passante e 0 na banda não-passante. No primeiro caso foi utilizado um filtro passa-alta, e no segundo caso um filtro passa-baixa. O valor da frequência de corte de $2kHz$ foi um valor qualquer escolhido apenas de forma a validar o conceito, da mesma forma que a característica

Figura 19 – Resposta ao impulso e curva de decaimento obtidas pelo algoritmo na sala sem a parede interna (em preto) e com parede interna (em azul) - com refração.



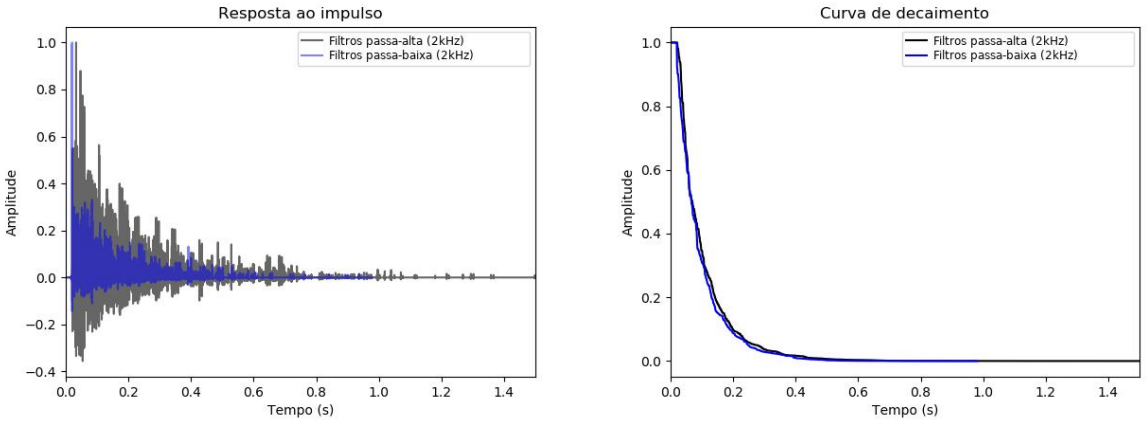
Fonte: Autoria própria

passa alta/baixa dos filtros, e não têm necessariamente relação com valores de literatura. Quaisquer valores e tipos de filtros poderiam ser utilizados.

A resposta ao impulso e a curva de decaimento gerada em cada um dos dois casos é mostrada na figura 20. Em preto a resposta resultante do ambiente com paredes passa-alta, e em azul a resposta com paredes passa-baixa.

A figura 21 mostra as mesmas respostas ao impulso da figura 20a. Entretanto, ela foi ampliada em um intervalo para possibilitar a visualização da característica dos sinais. É possível perceber que a resposta em azul, tem uma característica de variação mais lenta, com período maior, característica de uma resposta passa-baixa. A resposta em preto, uma variação mais rápida e período menor, característica de uma resposta passa-alta.

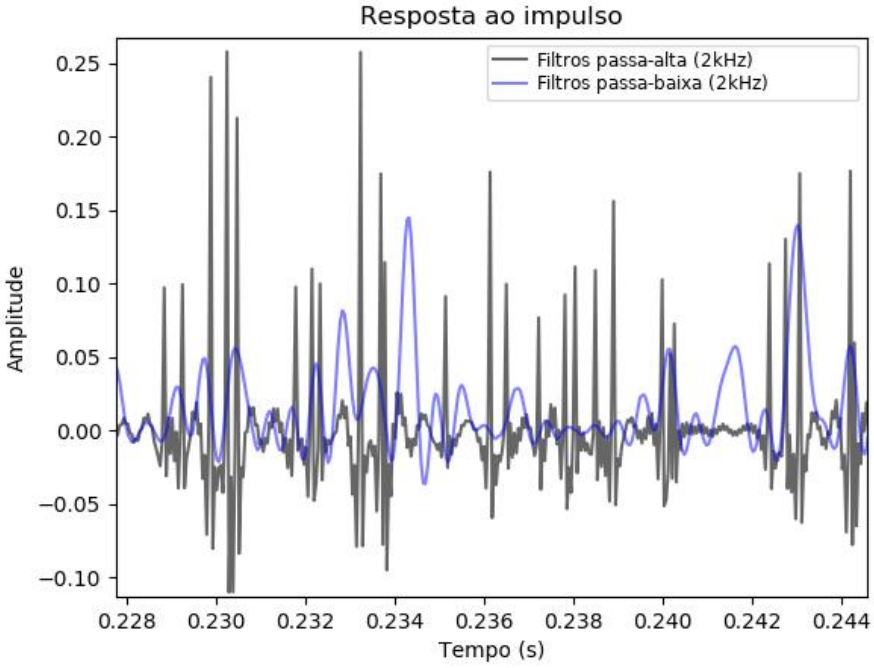
Figura 20 – Resposta ao impulso e curva de decaimento obtidas pelo algoritmo, considerando paredes com filtros passa-alta (em preto) e passa-baixa (em azul).



(a) (b)

Fonte: Autoria própria

Figura 21 – Ampliação da resposta ao impulso obtida pelo algoritmo, considerando paredes com filtros passa-alta (em preto) e passa-baixa (em azul).



(a)

Fonte: Autoria própria

5 CONCLUSÃO

O desenvolvimento do presente trabalho possibilitou uma visão geral sobre os métodos já existentes, a implementação e uma análise sobre características do algoritmo desenvolvido, algumas adaptações, além de uma análise dos resultados obtidos.

Foram encontrados desafios no desenvolvimento deste trabalho. Pode-se considerar que a implementação em geral do algoritmo foi trabalhosa devido a uma difícil depuração e verificação do funcionamento de cada etapa desenvolvida, pois se trabalha com uma grande quantidade de raios, e mesmo a depuração de um único raio não foi trivial, o que foi necessário para que pequenos problemas não se espalhassem conforme a propagação dos raios. Também houve dificuldade envolvendo a adaptação do algoritmo para se basear no domínio da frequência, principalmente no que se refere à obtenção não desejada de respostas não-causais, além do deslocamento acumulativo de fase, problemas que foram resolvidos com a adoção do filtro FIR na etapa final do processo. A adaptação do algoritmo para a execução com o *framework* Spark também foi fonte de desafios. Houve alguns problemas de execução e configuração do Spark junto com a biblioteca NumPy, de forma que resultados de tempo de processamento não estavam sendo coerentes quando executados com o Spark. Isso aconteceu pois o NumPy nativamente usa funções algébricas de uma implementação do BLAS (*Basic Linear Algebra Subprograms*), que por padrão já suporta paralelismo em algumas determinadas funções, o que aparentava estar ocasionando um mal funcionamento dentro do Spark. Para isso, após despender de tempo para descobrir e resolver esse problema, foi encontrada uma solução que limita para um o número máximo a quantidade de threads de cada processo executado, evitando a paralelização automática da biblioteca BLAS, assim evitando problemas com o Spark e resultando em tempos de processamento mais coerentes.

Após a validação dos resultados, percebeu-se através dos tempos de processamento uma diferença significante entre o método com paralelismo e sem paralelismo, sendo que a execução com paralelismo levou aproximadamente metade do tempo do que quando executado sem paralelismo. Esse era o ponto principal a ser validado com este projeto. Além disso, com a comparação da resposta obtida pelo algoritmo desenvolvido com a resposta analítica, obteve-se uma grande semelhança entre as duas respostas e um baixo valor de erro, demonstrando a validade e precisão do algoritmo. Também foi demostrada a validade do trabalho para ambientes não-convexos, além de sua capacidade de lidar no domínio da frequência, quando foram considerados objetos que absorvem a energia de forma quantitativamente diferente para cada banda de frequência.

Algumas evoluções e melhorias podem ser trabalhadas em cima do projeto desenvolvido. Como visto, 98,9% do tempo de processamento é responsável pela etapa do cálculo de interseção entre raios e triângulos, o que pode se tornar ainda maior ao se utilizar um ambiente com mais objetos e consequentemente mais triângulos, sendo esse o gargalo do programa desenvolvido. Com relação a isso existem formas de otimizar esse processo, realizando o cálculo de interseção não para todos os triângulos existentes no ambiente, mas sim para triângulos mais críticos, como os que estão a frente do raio. Com relação ainda ao tempo de processamento, existe a possibilidade de fazer as operações de soma de respostas de cada raio e deslocamento no tempo ainda no domínio da frequência, acumulando os resultados e realizando uma única transformada inversa. No que tange a precisão do algoritmo, uma propriedade que pode ser inserida é o espalhamento sonoro nas reflexões, onde o raio refletido é disperso em várias direções ao invés de uma única,

podendo ser adicionado um pequeno fator aleatório de forma a simular imperfeções nas superfícies. Também podem ser inseridas aproximações pra os efeitos ondulatórios do som, como difração, mais aparente em baixas frequências, além da interferência.

Com isso, acredita-se que o trabalho cumpriu com seus objetivos, apresentando um algoritmo em linguagem de alto nível e com suporte à paralelização, para a caracterização da resposta acústica de um ambiente qualquer, inclusive não-convexos, com atenuações de energia em função da frequência, tendo resultados satisfatórios quando comparados à resposta analítica apresentada, ressaltando o fato de que o programa pode passar por melhorias em trabalhos futuros, por ser escrito em uma linguagem de alto nível e de compreensão relativamente fácil, melhorias essas as quais podem resultar em um resultado ainda mais próximo da realidade e com uma boa performance computacional.

Referências

Apache Software Foundation. **Spark Overview**. 2021. Acesso em: 19 jul. 2021. Disponível em: https://spark.apache.org/docs/latest/. Citado na página 30.

Blender Foundation. **Home of the Blender project - free and open 3D creation software**. 2021. Acesso em: 19 jul. 2021. Disponível em: https://www.blender.org/. Citado na página 29.

CARINI, A.; CECCHI, S.; ROMOLI, L. Robust room impulse response measurement using perfect sequences for legendre nonlinear filters. **IEEE/ACM Transactions on Audio, Speech, and Language Processing**, 11 2016. Citado na página 13.

COMNINOS, P. **Mathematical and Computer Programming Techniques for Computer Graphics**. [S.l.: s.n.], 2006. Citado na página 22.

EATON, J. et al. Estimation of room acoustic parameters: The ace challenge. **IEEE/ACM Transactions on Audio, Speech, and Language Processing**, 10 2016. Citado na página 11.

GOMES, M. H. A.; BERTOLI, S. R.; DEDECA, J. G. Implementação de métodos para a simulação acústica e auralização de salas. **Revista da Sociedade Brasileira de Acústica (SOBRAC)**, 2007. Citado na página 33.

HAVILAND, J. K.; THANEDAR, B. D. Monte carlo applications to acoustical field solutions. p. 1442–1448, 1973. Citado na página 15.

KLEINER, M.; DALENBÄCK, B.-I.; SVENSSON, P. Auralization - an overview. 1993. Citado na página 10.

KULOWSKI, A. Algorithmic representation of the ray tracing technique. **Applied Acoustics**, v. 18, p. 449–469, 1985. Citado na página 22.

Library of Congress. **Wavefront OBJ File Format**. 2020. Acesso em: 19 jul. 2021. Disponível em: https://www.loc.gov/preservation/digital/formats/fdd/fdd000507.shtml. Citado na página 29.

MENEZES, J. V. P. D. Caracterização acústica de salas: Um método híbrido combinando os métodos do traçado de raios e das fontes virtuais. 2018. Citado na página 14.

MOORE, G. An approach to the analysis of sound in auditoria. model design and computer implementation. p. 561–571, 1984. Citado na página 15.

NEUBAUER R.; KOSTEK, B. **Prediction of the Reverberation Time in Rectangular Rooms with Non-Uniformly Distributed Sound Absorption**. 2021. Acesso em: 21 out. 2021. Disponível em: https://sound.eti.pg.gda.pl/papers/prediction_of_reverberation_time.pdf. Citado na página 33.

NOVAK, A.; RUND, F.; HONZÍK, P. Impulse response measurements using mls technique on nonsynchronous devices. **Journal of the Audio Engineering Society**, 2016. Citado na página 13.

OPPENHEIM, A. V.; WILLSKY, A. S. **Sinais e Sistemas**. [S.l.]: Pearson, 2010. Citado na página 18.

RILEY, K. F.; HOBSON, M. P.; BENCE, S. J. **Mathematical Methods for Physics and Engineering**. [S.l.: s.n.], 2006. Citado 2 vezes nas páginas 28 e 29.

SABINE, W. C. Theater acoustic. **Collected Papers on Acoustics**, p. 163–198, 1922. Citado na página 10.

SAVIOJA, L.; SVENSSON, U. Overview of geometrical room acoustic modeling techniques. **The Journal of the Acoustical Society of America**, p. 708–730, 08 2015. Citado 4 vezes nas páginas 10, 13, 14 e 15.

SCHROEDER, M. R. Computer models for concert hall acoustics. p. 461–471, 1973. Citado 2 vezes nas páginas 10 e 13.

SMITH, J. O. **Energy Decay Curve**. 2021. Acesso em: 27 jul. 2021. Disponível em: https://ccrma.stanford.edu/~jos/pasp/Energy_Decay_Curve.html. Citado na página 33.

YAMAUTI, Z. The light flux distribution of a system of interreflecting surfaces. p. 561–571, 1926. Citado na página 15.

ZAWISTOWSKI, T.; SHAH, P. **An Introduction to Sampling Theory**. 2021. Acesso em: 25 ago. 2021. Disponível em: http://www2.egr.uh.edu/~glover/applets/Sampling/Sampling.html. Citado na página 24.

Apêndices

APÊNDICE A – Execução e configurações do programa

A.1 Arquivo de parametrização do ambiente

Estrutura do arquivo no formato `json` que descreve a configuração do ambiente a ser utilizado na execução do algoritmo. Os valores dos parâmetros mostrados são valores de exemplo e podem ser editados conforme a necessidade.

```
{
    "sound_speed": 330,
    "sound_attenuation_freq_coef": [
        [0, 0.0]
    ],
    "object_params": {
        "defaults": {
            "reflection_filter": [
                [400, 0.9],
                [401, 0.0]
            ],
            "refraction_filter": [
                [0, 0.0]
            ],
            "aperture_degrees": 180
        }
    }
}
```

O arquivo contém os parâmetros do ambiente, dos seus objetos, e dos filtros das paredes.

Todos os parâmetros que descrevem filtros são especificados em formato de lista, onde são descritas as frequências e seus respectivos valores, onde os valores intermediários entre as frequências informadas são interpolados linearmente. Se apenas uma frequência é especificada, o mesmo valor é aplicado para todas as frequências.

O campo `object_params` possui os parâmetros dos objetos do ambiente, cada subitem desse grupo pode ser um objeto específico pelo seu nome, ou o subitem `defaults`, que contém parâmetros padrão que são aplicados a todos os objetos que não tiverem seus parâmetros definidos nesse arquivo.

- `sound_speed` especifica a velocidade do som no ambiente em m/s.
- `sound_attenuation_freq_coef` é um parâmetro no mesmo formato do filtro, que especifica o coeficiente de atenuação do som no ar para cada frequência.
- `reflection_filter` especifica o filtro com os coeficientes de reflexão de determinada parede
- `refraction_filter` especifica o filtro com os coeficientes de refração de determinada parede
- `aperture_degrees` especifica o ângulo de abertura em graus de um emissor em específico

A.2 Argumentos de execução do programa

A execução do programa principal para execução do cálculo da resposta ao impulso de uma sala é feita pelo arquivo Python `main_compute_impulse.py`, onde os argumentos são passados da seguinte forma:

```
1    python main_compute_impulse.py [--drawenv] [--
        plotresponse] [--spark] [--numworkers NUMWORKERS] [--
        numrays NUMRAYS] env envparams
```

- `env`: geometria do ambiente (arquivo `.obj`)
- `envparams`: parâmetros do ambiente (arquivo `.json`)
- `-drawenv`: desenha o ambiente
- `-plotresponse`: plota as respostas ao impulso geradas
- `-spark`: executa no modo spark
- `-numworkers`: número de workers utilizados pelo spark
- `-numrays`: número de raios a serem emitidos por cada emissor

APÊNDICE B – Trecho do código do programa em Spark

A seguir é mostrado o trecho principal de execução do código do algoritmo em Spark, com algumas simplificações de sintaxe de forma a facilitar a compreensão. O código completo desenvolvido está disponível em: https://gitlab.com/GiovanniZanetti/tcc2-roomimpulseresponse.

```
1
2  //torna a variavel environment como broadcast
3  env = sc.broadcast(environment)
4
5  //gera os raios dos emissores
6  initial_rays: list = gera_raios_emissores()
7
8  //cria um rdd a partir da lista de raios
9  rdd = sc.parallelize(initial_rays)
10
11 //processa o raio e retorna resposta ao impulso gerada por
       ele (na frequencia)
12 rdd = rdd.map(ray -> process_ray(ray, env))
13 // process_ray() retorna uma tupla (rid, dt, y), onde
14 // "rid" e um numero identificador (id) do receptor, "dt" o
       deslocamento a ser aplicado na resposta, "y" a resposta em
        frequencia gerada pelo raio
15
16 //filtra e exclui respostas com energia menor do que
       min_amplitude
17 rdd = rdd.filter((rid, dt, y) -> (y.mean() >= min_amplitude))
18
19 //faz a fft inversa para obter a resposta ao impulso no tempo
20 rdd = rdd.map((rid, dt, y) -> (rid, dt, inverse_fft(y)))
21
22 //junta respostas de cada receptor
23 rdd = rdd.groupByKey().mapValues(list)
24 //groupByKey() agrupa em forma de lista os valores de "dt" e
       "y", com base no valor de "rid"
25
26 //monta resposta ao impulso de cada receptor
27 rdd = rdd.map((rid, elems) -> (rid, load_from_elements(elems)
       ))
28 // load_from_elements() monta a resposta ao impulso com base
       na lista de elems (dt, y),
29 // deslocando cada resposta com base em "dt" e somando-as
```