

MATEUS DE CAMPOS DA SILVA

**ESTIMAÇÃO DE IMAGENS ACÚSTICAS USANDO
TRANSFORMADAS RÁPIDAS**

São Paulo

2015

MATEUS DE CAMPOS DA SILVA

**ESTIMAÇÃO DE IMAGENS ACÚSTICAS USANDO
TRANSFORMADAS RÁPIDAS**

Trabalho de Conclusão de Curso
apresentado à Escola Politécnica da
Universidade de São Paulo para obtenção
do título de Bacharel em Engenharia
Elétrica – Ênfase em Sistemas Eletrônicos.

Orientador: Prof. Dr. Vitor Heloiz
Nascimento

São Paulo
2015

MATEUS DE CAMPOS DA SILVA

**ESTIMAÇÃO DE IMAGENS ACÚSTICAS USANDO
TRANSFORMADAS RÁPIDAS**

Trabalho de Conclusão de Curso
apresentado à Escola Politécnica da
Universidade de São Paulo para obtenção
do título de Bacharel em Engenharia
Elétrica.

Área de Concentração: Sistemas
Eletrônicos.

Orientador: Prof. Dr. Vitor Heloiz
Nascimento.

São Paulo.

2015.

Aos meus pais, meus amigos e professores que me ajudaram neste trabalho.

AGRADECIMENTOS

Ao Prof. Dr. Vitor Heloiz Nascimento, pela imensa paciência para com a minha pessoa, pelo apoio e por ter acreditado em mim e neste trabalho.

À Universidade de São Paulo e à Escola Politécnica de Engenharia, em especial ao Departamento de Engenharia de Sistemas Eletrônicos e ao Laboratório de Processamento de Sinais – LPS, pelo apoio e disposição.

Aos colegas Felipe Luiz F. O. Chamon, Bruno S. Masiero, Fernando Gonçalves de Almeida Neto, Allan Eduardo Feitosa, Carlos Augusto Bergfeld Mori e Pedro Henrique Itio Hayashi pela ajuda e contribuições ao longo deste trabalho e da vida acadêmica.

A todos que de certa forma contribuíram, direta ou indiretamente, para a conclusão deste trabalho que aqui foram omitidos.

"Existem muitas hipóteses em ciência que estão erradas. Isso é perfeitamente aceitável, elas são a abertura para achar as que estão certas." - Carl Sagan.

RESUMO

O interesse da indústria em reduzir ruídos sonoros não é nenhuma novidade, principalmente quando estes ruídos são fontes prejudiciais para mecanismos e sistemas em áreas importantes, como aqueles causados por vibrações mecânicas ou os que denunciam falhas na estrutura de turbinas, como exemplos. Conseguir detectar a fonte desses ruídos e qual a intensidade de sua energia acústica são de extrema importância na construção de projetos de estruturas e máquinas.

Uma das principais maneiras de se fazer o levantamento dessas intensidades, que estão relacionadas às potências dos sinais dos ruídos sonoros, é através da estimação de imagens acústicas, feita utilizando arranjos de microfones. Nessa abordagem, um arranjo capta os sinais em seus microfones e estes sinais são processados por um algoritmo de estimação, que determina as potências vindas em cada direção. Muitos dos algoritmos mais conhecidos fazem um passo intermediário ao calcular a matriz de correlação dos sinais para realizar a estimação, e nesse passo são feitas hipóteses que acarretam em perda de informações. Ainda, conforme se aumenta a complexidade do arranjo, colocando mais microfones, o tempo que o algoritmo leva para realizar a estimação e sua complexidade aumentam, aumentando o custo computacional do algoritmo. Faz-se interessante um algoritmo que elimine essa perda de informações com baixo custo computacional.

Além disso, como proposto em trabalhos recentes [1] [2] [3], é possível propor uma transformada rápida que acelere os cálculos do algoritmo de estimação, o que diminuiria o tempo de processamento e o número de cálculos necessários para realizar a estimativa, com eficiência igual ou melhor que os trabalhos aqui referenciados.

Palavras-chave: Arranjo de Microfones, Algoritmos de Estimação, Imagens Acústicas, Matrizes de Autocorrelação, Transformada Rápida.

ABSTRACT

The industry's interest in reducing audible noise is nothing new, especially when these noises are harmful sources for mechanisms and systems in important areas such as those caused by mechanical vibrations or denouncing flaws in turbines structure, as examples. To be able to detect the source of these noises and what is the intensity of its acoustic energy are extremely important in projects of building structures and machinery.

One of the main ways to make the estimation of these intensities, which are related to the powers of the signals of audible noise, is through the estimation of acoustic images, which is made using an array of microphones. In this approach, an array captures the signals using its microphones and these signals are processed by an estimation algorithm, which determines the coming powers in each direction. Many of the best known algorithms have an intermediate step to calculate the correlation matrix of signals for performing the estimation, and in this step assumptions are made that lead to loss of information. Furthermore, as it increases the complexity of the array, placing more microphones, the time it takes the algorithm to perform the estimation and the complexity increase, increasing the computational cost of the algorithm. An algorithm that eliminates this loss of information with low computational cost makes it interesting.

In addition, as proposed in recent studies [1] [2] [3], it is possible to propose a fast transform to speed up the calculations of the estimation algorithm, which reduces processing time and number of computations required to perform an estimation in efficiency with equal or better than the works referenced herein.

Keywords: Microphones Array, Estimation Algorithms, Acoustic Images, Autocorrelation Matrices, Fast Transform.

LISTA DE FIGURAS

Figura 1 - Imageamento Acústico usando arranjos de microfones.....	16
Figura 2 - Árvore de Objetivos.....	22
Figura 3 – Janelas de tempo para casos de processos não estacionários	24
Figura 4 - Complexidade e Resolução dos algoritmos conhecidos na literatura	28
Figura 5 - Diagrama de Blocos do Projeto - Nível 0	29
Figura 6 - Diagrama de Blocos do Projeto - Nível 1	30
Figura 7 - Diagrama de Blocos do Projeto - Nível 2 - Maior Detalhamento e com as mudanças propostas já implementadas	31
Figura 8 - Bloco 1: Recepção dos sinais do arranjo de microfones.....	32
Figura 9 - Bloco 2: Cálculo da FFT.....	33
Figura 10 - Bloco 3: Algoritmo de Estimativa.....	34
Figura 11 - Espaço visto pelo arranjo de microfones, amostrado.....	37
Figura 12 - Exemplo de Arranjo de Microfones Separável e Não Uniforme	45
Figura 13 - Modelo de arranjo com as posições dos microfones com índices devidamente organizadas	46
Figura 14 - Sinais das três fontes consideradas no cenário inicial.....	50
Figura 15 - Visada do Arranjo de Microfones e Arranjo de Microfones utilizados no cenário inicial.....	51
Figura 16 - Sinais em cada um dos microfones, no tempo, para um exemplo simples de quatro microfones, sob as mesmas circunstâncias do cenário inicial analisado ..	52
Figura 17 - Imagem Acústica estimada pelo DAS para o cenário inicial	53
Figura 18 - Imagem acústica 3D para o DAS no cenário inicial	54
Figura 19 - Tempo de Processamento para cada realização para o DAS, sem e com da nova KAT, no cenário inicial	54
Figura 20 - Posição dos picos presentes na imagem acústica obtida para a regularização l1 no cenário inicial, para os dois casos, com e sem o uso da nova KAT	55

Figura 21 - Tempo de Processamento para cada realização para a regularização <i>l1</i> , sem e com uso da nova KAT, no cenário inicial.....	56
Figura 22 - Imagem Acústica estimada pela Regularização <i>TV</i> para o cenário inicial, para os dois casos, com e sem o uso da nova KAT.....	57
Figura 23 - Tempo de Processamento para cada realização para a regularização <i>TV</i> , sem e com o uso da nova KAT, no cenário inicial.....	57
Figura 24 - Tempo de Processamento para o Delay and Sum, variando-se o número de microfones no arranjo.....	59
Figura 25 - Tempo de Processamento para a Regularização <i>l1</i> , variando-se o número de microfones no arranjo.....	61
Figura 26 - Tempo de Processamento para a Regularização <i>TV</i> , variando-se o número de microfones no arranjo.....	63
Figura 27 - Fontes Correlacionadas e Descorrelacionadas, com a nova e antiga KAT, utilizando o método Delay and Sum.....	65
Figura 28 - Fontes Correlacionadas e Descorrelacionadas, com a nova e antiga KAT, utilizando o método <i>l1</i>	67
Figura 29 - Fontes Correlacionadas e Descorrelacionadas, com a nova e antiga KAT, utilizando o método <i>TV</i>	68
Figura 30 - Diagrama de Gantt do Projeto	73
Figura 31 - Etapas e Atividades do Projeto	74
Figura 32 - Custos de Projeto (em horas/homem trabalhadas).....	74

LISTA DE TABELAS

Tabela 1 - Média do Tempo de Reconstrução para cada método presente em [1]...	23
Tabela 2 - Relação das Necessidades da Árvore de Objetivos e os Requisitos do Projeto	25
Tabela 3 - Tabela de Conceitos	27
Tabela 4 - Tempo de Processamento para cada configuração do arranjo, para o método DAS	60
Tabela 5 - Tempo de Processamento para cada configuração do arranjo, para o método Regularização l1	62
Tabela 6 - Tempo de Processamento para cada configuração do arranjo, para o método Regularização TV	64

LISTA DE ABREVIATURA E SIGLAS

FPGA	Field Programmable Gate Array
KAT	Kronecker Array Transform
NNFFT	Non-equispaced in time and frequency fast Fourier transform
NFFT	Non-equispaced fast Fourier transform
DAS	Delay and Sum
DAMAS2	Deconvolution Approach for the Mapping of Acoustic Sources
TV	Total Variation

Sumário

1. Definição do Problema	16
2. Objetivos	17
3. Pesquisa de Levantamento da Situação	18
3.1. Visão Geral	19
3.2. Resumos de Conceitos Relevantes	20
3.2.1. <i>Non-equispaced in time and frequency fast Fourier transform (NNFFT), Non-equispaced fast Fourier transform (NFFT) e Kronecker array transform (KAT)</i>	20
3.2.2. Minimização por Regularização	20
3.2.3. FPGA – <i>Field-programmable gate array</i>	21
4. Árvore de Objetivos	22
5. Especificação de Requisitos	23
5.1. Restrições para o Sistema	26
6. Projeto	27
6.1. Decomposição Funcional: Nível 0	29
6.2. Decomposição Funcional: Nível 1	30
6.3. Decomposição Funcional: Nível 2	31
6.4. Análise de Riscos do Projeto	32
6.4.1. Bloco 1: Recepção dos sinais do arranjo de microfones	32
6.4.2. Bloco 2: Cálculo da FFT	33
6.4.3. Bloco 3: Algoritmo de estimativa	33
6.5. Prova de Conceito	34
6.6. Desenvolvimento do Projeto	35
6.6.1. Modelo de Propagação de Ondas	35
6.6.2. Métodos de Estimação de Imagens Acústicas	40

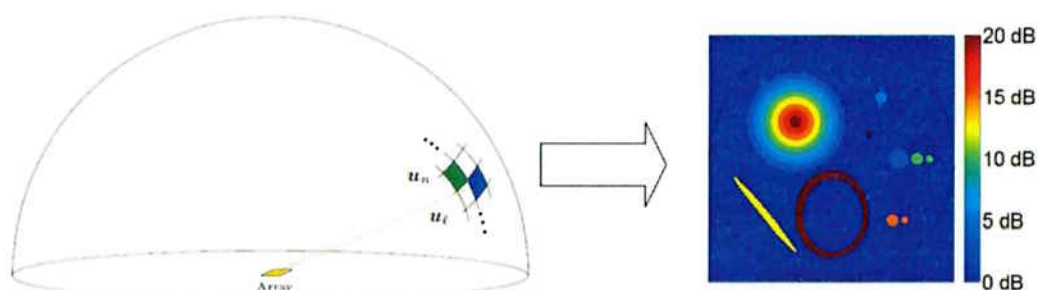
6.6.2.1. <i>Delay and Sum</i> : DAS.....	40
6.6.2.2. <i>Deconvolution Approach for the Mapping of Acoustic Sources</i> : DAMAS2 41	
6.6.2.3. Regularização <i>l1</i>	42
6.6.2.4. Regularização TV (<i>Total Variation</i>)	43
6.6.3. Transformadas Rápidas: KAT (<i>Kronocker Array Transform</i>) e Nova KAT proposta	44
6.7. Implementação dos Algoritmos	49
6.7.1. Geração do Modelo e Cenário Inicial Adotado	50
6.7.2. Resultados para o método <i>Delay and Sum</i>	53
6.7.3. Resultados para o método Regularização <i>l1</i>	55
6.7.4. Resultados para o método Regularização TV	56
6.8. Tempos de Processamento Experimentais dos Métodos com a Variação do Número de Microfones no Arranjo	59
6.8.1. Tempo de Processamento para o DAS	59
6.8.2. Tempo de Processamento para a Regularização <i>l1</i>	61
6.8.3. Tempo de Processamento para a Regularização TV	63
6.9. Uso de Fontes Correlacionadas e Não Correlacionadas	65
7. Verificação de Requisitos, Considerações Finais e Conclusões	69
7.1. Gerenciamento de Projeto	73
7.1.1. Cronograma do Projeto – Diagrama de Gantt	73
7.1.2. Etapas e Atividades do Projeto (EAP)	74
7.1.3. Custos de Projeto	74
8. Bibliografia	75
ANEXO A – Função "arranjo.m", que auxilia na modelagem do arranjo planar	77
ANEXO B – Função "steer.m", que calcula a matriz de direções <i>V</i>	78
ANEXO C – Código para o cenário inicial para o <i>Delay and Sum</i>	79

ANEXO D – Função “mapa.m”, que auxilia na geração da imagem acústica	82
ANEXO E - Código para o cenário inicial para a Regularização <i>I1</i> sem uso da nova KAT	83
ANEXO F - Código para o cenário inicial para a Regularização <i>I1</i> com uso da nova KAT	86
ANEXO G - Código para o cenário inicial para a Regularização <i>TV</i> sem uso da nova KAT	88
ANEXO H - Código para o cenário inicial para a Regularização <i>TV</i> com uso da nova KAT	91
ANEXO I – Função “FFT.m” que auxilia no cálculo da FFT dos sinais nos microfones	93
ANEXO J – Função “Mode1.m” e “Mode2.m” presentes no código da regularização <i>I1</i> e <i>TV</i>	94
Mode1.m	94
Mode2.m	94
ANEXO K – Função “AnonimaKAT.m”, função do tipo anônima que auxilia no processo de minimização, tanto na regularização <i>I1</i> como na <i>TV</i>	95
ANEXO L – Funções “plot_I1.m” e “plot_TV.m”	96
plot_I1.m	96
plot_TV.m	96

1. Definição do Problema

O imageamento acústico consiste em mapear as direções e intensidades dos sinais vindos de fontes de som espalhadas no espaço utilizando arranjos de microfones. O tipo de processamento que é realizado com esses arranjos é útil em diversas áreas da engenharia e da ciência, como na construção e análise de sinais de radares, na radioastronomia, em sonares, em comunicações, na detecção na direção de chegada dos sinais dessas fontes, em sismologia e no tratamento e diagnóstico médico.

Figura 1 - Imageamento Acústico usando arranjos de microfones



Fonte: "Acoustic Imaging Using the Kronecker Array," pp. 6, 10 Fevereiro 2015.

O imageamento acústico é particularmente usado para se projetar carros, trens e aviões que sejam mais eficientes aerodinamicamente e apresentem menos ruído, além de ser usados para o projeto de turbinas e salas de concerto com objetivo semelhante. Tal análise pode ainda ser utilizada para se detectar fontes de ruídos, que é feita em projetos nas etapas de prototipagem e evitam falhas no funcionamento dos produtos finais de tais projetos.

O nome de imageamento não é em vão. Assim como quando se tira uma foto com uma câmera fotográfica de uma imagem qualquer, o arranjo de microfones fará o mesmo com o som, mostrando onde são os pontos com maior intensidade de potência, utilizando para isso ou um mapa de cores ou uma escala de cinza, por exemplo, para facilitar a visualização. Nota-se, porém, que numa foto normal a cor está relacionada com a frequência do sinal, enquanto no imageamento acústico o mapa de cores é um artifício para auxiliar na compreensão dos resultados obtidos, uma vez que aqui neste problema as cores estão relacionadas às intensidades de potência num dado ponto.

Existem diversos métodos e algoritmos que realizam a estimação do valor das potências desses sinais que são bem conhecidos e amplamente utilizados para os propósitos anteriormente citados, porém se faz interessante que tais medições sejam realizadas com custo computacional menor, de maneira mais rápida e, quando necessário, em situações de tempo real. Assim sendo, um algoritmo que realize o mesmo processamento com maior eficiência é de grande interesse.

Dentro do universo do processamento de sinais usando arranjos, tal algoritmo possibilitaria que, com a sua aceleração, algoritmos de reconstrução mais avançados da imagem acústica sejam utilizados, como métodos que usem problemas de minimização com regularização, possibilitando a solução de problemas onde o número de microfones no arranjo seja maior, coisa que demoraria muito mais tempo usando os algoritmos conhecidos que são utilizados hoje em dia.

Para este projeto, todas as figuras utilizadas foram de autoria do próprio autor do projeto, salvo quando indicado.

2. Objetivos

Os objetivos desse trabalho de formatura consistem em adaptar algoritmos para aceleração do cálculo de imagens acústicas para métodos que não requeiram a estimação de matrizes de autocorrelação. Esses métodos têm diversas vantagens em face de métodos tradicionais que dependam de matrizes de autocorrelação, em particular menor tempo de cálculo, necessidade de usar um conjunto menor de dados para se obter uma imagem, e robustez frente às correlações entre os pixels da imagem estimada. Este processamento poderá ser embarcado fazendo uso de uma FPGA, para que seja possível se fazer tais testes de desempenho, além de ver como o algoritmo se comporta em tempo real. Far-se-á o uso da FPGA, pois outros dispositivos de processamento de dados, como um DSP ou um embarcado genérico, não seriam capazes de processar o volume de dados necessário para este problema de estimação, além do fato de que a FPGA possui certas facilidades na sua programação, como a paralelização de processos, coisa que não ocorre num DSP, que realiza seus processos de maneira seriada.

Portanto, os objetivos, em uma sequência lógica de realização, são:

- Estudar e utilizar os métodos mais conhecidos para a solução do problema de imageamento acústico e se familiarizar com tais algoritmos;
- Implementar o método para calcular a imagem acústica a partir do algoritmo proposto em [2];
- Comparar os resultados do método anterior com e sem uso de transformadas rápidas, também propostas em [2], para acelerar os cálculos de estimativa e verificar a melhoria de desempenho no tempo de processamento;
- Procurar uma melhoria no algoritmo presente em [2], evitando o uso de matrizes de autocorrelação e tentar achar uma transformada nos moldes daquela proposta também em [2], para acelerar os cálculos de estimação.
- Comparar o método presente em [2], que faz uso de matrizes de autocorrelação e de uma transformada rápida para aceleração dos cálculos de estimação, com o método proposto neste trabalho, que não faz uso de matrizes de autocorrelação e faz uso de uma nova transformada rápida com os mesmos objetivos da anterior;
- Validar a melhoria de desempenho obtida com o modelo proposto neste trabalho frente ao modelo anteriormente proposto em [2];

Como objetivos secundários, seriam interessantes os seguintes:

- Estudar a influência dos parâmetros presentes nos métodos (variáveis de regularização da minimização, janela utilizada quando se utiliza a Transformada de Fourier, etc.);
- Pensar em como utilizar a solução desse problema em um instante como condição inicial para o seguinte, para problemas em tempo real;
- Implementar o algoritmo de [1] ou o algoritmo com a melhoria numa FPGA, caso haja tempo para tanto.

3. Pesquisa de Levantamento da Situação

Nesta seção, serão discutidos alguns conceitos importantes para o entendimento do que está sendo desenvolvido neste projeto, além de melhor se explicar o problema que se pretende ser resolvido aqui neste trabalho.

3.1. Visão Geral

A literatura sobre o problema de imageamento acústico é bem conhecida e os métodos propostos para a solução do problema são os mais diversos possíveis, presentes numa série de artigos e livros acadêmicos.

Inicialmente, o imageamento acústico consiste numa série de métodos e algoritmos que permitam se estimar as intensidades de som num ponto qualquer do espaço chegando de diferentes direções. Utilizando tais métodos, pode-se verificar a presença de fontes de som e suas direções em relação ao arranjo de microfones.

O nível de som chegando num dado ponto de várias direções pode ser estimado através do uso de filtros espaciais, que estão intimamente ligados a geometria do arranjo utilizado e as direções das fontes, ou através da solução de um problema de otimização que englobe tais informações. Outro problema presente nestes métodos está em querer melhorar a resolução do arranjo de microfones sem alterar o número de sensores no arranjo, sendo que sua geometria influencia na qualidade da imagem acústica. Todos eles, nas soluções propostas, porém, fazem com que o custo computacional aumente conforme se tem soluções iterativas vindas de um problema de otimização que necessita do cálculo de produtos de vetores ou matrizes muito grandes. Também, deve-se lembrar que o posicionamento dos microfones possui erros, que alteram os resultados obtidos para a imagem acústica.

Além disso, esses métodos que foram utilizados até agora, em seus algoritmos de funcionamento, utilizam cálculos de matrizes de autocorrelação como passo intermediário para calcular a imagem. A KAT (*Kronecker Array Transform*), proposta em [2], é utilizada para acelerar os cálculos nesse caso. Mas o uso desses métodos com matrizes de autocorrelação envolve algumas hipóteses, que não são verdadeiras em geral, em particular, que as fontes de som são descorrelacionadas. A intenção, portanto, é tentar estender a ideia da KAT para métodos que não precisem usar a matriz de autocorrelação. Além de evitar o problema de aproximação, isso poderia resultar em algoritmos mais rápidos e com melhor qualidade de imagem, e que precisem de menos dados para chegar a um mesmo resultado. Isso poderia possibilitar a obtenção de imagens de fenômenos que variam rapidamente, por exemplo.

A intenção, portanto, desse trabalho, será a de utilizar técnicas propostas em [2], aplicadas especificamente para esse problema de imageamento acústico.

3.2. Resumos de Conceitos Relevantes

Serão discutidos agora alguns conceitos preliminares necessários para a melhor compreensão deste trabalho.

3.2.1. *Non-equispaced in time and frequency fast Fourier transform (NNFFT), Non-equispaced fast Fourier transform (NFFT) e Kronecker array transform (KAT)*

São transformadas que aceleram o cálculo de produtos de matrizes/vetores, explicitadas em [2]. A NNFFT é o mais geral, porém o mais lento método de aceleração, sendo que pode ser empregada para qualquer geometria de arranjo de microfones que se desejar, por exemplo.

A NFFT é mais rápida, mas restrita a casos onde as direções que são examinadas estão amostradas uniformemente, isto é, as direções analisadas são divididas espacialmente num determinado número de amostras.

Finalmente é proposta uma nova transformada, que, apesar de ser utilizada para casos bem específicos de geometria do arranjo de microfones e de organização de direções examinadas, como, por exemplo, num caso em que o arranjo e a organização tenham geometria separável, isto é, de grade retangular. Essa transformada, também presente e desenvolvida em [2] é a KAT. Ao contrário dos outros métodos, esta transformada trabalha bem estimando a imagem acústica quando as fontes de som estão mais próximas do arranjo, eliminando a restrição de fontes em campos distante, além de cumprir a diminuição do custo computacional.

3.2.2. Minimização por Regularização

É um método de resolução de problemas de minimização com restrições. Regularização, em matemática e estatística e, em particular nas áreas de aprendizado de máquina e problemas inversos, refere-se a um processo de introdução de informações adicionais, a fim de resolver um problema mal colocado ou para evitar superajuste. Esta informação é geralmente da forma de uma penalidade para a complexidade, tais como restrições sobre a norma vetorial.

3.2.3. FPGA – *Field-programmable gate array*

Uma FPGA (ou em português “Arranjo de Portas Programável em Campo”) é um circuito integrado projetado para ser programável por um projetista, sendo que isso é possível após sua fabricação. Diferente da grande maioria dos chips que se encontra no cotidiano, como em celulares, TVs, etc., que já vem pré-programados, uma FPGA é completamente reprogramável, sendo que suas funcionalidades são totalmente configuradas pelos usuários, e não pelo fabricante. É um dispositivo semicondutor altamente utilizado no processamento de dados digitais e, fisicamente, é praticamente composta por três componentes:

- Blocos de Entrada e Saída (I/O)
- Blocos Lógicos Configuráveis
- Chaves de Interconexão

4. Árvore de Objetivos

Atraves da metodologia de projeto adequada, foi gerada a seguinte arvore de objetivos, que ilustra de maneira visual o que se pretende obter no fim deste projeto de formatura.

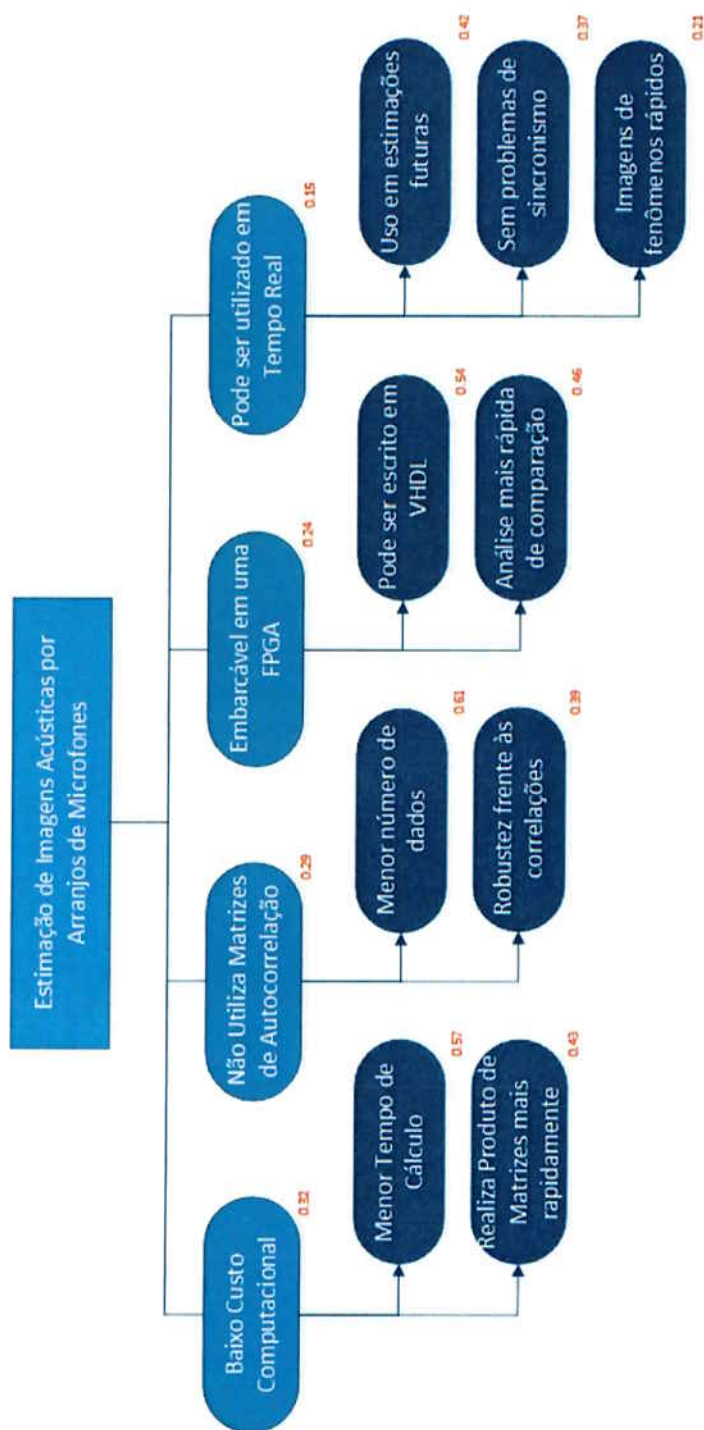


Figura 2 - Árvore de Objetivos

5. Especificação de Requisitos

Os principais requisitos de projeto deste trabalho são os seguintes:

- O estimador deve ser capaz de realizar a estimativa da imagem acústica utilizando para isso um algoritmo com a menor complexidade possível.
- O tempo de processamento do algoritmo utilizado pelo estimador deve ser no mínimo menor ou igual aos tempos de processamento já obtidos em outros trabalhos aqui referenciados, que utilizam matrizes de autocorrelação em seus algoritmos [1].
- A resolução das imagens obtidas pelo algoritmo deve ser igual ou melhor que aquelas obtidas em trabalhos aqui referenciados [1].
- A utilização de memória pelo algoritmo deve ser a menor possível para auxiliar a melhoria do tempo de processamento do mesmo, além de possibilitar a utilização de um volume menor de dados para solucionar o problema.
- O algoritmo deve funcionar tão bem quanto outros trabalhos aqui referenciados [1] quando os processos analisados forem estacionários, mas deve servir também para processos que sejam não estacionários (quando as fontes de som são móveis, por exemplo).

Os resultados apresentados em [1] apontam alguns limites, porém não todos, que este trabalho deverá alcançar respeitar os requisitos de projeto acima. Lá foram utilizados vários algoritmos de reconstrução de imagens acústicas, como o "Delay and Sum" (DAS), descrito em [2] e [4], e outros apresentados em [2], todos utilizando a transformada de arranjos de Kronecker (KAT), proposto em [2], para acelerar os algoritmos. Os tempos obtidos pelos diversos algoritmos que utilizavam a KAT estão na tabela a seguir:

Tabela 1 - Média do Tempo de Reconstrução para cada método presente em [1]

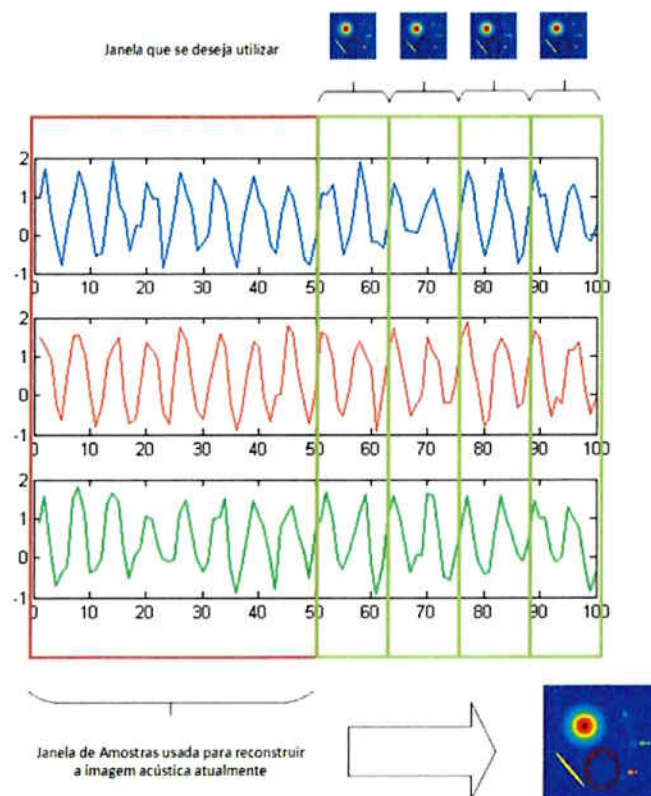
Algoritmos	Tempo [s]
DAS	0.13
DAMAS2	1.58
Regularização l_1	1.96
Regularização TV	2.46

Fonte: C. B. Arroyo, "Estimação de Imagens Acústicas com Arranjos de Microfones," São Paulo, 2015.

Outra análise pertinente presente em [2] que deve ser realizada neste trabalho é ver como é o comportamento do novo algoritmo aplicado diretamente, sem nenhuma ajuda de uma transformada que acelere os cálculos de estimação, e comparado ao comportamento do mesmo algoritmo, fazendo uso de uma transformada rápida que acelere seus cálculos. Neste trabalho, deve-se que chegar num resultado igual ou melhor que o apresentado em [1], [2] e [5] quando se elimina a matriz de autocorrelação presente nos algoritmos anteriores.

Finalmente, quando se diz que se quer que este novo algoritmo funcione para processos que não sejam estacionários, quer-se que o tempo necessário para que o algoritmo analise o sinal quando ele é aproximadamente estacionário seja o menor possível.

Figura 3 – Janelas de tempo para casos de processos não estacionários



Atualmente, os algoritmos conhecidos requerem que os processos analisados permaneçam aproximadamente estacionários (no sentido de média e correlações aproximadamente constantes) num intervalo de tempo relativamente longo, que permite o cálculo de transformadas e matrizes de autocorrelação. Porém este intervalo de tempo pode ser menor ou, em termos discretos, pode-se utilizar uma janela de

menos amostras a fim de se identificar esses trechos do processo. Quanto menos amostras, menor a janela utilizada ou menor o tempo desse intervalo, melhor será o algoritmo.

Relacionando estes requisitos com as necessidades apontadas na árvore de objetivos do projeto e considerando que:

- 1 – Baixo Custo Computacional
- 2 – Não utilização de Matrizes de Autocorrelação
- 3 – Pode ser embarcável em uma FPGA
- 4 – Pode ser utilizado em tempo real

tem-se a seguinte tabela:

Tabela 2 - Relação das Necessidades da Árvore de Objetivos e os Requisitos do Projeto

Necessidades da Árvore de Objetivos	Requisitos de Projeto	Justificativas
1 e 2	O estimador deve ser capaz de realizar a estimativa da imagem acústica utilizando para isso um algoritmo com a menor complexidade possível.	Baseado na comparação com algoritmos já existentes. O algoritmo que será proposto neste trabalho deve apresentar um grau de complexidade igual ou menor que os existentes na literatura, senão não se torna viável sua utilização.
1 e 2	O tempo de processamento do algoritmo utilizado pelo estimador deve ser no mínimo menor ou igual aos tempos de processamento já obtidos em outros trabalhos.	Calcular produtos de matrizes de maneira mais rápida, com o uso de transformadas adequadas, e utilizar um algoritmo que não necessite de aproximações criadas por se utilizar autocorrelação tornam o tempo de processamento e a complexidade menores.
1 e 2	A resolução das imagens obtidas pelo algoritmo deve ser igual ou melhor que aquelas obtidas em trabalhos.	Espera-se que as melhorias que serão obtidas aqui, além de diminuir a complexidade do algoritmo e o tempo de processamento, venham a ter como consequência a melhoria da resolução da imagem acústica obtida.

1,2 e 3	A utilização de memória pelo algoritmo deve ser a menor possível.	O menor uso de memória, relacionado a não utilização de matrizes de correlação, torna o algoritmo atraente, pois vai garantir que o custo computacional e o embarque sejam mais fáceis.
1-4	O algoritmo pode vir a funcionar para processos estacionários, mas deve servir também para processos que sejam não estacionários.	Se todas as melhorias forem alcançadas, espera-se que o algoritmo possa ser capaz de estimar imagens acústicas de fontes cujos processos são não estacionários com taxas de não estacionariedade maiores.

5.1. Restrições para o Sistema

As restrições técnicas mais importantes para este trabalho estão em querer um algoritmo que não utilize matrizes de correlação e suas aproximações decorrentes, pois isso vai gerar um algoritmo mais simples e rápido que os que existem hoje em dia. O embarque em uma FPGA é apenas uma maneira de tornar a análise e processamento dos dados mais fácil, pois a FPGA tem funcionalidade e poder de processamento que garantem isso. Porém a sua utilização pode ser feita apenas em longo prazo, pois serão as simulações que garantirão se o algoritmo alcançou os requisitos de projeto.

Ainda, o método de análise do desempenho do algoritmo proposto deve ser bem preciso. Assim como feito em [1], o número de cálculos realizados no algoritmo e o tempo que este leva para terminar sua operação devem ser e serão cuidadosamente analisados, pois a viabilidade do algoritmo depende disso.

6. Projeto

A tabela a seguir mostra algumas ferramentas que podem ser usadas para se chegar aos resultados pretendidos neste trabalho. Para se diminuir a complexidade do algoritmo, será usado o grande banco de funções e a facilidade de programação do MATLAB® Script. Tentar fazer o algoritmo em linguagem C ou outra do tipo não é plausível quando se tem em mãos uma ferramenta como o ambiente de simulação do MATLAB.

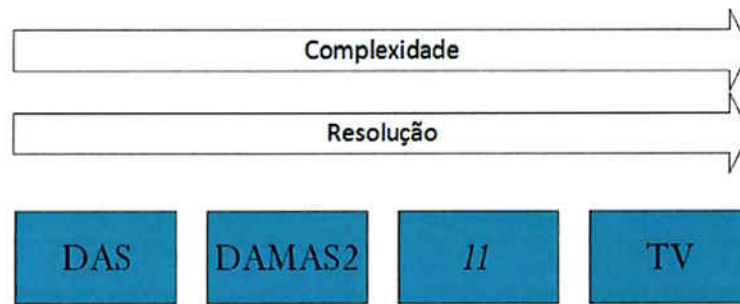
Tabela 3 - Tabela de Conceitos

Menor Complexidade	Menor Tempo de Processamento	Maior/Melhor resolução	Menor Uso de Memória e Dados
Uso do MATLAB® Script	NNFFT	DAS	DSP
Linguagem C	NFFT	DAMAS2	FPGA
Python	FFT	Regularização l_1	
	KAT	Regularização TV (Total Variation)	
Outra transformada			

Além disso, serão implementadas inicialmente todas as transformadas propostas para diminuir o tempo de processamento dos algoritmos e se optará por aquela que o fizer com a menor complexidade possível, reduzindo também o número de cálculos necessários para o algoritmo funcionar. Propor uma nova transformada que funcione como a KAT para os casos de algoritmos que não utilizam autocorrelação na estimação das imagens acústicas é extremamente interessante.

A sequência de métodos de reconstrução apresentada – DAS, DAMAS2, l_1 e TV, presentes em [2] – está em ordem de melhor resolução. Todas serão analisadas, pois sua complexidade também aumenta na ordem apresentada, e isso afeta o tempo de processamento, como anteriormente citado.

Figura 4 - Complexidade e Resolução dos algoritmos conhecidos na literatura



O melhor conjunto que obtiver uma resolução e um tempo de processamento igual ou melhor que os apresentados anteriormente será escolhido para o desenvolvimento deste trabalho. Este estudo será feito como parte preliminar deste trabalho, neste início do projeto.

Por fim, como já dito, fez-se a escolha pela FPGA por sua capacidade de processamento e paralelização de processos, que a diferem de um DSP. Como o problema que este trabalho trata possui um volume de dados considerável, uma FPGA se faz mais interessante.

Portanto, pode-se concluir que se tem como proposta para este projeto criar um estimador de imagens acústicas com as seguintes características:

- Elimine a necessidade de uma matriz de autocorrelação, devido à perda de informação causada pelas hipóteses feitas para facilitar cálculos intermediários (fontes não correlacionadas, portanto correlações cruzadas nulas).
- Seja tão eficiente quanto os algoritmos existentes (Baixo custo computacional, realiza cálculos mais rapidamente, etc).

6.1. Decomposição Funcional: Nível 0

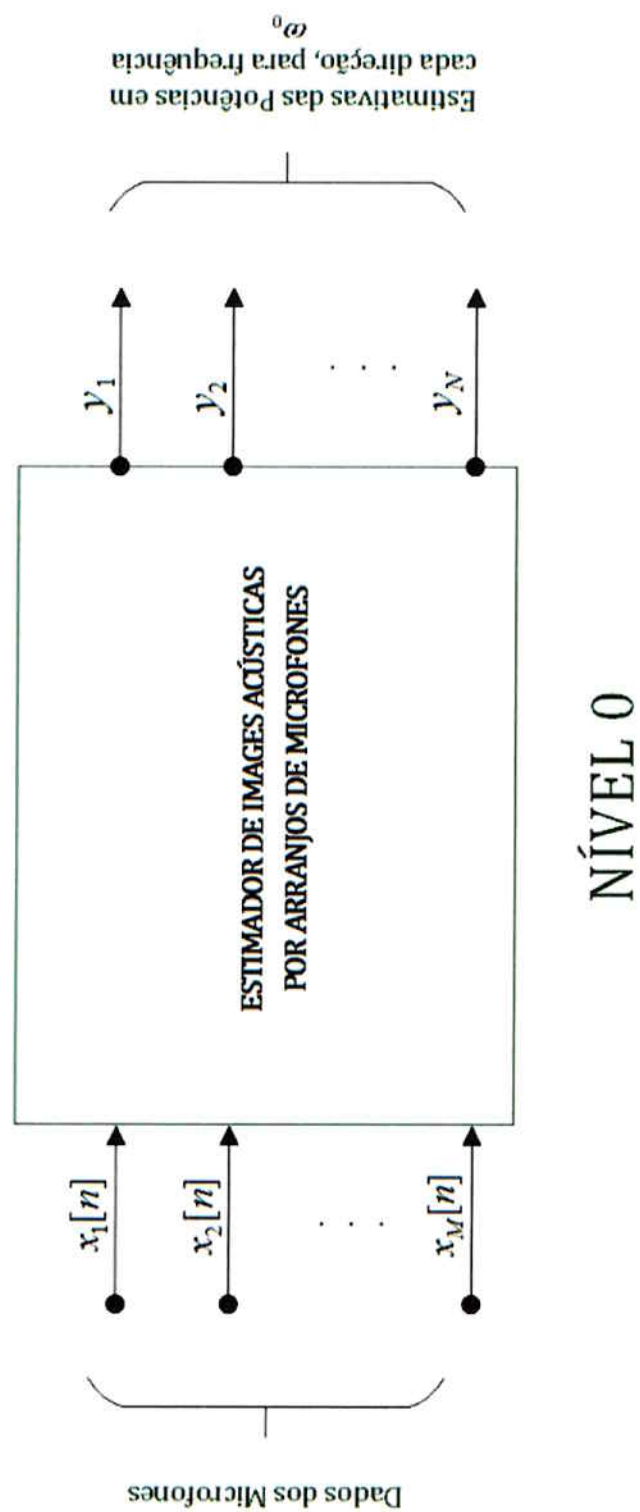


Figura 5 - Diagrama de Blocos do Projeto - Nível 0

6.2. Decomposição Funcional: Nível 1

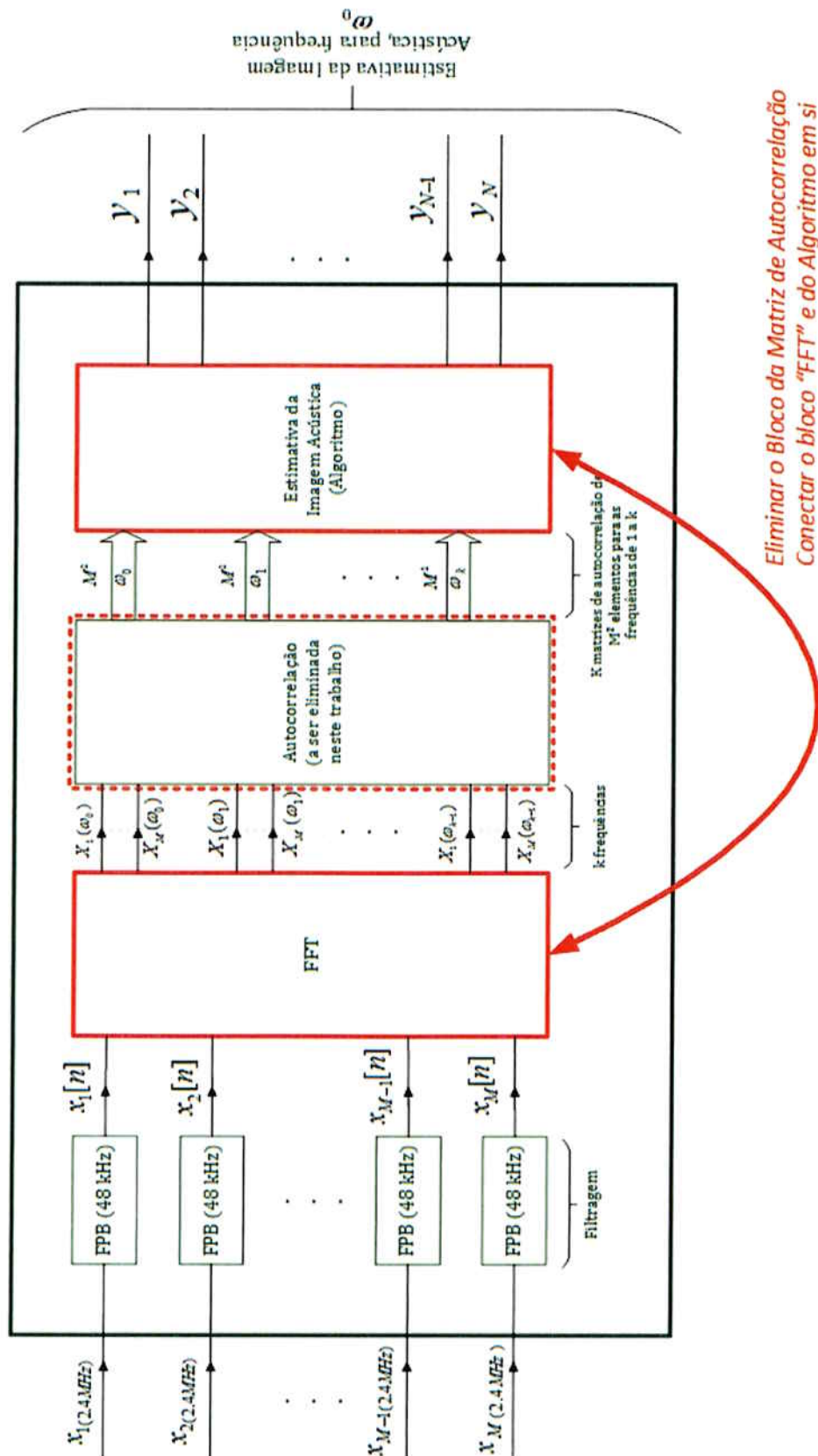


Figura 6 - Diagrama de Blocos do Projeto - Nível 1

6.3. Decomposição Funcional: Nível 2

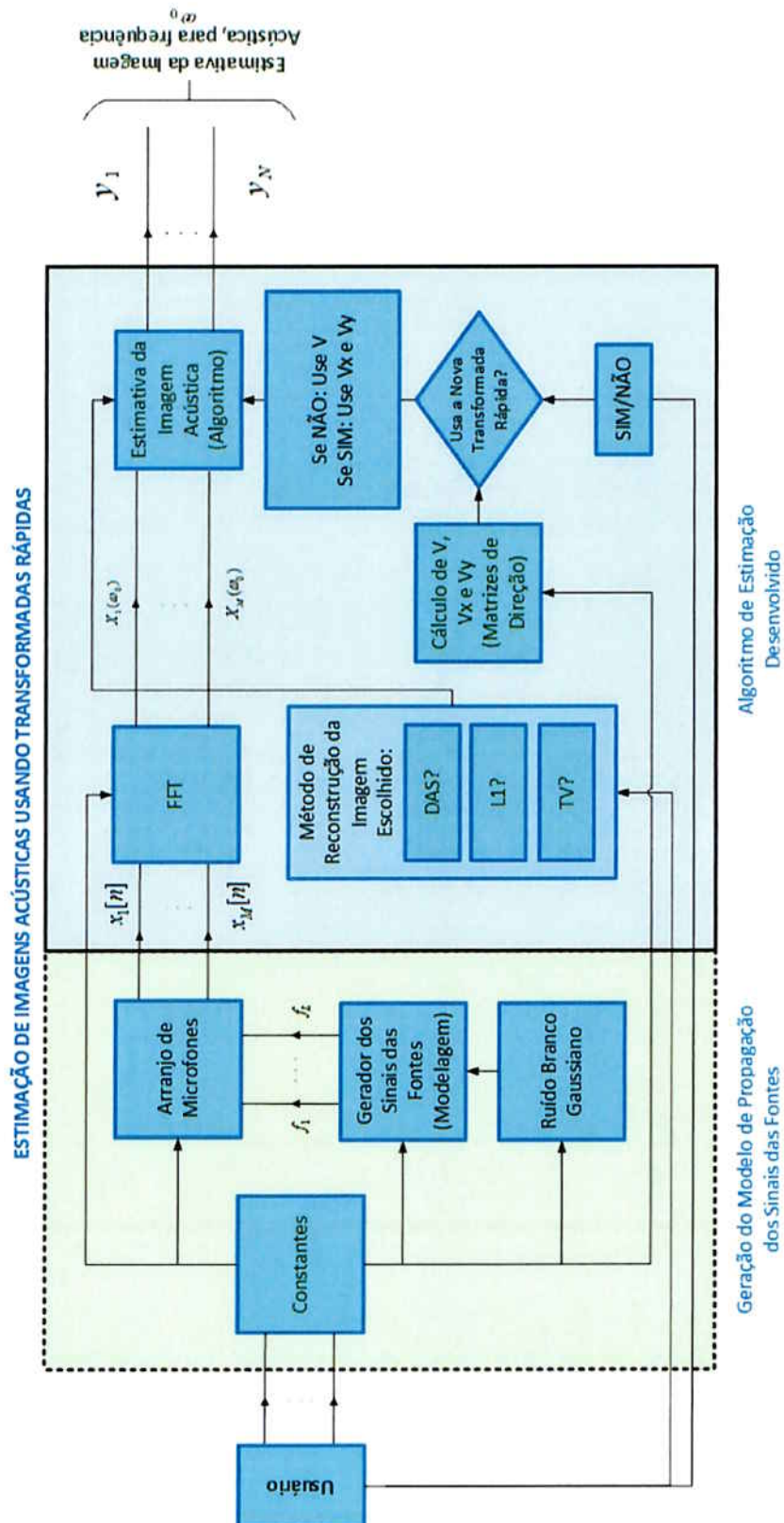


Figura 7 - Diagrama de Blocos do Projeto - Nível 2 - Maior Detalhamento e com as mudanças propostas já implementadas

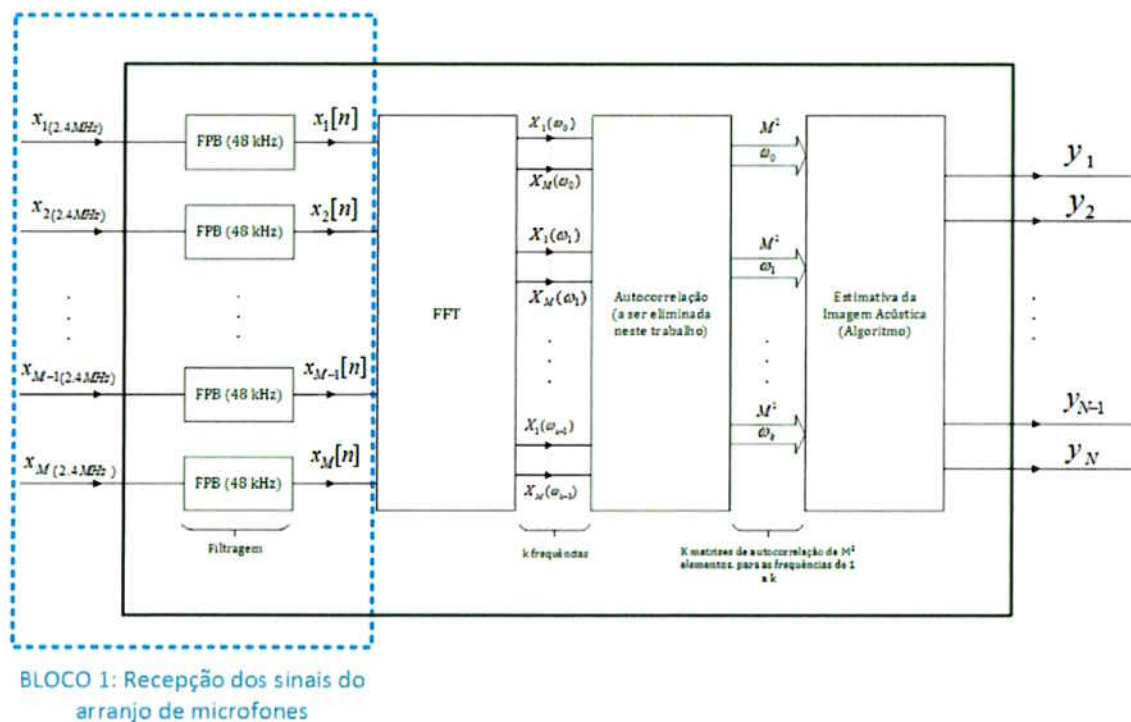
6.4. Análise de Riscos do Projeto

A análise de riscos desse trabalho será feita tendo como base o nível 1 de decomposição funcional do projeto, que é mais simples de ser analisado, porém consegue mostrar claramente onde os possíveis problemas podem vir a ocorrer.

6.4.1. Bloco 1: Recepção dos sinais do arranjo de microfones

Este é o primeiro bloco do projeto. É nele que os sinais vindos das fontes espalhadas no espaço são captados e processados (conversão Analógico/Digital). Os sinais que saem desse bloco são os sinais captados por cada um dos M microfones do arranjo, e são, também, as entradas do bloco seguinte, que realiza o cálculo da FFT de cada um desses sinais.

Figura 8 - Bloco 1: Recepção dos sinais do arranjo de microfones



Esse bloco possui alguns riscos intrínsecos no seu funcionamento, sendo alguns deles os seguintes:

- Arranjo de Microfones pode simplesmente não funcionar quando ele for necessário, e, como apenas há um arranjo deste tipo no LPS (Laboratório de Processamento de Sinais da POLI-USP), isso pode impedir a realização das medidas necessárias para a realização deste trabalho.

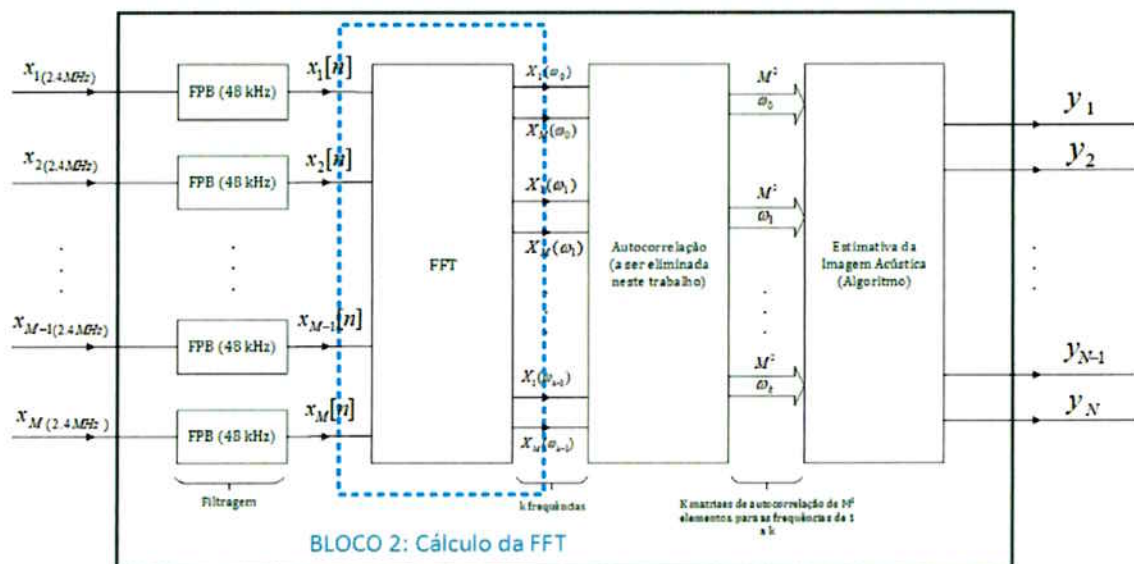
- Mesmo que ele funcione, por falta de calibração ou erro humano na instalação do equipamento de medição, o arranjo pode não realizar a coleta de dados da maneira desejada.
- Os microfones presentes no arranjo podem queimar, pois são bastante sensíveis.

A maior parte desses problemas foi solucionada, pois já há um banco de dados devidamente salvo, com as informações previamente coletadas.

6.4.2. Bloco 2: Cálculo da FFT

Este bloco tem a simples função de tomar a FFT (*Fast Fourier Transform*, ou Transformada Rápida de Fourier) de cada um dos sinais que vêm dos M microfones.

Figura 9 - Bloco 2: Cálculo da FFT



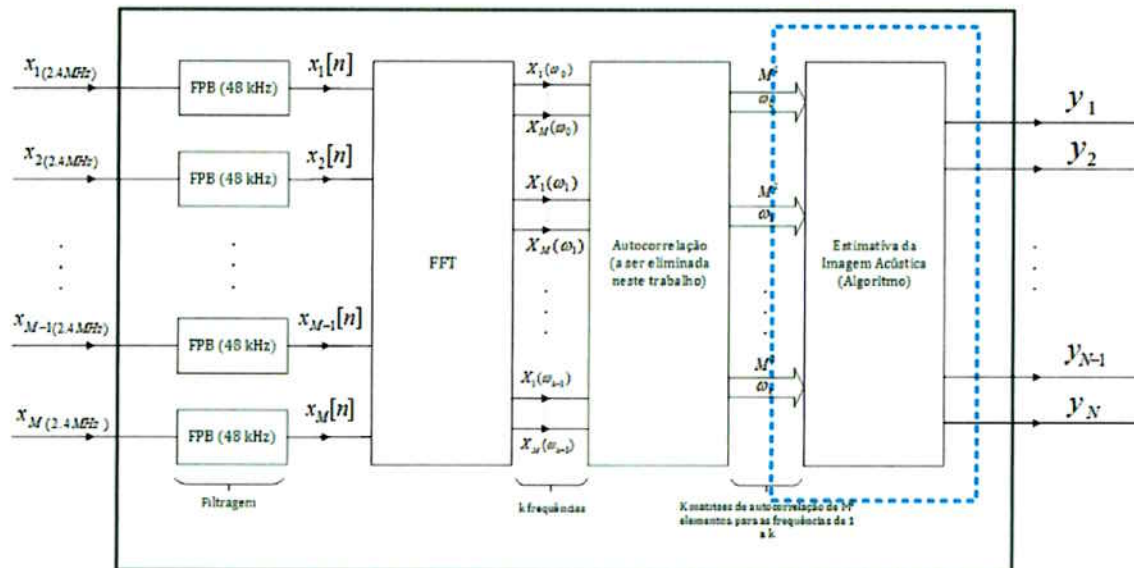
Ele possui poucos riscos, pois fazendo uso, por exemplo, do MATLAB® para se realizar este cálculo, ele se faz muito simples.

6.4.3. Bloco 3: Algoritmo de estimativa

Este é, sem dúvida, o bloco mais problemático do projeto. Ele é o cerne da mudança proposta nesse trabalho, pois é nele que será implementado o novo algoritmo que será aqui criado.

Figura 10 - Bloco 3: Algoritmo de Estimativa

BLOCO 3: Algoritmo de Estimativa



Os principais riscos neste bloco são os seguintes:

- Sabe-se que cálculos serão poupados quando este algoritmo for implementado (uma vez que a matriz de autocorrelação será eliminada), o que pode ajudar na melhora do tempo de processamento (custo computacional) e, possivelmente, na resolução, já que informação que antes era perdida agora será utilizada. Mas a qualidade da imagem ainda é uma incógnita.
- O método pode ser mais difícil de convergir, por exigir mais iterações para tanto.
- O algoritmo pode ser muito sensível aos parâmetros ou mais sensível que os algoritmos atuais.
- Por envolver números complexos, pode haver problemas não antecipados.

Apesar dos riscos aqui apresentados, na próxima seção vai se tentar mostrar a viabilidade deste projeto, tendo em mãos os resultados obtidos em duas publicações, devidamente referenciadas neste trabalho.

6.5. Prova de Conceito

A grande parte dos trabalhos na literatura para estimação de imagens acústicas envolve o cálculo de matrizes de autocorrelação como parte intermediária do seu método, como já foi exposto. Porém, também como já foi dito, há perda de informação quando isso é feito, por causa das hipóteses que são assumidas. Seria interessante

que essas hipóteses fossem eliminadas e se gerasse um algoritmo que conseguisse realizar a estimativa tendo como dados de entrada apenas os dados referentes à FFT dos sinais nos microfones. Isso já foi proposto no artigo referido em [6]: neste artigo, os autores propõem exatamente este método, isto é, não usam matrizes de autocorrelação no seu método de estimação, mas não fazem uso de alguma transformada nessa etapa que acelere os cálculos de estimação.

Já na tese de mestrado referida em [1], o autor faz exatamente o oposto, ou seja, ele demonstra a viabilidade de uma transformada (a KAT – *Kronecker Array Transform*) na diminuição dos cálculos necessários para se estimar as imagens acústicas, mas fazendo uso de um algoritmo de estimação que ainda usa matrizes de autocorrelação.

Portanto, sabendo dos resultados desses dois casos, pode-se afirmar que uní-los é possível e se faz muito interessante, pois se estaria unindo o melhor de dois mundos: a não perda de informações de um método, pelo não uso da matriz de autocorrelação, e a velocidade nos cálculos que se pode atingir fazendo uso de uma transformada que auxilie no caso deste projeto.

Finalmente, criar um método que una o algoritmo presente em [6] a uma transformada que acelere seus cálculos de estimação, como em [1], é o principal escopo deste projeto, e se prova bastante plausível tentar fazê-lo dados os resultados anteriores, ou seja, por não ser um problema trivial, mas com alta probabilidade de sucesso, pode-se afirmar que há viabilidade nesse projeto.

6.6. Desenvolvimento do Projeto

A parte inicial do desenvolvimento desse projeto se deu com o estudo e/ou revisão de conceitos que foram frequentemente utilizados no decorrer deste trabalho. Alguns desses conceitos valem ser comentados devido à sua importância para que se torne mais claro o entendimento do que está se implementando e quais são os objetivos em se estudar tais conceitos. A seguir, faz-se um breve resumo dos conceitos mais importantes para este trabalho.

6.6.1. Modelo de Propagação de Ondas

O tipo de onda analisado serão ondas sonoras e se pode modelar sua propagação no espaço com seguinte equação de onda:

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} + \frac{\partial^2 p}{\partial z^2} = \frac{1}{c^2} \frac{\partial^2 p}{\partial t^2},$$

onde p está relacionada à pressão sonora da onda, x, y e z são as coordenadas escolhidas e c é a velocidade de propagação da onda sonora no meio. A solução da equação anterior pode ser decomposta, para um ponto $\mathbf{p} = [p_x \ p_y \ p_z]^T$ suficientemente longe de todas as fontes, como uma superposição de ondas planas se propagando de diferentes direções.

Uma única onda se propagando na direção $\mathbf{k} = [k_x \ k_y \ k_z]^T$ tem a forma:

$$x(t, \mathbf{p}) = f(t - \mathbf{k}^T \mathbf{p}, u) = f(t + \frac{\omega \mathbf{u}^T \mathbf{p}}{c}, u),$$

onde \mathbf{k} é o vetor de número de onda, um vetor que aponta na direção de propagação da onda, cuja magnitude é $\|\mathbf{k}\| = \omega/c$, $\omega = 2\pi f$, f é a frequência do sinal, e $\mathbf{u} = -c\mathbf{k}/\omega$ é um vetor unitário que aponta para a direção de onde a onda está chegando ao arranjo (um vetor que tem origem no arranjo de sensores e aponta para a fonte do sinal).

Se o sinal da fonte apresentar apenas uma frequência, pode-se escrever que:

$$f(t, u) = F(\omega, u) e^{j\omega t},$$

e o sinal num ponto \mathbf{p} no espaço terá a forma:

$$x(t, \mathbf{p}) = X(\omega, \mathbf{p}) e^{j\omega t}, \text{ com } X(\omega, \mathbf{p}) = F(\omega, u) e^{j\frac{\omega}{c} \mathbf{u}^T \mathbf{p}}.$$

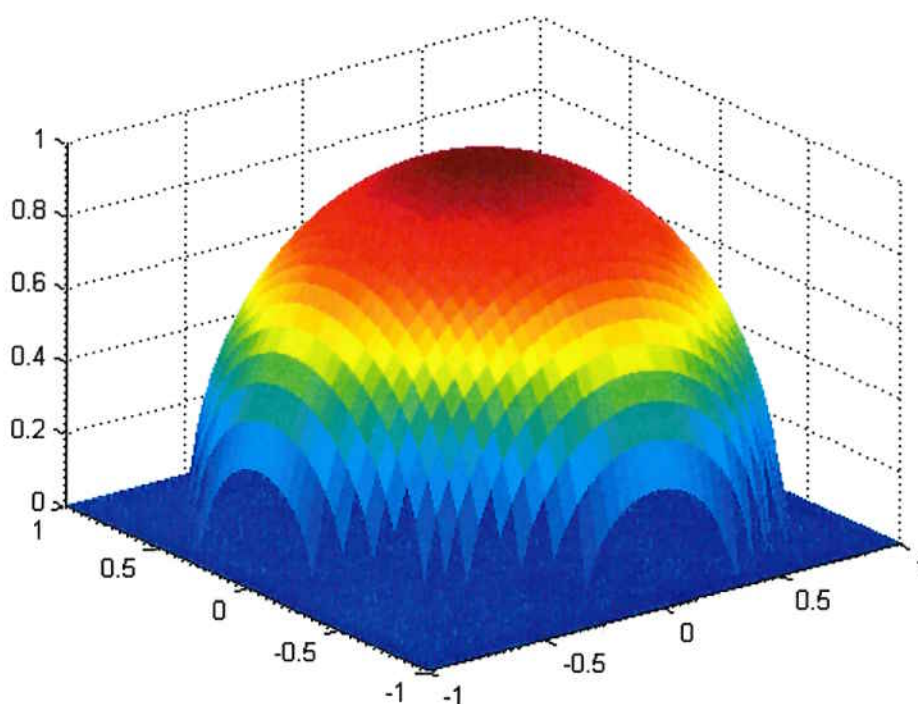
Neste trabalho, em todas as simulações, será considerado um arranjo de microfones planar de M sensores, isto é, o arranjo terá microfones com coordenadas $\mathbf{p}_m = [p_{m_x} \ p_{m_y} \ p_{m_z}]^T$, com $p_{m_z} = 0$, $m = 1, \dots, M$. Pode-se escrever o sinal que chega em cada um dos microfones quando a onda plana chega ao arranjo, sendo que esta onda vem da direção \mathbf{u} como:

$$X(\omega) = \begin{bmatrix} X(\omega, \mathbf{p}_1) \\ X(\omega, \mathbf{p}_2) \\ \vdots \\ X(\omega, \mathbf{p}_M) \end{bmatrix} = F(\omega, \mathbf{u}) \begin{bmatrix} e^{j\frac{\omega}{c} \mathbf{u}^T \mathbf{p}_1} \\ e^{j\frac{\omega}{c} \mathbf{u}^T \mathbf{p}_2} \\ \vdots \\ e^{j\frac{\omega}{c} \mathbf{u}^T \mathbf{p}_M} \end{bmatrix} \triangleq F(\omega, \mathbf{u}) \mathbf{v}(\mathbf{u}),$$

onde $\mathbf{v}(\mathbf{u})$ é conhecido como “array manifold vector” ou vetor de direções do arranjo. Este vetor contém as informações de atraso com o qual a onda que se propaga na direção $-\mathbf{u}$ chega ao arranjo. O arranjo consegue diferenciar ondas planas que chegam de diferentes direções por esses atrasos nos sinais em cada microfone. Esse vetor tem um papel fundamental em imageamento acústico exatamente por representar estes atrasos de maneira concisa e conveniente.

Para os casos onde há mais de uma fonte, se o campo sonoro próximo ao arranjo for linear e todas as fontes estiverem longe o suficiente para que a condição de onda plana chegando no arranjo seja respeitada o sinal recebido nos microfones será a superposição de um número infinito de ondas planas. Aqui essa superposição infinita será aproximada pelo caso discreto, que corresponde a aproximar os sinais que chegam ao arranjo como a superposição de um número finito de ondas planas vindo de direções previamente escolhidas, isto é, divide-se o espaço visto pelo arranjo de maneira discreta, com N amostras, onde pode haver uma onda plana vinda de cada direção, de \mathbf{u}_1 até \mathbf{u}_N , como na figura a seguir:

Figura 11 - Espaço visto pelo arranjo de microfones, amostrado



Fonte: Autoria Própria. Figura feita com auxílio do MATLAB.

O modelo resultante para o caso de mais de uma fonte fica sendo, portanto:

$$X(\omega) = \sum_{n=1}^N F(\omega, \mathbf{u}_n) v(\mathbf{u}_n) = [v(\mathbf{u}_1) \quad \dots \quad v(\mathbf{u}_N)] \begin{bmatrix} F(\omega, \mathbf{u}_1) \\ \vdots \\ F(\omega, \mathbf{u}_N) \end{bmatrix}.$$

Até agora, assumiu-se implicitamente que os sinais que vêm das fontes sonoras são sinais determinísticos. Neste caso, a imagem acústica discreta que se quer estimar é dada pela expressão $Y(\omega, \mathbf{u}_n) \triangleq |F(\omega, \mathbf{u}_n)|^2$.

É necessário que se expanda esse modelo para incluir sinais mais genéricos, podendo os sinais vindo de cada direção serem processos aleatórios estacionários. Um modelo como o anteriormente mostrado não está disponível, uma vez que processos estacionários não tem energia finita, apenas o espectro de potência é definido. Para lidar com esse caso, é necessário o uso da Transformada Discreta de Fourier (TDF), ou, mais especificamente na implementação em MATLAB, da FFT ("Fast Fourier Transform"). Assume-se aqui que a transformada toma K amostras para que seu cálculo seja realizado, e que as frequências do sinal e de amostragem respeitam as condições de Nyquist para não haver nenhum tipo de problema de amostragem.

Os métodos já testados em [2] começam a diferir dos métodos propostos neste trabalho a partir deste ponto. A estimação realizada em [2] é feita da seguinte maneira:

Seja $\mathbf{V} = [v(\mathbf{u}_1) \quad \dots \quad v(\mathbf{u}_N)]$. Nesse caso, a matriz de autocorrelação do sinal que chega aos microfones pode ser escrita como sendo, para uma dada frequência ω :

$$R_x(\omega) = E\{X(\omega)X(\omega)^H\} = \mathbf{V}R_F(\omega)\mathbf{V}^H,$$

onde $X(\omega)$ é a transformada de Fourier em tempo discreto dos sinais dos microfones e $R_F(\omega)$ é a matriz de autocorrelação dos sinais das fontes e pode ser escrita como:

$$R_F(\omega) = E\left\{ \begin{bmatrix} F(\omega, \mathbf{u}_1) \\ \vdots \\ F(\omega, \mathbf{u}_N) \end{bmatrix} [\text{conj}(F(\omega, \mathbf{u}_1)) \quad \dots \quad \text{conj}(F(\omega, \mathbf{u}_N))] \right\},$$

e $E\{\cdot\}$ é o valor esperado e $\{\cdot\}^H$ denota a operação que calcula o hermitiano de uma matriz ou vetor, que se define como o transposto conjugado da matriz. $R_F(\omega)$

normalmente é uma matriz cheia, significando que os sinais vindo de diferentes direções podem ser correlacionados. Porém, para tornar a solução do problema factível para valores grandes de N , é comum se assumir que não existe correlação entre as fontes. Isso faz com $R_F(\omega)$ seja uma matriz diagonal, o que facilita a abordagem para a estimação, porém cria uma aproximação para o que se é observado na realidade. Tendo essas hipóteses, pode-se afirmar que:

$$R_x = E\{XX^H\} = \sum_{n=1}^N v(u_n)Y(u_n)v^H(u_n),$$

com $Y(u_n) = E\{|F(\omega, u_n)|^2\}$ e são os elementos da diagonal de R_F e a imagem acústica que se deseja estimar. Para ajudar a resolver esse problema, de achar $Y(u_n)$ na equação anterior, sejam $r_x \triangleq \text{vec}(R_x)$ e $A = [\text{conj}(v(u_1)) \otimes v(u_1) \cdots \text{conj}(v(u_N)) \otimes v(u_N)]$, onde \otimes denota o Produto de Kronecker, definido da seguinte maneira:

Para matriz $A = \begin{bmatrix} a_{11} & a_{12} & \cdots \\ a_{21} & a_{22} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$ e B , matrizes A e B quaisquer. Então:

$$A \otimes B \triangleq \begin{bmatrix} a_{11}B & a_{12}B & \cdots \\ a_{21}B & a_{22}B & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}.$$

Aplicando para a equação $R_x = \sum_{n=1}^N v(u_n)Y(u_n)v^H(u_n)$ a seguinte propriedade:

$$\text{vec}(ACB) = (B^T \otimes A) \text{vec}(C),$$

podemos escrever que:

$$r_x = A \begin{bmatrix} Y(u_1) \\ \vdots \\ Y(u_N) \end{bmatrix} = Ay,$$

e se quer estimar y .

O modelo adotado neste trabalho, a estimação fica bem mais simples, uma vez que não há necessidade de se calcular as matrizes de autocorrelação dos sinais das fontes (que na prática não são conhecidas) e nem dos sinais dos microfones (que são as únicas informações conhecidas), apenas as TDFs de cada sinal e a matriz com os vetores de direções. Para este trabalho, sejam:

$$X(\omega) = \begin{bmatrix} X(\omega, p_1) \\ X(\omega, p_2) \\ \vdots \\ X(\omega, p_M) \end{bmatrix} = [X]_{M \times 1}$$

$$Y(\omega) = \begin{bmatrix} Y(\omega, u_1) \\ \vdots \\ Y(\omega, u_N) \end{bmatrix} = [Y]_{N \times 1}$$

$$V = \begin{bmatrix} v(u_1, p_1) & \cdots & v(u_N, p_1) \\ \vdots & \ddots & \vdots \\ v(u_1, p_M) & \cdots & v(u_N, p_M) \end{bmatrix} = [V]_{M \times N},$$

como definidos para o caso anterior, tem-se que o modelo que será utilizado para se realizar as estimativas dos sinais no microfones é o seguinte:

$$X(\omega) = V Y(\omega)$$

e o valor que se deseja estimar são os valores de $[Y]_{N \times 1}$, cujas potências médias formam a imagem acústica. Na seção a seguir são discutidos os métodos para realizar essas estimativas.

6.6.2. Métodos de Estimação de Imagens Acústicas

Existem diversas maneiras de se realizar a estimativa para as imagens acústicas, com vários níveis de complexidade, como já citado nas partes iniciais deste trabalho. Todos eles tem como hipótese o fato de que o arranjo se encontra da região de campo distante em relação às fontes dos sinais. Os métodos que foram utilizados em [2] e neste trabalho são os se seguem.

6.6.2.1. *Delay and Sum: DAS*

É um dos métodos mais simples de estimativa e é descrito com precisão em [4]. É um *beamformer* determinístico que tenta compensar o atraso relativo em cada sensor do arranjo e calcula a média dos sinais resultantes, a partir da seguinte expressão para o seu respectivo filtro espacial (que realiza a ponderação para cada sensor):

$$w_{DAS}(u) = \frac{1}{M} v(u) = \frac{v(u)}{v^H(u)v(u)}$$

Para se obter a imagem acústica, é preciso estimar a intensidade do som de cada direção u já pré-definida, para se obter um vetor de estimativas \hat{y} para y , do problema descrito em [2], ou $\hat{Y}(\omega, u_n), n = 1, \dots, N$ no caso deste trabalho. Tem-se para [2] que:

$$\hat{y} = \frac{1}{M^2} A^H r_x.$$

Nota-se que o resultado do segundo membro fornece um vetor cujos elementos são números reais, isto é, já é uma aproximação para as potências dos sinais vindo de cada direção. Agora, para este trabalho temos que:

$$\hat{Y} = V^H X,$$

sendo que o resultado obtido do produto do segundo membro é um vetor com elementos complexos. Para se calcular a potência, é necessário calcular $|\cdot|^2$ de cada elemento.

Vale notar que \hat{y} só seria igual a y somente se $A^{-1} = \frac{1}{M^2} A^H$, o que acontece apenas se as colunas de A fossem ortogonais e A fosse uma matriz quadrada, que são condições que normalmente não são encontradas na prática. São necessários métodos que condizem mais com a realidade.

6.6.2.2. *Deconvolution Approach for the Mapping of Acoustic Sources: DAMAS2*

É um método que utiliza deconvoluções 2D para estimar a imagem acústica. Denotando por Y a imagem acústica bidimensional, isto é, y vetor rearranjado como uma matriz (imagem bidimensional), e da mesma maneira definir \hat{Y} para a imagem acústica bidimensional estimada. O método DAMAS2 calcula uma melhor aproximação \hat{Y} para Y dada a estimativa realizada utilizando o método de Soma e Atraso (*Delay and Sum - DAS*), denotado como \tilde{Y} , de maneira iterativa, como mostrado na expressão a seguir:

$$\hat{Y}^{(k+1)} = \max \left\{ \hat{Y}^{(k)} + \frac{1}{a} [\tilde{Y} - (P * \hat{Y}^{(k)})], 0 \right\},$$

onde $*$ denota a deconvolução 2D, $\hat{Y}^{(k)}$ é a imagem reconstruída na iteração k com $\hat{Y}^{(0)} = 0$, P é a *Point Spread Function* (PSF) discretizada do arranjo, cuja definição

precisa está presente em [4], $\alpha = \sum_{i,j} |P|_{i,j}$ é uma constante e $\max\{\cdot\}$ retorna o máximo. A maneira como este método está definido é para garantir que as estimativas tenham valores estritamente positivos.

Esse é um dos problemas pelos quais a definição deste método para o problema deste trabalho se torna complexa. Neste trabalho, os valores de Y estão relacionados aos valores da transformada de Fourier em tempo discreto do sinal vindo de uma dada direção em uma frequência qualquer, e, portanto, seu valor assume valores complexos, e não se pode definir uma comparação tal que o número complexo seja estritamente positivo, pois não faz sentido algum. Outro ponto para se utilizar o DAMAS2 seria que o método teria que ser totalmente repensado para as novas hipóteses deste problema e definir uma deconvolução 2D para este novo caso mais uma PSF seria extremamente, além de não ser escopo deste trabalho. Tendo em vista esses problemas de definição para sua correta utilização, preferiu-se não utilizar esse método neste trabalho.

6.6.2.3. Regularização l_1

É um método de reconstrução que utiliza conceitos de otimização convexa, que podem ser resolvidas por métodos iterativos eficientes computacionalmente.

Assume-se que o campo acústico que chega ao arranjo de microfones foi gerado por apenas algumas fontes pontuais, compactas, isto é, a distribuição de fontes no espaço vista pelo arranjo é esparsa. Neste caso é possível aplicar uma restrição, presente no seguinte problema de otimização convexo proposto para o problema proposto em [2]:

$$\underset{(\hat{y}, \sigma^2)}{\text{minimizar}} \quad \|r_x - A\hat{y} - \sigma^2 \text{vec}(I)\|_2^2 \text{ tal que } \hat{y}_{i,j} \geq 0, \sigma^2 \geq 0 \text{ e } \|\hat{y}\|_1 \leq \lambda,$$

onde I é a matriz identidade, σ^2 é a variância de um ruído presente nos sinais (incluído no modelo anteriormente adotado nos sinais dos microfones) e $\|\hat{y}\|_1$ denota a norma l_1 , definida pela soma do módulo das componentes do vetor \hat{y} . A restrição da norma l_1 $\|\hat{y}\|_1 \leq \lambda$ serve para regularizar o problema enquanto se força que a esparsividade e λ é um parâmetro de regularização.

Outra opção para se definir a equação anterior como o que foi definido com "*basis pursuit denoising problem*", que tem a seguinte forma:

$$\text{minimizar}_{(\hat{Y})} \|\hat{Y}\|_1 \text{ tal que } \|r_x - A\hat{Y}\|_2 \leq \sigma,$$

que é a forma adotada em [2] para se estimar a imagem acústica e também será adotada para o problema deste trabalho, onde σ é o desvio padrão do ruído (assumiu-se esse valor como parâmetro de regularização). Para este trabalho, o problema de otimização convexo fica sendo descrito da seguinte maneira:

$$\text{minimizar}_{(\hat{Y})} \|\hat{Y}\|_1 \text{ tal que } \|X - V\hat{Y}\|_2 \leq \sigma.$$

6.6.2.4. Regularização TV (*Total Variation*)

Outra possibilidade ainda é reconstruir a imagem acústica utilizando um método que faz uso de regularização da variação total ("*Total Variation Regularization - TV*"). Define-se a seguinte norma, denominada norma isotrópica da variação total:

$$\|Y\|_{TV} = \sum_{i,j} \sqrt{[\nabla_x Y]_{i,j}^2 + [\nabla_y Y]_{i,j}^2},$$

onde ∇_x e ∇_y denotam os primeiros operadores de diferença nas direções x e y (estão relacionados com as primeiras derivadas) e i e j são os índices relacionados as coordenadas x e y , respectivamente. O seguinte problema de otimização pode, dada a definição anterior, ser resolvido para o problema proposto em [2]:

$$\text{minimizar}_{(\hat{Y})} \|\hat{Y}\|_{TV} + \mu \|r_x - A\hat{Y}\|_2^2 \text{ tal que } [\hat{Y}]_{i,j} \geq 0.$$

O primeiro termo mede como a imagem oscila e o segundo termo da minimização assegura uma boa aproximação entre a imagem reconstruída e os dados medidos do arranjo. Esse método consegue ter imagens acuradas e reconstruções estáveis com convergência garantida.

Esse mesmo método pode ser usado aqui neste trabalho, sendo que o problema de otimização fica sendo o seguinte:

$$\text{minimizar}_{(\hat{Y})} \|\hat{Y}\|_{TV} + \mu \|X - V\hat{Y}\|_2^2.$$

Deve-se notar, contudo, que este método de reconstrução não é apropriado para imagens pontuais, de maneira oposta ao método de regularização l_1 , este é um método adequado para imagens ditas suaves, onde a distribuição é continuada, e não apenas pontos esparsos.

6.6.3. Transformadas Rápidas: KAT (*Kronocker Array Transform*) e Nova KAT proposta

Apresentados os métodos anteriores, é possível estimar imagens acústicas com boa resolução, porém usando arranjos com um número de microfones moderado, condição que se deve a problemas de dimensionalidade das matrizes utilizadas no programa que implementa tais métodos. Alguns desses métodos são iterativos, o que na prática requer que haja operações de multiplicação de matrizes repetidas vezes, tanto na forma $A\hat{Y}$ (ou $V\hat{Y}$, no caso deste trabalho) ou $A^H r_x$ (ou $V^H X$).

Na prática, a matriz A para o caso de estimação que faz uso da matriz de autocorrelação é geralmente grande para se ter uma resolução muito boa das imagens acústicas. Um exemplo presente em [2] mostra que se um arranjo tiver 64 microfones e a imagem desejada tem 128×128 pixels, isto é, um número de microfones $M = 64$ e de divisões $N_x = N_y = 128$ ($N = N_x \times N_y = 128^2 = 16384$), a matriz A teria $M^2 \times N = 4096 \times 16384 = 67108864$ elementos. Esse exemplo evidencia que, quanto mais complexo for o método aplicado para reconstrução das imagens acústicas, mais tempo de processamento ele irá consumir. Como comparação, a mudança proposta neste trabalho já atua nesse problema de dimensionalidade: seguindo o exemplo anterior, isto é, com os mesmos valores de M e N , ter-se-ia uma matriz V de dimensões $M \times N = 64 \times 16384 = 1048576$ elementos, 64 vezes menos elementos que a do exemplo anterior.

A ideia central das transformadas rápidas, seja a NNFFT, NFFT ou da KAT (e da nova KAT, como será descrito nesta seção), é a de acelerar os cálculos desses produtos de matrizes, reorganizando a estrutura da matriz A no caso que utiliza matrizes de autocorrelação, e da matriz V no caso proposto neste trabalho, que não fará uso de matrizes de autocorrelação. Focou-se na utilização da KAT neste projeto, uma vez que esta transformada foi a que apresentou os melhores resultados [2]. A *Kronocker Array Transform*, como já foi descrita no início deste trabalho, é uma transformada cuja utilização depende de como os sensores estão arranjos e como o espaço visto pelo arranjo foi amostrado. A KAT vale para arranjos planares e para arranjos de geometria e amostragem separáveis, isto é, arranjos com sensores e amostragem distribuídos em arranjos retangulares, não necessariamente uniformes.

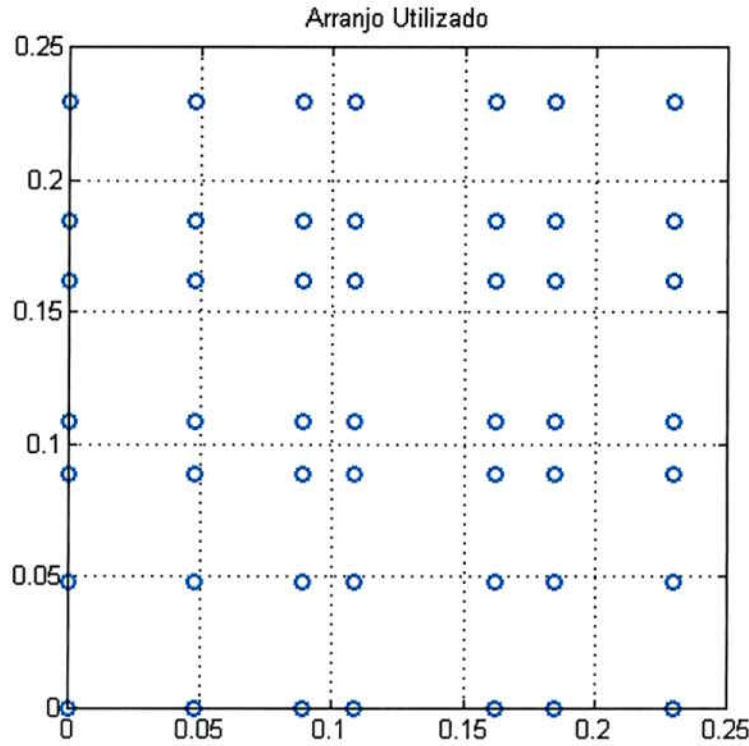


Figura 12 - Exemplo de Arranjo de Microfones Separável e Não Uniforme

Considere um arranjo planar, como microfones arranjados em uma malha retangular como na figura anterior e assuma que o sistema de coordenadas está no plano do arranjo seja o plano xy e que as direções \mathbf{u}_n foram escolhidas tais que:

$$\mathbf{u}_{j+(i-1)N_y} = \begin{bmatrix} u_{x,i} \\ u_{y,j} \\ \sqrt{1 - u_{x,i}^2 - u_{y,j}^2} \end{bmatrix}, 1 \leq i \leq N_x \text{ e } 1 \leq j \leq N_y$$

onde $-1 \leq u_{x,i} \leq 1$ e $-1 \leq u_{y,j} \leq 1$, N_x é o número de direções vistas na direção x e N_y na direção y . Esse vetor apresenta a informação de todas as direções vistas pelo arranjo de maneira organizada para tornar a implementação computacional mais fácil. O índice j caminha na direção y passando por todas as linhas da malha retangular, e o índice i caminha na direção x , passando por todos os elementos presentes na linha j .

Para a posição dos microfones, a malha retangular e a posição dos microfones nesta malha pode ser organizadas da seguinte maneira:

onde M_x é o número de microfones na direção x e M_y é o número de microfones na direção y , com $M_x \times M_y = M$, M total de microfones no arranjo. O índice s caminha na direção y passando por todas as linhas da malha retangular, e o índice r caminha na direção x , passando por todos os elementos presentes na linha s .

Diagrama de uma grade 6x6 de microfones. O eixo horizontal é rotulado "Nx microfones" e o eixo vertical "Ny microfones". Os pontos da grade são rotulados p_1 a p_{13} , com p_1 no canto inferior esquerdo e p_{13} no canto superior esquerdo. O texto "...e assim por diante" indica a continuação da grade. No topo da grade, há o texto "Arranjo Utilizado".

Dadas as hipóteses anteriores, pode-se decompor o vetor de direções numa dada direção da seguinte maneira, utilizando o produto de Kronecker já definido:

onde $\mathbf{v}_x(u_{x,i}) \triangleq \begin{bmatrix} e^{j\frac{\omega}{c}u_{x,i} p_{x,1}} \\ e^{j\frac{\omega}{c}u_{x,i} p_{x,2}} \\ \vdots \\ e^{j\frac{\omega}{c}u_{x,i} p_{x,M_x}} \end{bmatrix}$ e $\mathbf{v}_y(u_{y,j}) \triangleq \begin{bmatrix} e^{j\frac{\omega}{c}u_{y,j} p_{y,1}} \\ e^{j\frac{\omega}{c}u_{y,j} p_{y,2}} \\ \vdots \\ e^{j\frac{\omega}{c}u_{y,j} p_{y,M_y}} \end{bmatrix}$, cujas definições saem da

46

$$e^{j\frac{\omega}{c}u_n^T p_m} = e^{j\frac{\omega}{c}u_{x,i} p_{x,r}} \times e^{j\frac{\omega}{c}u_{y,j} p_{y,s}}, n = j + (i-1)N_y \text{ e } m = s + (r-1)M_y,$$

não havendo termos em z na expressão anterior, pois o arranjo é considerado planar, isto é, a posição z dos microfones é zero, como modelado anteriormente.

Sob estas condições, os produtos entre matrizes e vetores podem ser obtidos de maneira mais eficiente pelo rearranjo da matriz A , e obter-se assim um custo computacional menor que quando utilizada a multiplicação direta. Os resultados até este ponto apresentados valem para os dois casos, com e sem matrizes de autocorrelação (matrizes A e V).

Para o caso que faz uso das matrizes de autocorrelação presente em [2], pode-se definir matrizes V_x e V_y como sendo:

$$\begin{aligned} V_x &\triangleq \begin{bmatrix} v_x^*(u_{x,1}) \otimes v_x(u_{x,1}) & \cdots & v_x^*(u_{x,N_x}) \otimes v_x(u_{x,N_x}) \end{bmatrix} = \\ &= \begin{bmatrix} v_{x,1}^*(u_{x,1}) \times v_{x,1}(u_{x,1}) & \cdots & v_{x,1}^*(u_{x,N_x}) \times v_{x,1}(u_{x,N_x}) \\ v_{x,1}^*(u_{x,1}) \times v_{x,2}(u_{x,1}) & \cdots & v_{x,1}^*(u_{x,N_x}) \times v_{x,2}(u_{x,N_x}) \\ \vdots & \ddots & \vdots \\ v_{x,M_x}^*(u_{x,1}) \times v_{x,M_x}(u_{x,1}) & \cdots & v_{x,M_x}^*(u_{x,N_x}) \times v_{x,M_x}(u_{x,N_x}) \end{bmatrix} \in \mathbb{C}^{M_x^2 \times N_x} \\ V_y &\triangleq \begin{bmatrix} v_y^*(u_{y,1}) \otimes v_y(u_{y,1}) & \cdots & v_y^*(u_{y,N_y}) \otimes v_y(u_{y,N_y}) \end{bmatrix} = \\ &= \begin{bmatrix} v_{y,1}^*(u_{y,1}) \times v_{y,1}(u_{y,1}) & \cdots & v_{y,1}^*(u_{y,N_y}) \times v_{y,1}(u_{y,N_y}) \\ v_{y,1}^*(u_{y,1}) \times v_{y,2}(u_{y,1}) & \cdots & v_{y,1}^*(u_{y,N_y}) \times v_{y,2}(u_{y,N_y}) \\ \vdots & \ddots & \vdots \\ v_{y,M_y}^*(u_{y,1}) \times v_{y,M_y}(u_{y,1}) & \cdots & v_{y,M_y}^*(u_{y,N_y}) \times v_{y,M_y}(u_{y,N_y}) \end{bmatrix} \in \mathbb{C}^{M_y^2 \times N_y} \end{aligned}$$

Pode-se definir \hat{Y} para um dado vetor \hat{y} , tal que $\hat{y} = \text{vec}(\hat{Y})$ e $Z = V_y \hat{Y} V_x^T$. Para uma certa matriz de permutação H , responsável por reorganizar os elementos de um dado vetor, pode-se verificar que:

$$r_x = A\hat{y} = H \text{vec}(Z) = H \text{vec}(V_y \hat{Y} V_x^T)$$

e esta é a *Kronecker Array Transform*. O produto do tipo $\hat{y} = A^H \hat{r}$ pode ser obtido similarmente. Definindo \bar{Z} tal que $\text{vec}(\bar{Z}) = H^T \hat{r}$, pode-se mostrar que:

$$\text{vec}(A^H \hat{r}) = \text{vec}(V_y^H \bar{Z} V_x^*).$$

Todas as vezes que algum produto entre matrizes e vetores aparecer na implementação em MATLAB, por exemplo, a troca do produto direto pelo uso da transformada vai reduzir o tempo de processamento computacional necessário para se reconstruir a imagem acústica. Vale notar que a transformada, segundo o que é apontado em [2], funciona melhor que o produto direto e as demais transformadas comentadas para uma certa dimensão do problema (número de microfones e número de amostras do espaço), a partir do qual a transformada passa a ter vantagens computacionais.

Para o caso estudado neste trabalho, feitas as hipóteses como no caso anterior e lembrando que:

$$V = \begin{bmatrix} v(u_1, p_1) & \cdots & v(u_N, p_1) \\ \vdots & \ddots & \vdots \\ v(u_1, p_M) & \cdots & v(u_N, p_M) \end{bmatrix} = [V]_{M \times N},$$

com $M = M_x \times M_y$ e $N = N_x \times N_y$, podemos definir, neste caso de forma simplificada, que:

$$V_x \triangleq [v_x(u_{x,1}) \quad \cdots \quad v_x(u_{x,N_x})] = \begin{bmatrix} v_{x,1}(u_{x,1}) & \cdots & v_{x,1}(u_{x,N_x}) \\ v_{x,2}(u_{x,1}) & \cdots & v_{x,2}(u_{x,N_x}) \\ \vdots & \ddots & \vdots \\ v_{x,M_x}(u_{x,1}) & \cdots & v_{x,M_x}(u_{x,N_x}) \end{bmatrix} \in \mathbb{C}^{M_x \times N_x}$$

$$V_y \triangleq [v_y(u_{y,1}) \quad \cdots \quad v_y(u_{y,N_y})] = \begin{bmatrix} v_{y,1}(u_{y,1}) & \cdots & v_{y,1}(u_{y,N_y}) \\ v_{y,2}(u_{y,1}) & \cdots & v_{y,2}(u_{y,N_y}) \\ \vdots & \ddots & \vdots \\ v_{y,M_y}(u_{y,1}) & \cdots & v_{y,M_y}(u_{y,N_y}) \end{bmatrix} \in \mathbb{C}^{M_y \times N_y}$$

e como consequência das definições anteriores, podemos verificar que $V = V_x \otimes V_y$. Como no caso que faz uso das matrizes de autocorrelação, podemos definir os produtos $V\hat{Y}$ e $V^H X$, com $[\hat{Y}]_{(N_y \times N_x) \times 1}$ e $[X]_{(M_y \times M_x) \times 1}$, da seguinte maneira para o caso deste trabalho:

$$X = V\hat{Y} = (V_x \otimes V_y)\hat{Y} = \text{vec}(V_y \bar{Y} V_x^T), \text{ com } \bar{Y} = [\hat{Y}]_{N_y \times N_x},$$

$$V^H X = (V_x \otimes V_y)^H X = (V_x^H \otimes V_y^H) X = \text{vec}(V_y^H \bar{X} V_x^*), \text{ com } \bar{X} = [X]_{M_y \times M_x}$$

e deve-se notar que neste caso não existirá a necessidade de uma matriz de permutação como no caso anterior, mais um fator que torna esta análise bem mais simples.

Dadas todas as definições anteriores, passou-se para as etapas de implementação dos algoritmos propostos para, futuramente, comparar-se seus resultados e validar as melhorias propostas aqui nesta seção. As implementações realizadas estão presentes na seção seguinte.

6.7. Implementação dos Algoritmos

A etapa de implementação dos algoritmos para o caso que não faz uso da matriz de autocorrelação, escopo desse trabalho, se fez, primeiramente, com a geração do modelo de propagação dos sinais emitidos pelas fontes espalhadas no espaço e a análise dos sinais recebidos no arranjo de microfones, calculando-se a matriz formada pelos vetores de direções previamente escolhidas para tanto.

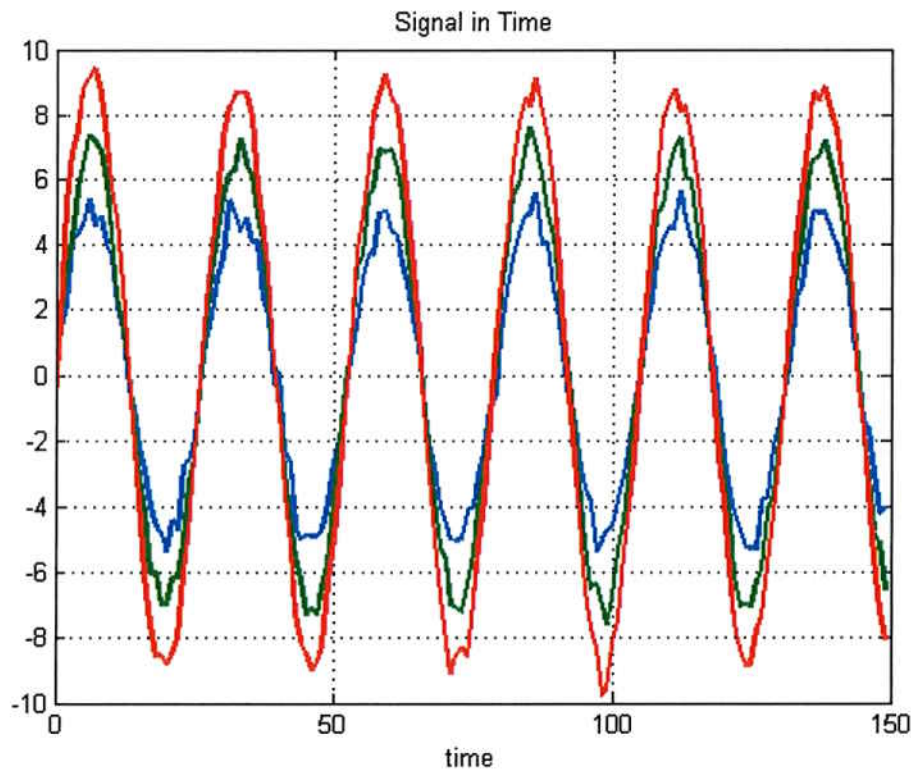
Foram escolhidos alguns valores constantes para todos os algoritmos possíveis de serem implementados, todos devidamente explicados na seção anterior, como critério de comparação inicial. Os códigos principais estão em anexos devidamente referenciados aqui nesta seção, mas algumas funções e trechos de códigos serão utilizados para explicar o funcionamento das partes em comum dos algoritmos implementados.

Os códigos desenvolvidos para cada método de reconstrução de imagens acústicas foram implementados primeiramente sem a utilização da nova KAT e posteriormente fazendo uso desta. As modificações principais nos algoritmos utilizados estão presentes nas seções dos códigos que realizam a estimação de fato. Em relação ao tempo de processamento dos algoritmos, serão analisados tanto o tempo que o código em MATLAB levou para estimar as imagens quanto o custo computacional de cada algoritmo em relação ao número de operações matemáticas necessárias para a estimação (operações estas de soma e multiplicação, presentes nos produtos entre matrizes que existem nos métodos de reconstrução). As comparações entre os resultados para outros cenários que não este inicial, feito com o objetivo de avaliar o sucesso dos métodos preliminarmente, serão realizadas em seções seguintes.

6.7.1. Geração do Modelo e Cenário Inicial Adotado

O cenário escolhido para os testes iniciais foi o seguinte: foi adotada uma frequência $F_s = 2^{17} = 131072$ amostras/segundo, tendo-se, portanto, um período de amostragem $T = \frac{1}{F_s} = 7.629e^{-6}$ segundo/amostra, tomando-se $K = 2^{16} = 65536$ amostras dos sinais das fontes. Foram tomados valores exponenciais com base 2 a fim de acelerar os cálculos da FFT, posteriormente. Foram consideradas 3 fontes espalhadas no espaço visto pelo arranjo de microfones, cujos sinais eram senoidais e tinham a mesma frequência $f_{sig} = 5$ kHz, vinda das direções $\mathbf{u}_{f1} = [-0.6 - 0.8 0]^T$, $\mathbf{u}_{f2} = [0 0 1]^T$ e $\mathbf{u}_{f3} = [0.5 0.5 0.7071]^T$, com amplitude de sinal para a fonte 1 de 5, para a fonte 2 de 7 e para a fonte 3 de 9, e defasagem nula entre os três sinais. Além disso, foi considerado que todos os sinais foram corrompidos por ruídos brancos gaussianos de variância $\sigma^2 = 0.1$. Considerou-se a velocidade de propagação de uma onda sonora no ar como $c = 340.29$ m/s.

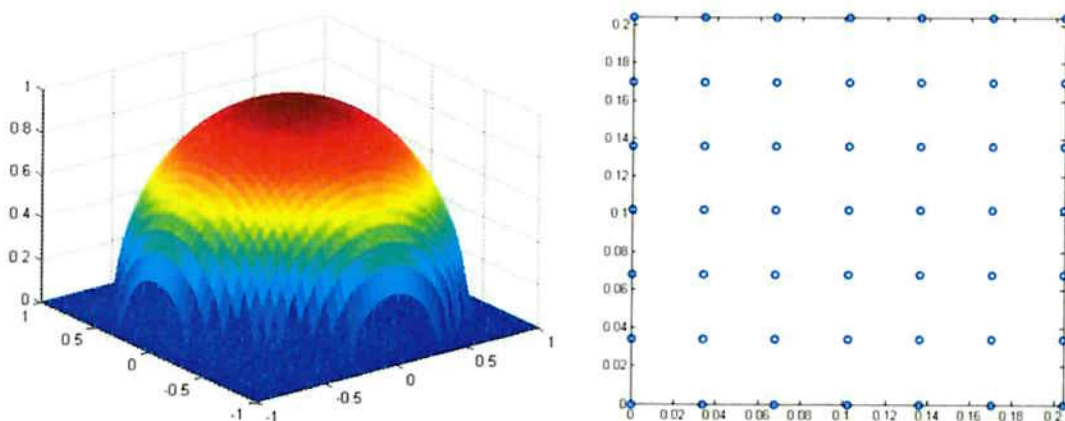
Figura 14 - Sinais das três fontes consideradas no cenário inicial



Para o arranjo, dada a frequência dos sinais, para simplificação dos algoritmos, foi adotado que a distância entre dois sensores no arranjo planar na direção do eixo x

ou na direção y seria equivalente à metade do comprimento de onda dos sinais, dado por $\lambda = \frac{c}{f_{sig}}$, isto é, a distância d entre os microfones foi considerada como sendo $d = \frac{\lambda}{2}$, sendo que $d = 0.034029 \text{ m}$ neste caso. Foi considerado em todos os casos um arranjo planar de microfones quadrado e com elementos uniformemente espaçados, com $M = 49$ microfones, isto é, $M_x = M_y = 7$ e foi considerada que a visada do arranjo, ou seja, o espaço visto pelo arranjo e suas respectivas direções foram amostradas com passo $k = 0.05$ de -1 a 1 nas direções x e y , ou seja, $N_x = N_y = 41$ e $N = 1681$ direções. O arranjo foi construído em MATLAB com o auxílio de uma função do MATLAB de autoria do aluno, chamada "arranjo.m", cujo código está no Anexo A. Ela faz uso da geometria do arranjo (um arranjo quadrado) para criar um arranjo planar dadas as posições dos microfones presentes ou no eixo x ou no eixo y (o arranjo é simétrico).

Figura 15 - Visada do Arranjo de Microfones e Arranjo de Microfones utilizados no cenário inicial

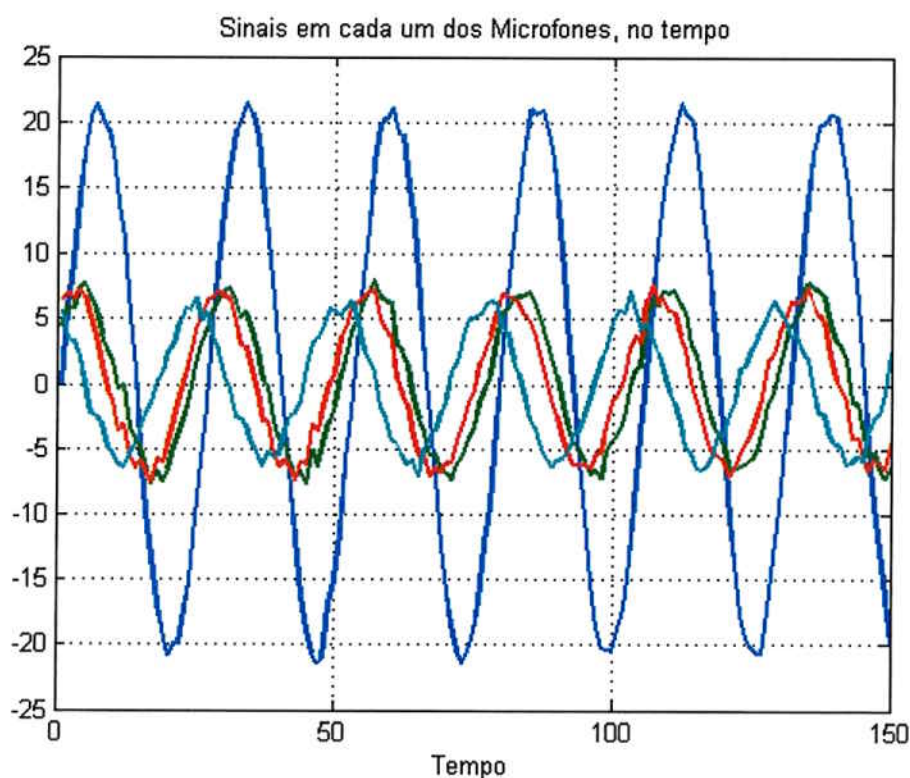


Construído o modelo para as direções vistas pelo arranjo e para o próprio arranjo, calculou-se a matriz de direções V para este caso inicial. Com auxílio de uma função criada no MATLAB, chamada "steer.m", de autoria do próprio aluno, foi possível realizar o cálculo da matriz V , uma vez conhecida sua definição (vide seção 6.6.1). O código desta função está presente no Anexo B. Esta função além de dar a matriz de direções nos fornece, também, dado o passo de amostragem do espaço visto pelo arranjo, todas as coordenadas destas direções.

Passou-se para o cálculo dos sinais nos microfones, utilizados na modelagem do problema a fim de serem utilizados para realizar a estimativa das potências em

cada direção vista pelo arranjo, e das suas respectivas FFTs. Calculadas as FFTs dos sinais, tomou-se o valor das transformadas dos três sinais na frequência analisada, de 5 kHz, portanto, três valores complexos. A seguir, tem-se um exemplo simples, com as mesmas premissas que este cenário inicial, porém com quatro microfones na geometria do arranjo, apenas para se ter uma ideia de como o resultado foi obtido, uma vez que a análise temporal desses valores para um número muito grande de microfones se torna inviável. Nota-se nesse exemplo a defasagem entre os sinais, causada pelo atraso de chegada da frente de onda plana em cada um dos microfones.

Figura 16 - Sinais em cada um dos microfones, no tempo, para um exemplo simples de quatro microfones, sob as mesmas circunstâncias do cenário inicial analisado



Por fim, o código que gera o modelo usa as informações anteriores e funções também já definidas anteriormente para calcular ou a matriz V para o caso que não faz uso da nova KAT ou as matrizes V_x e V_y , utilizadas pela nova KAT no processo de aceleração dos cálculos de estimação. A partir deste ponto, para cada método que foi implementado, há diferenças de implementação, que serão brevemente comentadas nas próximas seções, além das diferenças explícitas entre os códigos que fazem uso da nova KAT e aqueles que não. Os resultados obtidos para cada método estudado estão a seguir.

As simulações foram realizadas em um computador com as seguintes características:

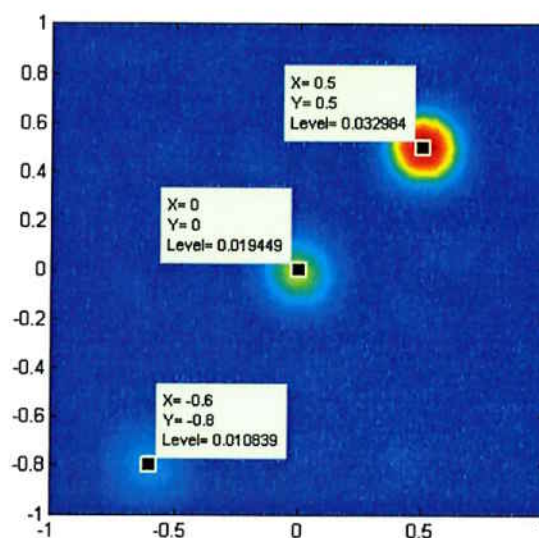
- Sistema Operacional Windows 10 Pro, 64-bit, processador x64-based
- Processador Intel(R) Core(TM)2 Quad CPU Q9550 @2.83GHz
- Memória RAM de 4GB

6.7.2. Resultados para o método *Delay and Sum*

O código gerado para implementar o método *Delay and Sum* está presente no Anexo C, para este cenário inicial. O mesmo código é utilizado para implementar o modelo proposto neste trabalho fazendo uso da nova KAT ou não. Foi utilizada uma outra função, denominada "mapa.m" para auxiliar na geração das imagens acústicas e seu código está presente no Anexo D, sendo que esta função está presente em todos os métodos. A fim de determinar uma estimativa para o tempo de processamento, foram utilizadas as funções *tic* e *toc* do MATLAB e um número de realizações do algoritmo de reconstrução de 250 vezes. As imagens acústicas obtidas pelos dois casos (com e sem a nova KAT) foram as mesmas, diferindo apenas no tempo de processamento, isso para o DAS, para a regularização l_1 e para a regularização TV .

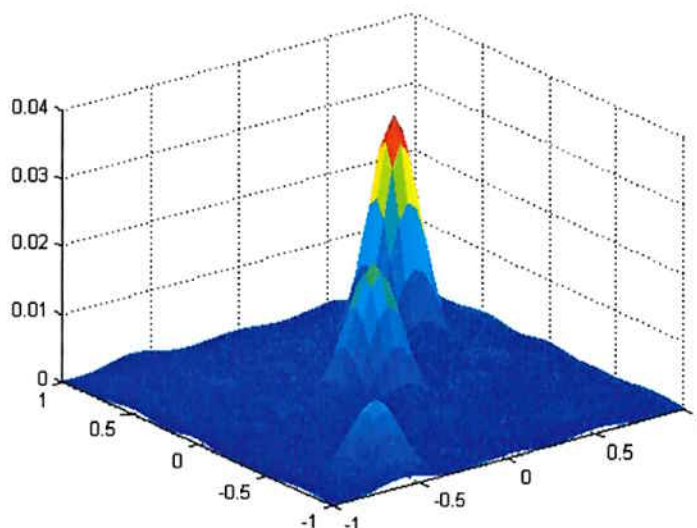
A imagem acústica obtida para este cenário inicial fazendo uso do DAS, sem o uso da nova KAT, foi a seguinte:

Figura 17 - Imagem Acústica estimada pelo DAS para o cenário inicial



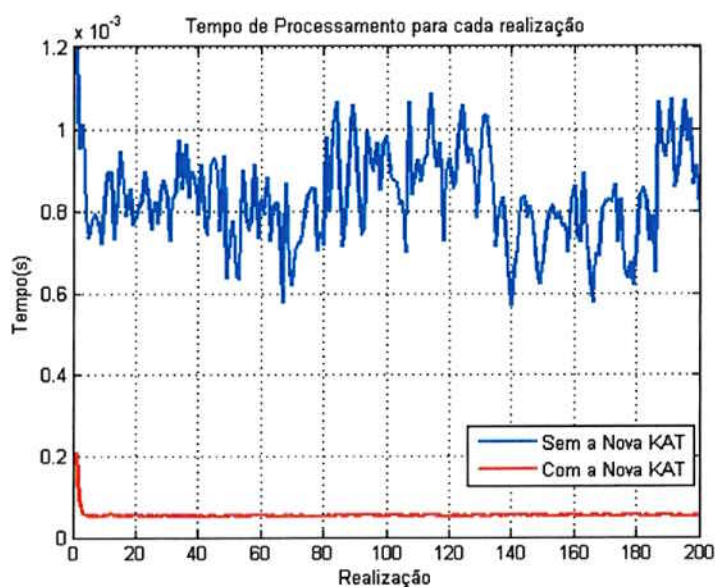
Analisando os picos desta figura, pode-se observar a direções estimadas pelo algoritmo. Observa-se que os picos da imagem coincidem com os pontos esperados, o que evidencia o sucesso da estimação. A imagem acústica em três dimensões observada foi a seguinte:

Figura 18 - Imagem acústica 3D para o DAS no cenário inicial



O tempo de processamento para cada uma das realizações é o presente no gráfico a seguir:

Figura 19 - Tempo de Processamento para cada realização para o DAS, sem e com da nova KAT, no cenário inicial



O valor médio observado para o tempo de processamento sem uso da nova KAT foi de 8.3008×10^{-4} segundos. Agora, para o caso que faz uso da nova KAT, os resultados obtidos para as imagens acústicas foram os mesmos, exceto pelo tempo de processamento, sendo que tempo de processamento médio neste caso foi de 5.3842×10^{-5} segundos. Há um desvio de aproximadamente 96.41%, isto é, a KAT nesse caso torna o algoritmo 93.51% mais rápido.

6.7.3. Resultados para o método Regularização l_1

O código gerado para implementar o método que usa regularização l_1 está presente no Anexo E, onde não se utiliza a nova KAT, e no Anexo F, onde se faz uso da nova KAT, para este cenário inicial. Faz-se uso também de um pacote para o MATLAB, disponível em [7], para solucionar exatamente o problema de regularização l_1 . Mais uma vez, foram utilizadas as funções *tic* e *toc* do MATLAB para se levantar o tempo de processamento da estimação, com um número de realizações igual a 250 vezes. Os resultados obtidos para a imagem acústica reconstruída foram os mesmos para os dois casos, sem e com a KAT, exceto no tempo de processamento. Estes resultados são apresentados a seguir.

Figura 20 - Posição dos picos presentes na imagem acústica obtida para a regularização l_1 no cenário inicial, para os dois casos, com e sem o uso da nova KAT

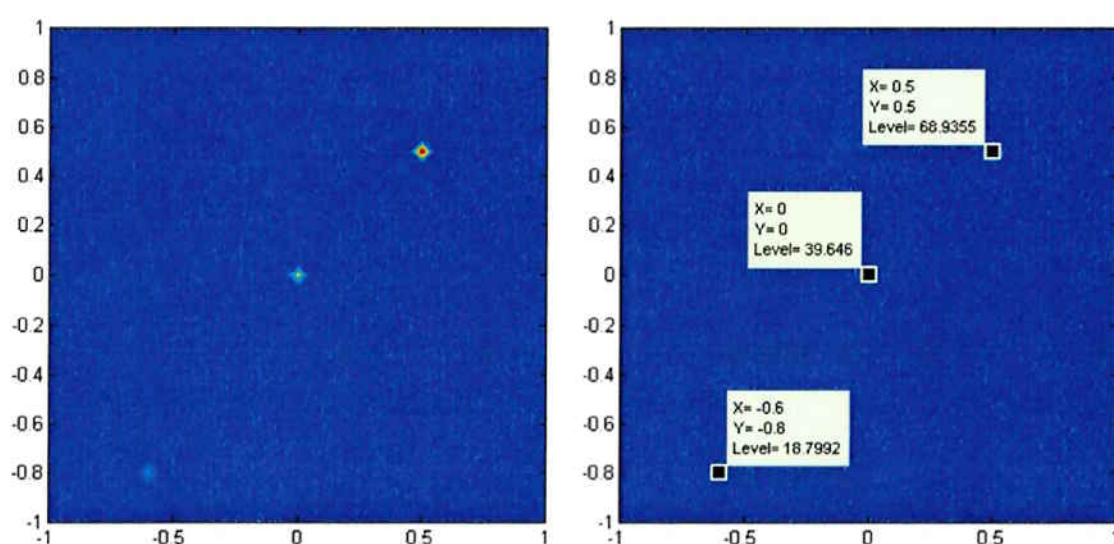
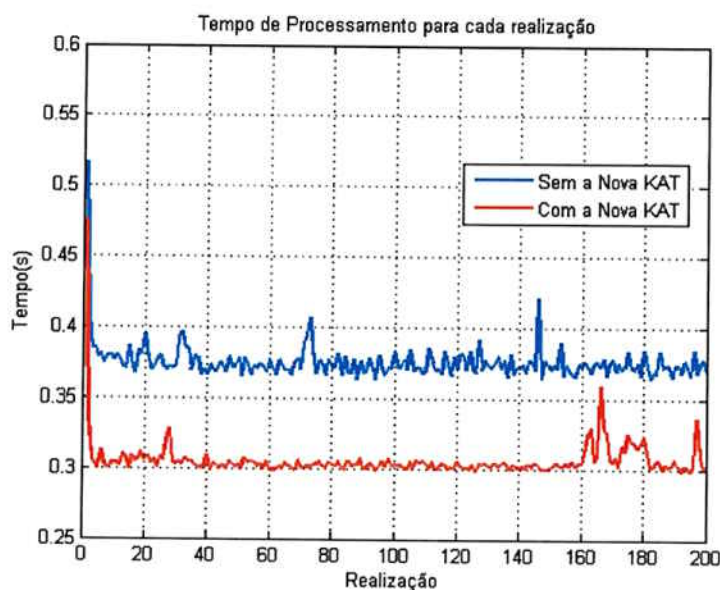


Figura 21 - Tempo de Processamento para cada realização para a regularização l_1 , sem e com uso da nova KAT, no cenário inicial



O tempo de processamento médio quando não se utiliza a nova KAT foi de 0,3751 segundos e quando se utiliza a nova KAT foi de 0,3058 segundos, isto é, o uso da nova KAT torna o processamento 18.48% mais rápido. Alguns testes comparativos serão realizados entre a imagem reconstruída com a KAT proposta neste trabalho e a mesma imagem estimada pela KAT proposta em [2] mais a frente neste trabalho, onde se mostra a qualidade das imagens dado que as fontes são ou não correlacionadas.

6.7.4. Resultados para o método Regularização TV

O código gerado para implementar o método que usa regularização TV está presente no Anexo G, onde não se utiliza a nova KAT, e no Anexo H, onde se faz uso da nova KAT, para este cenário inicial. Faz-se uso também de um pacote para o MATLAB, disponível em [8], para solucionar exatamente o problema de regularização TV . Mais uma vez, foram utilizadas as funções *tic* e *toc* do MATLAB para se levantar o tempo de processamento da estimação, com um número de realizações igual a 250 vezes. Os resultados obtidos para a imagem acústica reconstruída foram os mesmos para os dois casos, sem e com a KAT, exceto no tempo de processamento. Nota-se que as imagens acústicas possuem um erro de estimação na direção das fontes de aproximadamente 0.05 (adimensional) nas direções x e y . Estes resultados são apresentados a seguir.

Figura 22 - Imagem Acústica estimada pela Regularização TV para o cenário inicial, para os dois casos, com e sem o uso da nova KAT

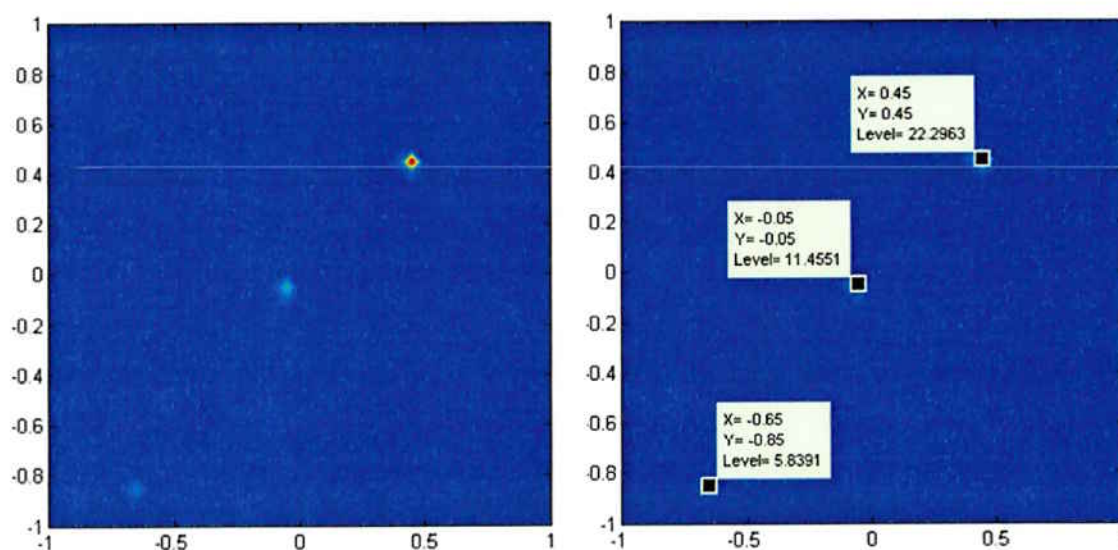
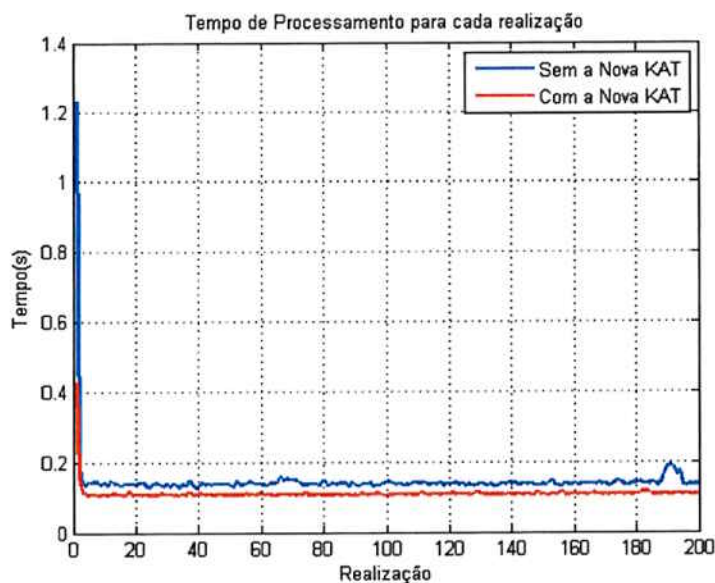


Figura 23 - Tempo de Processamento para cada realização para a regularização TV , sem e com o uso da nova KAT, no cenário inicial



O tempo de processamento médio quando não se utiliza a nova KAT foi de 0,1446 segundos e quando se utiliza a nova KAT foi de 0,1105 segundos, isto é, o uso da nova KAT torna o processamento 23.62% mais rápido. Portanto, apesar do erro de estimação da direção das fontes, nota-se que o uso da nova KAT nesse caso também diminui o tempo de processamento. Neste caso, ainda, a imagem estimada foi exatamente a mesma para os dois casos, sem e com o uso da nova KAT.

Vale notar que a regularização TV não é indicada para o caso do cenário inicial, onde as fontes são esparsas (pontuais), e sim para caso onde as imagens acústicas a serem reconstruídas são mais “suaves” (contínuas).

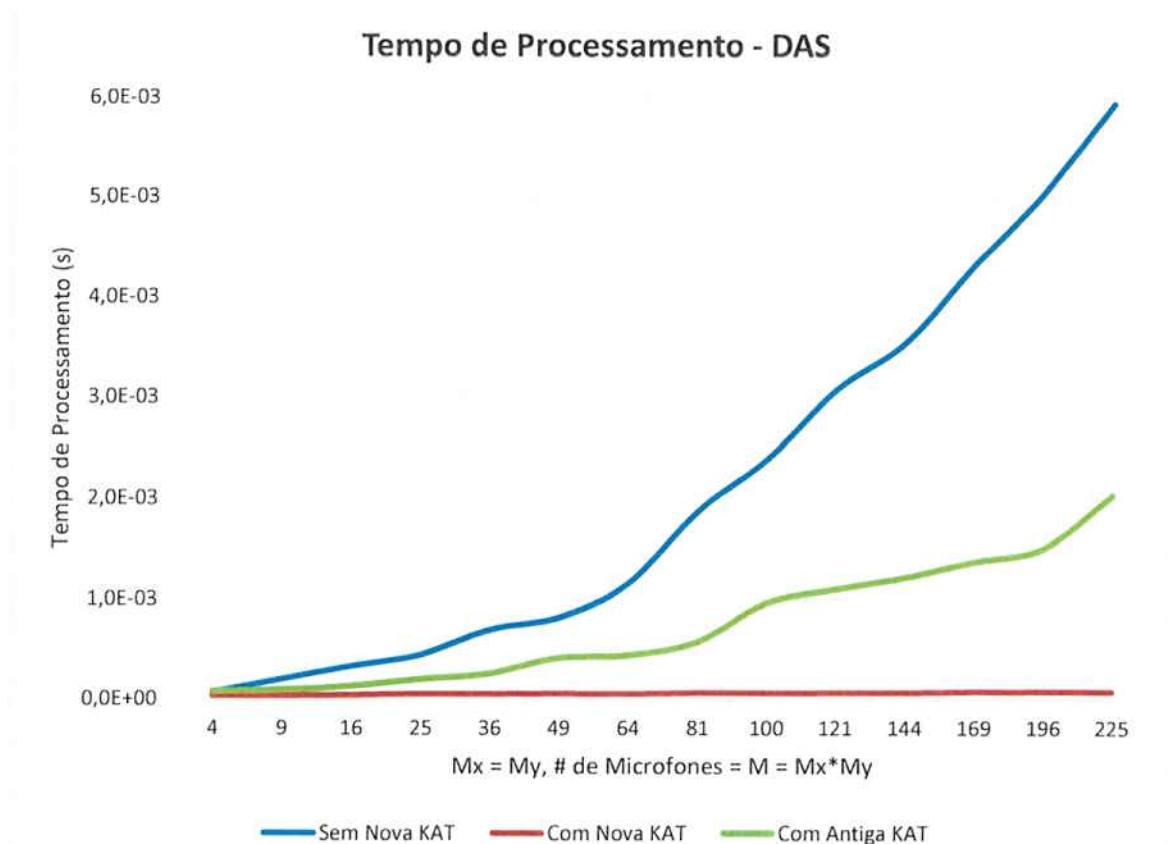
6.8. Tempos de Processamento Experimentais dos Métodos com a Variação do Número de Microfones no Arranjo

Foi feita uma análise experimental sobre o tempo de processamento dos algoritmos anteriormente implementados e ainda dos algoritmos que faziam uso da KAT proposta para o problema cujo modelo fazia uso da matriz de autocorrelação. As comparações que foram realizadas foram feitas em relação aos mesmos métodos, quando se fazia uso ou não da nova KAT e quando se utilizava a KAT que fazia uso da matriz de autocorrelação, variando-se o número de microfones no arranjo de microfones. Variou-se de $M_x = M_y = 2$ ($M = 4$ microfones) até $M_x = M_y = 15$ ($M = 225$ microfones).

6.8.1. Tempo de Processamento para o DAS

Os resultados experimentalmente obtidos para o tempo de processamento quando se utilizou o método DAS foi o observado na figura a seguir.

Figura 24 - Tempo de Processamento para o Delay and Sum, variando-se o número de microfones no arranjo



Os valores obtidos para cada configuração do arranjo e as discrepâncias entre os tempos de processamento sem e com o uso da nova KAT e da antiga KAT são os que se seguem:

Tabela 4 - Tempo de Processamento para cada configuração do arranjo, para o método DAS

Número de Microfones no Arranjo	Sem Nova KAT	Com Nova KAT	Aceleração de:	Com Antiga KAT	Sem Nova KAT x Com Antiga KAT	Com Nova KAT x Com Antiga KAT
4	6,7056E-05	2,9990E-05	55,28%	7,5296E-05	10,94%	60,17%
9	1,9959E-04	3,4651E-05	82,64%	8,9924E-05	-121,96%	61,47%
16	3,3083E-04	4,0103E-05	87,88%	1,2979E-04	-154,91%	69,10%
25	4,4594E-04	5,2093E-05	88,32%	1,9846E-04	-124,70%	73,75%
36	6,9697E-04	5,2308E-05	92,49%	2,5700E-04	-171,19%	79,65%
49	8,1585E-04	5,5805E-05	93,16%	4,1324E-04	-97,43%	86,50%
64	1,1601E-03	5,3999E-05	95,35%	4,4012E-04	-163,59%	87,73%
81	1,8756E-03	6,2467E-05	96,67%	5,7552E-04	-225,90%	89,15%
100	2,3832E-03	6,5259E-05	97,26%	9,6072E-04	-148,06%	93,21%
121	3,0711E-03	6,5579E-05	97,86%	1,1007E-03	-179,00%	94,04%
144	3,5403E-03	6,7453E-05	98,09%	1,2145E-03	-191,50%	94,45%
169	4,3221E-03	7,6579E-05	98,23%	1,3723E-03	-214,95%	94,42%
196	5,0416E-03	7,9461E-05	98,42%	1,5037E-03	-235,27%	94,72%
225	5,9404E-03	8,0309E-05	98,65%	2,0318E-03	-192,37%	96,05%

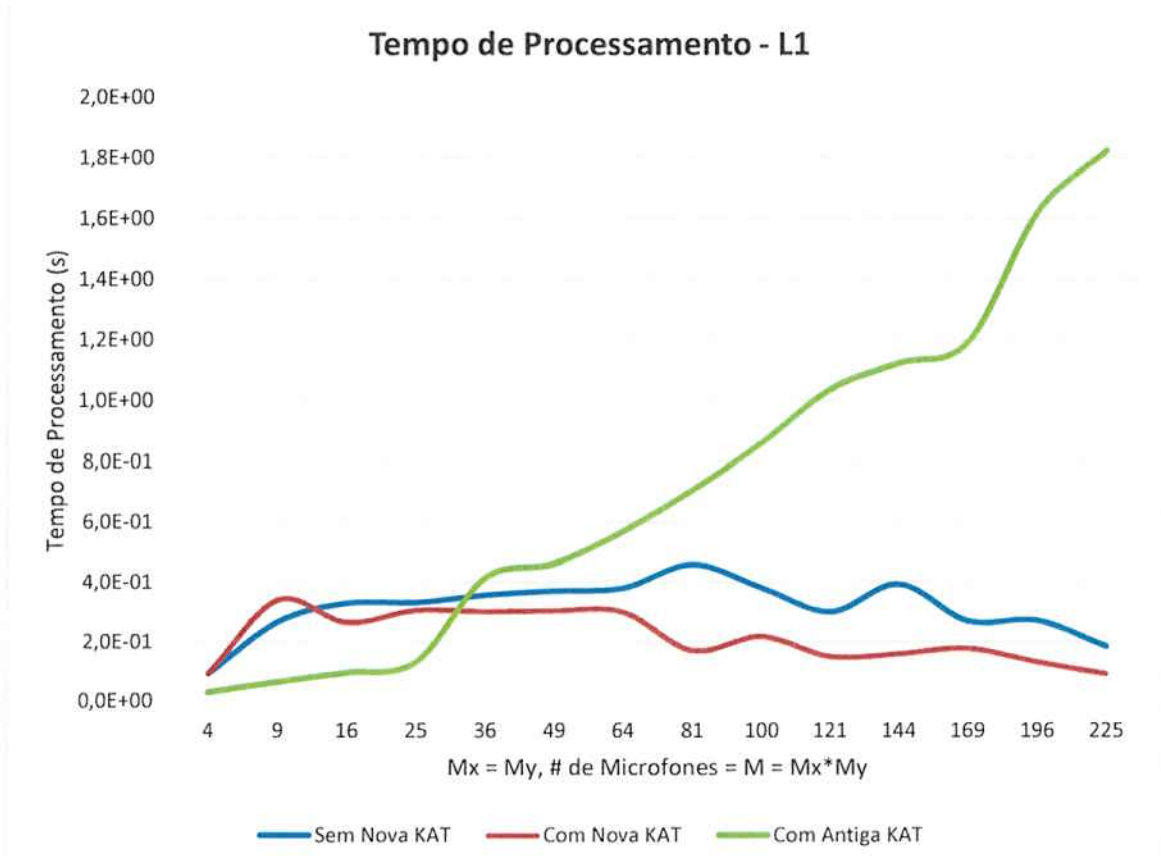
É notável o ganho se tem quando se utiliza a nova transformada rápida, visto no aumento significativo da aceleração obtido pela diferença entre os resultados quando se usa e quando não se usa a nova KAT. Nota-se também que o tempo de processamento aumenta, em ambos os casos, conforme se aumenta o número de microfones presentes no arranjo, assim como se aumenta a discrepância entre os valores quando se faz uso da nova KAT e quando não se faz uso desta.

Comparando o modelo desenvolvido no presente projeto, ainda sem usar a nova KAT, com o modelo apresentado em [2], já fazendo uso da antiga KAT, nota-se que, quando não se usa a nova KAT no modelo atual, o modelo antigo ganha em tempo de processamento, porém ao utilizá-la, tem-se ganhos crescentes de aceleração com o aumento da complexidade do arranjo, proporcional ao número de microfones presente no mesmo.

6.8.2. Tempo de Processamento para a Regularização l_1

Os tempos de processamento experimentalmente obtidos quando se variava a configuração do arranjo, isto é, quando o número de microfones presentes no arranjo variava, para o método regularização l_1 estão presentes na figura a seguir.

Figura 25 - Tempo de Processamento para a Regularização l_1 , variando-se o número de microfones no arranjo



Nota-se que o tempo de processamento quando se faz uso da nova KAT é na maioria das vezes menor que da situação onde esta transformada não é utilizada. Porém, o tempo de processamento em ambos os casos tem comportamento crescente até aproximadamente 49 microfones, quando volta a cair. Os tempos obtidos neste experimento são os seguintes:

Tabela 5 - Tempo de Processamento para cada configuração do arranjo, para o método Regularização l1

Número de Microfones no Arranjo	Sem Nova KAT	Com Nova KAT	Aceleração de:	Com Antiga KAT	Sem Nova KAT x Com Antiga KAT	Com Nova KAT x Com Antiga KAT
4	9,1307E-02	9,2508E-02	-1,32%	2,8716E-02	-217,97%	-222,15%
9	2,6604E-01	3,4228E-01	-28,66%	6,3535E-02	-318,72%	-438,73%
16	3,2886E-01	2,6706E-01	18,79%	9,4609E-02	-247,60%	-182,27%
25	3,3055E-01	3,0704E-01	7,11%	1,3020E-01	-153,87%	-135,82%
36	3,5512E-01	3,0254E-01	14,81%	4,1263E-01	13,94%	26,68%
49	3,6940E-01	3,0594E-01	19,18%	4,6001E-01	19,70%	33,49%
64	3,7853E-01	3,0157E-01	18,36%	5,6775E-01	33,33%	46,88%
81	4,5698E-01	1,7124E-01	62,53%	7,0501E-01	35,18%	75,71%
100	3,7936E-01	2,1922E-01	42,21%	8,5953E-01	55,86%	74,50%
121	2,9957E-01	1,5181E-01	49,33%	1,0350E+00	71,06%	85,33%
144	3,9169E-01	1,6044E-01	59,04%	1,1219E+00	65,09%	85,70%
169	2,6853E-01	1,7832E-01	33,59%	1,1909E+00	77,45%	85,03%
196	2,7052E-01	1,3301E-01	50,83%	1,6186E+00	83,29%	91,78%
225	1,8314E-01	9,2136E-02	49,69%	1,8218E+00	89,95%	94,94%

Comparando os casos do novo modelo, sem e com o uso da nova transformada rápida, nota-se que o tempo de processamento cresce até certo ponto a partir do qual ele volta a cair. A explicação mais plausível está na maneira como o método l1 faz sua minimização. O pacote utilizado, que implementa o método l1, o SPGL1 [7], possui critérios de parada, tal como menor erro aceitável nas estimativas ou número máximo de iterações. Quando o arranjo possui poucos microfones, a tendência é que o algoritmo pare pelo critério de número de iterações, o que faz ele demorar cada vez mais tempo com o aumento da complexidade do arranjo, pois ele utiliza todas as iterações, e com o aumento da complexidade, cada iteração leva mais tempo. A partir de certo ponto, com um número maior de microfones, o algoritmo termina sua estimação pelo critério do erro mínimo, pois ele encontra uma solução, e não utiliza todas as iterações, o que diminui o tempo de processamento observado. Essa hipótese precisa ser melhor estudada, mas é totalmente plausível de ser a responsável pelos resultados obtidos anteriormente.

Para arranjos com entre 4 ou 36 microfones, o tempo de processamento quando se utiliza a antiga transformada é menor que quando se utiliza o modelo sem autocorrelação, sem e com a nova transformada rápida. A partir de 36 microfones, nota-se que o novo modelo, tanto sem e com a nova transformada, possui tempo de

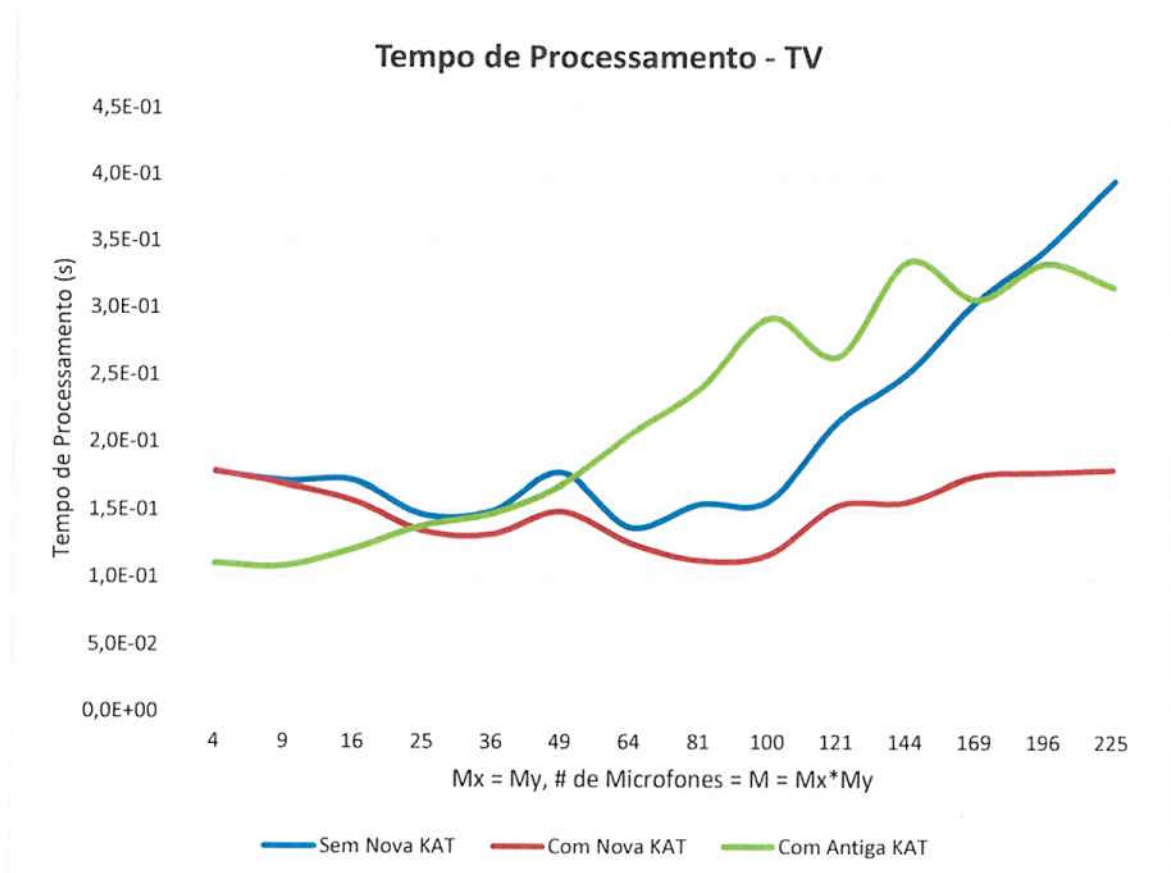
processamento menor, ponto a partir do qual os ganhos de aceleração do novo modelo frente à antiga KAT aumentam gradativamente.

Portanto, tem-se que, comparando o novo modelo sem e com o uso da nova transformada, os ganhos do uso da nova KAT surgem a partir de 16 microfones, e comparando o novo modelo usando a nova KAT com o modelo anterior, usando a antiga KAT, tem-se que a nova abordagem se torna vantajosa a partir de 36 microfones no arranjo.

6.8.3. Tempo de Processamento para a Regularização *TV*

As curvas experimentais para o tempo de processamento na reconstrução da imagem acústicas utilizando a regularização *TV* pelo número de microfones no arranjo de sensores, sem se fazer uso da nova KAT e fazendo uso desta e ainda da antiga KAT, estão apresentadas no gráfico a seguir.

Figura 26 - Tempo de Processamento para a Regularização *TV*, variando-se o número de microfones no arranjo



Nota-se que para quantidade de microfones pequenas quase não há distinção entre o uso ou não da nova transformada rápida, para o modelo sem autocorrelação, porém com o aumento da complexidade do arranjo, verifica-se que a discrepância entre o tempo de processamento sem e com o uso da nova KAT aumenta cada vez mais. Os valores obtidos foram os seguintes:

Tabela 6 - Tempo de Processamento para cada configuração do arranjo, para o método Regularização TV

Número de Microfones no Arranjo	Sem Nova KAT	Com Nova KAT	Aceleração de:	Com Antiga KAT	Sem Nova KAT x Com Antiga KAT	Com Nova KAT x Com Antiga KAT
4	1,7775E-01	1,7815E-01	-0,22%	1,1040E-01	-61,01%	-61,37%
9	1,7133E-01	1,6891E-01	1,42%	1,0867E-01	-57,66%	-55,43%
16	1,7177E-01	1,5707E-01	8,56%	1,2143E-01	-41,46%	-29,35%
25	1,4688E-01	1,3476E-01	8,25%	1,3870E-01	-5,90%	2,84%
36	1,4945E-01	1,3225E-01	11,51%	1,4755E-01	-1,29%	10,37%
49	1,7786E-01	1,4923E-01	16,09%	1,6845E-01	-5,58%	11,41%
64	1,3763E-01	1,2628E-01	8,25%	2,0615E-01	33,24%	38,74%
81	1,5472E-01	1,1311E-01	26,89%	2,4052E-01	35,67%	52,97%
100	1,5708E-01	1,1744E-01	25,23%	2,9281E-01	46,36%	59,89%
121	2,1595E-01	1,5366E-01	28,84%	2,6443E-01	18,33%	41,89%
144	2,5175E-01	1,5648E-01	37,84%	3,3547E-01	24,96%	53,36%
169	3,0588E-01	1,7548E-01	42,63%	3,0726E-01	0,45%	42,89%
196	3,4513E-01	1,7842E-01	48,30%	3,3415E-01	-3,29%	46,60%
225	3,9653E-01	1,8057E-01	54,46%	3,1680E-01	-25,17%	43,00%

Assim como o DAS, nota-se que com o aumento do número de microfones do arranjo, maior a discrepância entre os valores obtidos não usando e usando a nova KAT. Até o valor de 100 microfones, nota-se uma flutuação nos valores obtidos, mas a partir desse ponto, os tempos de processamento passam a aumentar e a divergir cada vez mais.

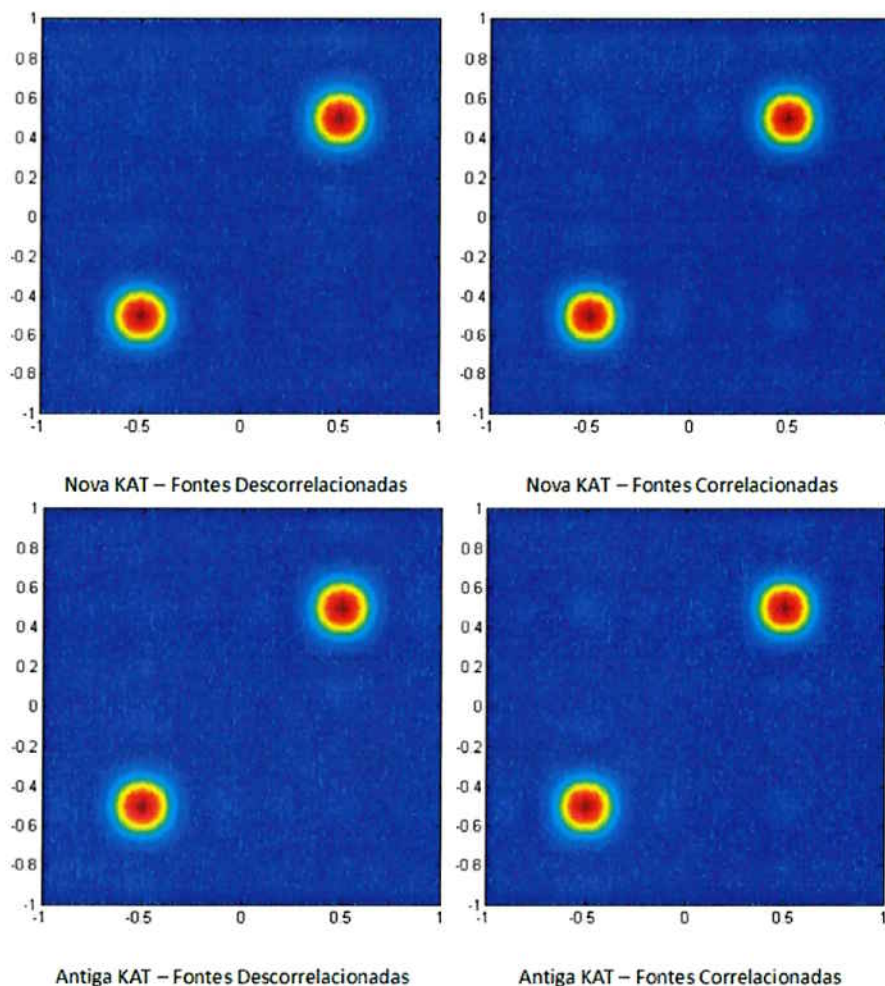
Quando comparada à antiga KAT, comparando o uso da nova KAT com a antiga KAT, tem-se que, a partir de 25 microfones, a nova KAT possui um tempo de processamento menor, ponto a partir do qual os ganhos de aceleração obtidos com o uso da nova KAT se mantêm em torno de 40%, na média.

Portanto, o uso da nova KAT frente a sua não utilização, para o modelo sem autocorrelação, é recomendado em quase todos os casos, e é recomendado a partir de 25 microfones frente à antiga KAT, cujo modelo utilizava autocorrelação.

6.9. Uso de Fontes Correlacionadas e Não Correlacionadas

Para se realizar comparações entre a nova transformada e a antiga transformada, criou-se um cenário comparativo semelhante ao da seção anterior, porém mudando o número de fontes para 2 fontes e as direções destas para, respectivamente, $\mathbf{u}_{f1} = [-0.5 \ -0.5 \ 0.7071]^T$ e $\mathbf{u}_{f2} = [0.5 \ 0.5 \ 0.7071]^T$. Foram comparados os métodos de reconstrução anteriormente analisados (DAS, l1 e TV) para os casos onde não havia uso da autocorrelação e se utilizava a nova transformada, e onde havia o uso da autocorrelação e se utilizava a antiga transformada rápida. Os resultados para o *Delay and Sum* foram os seguintes:

Figura 27 - Fontes Correlacionadas e Descorrelacionadas, com a nova e antiga KAT, utilizando o método Delay and Sum



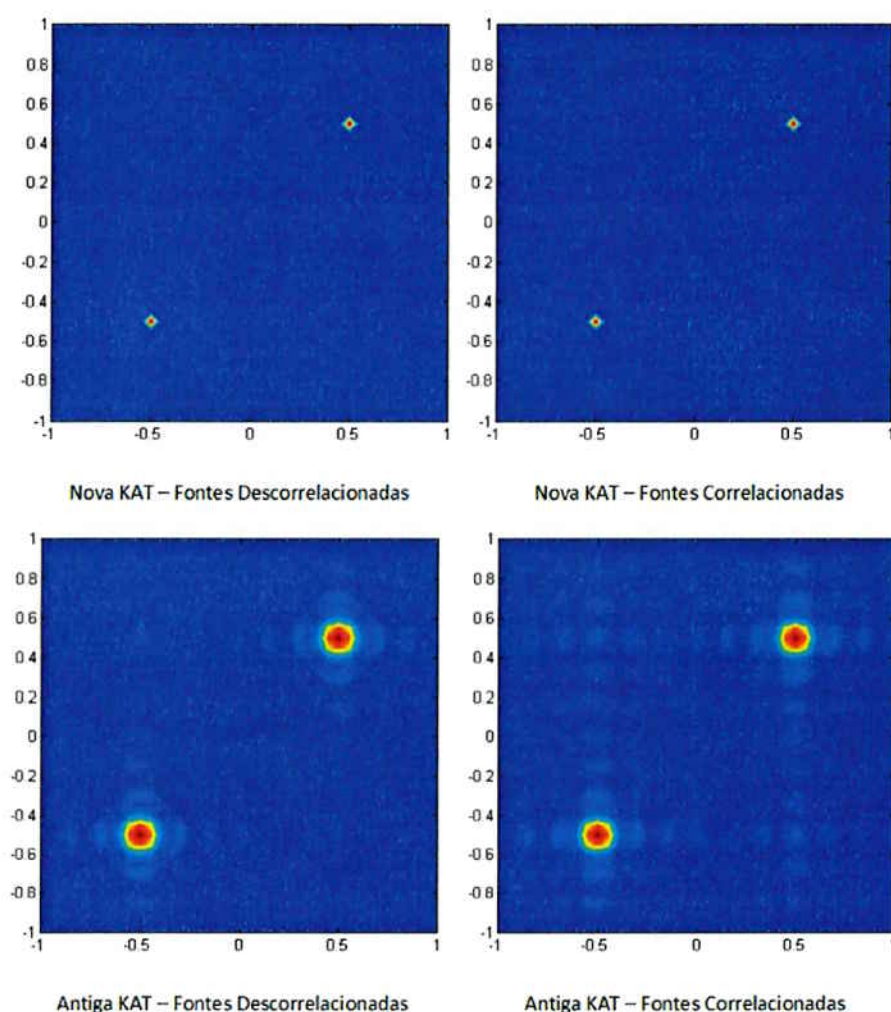
As imagens, com a nova e a antiga transformada, são bem semelhantes, mas o mais notável é a presença de mais pontos não nulos nas imagens cujas fontes são

correlacionadas. Quanto à qualidade das imagens, percebe-se que as imagens são praticamente iguais.

Deve-se acrescentar que, sobre o método *DAS*, tanto para o modelo presente em [2] como para o modelo aqui proposto, o resultado que a ser obtido deveria ser idêntico, como foi de fato observado. Isso ocorre pois, uma vez que o método não é iterativo, o único cálculo que este realiza para reconstruir a imagem foi construído matematicamente para que, nas duas maneiras apresentadas, o resultado fosse o mesmo.

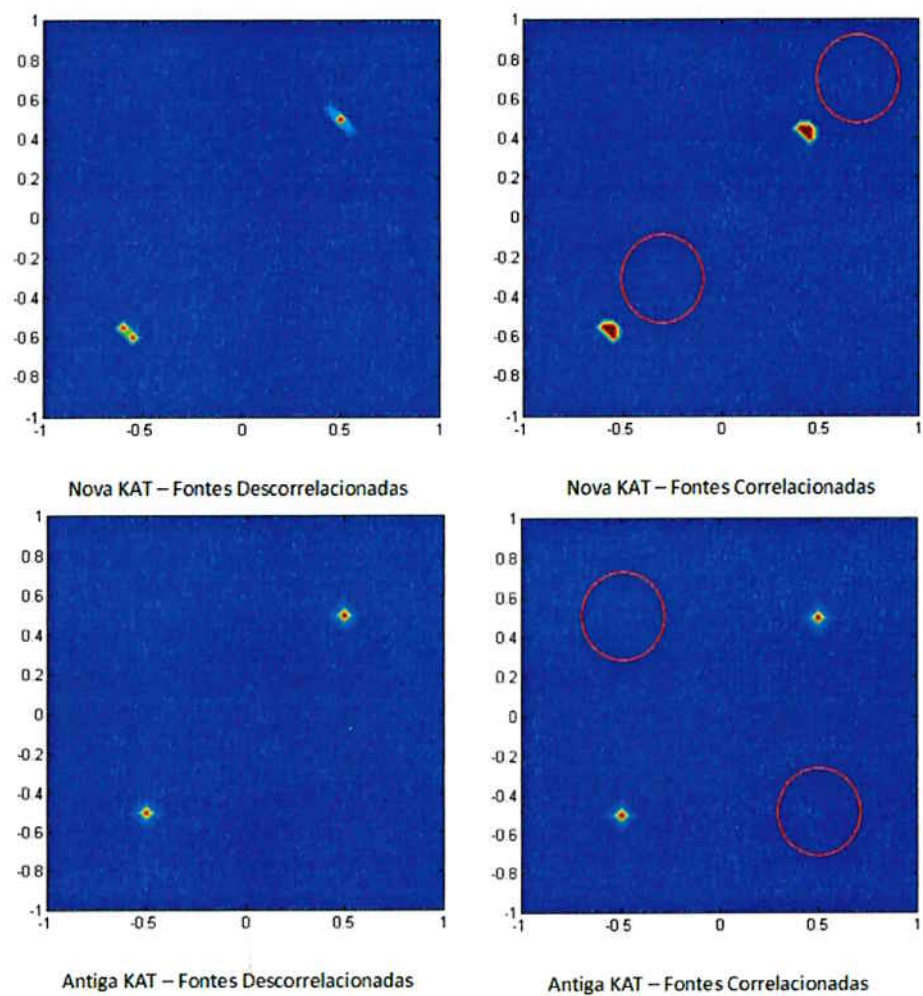
Fazendo as mesmas simulações para o método *l1*, sob as mesmas condições, tem-se os seguintes resultados, presentes na Figura 28. Aqui neste caso, um dos efeitos esperados para a nova transformada é facilmente notado: quando se usa a nova transformada, não se nota a presença de pontos não nulos, a não ser os próximos das direções estimadas das fontes, porém, com a antiga transformada, nota-se que tanto no caso com fontes descorrelacionadas como com fontes correlacionadas, há a presença desses pontos não nulos, sendo que a imagem estimada utilizando fontes correlacionadas o resultado piora ainda mais. Isso está diretamente relacionado ao fato de se utilizar um modelo onde parte dos dados da matriz de autocorrelação da transformada antiga é ignorado, lembrando que é feita a suposição de que esta matriz é diagonal, quando não o é.

Figura 28 - Fontes Correlacionadas e Descorrelacionadas, com a nova e antiga KAT, utilizando o método l1



Os resultados para o método *TV* estão presentes na Figura 29. A análise feita para este método mostra que a imagem obtida pelo método *TV* através do modelo proposto aqui neste trabalho possui uma piora em relação à imagem obtida pela antiga transformada rápida, e o uso de fontes correlacionadas piora a qualidade da imagem para o método atual. Mesmo o método antigo apresentando pontos não nulos, sua qualidade é melhor. Vale ressaltar que este cenário não é o ideal para realizar a comparação, pois teriam de ser gerados sinais com imagens acústicas “suaves” para que o método *TV* funcione para aquilo que ele foi projetado.

Figura 29 - Fontes Correlacionadas e Descorrelacionadas, com a nova e antiga KAT, utilizando o método TV



7. Verificação de Requisitos, Considerações Finais e Conclusões

Para que se possa realizar uma melhor comparação dos resultados obtidos neste trabalho, a seguir lembra-se os requisitos desejados para este projeto:

- O algoritmo desenvolvido, auxiliado pelo uso da transformada rápida de aceleração dos cálculos de estimação, deveria ser o menos complexo possível, comparado com o algoritmo proposto em [2].
- O tempo de processamento do algoritmo deveria ser menor ou igual ao tempo de processamento observado quando se utiliza o algoritmo proposto em [2].
- A qualidade e a resolução das imagens acústicas obtidas pelo algoritmo aqui implementado deveria ser melhor ou igual ao do algoritmo proposto em [2].
- O algoritmo deveria ser implementado de maneira tal que este utilizasse o menor espaço de memória possível, fazendo com que sua implementação em *hardware* fosse a mais simples possível.

O uso excessivo de memória, aliada à complexidade de implementação, fazem com que esta seja complicada para o algoritmo proposto em [2], uma vez que existem cálculos que são iterativos neste algoritmo, além de envolver operações matemáticas com matrizes de dimensões muito grandes, o que retarda o tempo de processamento em muito, problema que já foi mencionado anteriormente neste trabalho.

Comparando o modelo de estimação proposto neste trabalho, que não faz uso de matrizes de autocorrelação para se estimar potência vinda de cada direção do espaço, aliada ao uso de uma nova transformada rápida que acelera os cálculos de estimação feitos para se reconstruir a imagem acústica, com o modelo proposto em trabalhos anteriores, principalmente o referido em [2], podemos chegar às seguintes comparações de desempenho:

A eliminação da matriz de autocorrelação proposta neste projeto foi possível e trouxe mudanças no modelo que possibilitaram a melhora no tempo de processamento, observadas principalmente na seção onde foram levantadas as curvas para os tempos de processamento para os métodos *DAS*, *I1* e *TV*. Comparando apenas os dois modelos com o uso de suas respectivas transformadas rápidas, aqui chamadas de antiga *KAT* e nova *KAT*, observa-se que o novo modelo melhora substancialmente o tempo de processamento para o método de reconstrução *DAS*, e

tanto para o $l1$ como para o TV , a partir de determinado número de microfones no arranjo utilizado, observa-se um ganho de processamento acentuado. A melhor hipótese para o fato de que o método $l1$ e TV não ganharem sempre no tempo de processamento no novo modelo está exatamente na mudança que a eliminação da matriz de autocorrelação incorreu ao modelo: a matriz de direções V , para o modelo anterior, possuía dimensões $M^2 \times N$, sendo que no novo modelo tem-se que a mesma matriz tem dimensões $M \times N$, sendo este o principal catalisador para a melhora no tempo de processamento, assim como para diminuir a complexidade do algoritmo e conseqüente menor uso de memória. Porém, como o algoritmo no novo modelo estima um valor complexo, ligado à FFT do sinal das fontes, e não sua potência em si, como fazia o algoritmo anterior, temos que o novo algoritmo passa a estimar $2N$ variáveis, e não mais N , pois agora o algoritmo tem que estimar tanto a parte real como a parte imaginária do sinal vindo de cada direção. Apesar do ganho na matriz de direções, para arranjos pequenos, com poucos microfones, tem-se que este ganho é suplantado pelo número maior de variáveis que devem ser estimadas. Conforme se aumenta a complexidade do arranjo de microfones, observa-se que esse viés deixa de existir. Portanto, pode-se afirmar que a melhoria no tempo de processamento foi observada com sucesso satisfatório.

Quanto à complexidade de implementação, além das melhorias citadas anteriormente, tem-se que a implementação do algoritmo anteriormente proposto, em sua formulação, exigia uma permutação de um vetor para se utilizar a transformada rápida respectiva, como foi mostrada neste trabalho em sua devida seção, e como já foi mencionado, a nova transformada rápida não precisa de tal permutação, fazendo com que os cálculos sejam feitos de maneira direta. Isso, somado à eliminação da necessidade de estimar a matriz de autocorrelação e mais a redução de elementos na matriz de direções V , diminuem o número de blocos que devem ser implementados em *hardware*, o que a torna bem mais simples. Portanto, quanto à complexidade, conseguiu-se propor um algoritmo mais simples que o do modelo proposto em [2].

A qualidade da imagem é igual quando se usa o método DAS , bem melhor quando se usa a regularização $l1$ e pior para a regularização TV , mas a melhor explicação para esta piora está no fato de que o cenário utilizado para se comparar o desempenho do método antigo e o método antigo não corresponde a um exemplo

para o qual a regularização TV foi criado. Como foi explicado na seção onde se explicou os métodos de reconstrução e recorrentemente em outras partes desse projeto, a regularização TV é recomendado para imagens acústicas “suaves”, isto é, as fontes geradoras não são pontuais no espaço e geram um sinal senoidal e única frequência, cuja a imagem acústica ideal seria um ponto, exatamente o caso criado no cenário de teste. Seria necessário gerar uma imagem suave e comparar devidamente os dois métodos. Salvo isso, o tempo de processamento é menor a partir de certo ponto, como já mencionado. Portanto, quanto à qualidade e resolução da imagem, pode-se afirmar que conseguiu-se propor um algoritmo tão eficiente quanto o do modelo anterior.

Levando em conta os riscos levantados para cada parte do trabalho, nota-se que o surgimento de algum problema com os números complexos se fez verdade, como mencionando anteriormente, mas quanto à sensibilidade em relação aos parâmetros do programa em MATLAB que implementa o algoritmo do novo modelo, não foram observados problemas diretamente relacionados a isso, assim como não se notou problemas relacionados à possibilidade de o algoritmo não convergir e necessitar de muitas iterações para tanto.

Mas uma das mais importantes possibilidades que novo modelo gerou foi a possibilidade de usar uma janela de dados das imagens acústicas numa dada direção e conseguir se estimar o sinal nesta direção, isto é, da maneira como foi proposto e implementado o algoritmo, é possível propor uma anti-transformação capaz estimar o sinal no tempo (ou na frequência) vindo de cada direção, coisa que não era possível com o modelo antigo, pois este eliminava o sinal de fase das correlações cruzadas na matriz de autocorrelação, quando com o novo modelo todos os dados são utilizados. Essa ideia não foi desenvolvida neste trabalho, mas criar o método que resolve o problema oposto a este trabalho, de estimar o sinal vindo de qualquer direção dado uma série de dados das imagens acústicas dessa direção, se torna possível e interessante, e pode vir a ser explorada em trabalhos futuros e posteriores a esse projeto. Além disso, dados os resultados positivos desse trabalho, porém a falta de tempo, poderá ser possível se implementar de fato, em um momento posterior ao fim desse projeto, este algoritmo em uma FPGA, para se verificar o funcionamento desse trabalho em *hardware*.

Pode-se concluir, portanto, que, dentro dos objetivos desse trabalho de conclusão de curso, que eram os de criar um algoritmo que realizasse a estimação das imagens acústicas sem fazer uso de matrizes de autocorrelação, porém de maneira tão eficiente e com qualidade igual ou melhor que algoritmos presentes na literatura, além de ter simples implementação em *hardware*, dados os ganhos de tempo de processamento, do baixo custo computacional e do menor uso de memória em relação aos algoritmos presentes na literatura, além dos ganhos de dimensionalidade das matrizes presentes no algoritmo, o projeto apresentou resultados satisfatórios, atendendo todos os requisitos e pode-se considerar que sua realização foi dentro do esperado.

7.1. Gerenciamento de Projeto

Nesta seção são discutidas as formas de gerenciamento do projeto, que se deram através do cronograma e suas atualizações, além da organização das etapas e atividades do projeto por meio de um diagrama de blocos, e da análise dos custos de projeto.

7.1.1. Cronograma do Projeto – Diagrama de Gantt

Este cronograma feito com auxílio do programa Microsoft® Excel 2013.

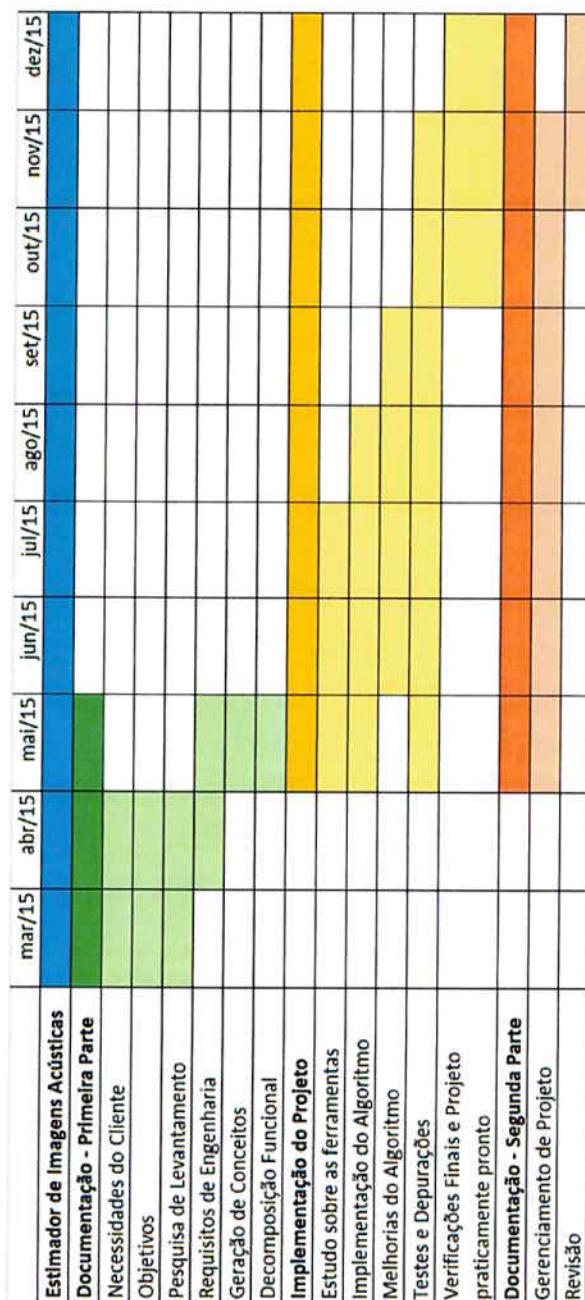


Figura 30 - Diagrama de Gantt do Projeto

7.1.2. Etapas e Atividades do Projeto (EAP)

Este diagrama foi gerado tendo como base o cronograma anterior, e foi feito com auxílio do programa Microsoft® Project 2013.

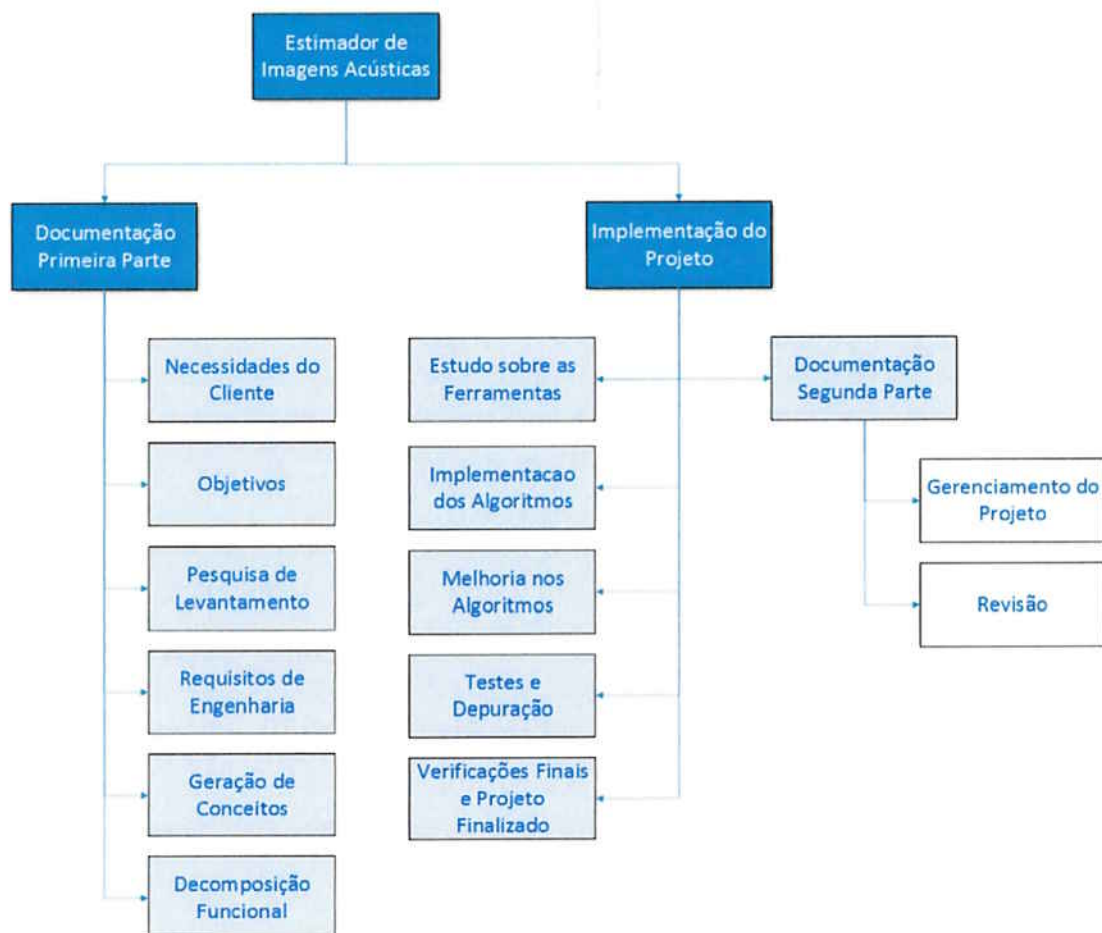


Figura 31 - Etapas e Atividades do Projeto

7.1.3. Custos de Projeto

Baseando-se no cronograma anteriormente apresentado, e sabendo que não serão utilizados outros materiais que não estejam já disponíveis e que o MATLAB® está disponível para o aluno para o desenvolvimento deste projeto, pode-se estimar o quanto este projeto vai custar ao aluno, tendo como base de medida de tal custo a unidade de horas/homem trabalhadas.

Figura 32 - Custos de Projeto (em horas/homem trabalhadas)

Estimativa de dias para confecção do Projeto	199 dias
Se forem usadas 18h/semana para o projeto	522 horas/homem trabalhadas

8. Bibliografia

- [1] C. B. Arroyo, "Estimação de Imagens Acústicas com Arranjos de Microfones," São Paulo, 2015.
- [2] V. H. Nascimento, F. P. Ribeiro e B. Mansiero, "Acoustic Imaging Using the Kronecker Array," pp. 1-28, 10 Fevereiro 2015.
- [3] "Fast Transforms for Acoustic Imaging— Part I: Theory," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, p. 12, 08 August 2011.
- [4] H. L. Van Trees, Optimum Array Processing - Detection, Estimation, and Modulation Theory - Part IV, New York: Wiley, 2002.
- [5] F. P. Ribeiro, "Arrays de Microfones para Medidas de Campos Acústicos," São Paulo, 2012.
- [6] A. Xenaki, P. Gerstoft e K. Mosegaard, "Compressive beamforming," *Acoustical Society of America*, p. 260–271, 21 Maio 2014.
- [7] UCDavis, "SPGL1: A solver for large-scale sparse reconstruction," 29 Abril 2015. [Online]. Available: <https://www.math.ucdavis.edu/~mpf/spgl1/>. [Acesso em Setembro 2015].
- [8] R. University, "TVAL3 Home - Rice University," 11 Junho 2013. [Online]. Available: <http://www.caam.rice.edu/~optimization/L1/TVAL3/>. [Acesso em Setembro 2015].
- [9] F. P. Ribeiro e V. H. Nascimento, "Fast Transforms for Acoustic Imaging - Part II: Applications," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, p. 7, 08 August 2011.
- [10] t. f. E. Wikipedia, "Field-programmable gate array," [Online]. Available: http://en.wikipedia.org/wiki/Field-programmable_gate_array. [Acesso em 07 Abril 2015].

[11] t. f. E. Wikipedia, "Regularization," [Online]. Available: http://en.wikipedia.org/wiki/Regularization_%28mathematics%29. [Acesso em 08 Abril 2015].

ANEXO A – Função “arranjo.m”, que auxilia na modelagem do arranjo planar

```
function [p] = arranjo(pos_x)

    pos_x = pos_x';
    M = length(pos_x)^2;

    for i = 1:sqrt(M)

        y(:,i) = ones(sqrt(M),1)*pos_x(i);

    end

    y = reshape(y,M,1);

    x = [];

    for i = 1:sqrt(M)

        x = [x; pos_x];

    end

    p = [x y zeros(M,1)];

end
```

ANEXO B – Função “steer.m”, que calcula a matriz de direções V

```
% Array Manifold Vector or Steering Vector
function [v,u] = steer(f,p,k,u)

c = 340.29; %m/s : velocidade do som
w = 2*pi*f;

if nargin == 4

    p = p';
    v = exp(1i*u*p*(w/c));

else

    figure()
    plot(p(:,1),p(:,2),'ob','linewidth',2)
    title('Arranjo Utilizado')
    grid

    x = -1:k:1;
    y = -1:k:1;

    for i = 1:length(x)
        for j = 1:length(y)

            if (x(i)^2 + y(j)^2 <= 1)

                uz(i,j) = real(sqrt(1 - x(i).^2 - y(j).^2));

            else

                uz(i,j) = 0;

            end

            ux(i,j) = x(i);
            uy(i,j) = y(j);

        end
    end

    mapa(x,y,uz);

    ux = ux'; ux = ux(:);
    uy = uy'; uy = uy(:);
    uz = uz'; uz = uz(:);

    u = [ux uy uz];

    p = p';

    v = exp(1i*u*p*(w/c));

end
end
```

ANEXO C – Código para o cenário inicial para o Delay and Sum

```
close all, clear all

%% Constantes

Fs = 2^17; % Frequência de Amostragem
T = 1/Fs; % Período de Amostragem
K = 2^16; % # de Amostras do Sinal
t = (0:K-1)*T; % Vetor do Tempo

fsig = 5000; % Sinal de 5 KHz

w = 2*pi*fsig; % Frequência Angular
c = 340.29; % Velocidade do Som no ar, m/s
k = 0.05; % Divisão do Espaço (Visada: -1 a 1) com passo k - Amostragem do Espaço
d = (c/fsig)/2; % Distância entre os microfones de meio comprimento de onda

%% FFT dos Sinais das Fontes

f1 = 5*sin(2*pi*fsig*t);
f2 = 7*sin(2*pi*fsig*t);
f3 = 9*sin(2*pi*fsig*t);

f = [f1;f2;f3]; % Sinais Analisados

uF = [-0.6 -0.8;0 0;0.5 0.5];
uF = [uF sqrt(1-uF(:,1).^2-uF(:,2).^2)]; % Direções das Fontes #1, #2 e #3

L = size(f,1); % # de Fontes

sigma2 = 0.1; % Variância do Ruído dos Sinais
% noise = sqrt(sigma2)*randn(L,length(t));

for i = 1:L

    noise(i,:) = sqrt(sigma2)*randn(1,length(t));

end

f = f + noise; % Sinal + Ruído de variância 0.1 e média nula

[F,W] = FFT(Fs,K,f); % Função FFT do sinal da fonte

Mod = abs(F);

%% Steering Vector, para todas as direções escolhidas

pos_x = [0 d 2*d 3*d 4*d 5*d 6*d];
p = arranjo(pos_x);

M = size(p,1); % # de Microfones

[V,u] = steer(fsig,p,k);
N = length(u); % # de Direções Vistas - Visada
```

```

%% Sinais nos Microfones no tempo

vu = steer(fsig,p,k,uF);

XX = F(1:K/2,:)*(vu);
xx = 2*real(K*ifft(XX,K));

figure()
plot(real(xx(1:150,:)), 'linewidth', 2), grid
title('Sinais em cada um dos Microfones, no tempo')
xlabel('Tempo')

%% Sinal nos Microfones, para uma frequência fsig (FFT)

index_sig = length(F)*fsig/Fs+1;
X = 2*XX(index_sig,:);
X = X.';

%% Cálculo de vx e vy
Mx = sqrt(M);
My = Mx; % arranjo com número quadrado de sensores

Nx = sqrt(N);
Ny = Nx;

ux = u(1:Nx,2)';
uy = ux';

px = p(1:Mx,1)';
py = px';

Vx = steer(fsig,px',k,ux');
Vy = steer(fsig,py,k,uy);

Vx = Vx.';
Vy = Vy.';

%% Delay and Sum - DAS:  $\hat{Y} = V^H X$ 

r = 200;
timel = zeros(r,1);
time2 = timel;

VV = kron(Vx,Vy)';
VVV = reshape(X,Mx,My);

for j = 1:r

    tic

    Y_hat1 = (1/M^2)*VV*X; % Sem a nova KAT

    timel(j) = toc;

end

```

```

for j = 1:r
    tic

    Y_hat2 = (1/M^2)*Vy'*VVV*conj(Vx); % Com a nova KAT

    time2(j) = toc;

end

Y_hat1 = reshape(Y_hat1,Ny,Nx);
Y_hat2 = reshape(Y_hat2,Ny,Nx);

%%

figure(), plot(1:r,time1,'linewidth',2)
hold on
plot(1:r,time2,'r','linewidth',2)
hold off
grid

title('Tempo de Processamento para cada realização')
xlabel('Realização')
ylabel('Tempo(s)')

legend('Sem a Nova KAT','Com a Nova KAT')

semKAT = mean(time1)
comKAT = mean(time2)

x = linspace(-1,1,sqrt(N));
y = x;
z = abs(Y_hat1).^2;
z = z';

mapa(x,y,z)

z = abs(Y_hat2).^2;
z = z';

mapa(x,y,z)

```

ANEXO D – Função “mapa.m”, que auxilia na geração da imagem acústica

```
%%% Colormap - TCC

function [] = mapa(x,y,uz)

    [X,Y] = meshgrid(x,y);

    Z = uz;

    figure()
    [~,h] = contourf(X,Y,Z,50);
    set(h,'EdgeColor','none');
    axis square

    figure()
    surf(X,Y,Z,'EdgeColor','none');

end
```

ANEXO E - Código para o cenário inicial para a Regularização l_1 sem uso da nova KAT

```
close all, clear all

%% Constantes

Fs = 2^17; % Frequência de Amostragem
T = 1/Fs; % Período de Amostragem
K = 2^16; % # de Amostras do Sinal
t = (0:K-1)*T; % Vetor do Tempo

fsig = 5000; % Sinal de 5 KHz

w = 2*pi*fsig; % Frequência Angular
c = 340.29; % Velocidade do Som no ar, m/s
k = 0.05; % Divisão do Espaço (Visada: -1 a 1) com passo k - Amostragem do Espaço
d = (c/fsig)/2; % Distância entre os microfones de meio comprimento de onda

%% FFT dos Sinais das Fontes

f1 = 5*sin(2*pi*fsig*t);
f2 = 7*sin(2*pi*fsig*t);
f3 = 9*sin(2*pi*fsig*t);

f = [f1;f2;f3]; % Sinais Analisados

uF = [-0.6 -0.8;0 0;.5 .5];
uF = [uF sqrt(1-uF(:,1).^2-uF(:,2).^2)]; % Direções das Fontes #1, #2 e #3

L = size(f,1); % # de Fontes

sigma2 = 0.1; % Variância do Ruído dos Sinais
% noise = sqrt(sigma2)*randn(L,length(t));

for i = 1:L

    noise(i,:) = sqrt(sigma2)*randn(1,length(t));

end

f = f + noise; % Sinal + Ruído de variância 0.1 e média nula

[F,W] = FFT(Fs,K,f); % Função FFT do sinal da fonte

Mod = abs(F);

%% Steering Vector, para todas as direções escolhidas

pos_x = [0 d 2*d 3*d 4*d 5*d 6*d];
p = arranjo(pos_x);

M = size(p,1); % # de Microfones
```

```

[V,u] = steer(fsig,p,k);
N = length(u); % # de Direções Vistas - Visada

%% Sinais nos Microfones no tempo

vu = steer(fsig,p,k,uF);

XX = F(1:K/2,:)*(vu);
xx = 2*real(K*ifft(XX,K));

figure()
plot(real(xx(1:150,:)), 'linewidth',2), grid
title('Sinais em cada um dos Microfones, no tempo')
xlabel('Tempo')

%% Sinal nos Microfones, para uma frequência fsig (FFT)

index_sig = length(F)*fsig/Fs+1;
X = 2*XX(index_sig,:);
X = X.';

%% Cálculo de vx e vy
Mx = sqrt(M);
My = Mx; % arranjo com número quadrado de sensores

Nx = sqrt(N);
Ny = Nx;

ux = u(1:Nx,2)';
uy = ux';

px = p(1:Mx,1)';
py = px';

Vx = steer(fsig,px',k,ux');
Vy = steer(fsig,py,k,uy);

Vx = Vx.';
Vy = Vy.';

%% Regularização l1 - Minimize ||Y||_1 subject to ||X - V*Y||_2 <= sigma

real = 1;
time_sem = zeros(real,1);
opts = spgSetParms('iterations', 200,'verbosity',1);

% Coeficiente de regularização
sigma = 2*sqrt(sigma2);

for i = 1:real
    tic
    Y = spg_bpdn(kron(Vx,Vy), X, sigma, opts); % Regularização L1 - Sem
nova KAT
    time_sem(i) = toc;
end

```

88

```
m_SemKAT = mean(time_sem(10:end))

figure(), plot(1:real,time_sem,'linewidth',2),grid
title('Tempo de Processamento para cada realização')
xlabel('Realização')
ylabel('Tempo(s)')

x = linspace(-1,1,sqrt(N));
y = x;
z = Y;
z = (abs(reshape(z,Nx,Ny))').^2;

mapa(x,y,z)
```

ANEXO F - Código para o cenário inicial para a Regularização l_1 com uso da nova KAT

```
close all, clear all

%% Constantes

Fs = 2^17; % Frequência de Amostragem
T = 1/Fs; % Período de Amostragem
K = 2^16; % # de Amostras do Sinal
t = (0:K-1)*T; % Vetor do Tempo

fsig = 5000; % Sinal de 5 KHz

w = 2*pi*fsig; % Frequência Angular
c = 340.29; % Velocidade do Som no ar, m/s
k = 0.05; % Divisão do Espaço (Visada: -1 a 1) com passo k - Amostragem do Espaço
d = (c/fsig)/2; % Distância entre os microfones de meio comprimento de onda

%% FFT dos Sinais das Fontes

f1 = 5*sin(2*pi*fsig*t);
f2 = 7*sin(2*pi*fsig*t);
f3 = 9*sin(2*pi*fsig*t);

f = [f1;f2;f3]; % Sinais Analisados

uF = [-0.6 -0.8;0 0;0.5 0.5];
uF = [uF sqrt(1-uF(:,1).^2-uF(:,2).^2)]; % Direções das Fontes #1, #2 e #3

L = size(f,1); % # de Fontes

sigma2 = 0.1; % Variância do Ruído dos Sinais
% noise = sqrt(sigma2)*randn(L,length(t));

for i = 1:L

    noise(i,:) = sqrt(sigma2)*randn(1,length(t));

end

f = f + noise; % Sinal + Ruído de variância 0.1 e média nula

[F,W] = FFT(Fs,K,f); % Função FFT do sinal da fonte

Mod = abs(F);

%% Steering Vector, para todas as direções escolhidas

pos_x = [0 d 2*d 3*d 4*d 5*d 6*d];
p = arranjo(pos_x);

M = size(p,1); % # de Microfones
```

```

[V,u] = steer(fsig,p,k);
N = length(u); % # de Direções Vistas - Visada

%% Sinais nos Microfones no tempo

vu = steer(fsig,p,k,uF);

XX = F(1:K/2,:)*(vu);
xx = 2*real(K*ifft(XX,K));

figure()
plot(real(xx(1:150,:)), 'linewidth',2), grid
title('Sinais em cada um dos Microfones, no tempo')
xlabel('Tempo')

%% Sinal nos Microfones, para uma frequência fsig (FFT)

index_sig = length(F)*fsig/Fs+1;
X = 2*XX(index_sig,:);
X = X.';

%% Cálculo de vx e vy
Mx = sqrt(M);
My = Mx; % arranjo com número quadrado de sensores

Nx = sqrt(N);
Ny = Nx;

ux = u(1:Nx,2)';
uy = ux';

px = p(1:Mx,1)';
py = px';

Vx = steer(fsig,px',k,ux');
Vy = steer(fsig,py,k,uy);

Vx = Vx.';
Vy = Vy.';

%% Regularização l1 - Minimize ||Y||_1 subject to ||X - V*Y||_2 <= sigma
- Com nova KAT

% Coeficiente de regularização
sigma = 2*sqrt(sigma2);

% Minimize ||Y||_1 subject to ||Vv*Y*Vx.' - X||_2 <= sigma
A = @(Y,mode) Anonima_KAT(Y, Mx, My, Nx, Ny, Vx, Vy, mode);

time_com = plot_l1(A, X, sigma, Nx, Ny); % Regularização L1

m_ComKAT = mean(time_com(10:end))

```

ANEXO G - Código para o cenário inicial para a Regularização TV sem uso da nova KAT

```
close all, clear all

%% Constantes

Fs = 2^17; % Frequência de Amostragem
T = 1/Fs; % Período de Amostragem
K = 2^16; % # de Amostras do Sinal
t = (0:K-1)*T; % Vetor do Tempo

fsig = 5000; % Sinal de 5 KHz

w = 2*pi*fsig; % Frequência Angular
c = 340.29; % Velocidade do Som no ar, m/s
k = 0.05; % Divisão do Espaço (Visada: -1 a 1) com passo k - Amostragem do Espaço
d = (c/fsig)/2; % Distância entre os microfones de meio comprimento de onda

%% FFT dos Sinais das Fontes

f1 = 5*sin(2*pi*fsig*t);
f2 = 7*sin(2*pi*fsig*t);
f3 = 9*sin(2*pi*fsig*t);

f = [f1;f2;f3]; % Sinais Analisados

uF = [-0.6 -0.8;0 0;.5 .5];
uF = [uF sqrt(1-uF(:,1).^2-uF(:,2).^2)]; % Direções das Fontes #1, #2 e #3

L = size(f,1); % # de Fontes

sigma2 = 0.1; % Variância do Ruído dos Sinais
% noise = sqrt(sigma2)*randn(L,length(t));

for i = 1:L

    noise(i,:) = sqrt(sigma2)*randn(1,length(t));

end

f = f + noise; % Sinal + Ruído de variância 0.1 e média nula

[F,W] = FFT(Fs,K,f); % Função FFT do sinal da fonte

Mod = abs(F);

%% Steering Vector, para todas as direções escolhidas

pos_x = [0 d 2*d 3*d 4*d 5*d 6*d 7*d ];
p = arranjo(pos_x);

M = size(p,1); % # de Microfones
```

```

[V,u] = steer(fsig,p,k);
N = length(u); % # de Direções Vistas - Visada

%% Sinais nos Microfones no tempo

vu = steer(fsig,p,k,uF);

XX = F(1:K/2,:)*(vu);
xx = 2*real(K*ifft(XX,K));

figure()
plot(real(xx(1:150,:)), 'linewidth',2), grid
title('Sinais em cada um dos Microfones, no tempo')
xlabel('Tempo')

%% Sinal nos Microfones, para uma frequência fsig (FFT)

index_sig = length(F)*fsig/Fs+1;
X = 2*XX(index_sig,:);
X = X.';

%% Cálculo de vx e vy
Mx = sqrt(M);
My = Mx; % arranjo com número quadrado de sensores

Nx = sqrt(N);
Ny = Nx;

ux = u(1:Nx,2)';
uy = ux';

px = p(1:Mx,1)';
py = px';

Vx = steer(fsig,px',k,ux');
Vy = steer(fsig,py,k,uy);

Vx = Vx.';
Vy = Vy.';

%% Total Variation - TV

opts.maxit = 100;
opts.isreal = true;
opts.nonneg = true;
opts.TVL2 = true;
opts.mu = 2^10;
opts.disp = false;

real = 1;
time_sem = zeros(real,1);

for i = 1:real
    tic
    Y = TVAL3(kron(Vx,Vy), X, Ny, Nx, opts); % Regularização TV - Sem nova
KAT
    time_sem(i) = toc;

```

```
end
```

```
%%
```

```
m_SemKAT = mean(time_sem(10:end))
```

```
figure(), plot(1:real,time_sem,'linewidth',2),grid  
title('Tempo de Processamento para cada realização')  
xlabel('Realização')  
ylabel('Tempo(s)')
```

```
x = linspace(-1,1,sqrt(Nx*Ny));  
y = x;  
z = Y;  
z = ((reshape(z,Ny,Nx))').^2;
```

```
mapa(x,y,z)
```

ANEXO H - Código para o cenário inicial para a Regularização TV com uso da nova KAT

```
close all, clear all

%% Constantes

Fs = 2^17; % Frequência de Amostragem
T = 1/Fs; % Período de Amostragem
K = 2^16; % # de Amostras do Sinal
t = (0:K-1)*T; % Vetor do Tempo

fsig = 5000; % Sinal de 5 KHz

w = 2*pi*fsig; % Frequência Angular
c = 340.29; % Velocidade do Som no ar, m/s
k = 0.05; % Divisão do Espaço (Visada: -1 a 1) com passo k - Amostragem do Espaço
d = (c/fsig)/2; % Distância entre os microfones de meio comprimento de onda

%% FFT dos Sinais das Fontes

f1 = 5*sin(2*pi*fsig*t);
f2 = 7*sin(2*pi*fsig*t);
f3 = 9*sin(2*pi*fsig*t);

f = [f1;f2;f3]; % Sinais Analisados

uF = [-0.6 -0.8;0 0;.5 .5];
uF = [uF sqrt(1-uF(:,1).^2-uF(:,2).^2)]; % Direções das Fontes #1, #2 e #3

L = size(f,1); % # de Fontes

sigma2 = 0.1; % Variância do Ruído dos Sinais
% noise = sqrt(sigma2)*randn(L,length(t));

for i = 1:L

    noise(i,:) = sqrt(sigma2)*randn(1,length(t));

end

f = f + noise; % Sinal + Ruído de variância 0.1 e média nula

[F,W] = FFT(Fs,K,f); % Função FFT do sinal da fonte

Mod = abs(F);

%% Steering Vector, para todas as direções escolhidas

pos_x = [0 d 2*d 3*d 4*d 5*d 6*d 7*d];
p = arranjo(pos_x);

M = size(p,1); % # de Microfones
```

```

[V,u] = steer(fsig,p,k);
N = length(u); % # de Direções Vistas - Visada

%% Sinais nos Microfones no tempo

vu = steer(fsig,p,k,uF);

XX = F(1:K/2,:)*(vu);
xx = 2*real(K*ifft(XX,K));

figure()
plot(real(xx(1:150,:)), 'linewidth',2), grid
title('Sinais em cada um dos Microfones, no tempo')
xlabel('Tempo')

%% Sinal nos Microfones, para uma frequência fsig (FFT)

index_sig = length(F)*fsig/Fs+1;
X = 2*XX(index_sig,:);
X = X.';

%% Cálculo de vx e vy
Mx = sqrt(M);
My = Mx; % arranjo com número quadrado de sensores

Nx = sqrt(N);
Ny = Nx;

ux = u(1:Nx,2)';
uy = ux';

px = p(1:Mx,1)';
py = px';

Vx = steer(fsig,px',k,ux');
Vy = steer(fsig,py,k,uy);

Vx = Vx.';
Vy = Vy.';

%% Total Variation - TV

A = @(Y,mode) Anonima_KAT(Y, Mx, My, Nx, Ny, Vx, Vy, mode);

time_com = plot_TV(A, X, Nx, Ny); % Regularização TV - Com nova KAT

m_ComKAT = mean(time_com(10:end))

```

ANEXO I – Função “FFT.m” que auxilia no cálculo da FFT dos sinais nos microfones

% Cálculo da FFT - Mateus de Campos da Silva

```
function [F,f] = FFT(Fs,K,x)

    t = (0:K-1)*(1/Fs);

    Ox = x(:,1:150);
    Ox = Ox';
    figure()
    plot(Fs*t(1:150),Ox,'linewidth',2)
    title('Signal in Time')
    xlabel('time')
    grid

    NFFT = 2^nextpow2(K); % Next power of 2 from length of y
    F = fft(x',NFFT)/K;
    f = Fs/2*linspace(0,1,NFFT/2+1);

    figure()
    % Plot single-sided amplitude spectrum.
    plot(f,2*abs(F(1:NFFT/2+1,:)),'linewidth',2)
    title('Single-Sided Amplitude Spectrum of x(t)')
    xlabel('Frequency (Hz)')
    ylabel('|F(f)|')
    grid

end
```

ANEXO J – Função “Mode1.m” e “Mode2.m” presentes no código da regularização l_1 e TV

Mode1.m

```
function Z = Model(Y,Vx,Vy)

[Mx,Nx] = size(Vx);
[My,Ny] = size(Vy);

cost1 = My * Nx * Ny/2 + Mx * My * Nx;
cost2 = Mx * Nx * Ny/2 + Mx * My * Ny;

if cost1 < cost2

    Z = (Vy * Y) * Vx.';

else

    Z = Vy * (Y * Vx.');
```

end

Mode2.m

```
function Z = Mode2(Y,Vx,Vy)

[Mx,Nx] = size(Vx);
[My,Ny] = size(Vy);

cost1 = Ny * Mx * (My + Nx/2);
cost2 = Nx * My * (Mx + Ny/2);

if cost1 < cost2
    % Z = (Vy' * Y) * conj(Vx);
    F1 = Vy' * Y;
    F2 = conj(Vx);
else
    % Z = Vy' * (Y * conj(Vx));
    F1 = Vy';
    F2 = Y * conj(Vx);
end

Z = F1*F2;
```

end

ANEXO K – Função “AnonimaKAT.m”, função do tipo anônima que auxilia no processo de minimização, tanto na regularização l_1 como na TV

```
function Z = Anonima_KAT(in_vec, Mx, My, Nx, Ny, Vx, Vy, mode)

    if mode == 1

        Y = reshape(in_vec, [Ny Nx]);
        % Z = Vy * Y * Vx.';
        Z = Model(Y, Vx, Vy);
        Z = Z(:);

    else

        Y = reshape(in_vec, [My Mx]);
        % Z = Vy' * Y * conj(Vx);
        Z = Mode2(Y, Vx, Vy);
        Z = Z(:);

    end

end
```

ANEXO L – Funções “plot_l1.m” e “plot_TV.m”

plot_l1.m

```
function [time] = plot_l1(A, b, sigma, Nx, Ny)

    options = spgSetParms('iterations', 200, 'verbosity', 1);
    real = 1;
    time = zeros(real, 1);

    for i = 1:real

        tic
        Y = spg_bpdn(A, b, sigma, options); % minimize ||X||_1 subject to
        ||AX - B|| <= SIGMA
        time(i) = toc;

    end

    figure(), plot(1:real, time, 'linewidth', 2), grid
    title('Tempo de Processamento para cada realização')
    xlabel('Realização')
    ylabel('Tempo(s)')

    x = linspace(-1, 1, sqrt(Nx*Ny));
    y = x;
    z = Y;
    z = abs(reshape(z, Nx, Ny))';
    % z = z+(z==0)*(1e-10);
    % z = db(z);
    z = z.^2;

    mapa(x, y, z)

end
```

plot_TV.m

```
function time = plot_TV(A, b, Nx, Ny)

    opts.maxit = 100;
    opts.isreal = true;
    opts.nonneg = true;
    opts.TVL2 = true;
    opts.mu = 2^10;
    opts.disp = false;

    real = 1;
    time = zeros(real, 1);

    for i = 1:real

        tic
        Y = TVAL3(A, b, Ny, Nx, opts);
        time(i) = toc;

    end

end
```

```

figure(), plot(1:real,time,'linewidth',2),grid
title('Tempo de Processamento para cada realização')
xlabel('Realização')
ylabel('Tempo(s)')

x = linspace(-1,1,sqrt(Nx*Ny));
y = x;
z = Y;
z = abs(reshape(z,Nx,Ny))';
z = z.^2;

mapa(x,y,z)

end

```