Universidade Federal do ABC

Vinícius Ormenesse

Construção de um sistema para estimação de direção de chegada de sinais acústicos

Santo André

2015

Vinícius Ormenesse

Construção de um sistema para estimação de direção de chegada de sinais acústicos

Trabalho de Graduação de Curso apresentado no curso de graduação em Engenharia de Informação na Universidade Federal do ABC como requisito parcial para obtenção do título de Bacharel em Engenharia de Informação.

Prof. Orientador: Murilo Bellezoni Loiola

Santo André

2015

DEDICATÓRIA

Dedico este trabalho a toda minha família e a todos que apoiaram em minha vida.

AGRADECIMENTOS

Gostaria de agradecer este trabalho aos meus pais que, apesar de todas dificuldades encontradas em suas vidas, me apoiaram e me puseram em primeiro plano incontáveis vezes.

Agradeço meu irmão que tanto me ensinou sobre a importância dos estudos e sobre o prazer que um livro pode nos trazer. Minhas avós, que sempre estiveram junto a mim toda a minha vida, e nunca deixaram de me apoiar mesmo não concordando com as minhas decisões.

A Thaysa, que todos os dias esteve junto a mim, na realização desse trabalho. Agradeço ainda mais por ter me ensinado a dar maior valor a minha vida e a vida do próximo.

Não posso me esquecer de todos meus grandes amigos, que embora não tenham feito parte do desenvolvimento deste trabalho, tantas vezes estiveram junto a mim durante a minha vida, e não posso me dar ao direito de dizer que cheguei aqui sem eles.

Agradeço ao governo brasileiro pode ter me oferecido uma universidade de altíssimo nível, contribuindo a realização de meus sonhos.

RESUMO

Este trabalho de graduação tem como objetivo estudar algoritmos que estimam a direção de chegada (DOA, do inglês *direction of arrival*) de sinais de áudio captados por arranjos de microfones.

Testou-se diferentes algoritmos de estimativa de chegada, como TDOA (diferença do tempo de chegada, do inglês *time difference-of-arrival*), MUSIC, ESPRIT, correlação e Beamforming

Após validar os resultados, implementou-se o algoritmo MUSIC em um sistema criado totalmente em laboratório, com um microcontrolador, microfones de eletreto e pré-amplificadores.

Obteve-se um erro quadrático médio entre a direção de chegada esperada e a direção de chegada estimada de 29.94º.

Este projeto de graduação envolve diretamente e indiretamente disciplinas como: sistemas microprocessados, processamento digital de sinais, transformadas em Sinas e sistemas lineares, eletrônica analógica, entre outras.

ABSTRACT

The main objective of this work is to study some algorithm that can estimate DOA (direction of arrival) with audio signals captured with an array of microphones.

Different direction of arrival algorithms were tested, as MUSIC (Multiple Signal Classification), TDOA (Time difference of arrival), correlation, Beamforming and ESPRIT (Estimation of Signal Parameter via Rotational Invariance Technique).

After validating the results, MUSIC, was implemented in a system, with microcontrollers, microphones and pre amplifiers.

We obtained a mean square error between the direction of expected arrival and direction of arrival estimated 29.94º.

SIGLAS

AD – Analógico-Digital

DOA – *Direction of Arrival* (Direção de chegada)

ESPRIT – *Estimation of Signal Parameter via Rotational Invariance Technique*

FIR – *Finite Impulse Response* (Resposta ao Impulso Finita)

MUSIC – *Multiple Signal Classification*

PRUSS - *Programmable Real-time Unit Sub System* (subsistema programável em tempo real)

SOI – *Signal of Interest* (Sinal de interesse)

TDE – *Time delay estimation* (estimativa de tempo de atraso)

TDOA – *Time difference of arrival* (diferença do tempo de chegada)

UFABC – Universidade Federal do ABC

ULA – *Uniform Linear Array* (Arranjo linear Uniforme)

Sumário

LISTA DE FIGURAS

CAPITULO 1 - INTRODUÇÃO

Introdução

Antenas inteligentes são conjuntos de antenas, arranjadas segundo uma determinada geometria (linear, circular, etc) que reagem dinamicamente às mudanças do ambiente, com o objetivo de fornecer um sinal de maior qualidade e gerar um melhor aproveitamento das faixas de frequência nas comunicações, além de prover dinamicamente informações sobre a fonte do sinal.

Para uma melhor analogia de antenas inteligentes, imaginemos duas pessoas que mantêm uma conversa no interior de um quarto escuro. Junto às duas pessoas, o ouvinte é capaz de determinar a localização da pessoa que fala à medida que ela se move pelo quarto, pois sua voz chega a cada sensor acústico, os ouvidos, em instantes diferentes de tempo. O processador humano de sinal, o cérebro, adiciona as intensidades dos sinais dos dois ouvidos para focar no sinal proveniente da direção calculada. Além do mais, se outras pessoas integram a conversa o cérebro pode eliminar a interferência indesejada e se concentrar em uma conversa de cada vez [1].

Sistemas elétricos de antenas inteligentes funcionam de maneira similar, usando duas ou mais antenas no lugar de ouvidos, ou até mesmo dois microfones dependendo da aplicação, e um processador digital de sinais ao invés de um cérebro humano. Portanto, depois do processador digital de sinais medir os atrasos temporais de cada sensor, ele calcula a direção de chegada (DOA) do sinal de interesse (SOI – Signal of Interest) e ajusta as excitações do sinal (ganhos e fases dos sinais) para produzir um diagrama de radiação que foque no SOI e, idealmente, elimine qualquer sinal que não seja de interesse [1]. Este processo é também conhecido como Beamforming.

Para sinais acústicos, é possível empregar as mesmas técnicas de estimação de chegada empregadas em antenas inteligentes, porém utilizando microfones no lugar de antenas como dispositivos sensores. A estimação de chegada para sinais acústicos é útil em dispositivos em geolocalização de todos os tipos como por exemplo em vídeo conferências, em uma reunião, onde um webcam é capaz de sempre focar na pessoa falando. Pode ser utilizada também em sonares

(Navegação e Determinação da Distância pelo Som), um instrumento utilizado para localização de outros elementos, muito utilizado em submarinos.

Objetivos

Os principais objetivos desse projeto são estudar os princípios e algoritmos de estimação de chegada (DOA), implementar um sistema que é capaz de adquirir sinais de áudio através de vários sensores dispostos linearmente e, calcular a direção de chegada desses sinais, além de obter medidas de desempenho e avaliar resultados obtidos com o arranjo.

Neste projeto, o sistema construído é composto por um arranjo linear de microfones, um sistema de pré-amplificação dos sinais captados pelos microfones e um hardware capaz de processar os sinais captados, como está ilustrado na Figura 1.

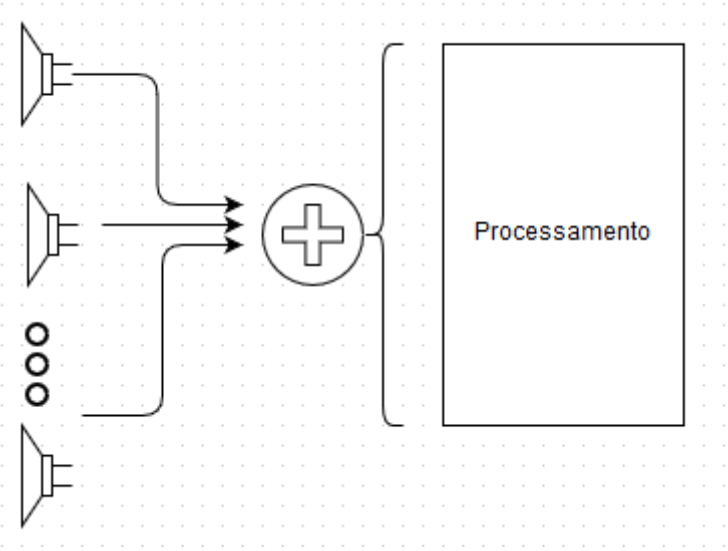


Figura 1. Exemplo de sistema para estimação DOA.

Metodologia

As seguintes atividades fizeram parte do desenvolvimento do projeto:

1. Estudo bibliográfico e das principais ferramentas utilizadas para a implementação de algoritmos de direção de chegada (Correlação Cruzada, MUSIC, ESPIRIT, Beamforming).

2. Estudo de aquisição de sinais a partir de uma placa de aquisição em sistemas operacionais Windows com MATLAB.

3. Implementação dos algoritmos em scripts MATLAB.

4. Simulação e validação dos algoritmos em MATLAB.

5. Testes a partir de placa de aquisição e validação de algoritmos de direção de chegada em laboratório.

6. Estudo bibliográfico para escolha de melhor microcontrolador para a implementação de algoritmos de direção de chegada.

7. Estudo e implementação de pré-amplificação de microfones de eletreto com intuito de utilizá-los no microcontrolador adequado.

8. Construção de um sistema para estimação de chegada.

9. Experimentos do sistema em espaço aberto e fechado.

CAPITULO 2 – FUNDAMENTOS TEÓRICOS

Neste capítulo serão apresentados alguns conceitos e modelos básicos relativos à sinais acústicos e arranjos de sensores. Os fundamentos abordados aqui serão empregados nos outros capítulos desde trabalho.

Geração de onda básica e transmissão

Uma onda sonora é uma onda longitudinal, mecânica, que transmite o que é associado pelos seres humanos como som [2]. A onda se propaga em um meio elástico e contínuo gerando uma variação local de pressão e densidade, que se transmite em forma de onda esférica periódica. As variações de pressão, humidade ou temperatura do meio, produzem um deslocamento das moléculas que as formam [2]. Cada molécula transmite a vibração provocando um movimento em cadeia [2].

Ilustrado na figura 2, pode-se ver uma onda senoidal. A componente vertical da onda, assimilada á sua fase, é desenhada de acordo com o eixo do tempo, mostrado na figura 2 na parte *A*. A cada 360º de rotação, a estrutura da onda, ou a forma de onda, começa novamente. Normalmente, utilizamos a frequência em Hz e o período em segundos para representar as ondas. [2]

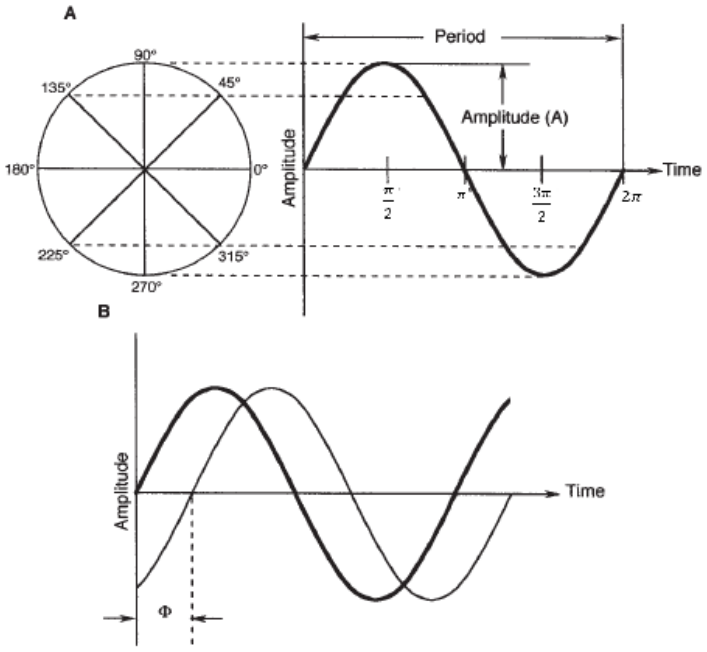


Figura 2. Geração de um sinal senoidal (A), relação de fase entre duas ondas senoidais (B). [2]

Para radiações de ondas senoidais em um meio físico como o ar, a figura acima representa uma onda em um ambiente estático de pressão e atmosfera, onde ondas senoidais são representadas por alternar valores de pressão positivos e negativos na pressão estática. O período então corresponde ao comprimento de onda dividido por sua velocidade.

A velocidade de propagação do som no ar é aproximadamente igual à 344 metros por segundo (m/s), e a relação entre comprimento de onda, velocidade do som e frequência (1/s) é dada por:

$$v\ (velocidade) = \ f\ (frequência)\ .\lambda\ (comprimento\ de\ onda) \qquad (1)$$

Outra relação fundamental entre duas formas de onda que possuem mesma frequência são as suas fases relativas (ϕ), ou seja, a diferença de fase que duas ondas senoidais possuem entre si. A fase é normalmente medida em graus ou em radianos. Se duas ondas de mesma amplitude e frequência são deslocadas 180º elas irão se cancelar desde que estejam em contra fase. Se não tiverem a mesma amplitude, estas não irão cancelar diretamente.

Dependência da velocidade do som com a temperatura

Para a maioria das amostragens feitas em ambientes fechados, podemos assumir que em temperaturas normais a velocidade do som prevalecerá como o abaixo, onde há uma pequena dependência da temperatura na propagação do som:

$$Velocidade\ \left(\frac{m}{s}\right) = \ 331.4 + 0.607 * ºC\ [4] \qquad (2)$$

onde *ºC* representa a temperatura em graus Celsius.

Potência Acústica

Potência é uma grandeza expressa em *watts* (W), ou *joules/segundo.* O *joule* é uma unidade de trabalho ou energia, e *joule/segundo* é a taxa com a qual o trabalho é feito, ou energia gasta. [2]

A intensidade *(I)* é definida como potência por unidade de área *(W/m²)*, ou a taxa de energia fluindo por unidade de área.

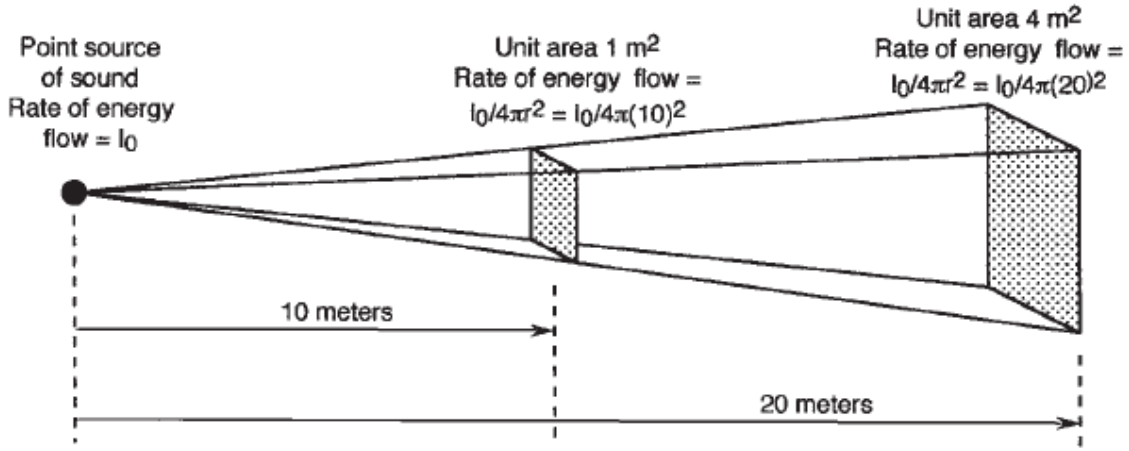


Figura 3. Variação da intensidade ao decorrer do espaço. [4]

Na figura 4, pode-se ver uma fonte de sinal e sua intensidade sendo dispersada no espaço livre. Aqui será examinado, como exemplo, um pequeno ponto de radiação. Numa distância de 10 m que o pequeno ângulo está irradiando através de um quadrado (observar figura acima) com uma área de 1 m², apenas uma pequena porção de *I* vai passar através desta área; a uma distância de 20 m a área necessária para acomodar a mesma energia é muito maior, cerca de 4 m², a partir desta figura, fica claro que a intensidade numa distância de 20 m será um quarto do que era a 10 m. Isso é necessário devido ao princípio da lei de conservação de energia.[2]

Portanto, a intensidade a uma distância *r* da fonte, é dada como:

$$I = \frac{P}{4 * \pi * r^2} \tag{3}$$

onde *P* é a potência acústica.

A pressão efetiva do som, que é a pressão da onda sonora medida em um certo ponto e um certo intervalo de tempo, nesta distância será de:

$$p = \sqrt{I\rho_0 v} \tag{4}$$

onde $\rho_0 v$ é a impedância acústica específica do ar (em geral, $405\ \ Pa.m^{-1}.s$) .

Campo distante

Para ondas em geral, o campo distante é a região do espaço onde a onda eletromagnética pode ser considerada plana. Ou seja, a partir de uma certa região do espaço, não é percebida como uma radiação esférica e sim como uma radiação. O campo distante é um fator muito importante para o desenvolvimento de sistemas de estimação de chegada, pois os algoritmos só funcionam em campo distante.

A distância de Frauenhofer [8], nomeada depois de Joseph von Frauenhofer, é definida como:

$$d_m \geq \frac{2D^2}{\lambda} \tag{5}$$

onde d_m é a distância mínima para que o sinal acústico seja considerado no campo distante. *D* é o maior comprimento linear da rede e λ é o comprimento de onda.

Aliasing Espacial

Em amostragem, a frequência de Nyquist é a frequência mínima que um sinal pode ser amostrado para que posteriormente, possa ser recriado sem haver perda de informação [7]. Um arranjo linear uniforme de sensores implementa uma amostragem espacial de sinal e com isso, existe uma condição análoga a Nyquist para evitar a perda de informação do sinal recebido originalmente.

Aliasing espacial é perda de informação por má configuração dos sensores no espaço assim como não respeitar a taxa de Nyquist induz a perda de informação na amostragem. Nyquist diz que a frequência de amostragem f_s deve ser:

$$f_s > 2f_{max} \tag{6}$$

ou seja, a frequência de amostragem deve ser no mínimo duas vezes maior que a frequência máxima presente.

Similarmente, para a amostragem espacial existe a necessidade de que

$$f_{xs} = \frac{1}{d_{ma}} > 2\, f_{max} \tag{7}$$

Onde f_{xs} é a frequência de amostragem espacial em amostragens por metro, e f_{max} é a maior frequência espacial no espectro angular do sinal e d_{ma} é o espaçamento entre sensores.

A frequência espacial é dada por [7]:

$$f_x = sen\theta cos\varphi/\lambda \tag{8}$$

Portanto, pode-se dizer que a frequência espacial máxima é dada por:

$$f_{xmax} = 1/\lambda_{min} \tag{9}$$

E, com isso, deduz-se que a distância entre os sensores deve ser

$$d_{ma} < \frac{\lambda_{min}}{2} = \frac{c}{2f_{max}} \tag{10}$$

onde λ_{min} é o menor comprimento de onda do sinal de interesse. Ou seja, o espaçamento entre os sensores deve ser menor que a metade do menor comprimento de onda do sinal amostrado para evitar o aliasing espacial. A equação acima é conhecida como o teorema de amostragem espacial.

Tempo de chegada entre dois sensores

Nesta seção, utilizando as definições estudadas acima, deduz-se um problema simples de TDOA (*time difference of arrival*), que facilita no entendimento da estimativa de direção de chegada.

No simples caso de dois microfones, o problema de TDOA é demonstrado na figura 4, onde *d* é a distância entre dois microfones e θ é o ângulo incidente do sinal do som da fonte.

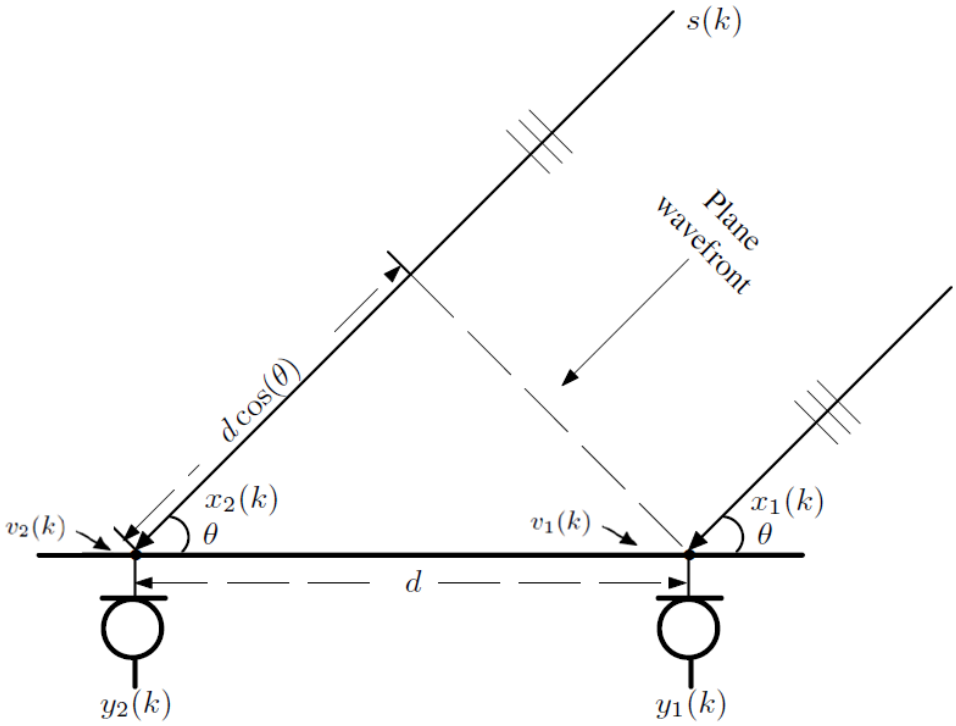


Figura 4. Representação caso ideal em campo livre.

Supondo também, que os sinais de áudio vem do campo distante (na figura, *'Plane wavefront'*) e que $x_1\,[k]\ e\ x_2[k]$ são as entradas dos microfones incluindo o ruído aditivo representados por $v_1\,[k]\ e\ v_2[k]$ nos microfones 1 e 2, respectivamente, e $y_1\,[k]\ e\ y_2[k]$ são os sinais amostrados no microfone. Os ruídos são supostos não correlacionados com o sinal da fonte e não correlacionados entre si.

A diferença do caminho acústico entre a fonte e os dois microfones é $d\ cos\theta$. O ângulo incidente pode ser calculado se o espaçamento do microfone *'d'* é conhecido e $d\ cos\theta$ determinado pelo tempo de atraso estimado entre os dois microfones e a fonte[3].

Apesar do ângulo incidente poder ser estimado com o uso de dois ou mais sensores, em um campo distante, estimar a distância entre a fonte de som e os microfones é muito difícil.

Entretanto, se a fonte se encontrar em um campo não distante, é agora possível estimar não apenas o ângulo de chegada por onde a onda atinge cada sensor, como também a distância entre a fonte e cada microfone.

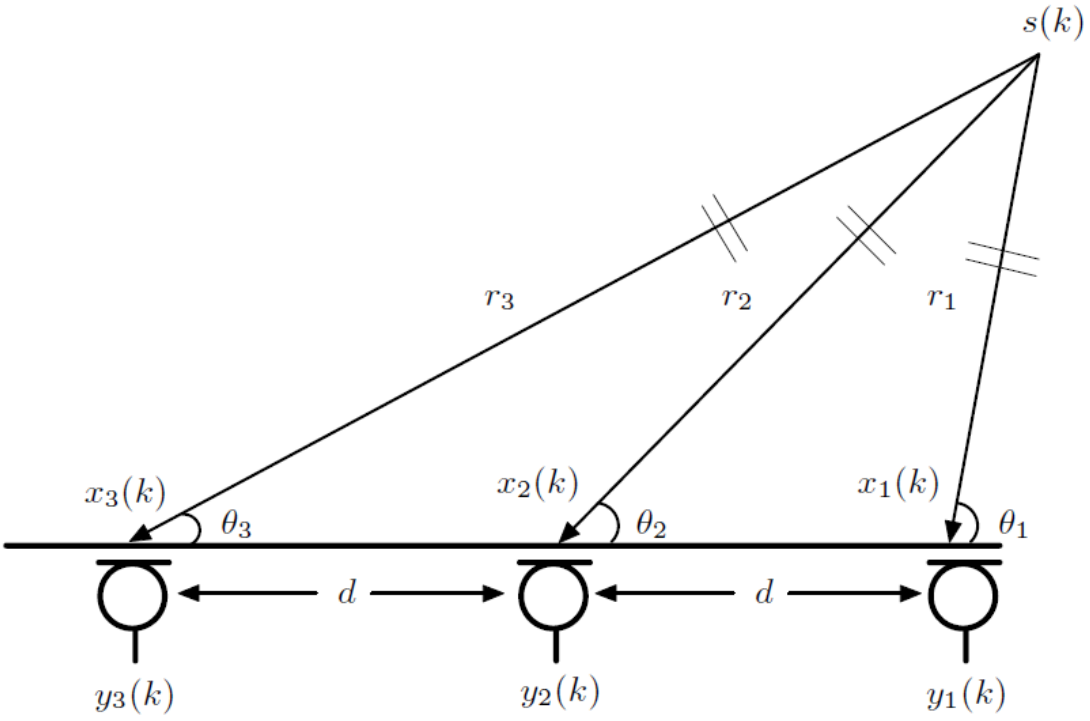


Figura 5. Problema com fonte em campo não distante. O espaçamento entre os dois vizinhos é 'd'. [3]

Para demonstrar, vamos considerar o simples exemplo mostrado na figura 5. Primeiramente, escolhemos o primeiro microfone como sensor de referência. Denotamos θ_n e r_n como sendo, respectivamente, o ângulo incidente e a distância entre a fonte de som e o microfone *n, n = 1,2,3*. O TDOA entre o primeiro e o segundo sensores é dada como:

$$\tau_{12} = \frac{r_2 - r_1}{v} \tag{11}$$

e o TDOA entre o terceiro e o primeiro sensores é:

$$\tau_{23} = \frac{r_3 - r_1}{v} \tag{12}$$

Aplicando a regra do cosseno para triangular na figura 6, obtém-se:

$$r_2^2 = r_1^2 + d^2 + 2\theta_1 d cos(\theta_1) \tag{13}$$

e

$$r_3^2 = r_1^2 + 4d^2 + 4\theta_1 d cos(\theta_1) \tag{14}$$

Para um sistema prático, o espaçamento *'d'* pode sempre ser medido uma vez que a geometria do sistema é conhecida. Se τ_{12} e τ_{13} estão disponíveis, então podemos calcular todos os parâmetros desconhecidos como: θ_1 , r_1 , r_2 e r_3 resolvendo as equações 11 e 12. Aplicando a lei dos senos nas equações 13 e 14, podemos obter as estimativas de θ_2 e θ_3.[4]

No entanto, os modelos matemáticos para os sinais de campo próximo e de campo distante são diferentes. Mesmo assim, o passo fundamental para obter a informação da fonte de origem é a estimativa da TDOA entre dois microfones diferentes. Entretanto, em situações práticas, o sinal da fonte é geralmente imerso em ambientes com ruídos, isso porque vivemos em um ambiente natural onde há ruído e a sua existência é inevitável. Além disso, cada observação do sinal pode conter múltiplas réplicas atenuadas e atrasadas da fonte devido às reflexões com paredes, objetos ente outros. Esse efeito de multi-caminho de propagação induz distorções de ecos e espectrais no sinal observado. O nome apropriado para este fenômeno é reverberação, que deteriora severamente o sinal. Em adição, a fonte pode se mover de tempos em tempos, resultando na mudança do ângulo de chegada, além dos erros de quantização no processador de sinais. Todos esses fatores fazem da direção de chegada complicada.

Redes lineares e uniformes de sensores

Em redes com arranjos lineares e uniformes (ULA), do inglês *uniform linear array*, considera-se um arranjo de *m* sensores idênticos uniformemente espaçados em uma linha, como na figura 6.

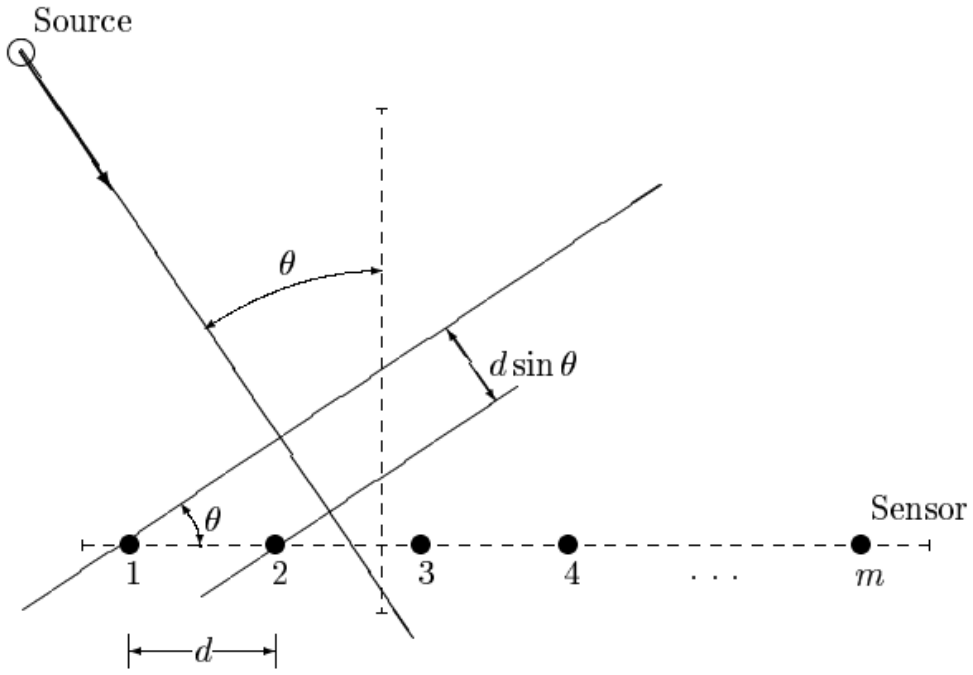


Figura 6. Exemplo de arranjo linear e uniforme.

Como apresentado na figura 6, deixa-se *d* representado como a distância entre dois sensores consecutivos e θ como sendo a direção de chegada do sinal. Então, como dito anteriormente, e assumindo que o primeiro sensor é o ponto de referência temos que [5]:

$$\tau_k = (k-1)\frac{d\sin\theta}{c}\ para\ \theta\ \in [-90º, 90º] \tag{15}$$

onde *c* é a velocidade de propagação do som.

O vetor de transferência [5], é o parâmetro de interesse no problema de direção de chegada. Ao, selecionar o primeiro sensor como ponto de referência e para apenas um sinal de referência o vetor pode ser dado como:

$$a(\theta) = [1\ e^{-iw_c\tau_2}\ \ldots\ e^{-iw_c\tau_m}]^T \quad (16)$$

onde w_c é a frequência de onda na qual está trabalhando o sistema.

Nota-se que o vetor de transferência é uma função dependentemente de θ somente já que w_c é a frequência de onda do sinal recebido e τ_k depende somente de θ. Através da equação acima pode-se então escrever a equação do sinal:

$$y(t) = a(\theta)s(t) + e(t) \quad (17)$$

onde $s(t)$ é o sinal acústico capturado e $e(t)$ é ruído aditivo:

$$y(t) = [y_1(t) \ldots y_m(t)]^T \quad (18)$$

$$e(t) = [e_1(t) \ldots e_m(t)]^T \quad (19)$$

é o vetor de saída do arranjo e o ruído aditivo, respectivamente [5].

Ao considerarmos:

$$w_s = 2\pi f_s = w_c \frac{d \sin\theta}{c} = w_c\tau_k \quad e \quad (20)$$

$$f_s = f_c \frac{d \sin\theta}{c} \quad sendo \quad f_c = \frac{c}{\lambda} \quad (21)$$

Pode-se reescrever o vetor de transferência como:

$$a(\theta) = [1\ e^{-iw_s}\ \ldots\ e^{-i(m-1)w_s}]^T \quad (22)$$

Os algoritmos de direção de chegada apresentados neste projeto serão desenvolvidos a partir do vetor de transferência de $a(\theta)$ acima.

Filtros FIR

Para um melhor resultado da direção de chegada, algoritmo exigem que o sinal recebido seja filtrado por um filtro banda estreita ou um filtro passa baixas. Dentre todos os filtros disponíveis, o filtro FIR oferece várias vantagens e por isso foi escolhido para a utilização do projeto.

O filtro FIR faz parte dos tipos primários utilizados em aplicações processamento digital de sinais. O nome FIR vem de *"Finite Impulsive Response"*, que em português significa resposta impulsiva finita. [9]

O filtro FIR é finito pois não há retroalimentação, e a falta de retroalimentação garante que a resposta ao impulso é finita. [9]

Os filtros com característica FIR possuem diversas vantagens, dentre elas:

- Podem ser facilmente projetados para ter uma fase linear.
- São fáceis de implementar. Em muitos processadores digitais de sinais, FIR pode ser implementado apenas com loop de uma instrução.
- Podem ser utilizados em aplicações multi taxa, ou seja, podem ser utilizados em aplicações que aumenta a taxa de amostragem ou reduzem a taxa de amostragem, FIR permite que alguns cálculos sejam omitidos, fazendo com que alguns cálculos sejam omitidos, aumentando assim, a eficiência.
- Podem ser utilizados com precisão finita de bit sem causar muitos problemas [9].
- Podem ser implementados com aritmética fracionária, ou seja, podem ser implementados com coeficientes com magnitude menor que 1.0.

Um filtro FIR é desenvolvido encontrando algumas especificações seus coeficientes e ordem de filtro, que podem estar no domínio do tempo ou no domínio da frequência (mais comum). Filtros adaptados realizam uma correlação cruzada entre o sinal de entrada e um tipo de pulso conhecido. A convolução FIR é uma correlação cruzada entre o sinal de entrada e uma cópia da resposta de impulso invertida pelo tempo. Portanto, a resposta ao impulso de filtros adaptados é "desenvolvido" amostrando a forma de pulso conhecida e utilizando essas amostras em ordem reversa como os coeficientes do filtro. [10]

CAPITULO 3 – ALGORITMOS DE ESTIMAÇÃO DE CHEGADA

Neste capítulo será apresentado os algoritmos de estimação de chegada. Os algoritmos aqui apresentados foram fundamentais para o desenvolvimento do sistemas de estimação de direção de chegada.

Método de Correlação Cruzada

É o método mais simples de para estimação do TDOA [4]. Considere uma fonte única e apenas dois sensores, N=2. A função de correlação entre os dois sinais observados, *y1(t) e y2(t),* pode ser definida como:

$$r_{y_1 y_2}(p) = E[y_1(t) y_2(t+p)] \tag{23}$$

Dada a função de correlação cruzada, pode-se obter uma estimativa de TDOA entre $y_1(t)\ e\ y_2(t)$ como:

$$\hat{\tau} = \underset{p}{argmax}\ r_{y_1 y_2}(p) \tag{24}$$

onde $p \in [-\tau_{\max}, \tau_{\max}]$, e $\tau_{\max}$ é o maior atraso possível, calculado na equação 15.

Após o cálculo do atraso pode-se encontrar o ângulo de chegada através de:

$$\theta = \operatorname{asin}(\frac{\tau * c}{d}) \tag{25}$$

onde *'c'* é a velocidade do som e *'d'* é a distância entre os dois sensores.

O método de correlação cruzada é simples de ser implementado. Maiores informações sobre as deduções deste método podem ser encontradas em [4].

Método correlação

O modo mais fácil de se estimar a direção de chegada em um sistema com vários microfones, é utilizando o método da correlação. [5]

O método de da correlação é dado por:

$$P_{corr}(\theta) = a^T(\theta) * \hat{R} \tag{26}$$

onde $P_{corr}(\theta)$ é um vetor varrido por 180º e $\hat{R}$ é a matriz de covariância dada por:

$$\hat{R} = \frac{1}{N} \sum_{t=m}^{N} y\,(t) y^*(t) \tag{27}$$

O vetor $P_{corr}(\theta)$ atinge seu máximo valor para certo θ na varredura.

Como objetivo é estimar o ângulo θ. Sabe-se pela desigualdade de Cauchy-Schwarz que a função $a^T(\theta) * \hat{R}$ atinge o seu valor máximo quando o ângulo varrido for igual ao ângulo da fonte sonora incidente.

Beamforming

Pode-se pensar que um arranjo linear de sensores, realiza uma filtragem temporal da onda incidente o que é análogo a uma filtragem temporal feita por uma linha de atraso de um filtro FIR (do inglês, *finite impulse response*). [7]

Após medido os atrasos temporais de cada sensor, O Beamforming foi pensado para para produzir um diagrama de radiação que foque no sinal de interesse e, idealmente, que elimine qualquer sinal que não seja de interesse

A equação para encontrar o ângulo de chegada pelo Beamforming pode ser dada através da seguinte equação:

$$P(\theta) = a^*(\theta)Ra(\theta) \tag{28}$$

onde *a* é o vetor de transferência da equação 16 e *R* é a matriz de correlação dada pela equação 27.

A direção de chegada é dada pelo maior pico na função. Maiores informações sobre o algoritmo Beamforming, pode ser encontrado em [5].

A seguir será apresentado, passo a passo, o algoritmo baseado em beamforming para estimação de DOA.

Algoritmo

Passo 1. Computar matriz de covariância, equação 38.

Passo 2. A direção de chegada do beamforming pode ser estimada como os ***n*** maiores picos da função

$$a^*(\theta)\hat{R}a(\theta) \tag{29}$$

MUSIC

O método MUSIC (Multiple Signal Classification) é o primeiro método considerado de alta resolução para computar a estimação de direção de chegada. Para isso, MUSIC utiliza autovalores e auto vetores para separar o subespaço de sinal do subespaço de ruído.

A varredura do algoritmo é feita através da seguinte equação:

$$\frac{1}{a^*(\theta)GG^*a(\theta)}, \qquad \theta \in [-90º, 90] \tag{30}$$

onde *'G'* representa a matriz de auto vetores contendo apenas o subespaço de sinal.

Maiores informações sobre o algoritmo MUSIC, pode ser encontrado em [5].

Passo 1. Computar a matriz de covariância

$$\hat{R} = \frac{1}{N}\sum_{t=m}^{N} y\,(t)y^*(t) \tag{31}$$

E seus auto valores e auto vetores. Denotamos $\hat{S}\ e\ \hat{G}$ como as matrizes similarmente a $S\ e\ G$, mas feitos com os auto vetores $\{\widehat{s_1}, \dots, \widehat{s_n}\}$ e $\{\widehat{g_1}, \dots, \widehat{g_{m-n}}\}$ de $\hat{R}$.

Passo 2. Determinar a estimação de frequências assim como a localização dos *n* maiores picos da função:

$$\frac{1}{a^*(\theta)GG^*a(\theta)}, \qquad \theta \in [-90º, 90] \tag{32}$$

Onde os maiores picos dessa varredura serão os ângulos de chegada estimados.

ESPRIT

No caso de um arranjo linear uniforme, ESPIRIT pode ser utilizado para estimação de chegada do mesmo jeito que pode ser utilizado para estimação de frequência [5]. Para arranjos não lineares, o algoritmo ESPIRIT pode ser utilizado apenas em alguns casos. Mais precisamente, a diferentemente de outros algoritmos, ESPIRIT pode ser utilizado apenas para encontrar a estimação de chegada se existem dois arranjos idênticos. Matematicamente, essa condição pode ser formulada como a seguinte. $\bar{m}$ é o número de sensores em dois arranjos lineares idênticos [5].

Sendo $A = [a(w_c\tau(\theta_1)) \dots a(w_c\tau(\theta_1))]$ $(\bar{m} \times \bar{m})$, A_1 *e* A_2 são as sub matrizes de A correspondentes a estes sub arranjos. Desde que os sensores no arranjo são numerados arbitrariamente, não existe restrição para assumir que A_1 é feito da primeira $\bar{m}$ colunas em A *e* A_2 da ultima $\bar{m}$:

$$\begin{aligned} A_1 &= [I_{\bar{m}} \quad 0]A \quad (\bar{m} \times \bar{m}) \\ A_2 &= [0 \quad I_{\bar{m}}]A \quad (\bar{m} \times \bar{m}) \end{aligned} \tag{33}$$

$I_{\bar{m}}$ – denota como a matriz identidade $(\bar{m} \times \bar{m})$.

Normalmente $\bar{m} = m/2$ e os dois arranjos não se sobrepõe.

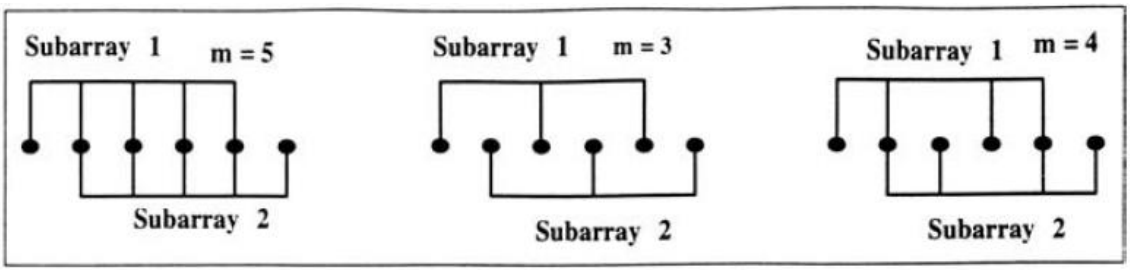


Figura 7. Exemplos de arranjos para o método ESPRIT. [6]

Matematicamente, o que o algoritmo ESPRIT exige é:

$$A_2 = A_1 D \tag{34}$$

Onde,

$$D = \begin{bmatrix} e^{-iw_c\tau(\theta_1)} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & e^{-iw_c\tau(\theta_n)} \end{bmatrix} \tag{35}$$

onde $\tau(\theta_n)$ denota o tempo necessário por uma frente de onda interferindo de um arranjo em uma direção θ viajando em dois arranjos iguais.

$$\tau(\theta) = \frac{dsin(\theta)}{c} \tag{36}$$

Onde d é a distância entre os dois arranjos. Portanto, a estimativa de chegada pode ser derivada da estimativa dos elementos diagonais de D, em (35) [5].

No ESPRIT a direção de chegada é resolvida resolvendo um problema de autovalores. No algoritmo não existe uma busca, como nos métodos anteriores, ou seja, não há separação da direção de chegada e do ruído. Apesar de tudo isso, o algoritmo ESPRIT só pode ser utilizado com a configuração de ULA descrita acima.

Maiores informações sobre o algoritmo ESPRIT, podem ser encontradas em [5].

A seguir é apresentado o algoritmo ESPIRIT, passo a passo para estimação de direção de chegada

Passo 1. Computar a matriz de covariância

$$\hat{R} = \frac{1}{N}\sum_{t=m}^{N} y\,(t)y^{*}(t) \tag{37}$$

Passo 2. Computar a SVD, extrair as matrizes $U_{NxN}, S_{NxN}, V_{NxN}$, respectivamente.

Passo 3. Da matriz U_{NxN} , criar uma matriz chamada D_{Nxn} , onde *'n'* representa a quantidade fontes de áudio.

Passo 4. Criar a partir de D a matriz $D1_{Gxn}$, onde G é igual a número de sensores no arranjo linear. Para a matriz D1, é pego da primeira linha da matriz D até a linha G-1.

Passo 5. Criar a partir de D a matriz $D2_{Gxn}$, onde G é igual a número de sensores no arranjo linear. Para a matriz D2, é pego da segunda linha da matriz D até a linha G.

Passo 6. Calcular Ψ:

$$\Psi = (D2^{H}D2)^{-1}D2^{H}D1 \tag{38}$$

Passo 7. Os ângulos de direção de chegada são as posições angulares dos autovalores de Ψ.

Simulação

Após ter estudado os algoritmos, foram feitas simulações dos algoritmos para a determinação do método mais viável para a construção de um sistema de chegada.

Na primeira simulação, foi considerado um arranjo linear com seis microfones, a frequência de onda considerada nos testes foi de 1700 Hz, esta frequência foi escolhida pois uma das primeiras referências utilizavam esta frequência [16].

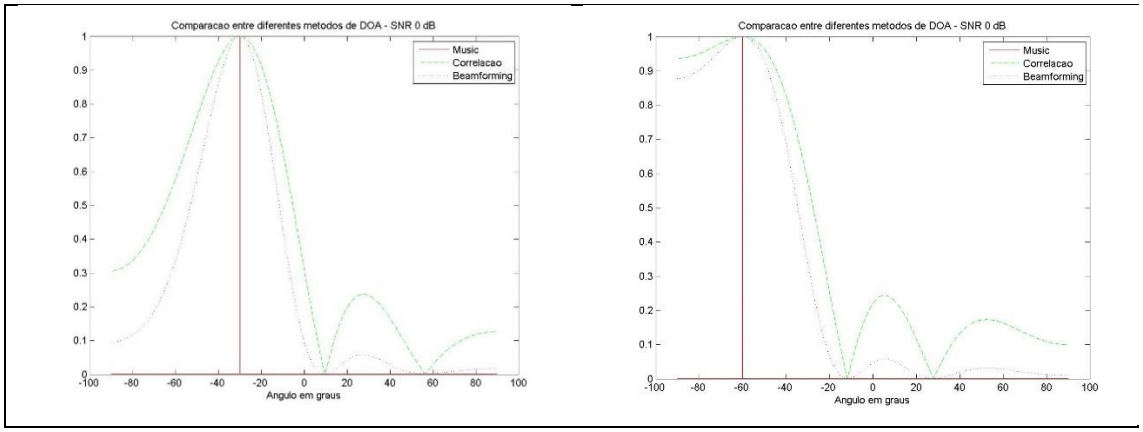
Considerando a velocidade do som, nos testes, de 340 m/s. E foi definido o espaçamento de cada sensor como sendo a metade do comprimento de onda da frequência de onda escolhida, de 1700 Hz.

Considerou-se a fonte no campo distante, e ausência de *aliasing espacial.*

Foi possível obter resposta gráfica apenas para MUSIC, correlação e Beamforming. O algoritmo ESPRIT produz apenas um resultado com o valor do ângulo incidente.

Na primeira parte, foi simulado um sinal sem ruído. Os resultados abaixo mostrados são para ângulos incidentes de: -90º, -60º, -30º, 0º, 30º, 60º, 90º. Para os métodos da Correlação, Beamforming e MUSIC a varredura do algoritmo foi de um passo de 0.1º, partindo de -90º até 90º.

Abaixo é mostrado figuras com os resultados para cada uma das simulações:



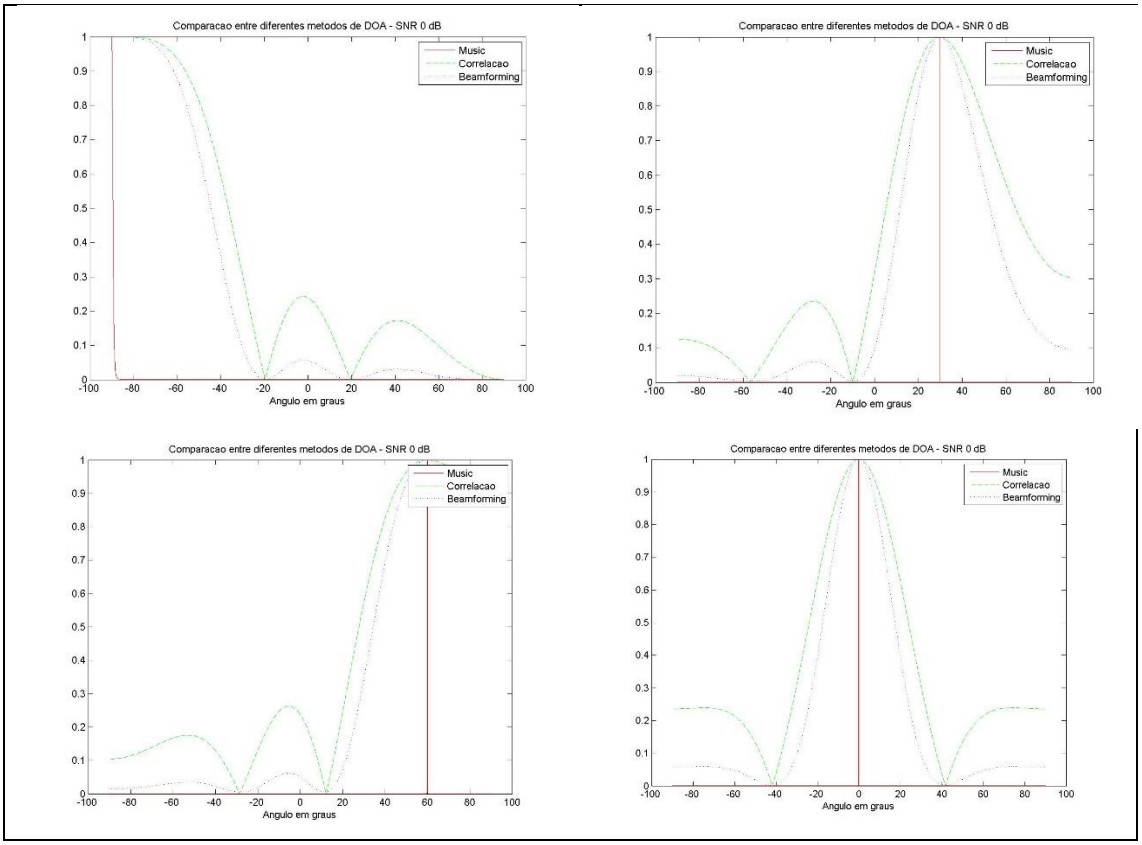


Figura 8. Resultado da simulação DOA utilizando algoritmos MUSIC, Beamforming e Correlação. Para ângulos de incidência: 0º,30º,60º,-30º,-60º,-90º.

Acima na figura, pode-se observar que embora cada algoritmo apresente uma curva de resultados distinta, o ângulo de incidência sinal foi encontrado por cada algoritmo. Enquanto Beamforming e Correlação apresentam resultados mais bruscos, MUSIC sugere apresentar resultados muito acurados, pois este é dado com um algoritmo de alta resolução [5].

Para ESPRIT, o resultado é apresentado na figura abaixo junto com os demais algoritmos já apresentados:

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT** (θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-90º	-90º (180º)	-90º	-90º
-60º	-60º	-60º (150º)	-60º	-60º
-30º	-30º	-30º (120º)	-30º	-30º
0º	0º	0º (90º)	0º	0º
30º	30º	30º (60º)	30º	30º
60º	60º	60º (30º)	60º	60º
90º	90º	90º (0º)	90º	90º

Figura 9. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), sem ruído com seis microfones.

No caso particular do ESPRIT, o algoritmo devolve os resultados entre 0º e 180º, por este motivo, na figura acima encontramos entre parêntesis o resultado real devolvido pelo algoritmo.

Como pôde-se observar, para a incidência de um sinal sem ruído, todos os algoritmos apresentaram ótima precisão.

Os próximos passos foram analisar os algoritmos com sinais de incidência apresentando ruído. Os algoritmos foram analisados com SNR (relação sinal ruído) de 10 dB, 7 dB, 5 dB, 3dB.

Para um arranjo linear com seis microfones, e sinal com uma SNR de 10 dB os resultados foram descritos na figura 10.

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT** (θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-89.8º	-89.3º (179.7º)	-89.8º	-89.2º
-60º	-60º	-60.1º (149.9º)	-60º	-60º
-30º	-30º	-30º (120º)	-30º	-29.9º
0º	0º	0.1º (89.9)	0º	0º
30º	30º	30º (60º)	30º	29.8º
60º	60º	60º (30º)	60º	60.2º
89º	89º	89º(1.04º)	83.6º	89º
90º	35.4º	-36.6º (126.6º)	-36.8º	0º

Figura 10. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), ruído de 10 dB.

Para um arranjo linear com seis microfones, e sinal com uma SNR de 7 dB os resultados foram descritos na figura 11.

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT** (θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-90º	-90º (180º)	-89.2º	-90º
-60º	-60º	-60º (150º)	-60º	-60º
-30º	-30º	-31º (119º)	-29.9º	-30º
0º	0º	0º (90º)	0º	0º
30º	30º	30º (60º)	29.8º	30º
60º	60º	60º (30º)	60.2º	60º
89º	88.3º	88.5º (1.5º)	83.6º	88.3º
90º	-54.5º	-38º (128º)	0º	-60º

Figura 11. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), ruído de 7 dB.

Para um arranjo linear com seis microfones, e sinal com uma SNR de 5 dB os resultados foram descritos na figura 12.

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT** (θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-90º	-90º (180º)	-89.3º	-90º
-60º	-60º	-61º (149º)	-60º	-60º
-30º	-30º	-31º (119º)	-29.9º	-30º
0º	0º	1º (89º)	0º	0º
30º	30º	30º (60º)	29.8º	30º
60º	60º	60º (30º)	60.2º	60º
89º	89.6º	88º (2º)	83.6º	89.6º
90º	22.4º	1.97º (89.97º)	0º	17.6º

Figura 12. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), ruído de 5 dB.

Para um arranjo linear com seis microfones, e sinal com uma SNR de 3 dB os resultados foram descritos na figura 13.

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT** (θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-90º	-89.2º (179.8º)	-89.2º	-90º
-60º	-60º	-61º (149º)	-60º	-60º
-30º	-30º	-31º (119º)	-29º	-30º
0º	0º	1º (89º)	0º	0º
30º	30º	30º (60º)	29.2º	30º
60º	60º	61º (29º)	60.2º	60º
89º	90º	90º (0º)	80.9º	90º
90º	-90º	90º (0º)	0º	-90º

Figura 13. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), ruído de 3 dB.

Erros de medição não foram considerados pois todos os dados foram criados pelo Matlab.

Os resultados simulados demostram que, apesar das diferentes SNR os algoritmos se comportaram muito bem com as variadas direções de chegada. Entretanto, percebeu-se que para direções de chegada perto de -90º e direções perto de 90º resultados inesperados foram encontrados.

Foi simulado todos os métodos, considerando um arranjo linear com quatro microfones para sinais sem ruído, SNR 10 dB e SNR de 5 dB. A figura 14 mostra os resultados dos algoritmos com sinais incidentes sem ruído.

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT**(θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-90º	-90º (180º)	-62.7º	-90º
-60º	-60º	-90º (180º)	-60º	-60º
-30º	-30º	-90º (180º)	-30º	-30º
0º	0º	0º (90º)	0º	0º
30º	30º	0º (º)	30º	30º
60º	60º	65º (65º)	60º	60º
89º	89º	89º (1º)	89º	89º
90º	90º	0º (0º)	-67.7º	-90º

Figura 14. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), sem ruído. 4 microfones.

Para um arranjo linear com quatro microfones, e sinal com uma SNR de 10 dB os resultados foram descritos na figura 15.

Ângulo esperado (θ)	Ângulo estimado **MUSIC**(θ)	Ângulo estimado **ESPRIT**(θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-89.5º	75º (75.4º)	-0.8º	-89.5º
-60º	-60º	65º (65.7º)	-58.9º	-60º
-30º	-30º	1.9º (1.9º)	-29.8º	-30º
0º	0º	0º (90º)	0º	0º
30º	30º	-88º (178º)	29.7º	30º
60º	60º	-90º (180º)	55.5º	60º
89º	87.4º	-90º (180º)	56.8º	87.4º
90º	-87.1º	-90º (180º)	-0.7º	87.1º

Figura 15. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), ruído 10 dB. 4 microfones.

Para um arranjo linear com quatro microfones, e sinal com uma SNR de 5 dB os resultados foram descritos na figura 16.

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT**(θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-89.6º	75º (75.4º)	-0.5º	-89.6º
-60º	-60º	65º (65.7º)	-58.9º	-60º
-30º	-30º	-30º (120º)	-29.8º	-30º
0º	0º	2.17º (2.17º)	0º	0º
30º	30º	89.9º (89.9º)	29.6º	30º
60º	60º	-88º (178º)	55º	60º
89º	87.1º	-90º (180º)	0.4º	87.1º
90º	-89.2º	0º (0º)	0º	89.2º

Figura 16. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), ruído 5 dB. 4 microfones.

Foi simulado todos os métodos, considerando um arranjo linear com dois microfones para sinais sem ruído, SNR 10 dB e SNR de 5 dB. A figura 17 mostra os resultados dos algoritmos com sinais incidentes sem ruído.

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT** (θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-90º	-90º (180.0º)	-67.5º	-90º
-60º	-60º	-60º (150º)	-60º	-60º
-30º	-30º	-30º (120º)	-30º	-30º
0º	0º	0º (90º)	0º	0º
30º	30º	60º (60º)	30º	30º
60º	60º	30º (30º)	60º	60º
89º	89º	89º (1º)	89º	89º
90º	-32.4º	0º (0º)	-28.3º	32.4º

Figura 17. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), sem ruído. 2 microfones.

Para um arranjo linear com dois microfones, e sinal com uma SNR de 10 dB os resultados foram descritos na figura 18.

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT** (θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-88.6º	-89º (179.0º)	-1.5º	-89.6º
-60º	-58.7º	-59º (149º)	-25.5º	-60º
-30º	-26.5º	-30º (120º)	-14.4º	-30º
0º	0º	0º (90º)	0º	0º
30º	26.5º	60º (60º)	14.4º	30º
60º	58.7º	30º (30º)	25.1º	60º
89º	30º	89º (1º)	26.2º	90º
90º	88.6º	0º (0º)	-0.3º	-32.3º

Figura 18. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), ruído 10 dB. 2 microfones.

Para um arranjo linear com dois microfones, e sinal com uma SNR de 10 dB os resultados foram descritos na figura 19.

Ângulo esperado (θ)	Ângulo estimado **MUSIC** (θ)	Ângulo estimado **ESPRIT** (θ)	Ângulo estimado **Correlação**(θ)	Ângulo estimado **Beamforming**(θ)
-90º	-30º	-90º (180º)	-0.8º	-90º
-60º	-55.7º	-59.9º (149.9º)	-25.5º	-60º
-30º	-24.5º	-30º (120º)	-14.4º	-30º
0º	0º	0º (90º)	0º	0º
30º	24.5º	60º (60º)	14.4º	30º
60º	55.7º	30º (30º)	25.5º	60º
89º	30º	88º (2º)	26º	89º
90º	-88.3º	0º (0º)	-0.1º	32.4º

Figura 19. Resultado da simulação de todos algoritmos (ESPRIT, Beamforming, MUSIC, correlação), ruído 5 dB. 2 microfones.

Para a simulação com arranjo linear de quatro microfones, percebeu-se que os resultados apresentados na tabela foram muito bons para o algoritmo ESPRIT. Pela tabela, o algoritmo Beamforming apresentou bons resultados, para sinais sem ruído, porém para sinais com ruído a curva dada pela varredura do algoritmo aparentava apresentar mais de um sinal de incidência com um pico quase mais tão alto quanto o pico de incidência original. O algoritmo de Correlação, não apresentou resultados coerentes quando se tratava de sinais com ruídos, para a curva varrida. O algoritmo MUSIC, apresentou bons resultados para sinais sem ruído, entretanto para sinais com ruído presente, o algoritmo apresentou apenas um pico, diferentemente do Beamforming.

Para a simulação com arranjo linear com dois microfones o algoritmo ESPRIT manteve-se precisa em comparação aos outros algoritmos. O algoritmo Correlação apresentou grandes discrepâncias entre o resultado desejado e o resultado obtido. O algoritmo Beamforming, embora tenha acertado, a curva de varredura continuou apresentando resultados discrepantes com mais de um pico, enquanto que o MUSIC, embora não tenha sido tão preciso, consegue manter ainda apenas um pico.

Neste trabalho, embora o ESPRIT tenha sido o algoritmo mais preciso, para a construção do sistema para estimação de chegada, foi escolhido o algoritmo MUSIC, pois este algoritmo é de simples implementação em placa, é um algoritmo de alta resolução, não apresentou problemas de mais vários picos na presença de ruído.

CAPITULO 4 – CONSTRUÇÃO DE SISTEMA DE DIREÇÃO DE CHEGADA

Este capítulo apresenta o sistema de estimação de direção de chegada, criado em laboratório através de um microcontrolador, pré amplificadores e dois microfones de eletreto, como sensores.

Construção de um sistema para estimação de chegada

No decorrer do projeto, foi desenvolvido uma placa com o intuito de adquirir sinais de áudio para a estimação de direção de chegada. O algoritmo implementado foi MUSIC. Foi escolhido o BeagleBoard:BeagleBoneBlack [11] como o microcontrolador a ser utilizado para calcular a estimação de direção de chegada.

BeagleBoneBack

O BeagleBoneBlack é um computador de baixa potência com hardware open-source produzido pela Texas Instruments com Digi-Key e Newark element14 [11]. Foi desenvolvido com o intuito de ser uma placa de baixo custo, para ser utilizada em faculdades ao redor do mundo.

O BeagleBoneBlack pode ser considerado como uma placa poderosa em relação ao seu tamanho. Dentre as características principais de hardware como motivo de escolha do hardware no projeto estão[11]:

- Processador AM3358BZCZ100, 1GHz
- Memória ram 512 MB DDR3L, 800 MHz
- Memória Flash 4 Gb
- Baixo consumo energético
- 8 conversores analógico digital
- 69 GPIO (General Purpose Input Output, Entrada/Saída de propósito geral)
- 1 saída USB
- Entrada de cartão de memória
- 1 PRUSS (Programmable Real-time Unit Sub System, subsistema programável em tempo real), para processamento em tempo real

- 4 Timers
- 4 portas seriais
- Possibilidade de utilização de sistemas operacionais como debian, Arch Linux, Android, etc.

Essas características acima proporcionaram maior facilidade no desenvolvimento do algoritmo e eletrônica do sistema.

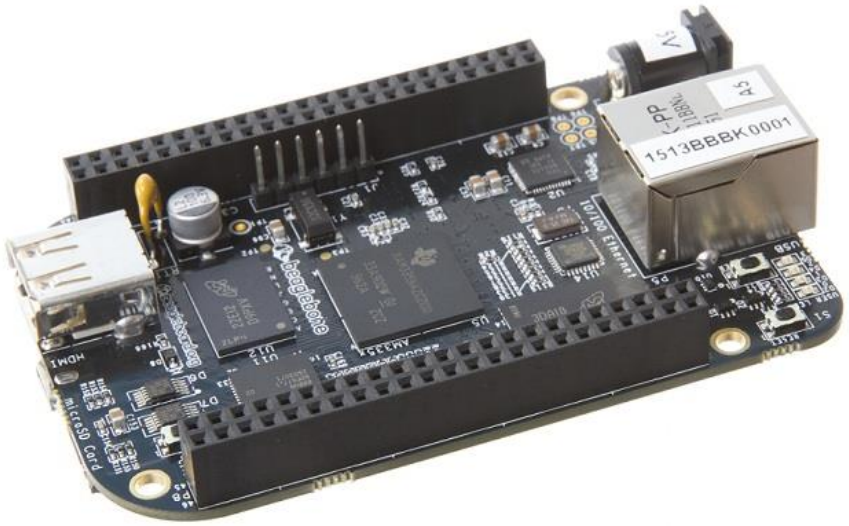


Figura 20. Imagem do BeagleBoneBlack, utilizado no projeto.[11]

Sistema Operacional e linguagem de programação

Para o desenvolvimento do projeto, o sistema operacional Debian foi instalado na placa assim como todas as dependências. A utilização de um sistema operacional facilitou muito ao escolher a linguagem de programação a ser utilizada, pois a placa não fica dependente de apenas uma linguagem de programação, e há vantagens de protocolos e serviços já instalados pelo sistema operacional.

Para a direção de chegada, foi escolhida a linguagem Python 2.7 pois sua sintaxe é simples, é uma linguagem orientada a objeto, possui muitas bibliotecas e principalmente é capaz de trabalhar com número, vetores e matrizes complexas. O Python possibilitou um desenvolvimento ágil e descomplicado do algoritmo MUSIC, com suas bibliotecas matemáticas e sua sintaxe simples.

Por motivos do sistema operacional e limitações do hardware, o conversor analógico digital não é kernel do sistema operacional, pois este não é capaz de dar a devida prioridade para tarefas de entrada e saída do conversor, limitando a amostragem em cerca de 10 amostras por segundo. Este problema pode ser contornado através da PRUSS do BeagleBoneBlack, que é um núcleo apartado do núcleo principal do microcontrolador.

Subsistema programável em tempo real

O subsistema programável em tempo real, ou PRUSS, consiste em dois núcleos apartado do processador, programado em tempo real. Cada núcleo consiste em ser 32 bit, com uma velocidade de 200 MHz e 8kB de memória de programa além de acesso direto à entrada e saída da placa. Esses núcleos estão conectados a várias memórias de dados, módulos periféricos e controles de interrupções para acesso do sistema.

A PRUSS é programada em Assembly, com a maioria das intruções podendo ser executada em apenas um ciclo do núcleo, não havendo necessidade de cache ou pipe-lining. A 200 MHz [13], a maioria das operações tardam cerca de 5ns para executar com exceção de tentativa de acesso a memória externa da PRU. Embora a PRU seja programada em assembly, há compiladores específicos e bibliotecas a nível de SO que permitem programação na linguagem C.

Figure 2. PRU-ICSS Integration

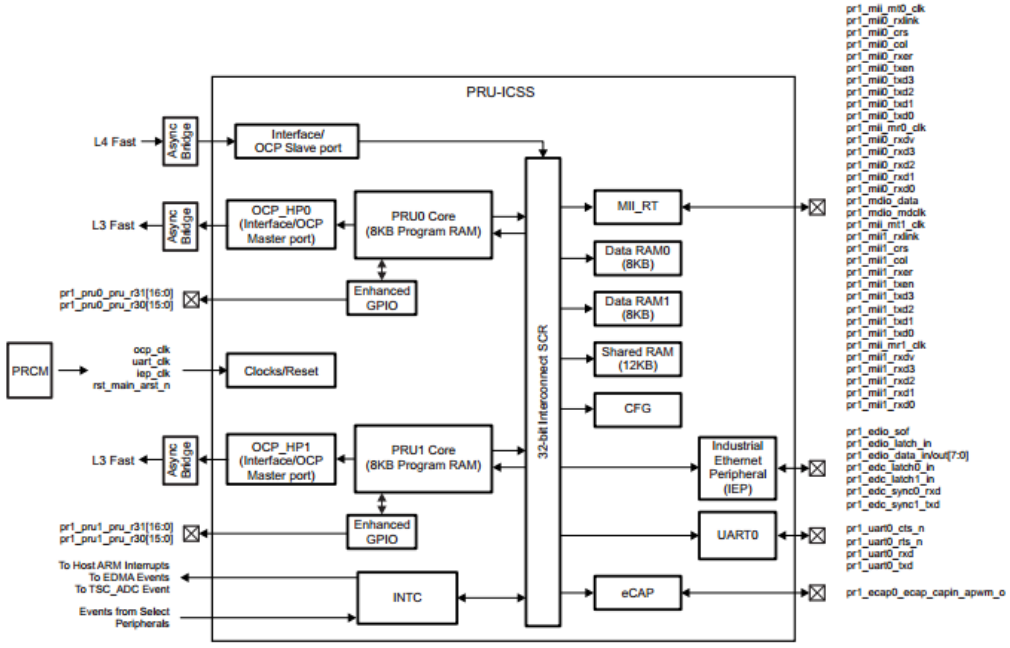


Figura 21. Integração PRUSS [13].

Neste projeto, a PRUSS foi programa em C sendo utilizado uma biblioteca e compiladores específicos, pois o desenvolvimento do código é mais simples que Assembly. O código em C compilado é processado através de uma requisição do Python para que apenas um script seja capaz de realizar todas as tarefas de uma só vez.

Entretanto, utilizar um código compilado por C resultou em uma PRUSS incapaz de utilizar a sua taxa de amostragem máxima, que é de 200 MHz [13]. Devido a dificuldades, não foi conseguido amostrar com a PRUSS em uma taxa fixa, esta taxa é variável. Entretanto, pôde-se ser feito uma média da taxa de amostragem, que foi de 33000kHz. Comparativamente, com a amostragem máxima, a taxa amostrada é bem abaixo porém para utilização de um sistema em tempo real que amostra sinais acústicos esta taxa de amostragem é mais que o suficiente.

Aquisição dos sinais acústicos

Para aquisição dos sinais acústicos, foi utilizado microfones de eletreto com um pré-amplificador, e este sistema foi conectado diretamente nos conversores analógico digital da placa.

Figura 22. Exemplo de microfone de eletreto.

As características gerais do microfone de eletreto são [13]:

- Omnidirecional
- Tensão permitida de 2V a 10V
- Máximo consumo de corrente de 0.5mA
- Impedância de saída de 2.2kΩ
- Frequência de operação de 50 até 16000 Hz
- Relação sinal ruído de 60 dB
- Temperatura de funcionamento de -20ºC até 60ºC

O microfone de eletreto é muito foi utilizado neste projeto devido a sua capacidade de funcionar em baixas tensões (neste projeto 3.3 V).

Entretanto o sinal capturado pelo microfone de eletreto é um sinal muito pequeno e este deve ser amplificado antes que o conversor analógico digital capture os dados. Neste projeto foi desenvolvido um pré-amplificador para amplificar este sinal.

Pré-amplificador

O pré-amplificador escolhido para o projeto foi emissor comum, que é baseado em um transistor bipolar, tipicamente utilizado como amplificador de tensão.

Esses circuitos, são utilizados para amplificar sinais de baixa tensão, tais como sinais acústicos captados pelo microfone de eletreto.

O circuito utilizado para amplificar o sinal acústico é mostrado na figura 23. O circuito utilizado foi encontrado em um site especializado em microcontroladores [15]. Foram feitas algumas modificações no circuito para adequar o pré-amplificador aos conversores AD do BeagleBoneBlack.

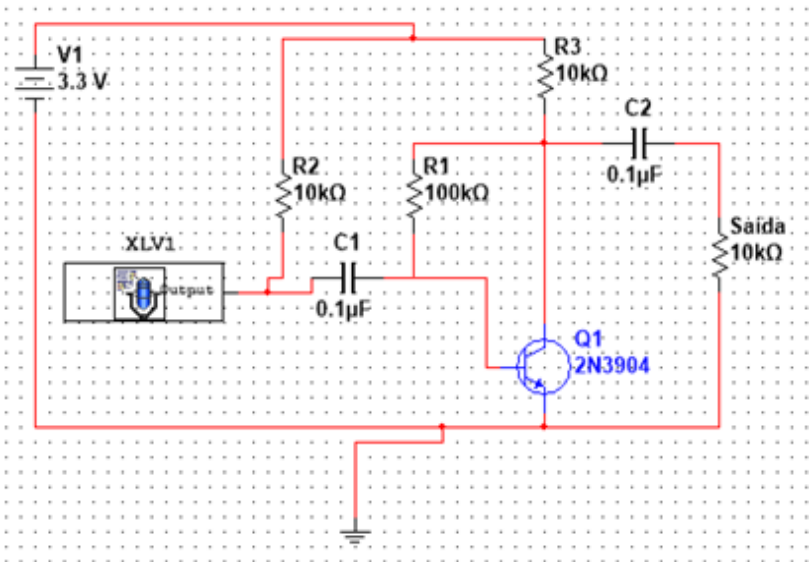


Figura 23. Pré amplificador microfone de eletreto[15].

Foi simulado em computador, com o software Multisim, o diagrama de bode a fim de saber qual seria o ganho do microfone com o pré-amplificador. O ganho simulado foi de 28 dB.

O diagrama pode ser visto na figura 24.

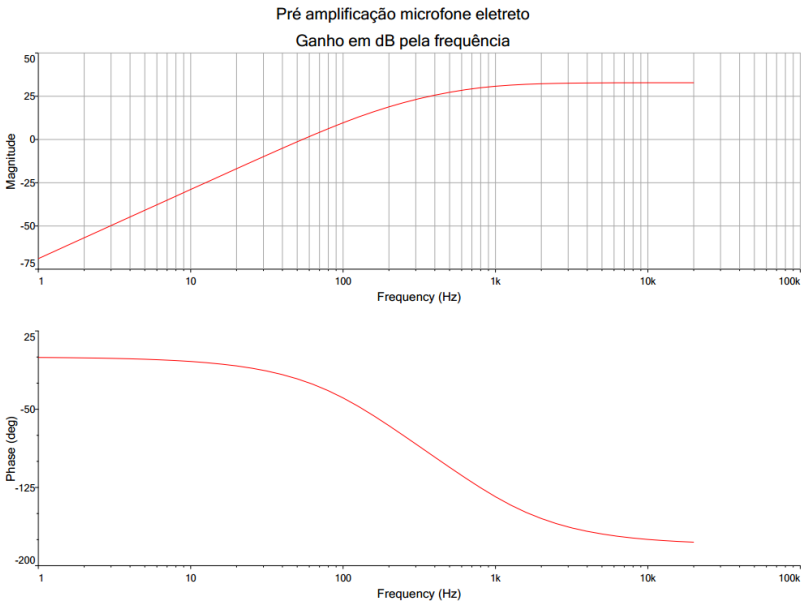


Figura 24. Diagrama de Bode para microfone de eletreto com pré-amplificador.

O próximo passo para o desenvolvimento do sistema foi definir o espaçamento dos microfones, assim como a frequência de onda que será utilizada para o sistema.

Arranjo linear de microfones

O arranjo linear desenvolvido neste projeto, possui apenas dois microfones, isso porque a PRUSS consegue amostrar ao mesmo tempo, sinais vindos apenas de dois conversores AD (analógico digital), pois embora a placa possua sete conversores AD, estes conversores estão multiplexados por apenas dois núcleos conversores. Caso fosse pensado em utilizar mais conversores AD, deve-se esperar um atraso na ordem de nano segundos vindos de pares de conversores.

O espaçamento dos microfones no arranjo linear de microfones desenvolvido é de (9.6±0.005) cm pois a placa utilizada para solda do sistema é uma placa quadrada de 10 cm x 10 cm. Fazendo os cálculos de aliasing espacial, considerando a velocidade do som de 340 m/s. E considerando a frequência do sinal acústico, a mesma frequência utilizada para a simulação. Temos:

$$d_{ma} < \frac{\lambda_{min}}{2} = \frac{c}{2f_{max}} = \frac{340}{2 * 1700} = 0.1\ m \tag{39}$$

Embora perto do limite de 0.1 m, este arranjo linear estará livre de aliasing espacial pois os microfones estão espaçados a menos que 0.1 m. O campo distante para esse sistema construído é dada pela equação de campo distante:

$$d_m \geq \frac{2D^2}{\lambda} = \frac{2(0.096)^2}{0.2} = (0.09216\ \pm 0.01)\ m \tag{40}$$

Portanto, o campo distante para este sistema é de (9.216±1) cm. O que não é uma distância muito grande e a fonte de sinal acústico não terá que ficar muito longe do sistema.

Na figura 25, é ilustrado o sistema completo.

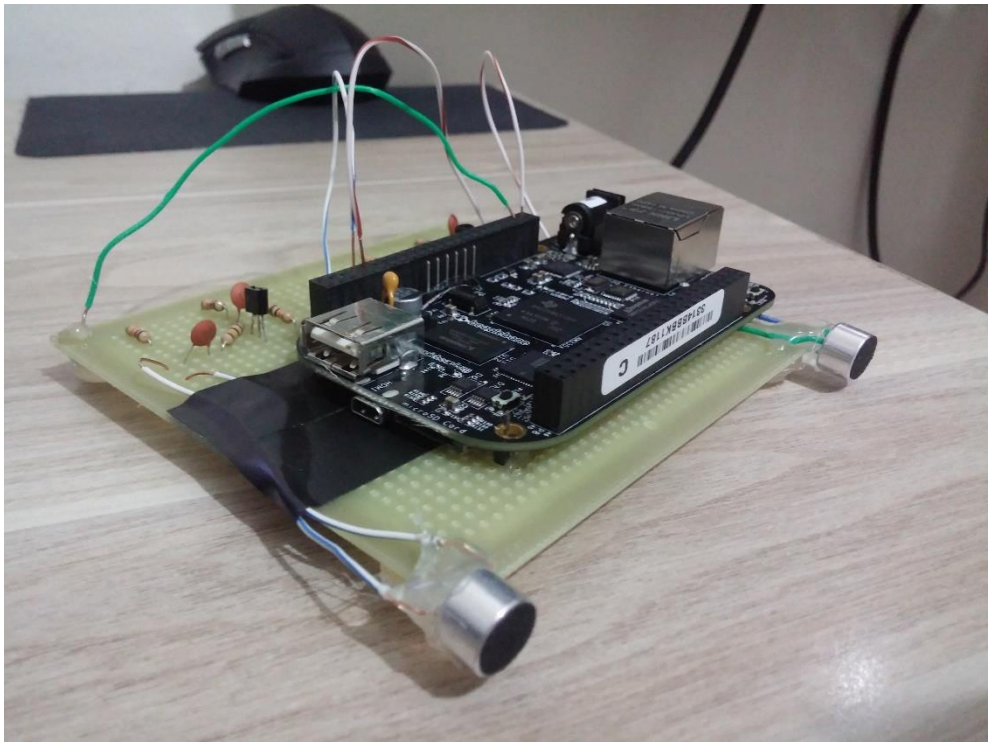


Figura 25. Sistema com arranjo linear, pré-amplificadores e microcontrolador.

Filtro FIR

Após amostrar o sinal acústico, filtra-se o sinal com um filtro FIR. Como a taxa de amostragem é variável, cada vez que o sinal é amostrado, deve-se criar um filtro para filtrar o sinal.

O filtro foi projetado para ser um filtro FIR passa banda com uma banda de 200 Hz que se estende de 1600 à 1800 Hz. O filtro possui uma ordem de 100, pois desejou-se obter o melhor resultado possível.

RESULTADOS E DISCUSSÕES

Resultado do sistema de estimação de chegada

Foi capturado uma amostra do sinal acústico capturado pelo BeagleBoneBlack, a fim de analisar a qualidade do sinal amostrado. Através da transformada de Fourier pôde-se ver as frequências presentes no sinal capturado.

Através da figura 26, verificou-se que o sinal amostrado possui um grande pico na frequência do sinal acústico da fonte geradora, de 1700 Hz.

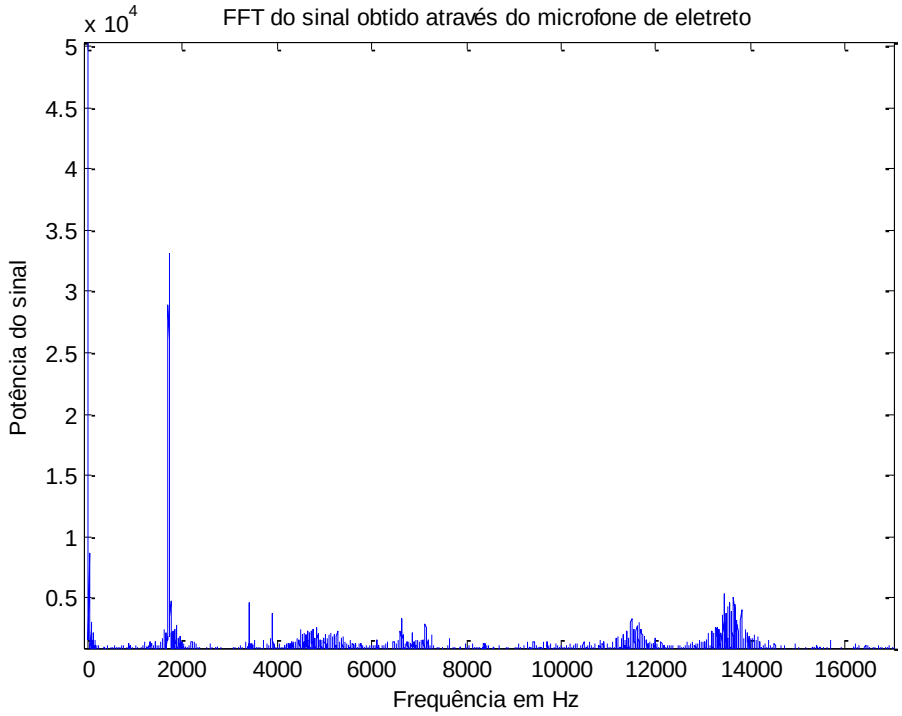


Figura 26. FFT do sinal obtido através do microfone de eletreto.

Na figura 27, é ilustrado um pico na frequência de 1700 Hz.

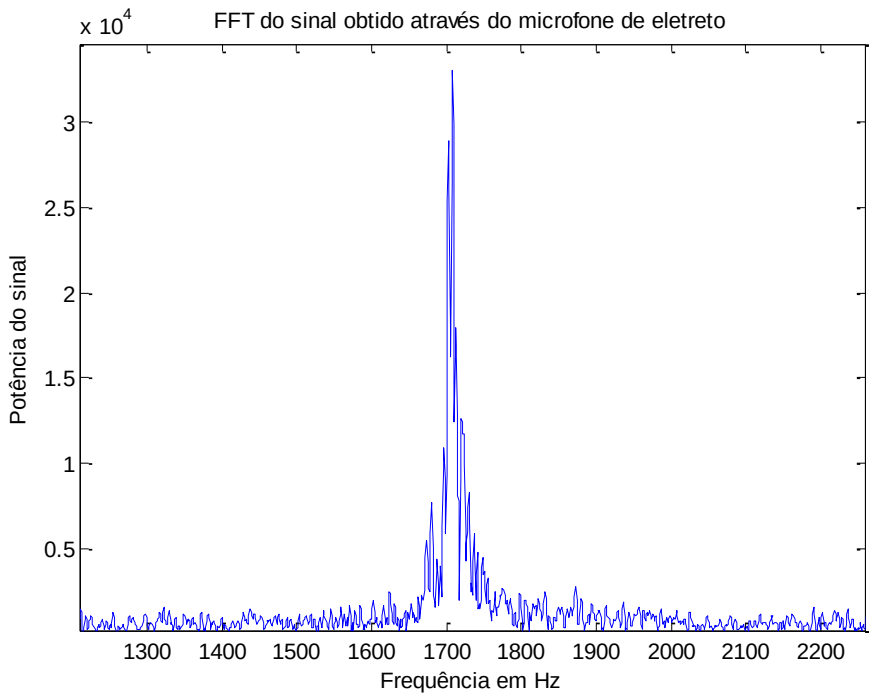


Figura 27. Pico em 1700 Hz no sinal amostrado pela placa.

Após o sinal ser amostrado, filtrou-se o sinal com um filtro FIR. A figura 28 ilustra o a FFT do sinal filtrado, verificou-se que o sinal filtrado possui praticamente apenas um pico em 1700 Hz como desejado para a obtenção do melhor desempenho do algoritmo.

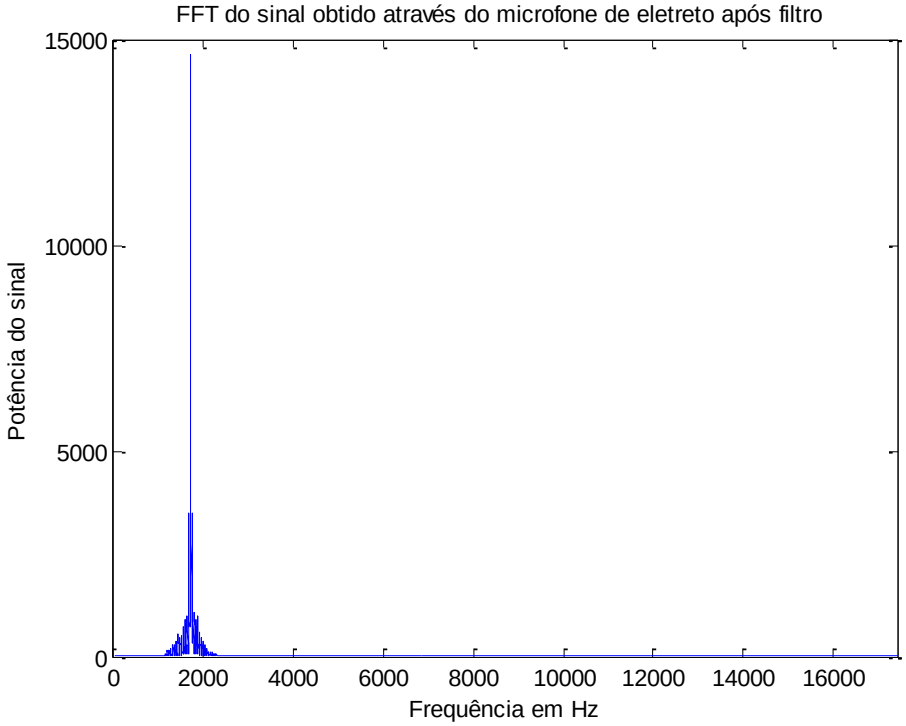


Figura 28. FFT do sinal após ser filtrado pelo filtro passa banda.

O algoritmo foi testado com todo o sistema de direção de chegada em um espaço aberto e uma fonte áudio com um sinal senoidal de 1700 Hz, dito anteriormente. A fonte de áudio foi colocada a (17±0.05) cm, respeitando os limites de campo distante. Para a estimação de direção de chegada foi utilizado um transferidor e um fio para garantir a direção desejada pelo transferidor. O método de medição é ilustrado na figura 29.

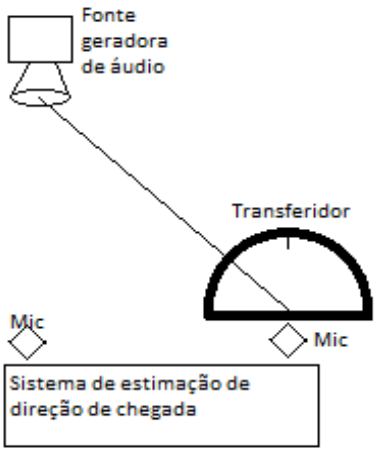


Figura 29. Método para a medição da estimação de direção de chegada.

Na figura 30, é ilustrado os resultados obtidos pelo algoritmo MUSIC no microcontrolador BeagleBoneBlack.

Ângulo Esperado (º)	Direção estimada (º)
-90±0,5	--
-80±0,5	--
-70±0,5	-75,7
-60±0,5	-50,2
-50±0,5	-43
-40±0,5	-33,2
-30±0,5	-40
-20±0,5	-16,7
-10±0,5	-0,1
0±0,5	4,3
10±0,5	16,1
20±0,5	22,9
30±0,5	32,9
40±0,5	34,1
50±0,5	47,6
60±0,5	75,7
70±0,5	80,2
80±0,5	--
90±0,5	--

Figura 30. Direção de chegada pelo sistema criado.

O erro quadrático médio da diferença entre o ângulo esperado e a direção estimada foi de 29,94º.

Entretanto, embora um erro quadrático médio grande, o sistema estimador de direção de chegada é capaz de distinguir muito bem a localização do sinal: esquerda, centro, direita.

Houveram diferenças entre a direção de chegada estimada e a direção real da fonte geradora do sinal e infelizmente houveram dificuldades para o sistema resolver direções de chegadas extremas (entre ±70º, ±80º, ±90º), devido ao número de microfones no sistema e também a capacidade desses microfones adquirir os sinais para tais ângulos.

CONCLUSÃO

Objetivo principal deste projeto foi a implementação de um sistema de direção de chegada para sinais de áudio. Os algoritmos de direção de chegada *Correlação, Beamforming, MUSIC, ESPRIT* e seus resultados foram simulados. E o processamento dos sinais adquiridos pela placa utilizada pelo microcontrolador foi feito com o algoritmo *MUSIC* pois este algoritmo representa uma implementação mais simples e ainda assim é um bom estimador de chegada.

Para as simulações e testes iniciais foram desenvolvidos programas em MATLAB que implementam todos os algoritmos desejados e simulam sinais de áudio provenientes de para cada microfone. Foram simulados arranjos lineares de dois, quatro e seis microfones.

Para o hardware construído através de um microcontrolador, construiu-se com uma placa de solda de 10 x 10 centímetros onde esta foi utilizada como placa para os dois pré-amplificadores, os dois microfones e o microcontrolador. Um dos fatores mais importantes foi a escolha do microcontrolador que fosse capaz de ser controlado através de um sistema operacional, o que ajudou no desenvolvimento de scripts do algoritmo e consequentemente na escolha da linguagem de programação, Python. Um dos grandes problemas do sistema operacional e da placa é o fato de não ser feito para funcionar em sistemas de tempo real, o que tornaria a aquisição do sinal a uma frequência razoável impossível.

A PRUSS encontrada na placa permitiu que este problema foi resolvido rapidamente, entretanto não houve tempo para a implementação na PRUSS para amostrar a uma taxa de amostragem constante, ou seja, nesta placa trabalhou-se sempre em taxas de amostragem variáveis. Para descobrir qual a taxa de amostragem que a placa estava trabalhando naquele tempo, utilizou-se um cronometro de início e fim de amostragem.

Pelo fato de possuir um sistema operacional Linux, a linguagem utilizada para no algoritmo foi Python. Python é uma linguagem muito simples, interpretada, e possui muitas bibliotecas matemáticas, que pouparam trabalho no desenvolvimento do algoritmo.

O pré-amplificador utilizado foi desenhado para ser o mais simples possível, um emissor comum, pois a placa possui restrições de tensão em seus conversores AD e por isso foi escolhido o sistema mais simples possível, garantindo um bom

funcionamento da placa e dos pré-amplificadores. Apesar de possuir apenas dois microfones, e um erro quadrático médio de 29,94º, a placa foi capaz de distinguir muito bem as direções do sinal incidente, o que já era esperado pois há muita perda de desempenho do algoritmo para sistemas com apenas dois microfones.

Como trabalho futuro, deseja-se aplicar sistemas de direção de chegada em aplicações como localização de um indivíduo através da fala. Ainda há muito o que melhorar no sistema criado neste projeto, como adicionar mais microfones no arranjo linear, ou implementar e testar outros algoritmos.

Os imprevistos enfrentados foram de bons para maior aperfeiçoamento e estudo dos métodos e algoritmos, assim como maior entendimento do problema, motivando uma maior busca de conhecimento em outras áreas não contempladas no projeto original.

BIBLIOGRAFIA

[1] - Balanis, C. A., **Teoria de Antenas: Análise e Síntese Volume II.** LTC, 2005, 3ª Edição.

[2] - Eargle, J.; **The Microphone Book, from mono to stereo to surround - a guide to microphone design and application.** Elsevier, 2004, 2ª edição.

[3] - Klein, U. ; Trinh Quoc Vo.Signal. **Direction-of-Arrival Estimation Using a Microphone Array with the Multichannel Cross-Correlation Method**.Processing and Information Technology (ISSPIT), 2012 IEEE International Symposium.

[4] - Benesty, J; Jindong C.; Yiteng H. **Microphone Array Signal Processing**. 2008, Springer.

[5] – Stoica, P; Moses, R. **Spectral Analysis of Signals**. 2005, PRENTICE HALL.

[6] – Zhizhang, C; Gopal, G; Yiqiang, Y. **Introduction to direction-of-Arrival Estimation**. 2010, ARTECH HOUSE.

[7] - Oppenhein, A. V; Schafer, R. W.; "**Sampling of continuous time signal" Discrete-time signal processing**; 2ª edição, Prentice Hall, p. 140-239

[8] - Balanis, C. A., **Advanced Engineering Electromagnetics.** 1st edition, Wiley, 1989.

[9] – Hamming, R. W., **Digital Filters**, 3ª edição, Lucent Technologies, 1989.

[10] – Oppenheim, A. V., **Signals and Systems**, 2ª edição, Prentice-Hall, 1996.

[11] – **BeagleBoard**, http://beagleboard.org/, acessado em 20/07/2015.

[12] – **Python**, https://www.python.org/, acessado em 20/07/2015.

[13] – **PRUSS,** http://elinux.org/Ti_AM33XX_PRUSSv2#Per_PRU, acessado em 20/07/2015.

[14] – **Electret Condenser Microphone,** ABM-713-RC, pro-signal. http://www.farnell.com/datasheets/1505849.pdf, acessado em 20/07/2015.

[15] – **Pré Amplificador para microfone de eletreto**, http://circuitdiagram.net/simple-audio-pre-amplifier.html, acessado em 20/07/2015.

[16] – Moraes e Silva, **Estudo e implementação de um arranjo de microfones para estimação de direção de chegada de um sinal de áudio de banda estreita,** UFRJ, 2010.

APÊNDICE

Algoritmo MUSIC

```
    %%MUSIC
    %
    %   Matriz correlação
    X = [x1 x2 x3 x4 x5 x6];

    R = X'*X;

    phs=[0:0.1:180]*(pi/180); %angulo que quero medir
    d_norm=0.25;
    M = size(R,1); mm = [0:M-1].'; % numero de antens no vetor
    %decomposicao espectral
    [V,Lam] = eig(R);
    [lambdas,idx] = sort(abs(diag(Lam)));
    [dmax,Nn] = max(diff(log10(lambdas+1e-3))); %aqui se calcula o
maximo das diferencas do vetor diagonal LAM( as diagonais viraram um
vetor). Disso pegase o maior como o numero utilizado
    Vn=V(:,idx(1:Nn)); % MxNn matriz do ruidos
    for i=1:length(phs)
       a = exp(j*2*pi*d_norm*cos(phs(i))*mm); % vetor de trans
       P(i) = 1/(a'*Vn*Vn'*a);
    end
    grau = [-90:0.1:90];
    %plot(grau,abs(P));
    [m pos] = max(abs(P));
    GrauMusic = [GrauMusic grau(pos)];

    %%FIM MUSIC
```

Algoritmo Correlação Cruzada:

```
%%CORRELACAO CRUZADA
    X = [x1 x2 x3 x4 x5 x6];
    R = X'*X;
    M = size(R,1); mm = [0:M-1].';
    phs=[0:0.1:180]*(pi/180);
    for i=1:length(phs)
       a = exp(j*2*pi*d_norm*cos(phs(i))*mm); % vetor de trans
       Xcorr(i) = sum(a'*R);
    end
    grau = [-90:0.1:90];
    [m pos] = max(abs(Xcorr));
```

```
    GrauCorr = [GrauCorr grau(pos)];
    %%FIM CORRELACAO CRUZADA
```

Algoritmo Beamforming:

```
    %%Beamforming
    X = [x1 x2 x3 x4 x5 x6];
    R = X'*X;
    phs=[0:0.1:180]*(pi/180); %angulo que quero medir
    d_norm=0.25;
    M = size(R,1); mm = [0:M-1].';
    for i=1:length(phs)
       a = exp(j*2*pi*d_norm*cos(phs(i))*mm); % vetor de trans
       Beamforming(i) = a'*R*a;
    end
    %grau = [-90:0.1:90];
    %plot(grau,abs(P));
    [m pos] = max(abs(Beamforming));
    GrauBEAM = [GrauBEAM grau(pos)];
    %FIM BEAMFORMING
```

Algortimo ESPRIT:

```
%%ESPRIT
    X = [x1'; x2'; x3'; x4'; x5'; x6'];

    dx = 0.25;%espacamento em comprimento de onda
    num_sig = 1; %numero de sinais presentes
    N = 1; %apenas uma fileira de array
    M = 6; %6 microfones

    R=X*X';                 %matriz de covariancia
    [U,D,V]=svd(R);         %decomposicao de R

    S=U(:,1:num_sig);       %computar o subespaço do sinal

    %computar angulos de chegada
    theta = S(1:max(N,M)-1,:)\S(2:max(N,M),:);
    w=angle(eig(theta));
    theta=acos(-w/(dx*2*pi));
    thetaESPRIT = theta*180/pi;
    GrauESPRIT = [GrauESPRIT thetaESPRIT];
    %%FIM ESPRIT
```

Algoritmo MUSIC implementado em BeagleBoneBlack:

```
"""
    ADC funciona nos pins listados abaixo
```

```
    "AIN4", "P9_33"
    "AIN6", "P9_35"
    "AIN5", "P9_36"
    "AIN2", "P9_37"
    "AIN3", "P9_38"
    "AIN0", "P9_39"
    "AIN1", "P9_40"

    A maxima tensao surportada eh de 1.8 V. Nao exceder.

    http://stackoverflow.com/questions/5953297/python-eigenvectors
    http://stackoverflow.com/questions/21562986/numpy-matrix-vector-multiplication
    http://www.numpy.org/
    http://stackoverflow.com/questions/8370637/complex-numbers-usage-in-python
    http://stackoverflow.com/questions/11838352/multiply-several-matrices-in-numpy

"""
import beaglebone_pru_adc as adc
import time
import numpy as np #aqui uso a biblioteca para trabalhar mais facilmente com numeros
complexos
from numpy import linalg as LA
from scipy import signal
import os

### LIMPANDO MEMORIA DO MICRO ###
os.system('sudo sh -c "sync; echo 3 > /proc/sys/vm/drop_caches"')
###



capture = adc.Capture()
capture.encoder0_pin = 0 # AIN0
capture.encoder1_pin = 2 # AIN2
capture.encoder0_threshold = 100
capture.encoder0_delay = 100
capture.cap_delay = 0
capture.encoder1_threshold = 100
capture.encoder1_delay = 100

x1 = []
x2 = []

time1 = time.time()
capture.start()
for _ in range(20000):
    #print capture.encoder0_values[0],capture.encoder1_values[0]
    x1.append(capture.encoder0_values[0])
    x2.append(capture.encoder1_values[0])
capture.stop()
time2 = time.time()
tempo = (time2-time1)*1000.0 #tempo demorado pela amostra em ms
capture.wait()
capture.close()




##################
# SALVANDO DADOS #
##################
with open('dados.csv', 'w') as f:
        for x in range(len(x1)):
                f.write(str(x1[x])+';'+str(x2[x])+ '\n')
with open('amostragem.csv', 'w') as f:
    f.write(str(tempo))



####################################
#       FILTRO FIR PASSA BANDA     #
####################################
```

```
taxa = (1000.0*20000.0)/tempo;
fs = taxa/2.0;
x = 1550.0/fs;
y = 1650.0/fs;
z = 1750.0/fs;
w = 1850.0/fs;
n = 100
taps = signal.firwin2(n,[0,x,y,z,w,1],[0,0,1,1,0,0])
x1 = signal.lfilter(taps,1,x1)
x2 = signal.lfilter(taps,1,x2)



####################################
# CRIANDO ARRAY [0:1:10] EM PYTHON #
####################################


tempo_Am = np.arange(0.0,tempo/1000.,(tempo/20000000.))
cos2 = np.cos((2*np.pi*1700)*tempo_Am)
sin2 = (-1) * np.sin((2*np.pi*1700)*tempo_Am)

x2real = x2*cos2
x2img = x2*sin2*complex(1j)
x2 = x2real+x2img

X = np.concatenate(([x1],[x2.conj()]),axis=0)



##################
#ALGORITMO MUSIC#
##################



R = reduce(np.dot,[X,X.conj().T])
R = R.conj()
d_norm = 0.5
phs = (np.pi/180)*np.arange(0.0,180.0,0.1) #angulo que quero medir
Lam,v = LA.eig(R)
mm = np.arange(0,2,1) #algoritmo para 2 microfones
mm.T
lambdas = np.sort(np.absolute(Lam))
#[dmax,Nn] = max(diff(log10(lambdas+1e-3)))
max = np.amax(np.diff(np.log10(lambdas+1e-3)))
Nn = 0
contador = 0
for element in np.diff(np.log10(lambdas+1e-3)):
    if element == max:
        break
    Nn = Nn+1


vc = np.fliplr(v)
Vn = vc[:,0:(Nn+1)]
P = []
for grau in phs:
    a = np.exp(1j*2*np.pi*d_norm*np.cos(grau)*np.array([0, 1]).T)
    P.append(1./reduce(np.dot, [a.conj().T,Vn,Vn.conj().T,a]))
P = np.abs(P)
DOA = -9999
menorDOA = P[0]
posicao_menor = 0
posicao = 0
buffer = 0
for element in P:
    if element >= DOA:
        DOA = element
        posicao = buffer
    if element <= menorDOA:
        menorDOA = element
        posicao_menor = buffer
    buffer = buffer+1
```

```
print 'Pico mais alto a: '+ str(0.1*posicao) + '\nPico mais baixo a: ' +
str(0.1*posicao_menor)


#####################
#FIM ALGORITMO MUSIC#
#####################
with open('final.csv', 'w') as f:
    for x in range(len(P)):
        f.write(str(P[x]) + '\n')
```