

UNIVERSIDADE LUTERANA DO BRASIL

PRÓ-REITORIA DE GRADUAÇÃO

CURSO DE ENGENHARIA ELÉTRICA

**DESENVOLVIMENTO DE UM SISTEMA PARA
RECONHECIMENTO DE PALAVRAS
PRONUNCIADAS ISOLADAMENTE**

Melissa Grahl Figueredo

Canoas, Dezembro de 2005.

MELISSA GRAHL FIGUEREDO

**DESENVOLVIMENTO DE UM SISTEMA PARA
RECONHECIMENTO DE PALAVRAS
PRONUNCIADAS ISOLADAMENTE**

**Trabalho de Conclusão de Curso
apresentado como requisito para a
obtenção do grau de Engenheiro
Eletricista pela Universidade Luterana
do Brasil – Campus Canoas - Curso de
Engenharia Elétrica.**

Orientador: Prof. Dr. Valner João Brusamarello

Canoas, dezembro de 2005.

MELISSA GRAHL FIGUEREDO

**DESENVOLVIMENTO DE UM SISTEMA PARA
RECONHECIMENTO DE PALAVRAS
PRONUNCIADAS ISOLADAMENTE**

Prof. Dr. Alexandre Balbinot _____

Prof. M.E. Luis Fernando Espinosa Cocian _____

ORIENTADOR:

Prof. Dr. Valner João Brusamarello _____

Trabalho apresentado e aprovado em:

Agradecimentos

São inúmeras as pessoas que me apoiaram e colaboraram para a minha evolução ao longo destes anos, mas gostaria de agradecer, principalmente, aos professores, por sua imensa paciência e tolerância para com os alunos, a minha família e aos amigos.

Não poderia deixar de agradecer a ULBRA pelas bolsas de iniciação científica das quais participei, pois através delas foi possível custear parte do curso e principalmente, adquirir muitos conhecimentos.

RESUMO

O presente trabalho propõe o desenvolvimento de um sistema para reconhecimento de voz dependente do locutor e para palavras pronunciadas isoladamente. São apresentados tópicos sobre processamento digital de sinais, onde surgem as formas de representação, transformação e manipulação de sinais. Ainda, o trabalho demonstra e emprega o método de reconhecimento *Dynamic Time Warping*. O programa desenvolvido utilizando DTW obteve um percentual de acerto de 72%, para um total de 800 amostras, utilizando um banco de dados com dezesseis palavras padrões para as comparações. Neste trabalho, também são apresentados os conceitos sobre a obtenção dos coeficientes MFCC e os respectivos resultados. Os programas foram desenvolvidos no Matlab®.

Palavras chave: Reconhecimento de voz; DTW; MFCC; Matlab.

ABSTRACT

The purpose of this paper is to develop a speaker-dependent and isolated words speech recognition system. Particularly, this paper offers a study about some conceptions, such as digital signal processing, from which emerge representation, transformation and signal manipulation. Also, this paper shows an application of Dynamic Time Warping recognition method. The software developed by using DTW, achieved 72% of correct recognition for 800 samples, using a data bank with 16 standard words. Finally, the Mel filters cepstrum coefficients was calculated as a way of representing the speech signal. In this work Matlab software was applied.

Key words: Voice Recognition; DTW; MFCC; Matlab.

SUMÁRIO

RESUMO	1
ABSTRACT	2
1 INTRODUÇÃO	8
2 FUNDAMENTAÇÃO TEÓRICA	12
2.1 CONCEITOS BÁSICOS	13
2.2 ANÁLISE DA VOZ	19
2.2.1 Medidas de Energia	20
2.2.2 <i>Dynamic Time Warping</i>	22
2.2.3 Coeficientes MFCC	24
2.2.4 Bancos Mel <i>Frequency</i>	24
2.2.5 Delta MFCC's	26
3 METODOLOGIA APLICADA	28
3.1 DYNAMIC TIME WARPING	28
3.1.1 Aquisição de Dados	29
3.1.2 Detecção dos limites da palavra	30
3.1.3 Identificação da Palavra	34
3.1.4 Interface Gráfica e Resultado	38

3.1.5	Função PRINCIPAL	40
3.1.6	Palavras Padrões	43
3.2	COEFICIENTES MEL CEPSTRAIS	45
3.2.1	Pré-ênfase	46
3.2.2	Janelamento e DFT	47
3.2.3	Banco de Filtros Mel	49
3.2.4	Coeficientes Mel Cepstrais	53
4	DISCUSSÃO DOS RESULTADOS.....	58
4.1	DYNAMIC TIME WARPING	58
4.1.1	Análise Estatística.....	61
4.2	COEFICIENTES MEL CEPSTRAIS	63
4.3	CONSTATAÇÕES E ALTERAÇÕES	65
5	CONCLUSÕES	67
6	REFERÊNCIAS BIBLIOGRÁFICAS.....	69
	ANEXO 1: LISTAGEM DOS PROGRAMAS.....	71
	ANEXO 2: FAIXA DE FREQUÊNCIAS AUDÍVEIS.....	92
	ANEXO 3: DADOS TÉCNICOS SOBRE OS EQUIPAMENTOS.....	94
	ANEXO 4: TESTES DETECÇÃO DOS LIMITES.....	104

LISTA DE FIGURAS

Figura 1 - Efeito <i>Aliasing</i> . Fonte: Iazzetta.....	15
Figura 2 - Janelas Blackman, Hamming, Bartlett e Hanning para 1024 pontos.....	17
Figura 3 - Aparelho fonador. Fonte: Costa e Costa, 2005.....	19
Figura 4 - (a) Gráfico da palavra /sete/ (b) Gráfico da Energia.....	21
Figura 5 - DTW (a) alinhamento entre as amostras (b) alinhamento local. Fonte: Tebelskis.....	23
Figura 6 - Diagrama Ilustrativo de um Banco de Filtros. Fonte: Becchetti e Ricotti.....	26
Figura 7 - Etapas para o Reconhecimento de Voz.....	28
Figura 8 - Gráfico das Etapas da detecção dos limites para a palavra cinco.....	31
Figura 9 - Fluxograma dos passos principais da função de detecção dos limites.....	33
Figura 10 - Exemplo de um Espectrograma para a palavra /nove/.....	36
Figura 11 – Alinhamento: (a) palavras diferentes (b) palavras iguais. Fonte: Harris.....	37
Figura 12 - Tela dos Resultados.....	38
Figura 13 - Fluxograma da função <i>principal.m</i>	42
Figura 14 - Diagrama de blocos das etapas para obtenção dos Coeficientes MFCC.....	45
Figura 15 - Palavra /zero/ e sua versão após a pré-ênfase.....	47
Figura 16 - Espectrograma da palavra /zero/ após pré-ênfase.....	48
Figura 17 - Gráfico da Distribuição das Frequências.....	50
Figura 18 - Fluxograma: etapas principais filtro triangular.....	51
Figura 19 - Exemplo da resposta para o filtro 22.....	52
Figura 20 - Gráficos das saídas das etapas implementadas.....	56
Figura 21 - Delta Coeficientes Cepstrais.....	57
Figura 22 - (a) Comparativo entre duas palavras /cinco/ (b) Detalhe da palavra /cinco/ aquisição com placa <i>onboard</i>	60

Figura 23 - Percentual de acertos para amostras aleatórias.	61
Figura 24 - Percentual de acertos para amostras seqüenciais.	62
Figura 25 - Comparativo entre os coeficientes de uma cossenóide de 500Hz.	64
Figura 26 - Coeficientes da cossenóide de 500Hz sem a pré-ênfase.	65
Figura 27 – Nova função principal.	66

LISTA DE ABREVIATURAS E SIGLAS

DCT	<i>Discrete Cosine Transform</i> – Transformada Discreta do Cosseno
DTW	<i>Dynamic Time Warping</i>
MFCC	<i>Mel Frequency Cepstrum Computation</i> – Coeficientes Mel Cepstrais
SIM	<i>Symetric Inverse Matrix</i> - Matriz Simétrica Inversa

1 INTRODUÇÃO

A medida em que novas tecnologias são desenvolvidas, as pessoas são beneficiadas com facilidades que outrora pertenciam a algum filme de ficção, mas que atualmente estão incorporados ao dia-a-dia. Dentre estas tecnologias pode-se citar o processamento digital de sinais, cujas aplicações abrangem vários segmentos da engenharia, que vão desde aplicações biomédicas complexas, até os corriqueiros telefones celulares.

O reconhecimento de voz, consequência da possibilidade do processamento digital de sinais, também tem se mostrado bastante usual atualmente, pois torna possível a interação entre o ser humano e a máquina, através do acionamento via voz. As interfaces para computadores acionadas via voz, são um tópico que tem atraído e fascinado a comunidade científica. Para muitos, a habilidade de conversar livremente com uma máquina representa o último desafio para a compreensão dos processos de produção e percepção envolvidos na comunicação da linguagem humana. Acionamentos via voz, além de ser um tópico estimulante, está se tornando rapidamente uma necessidade.

Reconhecer uma palavra pronunciada incorpora muitas tecnologias e aplicações diferentes. Em alguns casos, o interesse não reside no conteúdo da linguagem subjacente, mas sim na identidade do orador ou da linguagem utilizada. O reconhecimento de voz pode envolver a identificação de um orador específico de uma dada população conhecida ou verificar a identidade de um utilizador, permitindo dessa forma acesso controlado a locais e serviços.

Os processos de reconhecimento de voz usam métodos para padronizar o reconhecimento, correlações, medidas de distância e quantização dos vetores. Já os métodos para reconhecer palavras isoladas usam *Dynamic Time Warping* para alinhar as palavras pronunciadas em diferentes velocidades, modelos de Markov, entre outros.

O objetivo geral é desenvolver um sistema para o reconhecimento de voz. Especificamente, deseja-se reconhecer 16 palavras da língua portuguesa, que incluem os números de /zero/¹ à /nove/, as operações matemáticas básicas, representadas pelas palavras /mais/, /menos/, /vezes/ e /dividido/, a palavra /igual/, e por fim, a palavra /limpa/.

O sistema de reconhecimento de voz proposto será dependente do locutor e servirá apenas para reconhecer palavras isoladas, descartando-se desta forma as falas contínuas. Entende-se como palavra isolada, a pronúncia de uma única palavra em um intervalo de tempo pré-

¹ Indicações entre “/” caracterizam palavras que se encontram no formato de som.

definido. Sabendo-se que os sistemas de reconhecimento de voz independentes do locutor são deveras mais complexos de implementar-se do que os sistemas dependentes do locutor, optou-se pela simplicidade. Embora o termo “dependente do locutor” conduza a condição na qual o sistema reconhece apenas a voz do autor, não se considera um problema e/ou erro, se o sistema em questão reconhecer também outro locutor, visto que a proposta do trabalho não é identificar o mesmo.

Será utilizado o programa Matlab® e suas ferramentas, para o desenvolvimento do sistema. Para a detecção dos limites da palavra será utilizado o método da medida de energia e para o alinhamento e reconhecimento o método será o *Dynamic Time Warping*.

O sistema de reconhecimento voz será aplicado na implementação de uma calculadora, na qual será possível realizar as quatro operações básicas (soma, subtração, multiplicação e divisão), através de instruções verbais do locutor em questão.

O reconhecimento de voz apresenta inúmeras vantagens, pois facilita a interação entre homem e a máquina, sem a necessidade de acionamentos via contato físico. Vantagem essa, que pode ser empregada em sistemas para garantir o acesso de deficientes físicos, por exemplo.

O emprego do programa Matlab® como ferramenta no desenvolvimento do projeto, valida-se, pois oferece inúmeras funções matemáticas, entre outros benefícios, que facilitam o desenvolvimento do trabalho.

Baseando-se na tradição e relativa simplicidade do método de reconhecimento *Dynamic Time Warping*, este não poderia ficar de fora deste estudo, visto que é um dos métodos mais antigos e pode ser aplicado ao reconhecimento de palavras isoladas.

2 FUNDAMENTAÇÃO TEÓRICA

A vida baseia-se em grandezas físicas e químicas cujas informações são transportadas através de sinais, que podem ser medidos, processados, analisados, e que dão origem a decisões. O som, a temperatura e a luz são exemplos de grandezas físicas, cujos sinais são utilizados no dia-a-dia. Os ouvidos convertem o som em sinais elétricos, que chegam ao cérebro, e este é capaz de analisar algumas das suas propriedades, tais como amplitude, frequência e fase, determinar a direção em que se encontra a fonte de som, e reconhecê-lo, como música, fala ou o ruído.

O desenvolvimento que nos últimos anos se verificaram na microeletrônica, tornaram possível pôr efetivamente em uso o Processamento Digital de Sinal, que hoje é essencial nas telecomunicações, na instrumentação, na engenharia biomédica, entre tantos outros segmentos da engenharia.

Como este trabalho trata de processamento do sinal de voz, serão lembrados, de maneira sucinta, alguns conceitos a serem aplicados, e que podem ser vistos nos tópicos a seguir.

2.1 CONCEITOS BÁSICOS

Período de Amostragem

O período de amostragem, nada mais é do que o inverso da frequência utilizada para amostrar o sinal desejado.

$$Ts = \frac{1}{fs} \quad (1)$$

onde $fs[Hz]$ e $Ts[s]$ são, respectivamente, a frequência e o período de amostragem.

Número De Amostras

A equação (2) relaciona o número de amostras do sinal com o período de amostragem, o que resulta no tempo da aquisição.

$$t = n \cdot Ts \quad (2)$$

onde n é o número de amostras, $Ts[s]$ o período de amostragem e $t[s]$ o tempo total da amostra.

Redução da Amostragem

O termo em inglês *downsampling* significa diminuir a amostragem de um sinal e é dado pela Equação (3).

$$y[n] = x[Mn] \quad (3)$$

onde x é o sinal original e y será o sinal resultante M vezes menor que a original. Sendo M um número inteiro e positivo.

Teorema de Nyquist

Segundo IAZZETTA, a taxa de amostragem deve ser pelo menos duas vezes maior que a frequência que se deseja registrar. Esse valor é conhecido como frequência de Nyquist. Ao se tentar reproduzir uma frequência maior do que a frequência de Nyquist ocorre um fenômeno chamado *aliasing*, onde a frequência é "espelhada" ou "rebatida" para uma região mais grave do espectro. A Equação 4 demonstra a relação de Nyquist:

$$f_s > 2f_N \quad (4)$$

onde $f_s[Hz]$ e $f_N[Hz]$ são, respectivamente, as frequências de amostragem e de Nyquist.

A Figura 1 representa uma onda de 25.725 Hz (menor período) digitalizada com uma taxa de amostragem de 44.100 Hz. Cada amostra é representada pelos pontos brancos. A onda com maior período é a onda resultante do efeito *aliasing*.

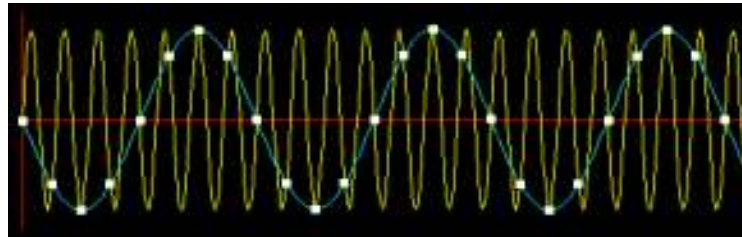


Figura 1 - Efeito *Aliasing*. Fonte: Iazzetta.

Nota-se que a forma de onda em azul na Figura 1, não representa a forma de onda original, que é a em amarelo. Esse resultado é consequência do não cumprimento do teorema de Nyquist.

Transformada de Fourier

A Transformada de Fourier é extremamente útil para análise de dados, pois representa um sinal qualquer através de componentes senusoidais em diferentes frequências.

Para dados amostrados, a análise de Fourier é realizada usando-se a Transformada Discreta de Fourier (DFT – *Discrete Fourier Transform*), conforme a equação a seguir:

$$X(k) = \sum_{n=1}^N x(n) \cdot w_N^{(n-1)(k-1)} \quad (5)$$

onde $x(n)$ é o sinal digital e $w_N = e^{(-2\pi i)/N}$.

Pré-ênfase

Conforme BECHETTI, as características do trato vocal definem o fonema pronunciado. Tais características são evidenciadas no domínio da frequência pela locação dos *formants*, isto é, os picos ressonantes do trato vocal onde há uma grande concentração de energia. Apesar de possuir informação relevante, as altas frequências possuem amplitudes menores, se comparadas às amplitudes das baixas frequências. A pré-ênfase das frequências altas torna as amplitudes mais homogêneas, já que existem informações importantes tanto nas frequências baixas como nas altas. A implementação da pré-ênfase pode ser realizada através de um filtro digital de primeira ordem, conforme a seguinte equação no domínio do tempo:

$$y(n) = x(n) - a \cdot x(n-1) \quad (6)$$

onde $y(n)$ é a resposta do filtros, $x(n)$ o sinal de entrada e a é o parâmetro responsável pela pré-ênfase.

Segundo BECCHETTI, fazendo $a = 0,95$, haverá um aumento de mais de 20dB para as altas frequências.

Janelamento

Nas aplicações práticas de processamento de sinais, é necessário trabalhar em termos de *frames* do sinal, ou seja, segmentar o sinal. Nesse caso é necessário selecionar uma porção de sinal que possa ser razoavelmente assumida como estacionária. Formalmente, defini-se um

frame de voz como sendo o produto de uma janela discreta $w(n)$ com a seqüência de voz discreta (pré-enfatizada) $y(n)$, conforme a equação (7) a seguir:

$$x_t(n) = y(n) \cdot w(n) \quad (7)$$

onde $x_t(n)$ é sinal janelado resultante nos *frame* t .

A escolha da janela é importante, pois determina a qualidade dos sinais. Tem-se diversas janelas, tais como as janelas de Bartlett (triangular), retangular, Blackman, Hamming, Hanning, entre outras.

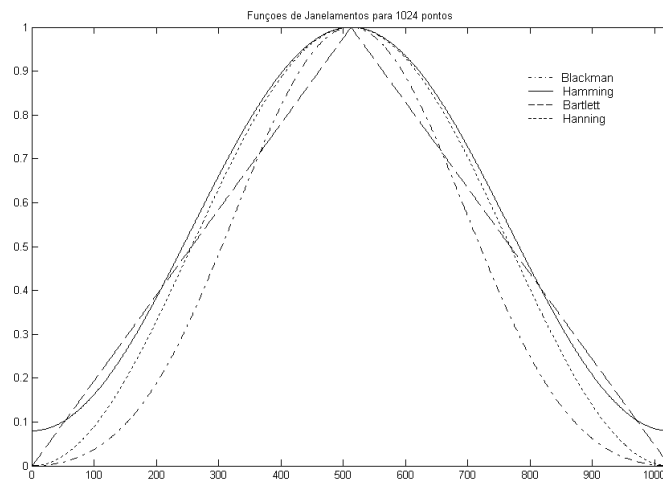


Figura 2 - Janelas Blackman, Hamming, Bartlett e Hanning para 1024 pontos.

A Figura 2 exibe o formato de quatro tipos janela de tamanho 1024. As expressões que definem essas janelas estão descritas a seguir:

- Janela de Blackman:

$$w_{Blackman}(k+1) = 0,42 - 0,5 \cdot \cos(2\pi \cdot \frac{k}{n-1}) + 0,08 \cdot \cos(4\pi \cdot \frac{k}{n-1}), \quad k = 0, \dots, n-1 \quad (8)$$

- Janela de Hamming:

$$w_{Hamming}(k+1) = 0,54 - 0,46 \cdot \cos(2\pi \cdot \frac{k}{n-1}), \quad k = 0, \dots, n-1 \quad (9)$$

- Janela de Bartlett:

$$w_{Bartlett}(k+1) = \begin{cases} \frac{2(k)}{n-1}, & 0 \leq k \leq \frac{n}{2}-1 \\ \frac{2(n-k-1)}{n-1}, & \frac{n}{2} \leq k \leq n-1 \end{cases} \quad (10)$$

- Janela de Hanning:

$$w_{Hanning}(k+1) = 0,5 \cdot \left[1 - \cos(2\pi \cdot \frac{k}{n-1}) \right], \quad k = 0, \dots, n-1 \quad (11)$$

onde, n é um número inteiro e define o tamanho da janela para as Equações (8), (9), (10) e (11).

Aplicando-se uma função de janelamento ao sinal que se deseja segmentar, é evitada a introdução de ruídos em frequência, devido à segmentação. Este fenômeno é denominado de *leakage*.

2.2 ANÁLISE DA VOZ

Todos os sons produzidos durante a geração da voz, são provenientes do aparelho fonador (Figura 3), que é responsável pela geração, articulação e radiação do som para o ambiente externo. Cada indivíduo possui características singulares, que são responsáveis por algumas restrições no reconhecimento de voz, tais como frequência do sinal, amplitude e velocidade na pronúncia.

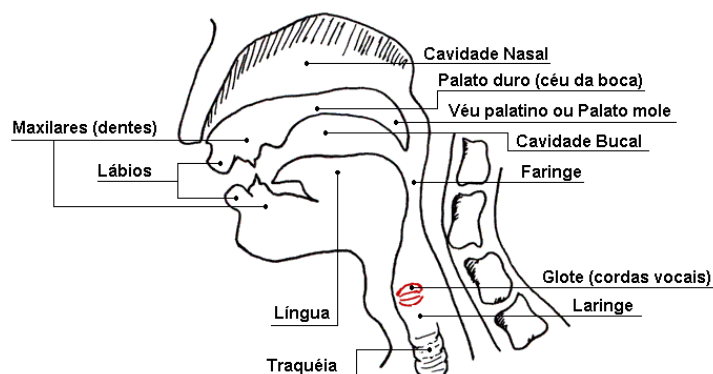


Figura 3 - Aparelho fonador. Fonte: Costa e Costa, 2005.

Partindo-se destas características singulares, torna-se necessário buscar formas para representar digitalmente os sinais de voz. Para tanto, são propostos métodos para essa representação, tais como a medida de energia, a medida de cruzamento por zero (*zero-crossing*), a análise espectral, a codificação linear preditiva (LPC), *dynamic time warping* (DTW), modelos de Markov, entre outros. A seguir podem ser observados os métodos que serão empregados no trabalho.

2.2.1 Medidas de Energia

Existem muitos métodos para representação digital de um sinal de voz. Estes métodos se propõem a representar o sinal da melhor maneira possível, de forma que seja viável reconstruir o sinal original. Para reconhecimento de voz não há necessidade de reconstruir o sinal, mas sim de representa-lo de maneira que se possa através dos parâmetros resultantes identificar o sinal.

O sinal de voz pode ser representado através de sua energia $E[n]$, conforme a seguinte equação:

$$E[n] = \sum_{m=0}^{N-1} [w[m] x[n-m]]^2 \quad (12)$$

onde $w[m]$ é a janela aplicada ao sinal $x[n]$, e N é o número de amostras da janela.

De acordo com LUFT, a escolha de N deve ser adequada, pois se o mesmo for muito pequeno, $E[n]$ apresentará muitas flutuações, do contrário, se N for muito grande, $E[n]$ terá variações imperceptíveis. Em ambas situações, $E[n]$ não representará de forma adequada o sinal. A fim de representar o sinal de voz da melhor maneira, costuma-se usar janelas com períodos na ordem de 10ms à 20ms.

O cálculo de $E[n]$ dá ênfase aos sinais de maior amplitude, e pode ser usado para separar sons vozeados dos não-vozeados, como também para encontrar os limites da palavra. Em se

tratando de encontrar os limites da palavra, escolhe-se um limiar de energia para o qual o sinal será considerado silêncio e, por meio de comparações seleciona-se somente o sinal de voz. Para exemplificar, a Figura 4 demonstra o gráfico da palavra /sete/ (a), assim como o gráfico da Energia (b).

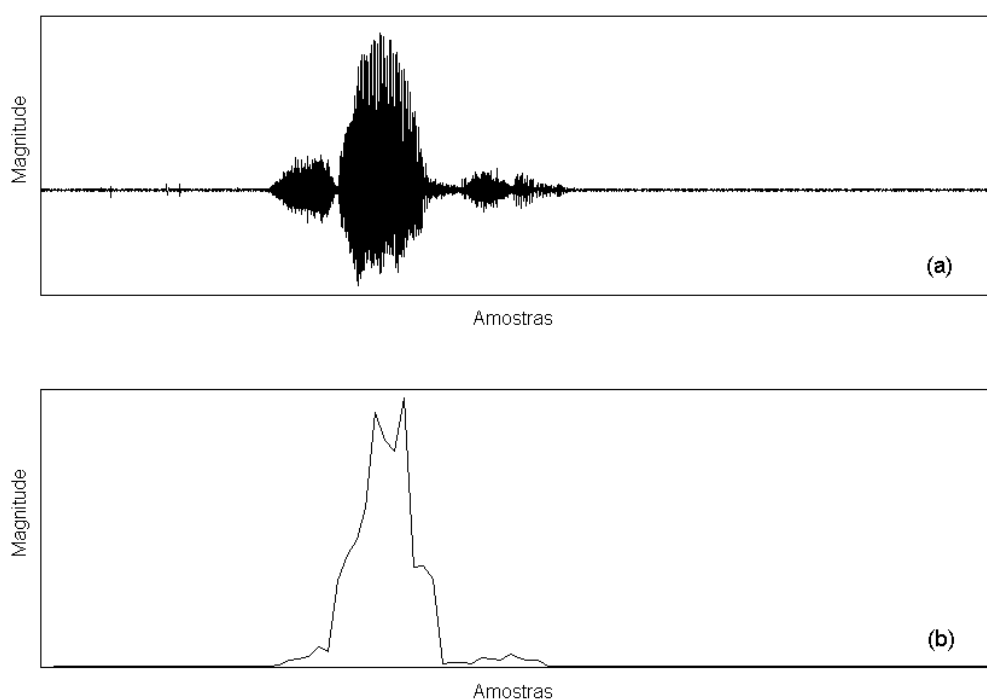


Figura 4 - (a) Gráfico da palavra /sete/ (b) Gráfico da Energia

Cabe ressaltar que a medida de energia sozinha, não se presta ao reconhecimento automático de voz, visto que o parâmetro resultante é desprovido de informações consistentes, pois é suscetível ao locutor e ao ambiente. Porém, se a medida de energia for utilizada em conjunto com outros parâmetros, poderá ser útil.

2.2.2 *Dynamic Time Warping*

DTW é um dos mais antigos e importantes algoritmos para o reconhecimento de voz. Uma das maneiras mais simples de reconhecer uma palavra isolada é compará-la a um padrão e determinar qual é a melhor combinação. Isso é complicado por inúmeros fatores. Primeiro porque palavras diferentes têm durações diferentes, o que pode ser eliminado normalizando-se a palavra padrão e a desconhecida para que tenham a mesma duração. Porém existe outro problema, a velocidade da pronúncia pode não ser a mesma, o que resultaria em uma não linearidade entre o padrão e a palavra a ser reconhecida. DTW é um método eficiente para encontrar o melhor alinhamento desta não linearidade.

DTW é um exemplo geral de uma classe de algoritmos conhecida como “*Dynamic Programming*” – Programação Dinâmica. Essa complexidade em relação ao tempo e espaço é meramente linear na duração da palavra amostrada e no tamanho do vocabulário. O algoritmo faz uma simples passagem através da matriz dos valores dos *frames*, enquanto calcula localmente, melhorando assim os segmentos do alinhamento global. Sendo $D(x,y)$ a distância Euclidiana entre o frame x da palavra amostrada, y a referência e $C(x,y)$ o resultado cumulativo do melhor alinhamento, tem-se:

$$C(x,y) = \min(C(x-1,y), C(x-1,y-1), C(x,y-1)) + D(x,y) \quad (13)$$

O alinhamento resultante pode ser visualizado como a menor distância Euclidiana, através do traçado sinuoso da matriz (Figura 5), começando em (0,0) e terminado no ponto final (x,y).

Mantendo a tendência dos pontos anteriores, o alinhamento completo pode ser recuperado traçando-se para trás, partindo-se do ponto (x,y). O melhor alinhamento é calculado para cada palavra referência e, o menor valor acumulado é considerado a palavra que mais se aproxima à palavra padrão.

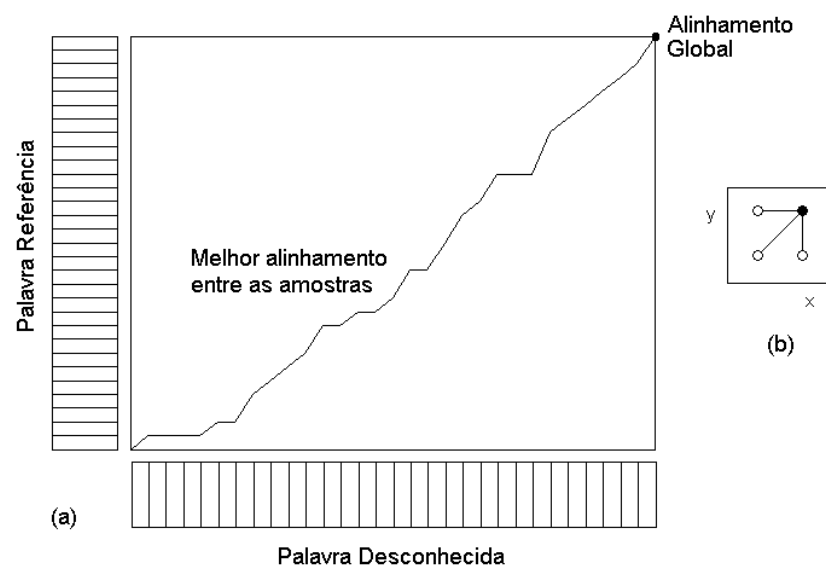


Figura 5 - DTW (a) alinhamento entre as amostras (b) alinhamento local. Fonte: Tebelskis.

Existe uma variação particularmente importante do DTW, que é uma extensão do reconhecimento de palavras isoladas para o reconhecimento de fala contínua. Essa extensão denomina-se *One Stage DTW algorithm*, e foi citada aqui apenas para conhecimento, já que não é do escopo do trabalho reconhecer falas contínuas.

2.2.3 Coeficientes MFCC

Uma das maneiras de se representar um sinal de voz é através dos dados provenientes da sua análise espectral. Tais dados revelam características relacionadas com o formato do trato vocal. Métodos como os do banco de filtros, Transformada de Fourier e predição linear podem ser empregados para tal finalidade.

O cálculo dos MFCC envolve o uso de um banco de filtros espaçados na escala Mel e o cálculo do logaritmo da energia na saída de cada filtro, seguido ainda, de uma Transformada Discreta do cosseno, como demonstra a equação a seguir:

$$c(k) = \sum_{k=1}^M \log_{10} X(k) \cdot \cos\left(n \cdot \left(k - \frac{1}{2}\right) \cdot \frac{\pi}{M}\right), \quad 1 \leq n \leq N \quad (14)$$

onde $X(k)$ é a energia da saída no k -ésimo filtro, M é o número de filtros e N é o número de coeficientes.

2.2.4 Bancos Mel *Frequency*

Sabendo-se que o sistema auditivo humano não está distribuído linearmente ao longo das frequências, a escala Mel define uma escala psicoacústica da sensibilidade do ouvido para diversas frequências do espectro audível. Um *mel* é uma unidade de medida de frequência percebida para uma determinada frequência de entrada. A equação 15 demonstra a relação entre a frequência recebida (f_{Hz}) e a frequência percebida (f_{mel}):

$$f_{mel} = 1000 \cdot \frac{\ln(1 + f_{Hz}/700)}{\ln(1 + 1000/700)} \quad (15)$$

onde f_{Hz} é a frequência em Hertz e f_{mel} é a frequência eu deseja-se encontrar na escala *Mel*.

Verifica-se que o desempenho de sistemas de reconhecimento de voz amplia-se com uso da escala Mel, associada ao uso dos bancos de filtros com uma função triangular. Esta função triangular encontra-se centrada na frequência principal e possui amplitude 1 neste ponto.

Segundo MAFRA os filtros têm suas frequências centrais espaçadas linearmente de 100*mel* ou mais, cobrindo o espectro entre 200Hz e 7kHz, onde a maior quantidade de energia se concentra.

Já BECCHETTI sugere 24 filtros onde os 10 primeiros são linearmente espaçados até a frequência de 1kHz. Acima de 1kHz os 14 filtros restantes são distribuídos conforme a aproximação sugerida pela equação a seguir:

$$\Delta_m = 1,2 \cdot \Delta_{m-1} \quad (16)$$

onde Δ_m é a banda de frequências em Hertz e m o filtro que representa.

A Figura 6 demonstra um diagrama ilustrativo para um banco de filtros de tamanho M, assim como as supostas faixas para as frequências de saídas indicadas pelos gráficos. O sinal

de entrada, indicado por Y, é resultado das etapas anteriores do processamento, que incluem a pré-ênfase do sinal, o seu janelamento e o cálculo da DFT.

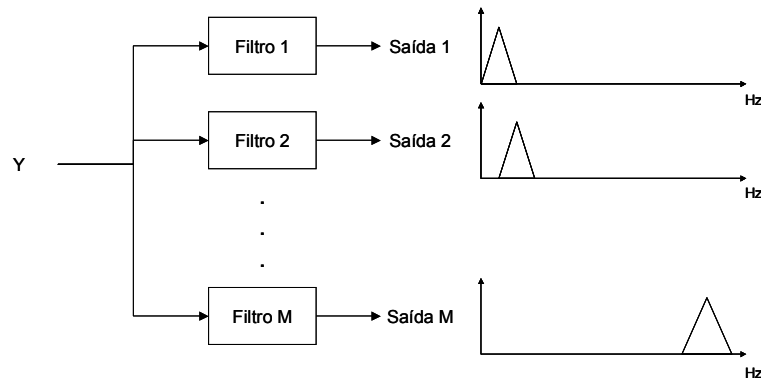


Figura 6 - Diagrama Ilustrativo de um Banco de Filtros. Fonte: Becchetti e Ricotti.

2.2.5 Delta MFCC's

O cálculo do delta MFCC ou delta cepstrum, disponibiliza informação temporal sobre a variação dos coeficientes MFCC, e é resultado da variação entre o coeficiente do segmento (*frame*) atual e um, ou dois coeficientes precedentes ou conseqüentes. Este parâmetro é útil, pois diferenças de energia são vistas entre diferentes fonemas.

Seja δw a distância entre o segmento atual w e os segmentos precedente e subseqüente considerados para diferenciação. Então o delta MFCC da componente m no segmento w é definido pela Equação 17 a seguir:

$$delta_{w,m} = MFCC_{(w+\delta w),m} - MFCC_{(w-\delta w),m} \quad (17)$$

Para $\delta w=1$, $w=1$ e $W=w$, têm-se:

$$\delta_{1,m} = \frac{MFCC_{2,m} - MFCC_{1,m}}{2} \quad (18a)$$

$$\delta_{W,m} = \frac{MFCC_{W,m} - MFCC_{(W-1),m}}{2} \quad (18b)$$

Sendo os deltas MFCC considerados como novos os coeficientes MFCC, serão então agregados à matriz dos coeficientes MFCC que terá seu tamanho dobrado, conforme indica a Equação 19.

$$B = \begin{bmatrix} C_{2,1} & \dots & C_{2,n} \\ \vdots & & \\ C_{17,1} & & C_{17,n} \\ D_{1,1} & \dots & D_{1,n} \\ \vdots & & \\ D_{16,1} & & D_{16,n} \end{bmatrix} \Rightarrow \begin{bmatrix} b_{1,1} & \dots & b_{1,n} \\ \vdots & & \\ b_{32,1} & & b_{32,n} \end{bmatrix} \quad (19)$$

A matriz da Equação 19 é composta por 16 coeficientes Mel cepstrais indicados pela letra C e por mais 16 deltas coeficientes D . Desta forma, a matriz final B possuirá 32 coeficientes b como pode ser visto.

3 METODOLOGIA APLICADA

3.1 DYNAMIC TIME WARPING

O conjunto do programa desenvolvido encontra-se dividido em funções, a fim de facilitar tanto a criação, como o entendimento geral. Cada uma das funções representa uma parte do processo e encontram-se descritas neste capítulo.

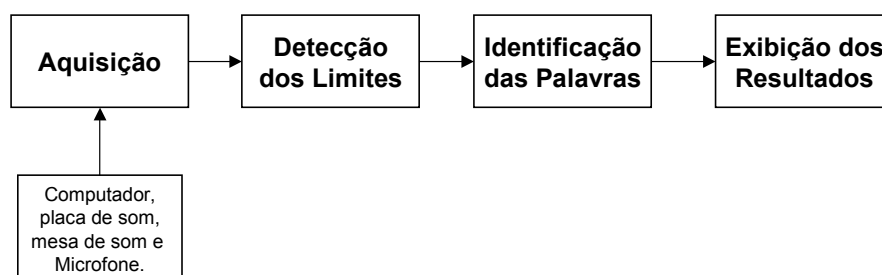


Figura 7 - Etapas para o Reconhecimento de Voz

A Figura 7 demonstra as principais etapas que o sistema percorre. Inicialmente tem-se a aquisição da voz, posteriormente a detecção dos limites da palavra pronunciada, para que então o dado passe para o processo de identificação. Identificada a palavra, o sistema exibe o resultado em uma janela.

Após desenvolver as funções necessárias para a aquisição e detecção dos limites das palavras, foram gravadas as palavras padrões para a comparação. Tendo os padrões estabelecidos, foram realizados testes com a função responsável pelo reconhecimento, finalizando assim o corpo principal do programa. Para fins de visualização, ainda foi desenvolvida mais uma função que exibe as palavras na tela, assim como a resposta da operação realizada.

3.1.1 Aquisição de Dados

Esta função é bem simples, pois emprega somente a função *wavrecord* do Matlab. Embora neste caso fosse possível empregar a função diretamente no corpo do arquivo principal, optou-se pela criação da função para que a etapa referente à aquisição fosse facilmente identificada por aqueles que desconhecem, tanto o Matlab como as suas funções.

Para utilizar a função *wavrecord*, é necessário que exista uma placa de som instalada no computador, senão o Matlab® gerará um erro. Além disso, alguns parâmetros devem ser estabelecidos, tais como a frequência de amostragem e o tempo de aquisição. As características da função e seus parâmetros são:

- Nome do arquivo: *aquisicao_voz.m*
- Nome da função: $[y, fs] = \text{aquisicao_voz}$
- Variáveis de entrada: *não requer*.
- Variáveis de saída: $[y, fs]$
 - *y*: vetor com os dados amostrados
 - *fs*: frequência de amostragem

- Funções utilizadas:
 - $wavrecord(t, fs, fs)$
- Valores estabelecidos:
 - $t = 2.5$ segundos.
 - $fs = 44100$ Hz.
 - 16 bits/amostra (valor padrão)

Sabendo-se que a frequência de amostragem é $fs = 44100Hz$ e que o valor da máxima frequência da voz humana (ANEXO 2) é $f_N = 10000Hz$ e, utilizando a equação (4), nota-se que o teorema de Nyquist foi satisfeito, já que $44100Hz$ é maior que $2 \cdot 10000Hz$.

3.1.2 Detecção dos limites da palavra

Como deseja-se processar somente os valores que tragam informações pertinentes, é importante detectar os limites da palavra pronunciada, eliminando assim os valores desnecessários.

Para tanto, criou-se uma função que detecta os limites da palavra, utilizando o método da medida de energia, conjuntamente a outras considerações.

A Figura 8 ilustra, a palavra /cinco/ normalizada, o gráfico da energia e por fim, o gráfico da palavra limitada. Embora no gráfico da palavra limitada exista o traçado além dos limites encontrados, este serve apenas para que se possa visualizar as três situações conjuntamente, já que o vetor retornado pela função é, obrigatoriamente, menor do que o vetor original, o que

não possibilitaria uma boa demonstração dos limites no gráfico. O vetor após a detecção dos limites é menor, pois dele foram retirados os “silêncios”, que não são valores que contenham informações pertinentes.

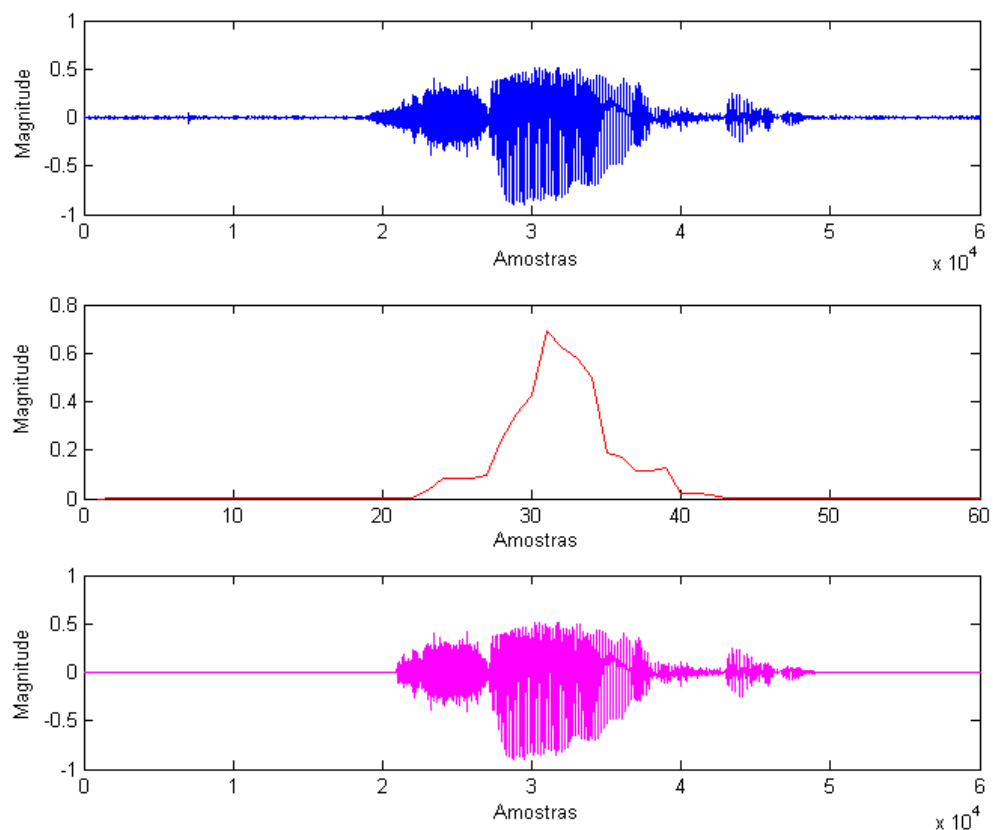


Figura 8 - Gráfico das Etapas da detecção dos limites para a palavra cinco.

Ao observar o gráfico da energia, pode-se imaginar duas formas para a detecção dos limites, uma na qual avaliam-se os valores a partir do início do vetor até o fim e outra na qual parte-se do valor máximo para as extremidades.

Desenvolveu-se a segunda opção, na qual encontra-se o valor máximo da energia e segue-se comparando, a um valor padrão, para a direita e depois para a esquerda, até que os limites

sejam encontrados e indiquem os índices para o vetor resultante. Ao escolher este procedimento foram levadas em conta duas vantagens, primeiro, não se empregaria tempo comparando valores que posteriormente seriam excluídos e, segundo, seria possível esquivar-se de ruídos que possuísem amplitude suficiente para serem considerados, durante a comparação, limite da palavra.

Antes de iniciar qualquer comparação, foi preciso normalizar todos os dados. Isso é feito através da função denominada *normaliza*, que normaliza os valores em relação ao máximo valor absoluto do vetor e depois multiplica por 0,9, a fim de evitar uma advertência do Matlab® em relação a amplitudes com valor 1,0.

Após a normalização, segue a implementação da equação (12), na qual é preciso definir a janela a ser aplicada. Estabelecendo-se, na equação (2), $n = 1000$ e, sabendo que a frequência de amostragem é 44100Hz e que o período é obtido com (1), tem-se $t = 22,67ms$, que é um valor superior aos valores sugeridos na literatura, que ficam entre 10ms e 20ms.

Na Figura 9 observa-se o fluxograma da função para detecção dos limites, onde observa-se as etapas principais. Havendo interesse nos detalhes construtivos, os mesmos podem ser verificados no ANEXO 1.

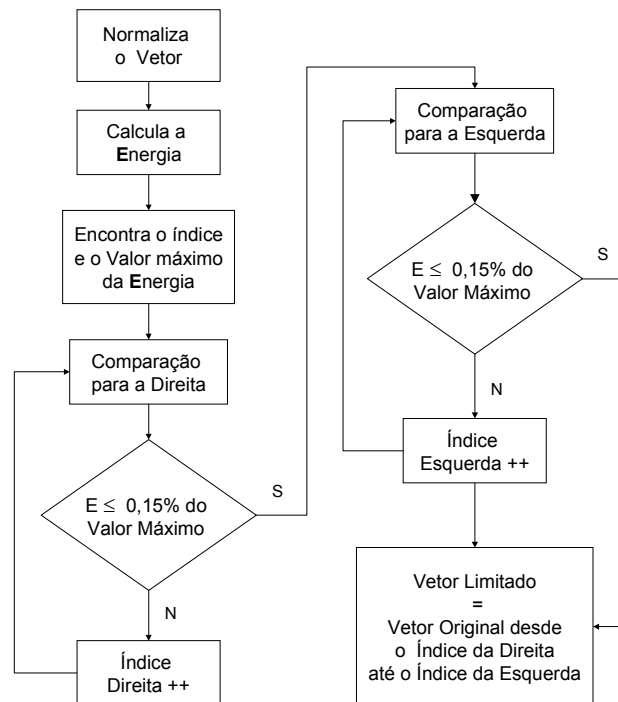


Figura 9 - Fluxograma dos passos principais da função de detecção dos limites.

No fluxograma da Figura 9, inicialmente os dados de entrada (voz) são normalizados para que se possa então calcular a energia do sinal, através da qual encontram-se os limites para a palavra, retornando assim, o vetor com a palavra limitada. O descritivo a seguir caracteriza a função e os parâmetros estabelecidos:

- Nome do arquivo: *retirazero.m*
- Nome da função: $[ysz, fs] = \text{retirazero}(y, fs)$
- Variáveis de entrada:
 - y : vetor com os dados amostrados
 - fs : frequência de amostragem
- Variáveis de saída: $[y, fs]$
 - ysz : vetor limitado
 - fs : frequência de amostragem
- Funções utilizadas:

- $[ynorm, fs] = normaliza(y, fs)$
- $[valor, indice] = max(y)$
- $sum(y)$
- Valores estabelecidos: *não requer*.

3.1.3 Identificação da Palavra

Para identificar a palavra pronunciada, foi utilizada a função denominada *números_vs2*, este nome é consequência das comparações da entrada com as palavras que servem de padrão, palavras estas que na verdade são, em sua maioria, números como já mencionado anteriormente.

O descritivo a seguir caracteriza a função e os parâmetros estabelecidos e, facilitará, posteriormente, a compreensão desta função.

- Nome do arquivo: *números_vs2.m*
- Nome da função: $[valorResultante, indice] = numeros_vs2(d2, sr)$
- Variáveis de entrada:
 - *d2*: vetor com o dado (palavra) a ser reconhecida
 - *sr*: frequência de amostragem
- Variáveis de saída: $[valorResultante, indice]$
 - *valorResultante*: vetor com os valores de todas as comparações realizadas entre as palavras padrões e a palavra a ser reconhecida.
 - *indice*: retorna o índice da palavra padrão que foi reconhecida como sendo igual a palavra de entrada.
- Funções utilizadas:
 - $B = specgram(A, NFFT, Fs, Window, Numoverlap)$

- $[y, fs] = wavread(arquivo)$
 - $M = simmx(A, B)$
 - $[p, q, C] = dp(M)$
 - $[Y, I] = min(y)$
 - $abs(x)$
- Valores estabelecidos:
 - $NFFT = 256$, que é o padrão da função.
 - $Window = \text{Hanning}$ de tamanho 256, que é o padrão da função.
 - $Numoverlap = \text{omitido}$; desta forma o Matlab usa o valor padrão que é metade do valor da janela.

A seguir serão fornecidas algumas informações sobre o funcionamento das funções *specgram*, *simmx* e *dp*.

Função SPECGRAM

A função *specgram* é proveniente do grupo de funções de processamento digital de sinais do Matlab® e retorna um vetor com a Transformada Rápida de Fourier (STFT), ou mostra esta informação na forma de um espectrograma. Maiores informações sobre o emprego desta função, podem ser encontradas na ajuda do Matlab®.

A Figura 10 demonstra, através de um exemplo, o espectrograma gerado pela função *specgram* para a palavra nove já com seus limites detectados.

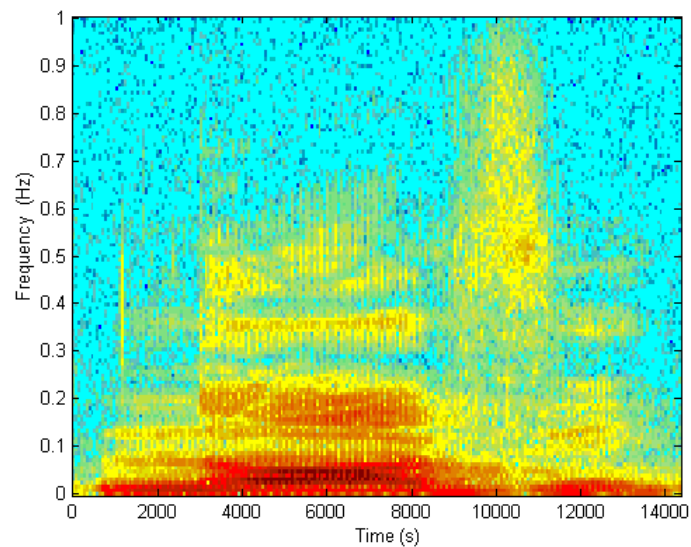


Figura 10 - Exemplo de um Espectrograma para a palavra /nove/

O programa utiliza a matriz resultante da função *specgram* para comparar os valores da palavra amostrada com os valores padrões. Isso é realizado através das funções *simmx* e *dp*, descritas a seguir.

Função SIMMX

Calcula a matriz SIM entre os espectrogramas das duas matrizes de entrada, a fim de caracterizá-las. Retorna uma matriz que será comparada na função *dp*. Nessa função, as palavras entram no formato dos valores absolutos do espectrograma. Tem-se então, o espectrograma da palavra padrão e o espectrograma da palavra de entrada que são multiplicados entre si, ao mesmo tempo em que tem seus valores normalizados.

A matriz resultante tem seu tamanho definido pelo número de colunas da matriz padrão (MP) e número de colunas da matriz que está sendo comparada (MC). Desta forma, a matriz resultante tem tamanho igual a (colunasMP x colunasMC), e é utilizada na função *dp*.

Função DP

Esta função aplica diretamente os conceitos do método *Dynamic Time Warping*, conforme a equação (13). Retorna um valor cuja grandeza servirá para identificar a palavra pronunciada. Com os dados provenientes da função *simmx*, são calculados os alinhamentos locais, cujo resultado é a mínima distância entre os valores comparados. Estes valores são somados, o valor resultante é o alinhamento global, que é tão menor quanto menor a diferença entre os dados comparados. Veja a ilustração na Figura 11.

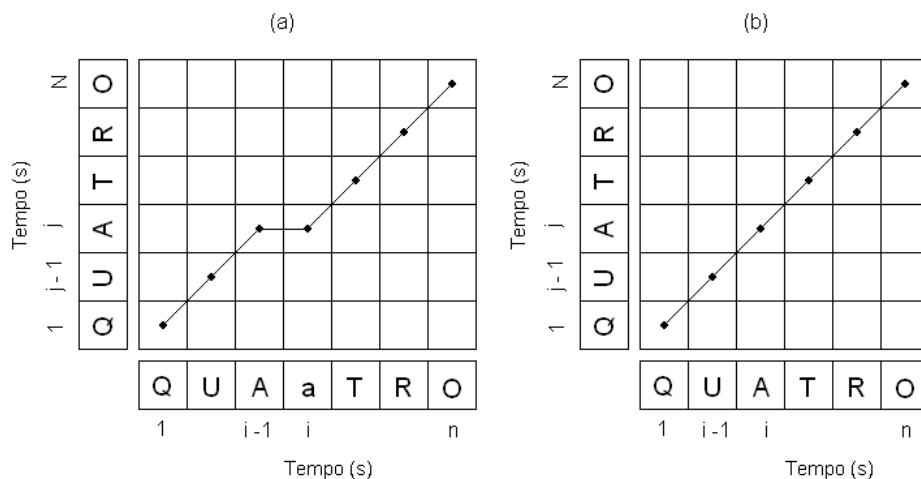


Figura 11 – Alinhamento: (a) palavras diferentes (b) palavras iguais. Fonte: Harris.

O alinhamento da Figura 11 (a) sugere que a palavra de entrada *QUAaTRO* é uma amostra com ruído o que seria notado pela inserção da letra *a*. Deve-se notar que a menor distância local entre a letra *a* e outra letra, é quando ela é comparada com a letra *A* da palavra padrão.

Já a Figura 11 (b) demonstra o alinhamento exato de duas amostras supostamente idênticas. Ao observar (a) e (b) conjuntamente, na Figura 11, constata-se que o alinhamento global resultante para a letra (b) será, logicamente, menor do que para a letra (a), visto que em (b) as palavras são iguais.

3.1.4 Interface Gráfica e Resultado

Com o objetivo de facilitar a visualização da resposta ao reconhecimento da palavra pronunciada, foram criadas as funções *avisol* e *grafico*. A Figura 12 demonstra a janela criada por esta função, as palavras pronunciadas, bem como o resultado da operação solicitada.

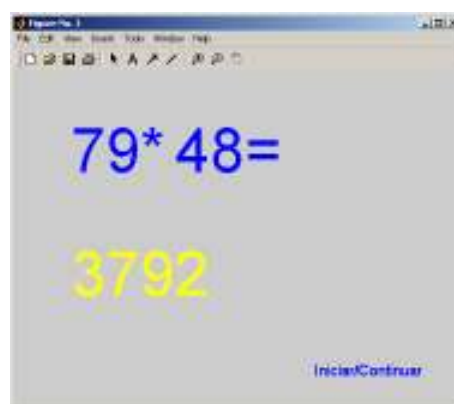


Figura 12 - Tela dos Resultados

Função AVISO1

Serve para que o usuário, ao clicar com o botão esquerdo do mouse sobre a janela que é aberta, inicie ou prossiga no processo de aquisição. Utiliza a função $[x,y,button] = ginput(...)$, que retorna as coordenadas (x,y) e qual o botão (*button*) do mouse foi pressionado.

Função GRAFICO

O propósito desta função é, através do índice que indica as palavras padrões, exibir o caractere correspondente na janela corrente e fornecer o caractere correspondente para a função que calcula o resultado da operação solicitada. Caracterização da função:

- Nome do arquivo: *grafico.m*
- Nome da função: $[variavel]=grafico(indice, a)$
- Variáveis de entrada:
 - *indice*: aponta o caractere da palavra que foi reconhecida.
 - *a*: índice que serve para posicionar os caracteres na tela.
- Variáveis de saída: $[variavel]$
 - *variavel*: retorna o caractere correspondente à palavra.
- Funções utilizadas:
 - $text(x, y, 'string', 'PropertyName', PropertyValue...)$
- Valores estabelecidos:
 - (x, y) : mudam conforme o posicionamento na tela.
 - *string*: é a variável que se deseja imprimir na tela.
 - *'PropertyName', PropertyValue*: ('FontSize', 58, 'color', 'B')
tamanho da letra igual a 58 e cor azul.

Função RESULTADO

Essa função é encarregada de calcular o valor da operação requerida pelo o usuário. Para tanto, requer que seja informado a operação selecionada (*indiceOp*), bem como os valores (*x*) com os quais se realizará esta operação. A resposta é exibida na janela, já aberta pelas etapas anteriores do programa. A seguir o descritivo da função resultado.

- Nome do arquivo: *resultado.m*
- Nome da função: *resultado(indiceOp, x)*
- Variáveis de entrada:
 - *indiceOp*: indica a operação matemática selecionada.
 - *x*: vetor com os dois valores, para a realização da operação.
- Variáveis de saída: *não possui*.
- Funções utilizadas:
 - *text(x, y, 'string', 'PropertyName', PropertyValue...)*
- Valores estabelecidos:
 - (*x, y*): *x* = 0 e *y* = 0,35, pois o posicionamento é fixo.
 - *String*: é o resultado da operação.
 - '*PropertyName*', *PropertyValue*: ('FontSize', 58, 'color', 'y')
tamanho da letra igual a 58 e cor amarela.

3.1.5 Função PRINCIPAL

A função *principal* é assim denominada, porque engloba todas as outras funções, que compõem o projeto. Segue então, o descritivo da função:

- Nome do arquivo: *principal.m*
- Nome da função: *principal*
 - Variáveis de entrada: *não requer*.

- Variáveis de saída: *não possui*.
- Funções utilizadas:
 - *aviso1*
 - $[y,fs] = \text{aquisicao_voz}$
 - $[ysz,fs] = \text{retirazero}(y,fs)$
 - $[\text{valorResultante}, \text{indice}] = \text{numeros_vs2}(ysz,fs)$
 - $\text{variavel}(a) = \text{grafico}(\text{indice},a)$
 - $\text{strcat}(s1, s2)$
 - $\text{numeric}(k)$
 - $\text{resultado}(\text{indiceOp}, x)$
- Valores estabelecidos: *não requer*.

No descritivo anterior, as funções *strcat* e *numeric*, serão brevemente explanadas a seguir. Para exibir, por exemplo, o número ‘1’ na tela, manda-se exibir o próprio caractere. Já a operação matemática não pode ser feita simplesmente utilizando-se o caractere diretamente. Então, para converter de caractere para número, usa-se a função *numeric*.

O programa realiza o reconhecimento de cada palavra individualmente, logo, o número ‘258’ é de fato a pronúncia dos números 2, 5 e 8, separadamente. Através da função *strcamp* é possível concatenar estes caracteres, transformá-los no número que representam e enfim, realizar a operação matemática desejada.

Através do fluxograma da Figura 13 é possível verificar as principais etapas da função *principal*.

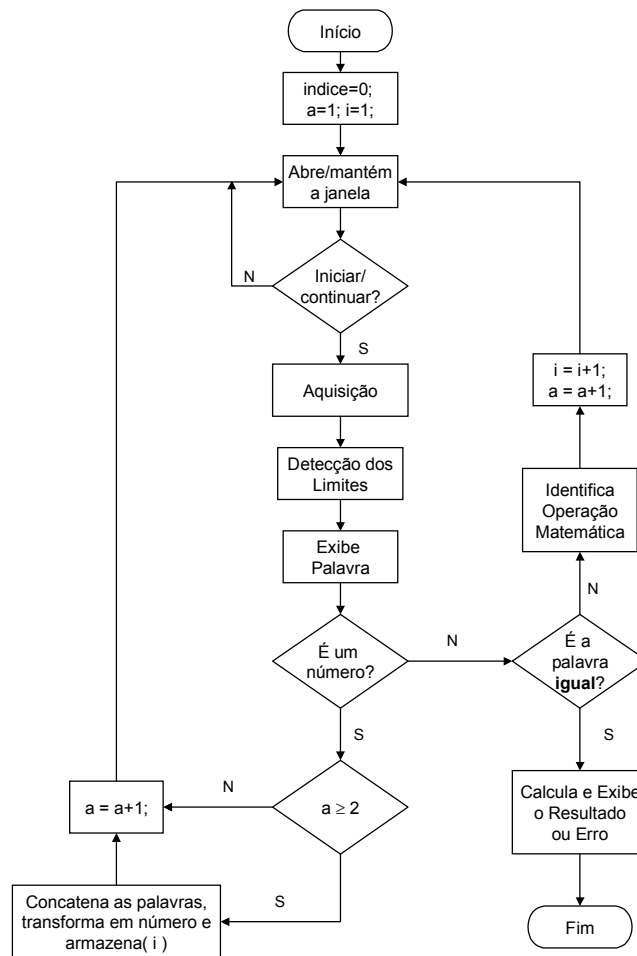


Figura 13 - Fluxograma da função *principal.m*.

A função principal (Figura 13) inicialmente abre uma janela, na qual o usuário deverá clicar com o botão esquerdo do mouse para iniciar a aquisição da primeira palavra. Após a aquisição seguem, a detecção dos limites da palavra, a realização do reconhecimento e, por fim, exibição da palavra reconhecida na tela. Em seguida, o programa verificar se a palavra é um número, se essa opção for verdadeira, é perguntado se é o primeiro número; se não for, ele concatena os números já pronunciados e segue para a próxima aquisição.

Se a palavra pronunciada não for um número, mas sim uma das operações, o programa identificará essa operação e seguirá para a próxima aquisição. Para iniciar a gravação de cada palavra pronunciada, deve-se clicar com o botão direito do mouse sobre a tela, isso serve para que o usuário tenha condições de se preparar para pronunciar a próxima palavra.

Por fim, quando a palavra ‘igual’ for identificada, o sistema calculará a resposta e exibirá, na mesma tela onde já estão as instruções (palavras) anteriormente identificadas, o resultado da operação solicitada. Se por ventura for solicitada a palavra igual em um momento impróprio, então será exibido um erro na tela e o usuário deverá fechar a janela, e chamar a função na janela de comandos do Matlab® para iniciá-la novamente.

3.1.6 Palavras Padrões

Foram gerados dezesseis padrões (Tabela 1) para as comparações, sendo dez deles os números de /zero/ até /nove/, as palavras /mais/, /menos/, /vezes/, /dividido/ e /limpa/. Utilizou-se o Matlab® para a aquisição e o programa desenvolvido para detectar os limites. Após a detecção dos limites, essas amostras foram armazenadas no diretório *work* do Matlab® e possuem o seguinte padrão para seus nomes: ‘nomedoarquivo_nsz.wav’, onde o nsz significa normalizado sem zero. A mesa de som estava regulada com um ganho (*gain*) em +4dB e o nível (*level*) na posição 2.

Palavra Pronunciada	Nome do arquivo gerado	Nome do arquivo para a Palavra Padrão
Zero	ZERO.wav	zero_nsz.wav
Um	UM.wav	um_nsz.wav
Dois	DOIS.wav	dois_nsz.wav
Três	TRES.wav	três_nsz.wav
Quatro	QUATRO.wav	quatro_nsz.wav
Cinco	CINCO.wav	cinco_nsz.wav
Seis	SEIS.wav	seis_nsz.wav
Sete	SETE.wav	sete_nsz.wav
Oito	OITO.wav	oito_nsz.wav
Nove	NOVE.wav	nove_nsz.wav
Mais	MAIS.wav	mais_nsz.wav
Menos	MENOS.wav	menos_nsz.wav
Vezes	VEZES.wav	vezes_nsz.wav
Dividido	DIVIDIDO.wav	dividido_nsz.wav
Igual	IGUAL.wav	igual_nsz.wav
Limpa	LIMPA.wav	limpa_nsz.wav

Tabela 1 - Descritivo com os nomes dos arquivos das palavras padrões.

Os seguintes equipamentos, cujas características técnicas podem ser vistas no ANEXO 3, foram utilizados para a aquisição, definição de algumas constantes no programa e para a realização dos testes:

- Computador AMD Duron 846 MHz e 480MB de RAM;
- Placa de som Audiotrack, modelo Maya1010 – High quality, 24-bit, 10in, 10out, Audio/MIDI interface with breakout box;
- Mesa de som Behringer, modelo Eurorack UB1002 - Ultra Low-noise Design, 10-input, 2 Bus mic / Line Mixer;
- Microfone Behringer, modelo Single Diaphragm Studio - Condenser Microphone B-1.

3.2 COEFICIENTES MEL CEPSTRAIS

Nesta segunda etapa do projeto são encontrados os coeficientes Mel cepstrais – MFCC, assim como o delta cepstrum. A extração destes parâmetros, cujo resultado encontram-se em forma de vetores, pode ser empregado em métodos de reconhecimento tais como *Hidden Markov Models* (HMMs), *Gaussian Mixture Models* (GMM) ou quantização vetorial.

As funções para aquisição de dados, bem como a de detecção dos limites, são as mesmas já apresentadas neste trabalho. Coube, nesta fase do trabalho, o desenvolvimento das etapas de pré-ênfase, janelamento, modelagem dos bancos de filtros, extração dos MFCC e dos delta cepstrum.

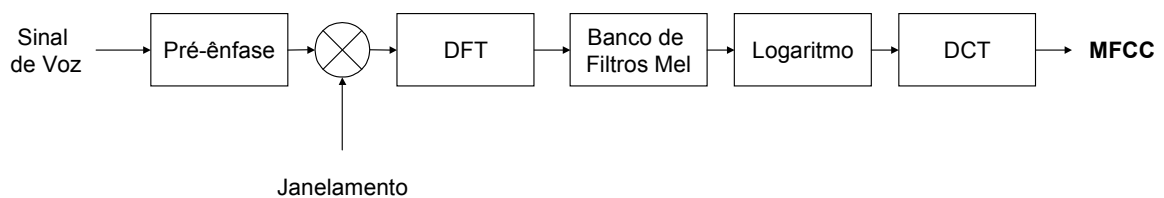


Figura 14 - Diagrama de blocos das etapas para obtenção dos Coeficientes MFCC

A Figura 14 demonstra as etapas necessárias à obtenção dos coeficientes Mel cepstrais. Os tópicos a seguir apresentam cada bloco separadamente.

3.2.1 Pré-ênfase

Como descrito em 2.1, esse bloco serve apenas para tornar as amplitudes mais homogêneas. A função criada denomina-se *preEnfase.m* e seu descritivo pode ser visto a seguir:

- Nome do arquivo: *preEnfase.m*
- Nome da função: *[yPre]=preEnfase(sinal)*
- Variáveis de entrada:
 - *sinal*: sinal de voz amostrado
- Variáveis de saída: *[yPre]*
 - *yPre*: vetor com o sinal de entrada depois do filtro
- Funções utilizadas:
 - $y = \text{filter}(b,a,X)$
 - b = numerador;
 - a = denominador;
 - X = sinal de entrada;
- Valores estabelecidos:
 - $b = [1 - 0.95]$; (0,95 coeficiente sugerido em 2.16)
 - $a = [1]$;

A função *filter* implementa um filtro, conforme os coeficientes estabelecidos nos vetores a e b , e retorna em y o resultado do sinal de entrada X já filtrado.

A Figura 15 demonstra o gráfico para a palavra /zero/ e o gráfico da mesma palavra após a pré-ênfase, onde é possível notar a homogeneização das amplitudes.

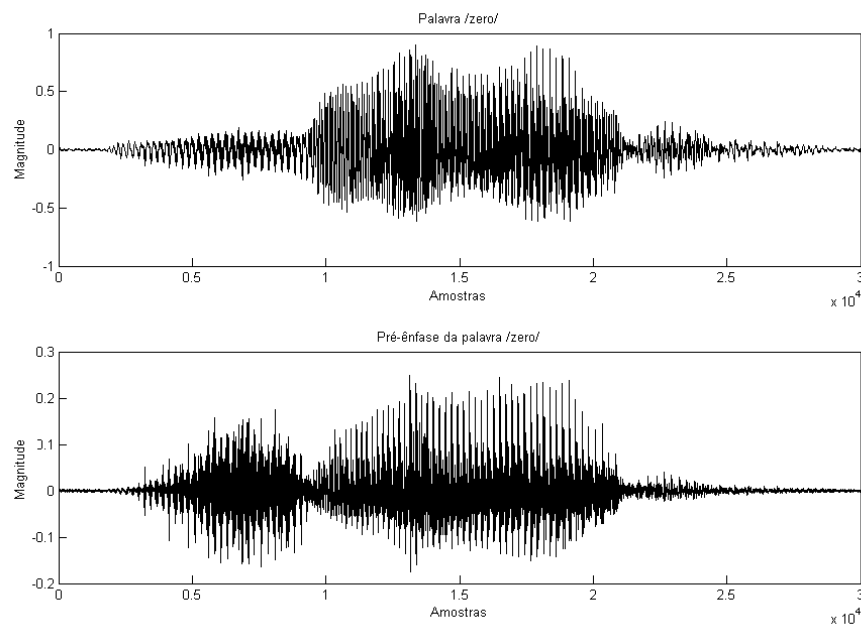


Figura 15 - Palavra /zero/ e sua versão após a pré-ênfase

3.2.2 Janelamento e DFT

O janelamento e o bloco que calcula a DFT são implementados conjuntamente e são obtidos através da função *specgram* do Matlab, função já empregada na parte 1 deste trabalho. O gráfico gerado por esta função pode ser visto, a seguir, na Figura 16.

Os dados que geram o espectrograma demonstrado na Figura 16, são resultado dos seguintes parâmetros:

- Função utilizada e os valores estabelecidos:
 - $B = \text{specgram}(A, NFFT, Fs, Window, Numoverlap)$
 - A = sinal de voz após pré-ênfase;
 - $NFFT = 2048$;
 - Fs = frequência de amostragem (44100Hz);

- $Window = \text{hamming}(1024);$
- $Numoverlap = 256$ (25% de 1024).

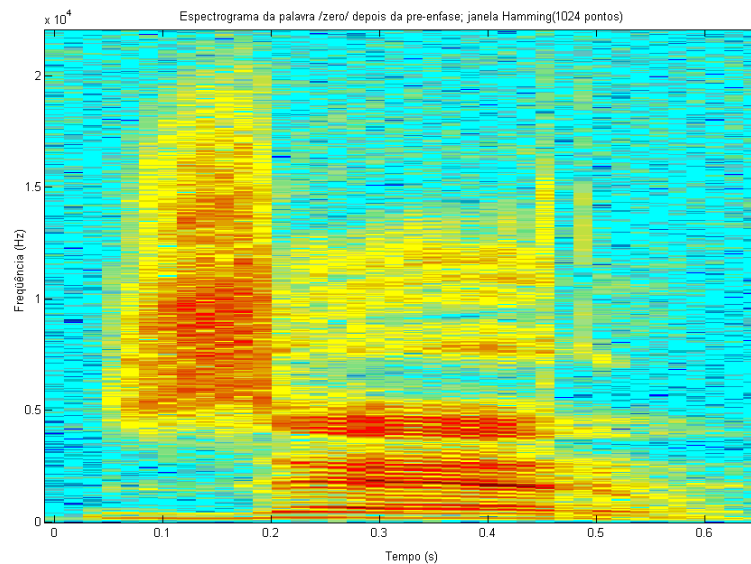


Figura 16 - Espectrograma da palavra /zero/ após pré-ênfase

Note que no descritivo anterior o tamanho $NFFT$ é diferente do tamanho da janela. Isso deve-se ao fato da função *specgram* retornar a matriz B com o número de colunas igual a $NFFT/2$. Como deseja-se uma matriz resultante com 1024 colunas, estabelece-se $NFFT=2048$.

As janelas Hamming ou Hanning são as comumente utilizadas em aplicações para o reconhecimento automático de voz. O tamanho normalmente empregadas nas janelas fica entre 10ms e 30ms (BECCHETTI). A janela aplicada é igual a 23,22ms, e é resultante da Equação 2: $t = n \cdot Ts$, onde $n=1024$ e $Ts = 1/44100Hz$. O valor 1024 pode ser identificado no parâmetro *Window*, estabelecido anteriormente, onde define-se uma janela Hamming de tamanho 1024.

3.2.3 Banco de Filtros Mel

A elaboração do banco de filtros envolve duas etapas, a primeira consiste em determinar os valores das frequências centrais, assim como as frequências de corte inferior e superior, já que se trata de filtros passa-faixa. A segunda etapa é a implementação do filtro em si e utiliza os valores obtidos na primeira etapa.

Nesta primeira etapa utiliza-se os conceitos abordados em 2.2.4, desta forma, tem-se 10 filtros distribuídos igualmente, com as frequências centrais variando de 100 em 100*mel* e os 14 filtros restantes seguindo a aproximação dada pela Equação 16. Os valores desta distribuição podem ser observados na Tabela 2:

	delta	Frequência Central (Hz)	Limite inferior (Hz)	Limite superior (Hz)
1	65	65	0	130
2	65	130	65	195
3	65	195	130	260
4	65	260	195	325
5	65	325	260	390
6	65	390	325	455
7	65	455	390	520
8	65	520	455	585
9	65	585	520	650
10	65	650	585	715
11	78	728	650	806
12	93,6	821,6	728	915,2
13	112,32	933,92	821,60	1046,24
14	134,78	1068,70	933,92	1203,49
15	161,74	1230,44	1068,70	1392,19
16	194,09	1424,53	1230,44	1618,62
17	232,91	1657,44	1424,53	1890,35
18	279,49	1936,93	1657,44	2216,42
19	335,39	2272,31	1936,93	2607,70
20	402,46	2674,78	2272,31	3077,24
21	482,96	3157,73	2674,78	3640,69
22	579,55	3737,28	3157,73	4316,83
23	695,46	4432,74	3737,28	5128,19
24	834,55	5267,28	4432,74	6101,83

Tabela 2 - Distribuição das frequências

Nota-se na Tabela 2 a distribuição linear até a linha (filtro) 10, onde a base do triângulo, que é a faixa de frequências, está denominada como delta e vale 65Hz.

A Figura 17 exibe o gráfico da distribuição das frequências, conforme a Tabela 2. Embora apresente as frequências até o filtro 16, por uma questão de legibilidade do gráfico, é possível verificar a linearidade da distribuição até a faixa referente ao filtro 10, bem como a distribuição não linear dos gráficos que seguem. Isto pode ser identificado pela largura diferenciada da base dos triângulos formados por cada filtro.

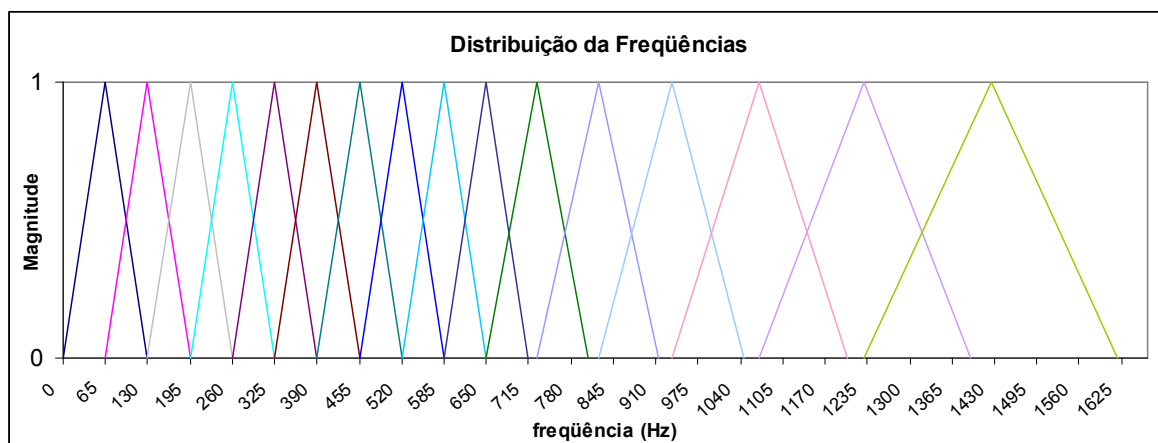


Figura 17 - Gráfico da Distribuição das Frequências

A segunda etapa é a aplicação destes valores na implementação dos filtros. No Matlab® foram criadas duas funções, uma primeira, denominada *frequencias.m* que calcula os valores dados na Tabela 2 e uma segunda, que implementa o filtro triangular que é denominada *filtro_triangular.m*, cujo fluxograma pode ser visto a seguir.

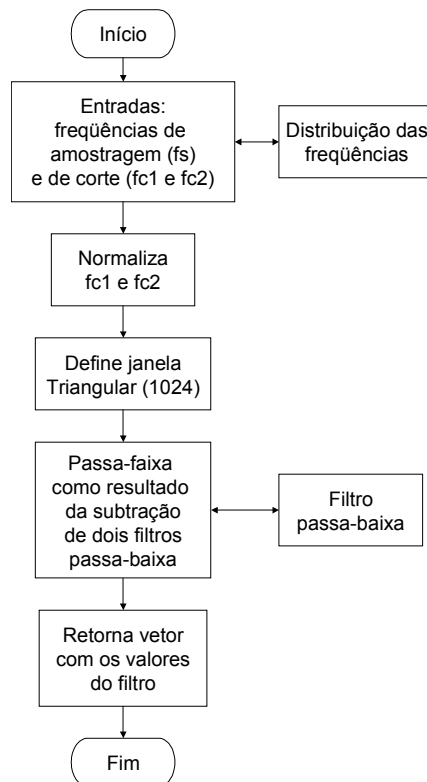


Figura 18 - Fluxograma: etapas principais filtro triangular.

Na Figura 18 é possível observar o fluxograma das etapas principais decorrentes da implementação do banco de filtros. A função base é denominada *filtro_triangular.m* e emprega as funções *frequencia.m* e *passa_baixa.m*. A seguir é possível ver um descritivo com as características da função:

- Nome do arquivo: *filtro_triangular.m*
- Nome da função: $[h] = \text{filtro_triangular}(fs, fc1, fc2)$
- Variáveis de entrada:
 - fs : frequência de amostragem
 - $fc1$: frequência de corte inferior;
 - $fc2$ = frequência de corte superior.
- Variáveis de saída: $[h]$
 - h : vetor com o filtro.

- Funções utilizadas:
 - $[freqCentral, limitesInferior, limitesSuperior] = frequencias$
 - $freqCentral$ = frequência de central;
 - $limitesInferior$ = frequência de corte - limite inferior;
 - $limitesSuperior$ = frequência de corte - limite superior
 - $hd=passa_baixa(wc,M)$
 - wc = frequência de corte normalizada;
 - M = tamanho da janela;
- Valores estabelecidos:
 - $M = 1024$;

Um exemplo da resposta do filtro implementado pode ser visto na Figura 19.

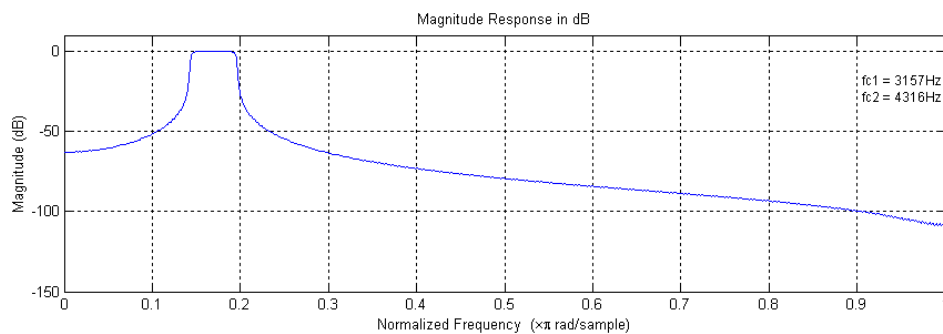


Figura 19 - Exemplo da resposta para o filtro 22.

Foram implementados 24 filtros, cujos valores das frequências são provenientes da função *frequencia.m*, que retorna um vetor com as frequências centrais e de corte. A Figura 19 demonstra a resposta do filtro número 22, cuja frequência central é 3737Hz, e as frequências de corte inferior e superior são, respectivamente, 3157Hz e 4316Hz. Estando o eixo das

freqüências normalizado em relação a π e estando a unidade em radianos por amostra, pode-se confirmar visualmente que, para freqüência de corte inferior $fc1 = 3157\text{Hz}$, a equivalente em radianos é aproximadamente igual a $0.14 \cdot \pi \text{ (rad / amostra)}$ e pode ser visualmente identificada no gráfico da Figura 19. A Equação 20 define a freqüência normalizada:

$$w_{\text{normalizado}} = \frac{2 \cdot f}{fs} \cdot \pi \text{ [rad / amostra]} \quad (20)$$

onde f é a freqüência que deseja-se encontrar e fs a freqüência de amostragem do sinal, ambas em Hertz.

Seguem os cálculos para os valores das freqüências normalizadas anteriormente, e que podem ser vistas na Figura 19:

$$w_{c1} = \frac{2 \cdot fc1}{44100} \cdot \pi \cong 0,14 \cdot \pi \text{ (rad / amostra)}, \quad w_{c2} = \frac{2 \cdot fc2}{44100} \cdot \pi \cong 0,20 \cdot \pi \text{ (rad / amostra)} \quad \text{e}$$

$$w_c = \frac{2 \cdot fc}{44100} \cdot \pi \cong 0,17 \cdot \pi \text{ (rad / amostra)}.$$

3.2.4 Coeficientes Mel Cepstrais

Este tópico apresenta, por fim, a obtenção dos coeficientes MFCC – *Mel Frequency Cepstrum Computation*. Na Figura 14, podem ser vistos os blocos denominados Logaritmo e DCT, ambos são obtidos, respectivamente, pelas funções $\log_{10}(a)$ e $dct(b)$ disponíveis no Matlab. O descritivo a seguir mostra a função *MelCoeficientes.m* criada para gerar os MFCC

coeficientes, e que faz uso de todas as funções demonstradas nos tópicos anteriores, bem como as funções *dct* e *log10*:

- Nome do arquivo: *MelCoeficientes.m*
- Nome da função: *[MFCC, delta_C]=MelCoeficientes(sinal, fs)*
- Variáveis de entrada:
 - *Sinal* = sinal amostrado e limitado
 - *fs*: frequência de amostragem
- Variáveis de saída: *[MFCC, delta_C]*
 - *MFCC*: matriz com os coeficientes Mel cepstrais
 - *delta_C* = matriz com os delta de 1ª ordem dos MFCC
- Funções utilizadas:
 - *B = specgram(a,nfft,fs>window,numoverlap);*
 - *[freqCentral, limitesInferior, limitesSuperior]=frequencias;*
 - *[h]=filtro_triangular(fs,fc1,fc2);*
 - *a = log10(b);*
 - *y = dct(x);*

A função *log10* é responsável apenas por calcular o logaritmo dos valores resultantes dos blocos anteriores, como pode-se ver na Figura 14. A função *dct* calcula a *Discrete Cosine Transform*, as duas funções conjuntamente oferecem o resultado esperado na Equação 14.

Conforme BECCHETTI o coeficiente de ordem zero, ou seja, o primeiro coeficiente, é equivalente ao *log* do *frame*, por isso é normalmente descartado. Desta forma, adotou-se este procedimento na obtenção dos coeficientes, como segue.

O programa desenvolvido gera uma matriz (Equação 21(a)) de 24 linhas pelo o número de *frames*. As linhas são o número de filtros, e os *frames* variam conforme o tamanho (tempo de

duração) da palavra que está sendo avaliada. Desta forma, para ignorar o(s) primeiro(s) coeficientes, basta ignorar a primeira linha desta matriz.

A respeito do número de coeficientes a serem adotados, MAFRA sugere que se adote entre 8 e 16 coeficientes. Partindo desse pressuposto, foram adotados 16 coeficientes, o que resulta numa matriz de 16 linhas pelo número de *frames*, mostrada Equação 21(b).

$$\begin{aligned}
 \text{Matriz} &= \underbrace{\begin{bmatrix} C_{1,1} & \cdots & C_{1,n} \\ C_{2,1} & & C_{2,n} \\ \vdots & & \vdots \\ C_{24,1} & & C_{24,n} \end{bmatrix}}_{\text{FRAMES}} \begin{bmatrix} 1 \\ 2 \\ \vdots \\ 24 \end{bmatrix} \left. \vphantom{\begin{bmatrix} C_{1,1} \\ C_{2,1} \\ \vdots \\ C_{24,1} \end{bmatrix}} \right\} \text{filtros} & \quad \text{CoeficientesMFCC} = \begin{bmatrix} C_{2,1} & \cdots & C_{2,n} \\ \vdots & & \vdots \\ C_{17,1} & & C_{17,n} \end{bmatrix} \\
 (a) & \quad (b)
 \end{aligned}$$

Equação 21 – (a) Matriz resultante (b) Coeficientes MFCC

A Equação 21(b) mostra a matriz resultante para os coeficientes Mel, cujo gráfico do primeiro *frame*, da palavra /zero/, pode ser visto na Figura 20(d), assim como o gráfico da saída do banco de filtros (a), depois do logaritmo (b) e após a *Discrete Cosine Transform* (c).

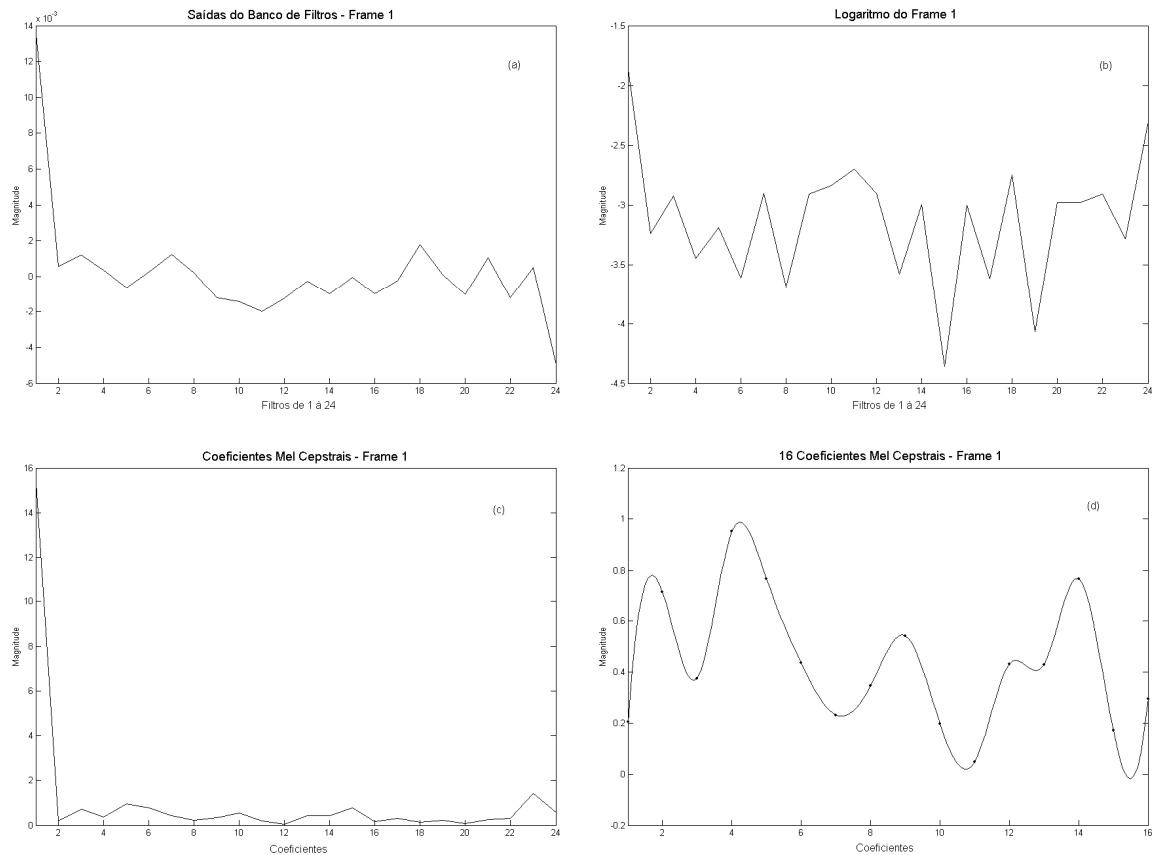


Figura 20 - Gráficos das saídas das etapas implementadas.

O gráfico da Figura 20 (a) representa a saída dos 24 filtros para o *frame* 1 da palavra /zero/. A Figura 20 (b) traz esses mesmos valores após aplicar-se o logaritmo. Em (c) são representados os valores resultantes da DCT. Por fim, em (d) tem-se os 16 coeficientes MFCC.

Como abordado 2.2.5, os deltas coeficientes MFCCs disponibilizam informação temporal sobre a variação dos MFCC coeficientes. Esse cálculo faz parte da função *MelCoeficientes.m* e detalhes sobre a sua implementação, pode ser vistas no ANEXO 1. A Figura 21 demonstra um exemplo para os Delta Coeficientes Cepstrais.

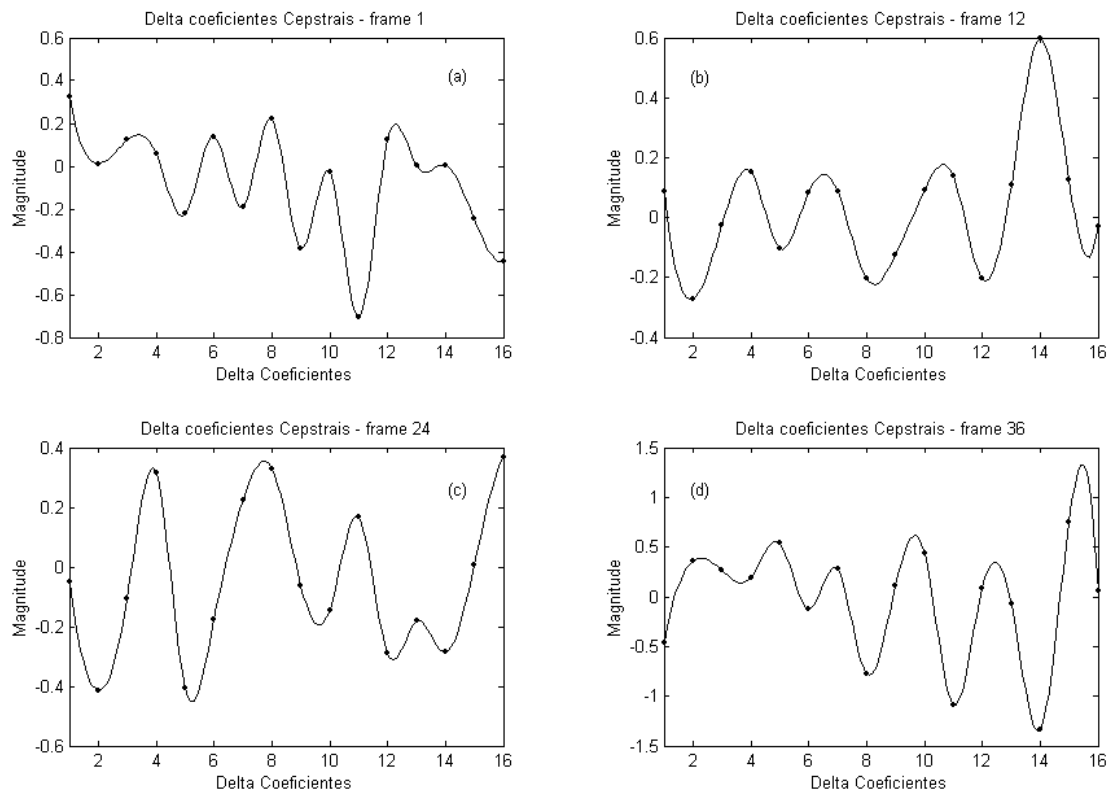


Figura 21 - Delta Coeficientes Cepstrais

Na Figura 21 tem-se os delta coeficientes para a palavra /dividido/ nos *frames* 1, 12, 24 e 36, representados pelas letras (a), (b), (c) e (d) respectivamente.

4 DISCUSSÃO DOS RESULTADOS

4.1 DYNAMIC TIME WARPING

Esta parte refere-se as etapas desenvolvidas e que encontram-se sob o mesmo título em 3.1, seguem então as constatações a respeito destes tópicos.

A função para detecção dos limites foi a primeira a ser desenvolvida e mostrou-se eficaz em seus resultados. Foram realizados alguns testes e ocorreram erros de detecção de limites apenas para as palavras /oito/ e /quatro/ como é possível constatar nos gráficos do ANEXO 4.

Como o objetivo é reconhecer apenas um conjunto de 16 palavras, que não possuem grande semelhança entre si, o sistema de reconhecimento não é afetado se, por exemplo, a palavra /oito/ tiver como resultado da detecção dos limites somente a sílaba /oi/. Por outro lado, se a palavra /oi/ fizesse parte das palavras padrões, então, sem dúvida, haveria um reconhecimento falho para a palavra /oito/.

O emprego do DTW no reconhecimento apresentou bons resultados, mesmo que inicialmente o tempo total da comparação e reconhecimento de uma palavra tenha ficado em aproximadamente 30 segundos, não aconteceram reconhecimentos errados. Para sanar a questão do tempo, foi necessário reduzir a amostragem dos sinais. Essa redução não ocasionou problema algum para o reconhecimento e melhorou o tempo significativamente, ficando em torno de 3 segundos.

Ao realizar o teste (Figura 22), desta vez com um microfone comum e com a placa som *onboard*, verificou-se no sinal amostrado além de ruído, nível DC. Isso afetou drasticamente o programa, visto que o mesmo estava com os valores adequados ao conjunto de equipamentos citados anteriormente. Para implementar o sistema usando a placa *onboard* e o microfone comum, seria necessário tratar o sinal amostrado retirando tanto o nível DC e o ruído, para que então fosse possível empregar o mesmo programa.

Na Figura 22 (a) observam-se gráficos comparativos entre a palavra padrão /cinco/, cuja aquisição foi realizada com os equipamentos citados no item 3.1.6, e a palavra /cinco/, resultado da aquisição com a placa *onboard* e microfone comum. Notam-se claramente as diferenças, tanto na amplitude como no nível DC. Já a Figura 22 (b) demonstra somente a palavra /cinco/ resultante da aquisição com a placa *onboard*, neste gráfico é possível identificar mais claramente os ruídos que ocorrem durante os silêncios.

Embora tenha sido identificados a ocorrência de nível DC e ruído na aquisição com a placa *onboard* e microfone comum, não foram implementadas soluções para tais problemas. Seria apropriado a realização de mais testes com a placa *onboard*, para que desta forma os

dados pudessem ser melhor analisados. Partindo-se desta análise seria possível implementar as etapas de tratamento deste sinal, para então empregá-lo no sistema de reconhecimento desenvolvido.

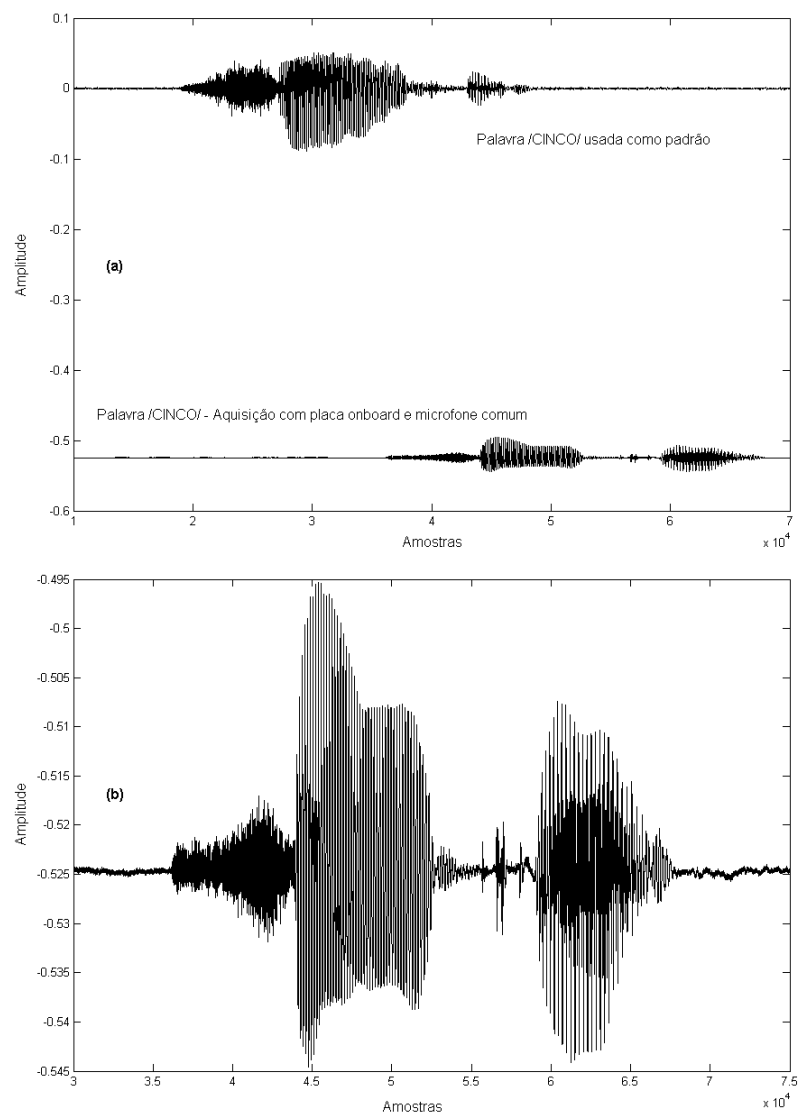


Figura 22 - (a) Comparativo entre duas palavras /cinco/ (b) Detalhe da palavra /cinco/ aquisição com placa *onboard*.

4.1.1 Análise Estatística

Em se tratando de um sistema de reconhecimento de palavras, faz-se importante verificar o percentual de acertos, ou seja, a capacidade do sistema em cumprir a sua função. Para tanto, foram realizados dois testes, ambos empregando o programa desenvolvido em 3.1 deste trabalho. No primeiro teste colheu-se 25 amostras para cada palavra, sendo que as palavras foram pronunciada aleatoriamente. Ainda, cabe informar que incluem-se nesta gama de amostras, algumas com ruído de fundo, diferente posicionamento em relação ao microfone, velocidade e altura (nível do som) diferenciadas.

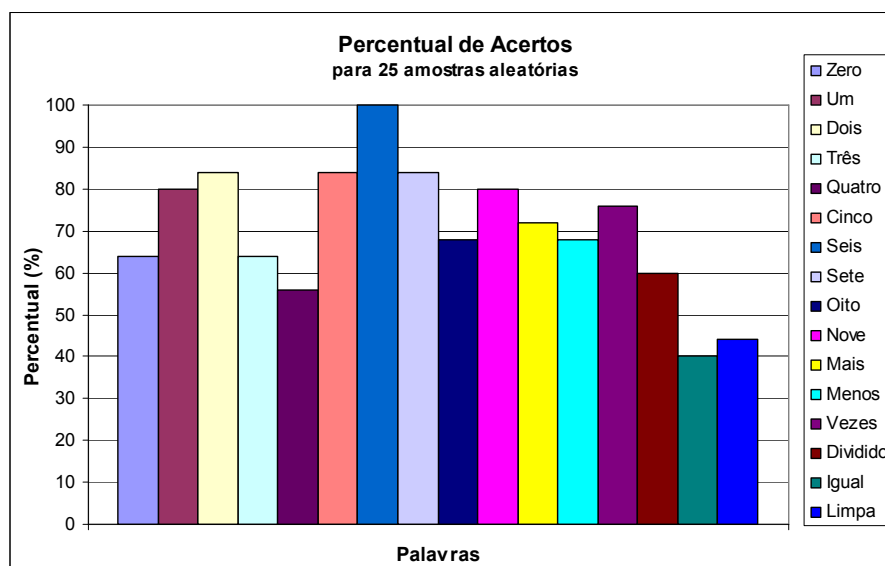


Figura 23 - Percentual de acertos para amostras aleatórias.

A Figura 23 revela as repostas percentuais obtidas para 25 amostras colhidas no mesmo dia. É possível identificar que para a palavra /seis/ houve 100% de acerto no reconhecimento, por outro lado, a palavra /igual/ teve um percentual de acertos próximo a 40%.

Ainda, foi realizado mais um levantamento de 25 amostras para cada palavra, no qual pronunciou-se seqüencialmente, de cinco em cinco, cada palavra. O resultado pode ser visto na Figura 24.

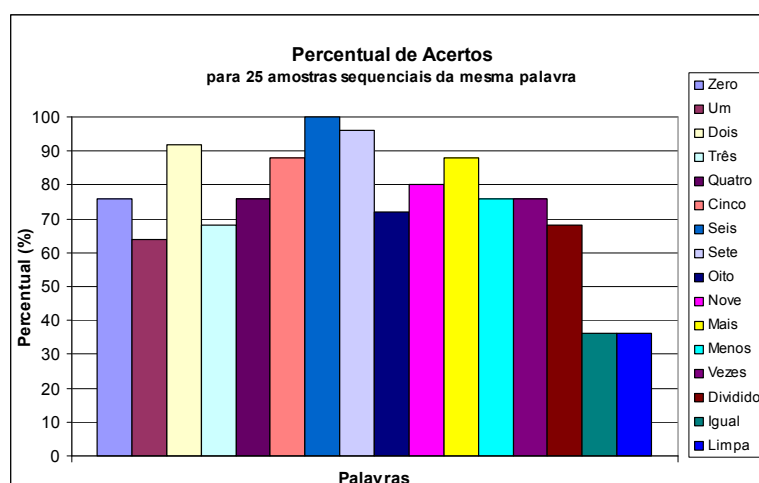


Figura 24 - Percentual de acertos para amostras seqüenciais.

Exceto pelas palavras /igual/ e /limpa/, o percentual de acertos mostrou-se maior em relação aos valores da Figura 23. Isso ocorre, pois quando pronuncia-se qualquer palavra que seja, repetidamente, a tendência é que características como a altura do som, entonação e velocidade mantenha-se constante.

Em relação aos baixos percentuais de acertos, notou-se que singularidades nas palavras usadas como padrão, provocaram tais erros. Pode-se notar que a palavra /seis/, que teve 100% de acerto nas duas amostragens, não possui muitas formas de ser pronunciada. Já as palavras /sete/ e /cinco/, entre outras, podem ter sua pronuncia alterada para /seti/ e /cincu/, o que ocasionaria dificuldades para reconhecimento em relação ao padrão adotado. Desta forma

constatou-se que seria mais apropriado aumentar-se o número de amostras padrão, ao invés de adotar apenas uma única palavra.

4.2 COEFICIENTES MEL CEPSTRAIS

Os bancos de filtros *Mel Frequency* englobam dois conceitos em sua implementação, o primeiro é a distribuição das frequências na escala Mel e o segundo diz respeito à implementação do filtro propriamente dito. Após a primeira implementação do banco de filtros, os resultados mostraram-se bastante diferentes das respostas encontradas por MAFRA. Desta forma foi necessário, gerar uma nova lógica para a distribuição das frequências, além de revisar e identificar os erros na lógica implementada para os filtros.

Ao confrontar novamente os novos coeficientes obtidos com o programa reestruturado, verificou-se certa semelhança entre os resultados, se comparados aos resultados provenientes de programas de outros autores. Diz-se certa semelhança, pois cada autor adota parâmetros diferenciados, seja na distribuição das frequências dos filtros, seja no tamanho da janela utilizada. Na busca de validação destes coeficientes, um comparativo entre 16 coeficientes cepstrais para diferentes *frames* de uma mesma cossenóide de 500Hz, podem ser observados na Figura 25, a seguir.

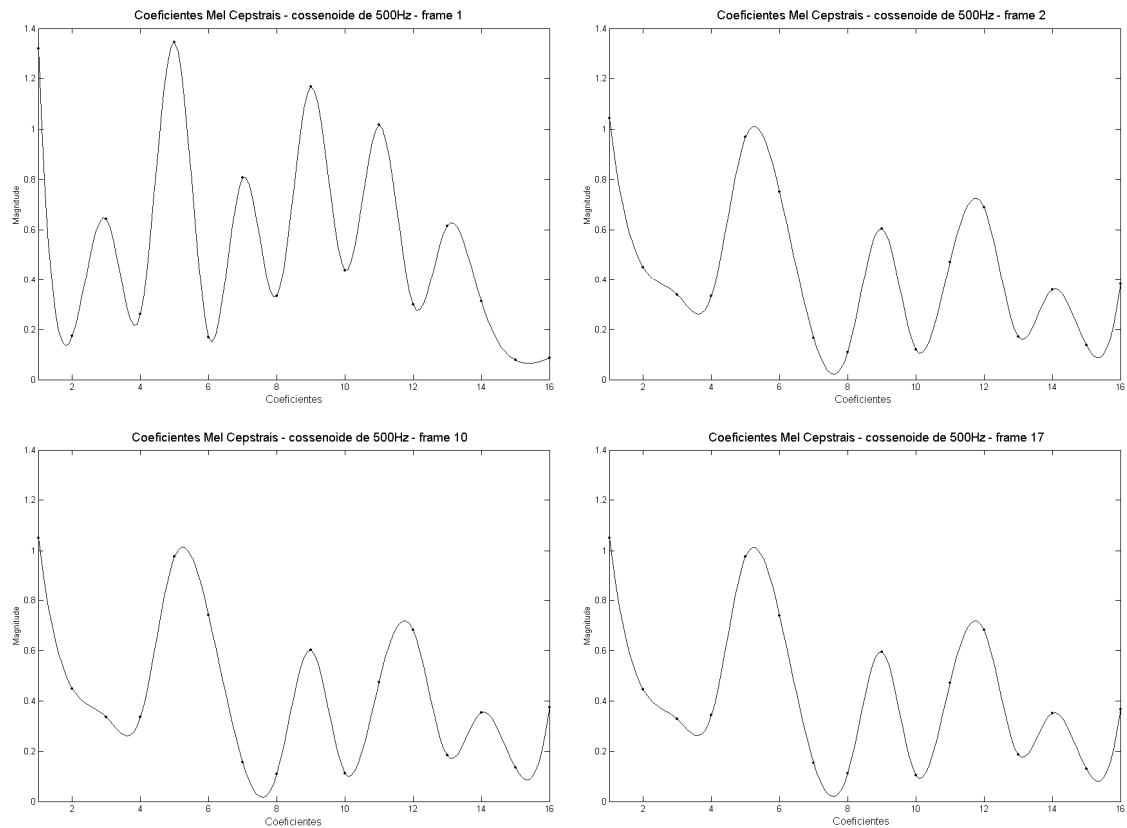


Figura 25 - Comparativo entre os coeficientes de uma cossenóide de 500Hz.

Nota-se, conforme ilustrado na Figura 25, que o primeiro *frame* possui coeficientes diferentes dos *frames* 2, 10 e 17, isso é decorrência da pré-ênfase, pois ao utilizar o sinal sem esta etapa, todos os *frames* tiveram os mesmos coeficientes como pode ser visto na Figura 26. Desta forma, a identificação desta cossenóide seria clara e corresponderia ao objetivo dos coeficientes cepstrais que é representar um sinal qualquer.

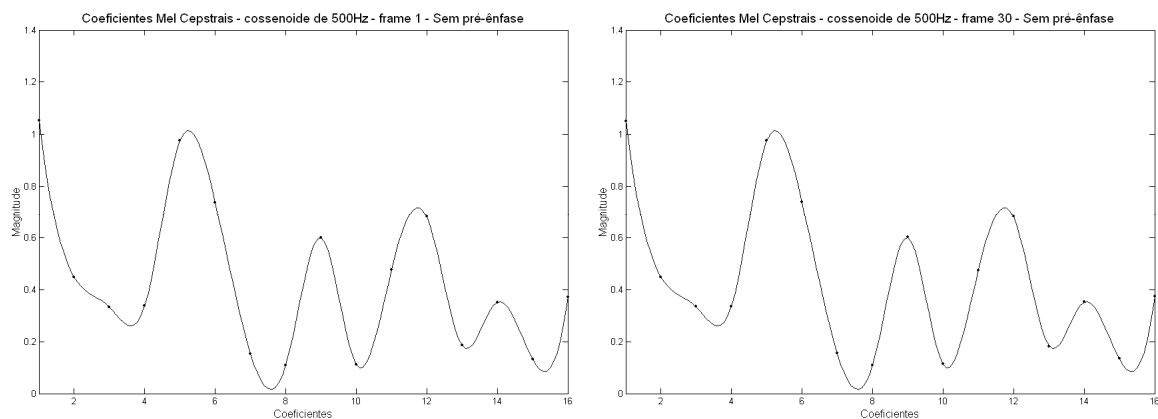


Figura 26 - Coeficientes da cossenóide de 500Hz sem a pré-ênfase.

A Figura 26 demonstra os coeficientes cepstrais da mesma cossenóide utilizada para os coeficientes da Figura 25. A diferença está na não utilização da etapa de pré-ênfase, ou seja, os coeficientes são resultados do sinal original, a cossenóide de 500Hz. Nota-se que os coeficientes para o *frame* 1, assim como os do *frame* 30 são, como já comentado anteriormente, iguais. Isso significa que independe do *frame*, os coeficientes são os mesmos, já que resultam da mesma forma de onda.

4.3 CONSTATAÇÕES E ALTERAÇÕES

A medida em que o trabalho se desenvolveu observou-se que a facilidade de acesso do usuário não estava sendo cumprida, já que era necessário utilizar o mouse para dar continuidade à aquisição da voz. Tal necessidade é indicada pela decisão “Iniciar/continuar?” no fluxograma da Figura 13. Em relação a esta situação é implementada uma nova função, conforme demonstra o novo fluxograma, na Figura 27.

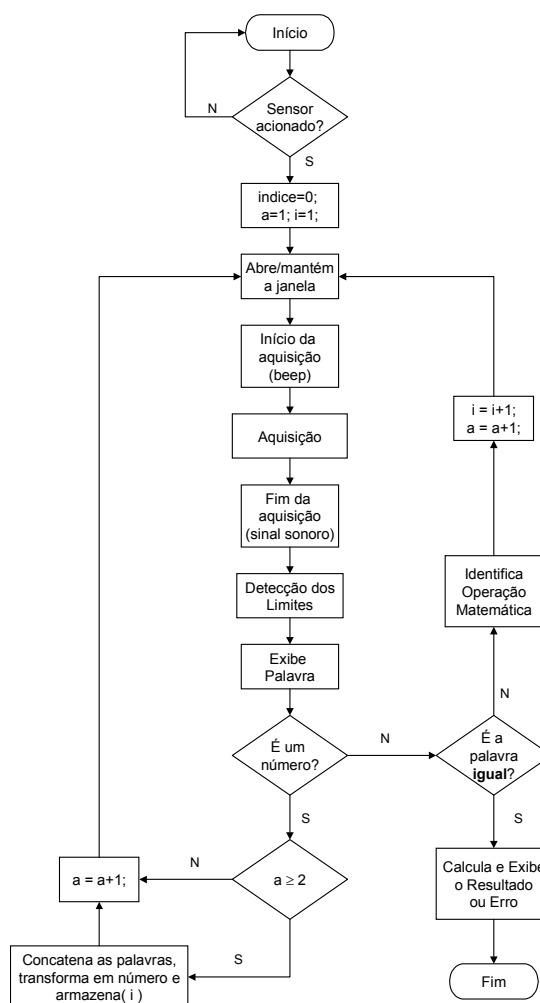


Figura 27 – Nova função principal.

O fluxograma da Figura 27 traz duas alterações, uma diz respeito à necessidade do usuário em clicar com o mouse, esta foi sanada retirando-se a solicitação de clique e inserindo-se dois avisos sonoros que indicam o início e o fim da aquisição. Ainda, a decisão denominada “Sensor acionado?” foi adicionada, e serve para iniciar o programa frente a detecção de alguma presença. A leitura deste suposto sensor é feita através do bit 0 da porta paralela e é realizada pela função denominada *sensor.m*, que pode ser vista no ANEXO 1.

5 CONCLUSÕES

Em LUFT observa-se o resultado do teste realizado para o programa implementado utilizando DTW. O percentual de acertos encontrado por LUFT, foi de 96,3% em um vocabulário de 47 palavras, para 188 amostras. Porém não são informadas as condições da aquisição das amostras, bem como a seqüencialidade das amostras. Ainda, a capacidade de reconhecimentos observada em outras publicações tem percentual de acertos em torno dos 90%.

Desta forma, ponderando sobre a taxa de acertos do programa desenvolvido neste trabalho, verifica-se que a média de acertos é de 70,2% para 400 amostras aleatórias e de 74,5% para as amostras seqüenciais. Em ambos os casos foram colhidas 25 amostras para cada uma das 16 palavras. Embora não seja um percentual de acertos tão baixo, é inferior aos percentuais citados por outros autores. Ainda, analisando-se o reconhecimento de algumas palavras em particular, principalmente às palavras /igual/ e /limpa/, conclui-se que o sistema ainda não é completamente apropriado, necessitando algumas adequações para ser completamente confiável.

Uma alternativa para melhorar o percentual de acertos seria aumentar o banco de dados, adicionado mais amostras para uma mesma palavra. Isso pode ser feito, armazenado-se as palavras na forma de dados em arquivos de texto, ao invés de arquivos de som, como é feito atualmente. Pensando-se em facilitar o entendimento geral, inicialmente optou-se por armazenar os dados como arquivos de som, pois possibilitariam de forma direta eventuais alterações nestes padrões. Tal ocorrência seria menos visível se fosse necessário substituir os padrões em forma de dados armazenados em arquivos de texto.

Ainda, a fim de otimizar o programa, sugere-se a aquisição contínua do conjunto de comandos pronunciados, para então isolar as palavras e fazer o reconhecimento. Com isso a aquisição ficaria mais agradável e rápida ao usuário, que não precisaria aguardar o tempo da aquisição de cada uma das palavras separadamente. Para tal implementação seria necessário desenvolver um novo programa de detecção dos limites da palavra.

Sobre os coeficientes MFCC observou-se que ao aplicar no banco de filtros desenvolvido uma cossenóide de frequência conhecida, obteve-se para todos os *frames* deste sinal, os mesmos coeficientes Mel cepstrais. Este resultado indica que o banco de filtros implementado oferece respostas iguais para todos os *frames*, comprovando assim a sua funcionalidade e validando os coeficientes obtidos.

Finalizando, o trabalho apresentou os resultados esperados, já que os objetivos propostos inicialmente foram todos atingidos.

6 REFERÊNCIAS BIBLIOGRÁFICAS

1. AUDIOTRACK *Digital Audio Interfaces*. Disponível em: <http://www.audiotrak.de/eng/manual.html>. Acessada em outubro de 2005.
2. BECCHETTI, Cláudio e RICOTTI, Lucio Prina. **Speech Recognition - Theory in C++ Implementation**. England: Wiley, 1999.
3. BEHRINGER Audio Products. Disponível em: <http://www.behringer.com/>. Acessada em outubro de 2005.
4. CHILDERS, Donald G. **Speech Processing and Synthesis Toolboxes**. USA: Wiley, 2000.
5. CICLOTRON Indústria de Eletrônica LTDA - Equipamentos para sonorização profissional. Disponível em: www.ciclotron.com.br/manuais/techvox/tge2312s.pdf. Acessada em outubro de 2005.
6. COSTA, Sara e COSTA, Teresa. **Fonologia e Fonética**. Disponível em: <http://criarmundos.do.sapo.pt/Linguistica/pesquisalinguistica02.html#fonetica1>. Acessada em agosto de 2005.
7. HANSELMAN, Duane e LITTLEFIELD, Brude. **Matlab - Guia do Usuário**. São Paulo: Makron Books, 1997.
8. HARRIS, John G. **Isolated word, Speech recognition using Dynamic Time Warping towards smart appliances**. Disponível em: <http://www.cnel.ufl.edu/~kkale/6825Project.html>. Acessada em agosto de 2005.

9. IAZZETTA, Fernando. Tutoriais de Áudio e Acústica. Departamento de Música da ECA-USP. Disponível em: <http://www.eca.usp.br/prof/iazzetta/tutor/index.html> Acessada em outubro de 2005.
10. IEEE. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. USA: IEEE, v. 77, n. 2, fev. 1989. 30 p.
11. LIMA, Teófilo Lourenço. **Manual Básico para Elaboração de Monografia**. 3.ed. Canoas: Editora da Ulbra, 2002.
12. LUFT, Joel Augusto. **Reconhecimento Automático de Voz Para Palavras Isoladas Independente Do Locutor**. 1994. Dissertação (Mestrado em Engenharia) – Departamento de Engenharia Elétrica, Universidade Federal do Rio Grande do Sul, Porto Alegre.
13. MAFRA, Alexandre Teixeira. **Reconhecimento Automático de Locutor em Modo Independente de Texto por Self-Organizing Maps**. 2002. Dissertação (Mestrado em Engenharia) - Escola Politécnica da Universidade de São Paulo, São Paulo.
14. MATLAB®, version 6.0.0.88 Release 12: The MathWorks Inc, 2000.
15. OPPENHEIM, Alan V, e SCHAFER, Ronald W.. **Discrete-Time Signal Processing**. USA: Prentice Hall, 1999.
16. QUEVEDO, Carlos Peres. **Cálculo Avançado**. Rio de Janeiro: Editora Interciência Ltda, 2000.
17. SPIEGEL, Murray R. **Manual de Fórmulas, Métodos e Tabelas de Matemática**. 2.ed. Makron Books.
18. TEBELSKIS, Joe. Review of Speech Recognition. In: **Speech Recognition using Neural Networks**. 1995 cap. 2, p. 9-71. Thesis (Doctor of Philosophy in Computer Science) - School of Computer Science, Carnegie Mellon University Pittsburgh, Pennsylvania.
19. The Scientist and Engineer's Guide to Digital Signal Processing by Steven W. Smith, Ph.D. California Technical Publishing ISBN 0-9660176-3-3 (1997).

Anexo 1: Listagem dos Programas

(1) Função *aquisicao_voz.m*

```
function [y,fs]=aquisicao_voz
%
% AQUISICAO_VOZ
%      [y,fs] = aquisicao_voz
%
%      Este dado tera um intervalo de tempo igual a 2.5 segundos e frequencia de amostragem de
44100Hz;
%
%      Retorna o vetor y referente ao som amostrado e a frequencia de amostragem fs

% Autor: Melissa Grahl
% 10/08/2005.
% VARIAVEIS ENVOLVIDAS:
%
% fs -> frequencia de amostragem em Hz (os valores padroes sao: 8000, 11025, 2250, e 44100 amostras por
segundo)
% tempo -> define o tempo total para leitura do som proveniente do microfone.
% N -> numero de bits que deve ser menor ou igual a 16

clear all % apaga as variaveis anteriores

fs=44100;
tempo=2.5; % tempo em segundos para captar o sinal
%N=16;

y = wavrecord(tempo*fs,fs); % faz a leitura do som
```

(2) Função *retirazero.m*

```

function [ysz,fs]=retirazero(y,fs)
%
% RETIRAZERO
%      [ysz,fs]=retirazero(y,fs)
%
%      Programa para identificar e retirar os silencias da palavra
%
% -> Variaveis de Entrada:
%      * sinal de voz na forma de um vetor y
%      * fs - frequencia de amostragem fs
%
% -> Variaveis de Saida:
%      * ysz -> vetor do sinal resultante
%      * fs - frequencia de amostragem fs

% Autor: Melissa Grahl
%
% 19/08/2005.

%tic

[tamanho, col]=size(y); % tamanho do vetor

[ynorm,fs]=normalizacao(y,fs);

E=(ynorm(1:1000:tamanho)).^2; % t=n*T -> t=1000*(1/44100) -> t= 22.67ms => maior do que a janela
sugerida na tese

Somatorio(1:size(E,1))=0;

for i=2:1:size(E,1)-2
    Somatorio(i)=sum(E(i-1:i+2));
end

%subplot(3,1,2);
%plot(Somatorio,'k')
%hold off
%subplot(3,1,1); plot(ynorm,'k')

% achando os limites

[ValorMax, posicaoMax]=max(Somatorio);

for i=posicaoMax:size(E,1) % limite da DIREITA

    if Somatorio(i)<=ValorMax*0.0015 % 0,15% do valor maximo

        LimiteDireita=(i-1)*1000+1;

```

```

        break
    end
end

for i=posicaoMax:-1:1 % limite da ESQUERDA

    if Somatorio(i)<=ValorMax*0.015

        LimiteEsquerda=(i-1)*1000+1;

        break
    end
end

ysz1(1:size(ynorm,1))=0;
ysz1(LimiteEsquerda:LimiteDireita)=ynorm(LimiteEsquerda:LimiteDireita);

%subplot(3,1,3);plot(ysz1,'m')

ysz=ynorm(LimiteEsquerda:LimiteDireita);

%toc

%wavplay(ysz,fs)

```

(3) Função *números_vs2.m*

```

function [valorResultante,indice]=numeros_vs2(d2,sr)
%
% Deve-se entrar com o vetor (d2) da palavra normalizada e sem os zeros e com a freq de amostragem (sr)
% Retorna o valor do alinhamento global e o indice equivalente a palavra pronunciada
%
indice=0;

d2a=d2(1:10:size(d2,1)); % Downsampling para tornar mais rapido
D2 = specgram(d2a,256,sr,256); % espectro com os valores padroes do matlab

for palavras=1:1:16 % Compara com todos as palavras padroes

    switch palavras

    case 1
        arquivo='cinco_nsz.wav';
    case 2
        arquivo='quatro_nsz.wav';
    case 3
        arquivo='seis_nsz.wav';
    case 4
        arquivo='tres_nsz.wav';
    case 5
        arquivo='sete_nsz.wav';
    case 6
        arquivo='dois_nsz.wav';
    case 7
        arquivo='oito_nsz.wav';
    case 8
        arquivo='um_nsz.wav';
    case 9
        arquivo='nove_nsz.wav';
    case 10
        arquivo='zero_nsz.wav';
    case 11
        arquivo='mais_nsz.wav';
    case 12
        arquivo='menos_nsz.wav';
    case 13
        arquivo='vezes_nsz.wav';
    case 14
        arquivo='dividido_nsz.wav';
    case 15
        arquivo='igual_nsz.wav';
    case 16
        arquivo='limpa_nsz.wav';
    end

    % Le uma das palvaras padroes, indicada pela variavel 'arquivo'

```

```

[d1,sr] = wavread(arquivo);
d1a=d1(1:10:size(d1,1)); % Downsampling para tornar mais rapido
D1 = specgram(d1a,256,sr,256); % espectro com os valores padroes do matlab

% Construct the 'local match' scores matrix as the cosine distance between the STFT magnitudes
% Calcula os alinhamentos locais entre os valores da STFT
SM = simmx(abs(D1),abs(D2)); % valor absoluto de um no. complexo e o seu modulo

% Use dynamic programming to find the lowest-cost path between the
% opposite corners of the cost matrix
% Note that we use 1-SM because dp will find the *lowest* total cost
[p,q,C] = dp(1-SM);

% a variavel 'valor Resultante' recebe o alinhamento global
% de cada palavra padrao testada
valorResultante(palavras)=C(size(C,1),size(C,2));

% Encontra o valor e o indice do menor alinhamento global,
% que e a palavra identificada como igual
[c,indice]=min(valorResultante);

end

```


(4) Função *simmx.m*

```

function M = simmx(A,B)
% M = simmx(A,B)
% calculate a sim matrix between specgram-like feature matrices A and B.
% size(M) = [size(A,2) size(B,2)]; A and B have same #rows.
% 2003-03-15 dpwe@ee.columbia.edu

EA = sqrt(sum(A.^2));
EB = sqrt(sum(B.^2));

%ncA = size(A,2);
%ncB = size(B,2);
%M = zeros(ncA, ncB);
%for i = 1:ncA
% for j = 1:ncB
% % normalized inner product i.e. cos(angle between vectors)
% M(i,j) = (A(:,i)*B(:,j))/(EA(i)*EB(j));
% end
%end

% this is 10x faster
M = (A*B)./(EA'*EB);

```

(5) Função *dp.m*

```
function [p,q,D] = dp(M)
% [p,q] = dp(M)
%   Encontra o resultado do alinhamento global da matriz M,
%   cujos valores sao proveniente da funcao simmx

[r,c] = size(M);

D = zeros(r+1, c+1);
D(1,:) = NaN;
D(:,1) = NaN;
D(1,1) = 0;
D(2:(r+1), 2:(c+1)) = M;

for i = 1:r;
    for j = 1:c;
        [dmax, tb] = min([D(i, j), D(i, j+1), D(i+1, j)]);
        D(i+1,j+1) = D(i+1,j+1)+dmax;
    end
end
end
```

(6) Função *aviso1.m*

```
function aviso1
%
% AVISO1
% Funcao destinada a solicitar que o usuario
% selecione, ao clicar com o botao direito do
% mouse sobre a tela, o inicio da aquisicao
%
% 12/09/2005.

texto=text(0.7,0,'Iniciar/Continuar', 'FontSize',14,'color','B');
axis off

but = 3;
while but == 3 % para prosseguir e preciso clicar sobre a tela com o botao
    [x,y,but] = ginput(1);
end
```

(7) Função *grafico.m*

```

function [variavel]=grafico(indice, a)
%
% GRAFICO
% [variavel]=grafico(indice, a)
%
% Funcao para exibir na tela o caractere identificado;
%
% Variaveis de entrada:
%         indice = aponta o numero identificado
%         a = serve para posicionar o caracter na tela
%
% Variaveis de saida:
%         variavel = caractere correspondente ao indice
%
% 14/09/2005.

%clc
x=-0.1;

for i=1:a
    x=x+0.1;
end

%switch a
%case 1
%    x=0;
%case 2
%    x=0.1 ;
%case 3
%    x=0.2;
%case 4
%    x=0.3;
%case 5
%    x=0.4;
%case 6
%    x=0.5;
%case 7
%    x=0.6;
%end

y=0.8;

switch indice

case 1
    variavel='5';
case 2
    variavel='4';
case 3
    variavel='6';
case 4
    variavel='3';

```

```
case 5
    variabel='7';
case 6
    variabel='2';
case 7
    variabel='8';
case 8
    variabel='1';
case 9
    variabel='9';
case 10
    variabel='0';
case 11
    variabel='+';
case 12
    variabel='-';
case 13
    variabel='*';
case 14
    variabel='/';
case 15
    variabel='=';
case 16
    close all hidden
end
texto=text(x,y,variabel, 'FontSize',58,'color','B');
axis off
```

(8) Função *resultado.m*

```

function resultado(indiceOp,x)
%
% RESULTADO
%     resultado(indiceOp,x)
%
%     Variaveis de entrada:
%         indiceOp = aponta para a operação matematica
%         x = vetor com os dados para realizar a operação
%
% Esta função calcula o resultado de uma das operações matematicas basicas seleccionadas,
% anteriormente, pelo sistema de reconhecimento.
%
% Retorna o resultado na janela ja aberta pelas etapas anteriores.
%
% 19/09/2005.

switch indiceOp
case 11
    op=1; %+
case 12
    op=2; %-
case 13
    op=3; %*
case 14
    op=4; %/
end

switch op
case 1
    resposta=x(1)+x(2);
case 2
    resposta=x(1)-x(2);
case 3
    resposta=x(1)*x(2);
case 4
    resposta=x(1)/x(2);
end

hold on

texto=text(0,0.35,sprintf('%d',resposta), 'FontSize',58,'color','y');
end

```

(9) Função *principal.m*

```

function principal
%
%      Função criada apenas para chamar o programa diretamente no prompt do Matlab
%
%      Versao 1
%
% 21/09/2005.

clear all
close all
clc

indice=0;
a=1;
i=1;

while indice~=15 % diferente de igual

    aviso1

    [y,fs]=aquisicao_voz;% Função que lera um novo arquivo de som
    beep % fim da detecção
    [ysz,fs]=retirazero(y,fs);
    [valorResultante, indice]=numeros_vs2(ysz,fs);
    variavel(a)=grafico(indice,a); % exibe na tela o num.

    if indice<=11 % e um numero?

        if a>=2 % se a for maior que 2 quer dizer que foram pronunciados 2 numeros

            k=strcat(variavel(1),variavel(2));% concatena as duas palavras - numeros
            x(i)=numeric(k); % trasforma de string para numero

        end

    else
        [valorResultante, indiceOp]=numeros_vs2(ysz,fs);
        variavel(a)=grafico(indiceOp,a); % identifica a operação
        i=i+1;
    end

end

if a==1

    switch indiceOp
    case 11
        text(0,0.5,'ERRO!', 'FontSize',68,'color','g');
        text(0,0.3,'Reinicie o Programa', 'FontSize',38,'color','m');
        break
    case 12

```

```
        text(0,0.5,'ERRO!', 'FontSize',68,'color','g');
        text(0,0.3,'Reinicie o Programa', 'FontSize',38,'color','m');
        break
    case 13
        text(0,0.5,'ERRO!', 'FontSize',68,'color','g');
        text(0,0.3,'Reinicie o Programa', 'FontSize',38,'color','m');
        break
    case 14
        text(0,0.5,'ERRO!', 'FontSize',68,'color','g');
        text(0,0.3,'Reinicie o Programa', 'FontSize',38,'color','m');
        break
    end
end

a=a+1;

end

resultado(indiceOp, x)
```


(10) Função *MelCoeficientes.m*

```

function [C, delta_C]=MelCoeficientes(sinal, fs)
% MEL COEFICIENTES
%      [C, delta_C]=MelCoeficientes(sinal)
%
% Função para o calculo dos coeficientes Mel Cepstrais e o Delta
%
% Entradas:
%      sinal = Sinal de voz amostrado
%      fs = frequencia de amostragem
% Saidas:
%      C = matriz com 16 Coeficientes Mel Ceptrais
%      delta_C = matriz dos deltas de primeira ordem dos MFCC coeficientes

% 11/11/2005

clear all
close all
clc

[y,fs]=wavread('zero_nsz.wav');
[yPre]=preEnfase(y);
dft=abs(specgram(yPre,2048,fs,hamming(1024),256));
figure
specgram(yPre,1024,fs,hamming(1024),256)
title('Espectrograma da palavra /zero/ depois da pre-enfase; janela Hamming(1024 pontos)')

[linha, num_janelas]=size(dft);

[freqCentral, limitesInferior, limitesSuperior]=frequencias;

for filtro=1:24 % troca o filtro 1 ate 24
    [h]=filtro_triangular(fs,limitesInferior(1,filtro),limitesSuperior(1,filtro));

    for frame=1:num_janelas
        Auxiliar(1:1024,frame)=dft(1:1024,frame).*h';% frame 1 x filtro
        SaidaBancos(filtro,frame)=sum(Auxiliar(1:1024,frame));
    end
end

end

% figure
% hold on
% plot(abs(SaidaBancos))
% title('Saidas do Banco de Filtros')
% xlabel('Filtros (1:24)')
% ylabel('Magnitude')

logaritmo=log10((abs(SaidaBancos)));

```

```

% figure
% plot(logaritmo) % logaritmo (1:26, frame) se quise imprimir so um frame
% title('Logaritmo do Banco de Filtros')
% xlabel('Filtros (1:24)')
% ylabel('Magnitude')

%%%%%%%%%%%%%%

DCT = dct(logaritmo);
% figure
% plot(abs(DCT))
% title('DCT - Discrete Cosine Transform - palavra /zero/')
% xlabel('Coeficientes')
% ylabel('Magnitude')

% interpolacao cepstral 16 coeficientes
% figure
% x=1:1:16;
% xx=1:0.1:16;
% y = interp1(x,abs(DCT(2:17,1)),xx,'spline');
% %subplot(2,1,1)
% plot(xx,y,'k')
% hold on
% plot([1:1:16],abs(DCT(2:17,1)), 'k.')
% title('Coeficientes Mel Cepstrais - palavra /zero/ - filtro 1')
% xlabel('Coeficientes')
% ylabel('Magnitude')

% figure

MFCC=abs(DCT);

% calculando os Delta Coeficientes Cepstrais

C=MFCC(3:18,:);% atenção! voce tem agora os frames nas linhas... e 16 coeficientes
[linha, coluna]=size(C);

for w=1:linha% frames

    for m=1:coluna % coeficientes

        if w==1
            delta_C(w,m)=(C(2,m)-C(1,m))/2;
        else
            delta_C(w,m)=(C(w,m)-C(w-1,m));
        end
    end
end

[deltaLINHA,deltaCOLUNA]=size(delta_C); % os delta fazem com que a matriz original MFCC dobre ja
que estes valores sao agregados
% figure
%plot(delta_C)

```

(11) Função *preEnfase.m*

```
function [yPre]=preEnfase(sinal)
%
% PRE ENFASE
%      [yPre]=preEnfase(sinal)
%
% Aplica um filtro ao vetor de entrada denominado sinal

numerador=[1];
denominador=;

yPre = filter([1 -0.95],[1],sinal);
```

(12) Função *frequencias.m*

```
function [freqCentral, limitesInferior, limitesSuperior]=frequencias
%
% FREQUENCIAS MEL
% [freqCentral, limitesInferior, limitesSuperior]=frequencias
%
% calculo dos deltas de freq. e calculo da distribuicao das freq centrais e
%
% numero de filtros = 24
% 10 igualmente espaçados
% 14 logaritmicamente espaçados 1.2*delta_anterior
% os filtros tem sua freq. espaçadas de 100mel = 64.95Hz = 65

%clc

delta(1:10)=130; % distribuição linear ate 10

for i=11:24
    delta(i)=1.2*delta(i-1);% distribuição logaritima de 11 ate 24 (sugerida no livro pg 131)
end

b(1)=65;

for i=2:24

    b(i)=b(i-1)+delta(i)/2; % frequencia central
end

limitesInferior=b-delta/2;
limitesSuperior=b+delta/2;
freqCentral=b;
```

(13) Função *filtro_triangular.m*

```
function [h]=filtro_triangular(fs,fc1,fc2)
%
% FILTRO TRIANGULAR
%      [h]=filtro_triangular(fs,fc1,fc2)
%
% fs = frequencia de amostragem (sample)
% fc1 = frequencia de corte limite inferior
% fc2 = frequencia de corte limite superior
%
% retorna h que e o filtro FIR passa faixa janelado de Bartlett

wc1=(2*fc1*pi)/fs; % freq de corte inferior em rad/s normalizada (fs)
wc2=(2*fc2*pi)/fs; % freq de corte superior em rad/s normalizada (fs)

M=1024;% tamanho da janela
delta_w=(6.1/(M*fs))*pi; % janela de Bartlett normalizada => delta_w = wp-ws (p=pass e s=stop)

ws1=wc1-(delta_w/2);
wp1=wc1+(delta_w/2);

wp2=wc2-delta_w/2;
ws2=wc2+delta_w/2;

hd=passa_baixa(wc2,M)-passa_baixa(wc1,M);% ideal passa-faixa
w=(bartlett(M))'; % janela triangular Bartlett
h=hd.* w; % filtro x janela triangular
```

(13) Função *passa_baixa.m*

```
function hd=passa_baixa(wc,M)
% Filtro Passa-Baixa
% -----
% [hd] = passa_baixa(wc,M)
%
% wc = frequencia de corte em radianos
% M = tamanho do filtro

alfa = (M-2)/2;
n=[0:1:(M-1)];
m = n-alfa+eps;
hd=sin(wc*m)./(pi*m);
```

(14) Função *sensor.m*

```
function [s]=sensor
%
%   SENSOR
%       [s]=sensor
%
% Le o bit 0 da porta paralela

DIO = digitalio('parallel','LPT1');

in_lines = addline(DIO,0,0,'in');

s=getvalue(DIO); % le o bit 0 da paralela

% delete(DIO); % apaga a configuracao feita
```

Anexo 2: Faixa de Frequências Audíveis

Anexo 3: Dados Técnicos sobre os equipamentos

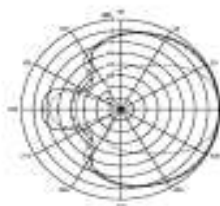
(1) Microfone

B-1**STUDIO CONDENSER MICROPHONE****Technical Specifications****ENGLISH**

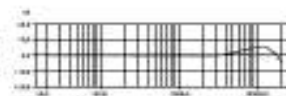
Version 1.0 December 2001



Transducer type:	condenser, 1" single diaphragm
Operating principle:	pressure gradient
Polar pattern:	cardioid
Connection:	gold-plated balanced XLR connector
Open circuit voltage at 1 kHz:	-34 +/- 2 dBV (0 dBV = 1 VPa)
Open circuit sensitivity:	20 mV/PA
Frequency range:	20 Hz - 20 kHz
Level attenuation:	-10 dB (switchable)
Low-Cut filter:	6 dB/Octave at 75 Hz (switchable)
Max. SPL (1% THD @ 1 kHz):	138 dB (0 dB), 148 dB (-10 dB)
Equivalent SPL (IEC 268-4):	13 dB-A
Signal-to-noise ratio re 1 Pa:	81 dB A-weighted
Nominal impedance:	50 Ω
Load impedance:	> 1 k Ω
Supply voltage:	+48 V
Supply current:	3 mA
Dimensions:	\varnothing : 58 mm, length: 174 mm
Weight:	0,461 kg



Polar pattern



Frequency response

BEHRINGER is constantly striving to maintain the highest professional standards. As a result of these efforts, modifications may be made from time to time to existing products without prior notice. Specifications and appearance may differ from those listed or illustrated.

The information contained in this sheet is subject to change without notice. No part of this sheet may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording of any kind, for any purpose, without the express written permission of BEHRINGER Spezielle Studiotechnik GmbH.

BEHRINGER is a registered trademark. ALL RIGHTS RESERVED.

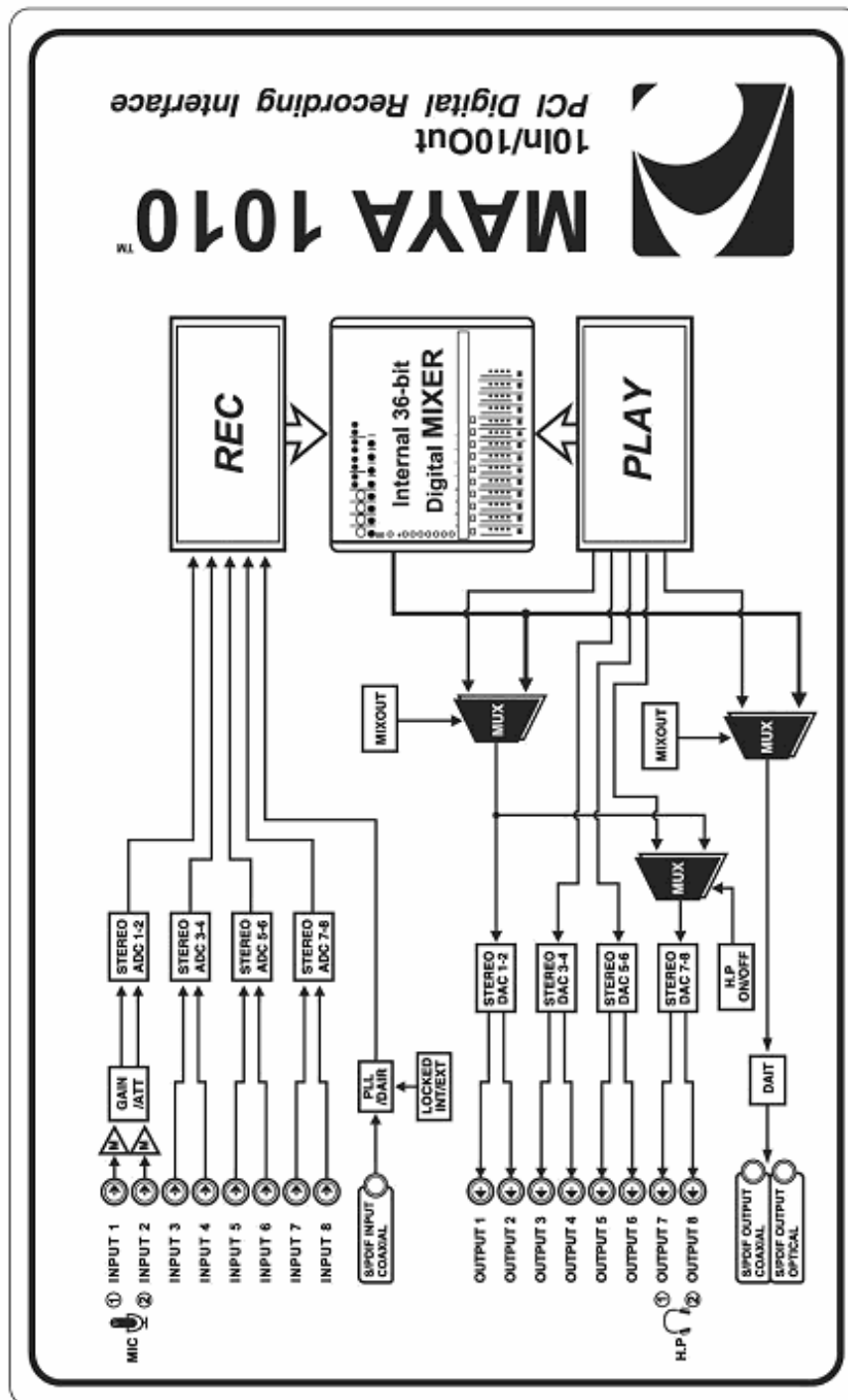
© 2001 BEHRINGER Spezielle Studiotechnik GmbH

BEHRINGER Spezielle Studiotechnik GmbH, Harro-Martin-Schäfer-Str. 36-38, 47577 Mönchenmünch, Germany
Tel.: +49 (0) 21 54 / 32 35-0, Fax +49 (0) 21 54 / 32 06-33



www.behringer.com

(2) Placa de Som



(3) Mesa de Som

-

Anexo 4: Testes Detecção dos limites

