UNIVERSIDADE FEDERAL DE UBERLÂNDIA

FACULDADE DE ENGENHARIA ELÉTRICA

CARLOS ALBERTO LOPES DA SILVA

UM SISTEMA PARA CLASSIFICAÇÃO E ANÁLISE DE QUALIDADE DE SANIDADE DE CRUZETAS DE MADEIRAS

UBERLÂNDIA – MG

SETEMBRO - 2016

CARLOS ALBERTO LOPES DA SILVA

UM SISTEMA PARA CLASSIFICAÇÃO E ANÁLISE DE QUALIDADE DE SANIDADE DE CRUZETAS DE MADEIRAS

Tese de Doutorado apresentada como requisito parcial à obtenção do grau de Doutor em Ciências, pelo programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Uberlândia.

Orientador: Dr. Prof. Luciano Vieira Lima

Banca Examinadora:
Prof. Dr. Luciano Vieira Lima
Prof. Dr. Luciano Martins Neto
Prof. Dra. Junia Magalhães Rocha
Prof. Dr. Igor Santos Peretta
Dr. Reny Cury Filho

UBERLÂNDIA – MG

SETEMBRO - 2016

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU, MG, Brasil.

C284s 2016

Silva, Carlos Alberto Lopes da, 1968-
Um sistema para classificação e análise de qualidade de sanidade de cruzetas de madeiras / Carlos Alberto Lopes da Silva. - 2016.
190 f. : il.

Orientador: Luciano Vieira Lima.
Tese (doutorado) - Universidade Federal de Uberlândia, Programa de Pós-Graduação em Engenharia Elétrica.
Inclui bibliografia.

1. Engenharia elétrica - Teses. 2. Madeira - Deterioração - Teses. 3. Postes (Engenharia - Teses. 4. Linhas elétricas - Postes e torres - Teses. I. Lima, Luciano Vieira, 1960-. II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

CDU: 621.3

DEDICATÓRIAS

Dedico este trabalho aos meus pais Geraldo Silva (in memorian) e Ivêta Lopes da Silva que sempre deram total apoio e incentivo a minha carreira com muito trabalho e esforço. Dedico também este trabalho ao meu segundo pai de coração, Wilson dos Reis (in memorian), que acreditou em todos os momentos em meu futuro, sem ele talvez eu não teria iniciado este caminho de sucesso. Todos eles foram o meu motivo de inspiração e suporte de vida. Dedico também este trabalho a minha companheira Anna Paula Curi pela compreensão, incentivo e companheirismo, mesmo nos piores momentos da vida dela, atravessando uma luta e vencendo um câncer, para juntos obtermos este sucesso.

AGRADECIMENTOS

Meus agradecimos ao Prof. Dr. Luciano Vieira Lima que, como orientador e amigo, direcionou e solucionou os diversos problemas encontrados neste trabalho.

Meus agradecimentos ao Prof. Antônio Eduardo Costa pelos esclarecimentos e toda a sua disponibilidade em solucionar os problemas diversos encontrados.

Meus agradecimentos a secretaria da pós-graduação Cinara Mattos pelo impecável atendimento e paciência com nós orientados.

Meus agradecimentos ao amigo e companheiro de doutorado, Rubens Barbosa Filho, pelas dicas e esclarecimentos referentes a este trabalho e vida profissional, devo muito a ele.

Meus agradecimentos ao grande amigo Divino Rosa Silva Ferreira (DJ Divinex) pela ajuda no desenvolvimento deste trabalho, principalmente a parte de hardware, incentivo para nunca desistir e companheirismo nos piores momentos de minha vida. Ele “é o cara”.

Meus agradecimentos a todos os membros da banca que aceitaram participar deste trabalho com uma contribuição excepcional.

Meus agradecimentos a todos os profissionais da Empresa CEMIG de Uberlândia-MG que foram essenciais no êxito deste trabalho.

Meus agradecimentos a todos os meus amigos que sempre esperaram este momento de vitória em minha vida.

Agradeço a todos que de uma forma ou outra participaram direta ou indiretamente nesta etapa tão importante em minha vida e carreira profissional.

RESUMO

Problemas de deterioração em madeiras são constantes e envolvem muitos estudos para determinar o tempo útil de vida das mesmas em construções, postes, cercas, dormentes, etc. Neste trabalho é desenvolvida uma técnica para determinar a vida útil de madeiras utilizadas em cruzetas para sustentação de cabos elétricos. A vida útil de madeiras de postes é definida como estado de sanidade. Nos estudos são considerados três estados de sanidade onde, para cada estado, foram feitas diversas coletas de dados. Os dados foram classificados com o uso de captura de sinais através de microfone específico, extração dos coeficientes FFT, LPC, MFCC, ENERGIA, ZERO CROSSING e ÁREA para determinar as características destes sinais. O algoritmo K-NN foi utilizado para treino das características de cada estado de sanidade. Ao final foi gerado um equipamento específico para o teste em campo das sanidades propostas.

Palavras-chave: Deterioração em Madeira. Vida Útil de Madeiras. Estado de Sanidade. K-Médias. K-NN.

ABSTRACT

Deterioration problems in woods are constant and involve many studies to determine the useful life of the same in buildings, poles, fences, sleepers, etc. In this work, a technique is developed to determine the useful life of wood used in crosspieces to support electrical cables. The useful life of poles of wood is defined as health status. In the studies are considered three health states where, for each state, different data collections were made. The data were classified with the signal capture through use of specific microphone, extraction of FFT, LPC, MFCC, ENERGY, ZERO CROSSING and AREA coefficients to determine the characteristics of these signals. The K-NN algorithm is used for training the characteristics of each state of health. At the end it was generated specific equipment for the field testing of sanities proposals.

Keywords : Deterioration in Wood. Life Timber . State of Sanity. K-Means. K - NN .

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CEMIG – Centrais Elétricas de Minas Gerais

LPC – Linear Prediction Coefficients

MFCC - Mel Frequency Cepstral Coefficients

ZC - Zero Crossing

SPACSIM - Sistema Portátil de Análise de Cruzeta sem Interferência do Meio

PCA – Principal Components Analysis (Análise dos Componentes Principais)

SUMÁRIO

1 – INTRODUÇÃO

A madeira é o material mais amplamente utilizado pelo homem desde os primórdios dos tempos. O homem sempre percebeu que diferentes tipos de madeira, quando expostos a condições iguais, decompunham de forma diferente e quando expunha madeiras iguais a condições ambientais diferentes, a degradação também acontecia de forma diferente (Vidor, 2003).

Na transmissão de energia elétrica sempre foram utilizados postes e cruzetas de madeira para fixação dos cabos elétricos. Com o passar do tempo e com a exposição destes postes e cruzetas pela ação do tempo (sol, chuva, ventos, etc...) a madeira torna-se frágil, podendo quebrar e causar sérios acidentes na natureza e para o homem.

Não somente nos casos de postes e cruzetas de madeiras, mas realizar a análise da vida útil de madeiras é fundamental para vários setores da cadeia produtiva, tais como cercas de empresas rurais, postes de distribuição de energia e telefonia, dormentes de estrada de ferro e várias estruturas de distribuição de energia urbana e rural. Defeitos ocorridos nestes equipamentos acarretam danos materiais, físicos e financeiros com relevante frequência.

Na utilização de postes e cruzetas de madeiras como suporte para as redes elétricas urbanas e rurais, o conhecimento do estado de conservação das madeiras e informações sobre a qualidade da estrutura das mesmas faz-se necessário para a tomada de decisões sobre a troca ou não destes postes ou cruzetas pelos departamentos competentes das empresas do setor elétrico. Defeitos ocorridos nos postes e cruzetas acarretam danos materiais, físicos e financeiros com relevante frequência (Lepage, 1986).

1.1 - O Problema das Cruzetas

As cruzetas de madeira foram criadas para sustentar os cabos para a transmissão de energia elétrica.

São peças de madeira de medidas (tamanho, largura e altura) e furações definidas através da NBR 8458 e NBR 8459 (Associação Brasileira de Normas Técnicas, 1984a) (Associação Brasileira de Normas Técnicas, 1984b) (CODI, 1992).

A vida útil de uma cruzeta de madeira é em média 30 anos.

As ações da natureza (sol, chuva, ventos, umidade, exposição a diversos agentes) causam a deterioração destas cruzetas. Com o passar do tempo faz-se necessário a troca das mesmas para a prevenção da rede elétrica e acidentes.

Estas trocas não possuem uma data correta. É necessária uma inspeção visual e por toque (batida) da cruzeta de tempos em tempos para que seja agendada a sua troca, caso seja necessária.

A falta de inspeção e a falta de trocas nos tempos oportunos podem levar à quebra das mesmas, causando danos elétricos tais como interrupção do fornecimento de energia elétrica para as cidades, parada da produção de empresas essenciais em novas vidas, acidentes com a queda das cruzetas e/ou cabos elétricos em seres humanos ou animais.

A inspeção é feita no local de instalação da cruzeta onde o inspetor deverá chegar perto da cruzeta, ou seja, deverá ser elevado a altura da cruzeta para observar e efetuar a batida (toque) na cruzeta para verificar o seu estado de vida útil.

Para esta inspeção diversas ações são necessárias, tais como agendamento para desligamento da rede elétrica, proteção dos inspetores contra riscos de queda e choques elétricos, isolamento do local, etc. Um processo de troca é mostrado na figura 1.

Figura 1 - Manobra para Troca de Cruzeta

Fonte: CEMIG D

A troca das cruzetas de madeira feita através de inspeção visual pode ocasionar trocas indevidas, ou seja, são trocadas cruzetas que ainda poderiam ser utilizadas por um maior

tempo. Isto ocasiona um grande dispêndio financeiro para as concessionárias de distribuição de energia elétrica.

Um caso típico é o enfrentado pela CEMIG (Centrais Elétricas de Minas Gerais), a qual possui aproximadamente 1,7 milhões de cruzetas de madeira instaladas em suas redes de distribuição, originando 70 mil trocas anuais devido à provável perda de sua vida útil. Existe dificuldade de identificação e definição na priorização de substituição destes componentes uma vez que as inspeções visuais são subjetivas dependendo do grau de conhecimento/experiência do inspetor. Devido a esta subjetividade das inspeções visuais ocorrem 30% de trocas de elementos ainda sadios causando prejuízos financeiros.

Desta forma necessita-se do desenvolvimento de um equipamento que possa realizar a análise das cruzetas de madeira, a partir do solo, no qual um sistema eletrônico emita e colete um sinal (potenciais induzidos e evocados), o qual será avaliado por um software que indicará, através de parâmetros pré-estabelecidos e estatisticamente pré-determinados, qual o estado de vida útil da mesma: bom, ruim, indefinido, etc.

A vida útil de uma cruzeta de madeira é definida em termos de sanidades. Estas sanidades são previamente definidas pelos inspetores que, inicialmente relatam que os estados de sanidades são separados em cinco (05) grupos:

- Excelente ou Nova
- Médio
- Boa
- Regular
- Péssima

1.2 - Trabalhos Correlatos

Alguns estudos estão sendo feitos para melhorar o tempo de vida útil de madeiras usadas em cruzetas.

(Vidor, 2003) mostrou em seu trabalho um processo para inspeção e tratamento de postes de madeira, conseguindo com isto aumentar consideravelmente o tempo de vida útil das madeiras mas mostrou também que há a necessidade de implantação de um sistema de controle e manutenção pois estruturas em um estado de conservação

degradante podem por vezes comprometer o fornecimento de energia elétrica a um número representativo de consumidores.

A Companhia Paulista de Força e Luz (CPFL) em conjunto com a Universidade Estadual de Campinas (Unicamp) e Agricef Soluções Tecnológicas (CPFL, Unicamp, & Tecnológicas, 2012) estão desenvolvendo um sistema para avaliar a situação de postes e cruzetas de madeira através do uso de ultrassom, mas ainda encontra-se em fase de resultados experimentais até o presente momento deste trabalho e praticamente sendo utilizado somente em postes e não em cruzetas, com um bom nível de acertos. Neste trabalho é necessária a inserção de pequenos pinos na madeira, pinos estes que sofrem batidas manuais de pequenos martelos para que entre dentro da madeira e desta forma o sistema de ultrassom funcione corretamente.

1.3 – Objetivos e Justificativa do Trabalho

Os estudos propostos neste trabalham focam sobre como testar a vida útil de cruzetas de madeira fixadas no topo de postes de transmissão de energia elétrica sem a necessidade de o inspetor ter de subir nos postes, ou passar por perigos elétricos para chegar à cruzeta, foco do teste.

No sistema já existente proposto por (CPFL et al., 2012), como já dito anteriormente, é necessária a inserção de pequenos pinos de metal no interior da madeira para que seja possível o sistema de ultrassom funcionar.

No caso específico das cruzetas torna-se inviável este tipo de manobra, pois seria necessário todo o processo desgastante de desligamento da rede elétrica com agendamento prévio, isolamento e segurança dos inspetores de campo, além de um gasto dispendioso de tempo para o teste de ultrassom.

Estudos em distribuidoras de energia elétrica mostraram alguns aspectos essências para a análise de alguns tipos de vida útil de cruzetas. Ao final será proposto um sistema que faça a análise da cruzeta definindo o estado de sanidade da mesma e gerando um banco de dados com informações sobre as possíveis datas para a troca das mesmas.

Portanto, como justificativa, neste trabalho é proposto um sistema que, com uma única batida em locais específicos da cruzeta será possível determinar imediatamente a sua sanidade.

Características do modelo proposto:

- Uso com vara de manobra
- Teste imediato da cruzeta
- Não será necessário o agendamento de desligamento da rede elétrica
- Total segurança para os inspetores de campo
- Uso em qualquer local, pois o equipamento poderá ser utilizado com energia elétrica proveniente da rede elétrica (110 ou 220 Volts) ou através do uso de bateria de 12 Volts
- Cadastro das cruzetas em banco de dados para agendamento da troca ou próxima vistoria

Neste trabalho foi adotada a denominação SANIDADE DA MADEIRA como sendo um estado da vida útil desta madeira, como sendo nova, boa, ruim, muito ruim, péssima, etc.

1.4 - Organização do Trabalho

Este trabalho está organizado em seis (06) capítulos.

No Capítulo 1 é feita uma explanação do problema de sanidade de cruzetas de madeiras, trabalhos desenvolvidos nesta área e são apresentados os objetivos de estudo deste trabalho.

No Capítulo 2 é mostrado o estudo sobre sanidade de cruzetas e a forma como são definidos os estados de sanidade.

No Capítulo 3 é feita a descrição das ferramentas utilizadas no desenvolvimento deste trabalho, com o uso de Transformadas de Fourier, Coeficientes LPC e MFCC, Energia do Sinal, Zero Crossing e Área da região do sinal amostrado. São mostradas também como estas ferramentas irão resultar no reconhecimento dos estados de sanidades de cruzetas.

No Capítulo 4 são mostradas todas as etapas de desenvolvimento do hardware utilizado para a extração das características das cruzetas bem como para o reconhecimento futuro destas características.

No Capitulo 5 são mostrados os resultados iniciais para o treinamento/reconhecimento dos diversos tipos de sanidades de cruzetas bem como a validação dos algoritmos propostos utilizando o hardware definido no Capítulo 3.

No Capítulo 6 são mostradas as conclusões provenientes dos resultados obtidos com este trabalho.

2 - ESTUDOS SOBRE CRUZETAS

A madeira é um material produzido a partir do tecido formado pelas plantas lenhosas com funções de sustentação mecânica. É um material orgânico, sólido, de composição complexa, onde predominam as fibras de celulose e hemicelulose unidas por lenhina. Caracteriza-se por absorver facilmente água e por apresentar propriedades físicas diferentes, consoante a orientação espacial.

As plantas que produzem madeira (árvores) são perenes e lenhosas, caracterizadas pela presença de caules de grandes dimensões, em geral denominados troncos, que crescem em diâmetro ano após ano.

Pela sua disponibilidade e características, a madeira foi um dos primeiros materiais a ser utilizado pela humanidade, mantendo, apesar do aparecimento dos materiais sintéticos, uma imensidade de usos diretos e servindo de matéria-prima para múltiplos outros produtos. É também uma importante fonte de energia, sendo utilizada como lenha para cozinhar e outros usos domésticos numa parte importante do mundo.

A sua utilização para a produção de polpa está na origem da indústria papeleira e de algumas indústrias químicas nas quais é utilizada como fonte de diversos compostos orgânicos.

A sua utilização na indústria de marcenaria para fabricação de móveis é uma das mais expandidas, o mesmo acontecendo na sua utilização em carpintaria para construção de diversas estruturas, incluindo navios.

Sendo um material naturalmente resistente e relativamente leve, é frequentemente utilizado para fins estruturais e de sustentação de construções é muito utilizado em arquitetura e engenharia civil.

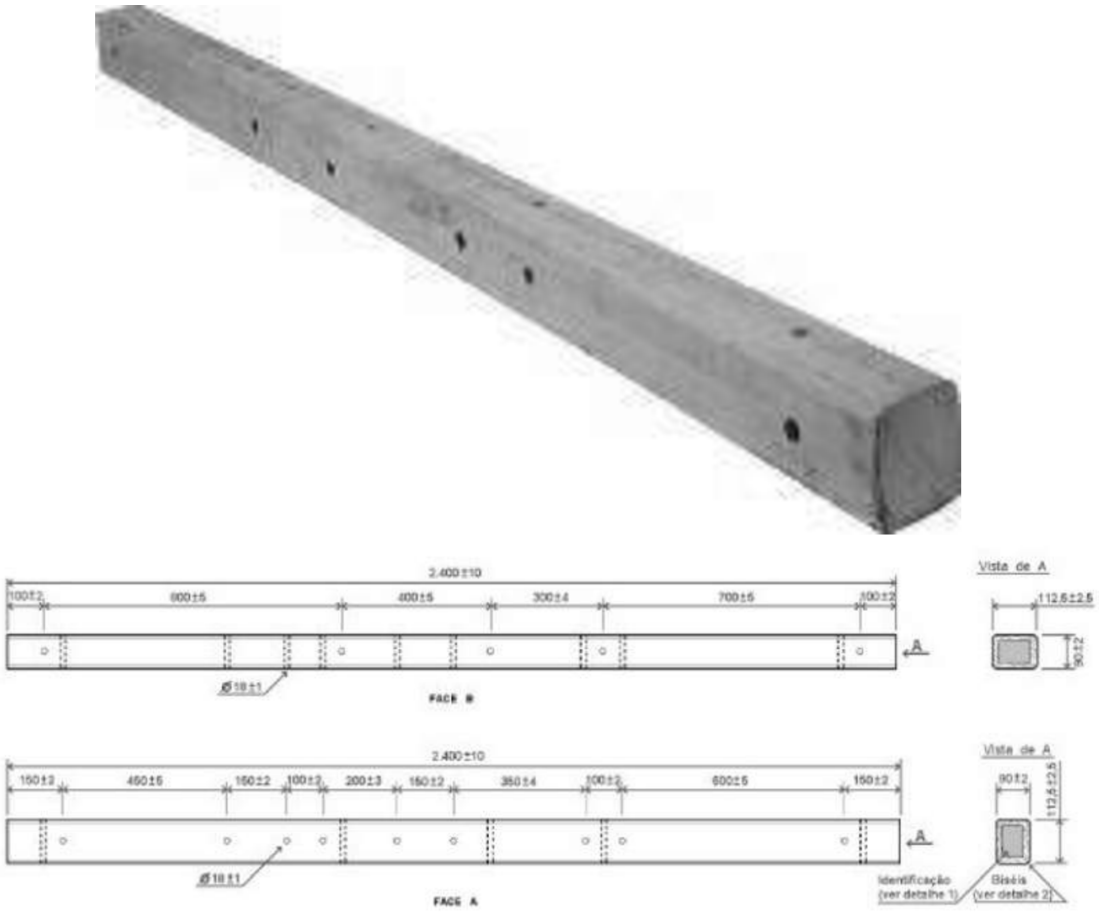
A indústria florestal ocupa vastas áreas da Terra e a exploração de madeira em florestas naturais continua a ser uma das principais causas de desmatamento e de perda de habitat para múltiplas espécies, ameaçando severamente a biodiversidade a nível planetário.

Sendo um material naturalmente resistente e relativamente leve, é frequentemente utilizado para fins estruturais e de sustentação de construções.

Na área de distribuição elétrica a madeira é amplamente utilizada para sustentação de cabos de média tensão, devido a suas características de isolamento e resistência física.

Estas madeiras são denominadas cruzetas e suas especificações são mostradas na figura 2.

Figura 2 – Modelo de Cruzeta



Fonte: CEMIG D e (CODI, 1992)

Segundo a [NBR 8458] a confecção de cruzetas devem ser de puro cerne ou cerne alburno com tratamento preservativo.

Na atualidade as cruzetas de madeiras estão sendo substituídas por cruzetas de polímero. (Dalfré, 2007) mostra como são desenvolvidas estes tipos de cruzetas de polímero para satisfazer as características da [NBR 8458].

2.1 - Troca de Cruzetas

Através de vistas técnicas a CEMIG D (CEMIG Distribuição) foram relatadas as especificações que são utilizadas para determinação de quais cruzetas são defeituosas e necessitam serem trocadas.

Modelagem para a troca de cruzetas:

a) Inicialmente são feitas rotas de inspeção. Estas rotas de inspeção são feitas de ano em ano, aproximadamente.

b) Nesta rota de inspeção é feito o teste visual das cruzetas dos postes. Neste teste visual é observada a coloração da cruzeta, manchas brancas provenientes de fungos e apodrecimento, conforme mostram as Figuras 3, 4 e 5.

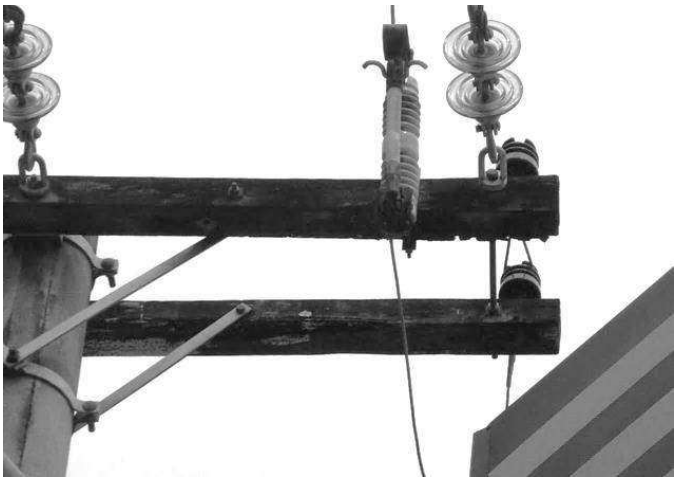
Figura 3 - Cruzetas já inspecionadas e que devem ser trocadas

Fonte: Acervo Pessoal

Figura 4 - Manchas Brancas, sinais de apodrecimento

Fonte: Acervo Pessoal

Figura 5 - Cruzetas com pontas quebradas, já necessitando serem trocadas



Fonte: Acervo Pessoal

c) Quando a inspeção visual não é suficiente é feito um teste com batimento com martelo na cruzeta. O som do batimento irá definir se a cruzeta está na hora de ser trocada (ocas). Mas para este batimento há a necessidade de isolar a área, posicionar caminhão, às vezes parar o trânsito, entre outras manobras. Desta forma torna-se um trabalho complicado e que dispensa muito tempo e organização, conforme mostra a Figura 6.

Figura 6 - Visão da pessoa que está fazendo a observação

Fonte: Acervo Pessoal

d) As cruzetas que são definidas como que devem ser trocadas entram em um programa de troca. Este programa define a data correta que deverá ser efetuada a troca da cruzeta. Todas as manobras são previamente programadas e agendadas.

e) Para a manobra de troca de uma cruzeta é feito um Pedido de Interrupção (PI) programada com vários dias de antecedência para que os usuários da energia elétrica façam a sua programação.

f) A troca da cruzeta danificada é feita com a linha de alimentação de média tensão energizada, ou seja, uso de linha viva.

g) Este tipo de troca somente é efetuada por equipes da própria CEMIG D, por se tratar de um trabalho de alto risco e necessitar de pessoal capacitado.

h) Trocas onde todo o sistema elétrico é desligado são efetuadas por empresas terceirizadas, não sendo necessária a intervenção da própria CEMIG D. A Figura 7 mostra uma manobra para a troca de cruzetas com linha viva.

Figura 7 - Manobra para a Troca de Cruzetas com Linha Viva

Fonte: Acervo Pessoal

Os equipamentos de segurança são utilizados e inspecionados periodicamente.

Luvas e materiais de uso dos funcionários são inspecionados as cada três (03) meses. Caminhões são inspecionados a cada seis (06) meses.

O caminhão utilizado para esta manutenção é equipado com braço mecânico isolado para não haver aterramento da energia elétrica através destes.

2.2 - Considerações sobre Possibilidades de Avaliações Errôneas na Definição do Nível de Deterioração de uma Cruzeta

Somente a observação visual pode causar erros de avaliação principalmente se a mesma for feita apressadamente ou somente da parte de baixo do poste, a uma distância não muito boa do inspetor da cruzeta.

Uma cruzeta possui quatro lados para serem observados. Com uma visão da parte de baixo do poste somente três lados poderão ser observados com qualidade a olho nu ou binóculo. São os lados laterais e o lado de baixo. O lado superior somente poderá ser observado através de manobras específicas com pessoal especializado com algum equipamento que possa ser manobrado da parte de baixo do poste.

Desta forma muitas cruzetas são classificadas em um nível errôneo. As Figuras 8 e 9 mostram isto claramente, pois neste caso tem-se uma cruzeta que possivelmente estava em bom estado olhando nas laterais e na parte de baixo, mas após a sua troca foi verificado que a mesma estava em um estado muito mais avançado de deterioração.

Figura 8 - Cruzeta observada de baixo com as laterais e parte de baixo boas

Fonte: Acervo Pessoal

Figura 9 - A mesma cruzeta observada na parte de cima

Fonte: Acervo Pessoal

A parte frontal (corte) de uma cruzeta também deve ser observada para a análise. Uma cruzeta também pode estar com as laterais e parte de baixo em bons estados, até mesmo a parte de cima também pode estar em bom estado, mas observando o corte da seção transveral poderá ser visto que a mesma já está muito corrompida e até mesmo totalmente podre, como mostra a Figura 10.

Figura 10 - Cruzeta com o centro deteriorado

Fonte: Acervo Pessoal

Além do processo de observação visual da cruzeta há também o processo de observação do áudio proveniente da batida de uma marreta a cruzeta. Através deste processo o inspetor avalia não somente o estado visual da cruzeta, mas também a ressonância do som proveniente desta batida.

Cruzetas que são classificadas com qualidade boa possuem um som mais forte, de pouca duração e ressonância. Cruzetas que são classificadas com qualidade ruim possuem um som mais “oco”, de maior duração e ressonância. Cruzetas com estado de sanidade considerada Emergencial ficam com um som entre os dois anteriores.

2.3 – Conclusões

Com todas estas colocações mostra-se claramente o grau de dificuldade que é a troca de uma cruzeta, onde é necessário o envolvimento de diversos profissionais, uma quantidade enorme de equipamentos e avaliações humanas que podem gerar erros com grandes desperdícios financeiros, pois este processo de avaliação depende de cada inspetor.

No capítulo seguinte serão mostradas as metodologias adotadas para a análise das cruzetas utilizando técnicas de inteligência artificial para a solução do problema.

3 - METODOLOGIA

Nesta etapa serão definidos os passos para a geração do sistema inteligente para a separação das sanidades de cruzetas. Esta etapa será dividida em:

- Critérios para a Análise e Seleção de Cruzetas Danificadas: estudo sobre como foram gerados os níveis de sanidade de classificação das cruzetas
- Estudo Estatístico: estudo necessário para a validação das amostras obtidas.
- Processamento: estudo sobre os métodos e algoritmos matemáticos utilizados
- Algoritmo de Análise e Reconhecimento: etapa que mostra como os dados foram gerados, organizados e analisados.

3.1 - Critérios para a Análise e Seleção de Cruzetas Danificadas

Após diversas observações e reuniões com pessoal especializado do setor de Manutenção em Redes da CEMIG D foi concluído que não são utilizados níveis em termos de porcentagem para a análise das cruzetas.

Inicialmente foram definidos os seguintes níveis de sanidade de cruzetas:

- Excelente ou Nova
- Médio
- Boa
- Regular
- Péssima

Este tipo de análise não é utilizado pela equipe de Manutenção em Redes. Para um bom casamento de informações e o desenvolvimento de um sistema especialista que se adéque as necessidades da CEMIG D foram adotadas as nomenclaturas usadas entre os inspetores de redes. Esta nomenclatura é divida em:

a) Emergencial
b) Urgência
c) 1 mês
d) 2 meses
e) 6 meses
f) 1 ano

Com esta nomenclatura são definidos seis (06) níveis de classificação (EMERGENCIAL, URGÊNCIA, 1 MÊS, 2 MESES, 6 MESES e 1 ANO) onde o primeiro é o que mais deverá ser observado pois trata-se de uma troca sem programação. Após novos estudos e em comum acordo com os novos formatos utilizados pela CEMIG D, os seis níveis anteriormente definidos foram reduzidos somente para três (03) níveis:

a) **Emergencial**: Cruzetas que deverão ser trocadas imediatamente sem um agendamento com um longo prazo. Neste caso a cruzeta já está totalmente deteriorada ou quebrada, conforme mostra a Figura 11.

Figura 11 - Cruzeta Emergencial

Fonte: Acervo Pessoal

b) **Urgencial** – São cruzetas que, teoricamente, ainda suportam ficar nos postes por no máximo um mês. São cruzetas que possuem um grau de durabilidade maior que as cruzetas consideradas no estado Emergencial. Estas cruzetas podem ter um agendamento com uma boa antecedência para a sua troca. Neste caso a cruzeta apresenta uma visualização aparentemente boa mas com uma observação mais profunda poderá quebrar em pouco tempo, conforme mostra a Figura 12.

Figura 12 - Cruzeta Urgencial

Fonte: Acervo Pessoal

c) **1 Ano** - São cruzetas consideradas em ótimo estado de conservação. Após um ano estas cruzetas serão novamente inspecionadas para saber se devem mudar de nível ou não. Neste caso a cruzeta apresenta uma boa visualização e também uma boa consistência, conforme mostra a Figura 13.

Figura 13 - Cruzetas de 1 Ano

Fonte: Acervo Pessoal

3.2 – Estudo Estatístico

Nesta etapa é mostrado o processo de classificação das cruzetas de madeira, mostrando os passos do desenvolvimento das rotinas de pré-processamento, extração de *Features* e Classificação dos Sinais.

Para a validação dos dados, com exclusão de amostras desnecessárias e localização de amostras com maior energia para serem utilizadas como referência, serão utilizados estudos estatísticos utilizando a técnica de Média, Mediana, Variância, Desvio Padrão, Quartis e Percentis, onde toda definição teórica é vista em (Medri, 2011), onde os 16 pontos em torno do ponto de maior energia serão utilizados para serem as amostras de referência e os pontos abaixo do ponto de Quartil e acima do ponto de Percentil serão desprezados por se tratarem de pontos de pouca energia ou ruidosos (Faceli, Lorena, Gama, & Carvalho, 2011). Este estudo a ser feito leva-se em consideração todo o universo de amostras P, constituídas de N indivíduos, para que sejam descartadas as amostras com pouca representatividade e selecionadas as amostras com maior representatividade. Consideremos um sinal amostrado com sendo um conjunto A de pontos, conforme a Equação 1:

$$A = \{a_1, a_2, \ldots, a_N\} \quad (1)$$

onde a_i, com i=1,2,...N, é um ponto amostrado.

O conjunto X de amostras para cada sanidade será de N_S amostras, onde S é a quantidade de amostras da sanidade, conforme Equação 2:

$$X = \{A_1, A_2, \ldots, A_S\} \quad (2)$$

Os cálculos a seguir serão mostrados para um único grupo de sanidade. Para todas as sanidades os mesmos cálculos também deverão ser efetuados, não havendo necessidade da repetição do equacionamento, pois os resultados não possuem dependência entre os grupos.

Para cada amostra de cada grupo (sinal de áudio capturado) é calculada a média e variância, conforme mostram as Equações 3 e 4

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \quad (3)$$

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu)^2 \quad (4)$$

onde N é a quantidade de pontos de cada vetor do grupo.

Para cada grupo de sanidade é gerado um vetor com as médias e variâncias de cada amostra, conforme Equação 5 e 6:

$$T = [\mu_1\ \mu_2 \ldots \mu_M] \quad (5)$$

$$\sigma^2 T = [\sigma_1^2 \sigma_2^2 \ldots \sigma_M^2] \quad (6)$$

onde M é a quantidade de vetores do grupo.

Calcula-se também a média e a variância para cada grupo de amostras através das Equações 7 e 8.

$$\mu_G = \frac{1}{M}\sum_{i=1}^{M} \mu T_i \quad (7)$$

$$\sigma^2 T_G = \frac{1}{M-1}\sum_{i=1}^{M}(\sigma^2 T_i - \mu_G)^2 \quad (8)$$

onde M é a quantidade de vetores do grupo.

Os valores resultantes da Equação 7 são normalizados pela média e variância do grupo (Equações 5 e 6). Temos agora novos dados, normalizados pela média e variância do grupo, Equação 9:

$$X_{NOVO} = \frac{X - \mu_G}{\sigma T_G} \quad (9)$$

Estes novos dados, que representam cada amostra, tem média zero e variância unitária e serão utilizados somente para a definição das amostras a serem eliminadas e quais serão as melhores amostras a serem utilizadas como sinais de referência.

O próximo passo é selecionar a melhores amostras para serem utilizadas como referência, bem como descartar as amostras que forem consideradas como discrepantes (*outliers* - discrepantes). Para isto os dados normalizados na Equação 9 são ordenados crescentemente e agrupados em faixas para a geração de um histograma para que possamos observar a distribuição das amostras. O range de frequências a serem utilizadas em cada faixa do histograma R_H é calculado pela a Equação 10:

$$R_H = \frac{\max(A) - \min(A)}{\sqrt{S}} \quad (10)$$

onde max(A) é o valor máximo do vetor normalizado, min(A) é o valor mínimo do vetor normalizado e S é quantidade de valores do vetor normalizado. Os dados são agrupados na Tabela 1, gerando a distribuição de frequência acumulada, o range e quantidade acumulada para cada range.

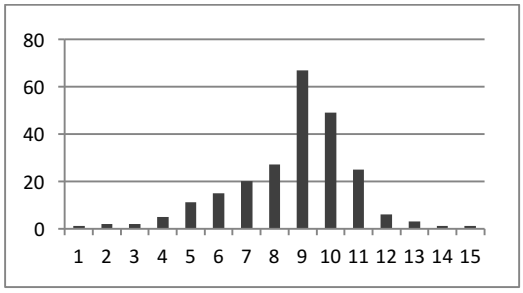
Tabela 1 - Exemplo de Distribuição dos Dados

i	Range		Quant. fi	Ponto Médio	Freq. Acum. F	(xi.fi)	(xi2.fi)
1	-4,3	-3,8	1	0,25	1	0,250	0,063
2	-3,8	-3,3	2	0,25	1	0,250	0,063
3	-3,3	-2,8	2	0,25	1	0,250	0,063
4	-2,8	-2,3	5	0,25	6	1,500	0,375
5	-2,3	-1,8	11	0,25	17	4,250	1,063
6	-1,8	-1,3	15	0,25	18	4,500	1,125
7	-1,3	-0,8	20	0,25	25	6,250	1,563
8	-0,8	-0,3	27	0,25	52	13,000	3,250
9	-0,3	0,2	67	0,25	119	29,750	7,438
10	0,2	0,7	49	0,25	168	42,000	10,500
11	0,7	1,2	25	0,25	193	48,250	12,063
12	1,2	1,7	6	0,25	199	49,750	12,438
13	1,7	2,2	3	0,25	202	50,500	12,625
14	2,2	2,7	1	0,25	203	50,750	12,688
15	2,68	3,18	1	0,25	204	51,000	12,750

Fonte: Acervo Pessoal

Os dados são agrupados em um histograma gerando-se um resultado semelhante ao mostrado na Figura 14.

Figura 14 - Histograma dos Sinais Normalizados



Fonte: Acervo Pessoal

Segundo (Correa, 2003) os pontos de primeiro quartil (Q1) e terceiro quartil (Q3) definem o limiar para os dados que podem ser classificados como outliers. Q1 e Q3 são calculados através das Equações 11 e 12:

$$Q_1 = L_{inf} + \left[\frac{\sum_{x=1}^{M} f_x}{4} - f_{ant}\right] * \frac{A}{f_i} \quad (11)$$

$$Q_3 = L_{inf} + \left[\frac{3*\sum_{x=1}^{M} f_x}{4} - f_{ant}\right] * \frac{A}{f_i} \quad (12)$$

onde M é a quantidade de faixas do histograma, A é amplitude do range, L_{inf} é o limite inferior e f_i é a frequência inferior. Na Tabela 1 o primeiro quartil é mostrado nas linhas com tom de cinza mais claro e o terceiro quartil é mostrado nas linhas com tom de cinza mais escuro. O valor limite inferior LI é calculado pela Equação 13:

$$LI = Q_1 - 1.5 * (Q_3 - Q_1) \quad (13)$$

O valor limite superior LS é calculado pela Equação 14:

$$LS = 3 + 1.5 * (Q_3 - Q_1) \quad (14)$$

Todos os dados abaixo de LI e acima de LS poderão ser descartados sem comprometimento do universo de amostras. As melhores amostras estão em torno da mediana, definida como o segundo quartil (Q2), que é calculada através da Equação 15:

$$Q_2 = L_{inf} + \left[\frac{2*\sum_{x=1}^{M} f_x}{4} - f_{ant}\right] * \frac{A}{f_i} \quad (15)$$

(Faceli et al., 2011) mostrou que a quantidade de dados a serem utilizadas como referência não necessita ser de um tamanho grande. Com o aumento da quantidade de dados é possível ocorrer a introdução de ruído no treinamento. Para este trabalho foram

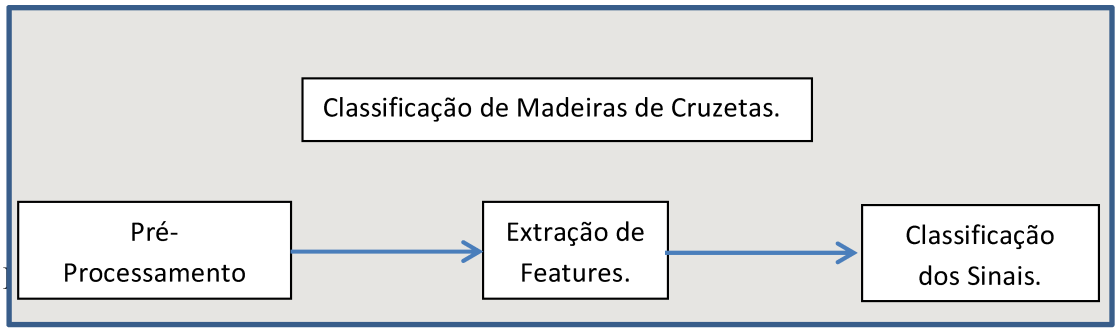
feitos testes com 9, 16 e 21 amostras de treinamento. A partir de 16 amostras os resultados permaneceram constantes ou pioraram. Desta forma a quantidade de 16 amostras para treinamento ficou definida neste artigo.

3.3 - Processamento

O processo de classificação de cruzetas de madeiras obedece a uma sequência encadeada de três etapas. As três etapas são, conforme mostra a Figura 15:

- pré-processamento dos sinais captados
- extração de "*features*"
- classificação dos sinais

Figura 15 - Esquema de classificação de madeiras de cruzetas



Fonte: Acervo Pessoal

3.3.1 - Pré-Processamento

Nesta etapa, o objetivo é aplicar (induzir) um sinal na madeira e receber o seu retorno de forma limpa e normalizada, isto é, um sinal é injetado na madeira por meio de um martelo eletromecânico, microfones espalhados pela superfície da madeira captam este sinal injetado que sofreu alterações ao percorrer pelo interior da madeira.

Uma vez captado este sinal, o mesmo é submetido a uma série de processos com o objetivo de aumentar a representatividade deles em relação ao sistema como um todo. Esse aumento da representatividade é importante e necessário porque permite separar o ruído da informação propriamente dita.

Na etapa de pré-processamento é necessário extrair características que façam com que o sinal captado identifique corretamente a madeira, com nível suficiente de amplitude e

quantidade de amostras sempre constante e com valor em termos de 2^n amostras, garantindo desta forma o cálculo correto da FFT.

Os processos utilizados foram:

a) Eliminação de Nível DC
b) Detecção de Limiar
c) Normalização

Eliminação de Nível DC

Em um arquivo de áudio com armazenamento de 16 bits o valor mínimo e máximo de um sinal varia de -32765 a +32765. O ponto central é definido como zero. No processo de digitalização o ponto central pode ser deslocado do zero em um nível acima ou abaixo de zero devido a erros na placa de aquisição ou ruídos e, com isto, causar erros no processo de treinamento e reconhecimento. Faz-se necessário a eliminação/correção desde deslocamento. Isto pode ser feito no processo de captura do sinal, através de correção na placa de captura do sinal ou após a captura, utilizando filtros passa-alta para eliminar o nível DC.

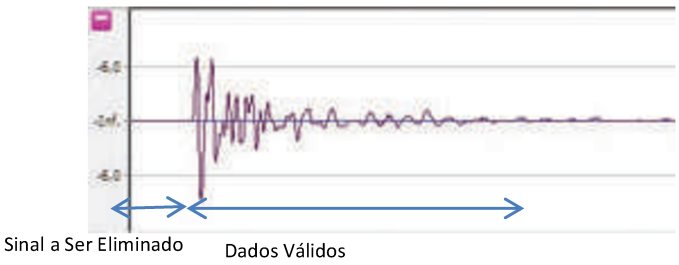
(Machado & Lima, 2007) mostra as características do nível DC e como extrair usando o programa Sound Forge da Empresa Sony.

(El-naggar, 2011) mostra a definição de filtros dinâmicos para a remoção de nível DC, em formas de onda de corrente e voltagem.

Detecção de Limiar

Nesta etapa será detectado o ínicio correto do sinal amostrado, pois o sinal amostrado possui no seu início silêncio ou ruído que deverá ser eliminado ou marcado para que o início do sinal de áudio seja sempre no ponto correto e, desta forma, não comprometer o processo de treinamento dos algoritmos que serão utilizados. A Figura 16 mostra o sinal obtido da batida de um martelo na cruzeta.

Figura 16 - Forma de Onde de um Sinal Amostrado



Fonte: Acervo Pessoal

O sinal a ser eliminado é calculado através da detecção de um nível mínimo (limiar) de sinal que seja suficiente para não incluir o ruído ou sinal em silêncio no início do sinal amostrado.

O sinal amostrado é percorrido testando se o ponto em teste está acima do limiar de detecção. Caso esteja este ponto é armazenado e o processo de busca do limiar é finalizado.

Em um arquivo de áudio com armazenamento de 16 bits o valor mínimo e máximo de um sinal varia de -32765 a +32765. Estes valores equivalem ao ponto de mínimo e máximo do sinal amostrado pela placa de captura de áudio. Níveis de ruído e sinal em silencio estão em torno de valores de amostras entre -500 a 500 no range de detecção da placa de som para uma amostragem de 16 bits.

Dados válidos será a quantidade de amostras que o sistema irá armazenar para serem tratadas durante o processo de treinamento e reconhecimento.

A utilização da técnica de Janela de Energia também pode ser utilizada para a detecção do ponto inicial de sinal válido. Neste caso aplicam-se pequenas janelas de dados, em torno de 128 amostras, e calcula-se a energia do sinal dentro desta janela.

O cálculo da energia da janela é dado pela seguinte equação, como mostra (Shanmugapriya, 2010) e (Gerhard, 2003).

$$E = \left(\frac{1}{N}\sum_{0}^{N-1} x(i)\right), i = 0..(N-1) \quad (16)$$

onde *N* é quantidade de amostras da janela e *x(i)* é o nível do sinal na posição i.

Em pontos de silêncio a energia do sinal é pequena e em pontos de sinal válido o nível da energia é alto.

Desta forma percorre-se o início do arquivo de sinal procurando-se uma mudança brusca no valor da energia para a janela de varredura. A janela na qual o valor da janela ocasionar esta mudança busca será o início do sinal válido.

Para simplificar o processamento são armazenados os pontos de início do sinal válido e a quantidade de amostras para os dados válidos, não sendo necessária a mudança/destruição do arquivo de dados original.

Normalização

Neste processo faz-se com que o sinal amostrado e válido possua uma amplitude sempre constante sem causar mudanças no envelope do sinal, não alterando as características de tonalidade do sinal. Desta forma sinais de menor intensidade não irão causar erros de cálculos em comparação com sinais de áudio com maior intensidade.

(Faceli et al., 2011) mostra que a normalização pode ser por reescala ou por padrozinação.

a) Normalização por Reescala

Neste método define-se uma nova escala de valores, limites mínimos e máximos, para todos os pontos do sinal amostrado. Este modelo de normalização também é chamado de *min-max*, de forma que o sinal original terá uma nova escala mínima e máxima.

Para cada ponto do sinal amostrado é aplicada a equação seguinte:

$$v_{novo} = \min + \frac{v_{atual-menor}}{maior-menor}(\max - \min) \quad (17)$$

onde *min* é o valor mínimo da nova escala, *max* é o valor máximo da nova escala e v_{atual} é a valor da posição atual do sinal de dados.

Como exemplo se quisermos normalizar um sinal para o range entre 0 e 1, definimos na equação anterior o valor de *min* para 0 e *max* para 1.

b) Normalização por Padronização

Neste método são utilizados cálculos estatísticos do sinal para o cálculo da normalização. É necessário calcular a média e a variância de todo o sinal e aplicar na Equação 17.

$$v_{novo} = \frac{v_{atual} - \mu}{\sigma} \quad (17)$$

onde μ é a média e σ é a variância do sinal.

Com isto diferentes sinais de dados podem ter limites inferiores e superiores diferentes e, visualmente não parecerem estar normalizados na amplitude, mas terão após a normalização um novo conjunto de valores com média 0 e variância 1.

3.3.2 - Extração de *Features*

Features, ou características, são informações que serão extraídas do sinal amostrado, que representem suas características e de forma que, posteriormente, o sinal original possa ser reconstruído ou recuperadas todas as suas informações.

Existem diversos processos de extração de *features*. Neste trabalho utilizaremos como *features* os pontos da Transformada de Fourier (FFT), os coeficientes LPC, os coeficientes MFCC, a taxa de cruzamento por zero (ZC), a energia total do sinal (Energy) e a área (Area) do sinal amostrado.

FFT, LPC, MFCC, ZC, Energy e Area geram coeficientes numéricos, extraídos de cada uma das características, por exemplo os pontos de uma FFT, conforme mostrado abaixo, onde cada coeficiente é definido como uma *feature*.

A FFT, por exemplo, gera 1024 pontos, portanto teremos 1024 *features* para a FFT. Se forem gerados 10 coeficientes LPC, então teremos 10 *features* para o LPC. Se forem gerados 12 coeficientes MFCC, então teremos 12 *features* para o MFCC. Se forem gerados 4 janelas para o cálculo de ZC então teremos 4 *features* para o ZC. Se forem geradas 4 janelas para o cálculo da Energy então termos 4 *features* para a Energy. A Area retorna uma valor único, então teremos uma *feature* para a Area.

O resultado final será um vetor com a adição de todas as *features* individualmente. O vetor resultante será a adição de 1024 *features* FFT, 10 *features* LPC, 12 *features* MFCC, 4 *features* ZC, 4 *features* Energy e 1 *feature* Area, gerando no total um Vetor com 1055 posições, onde as 1024 primeiras posições representam as informações FFT, as 10 seguintes as informações LPC, as próximas 12 representam as informações MFCC, as 4 seguintes representam as informações ZC, as 4 seguintes representam as informações Energy e a última representa a informação Area. Portanto o que irá

diferenciar uma *feature* da outra será a sua característica em função da posição que a mesma ocupa no vetor. Caso o vetor seja somente de uma característica, todas as *features* serão do mesmo tipo.

A Figura 17 representa como ficará o vetor de *features*.

Figura 17 - Vetor representante de todas as *Features*

FFT	LPC	MFCC	ZC	Energy	Area

Fonte: Acervo Pessoal

Características isoladas também podem ser utilizadas, tal como utilizar somente os coeficientes LPC, Desta forma o vetor resultante de *features* teria somente 10 posições.

A seguir será definida individualmente cada característica acima citadas.

FFT

Um sinal de áudio pode ser considerado como uma sequência de intervalos estacionários dentro dos quais a distribuição espectral de potência é mais ou menos constante, adotando um comportamento não-estacionário. Desta forma a análise espectral de Fourier pode ser utilizada como uma parametrização do sinal de áudio.

Os parâmetros da análise de Fourier são dados pelos coeficientes da Transformada Discreta de Fourier (DFT – Discret Fourier Transforma), representados pela equação seguinte:

$$X(k) = \sum_{n=1}^{N} x(n) e^{\frac{-j2\pi(k-1)(n-1)}{N}} \quad (18)$$

onde $1 \leq k \leq N$ e x(n) é cada ponto do sinal de áudio digitalizado com um tamanho N.

Para este tipo de parametrização a resolução em frequência é dada por $\Delta f = 1/NT$, onde T é o período de amostragem.

Desta forma podemos utilizar, no domínio digital, o algorítimo da Transformada Rápida de Fourier (FFT – Fast Fourier Transform), possuindo o mesmo resultado e ganho na diminuição da complexidade computacional. (Stojanovi, 2006) mostra basicamente como desenvolver o algoritmo para o cálculo da FFT e o seu grau de complexidade.

LPC

A predição linear LPC é um dos métodos dominantes para estimar os parâmetros do sinal de voz. A idéia básica da predição linear é a de que o valor de uma amostra do sinal de áudio pode ser aproximado por combinação linear dos valores das amostras anteriores, com base na correlação entre as amostras anteriores.

(Muniz, 2009) mostra que o sinal referente a fala humana é produzido através de um ruído branco no final de um tubo, onde neste ocasionalmente são adicionados sons de assovio ou estouro. Embora aparentemente grosseiro este modelo é uma boa aproximação da produção da voz humana. A glote (o espaço entre as pregas vocais) produz o ruido inicial, que é caracterizado pela sua intensidade e frequência. O trato vocal (boca e garganta) compõe o tubo, que é caracterizada pela sua ressonância, dando origem a frequência fundamental do som produzido. Silvos e estouros são gerados pela ação da língua, lábios e garganta durante a produção da fala.

Os coeficientes LPC retirados dessa análise estimam as características do sinal falado a partir de um ruído inicial ou de uma sequência periódica de impulsos. Tais coeficientes servem para modelar um filtro que se aproxima do trato vocal. Estes coeficientes são estimados ao minimizar o erro quadrático entre a amostra analisada atualmente e sua predição.

O sinal estudado neste trabalho assemelha-se ao sinal de áudio produzido pela fala do ser humano (uma batida de martelo pode ser totalmente reproduzida pelo ser humano, estando na mesma região de frequências, conforme será mostrado no decorrer deste trabalho), possuindo uma característica tonal bem definida (Bradbury, 2000), pois este sinal capturado foi da batida de um martelo eletrônico/manual na madeira da cruzeta.

O modelo particular utilizado em LPC é chamado de codificação preditiva linear, possuindo dois componentes: Análise ou Codificação e síntese ou Decodificação. A parte de análise de LPC envolve o exame do sinal e divisão do mesmo em segmentos ou blocos (Bradbury, 2000).

Em (Cinnéide, 2008) é mostrado que os coeficientes LPC a_i $\{a\{a_i\}_{i=1}^{M}$ são calculados com certa facilidade através das funções de autocorrelação:

$$r_{x_m(l)} \triangleq \sum_{n=-\infty}^{\infty} x_m(n)x_m(n+l) = DFT^{-1}|X_m|^2 \quad (19)$$

Para obter os M coeficientes $\{a_1,...,a_M\}$ é necessário resolver a matriz M x M de equações lineares:

$$\sum_{i=1}^{M} a_i r_{x_m(|i-j|)} = -r_{x_m(j)}, j = 1,2,...,M \quad (20)$$

MFCC

Em processamento digital de sinais, MFC (Mel-Frequency Cepstrum) é uma representação do espectro de energia de curto prazo de um sinal amostrado, baseado na Transformada Linear do Cosseno do espectro de potência logarítmico numa escala de frequência Mel não-linear.

MFCCs (Mel-Frequency Cepstral Coeficientes) são os coeficientes que formam um MFC. Eles são derivados a partir de um tipo de representação cepstral do sinal de áudio. A diferença entre o cepstrum e o MFC é que as bandas de frequência são igualmente espaçados na escala mel, o qual se aproxima a resposta do sistema auditivo humano de forma mais estreita do que as bandas de frequência linearmente espaçados utilizados na cepstrum normal. Esta deformação frequência pode permitir uma melhor representação do som, por exemplo, na compressão de áudio.

(Ganchev et al., 2005) mostra como computar os coeficientes MFCC através da seguinte equação:

$$C_j = \sum_{i=1}^{M} X_i \cos\left(j \left(i - \frac{1}{2}\right)\frac{\pi}{M}\right) \quad (20)$$

onde M é o número de filtros no banco de filtros, J é o números de coeficientes MFCCs que deverão ser computados (normalmente $J < M$), e X_i definido como "a saída logarítmica de energia do *i-ésimo* filtro" mostrado em (Davis S. and Mermelstein, 1980) através da seguintes equação:

$$X_i = log_{10}\left(\sum_{k=0}^{N-1} |X(k)| H_i(k)\right), i = 1, 2, \ldots, M \quad (21)$$

onde *X(k)* é dado por

$$X(k) = \sum_{n=0}^{N-1} x(n) \exp\left(\frac{-j2\pi nk}{N}\right), k = 0, 1, \ldots, N-1 \quad (22)$$

onde *x(n)* é o sinal amostrado e $H_i(k)$ é dado por:

$$H_i(k) = \begin{cases} 0 & para\ k < f_{b_{i-1}} \\ \frac{(k - f_{b_{i-1}})}{(f_{b_i} - f_{b_{i-1}})} & para\ f_{b_{i-1}} \leq k \leq f_{b_i} \\ \frac{(f_{b_{i+1}} - k)}{(f_{b_{i+1}} - f_{b_i})} & para\ f_{b_i} \leq k \leq f_{b_{i+1}} \\ 0 & para\ k > f_{b_{i+1}} \end{cases} \quad , \text{i} = 1, 2, ..., \text{M} \quad (22)$$

onde *i* refere-se ao *i-ésimo* filtro, f_{b_i} são os pontos limites dos filtros e *k = 1, 2, ..., N* correspondem ao *k-ésimo* coeficientes do *N-ésimo* ponto da DFT do sinal x(n). As definições dos filtros f_{b_i} é demonstrada em (Ganchev et al., 2005).

ZC

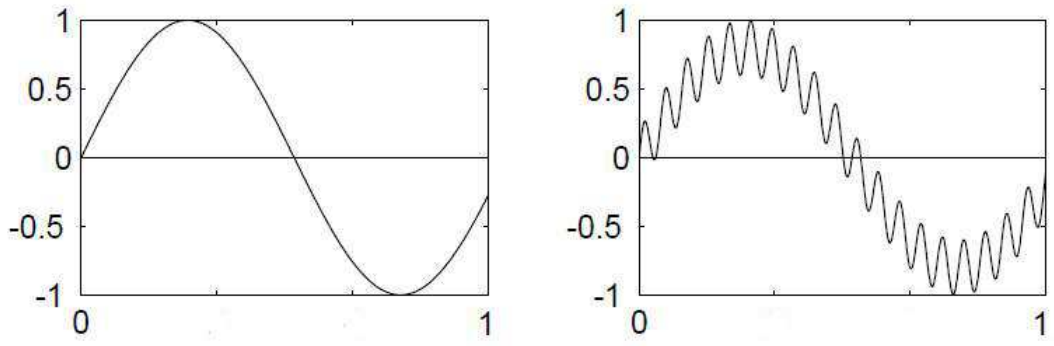
Zero Crossing (ZC) é definido como a taxa de cruzamento ponto de limiar do sinal, ou ponto zero de cruzamento.

ZC é uma medida de quantas vezes o sinal cruza por zero por unidade de tempo. A ideia é que a ZCR dá informações sobre o conteúdo espectral do sinal.

(Gerhard, 2003) e (Kedem, 1986) mostram que a ideia do ZC deve estar diretamente relacionada com o número de vezes que o sinal repetido por unidade de tempo, que é a frequência fundamental f_0, passa pelo ponto de intersecção com zero. A medida da frequência fundamental tornou-se difícil se o sinal fosse espectralmente ruidoso.

Uma senóide pura irá atravessar a linha zero duas vezes por ciclo, como mostrado na Figura 18 (a).

Figura 18 - Influência de harmônicas na taxa de cruzamento por zero



(a) (b)

Fonte: Acervo Pessoal

Mas se é espectralmente rica em harmônicas, como na Figura 17 (b), irá atravessar o ponto zero em mais vezes por ciclo.

Desta forma um sinal ruidoso irá ter uma ZC muito maior do que um sinal sem ruídos. E um sinal com muitos harmônicos altos também terá uma ZC maior do que um sinal com poucos harmônicos.

ENERGY

A energia é utilizada para descobrir o silêncio num sinal. A energia de um sinal é tipicamente calculada numa base de curto tempo, o sinal de uma janela em um determinado momento, em quadratura com as amostras e considerando a média como já mostrado na (eq. 01), também mostrado por (Gerhard, 2003).

AREA

Área é definida como sendo o cálculo da região sob a curva do sinal de áudio.

Como um sinal de áudio é periódico, variando de -32755 a 32755, como já mostrado na figura 5, e se este sinal for uma senóide pura, o resultado do cálculo da área será zero, não possuindo nenhuma representatividade em nosso caso. Desta forma este cálculo é feito através do rebatimento do sinal negativo no eixo positivo, através cálculo da sua norma, ou seja, |x(n)|. Desta forma o sinal será rebatido e teremos um sinal com representatividade.

3.3.3 - Classificação dos Sinais

Para esta etapa do projeto foram implementados os algoritmos para a fase de separação dos dados nas sanidades de busca: 1 Ano, Urgencial e Emergencial.

É feito também um processo de validação do programa de reconhecimento das sanidades e dos microfones de captura de áudio, através de testes básicos e testes de batidas em cruzetas em laboratório.

Posteriormente o programa foi inserido em uma micro-motherboard dedicada para redução de dimensões.

Será utilizada a Distâncias Euclidiana para o teste de distância entre os pontos de referência e os pontos de testes, dada como mostra a Equação 09:

$$D = \sqrt{\sum_{i=1}^{N}(y_i - x_i)^2} \quad (23)$$

onde y_i são os pontos de referência, x_i são os pontos de testes e N é a quantidade de pontos do vetor de *features*.

O algoritmo utilizado para o reconhecimento das amostras foi o K-NN, descrito a seguir.

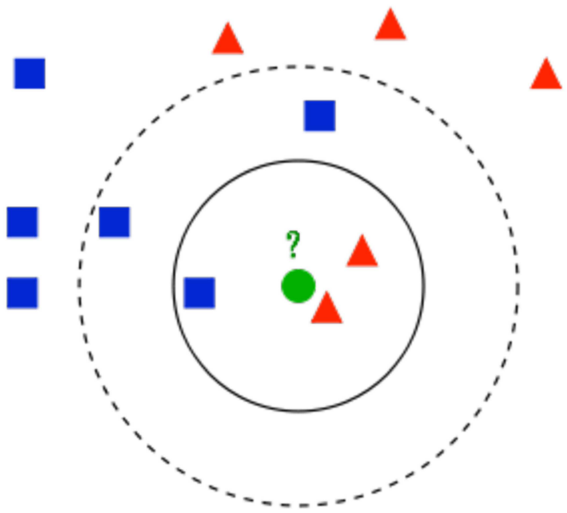
3.3.3.1 - K-NN

K-NN significa *K*-Nearest Neighbor Algorithm (Algorítmo do K Vizinho Mais Próximo). Em reconhecimento de padrões, o algoritmo K-NN é um método para classificar objetos com base em exemplos mais próximos de uma referência conhecida.

O algoritmo K-NN está entre o mais simples de todos os algoritmos de aprendizagem de máquina: um objeto é classificado por maioria de votos de seus vizinhos com o objeto que está sendo atribuído à classe mais comum entre os seus k vizinhos mais próximos (K é um inteiro positivo, geralmente pequena). Se K = 1, então o objeto é simplesmente atribuído à classe do referido único vizinho mais próximo (Faceli et al., 2011)

A Figura 19 em conjunto com a descrição a seguir mostra o funcionamento deste algoritmo.

Figura 19 - Algoritmo K-NN



Fonte: (Faceli et al., 2011)

Neste exemplo temos dois tipos de dados, quadrados e triângulos. O círculo central (com uma interrogação em cima) é o dado a testado, ou seja, para qual tipo de quadrados ou triângulo ele pertence. O algoritmo baseia-se na procura de K dados mais próximos. Para K=1 tem-se somente um triângulo mais perto do dado a ser testado. Para K=2 tem-se dois triângulos mais próximos. Para K=3 tem-se dois triângulos e um quadrado. Neste caso define-se a maior quantidade de mesmos objetos próximos, que fica com os triângulos. Para K=5 observa-se que tem dois triângulos e três quadrados. Neste caso define-se a maior quantidade de mesmos objetos próximos, que fica com os quadrados.

Para o caso de sanidades os dados de testes ficam armazenados para posterior processamento. Cada dado de teste fica associado a uma sanidade.

Quanto um sinal a ser testado é capturado, pesquisa-se a qual dado ele está mais próximo (através do uso simples de Distância Euclidiano ou Distância de Mahalanobis entre a *feature* armazenada para a comparação e a *feature* do sinal amostrado no momento), sendo então a sanidade associada a este dado a sanidade do sinal a ser testado, isto para o caso de K=1.

3.4 – Algoritmo de Análise e Reconhecimento

De posse de todas estas informações, para esta etapa será definido o algoritmo final para o treinamento e reconhecimento das sanidades Emergencial, Urgente e 1 Ano.

Este processo é divido em duas etapas:

- **Treinamento:** Nesta etapa serão separadas amostras de cruzetas nos três (03) níveis de sanidades de pesquisa, por pessoal especializado da CEMIG D, onde estas amostras serão rotuladas e o sistema será treinado para o reconhecimento destas amostras.
- **Reconhecimento:** Nesta etapa serão também separadas amostras de cruzetas nos três (03) níveis de sanidades de pesquisa, por pessoal especializado da CEMIG D, onde estas amostras serão rotuladas e o sistema irá fazer o reconhecimento das mesmas.

Todas as amostras serão armazenadas em arquivos no formato wave (.WAV).

O processo de separação de cruzetas nos estados de sanidades de pesquisa é extremamente complexo. Isto se deve ao fato de que os inspetores que separam as cruzetas muitas vezes separam de forma errônea ou somente por inspeção visual, fato este explanado pelos próprios inspetores.

O sistema precisa ser alimentado com informações extremamente corretas para a obtenção de resultados mais exatos. Desta forma é criado um protocolo para a obtenção destas informações juntos com os inspetores especializados da CEMIG D.

Depois de separadas as informações de sinais provenientes dos três tipos de sanidades, estes dados serão inseridos no sistema, serão pré-processados, suas *features* serão extraídas e então estes dados serão classificados segundo o algoritmo K-NN.

Os resultados da classificação serão armazenados para serem utilizados no processo de reconhecimento, quando necessário.

3.4.1 - Metodologia para Aquisição de Dados

Conforme resultados obtidos em laboratório com total controle e conhecimento da sanidade das cruzetas analisadas, sem levar em consideração seus aspectos externos (conforme será mostrado no capítulo referente a resultados), confrontados com as análises obtidas na CEMIG D pelos especialistas, notou-se que o fato do analista estar visualizando a forma física da cruzeta antes de analisar o sinal evocado pelo martelo (áudio da batida do martelo), levou os especialistas a concluírem erroneamente o grau de sanidade de muitos pontos amostrados, ou seja, submetendo o analista a escutar somente o som da batida do martelo, sem visualizar a forma física da cruzeta, notou-se que o resultado não coincidia com o resultado inicialmente realizado.

Assim, um protocolo foi criado seguindo normas rígidas utilizadas pelos pesquisadores na área da medicina na coleta de dados, com duplo, triplo cego.

Realizada uma captura de sinal confiável, descartadas as análises conflitantes e dúbias, é necessário que sejam estatisticamente validadas todas as amostras de dados, bem como deve ser validado os sensores que colheram os dados, os profissionais humanos que fizeram a classificação.

3.4.2 – Protocolo de Aquisição de Dados

O protocolo baseia no fato de que o especialista humano, quando for classificar um sinal sonoro evocado em uma cruzeta pela percussão de um martelo manual ou elétrico, o mesmo não poderá ver a cruzeta ou saber previamente a classificação da mesma.

Para tanto, será necessário uma análise de campo e outra em laboratório para que se possa montar uma base de dados confiáveis que será utilizada como treinamento do protótipo de análise de sanidade de cruzeta para futuras classificações.

Três especialistas serão necessários (quantidade mínima) para que se garantam os índices estatísticos que traduzam em confiabilidade nos resultados a serem obtidos.

3.4.2.1 - Definições Iniciais para a Geração do Protocolo

P.1 – A CEMIG D deverá selecionar três (03) especialistas em reconhecimento de sanidade de cruzetas e disponibilizá-los para o processo de aquisição de dados e análises iniciais.

P.2 – Cada especialista receberá um código que o identificará em todo o processo de análise.

P.3 – Cada um destes três (03) especialistas deverá selecionar diversas cruzetas que os mesmos entendam que obedeçam as sanidades de 1 Ano, Urgencial (30 dias) e Emergencial (troca imediata) para serem analisadas.

Poderão ser separadas também partes de cruzetas que obedeçam as sanidades desejadas.

As cruzetas serão analisadas através do uso de seções/segmentos transversais das mesmas, onde cada seção/segmento poderá estar em uma sanidade específica, visto que em uma cruzeta todas as sanidades poderão estar presentes em sua extensão. A Figura 20 mostra a divisão de uma cruzeta em segmentos.

Figura 20 - Seção Transversal da Cruzeta

Fonte: Acervo Pessoal

Uma seção transversal/segmento não poderá estar sobre a furação padrão da cruzeta conforme mostra a Figura 21.

Figura 21 - Figura dos Locais de Furos da Cruzeta

Fonte: Acervo Pessoal

Estas cruzetas já deverão estar separadas previamente pelos especialistas antes do processo de aquisição dos dados.

As cruzetas deverão ficar armazenadas em local onde não sofram as ações do tempo (sol, chuva, ventos, etc).

P.4 – A marcação de cada cruzeta deverá obedecer ao seguinte critério, conforme mostra a Tabela 2:

- Número da cruzeta
- Possível sanidade que ocasionou a troca da cruzeta, obedecendo ao seguinte critério: **1A** – 1 Ano, **UR** – Urgencial, **EM** – Emergencial

Tabela 2 - Modelo de Etiqueta de Marcação de Cruzeta

CRUZETA No.	
SANIDADE	

Fonte: Acervo Pessoal

P.5 - Segmentar a cruzeta em diversas partes, evitando furos originais da cruzeta. Nesta etapa serão marcados pontos na cruzeta, pontos estes que serão o foco da análise posterior.

P.6 - Marcar cada segmento com uma etiqueta. Esta etiqueta deverá conter as seguintes informações:

Tabela 3 - Modelo de Etiqueta de Marcação de Seção/Segmento

CRUZETA No.	
SEGMENTO No.	

Fonte: Acervo Pessoal

P.7 - Fotografar toda a cruzeta mostrando a etiqueta da mesma

P.8 - Fotografar cada segmento a ser analisado mostrando a etiqueta do ponto de análise

P.9 - Cada segmento deverá ser analisado pelos três especialistas utilizando o processo de batida manual com marreta.

P.10 – Pré-Processamento para cada segmento:

P.10.1 – Será amostrado um sinal de áudio proveniente de um microfone de contato posicionado no lado contrário a batida da marreta. A Figura 22 mostra o detalhe do microfone de contato a ser utilizado.

Figura 22 - Modelo do Posicionamento do Microfone

Fonte: Acervo Pessoal

P.10.2 - O sinal amostrado/evocado, proveniente da batida da marreta de cada um dos especialistas, deverá ser armazenado/guardado em pastas específicas da seguinte forma:

- Pasta Principal: Nome Sanidade
- Primeira Sub-Pasta: Cruzeta No. X (conforme está marcado na etiqueta da cruzeta)
- Segunda Sub-Pasta: Modo de captura (Automatico ou Manual)
- Dentro desta sub-pasta deverão ter novas pastas para identificar o especialista que fez a análise do sinal

Dentro de cada pasta referente ao especialista deverá conter três (03) arquivos para cada amostra capturada, referentes a cada segmento analisado da cruzeta:

  - **ax.wav** (arquivo de som contento o sinal de áudio)
  - **ax.jpg** (foto proveniente da câmera posicionada na garra de sustentação)
  - **ax.txt** (arquivo de texto contendo o valor, em amostras, referente ao ponto de início da onda do sinal para posicionamento do software de análise)
- Dentro da sub-pasta do modo automático deverão ser armazenado os sinais proveniente da batida do martelo eletrônico organizados em conformidade com o item anterior

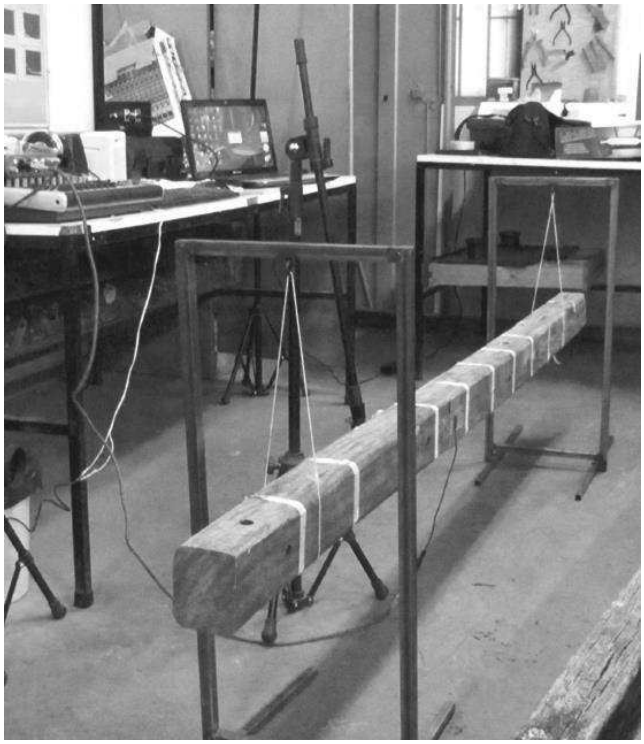
P.10.3 - Cada especialista deverá dar a sua avaliação, em termos de grau de sanidade, para cada segmento analisado (1Ano, Urgencial, Emergencial), analisando em primeira instância tanto o som quanto o aspecto visual da cruzeta. Os dados serão montados em uma planilha para cada especialista com as seguintes colunas: Cruzeta no., Segmento no., Sanidade Avaliada.

3.4.2.2 - Análise em Campo

C.1 – Cada especialista deverá fazer a análise das cruzetas e não poderá ter outro especialista neste momento da análise (somente um especialista por vez irá fazer a análise das cruzetas)

C.2 – Cada cruzeta a ser analisada deverá ser montada no **SPACSIM** (Sistema Portátil de Análise de Cruzeta sem Interferência do Meio), conforme mostra a Figura 23. A definição sobre o SPACSIM será mostrada mais a frente.

Figura 23 - Cruzeta montada no SPACSIM



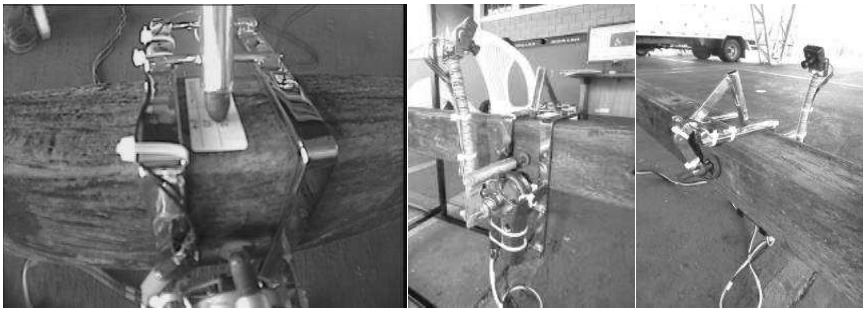
Fonte: Acervo Pessoal

C.3 – Fotografar a cruzeta no SPACSIM mostrando a etiqueta de identificação da cruzeta.

C.4 – Em cada segmento deverá ser montada a garra contendo os microfones para a captação dos sinais provenientes da batida manual da marreta do especialista,

posicionando os microfones sobre a marcação do segmento, bem como câmera de vídeo para o registro da imagem da posição da batida, conforme mostra a Figura 24.

Figura 24 - Montagem do Martelo Eletrônico



Fonte: Acervo Pessoal

C.5 – O especialista irá efetuar a batida da marreta no segmento da cruzeta. Momento este em que o sinal de áudio do microfone da garra será capturado e armazenado.

C.6 – O especialista irá definir em qual sanidade o segmento encontra-se (1A, UR, EM).

C.7 – Antes do especialista passar para o próximo segmento, será feita a coleta dos dados referentes a batida do martelo eletrônico. Se o sinal estiver satisfatório, será feito o seu armazenamento, caso contrário nova aquisição dos sinais provenientes do martelo eletrônico deverá ser feita até que esteja satisfatório (nível aceitável de amplitude e menor quantidade possível de ruído).

C.8 – Os dados coletados deverão ser armazenados na pasta correspondente a cruzeta analisada e ao especialista do momento, bem como em uma planilha para avaliação, conforme mostra a Tabela 4.

Tabela 4 - Tabela de Referência do Grau de sanidade

Especialista em Análise:		(código do especialista que está analisando a cruzeta)	
Dados do Especialista:		(código do especialista que separou as cruzetas)	
No.:	**Cruzeta no.:**	**Segmento no.:**	**Código da Sanidade Avaliada**
1			
2			
3			
4			
5			
..			
..			
100			

Fonte: Acervo Pessoal

OBS.: Os outros dois especialistas também deverão fazer a análise das mesmas cruzetas do primeiro especialista, sem a presença do mesmo.

As planilhas com os resultados de todas as análises deverão ser avaliadas da seguinte forma:

A.1 – Para cada segmento analisado por todos os especialistas determinar se houve unanimidade nos resultados.

A.2 – Caso não haja unanimidade nos resultados este segmento deverá novamente ser avaliado pelos especialistas para determinar a causa do erro na análise, utilizando-se os seguintes processos:

A.2.1 – Efetuar novamente a batida de marreta no segmento em questão.

A.2.2 – Efetuar o corte da cruzeta sobre o segmento em questão (através do uso de equipamentos adequados) perante todos os especialistas para avaliar o estado da

madeira da cruzeta naquele ponto. A Figura 25 mostra os especialistas fazendo novamente a análise de uma cruzeta.

Figura 25 - Reavaliação de uma Cruzeta

Fonte: Acervo Pessoal

A.3 – Com os critérios descritos determinar em qual sanidade correta encontra-se o segmento de cruzeta em questão. Havendo divergência, descartar o segmento amostrado bem como todos os seus dados armazenados.

A.4 – Efetuar os processos dos itens **A.2** e **A.3** para todos os segmentos que gerem dúvidas no processo de análise dos especialistas.

3.4.2.3 - Análise em Laboratório

Em data e horário a serem previamente definidos para cada especialista deverá comparecer ao Laboratório de Testes do Projeto, um de cada vez, onde serão acompanhados efetivamente por um analista que observará o julgamento do mesmo em cada amostra de tal forma que se possa validar ou não cada análise feita pelo(s) mesmo(s).

Cada especialista irá reavaliar os sinais amostrados da seguinte forma:

L.1 – Os sinais de todos os microfones capturados para cada segmento será analisado somente através do áudio deste sinal (sem visualizar a imagem do segmento). O especialista deverá definir a qual sanidade pertence o segmento.

L.2 – Esta nova avaliação deverá ser confrontada com a avaliação feita no item **ANÁLISE EM CAMPO** e, havendo divergência, o especialista deverá ser questionado para que responda a causa deste erro.

L.3 – Caso o especialista não consiga definir a qual grau de sanidade pertente o segmento em análise, os dados deste segmento deverão ser descartados.

L.4 – O especialista também deverá analisar os sinais capturados pelos outros especialistas para confronto na especificação da sanidade do segmento.

L.5 – Somente sinais que possuam a mesma sanidade tanto na análise inicial feita em campo e na etapa do item **L.1** por todos os especialistas deverão ser armazenados para treinamento do sistema.

3.4.3 - Descrição do Algoritmo de Treinamento/Reconhecimento

O algoritmo de análise e reconhecimento é dividido em duas (02) etapas:

- Treinamento
- Reconhecimento

No treinamento um grupo de amostras pré-definidas através do protocolo de aquisição de dados é inserido no sistema e resultados para comparação são gerados. Com a utilização do método K-NN é armazenada uma matriz contendo todas as *features* de teste, onde cada linha é referente a uma amostra, ordenados de forma que as primeiras N amostras referem-se ao estado de sanidade 1 Ano, as próximas M amostras referem-se ao estado de sanidade Urgencial e as próximas P amostras referem-se ao estado de sanidade Emergencial. A Figura 26 mostra esta organização para o algoritmo K-NN.

Figura 26 - Organização da Matriz de Features

***Features* Amostra 01**	**N AMOSTRAS**
***Features* Amostra 02**	
***Features* Amostra 03**	
***Features* Amostra 04**	**M AMOSTRAS**
***Features* Amostra 05**	
...	
...	**P AMOSTRAS**
***Features* Amostra (N + M + P)**	

Fonte: Acervo Pessoal

O número de posições do vetor de *features*, caso seja muito grande, poderá ser reduzido através do uso da técnica de PCA (Principal Componentes Analysis) conforme descreve (Khan & Farooq, 2012) e (Baker, 2013). Vale salientar que este processo de redução de *features* através do uso do PCA pode causar a mudança das características desejadas, pois são retornadas as *features* com maior energia e representatividade dentro da matriz de *features*, como mostra (Faceli et al., 2011).

Na subseção 3.3.3.1 é feita a descrição dos dois algoritmos (Treinamento e Reconhecimento).

3.4.3.1 - Treinamento

1 - Início

2 - Definir variáveis do ambiente

2.1 - Quant. amostras a serem processadas: N

2.2 - Taxa de amostragem: TX

2.3 - Quantidade de aproximações para K-NN: KNN

3 - Capturar sinal

4 - Obter ponto de início do sinal

5 - Extrair a quantidade de amostras do sinal

6 - Eliminar nível DC e normalizar o sinal

7 - Adicionar amostra normalizada a matriz de dados

8 - Extrair *features* para cada sinal amostrado

8.1 - para cada sinal extrair a *feature* desejada

8.2 - adicionar estas *features* ao vetor de *features* para a amostra

8.3 - voltar ao passo 8.1 até que todas as *features* sejam extraídas

8.4 - adicionar este vetor a matriz de todas as *features* onde cada linha representa uma amostra

9 - Voltar ao item 3 até que todos os sinais estejam capturados

10 - Caso necessário utilizar PCA para redução de *features*

11 - Armazenar a matriz de *features*

12 – Ao utilizar K-NN

12.1 - Utilizar o número de aproximações previamente definidas

12.2 - Definir/associar a cada linha da matriz de *features* o seu estado de sanidade ordenados em ordem crescente para 1 Ano, Urgente e Emergencial

12.3 - Armazenar esta matriz com as definições/associação de sanidades para que seja as *features* de referência para o reconhecimento no algoritmo K-NN

13 - Fim

3.4.3.2 - Reconhecimento

1 - Início

2 - Definir variáveis do ambiente

2.1 - Quant. amostras a serem processadas: N

2.2 – Taxa de amostragem: TX

2.3 - Quantidade de aproximações para K-NN: KNN

2.4 - Ler a matriz de *features*

2.5 - Ler os centroides previamente calculados

3 - Capturar sinal

4 - Obter ponto de início do sinal

5 - Extrair a quantidade de amostras do sinal

6 - Eliminar nível DC e normalizar o sinal

7 - Adicionar a amostra normalizada a matriz de dados

8 - Extrair *features* para cada sinal amostrado

8.1 - Extrair a *feature* desejada do sinal

8.2 - Se for utilizado PCA no processo de treinamento, utilizar também no processo de reconhecimento com os mesmos parâmetros utilizados no processo de treinamento

8.3 - Adicionar estas *features* ao vetor de *features* para a amostra

8.4 - Voltar ao passo 8.1 até que todas as *features* sejam extraídas

9 – Para K-NN

9.1 - Utilizar o número de aproximações previamente definidas

9.2 - Calcular a distância euclidiana da amostra obtida em relação a cada linha da matriz de *features*, obtida no item 2.5

9.2 – A menor distância euclidiana com relação a linha da matriz de *features*, resultante do teste anterior, será a sanidade procurada (a matriz de *features* está ordenada com relação as sanidades)

10 – Fim

3.5 – Conclusões

Com a utilização de três (03) especialistas foi possível obter dados de uma forma correta, consistente, desprezando-se possíveis análises errôneas, e com isto ter amostras válidas para o funcionamento de todo o sistema. Resta somente uma análise estatística para obtenção dos melhores sinais a serem utilizados como referência para o algoritmo de treinamento, bem como a exclusão, caso seja necessário, de amostras que possam ser estatisticamente desprezadas, conforme será mostrado no próximo capítulo,.

No próximo capítulo será mostrado um estudo sobre como sinal deve ser capturado, tais como posicionamento de microfones, posicionamento da batida do martelo, formas de onda e o desenvolvimento de todo o hardware necessário para a utilização em campo.

4 – ANÁLISES DO SINAL E DESENVOLVIMENTO DO HARDWARE

Para o desenvolvimento de um hardware que faça a captura e processamento de todos os dados primeiro será necessário alguns estudos sobre como será a análise dos dados.

O sinal de áudio deverá ser analisado para ser definida a melhor forma de ser capturado. Os tópicos seguintes descrevem todos os passos necessários.

4.1 - Análises Iniciais de Capturas de Áudio

A empresa CEMIG D forneceu treze (13) cruzetas para análise, desde uma cruzeta nova (que ainda não teria sido colocada em um poste) até cruzetas quebradas que foram retiradas de postes, pois estavam causando problemas na rede elétrica.

Uma vara de manobras também nos foi fornecida para testes. Inicialmente foram capturados sinais de áudio com a cruzeta nova, colocada sobre a bancada de trabalho. Batidas de martelo, simulando o trabalho das equipes de testes da empresa, foram realizadas na cruzeta. Os equipamentos utilizados para estes testes iniciais foram: Software Sound Forge 10.0 Versão Demonstrativa para captura do sinal, Placa de som U46DJ ESI (4 entradas e 6 saídas), com entrada para microfone dinâmico, via USB 2.0, Microfone dinâmico CSR-58, Pedestal tipo girafa para fixação do microfone (captura do áudio sem interferência do operador do software), O áudio das batidas do martelo foi capturado em diversas posições na cruzeta.

Algumas conclusões puderam ser tiradas deste teste inicial:

a) O sinal de áudio varia conforme a posição de captura do microfone (microfone junto da batida, microfone na parte oposta da batida);

b) O sinal de áudio varia conforme a distância da batida do martelo com relação à posição do microfone;

c) O sinal de áudio varia conforme o local de colocação da cruzeta;

d) O sinal de áudio varia conforme a força da batida do martelo na cruzeta;

e) O sinal de áudio varia conforme os objetos fixados na cruzeta.

Esta variação do sinal inclui amplitude, potência e nível de ruído do mesmo.

4.1.1 - Posições de captura do microfone

A posição em que o microfone é colocado com relação à cruzeta e a batida do martelo geram sinais de áudio que, após análise de sua FFT pode-se concluir que o sinal é diferente um do outro (intensidade, frequências díspares e número de harmônicos) ocasionando uma análise errônea. Este tipo de posicionamento é mostrado nas Figuras 27-30, utilizando microfone comum para testes iniciais:

Figura 27 - Martelo colocado ao lado do microfone e em paralelo

Fonte: Acervo Pessoal

Figura 28 - Microfone não paralelo ao martelo

Fonte: Acervo Pessoal

Figura 29 - Martelo posicionado do lado oposto ao microfone

Fonte: Acervo Pessoal

Figura 30 - Martelo posicionado na parte superior da cruzeta e acima do microfone

Fonte: Acervo Pessoal

4.1.2 - Distâncias da batida do martelo

Com o microfone fixo em uma posição, foram realizadas diversas capturas de áudio efetuando-se batidas com o martelo em posições diferentes da cruzeta.

Os áudios capturados também foram diferentes como no caso anterior, gerando sinais diferentes, conforme a posição da batida e a distância com relação do microfone.

Este tipo de posicionamento é mostrado nas Figuras 31 e 32.

Figura 31 - Martelo posicionado perto do microfone

Fonte: Acervo Pessoal

Figura 32 - Martelo posicionado longe do microfone

Fonte: Acervo Pessoal

4.1.3 - Local de Colocação da Cruzeta

Outro fator importante na análise do sinal de áudio é onde a cruzeta está fixada para esta captura. Conforme o estado de fixação da cruzeta o sinal de áudio capturado também pode variar gerando sinais com frequências diferentes das que são observadas no dia-a-dia dos profissionais de campo.

Nos testes iniciais a cruzeta foi colocada diretamente sobre uma bancada de trabalho. No dia-a-dia dos profissionais de campo as cruzetas são fixadas diretamente nos postes e, portanto, têm pontos de sustentação totalmente diferentes dos testes iniciais, podendo mudar as frequências harmônicas do sinal, modulando as frequências geradas.

4.1.4 - Força da batida do martelo

A intensidade da força com que o martelo é pressionado na cruzeta também gera informações divergentes. Em um determinado instante o agente de campo poderá efetuar uma batida com uma determinada força e em outro instante com outra força. Este problema também foi detectado nos testes iniciais. Conforme a força da batida, um tipo diferente de áudio é capturado. É necessário neste caso termos uma batida constante para podermos capturar sempre sinais de áudio compatíveis.

4.1.5 - Objetos fixados na cruzeta

Nos testes realizados em laboratório, a cruzeta avaliada estava totalmente limpa, ou seja, somente a cruzeta sem parafusos, sem ferragem, sem castanhas, etc.

Conforme o estado com que a madeira está presa, influência de pressão sobre ela e também tração sobre a mesma (fios fixados entre as cruzetas) fazem com que o sinal de áudio sofra influência direta, causando discrepâncias nas leituras do equipamento proposto para a análise da sanidade das cruzetas.

4.1.6 - Conclusões sobre as Análises Iniciais

Conforme pode ser visto, várias influências poderão ocasionar discrepâncias nas leituras dos sinais provenientes das cruzetas. Portanto serão necessários ajustes para a captura dos sinais de áudio para a implementação do sistema inteligente de análise de sanidade de cruzetas.

Um sistema para captura de sinais em laboratório deverá ser definido para que a influência do meio não cause problemas no sinal amostrado. Também deverão ser montados pequenos postes para testes, com alturas menores, para que sejam feitos testes de campo conforme são feitos pelos agentes de campo. Deverão ser fixados postes com altura de no máximo 2 metros. Todos os componentes reais de fixação de uma cruzeta

serão utilizados nestes postes, tais como, parafusos, ferragens, castanhas, cabos interligando um poste ao outro, etc.

Será desenvolvido um equipamento para a obtenção do sinal de áudio onde um solenoide será utilizado para realizar a batida na cruzeta, sempre a uma mesma distância do microfone de captação e da cruzeta.

4.3 - Sistema de Captura de Dados em Laboratório

A seguir são definidos como serão obtidos os dados a serem processados.

4.3.1 - Estudos sobre aquisição de dados da cruzeta sobre bancadas

4.3.1.1 - Ensaios

Um pequeno banco de cruzetas foi levado ao laboratório de testes, contendo várias cruzetas de cada grau de sanidade acima mencionado.

Estas cruzetas foram acomodadas no laboratório para posterior ensaio, não estando nas condições originais do estado de uso das mesmas, tais como sol, chuva, ventos, forças mecânicas e elétricas nas mesmas.

Para um estudo inicial foi feito somente testes percussivos, simulando os ensaios feitos pelas equipes externas de testes da CEMIG. Este teste consiste na aplicação de uma batida de martelo na cruzeta e através de observação sonora definir o estado de sanidade da cruzeta.

Uma batida com martelo será aplicada na cruzeta e o som proveniente desta batida será analisado por um sistema especialista a ser desenvolvido pela equipe do projeto.

As cruzetas utilizadas em nossos estudos possuem dimensão de 24000 x 90 x 112,5 mm.

Para uma análise de toda a estrutura da cruzeta, a mesma foi dividida em setores de 20 cm, totalizando 12 setores, conforme já mostrado na Figura 20.

Nos estudos iniciais o sinal a ser analisado foi coletado na junção de cada setor, em todos os lados da cruzeta: lado esquerdo, lado direito, lado superior e lado inferior.

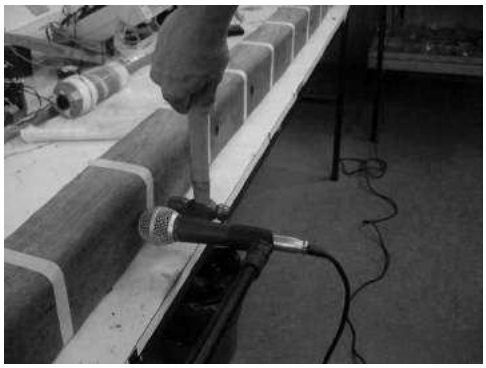
A análise inicial foi feita em uma cruzeta nova, para posterior comparação com análise em cruzetas com os níveis de sanidade acima descritos.

Em uma primeira tomada de decisão, objetivou-se extrair características que possam inferir parâmetros referentes às características da madeira de boa qualidade e posteriormente passar para a análise de madeiras que estejam já comprometidas, que é o foco principal deste projeto.

4.3.1.1.1 - Montagem Inicial

O sinal a ser injetado atualmente é realizado através de batida com martelo, perpendicular a cruzeta, do lado direito, conforme a Figura 33.

Figura 33 - Posicionamento para a batida do martelo



Fonte: Acervo Pessoal

O sinal é coletado por microfone dinâmico CSR-58 e placa de som U46DJ ESI (4 entradas e 6 saídas), com entrada para microfone dinâmico, via USB 2.0, conforme Figura 34.

Figura 34 - Kit de Microfone e Placa de Som USB

Fonte: Acervo Pessoal

Este kit será utilizado inicialmente e posteriormente serão substituídos por hardware específico. A batida a ser coletada pelo kit de microfone (sinal evocado) e placa de som USB será sempre na terceira junção, para se ter uma referência e comparação do sinal nos testes iniciais. O software para a aquisição de dados será inicialmente o LabView Versão Demo e o Sound Forge 10.0 Versão Demonstrativa.

Nos testes iniciais alguns problemas foram detectados:

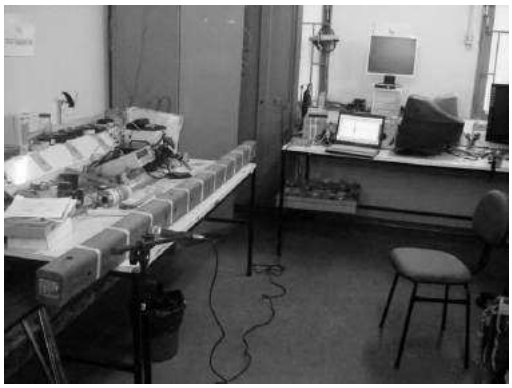
a) Com a cruzeta apoiada totalmente na bancada de testes o sinal evocado irá sofrer influência dos objetos que estão juntos com a cruzeta. Neste caso a superfície de apoio irá gerar um sinal que não é o sinal esperado.

b) Com a cruzeta somente apoiada nas extremidades, entre duas bancadas de testes, o sinal evocado também irá sofrer influências dos apoios, causando ressonância na madeira, causando erros na medição.

4.3.1.1.2 - Montagem em Bancada Única

O primeiro teste foi realizado com a cruzeta apoiada somente em uma bancada, com toda a sua parte inferior (superfície de baixo) apoiada na superfície da bancada, conforme mostra a Figura 35.

Figura 35 - Cruzeta apoiada sobre bancada



Fonte: Acervo Pessoal

4.3.1.1.3 - Montagem em Bancada Dupla

O segundo teste foi realizado com a cruzeta apoiada entre duas bancadas, com somente as extremidades da cruzeta sendo sustentada por cada bancada, conforme Figura 36.

Figura 36 - Cruzeta apoiada entre duas bancadas

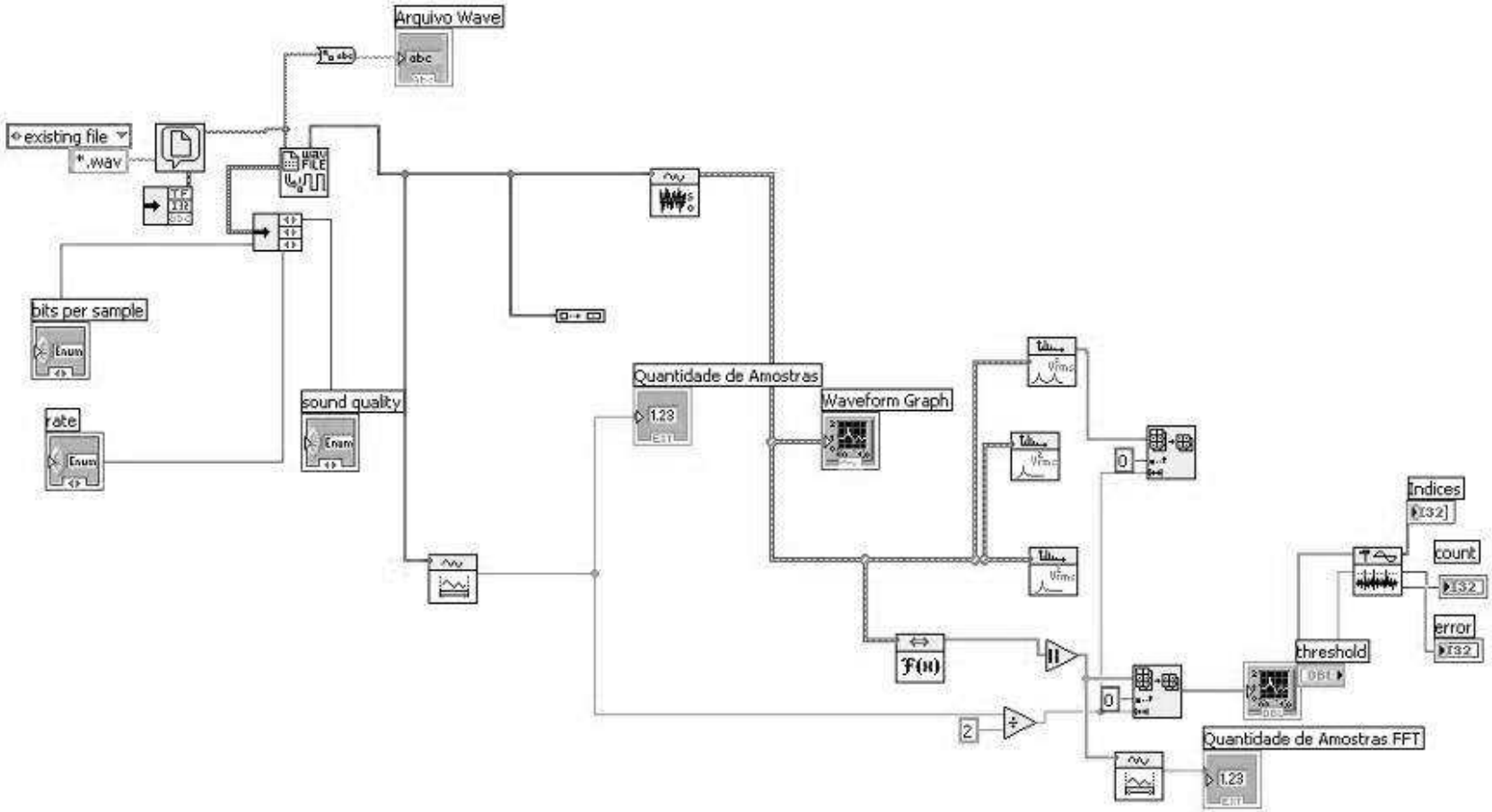
Fonte: Acervo Pessoal

4.3.1.1.3 - Resultados e Observações

Para a comparação do sinal obtido nos dois casos propostos (cruzeta apoiada em uma única bancada e cruzeta apoiada em duas bancadas) foi desenvolvido um software para análise, utilizando o programa da National Instruments, **LABVIEW 8.6**, onde são apresentadas janelas com o sinal de áudio capturado (sinal evocado) e seu respectivo espectro obtido, inicialmente, pela transformada Rápida de Fourier (FFT).

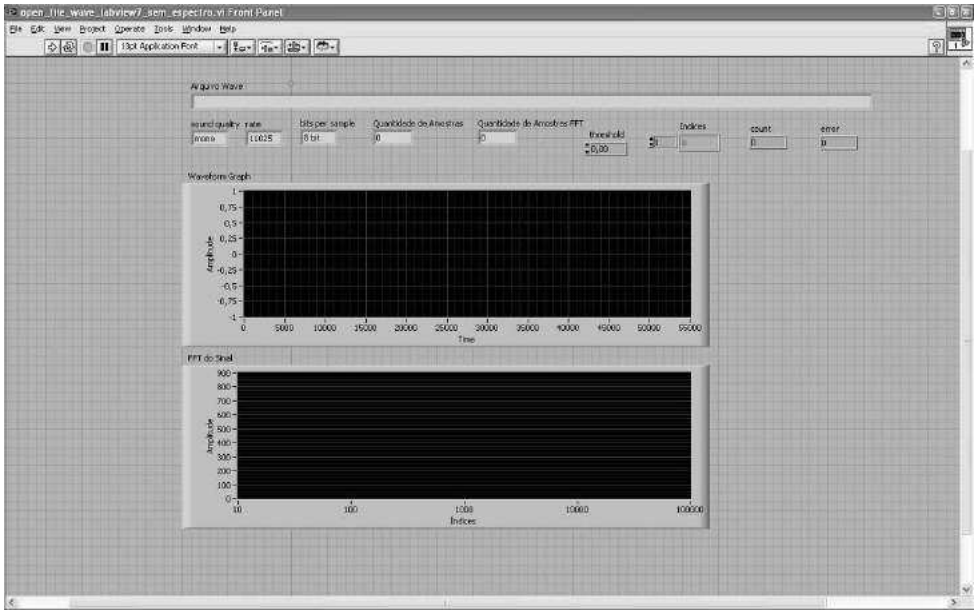
O diagrama de blocos e a amostra da janela do programa são apresentados nas Figuras 37 e 38.

Figura 37 - Diagrama em Blocos do Programa de Captação do Sinal



Fonte: Acervo Pessoal

Figura 38 - Janela do Programa de Captação do Sinal

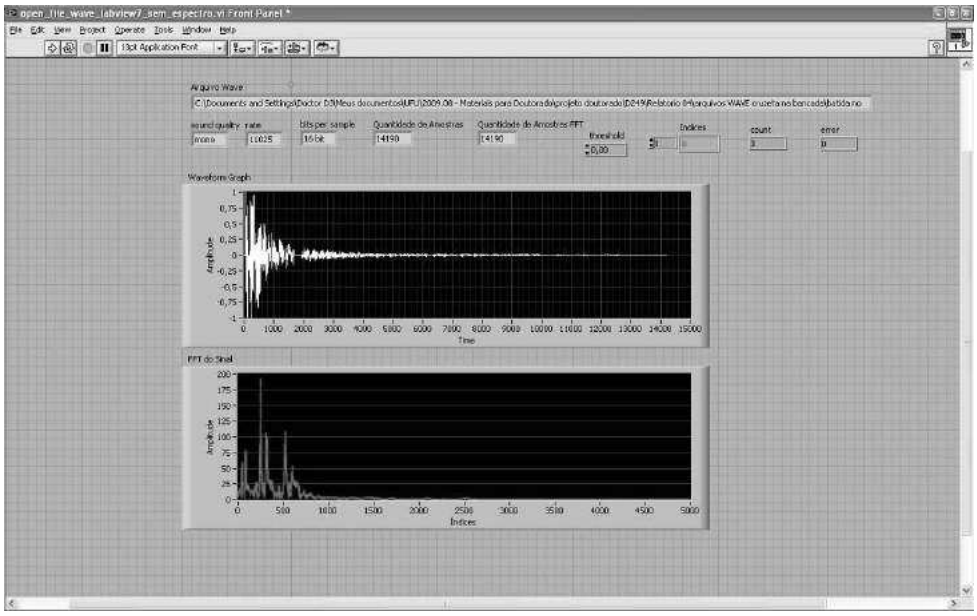


Fonte: Acervo Pessoal

Para os estudos iniciais foi feita uma análise através da Transformada Rápida de Fourier (FFT) para obter uma comparação entre o espectro de frequência nos dois tipos de sinais capturados.

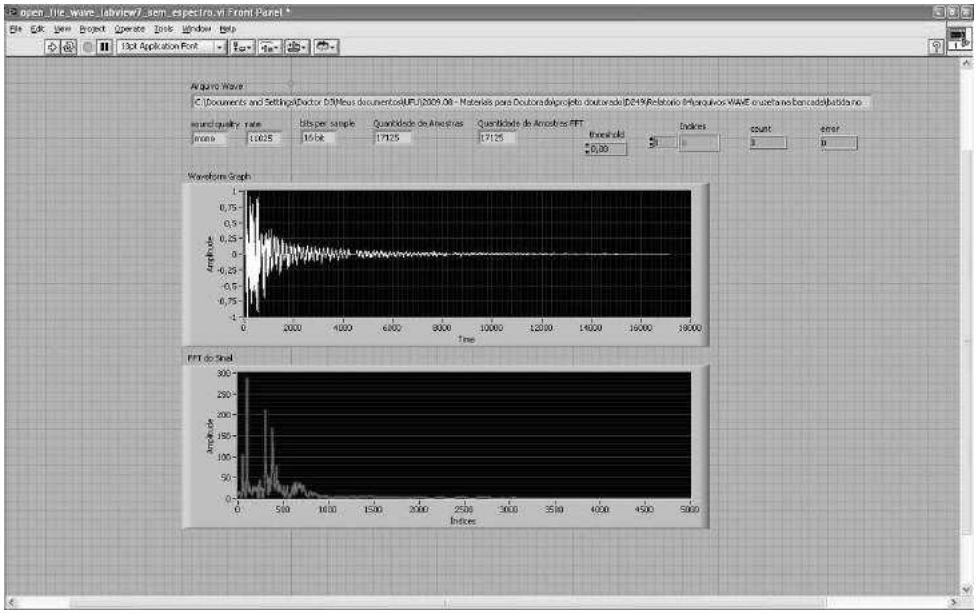
As Figuras 39 e 40 mostram a janela de captura dos sinais nos dois casos propostos:

Figura 39 - Sinal da Cruzeta sobre uma Única Bancada



Fonte: Acervo Pessoal

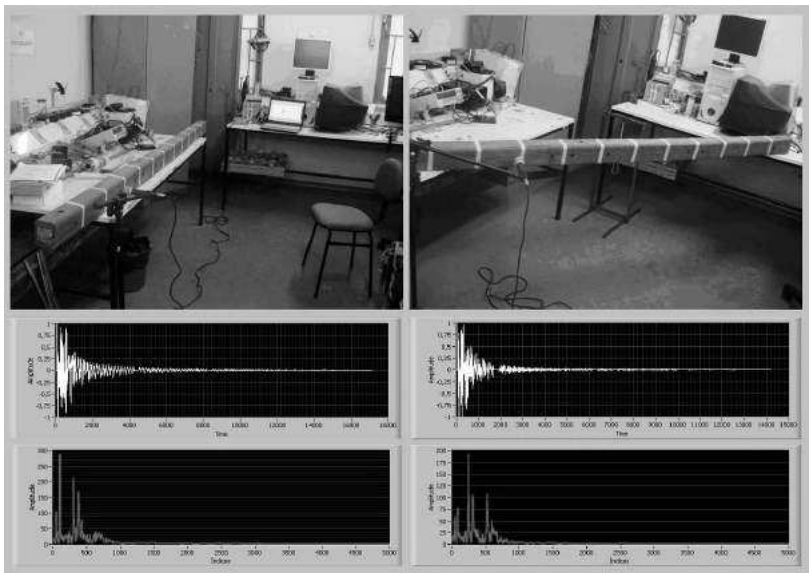
Figura 40 - Sinal da Cruzeta apoiada sobre Duas Bancadas



Fonte: Acervo Pessoal

A Figura 41 mostra a comparação entre os dois tipos de análise (sobre uma bancada e sobre duas bancadas) do sinal obtido através da captura da batida nos dois casos propostos (cruzeta sobre uma bancada e cruzeta sobre duas bancadas), ficando bem claro a disccrepância que é introduzida nos sinais com relação ao tipo de apoio.

Figura 41 - Comparação entre as análises das cruzetas com apoios diferentes: (a) Apoiada sobre uma bancada e (b) Apoiada com as extremidades em duas bancadas



(a) **(b)**

Fonte: Acervo Pessoal

Observa-se que o sinal e seu espectro variam conforme a superfície de apoio utilizada no teste. Esta variação é devido a ressonância do meio (no caso a bancada) que interfere no sinal evocado na cruzeta. Com isto comprova-se a necessidade da criação de um sistema que não introduza erros nas medições para que sejam observados sempre os mesmo parâmetros de análise, bem como uma solução para análise em laboratório, sem interferência do meio.

É necessário ter-se um conhecimento de como a cruzeta, sem nenhuma interferência do meio, reage com o sinal injetado (batida do martelo). De posse disse ter-se-á o sinal puro da cruzeta e, posteriormente, serão inseridos os componentes que fazem parte de uma cruzeta normalmente fixada em um poste, tais como parafusos de fixação, isoladores, etc, e com isto serão descobertos os efeitos que eles causam na cruzeta que, poderão ser considerados como ruídos externos.

Todos estes sinais observados serão de grande importância para a análise futura, pois em cada caso da fixação de uma cruzeta, diversos fatores irão contribuir para a detecção do estado de sanidade da mesma:

- Ponto de fixação da cruzeta no poste (centro, lateral, etc..)
- Quantidade de acessórios utilizados (parafusos, pinos, isoladores, etc..)
- Tipo do poste
- Tipo da madeira da cruzeta

4.3.2 - Estudos sobre aquisição de dados da cruzeta sem interferência do meio

Conforme visto anteriormente necessita-se de uma estrutura que permita a análise das cruzetas em laboratório sem interferência do meio.

Estas interferências podem ser a superfície de contato onde a cruzeta é colocada, os acessórios fixados na mesma, tais como: parafusos, isoladores, cabos, etc.

Durante todo o projeto serão feitos testes exaustivos em cruzetas de diversos tipos de sanidade. Para isto seria impossível transportar todas estas cruzetas para o laboratório de testes ou montá-las nos postes dentro do Laboratório de Campo (Apêndice A).

Em estados críticos de quebra de cruzeta o ideal é fazer os testes no exato momento da troca da cruzeta ou detecção da quebra. Para que o transporte da cruzeta fosse feito para o laboratório de testes o estado da cruzeta poderia ser alterado devido ao tempo gasto para a aquisição da cruzeta e bem como a chegada da mesma ao laboratório.

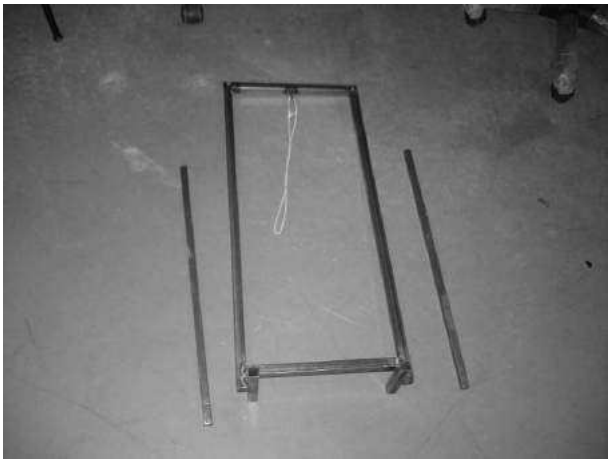
Diante disto, fez-se necessário a criação de um sistema que pudesse manipular as cruzetas sem interferências do meio, tais como apoio para a superfície, equipamentos colocados nas mesmas e bem como o teste no local onde a mesma já está sendo trocada, para a obtenção do sinal exato no momento da troca.

Com isto foi desenvolvido um sistema de cavaletes, onde a cruzeta ficará suspensa por cordoalhas que não irão interferir na aquisição dos dados (batidas) das cruzetas.

Este sistema foi chamado de **SISTEMA PORTÁTIL DE ANÁLISE DE CRUZETA SEM INTERFERÊNCIA DO MEIO (SPACSIM)**, onde a mesma será apoiada nas extremidades com cordão de nylon cuja massa e características acústicas não interferem na analise da cruzeta.

Estes cavaletes são mostrados nas Figuras 42 e 43.

Figura 42 - SPACSIM Desmontado



Fonte: Acervo Pessoal

Figura 43 - SPACSIM Montado

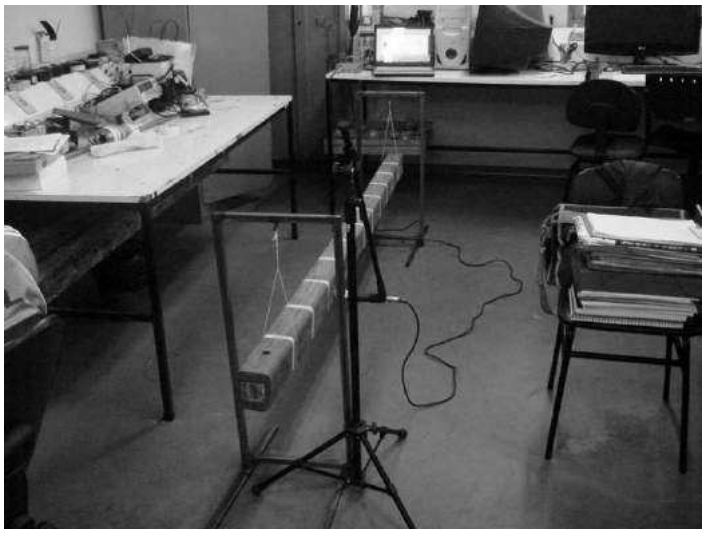
Fonte: Acervo Pessoal

4.3.2.1 - Ensaio

A mesma cruzeta utilizada nos testes anteriores agora será testada no SPACSIM para comparação entre os sinais.

A Figura 44 mostra como foi feito o ensaio da cruzeta.

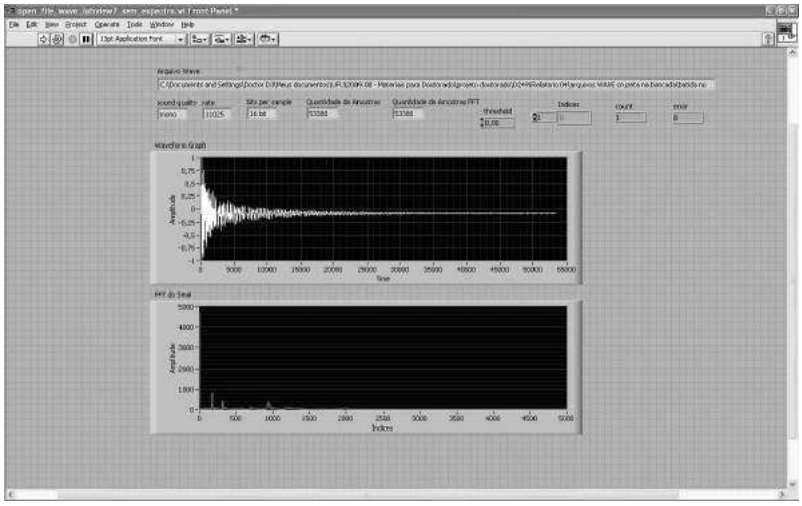
Figura 44 - Sistema Completo em Funcionamento



Fonte: Acervo Pessoal

Através da Figura 45 é possível analisar o sinal capturado através da batida do martelo na terceira marca da cruzeta, do lado esquerdo.

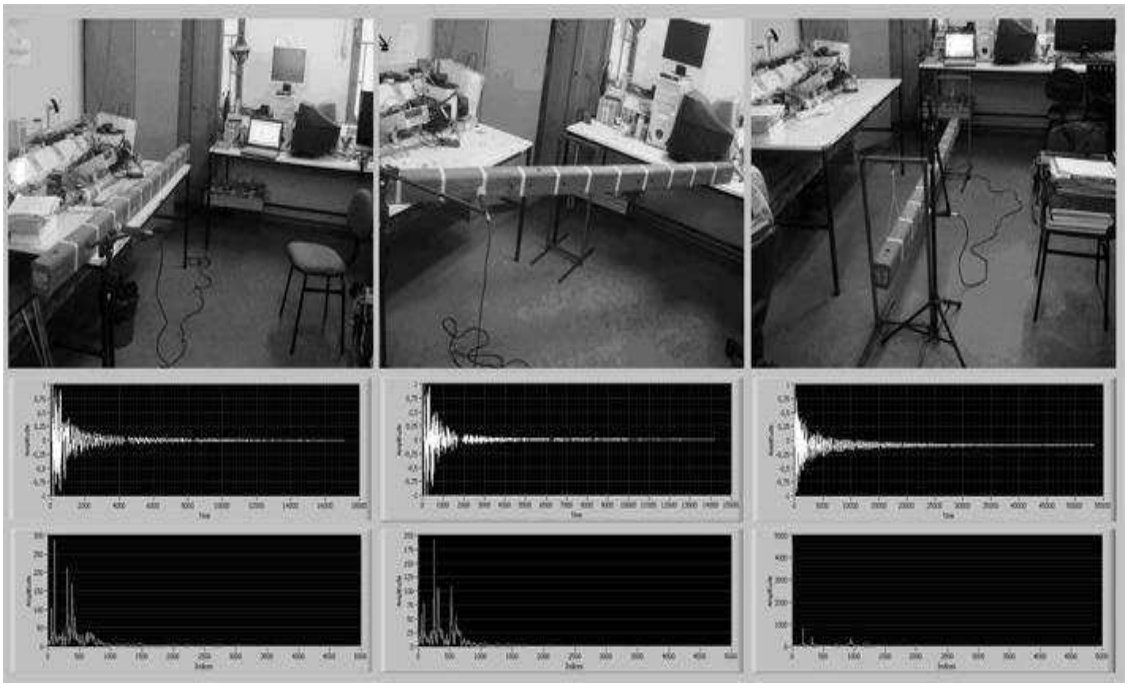
Figura 45 - Forma de Onda e Análise do Sinal montado no SPACSIM



Fonte: Acervo Pessoal

Na Figura 46 é possível agora fazer uma comparação entre os sinais nos três ensaios.

Figura 46 - Comparação entre os 3 tipos de análise: (a) Cruzeta apoiada sobre uma bancada, (b) Cruzeta com as extremidades apoiada entre duas bancadas e (c) Cruzeta no SPACSIM



(a) **(b)** **(c)**

Fonte: Acervo Pessoal

É possível observar que quando a cruzeta está apoiada com suas extremidades nas bancadas o sinal proveniente da batida ressona mais causando a introdução de mais harmônicos no sinal original. Diferente de quando está no SPACSIM, que não gera estes harmônicos, sendo possível observar somente o sinal sem a introdução destes harmônicos.

Com este sistema será possível o deslocamento até as unidades onde deverão ser feitas as medições, facilitando a aquisição de dados. Como é um sistema pequeno, de fácil transporte e de fácil manuseio, uma quantidade enorme de medições poderá ser feitas em um único local e dia específico, facilitando em muito o processo de ensaios.

Sinais poderão ser coletados no local onde ocorrer um problema. Este sinal irá compor o banco de dados de informações. Estima-se que analisando a cruzeta no Sistema Portátil e comparando com as cruzetas montadas nos postes possa-se conhecer e eliminar as características que os equipamentos fatalmente irão inserir nos sinais evocados nas cruzetas.

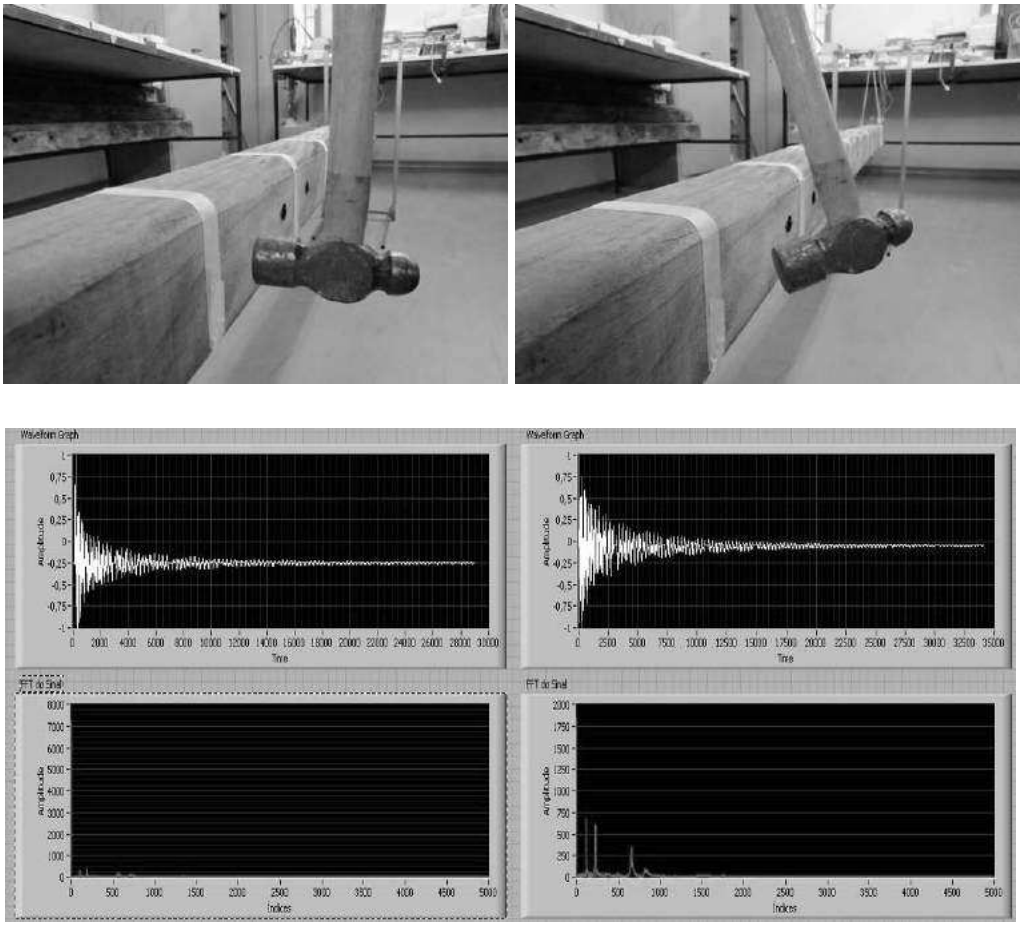
4.3.3 - Estudos sobre características do sinal a ser injetado na cruzeta

Nas visitas técnicas e manobras efetuadas nas instalações da CEMIG D foi constatado que a forma de análise do sinal é através de batidas de martelo nas cruzetas e posterior observação audível do sinal proveniente desta batida.

Em testes empíricos em laboratório foi observado que alguns fatores vão interferir no resultado final do sinal capturado. Foi possível observar os seguintes fatores:

a) **Posição do martelo com relação à cruzeta:** conforme pode ser visto na Figura 47 o sinal resultado de uma batida do martelo em posições diferentes causa resultados diferentes na análise espectral do sinal. Batidas com o martelo em um ângulo reto com relação à cruzeta gera menor ressonância, com uma quantidade maior de harmônicos, do que uma batida com o martelo em uma posição mais inclinada com relação à cruzeta.

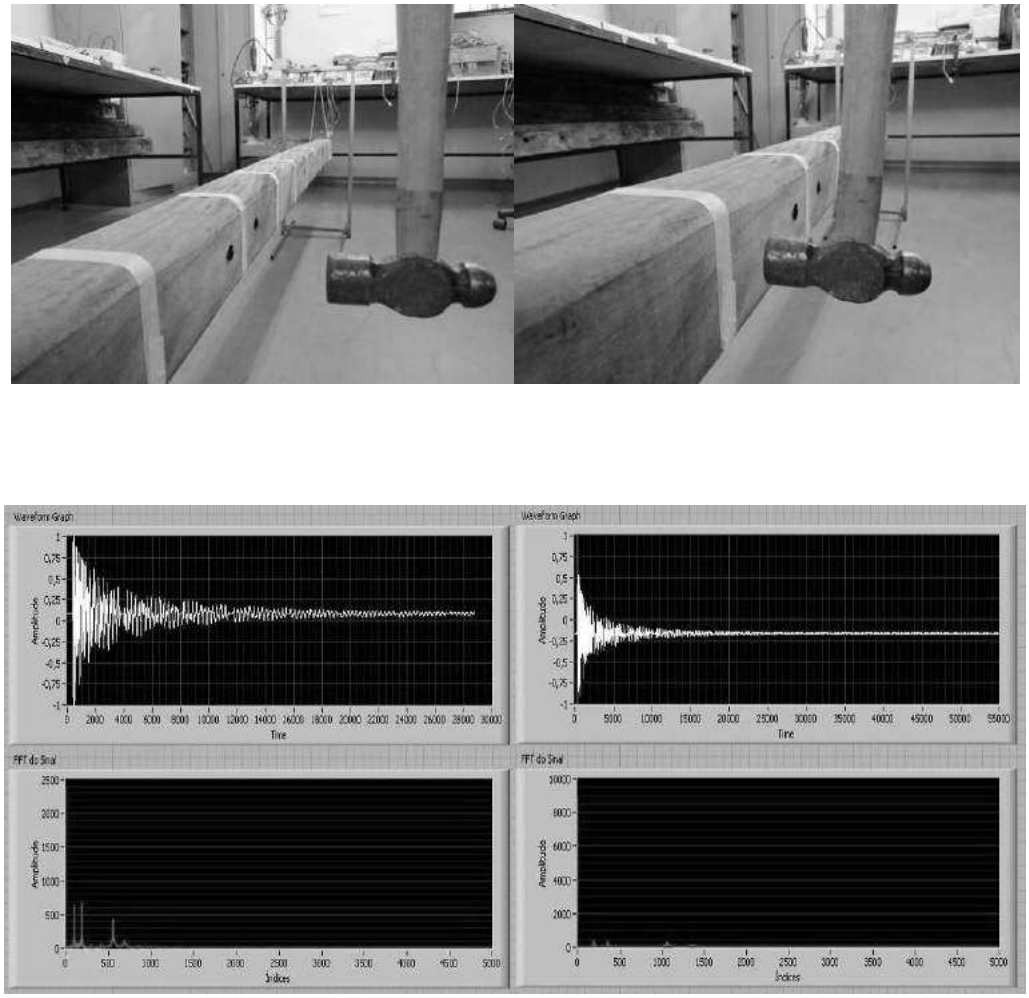
Figura 47 - Análise do Sinal para batidas com posições diferentes do martelo



Fonte: Acervo Pessoal

b) **Intensidade da batida do martelo:** a força aplicada no martelo durante a batida também resulta em análise espectral diferente no sinal obtido, conforme mostra a Figura 48. Na Figura 48.a tem-se aplicação de um sinal com maior intensidade do que na Figura 48.b devido a um maior afastamento do martelo da cruzeta no instante da batida gerando-se com isto um sinal com maior intensidade e duração.

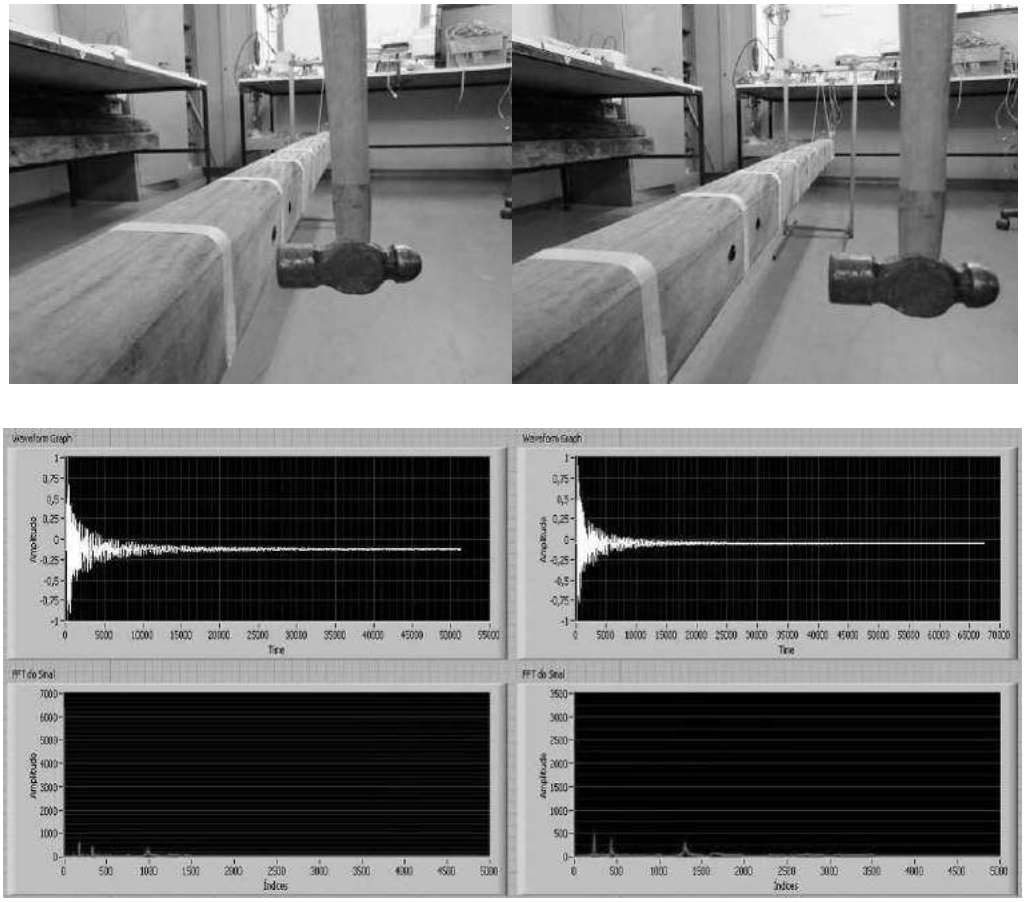
Figura 48 - Análise do Sinal para Batidas com Intensidades Diferentes



Fonte: Acervo Pessoal

c) **Distância entre martelo e cruzeta:** a distância com que o martelo é afastado da cruzeta para ser executada a batida também gera uma análise espectral diferente no sinal obtido, pois conforme a distância de afastamento do martelo maior ou menor será a força do sinal e também a posição final da batida poderá ser alterada. A Figura 49 mostra os resultados para este tipo de análise. Na Figura 49.a a batida foi com uma distância menor do que a da Figura 49.b. Devido à diferença de distância, mas mantendo-se a mesma forma, a Figura 49.a mostra um sinal mais consistente, com pouca ressonância. Já o sinal da Figura 49.b é mais ressonante, pois o martelo percorreu um maior espaço.

Figura 49 - Análise do Sinal para Distâncias diferentes do martelo com relação a cruzeta



Fonte: Acervo Pesso

Com estas conclusões faz-se necessário a criação de um sistema que efetue sempre a mesma batida seguindo as seguintes características:

- Mesma posição (inclinação) com relação à cruzeta
- Mesma intensidade de batida
- Mesma distância do sistema com relação à cruzeta

O sistema a ser produzido e/ou adquirido será denominado “Martelo Magnético”.

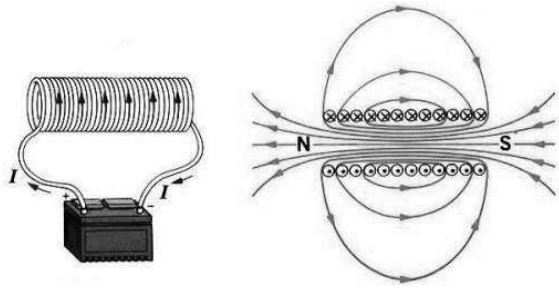
4.3.3.1 - Martelo Magnético

Diversas buscas foram feitas no mercado de eletrônicos para a obtenção de um solenoide que fosse compatível com o sistema a ser utilizado, mas nada foi encontrado. Optou-se pela criação de um sistema eletrônico que efetuasse as batidas conforme as necessidades. O

principio de funcionamento do martelo magnético deriva do funcionamento de um solenoide.

Um solenoide pode ser definido como um conjunto de espiras de mesmo eixo espaçadas uniformemente, como mostra a Figura 50.a:

Figura 50 - Solenoide



(a) (b)

Fonte: Internet

Ao alimentar a bobina eletricamente, um campo magnético é formado em seu interior. A figura 50.b mostra as linhas do campo. A densidade do campo magnético gerado é dada pela Equação 24:

$$B = \mu_0 \frac{N}{L} i \quad (24)$$

Onde: μ_0 é a permissividade eletromagnética;

N é o numero de espiras;

L é o comprimento da bobina;

i é a corrente que passa pela bobina.

Quando um êmbolo de ferro é posicionado no interior da bobina, o campo criado no momento da energização o atrai fazendo com que o embolo tente se alinhar com o centro do campo magnético, conforme mostra a Figura 51.

Figura 51 - Êmbolo

Fonte: Internet

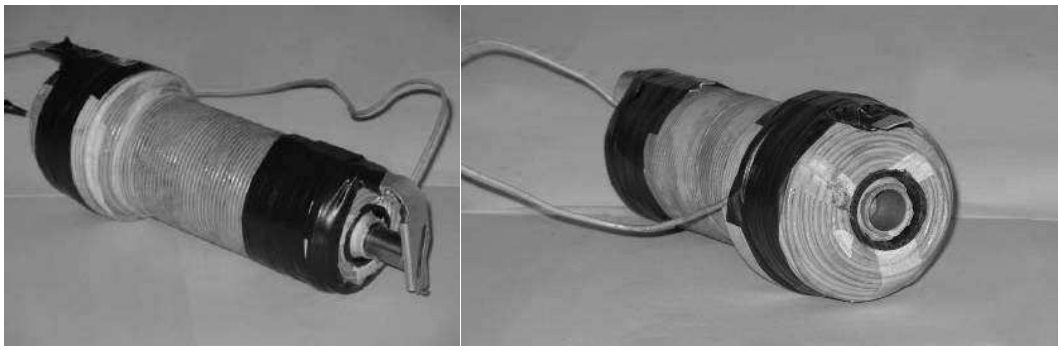
O centro magnético do campo induzido no interior da bobina pode ser deslocado se for concentrado um maior número de espiras na ponta da bobina.

A bobina foi fabricada com as seguintes especificações:

- 22,50 cm de comprimento;
- Núcleo com 1 polegada de espessura;
- Fio rígido encapado de cobre com 1,50 mm de espessura;
- 480 espiras, sendo 6 camadas com 70 espiras e 4 camadas com 15 espiras;
- 16 Ω de resistência medida nos terminais da bobina.

A figura 52 mostra o solenoide após a finalização de todos os testes.

Figura 52 - Vista Final do Solenoide



Fonte: Acervo Pessoal

O calculo da força magnética que empurra o êmbolo pode ser feito a partir da aplicação das seguintes fórmulas:

$$F_m = BIL \sin\theta \quad (25)$$

Onde: B é a densidade de fluxo magnético;

I é a corrente que percorre as espiras da bobina;

L é a indutância do;

θ é o ângulo entre a corrente e a densidade de fluxo.

Sabendo que a densidade de fluxo magnético é dada pela fórmula:

$$\mathrm{B} = \frac{LI}{NA} \quad (26)$$

E que a indutância da bobina é dada por

$$L = \mu_0 N A_0 \quad (27)$$

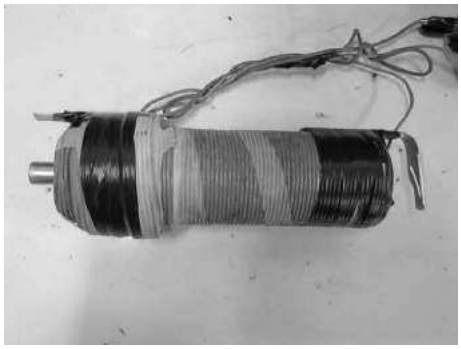
Logo a forca magnética é dada por:

$$F_M = (I\mu_0)^2 N A_0 \sin\theta \quad (28)$$

Onde μ_0 é a permeabilidade magnética, N é o número de espiras e A é a área da bobina.

Foi verificado empiricamente que ao se colocar o êmbolo numa posição inicial de 2 cm para fora da bobina e aplicando-se uma tensão sobre esta bobina, o êmbolo tentará se alinhar com o campo, mas como o centro magnético do campo na bobina está concentrado na ponta, parte do êmbolo ficará de fora da bobina como mostra a Figura 53.

Figura 53 - Solenoide com o Centro do Campo magnético deslocado



Fonte: Acervo Pessoal

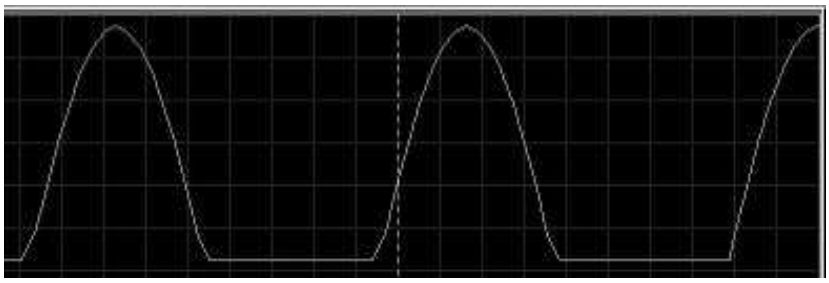
Se a cruzeta for posicionada na frente do êmbolo, no momento em que o êmbolo tentar se alinhar com o campo, ele irá bater na cruzeta fazendo com que ela emita um som. Este som será capturado pelo sistema especialista e analisado posteriormente.

A batida do êmbolo na cruzeta não pode ser muito intensa de tal forma que danifique a cruzeta, mas também não pode ser tão fraca a ponto de ser imperceptível.

A força com que o êmbolo irá bater na cruzeta irá depender da distancia que esta última está do embolo, da força magnética que o campo empurra o embolo e do posicionamento inicial do êmbolo.

Foi aplicada nos terminais do solenoide uma tensão alternada de 110 Volts e 60 Hz após passar por um diodo que suporta uma corrente direta de 50 A. Este diodo retifica a onda senoidal cortando os semi-ciclos negativos funcionando como um retificador de meia onda como mostra a Figura 54.

Figura 54 - Forma de onda de tensão aplicada nos terminais da bobina

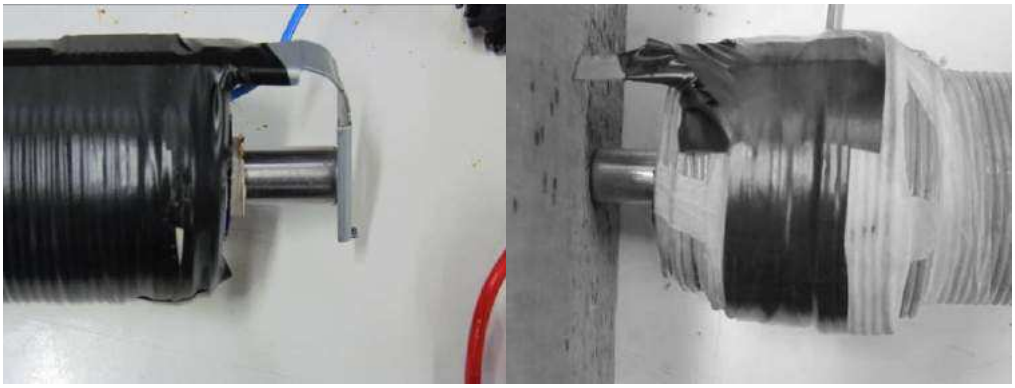


Fonte: Acervo Pessoal

Através de teste empíricos foram determinadas as distâncias de acomodação inicial do êmbolo, bem como a distância que deverá ser posicionado o martelo com relação a cruzeta. Em testes iniciais foi constatado que a força final do martelo é muito intensa e que se fosse utilizado todo o tamanho do êmbolo para a aplicação da energia a batida do êmbolo na cruzeta seria algo desastroso.

Com os testes empíricos as distâncias de acomodação inicial do êmbolo e da cruzeta ficaram em 2 cm, conforme mostram a Figura 55.

Figura 55 - (a) Distância de acomodamento inicial do êmbolo e (b) distância entre o martelo e a cruzeta



(a) (b)

Fonte: Acervo Pessoal

Com este sistema, denominado Martelo Magnético, ter-se-á sempre uma batida de mesma intensidade, mesma distância com relação a cruzeta e mesmo ângulo.

Desta forma o sinal capturado sempre será o mesmo com relação a cruzeta, podendo ser analisado corretamente, livre de influências do meio em que está sendo feita a medição.

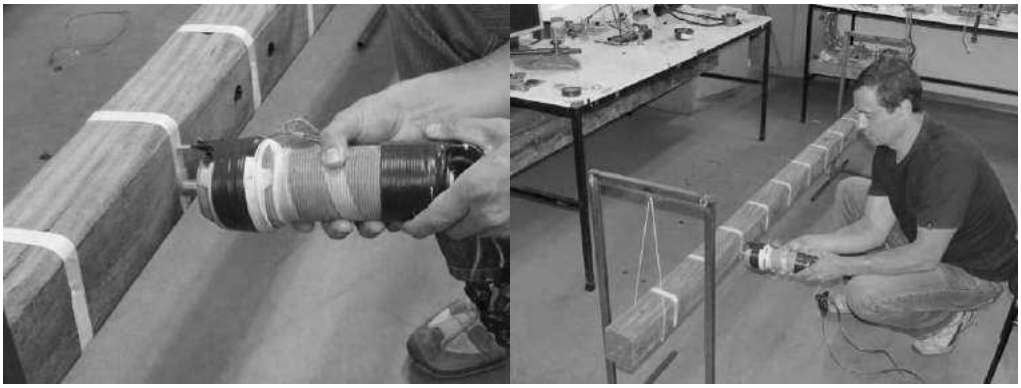
Este modelo foi criado para demonstrar a necessidade do desenvolvimento de um equipamento específico. Nos próximos itens já será mostrado o modelo final utilizado para a captura dos dados definitivamente.

4.3.3.1.1 - Ensaios

Com a criação do martelo magnético foi possível já efetuar testes iniciais de captura de sinal para comparação com os sinais já obtidos resultantes de batidas de martelo manual na cruzeta.

A Figura 56 mostra a aquisição de sinais proveniente da utilização do martelo magnético em conjunto com o SPACSIM.

Figura 56 - Aplicação do Martelo Magnético na Cruzeta



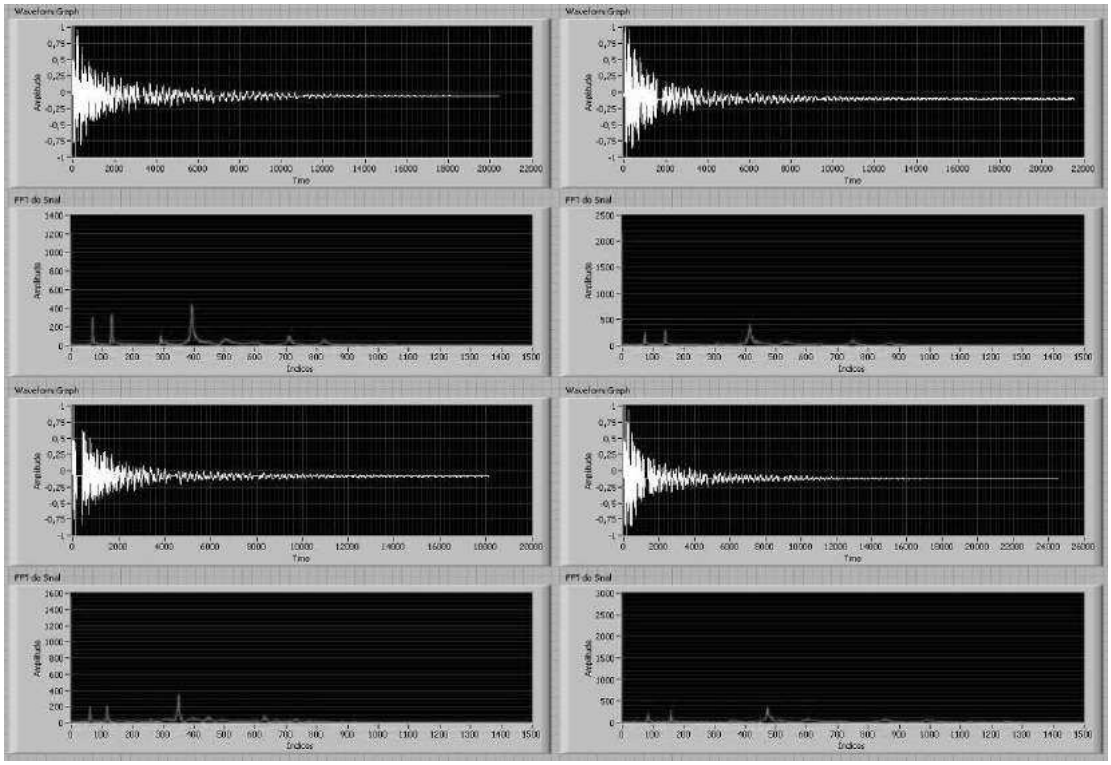
Fonte: Acervo Pessoal

4.3.3.1.2 - Resultados e Observações

Com o sistema montado e já em funcionamento alguns sinais foram capturados para demonstrar o efeito do uso do martelo magnético. Uma série de quatro (04) amostras foi feita individualmente e em intervalos de tempo distintos.

Na Figura 57 são mostradas as formas de ondas e a análise espectral obtida dos sinais capturados utilizando sempre uma batida na mesma posição na cruzeta, ou seja, terceiro seguimento e sempre na lateral esquerda da cruzeta, sendo da mesma forma que nas outras medições. Para estas figuras foram aumentadas as áreas de amostragem para uma boa comparação entre os sinais. É possível notar que as formas de onda, bem como amplitudes, passam a ser constantes, independentes da batida. Na análise espectral nota-se também que existem os mesmos harmônicos em todas as batidas.

Figura 57 - Formas de ondas e análises espectrais da aquisição de quatro batidas independentes do martelo magnético em uma cruzeta boa



Fonte: Acervo Pessoal

4.3.3.1.3 - Conclusões

Através da análise dos espectros de frequência das amostras obtidas utilizando o martelo magnético é possível observar que o sinal é bem similar, não sendo interferido pelos problemas causados por posição, força e distância da batida na cruzeta.

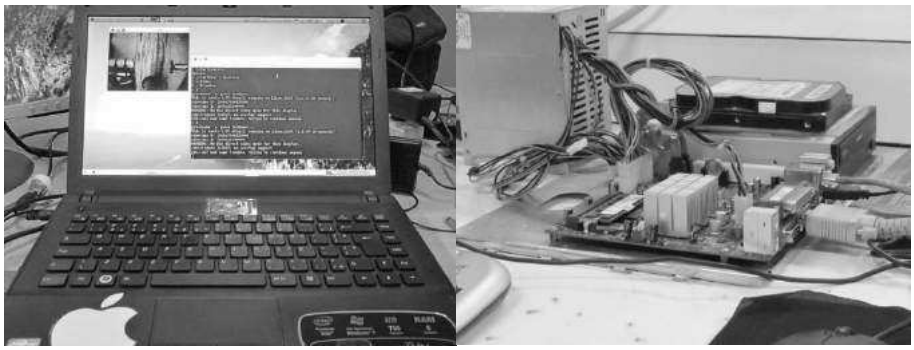
Com o uso do martelo magnético poderá ser feita uma análise real do sinal capturado e, com isto, obter-se-á um sinal puro cujas informações serão importantes no estudo da sanidade de uma cruzeta nova (sadia) para obtenção de parâmetros necessários para comparação com cruzetas defeituosas.

4.4 - Desenvolvimento do Hardware

4.4.1 - Hardware

Inicialmente o sistema foi sendo desenvolvido em um computador notebook para testes em laboratório e posteriormente inserido em uma micro motherboard com processador Atom e entradas/saídas usb. A Figura 58 mostra o notebook e a micro motherboard.

Figura 58 - (a) Computador Notebook e (b) Micro Motherboard Atom



(a) (b)

Fonte: Acervo Pessoal

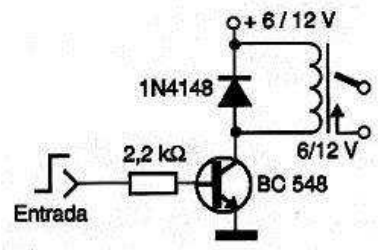
A captura do sinal proveniente da batida do martelo é feita através de microfones de contato do tipo Captador de Instrumentos DEVAL GCS, conforme mostra a Figura 58.

Figura 59 - Captador para a batida do martelo

Fonte: Internet

O disparo do martelo eletrônico foi implementado através de porta PARALELA, saída de dados D0 e utilização de driver para fechamento e abertura de relê conforme mostra a Figura 60.

Figura 60 - Driver de acesso ao relê de disparo do martelo eletrônico



Fonte: Internet

Para a aquisição de imagem (que será utilizada na varredura com a câmera no topo da vara de manobra) está sendo utilizada uma interface de captura de áudio/vídeo do tipo Dazzle Digital Vídeo Creator 90, conforme mostra a Figura 61. A opção para uma interface de captura de vídeo e não a utilização de câmeras USB deve-se ao fato da distância entre a câmera e a porta USB. No caso da vara de manobra esta distância irá superar a distância máxima de 2 metros para que câmeras USB possam transmitir com qualidade.

Figura 61 - Interface de Captura de Vídeo

Fonte: Acervo Pessoal

A esta interface é acoplada uma mini-câmera de vigilância colorida, conforme mostra a Figura 62.

Figura 62 - Micro Câmera

Fonte: Acervo Pessoal

Para a captura dos sinais de áudio será utilizada uma placa de som externa (USB) modelo MAYA44 USB. A utilização deste modelo de placa de som faz-se necessário neste estágio de desenvolvimento, pois será necessário capturar vários sinais de áudio provenientes da batida do martelo na cruzeta, batidas estas oriundas juntamente ao lado da batida do martelo,

lado posterior a batido do martelo (propagação na sessão transversal da cruzeta) e lado superior da cruzeta (visão da câmera).

Esta placa de som possui quatro (04) entradas e quatro (04) saídas de áudio. Possui baixa latência (atraso entre o sinal de áudio de entrada e o sinal de áudio de saída após processamento no computador). A Figura 63 mostra o modelo desta placa.

Figura 63 - Placa de Som MAYA44 USB

Fonte: Internet

O Pré-Processamento do sinal de áudio provenientes dos microfones de contato será feito através de um minimixer MX400 Behringer, conforme mostra a Figura 64. A opção por este modelo de mixer de som fez-se necessário devido ao seu tamanho reduzido e baixo ruído dos pré-amplificadores.

Figura 64 - Equipamento utilizado como pré-processador de áudio

Fonte: Internet

4.4.3 - Martelo Eletrônico

O martelo eletrônico foi confeccionado em liga de metal em formato de garra para ser acoplado diretamente na cruzeta, através de vara de manobra com isolamento elétrico.

As Figuras 65-66 mostram os detalhes do martelo eletrônico.

Figura 65 - Detalhe da Estrutura do Martelo Eletrônico

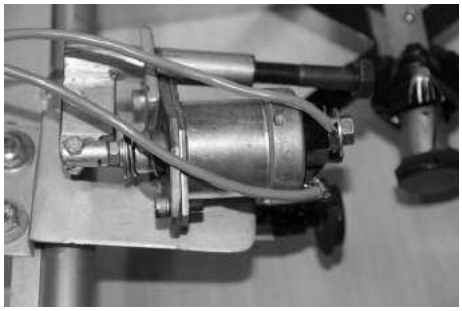
Fonte: Acervo Pessoal

Figura 66 - Estrutura para fixação do martelo a vara de manobra

Fonte: Acervo Pessoal

A bobina de ativação do martelo eletrônico foi confeccionada com potência suficiente para simular a batida de uma marreta que é utilizada normalmente pelos especialistas de campo da Cemig. Os detalhes da bobina do martelo eletrônico são mostrados na Figura 67.

Figura 67 - Bobina de ativação do martelo eletrônico



Fonte: Acervo Pessoal

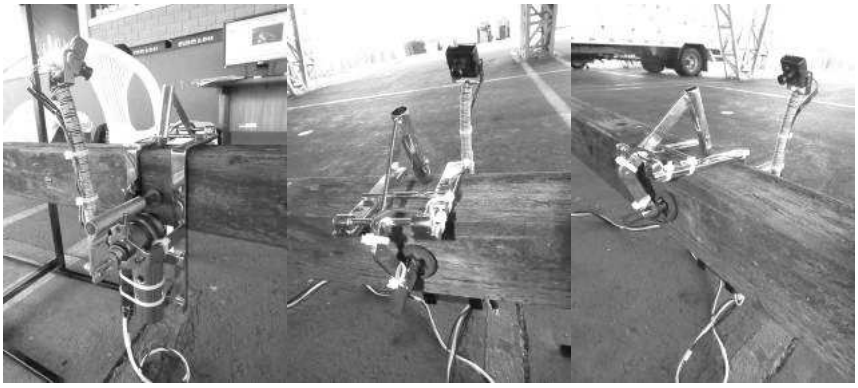
Um martelo manual também foi desenvolvido para, caso necessário, seja acionado diretamente pelo operador da vara de manobra, através de cabo de disparo. Este modelo de martelo é mostrado na Figura 68.

Figura 68 - Martelo Manual

Fonte: Acervo Pessoal

A forma como o martelo eletrônico fica acoplado à cruzeta é mostrada na Figura 69.

Figura 69 - Martelo Acoplado a Cruzeta



Fonte: Acervo Pessoal

O acoplamento em uma cruzeta real, acoplada em poste e com todos os acessórios e cabos é mostrado na Figura 70.

Figura 70 - Cruzeta e Martelo Eletrônico em Lab. de Campo

Fonte: Acervo Pessoal

O desenvolvimento do martelo eletrônico possui a finalidade de teste em campo e, para isto, é necessário a fixação do mesmo em uma vara de manobra. A Figura 71 mostra o martelo fixado nesta vara.

Figura 71 - Martelo fixado em Vara de Manobra. (a) Manual (b) Eletrônico

Fonte: Acervo Pessoal

Através da Figura 72 é possível notar que o martelo eletrônico é de fácil manuseio com a vara de manobra com seus estágios elevados.

Figura 72 - Martelo Eletrônico em Vara de Manobra Estendida

Fonte: Acervo Pessoal

4.4.4 - Captura de Sinais

Com o hardware implementado foi agendada uma visita as instalações da Cemig para a coleta de dados para testes iniciais e estudos sobre a qualidade do sinal amostrado bem como validação de dados.

Um especialista em avaliação de cruzetas da Cemig, Eurico Farina, foi solicitado para fazer a avaliação das cruzetas.

Para este processo foram utilizadas sessões transversais da cruzeta para a análise de sanidade. Em diversos estudos foi constatada que a sanidade de uma cruzeta não se refere a toda à extensão da cruzeta, mas sim em partes da mesma. Uma cruzeta pode possui os três estados de sanidades procurados em toda a sua extensão, ou seja, em uma parte da cruzeta a

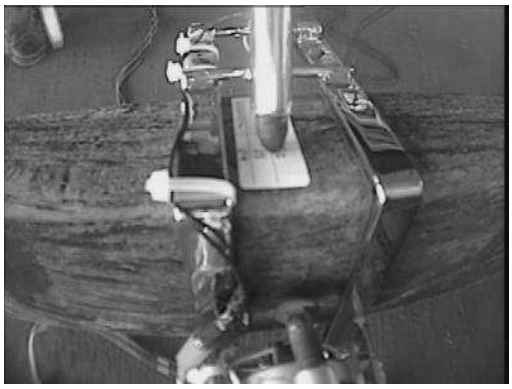
sanidade poderá ser de **1 ANO**, em outro pode ser **EMERGENCIAL** e em outro poderá ser **URGENCIAL**. Por isto faz-se necessária a avaliação em pedaços isolados da cruzeta.

4.5.4.1 - Organização dos Arquivos Coletados

Para cada tipo de medição existem três (03) arquivos:

ax.jpg : imagem do ponto de medição com o martelo, conforme mostra a Figura 73.

Figura 73 - Foto capturada da câmera do Martelo Eletrônico



Fonte: Acervo Pessoal

ax.txt : posição no arquivo de áudio onde inicia o sinal a ser analisado (offset), medido em amostras.

ax.wav : arquivo de som contendo os dados do microfone de contato fixado do lado oposto ao dispositivo de batida do martelo eletrônico.

A Figura 74 mostra como foram feitos os trabalhos na CEMIG D.

Figura 74 - Coleta de Dados na Cemig: a) Hardware b) Montagem da Cruzeta

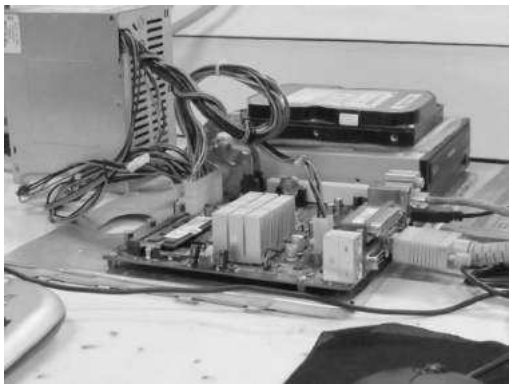
Fonte: Acervo Pessoal

4.4.5 - Hardware Reduzido

O hardware inicial utilizado na captura dos sinais na CEMIG D possui uma dimensão e componentes extremamente grandes. Faz-se necessária a redução na sua dimensão bem como utilização de hardware dedicado para o processamento dos sinais.

Para o sistema de processamento foi adquirida uma Mini-Motherboard com processador Atom e entradas/saídas usb, Hard Disk para notebook, 2 GB de Memória RAM, conforme mostra a Figura 75.

Figura 75 - Mini-Motherboard



Fonte: Acervo Pessoal

Para o monitor de saída será utilizado um monitor de 7 polegadas, conforme mostra a Figura 76.

Figura 76 - Mini Monitor

Fonte: Internet

A interação do usuário com o protótipo será através de um mini teclado numérico com as teclas pré-programadas somente com as funções necessárias ao programa. A Figura 77 mostra este mini teclado.

Figura 77 – Teclado Numérico

Fonte: Internet

Com o uso destes componentes foi obtida uma redução considerável ao protótipo. A Figura 78 mostra um comparativo entre o hardware inicial e o novo hardware, onde o hardware de tamanho maior é mostrado a esquerda da figura e o novo hardware é mostrado a direita da figura.

Figura 78 - Comparativo dos Hardwares

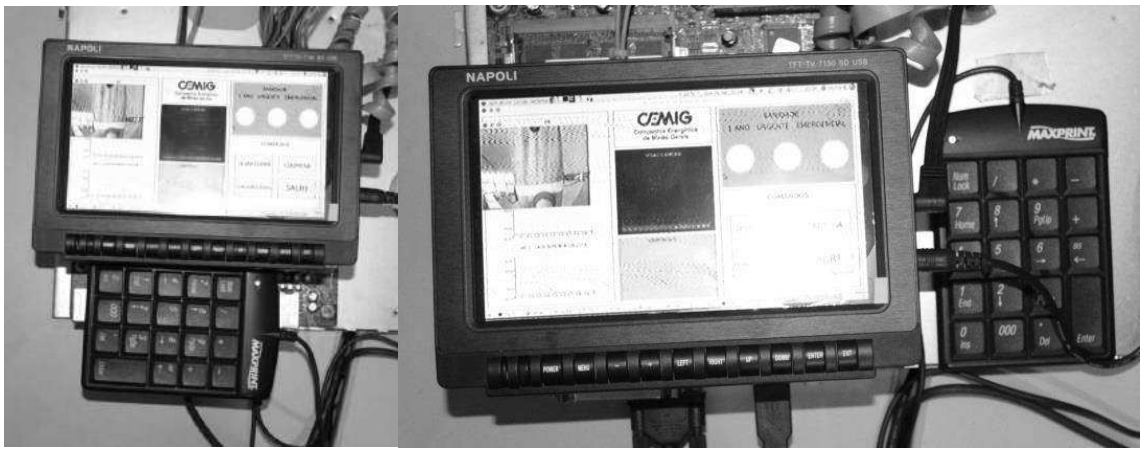
Fonte: Acervo Pessoal

A alimentação elétrica do sistema é feita de duas maneiras:

- Através de bateria de 12 Volts
- Através de fonte de alimentação externa, ligada diretamente na rede elétrica 110/220 Volts.

O protótipo final, após redução do hardware, pode ser visualizado nas Figuras 79 e 80.

Figura 79 - Novo Hardware em funcionamento



Fonte: Acervo Pessoal

Figura 80 - Hardware Final

Fonte: Acervo Pessoal

4.5 - Definições do Software

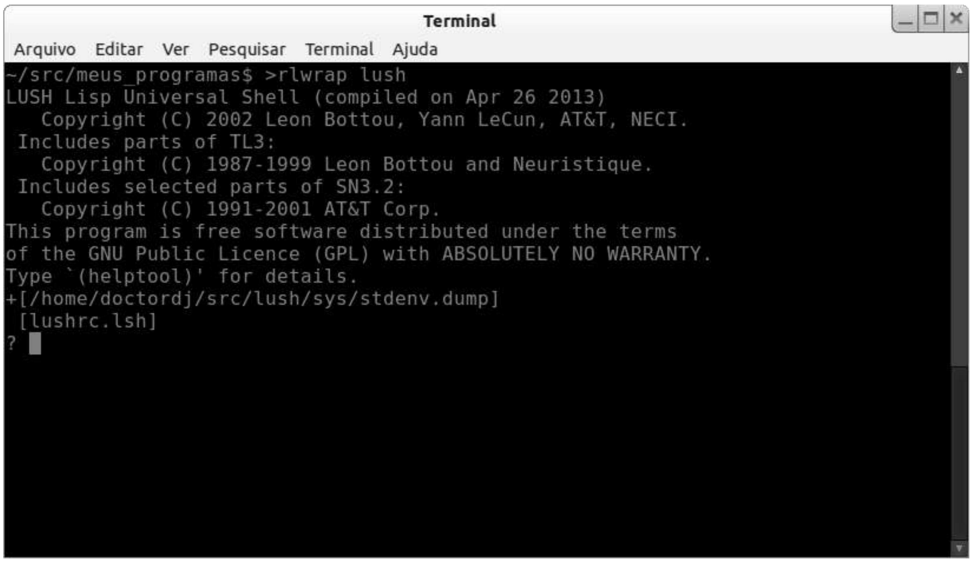
O desenvolvimento destes algoritmos foi feito utilizando a linguagem de programação Lush versão 1.2.1 (http://sourceforge.net/projects/lush/files/lush/), no ambiente do sistema Operacional Linux.

O objetivo da utilização essa linguagem se concentra no fato da sua portabilidade integral para o hardware que será utilizado no projeto bem como integração com uma linguagem de desenvolvimento mais dinâmica e rápida e um sistema operacional compacto. As micro

motherboards utilizadas para captura e tratamento de sinais aceitam integralmente os códigos produzidos por esta linguagem.

A Figura 81 mostra a tela inicial da linguagem de programação Lush.

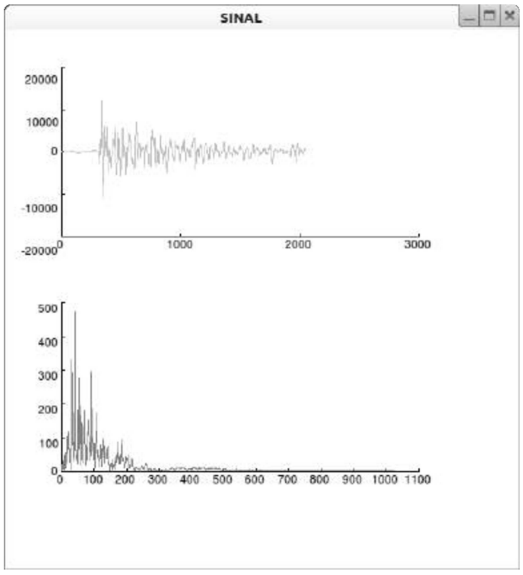
Figura 81 - Tela Inicial da Linguagem Lush 1.2.1



Fonte: Acervo Pessoal

O sinal obtido através da captura é mostrado na Figura 82 bem como a análise do power spectro deste sinal.

Figura 82 - Resultado da Programação Lush



Fonte: Acervo Pessoal

A Figura 83 é mostra a janela obtida para que seja visualizada a imagem da câmera a ser acoplada no topo da vara de manobra.

Figura 83 - Captura de Vídeo

Fonte: Acervo Pessoal

A tela final para o programa de captura e análise de dados é mostrada na Figura 84, onde é possível visualizar os botões para a captura de dados, bem como a janela da câmera da imagem todo topo da cruzeta.

Figura 84 - Tela Inicial do Programa

Fonte: Acervo Pessoal

Este programa faz a captura da batida do martelo em dois modos: Automático e Manual

No modo **Automático** o programa dispara através de porta externa o martelo eletrônico e captura o áudio da placa de som.

No modo **Manual** o programa não dispara o martelo eletrônico e somente captura o áudio da placa de som. Este processo é utilizado para a captura da batida do martelo feita manualmente por uma pessoa.

O Modo Alternado é utilizado para, em um instante ser feita a captura no modo Manual e em outro instante ser feita a captura no modo Automático. Este processo foi implementado para que seja feita uma captura da batida manualmente e em seguida uma batida do martelo eletrônico, salvando-se os dados sequencialmente para posterior análise e comparação.

O botão **Câmera** mostra a tela da câmera de vídeo para que o operador possa visualizar o estado superior da cruzeta.

Quando o botão **Capturar** é pressionado os três sinais capturados são mostrados bem como uma fotografia do estado superior da cruzeta é capturada e mostrada na janela central do programa.

4.6 – Conclusões

Neste capítulo foi possível mostrar como foi construído o hardware e sua utilização. No próximo capítulo serão mostrados os resultados obtidos e validação do algoritmo utilizado.

5 - RESULTADOS

Para esta etapa do projeto foram implementados os algoritmos para a fase de separação dos dados nas sanidades de busca: 1 Ano, Urgente e Emergencial.

É feita também um processo de validação do programa de reconhecimento das sanidades e dos microfones de captura de áudio, através de testes básicos e testes de batidas em cruzetas em laboratório. O algoritmo utilizado nesta etapa foi o KNN. Este algoritmo foi adotado para o protótipo devido a sua simplicidade de implementação e baixo consumo de processamento. O código deste algoritmo é mostrado no Apêndice B.

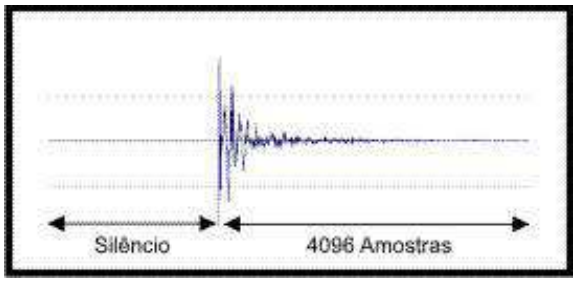
5.1 - Validação do Programa Gerado

5.1.1 - Descrição da Experiência

A experiência consiste em soltar uma chave de fenda em queda livre, a uma altura aproximada de vinte (20) centímetros, em três tipos diferentes de superfície de madeira, gravando o sinal para treinamento e posterior processamento pelo Algoritmo K-NN (K=1) para reconhecimento de três (03) grupos diferentes.

No processo de treinamento do Algoritmo KNN foram utilizadas duas (02) amostras de cada tipo de grupo. Para cada amostra foram utilizados 4096 pontos posicionados a partir do início efetivo do sinal. O início efetivo do sinal foi obtido através da extração do limiar de silêncio do início do arquivo e através do cálculo da mudança brusca de energia, ou seja, quando existe silêncio no início do arquivo a energia é pequena e quanto o sinal efetivo começa existe uma mudança brusca de valores de energia. A Figura 85 mostra como o sinal é amostrado e o que será extraída do início do mesmo.

Figura 85 - Forma de Onda do Sinal a ser processado



Fonte: Acervo Pessoal

Para o processo de reconhecimento/teste foram utilizadas as seguintes seis (06) *features* (padrões de reconhecimentos):

1 – FFT (Fast Transform Fourier) - todas as 4096 amostras foram processados e foram utilizados os 2048 primeiros coeficientes da FFT.

2 – LPC (Linear Prediction Coeffients) – foram utilizados 16 coeficientes LPC.

3 – MFCC (Mel-Frequency Cepstrum) – foram utilizados 10 coeficientes MFCC.

4 – ZERO CROSSING (Taxa de Passagem por Zero) – foram utilizadas janelas de 128 amostras para o cálculo do ZC.

5 – ENERGY (Energia do Sinal) – foram utilizadas janelas de 128 amostras para o cálculo da energia do sinal.

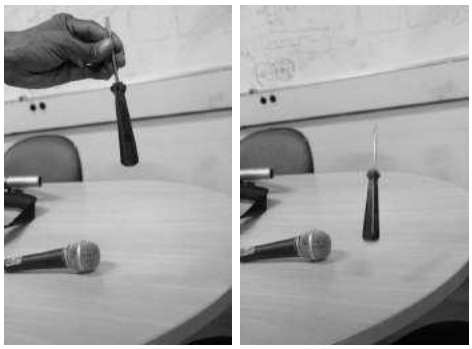
6 – AREA – foi calculada a área sob a forma de onda do sinal, sendo este sinal rebatido (normalizado) para o cálculo desta área.

No processo de reconhecimento/teste foram utilizadas todas as amostras captadas. As amostras de treinamento foram utilizadas neste momento para garantir a validação do algoritmo, pois se o algoritmo não reconhecer os mesmos sinais de treinamento ele não terá garantia de funcionamento.

5.1.2 - Experimento 1: Teste Inicial para verificação do Algoritmo K-NN

Uma chave de fenda será solta a uma altura aproximada de 20 cm em queda livre e irá colidir com a superfície de madeira. O som proveniente da batida da chave de fenda na superfície será capturado por um microfone padrão, conforme mostra a Figura 86.

Figura 86 - Experimento 1: (a) posicionamento da chave de fenda (b) chave de fenda em queda livre



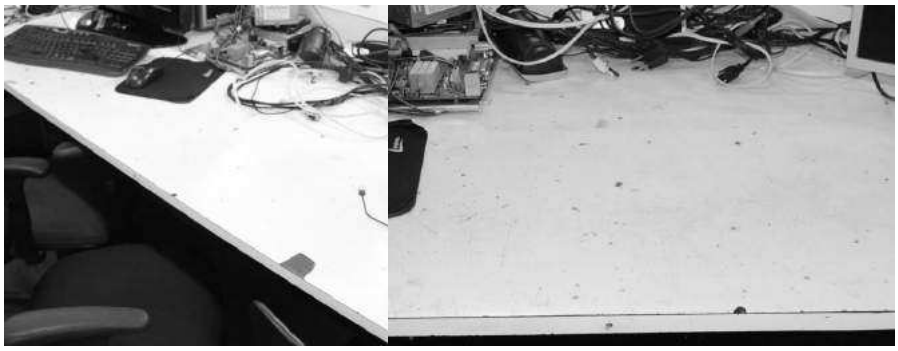
(a) (b)

Fonte: Acervo Pessoal

Para este teste foram armazenadas quatro (04) batidas da chave de fenda em três (03) tipos diferentes de superfície:

1 - Mesa de formica

Figura 87 - Mesa de Fórmica Utilizada nos Testes



Fonte: Acervo Pessoal

2 - Mesa Compensado

Figura 88 - Mesa de Compensado Utilizada nos Testes

Fonte: Acervo Pessoal

3 - Caixa Protótipo

Figura 89 - Caixa Utilizada nos Testes

Fonte: Acervo Pessoal

Para o treinamento foram utilizadas duas (02) amostras de cada conjunto de sinais captados para cada superfície. Para o reconhecimento foram utilizadas todas as amostras (04) de cada conjunto de sinais captados para cada superfície, totalizando doze (12) amostras. Os sinais de treinamento também foram utilizados no processo de teste do algoritmo como um processo de validação. A tabela 5 mostra a quantidade de acertos para cada *feature* em cada superfície. Mostra também a quantidade de acertos totais (soma de todos os acertos em todas as superfícies) por *feature*, bem como em termos de porcentagem com relação a quantidade total de *features*.

Tabela 5 - Tabela de Resultados para o Experimento 1

	FEATURE					
	FFT	LPC	MFCC	ZC	ENERGY	AREA
Quantidade de Acertos Superfície 1	4	3	4	4	4	4
Quantidade de Acertos Superfície 2	4	2	4	4	4	4
Quantidade de Acertos Superfície 3	4	3	4	3	4	4
Quantidade Total de Acertos com Todas as Amostras	12	8	12	11	12	12
Porcentagem Total de Acertos (%)	100,00	66,67	100,00	91,67	100,00	100,00

Fonte: Acervo Pessoal

5.1.3 - Experimento 2: Teste com o microfone

Neste teste foi utilizado o microfone para captação de dados, utilizado no martelo eletrônico. Neste teste uma chave de fenda é liberada em queda livre a uma altura aproximada de 20 cm em três (03) superfícies diferentes para que o microfone efetue a captura do som da batida da chave de fenda nas superfícies. Foram capturados dez (10) sinais para cada superfície. Duas (02) amostras para cada superfície foi utilizada para treinamento do algoritmo. A Figura 90 mostra como foi efetuado este teste.

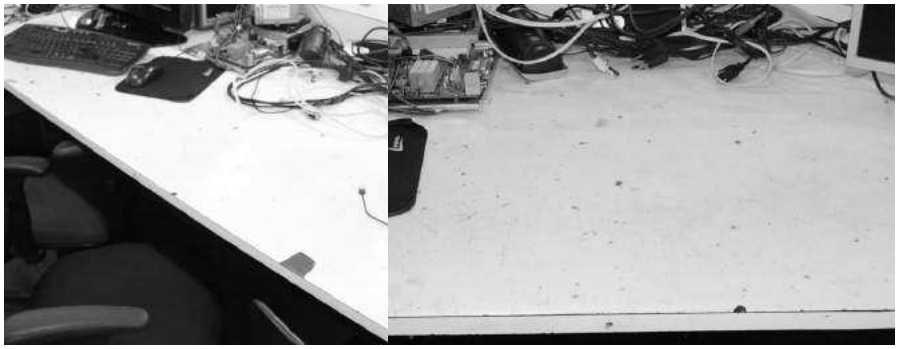
Figura 90 - Experimento 2: posicionamento da chave de fenda para captura do sinal

Fonte: Acervo Pessoal

As superfícies da experiência foram:

1 - Mesa de formica

Figura 91 - Mesa de Fórmica Utilizada nos Testes



Fonte: Acervo Pessoal

2 - Mesa Compensado

Figura 92 - Mesa de Compensado Utilizada nos Testes

Fonte: Acervo Pessoal

3 – Mesa de Computador

Figura 93 - Mesa de Computador Utilizada nos Testes

Fonte: Acervo Pessoal

Para o treinamento foram utilizadas duas (02) amostras de cada conjunto de sinais captados para cada superfície. Para o reconhecimento foram utilizadas todas as amostras (10) de cada conjunto de sinais captados para cada superfície, totalizando trinta (30) amostras. Os sinais de treinamento também foram utilizados no processo de teste do algoritmo como um processo de validação.

A tabela 6 mostra a quantidade de acertos para cada *feature* em cada superfície. Mostra também a quantidade de acertos totais (soma de todos os acertos em todas as superfícies) por *feature*, bem como em termos de porcentagem com relação a quantidade total de *features*.

Tabela 6 - Tabela de Resultados para o Experimento 2

	FEATURE					
	FFT	LPC	MFCC	ZC	ENERGY	AREA
Quantidade de Acertos Superfície 1	10	10	10	10	10	6
Quantidade de Acertos Superfície 2	10	9	10	7	10	10
Quantidade de Acertos Superfície 3	9	4	10	2	8	6
Quantidade Total de Acertos com Todas as Amostras	29	23	30	19	28	22
Porcentagem Total de Acertos (%)	96,67	76,67	100,00	63,33	93,33	73,33

Fonte: Acervo Pessoal

5.1.4 - Experimento 3: Testes com as Cruzetas

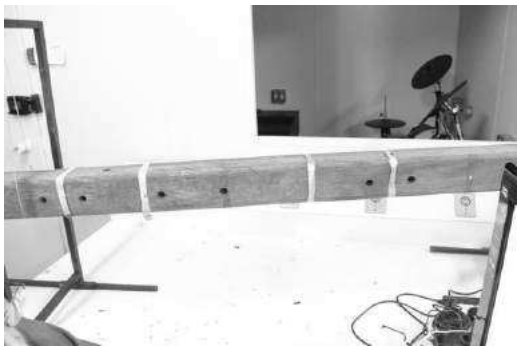
Neste experimento foram selecionadas três (03) cruzetas em laboratório que poderiam pertencer aos três (03) estados de sanidades desejadas para o projeto: 1 Ano, Urgencial e Emergencial.

Como o grupo de trabalho do laboratório não possui as especialidades necessárias para a seleção de cruzetas, foi feito uma análise visual para a seleção da mesma somente para critério de validação do algoritmo K-NN.

As cruzetas selecionadas são mostradas a seguir:

1 - Cruzeta com sanidade aparente de 1 Ano ou Mais

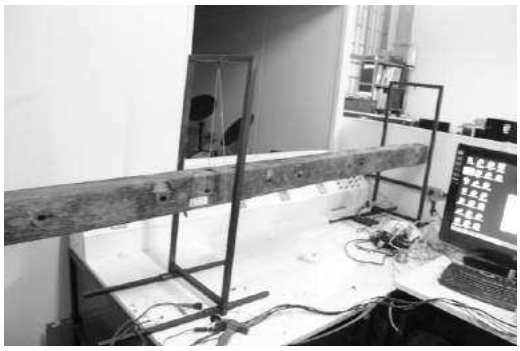
Figura 94 - Cruzeta de Sanidade 1 Ano



Fonte: Acervo Pessoal

2 - Cruzeta com sanidade aparente de 30 dias (marcada na cruzeta sanidade de 60 dias, que foi fundida com a sanidade de 30 dias pelos especialistas da Cemig)

Figura 95 - Cruzeta com Sanidade Urgencial



Fonte: Acervo Pessoal

3 - Cruzeta com sanidade aparente de Emergencial

Figura 96 - Cruzeta com Sanidade Emergencial

Fonte: Acervo Pessoal

Para este teste foi efetuada a batida de um martelo comum em um lado da cruzeta e o sinal evocado foi capturado do outro lado pelo microfone, conforme mostra a Figura 100.

Figura 97 - Posicionamento do Martelo e Microfone para Teste da Batida

Fonte: Acervo Pessoal

A batida foi efetuada na região (segmento) considerada ser a sanidade desejada para o teste. Não houve muita variação da posição devido ao fato de que uma cruzeta não possui somente uma sanidade em sua extensão podendo, em uma mesma cruzeta, existir as três possíveis sanidades procuradas.

Uma distância padrão de aproximadamente cinco (05) cm do martelo a cruzeta foi mantida para a batida, bem como foi mantida uma pressão aproximadamente igual para todas as batidas.

A tabela 7 mostra a quantidade de acertos para cada *feature* em cada superfície. Mostra também a quantidade de acertos totais (soma de todos os acertos em todas as superfícies) por *feature*, bem como em termos de porcentagem com relação a quantidade total de *features*.

Tabela 7 - Tabela Geral para o Teste das Cruzetas

	FEATURE					
	FFT	LPC	MFCC	ZC	ENERGY	AREA
Quantidade de Acertos Superfície 1	10	4	6	4	5	4
Quantidade de Acertos Superfície 2	10	6	5	9	8	10
Quantidade de Acertos Superfície 3	10	5	8	4	10	10
Quantidade Total de Acertos com Todas as Amostras	30	15	19	17	23	24
Porcentagem Total de Acertos (%)	100,00	50,00	63,33	56,67	76,67	80,00

Fonte: Acervo Pessoal

5.1.6 – Conclusões sobre a Validação do Algoritmo

Através dos Experimentos 1 e 2 foi possível confirmar que o Algoritmo K-NN, com K=1, está funcionando corretamente para o reconhecimento de todas as superfícies de testes, com pequenas variações com relação a escolha da *feature* de teste.

Para o experimento das cruzetas também foi possível notar um alto valor de acerto no reconhecimento das cruzetas principalmente para a *feature* FFT, que obteve 100% de acerto em todas as sanidades.

Para as outras *features* foram encontrados muitos erros, mas que podem ser justificados devido ao fato de que as cruzetas e as regiões (segmentos) escolhidas para testes não corresponderem exatamente a sanidade certa, pois as pessoas que fizeram as escolhas não são os especialistas que irão selecionar os segmentos para validação e treinamento do algoritmo.

5.2 - Resultados das Análises dos Sinais Capturados na CEMIG D

Os dados foram obtidos em conformidade com o protocolo utilizado. As madeiras foram separadas nas dependências da CEMIG D., conforme mostram as Figura 101 e 102.

Figura 98 - (a) Banco de Amostras (b) Pontos de

(a) (b)

Fonte: Acervo Pessoal

Figura 99 – (a) Avaliação e (b) Medição de Cruzetas

(c) (d)

Fonte: Acervo Pessoal

Nos trabalhos iniciais nas dependências da CEMIG D foram obtidas 90 amostras para cada estado de sanidade, sendo que destas 90 amostras 30 amostras foram obtidas por cada uns dos especialistas para serem utilizadas no protocolo de validação de dados, proposto neste trabalho. Com a utilização do protocolo de validação dos dados, através do confronto de dados com os três (03) especialistas, foram eliminadas 12 amostras para a sanidade 1 ANO, 12 amostras para a sanidade URGENCIAL E 13 amostras para a sanidade EMERGENCIAL, resultando em 78 amostras para a sanidade 1 ANO, 78 amostras para a sanidade URGENCIAL e 77 amostras para a sanidade EMERGENCIAL. A Tabela 8 mostra estes resultados.

Tabela 8 - Dados Resultantes da Utilização do Protocolo de Validação dos Dados

Sanidade	Amostras Obtidas	Amostras Eliminadas	Amostras Restantes
1 ANO	90	12	78
URGENCIAL	90	12	78
EMERGENCIAL	90	13	77
TOTAL	270	37	233

Fonte: Acervo Particular

Para cada grupo de sanidade as amostras foram normalizadas pela média do grupo e agrupadas em um histograma para determinar o 1º e 3º Quartil. Também foi determinada a mediana para cada grupo de sanidade. As Tabelas 9, 10 e 11 mostram a distribuição estatística para as três sanidades propostas.

Tabela 9 - Distribuição para Sanidade 1 Ano

i	Range		Quantidade fi	Ponto Médio	Freq. Acumulada F	(xi.fi)	(xi2.fi)	
1	-2,9	-2,1	3	0,4	3	1,200	0,480	
2	-2,1	-1,3	4	0,4	7	2,800	1,120	
3	-1,3	-0,5	4	0,4	11	4,400	1,760	
4	-0,5	0,3	49	0,4	60	24,000	9,600	Q1
Q5	0,3	1,1	13	0,4	73	29,200	11,680	Q3
6	1,1	1,9	3	0,4	76	30,400	12,160	
7	1,9	2,7	0	0,4	76	30,400	12,160	
8	2,7	3,5	1	0,4	77	30,800	12,320	
9	3,5	4,3	0	0,4	77	30,800	12,320	
10	4,3	5,1	1	0,4	78	31,200	12,480	

Fonte: Acervo Pessoal

Tabela 10 - Distribuição para Sanidade Urgencial

i	Range		Quantidade fi	Ponto Médio	Freq. Acumulada F	(xi.fi)	(xi2.fi)	
1	-1,0	-0,2	40	0,4	40	16,000	6,400	Q1
2	-0,2	0,6	27	0,4	67	26,800	10,720	Q3
3	0,6	1,4	6	0,4	73	29,200	11,680	
4	1,4	2,2	2	0,4	75	30,000	12,000	
5	2,2	3,0	0	0,4	75	30,000	12,000	
6	3,0	3,8	2	0,4	77	30,800	12,320	
7	3,8	4,6	0	0,4	77	30,800	12,320	
8	4,6	5,4	0	0,4	77	30,800	12,320	
9	5,4	6,2	1	0,4	78	31,200	12,480	

Fonte: Acervo Pessoal

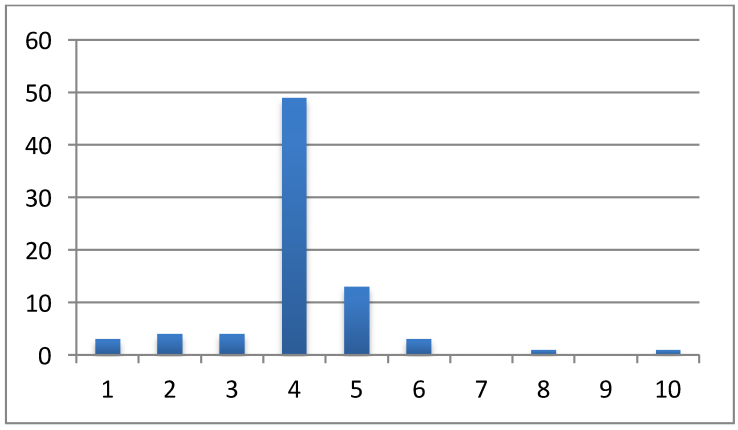
Tabela 11 - Distribuição para Sanidade Emergencial

i	Range		Quantidade fi	Ponto Médio	Freq. Acumulada F	(xi.fi)	(xi2.fi)	
1	-3,4	-2,7	2	0,3665	2	0,733	0,269	
2	-2,7	-1,9	2	0,3665	4	1,466	0,537	
3	-1,9	-1,2	2	0,3665	6	2,199	0,806	
4	-1,2	-0,5	8	0,3665	14	5,131	1,881	
5	-0,5	0,3	47	0,3665	61	22,357	8,194	Q1
6	0,3	1,0	8	0,3665	69	25,289	9,268	Q3
7	1,0	1,7	3	0,3665	72	26,388	9,671	
8	1,7	2,5	3	0,3665	75	27,488	10,074	
9	2,5	3,2	2	0,3665	77	28,221	10,343	

Fonte: Acervo Pessoal

Para a sanidade de 1 ANO temos Q1 = -0.5, Q2 = Mediana = 0 e Q3 = 0,2. Desta forma as amostras que possuem médias abaixo de Q1 e acima de Q3 serão eliminadas por serem consideradas. Foram eliminadas 10 amostras para este caso. O histograma para esta distribuição é mostrado na Figura 103.

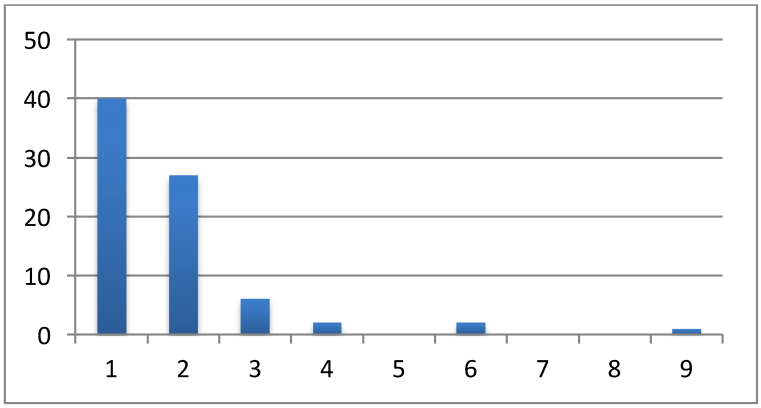
Figura 100 - Histograma para a Sanidade 1 ANO



Fonte: Acervo Pessoal

Para a sanidade URGENCIAL temos Q1 = - 0.5, Q2 = Mediana =0 e Q3 = 0,3. Desta forma as amostras que possuem médias abaixo de Q1 e acima de Q3 serão eliminadas por serem consideradas como *outliers*. Foram eliminadas 8 amostras para este caso. O histograma para esta distribuição é mostrado na Figura 104.

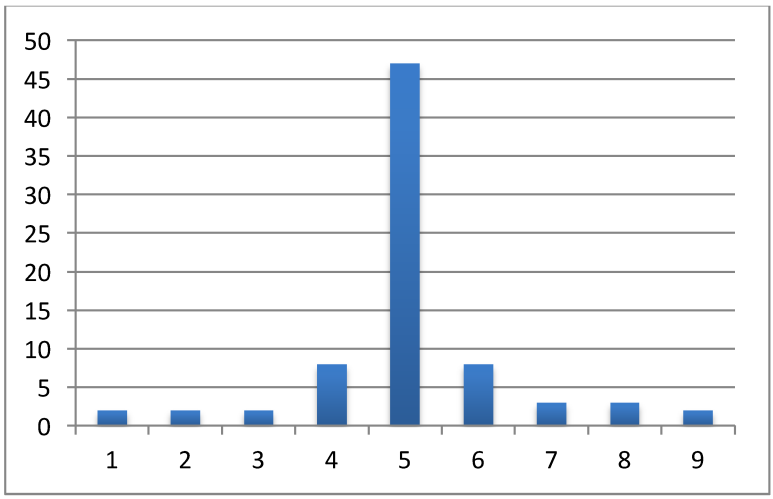
Figura 101 - Histograma para a Sanidade URGENCIAL



Fonte: Acervo Pessoal

Para a sanidade EMERGENCIAL temos Q1 = -0.4, Q2 = Mediana = 0 e Q3 = 0,2. Desta forma as amostras que possuem médias abaixo de Q1 e acima de Q3 serão eliminadas por serem consideradas como *outliers*.

Figura 102 - Histograma para a Sanidade EMERGENCIAL



Fonte: Acervo Pessoal

As amostras que possuem maior energia são as que estão em torno da mediana, que são os pontos de pico dos histogramas das figuras vistas anteriormente. Desta forma foram selecionadas 16 amostras de cada sanidade para serem utilizadas como referência para o algoritmo K-NN. A Tabela 12 mostra estes resultados.

Tabela 12 - Resultados para as Amostras Finais

Sanidade	Amostras Obtidas	Amostras Eliminadas	Amostras Restantes	Amostras de Treinamento	Amostras de Teste
1 ANO	78	10	68	16	52
URGENCIAL	78	8	70	16	54
EMERGENCIAL	77	12	65	16	49

Fonte: Acervo Pessoal

Após estas análises as amostras foram processadas através do algoritmo K-NN, para K=1 para a geração do vetor de *features* para referência do algoritmo, com o uso de 48 amostras para esta geração. Para o teste e/ou resultados foram utilizadas todas as amostras, ou seja, amostras de treinamento mais amostras de teste).

A Tabela 13 mostra os índices de acertos para o reconhecimento das sanidades propostas.

Tabela 13 - Índice de Acertos

	ACERTOS	
FFT	74,38	%
LPC	62,56	%
MFCC	**81,77**	**%**
ZERO CROSSING	55,67	%
ENERGY	56,16	%
AREA	47,78	%
FFT-LPC	74,38	%
FFT-MFCC	74,38	%
LPC-MFCC	75,86	%
FFT-LPC-MFCC	74,38	%

Fonte: Acervo Pessoal

Através da Tabela 13 pode-se observar que as *features* obtidas através dos coeficientes MFCC foram as que obtiveram melhores resultados, conforme já era previsto através da validação do algoritmo, mostrado nas Tabelas 5, 6 e 7. Estas tabelas já indicavam que as *features* MFCC sempre convergiam para 100% de acertos. Somente na Tabela 8 é que não convergiu, justificado pelo fato de que os pontos capturadas das cruzetas não foram selecionados por especialistas da CEMIG D.

Para o protótipo final a ser entregue estas features (MFCC) serão as utilizadas para o reconhecimento em campo, podendo também, via software, ser selecionada outras features caso o analista/especialista em campo deseje fazer comparações.

5.3 – Finalização e Entrega do Projeto Final a CEMIG D

O protótipo final foi apresentado a comunidade através do XX Seminário Nacional de Distribuição de Energia Elétrica, realizado no período de 08 a 13 de Novembro de 2014, em Santos-SP, especificamente no dia 12 de Novembro de 2014, conforme mostra a Figura 106.

Figura 103 - Comprovante de Participação no XX SENDI

PROGRAMAÇÃO DO DIA
12/11/2014 - Quarta-Feira

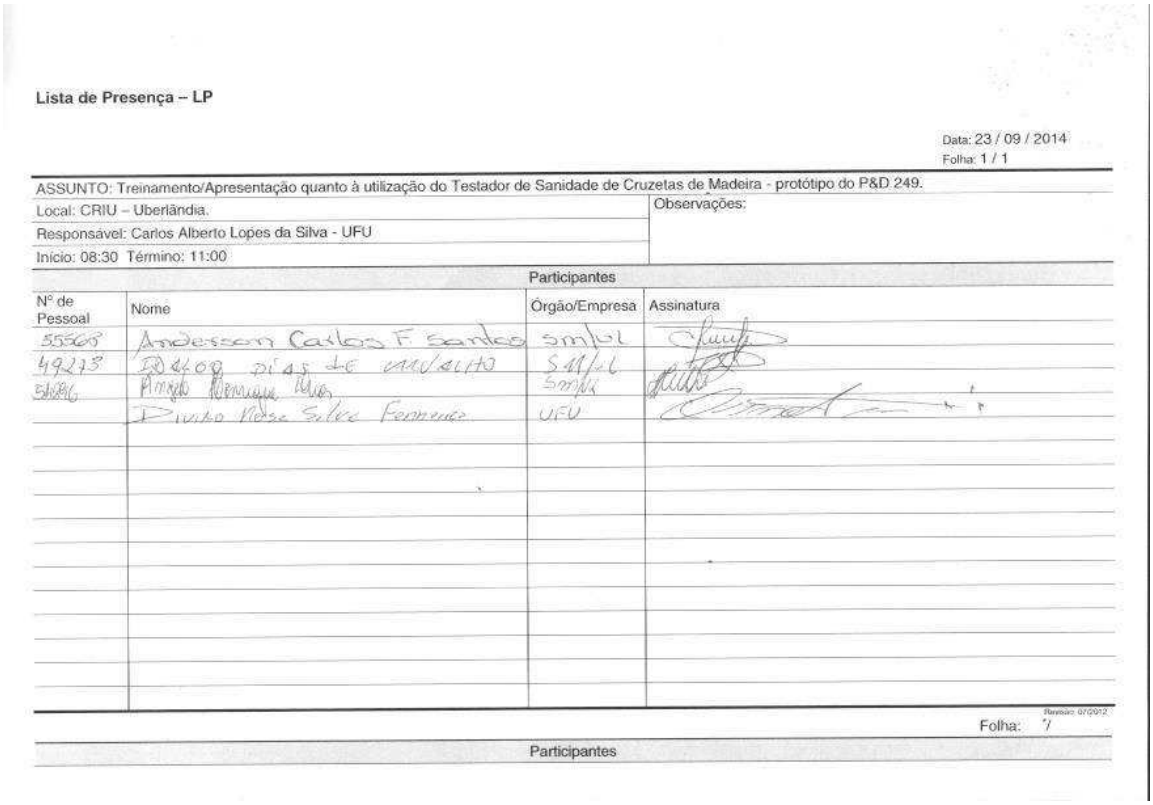
Sala Terra – 1º. Piso		
Tema:	Redes de Distribuição até 34,5 kV	
Horário: 13:30 às 15:30		
Presidente:	William Fernandes	AES ELETROPAULO
Relator:	Fernando Vargas Baldotto	EDP BANDEIRANTE
Horário: 16:00 às 18:00		
Presidente:	William Fernandes	AES ELETROPAULO
Relator:	Fernando Vargas Baldotto	EDP BANDEIRANTE

HORÁRIO	ID	ÁREA	TEMA	TÍTULO	AUTORES	EMPRESAS
13:30 às 13:55	3261	Técnico	Redes de distribuição até 34,5 kV	Procedimento Inteligente para Manutenção do Sistema de Iluminação Pública Fazendo Uso de Dispositivos Dedicados	Cláudia Maria Coimbra | Luiz Fernando Vansan | Walter Pinheiro | Gil Fortes Vasconcellos | Sergio F. de Angelo Filho | Ednilson Jose Menatti	CPFL PIRATININGA| CPFL | MATRIX | PLAYMUSIC
14:00 às 14:25	2382	Técnico	Redes de distribuição até 34,5 kV	Compensação reativa de rede distribuição ferroviária	Alexandre Lima de Carvalho | Leandro Ramos de Araujo	MRS LOGÍSTICA SA | UFJF
14:30 às 14:55	2464	Técnico	Redes de distribuição até 34,5 kV	Nova metodologia para inspeção e análise de sanidade de cruzetas de madeira.	Luciano Magno da Silva | Luciano Vieira Lima | Jonmil Marques Borges | Edelcio Antonio Martins	CEMIG D | UFU
15:00 às 15:25	2776	Técnico	Redes de distribuição até 34,5 kV	Estudo Sobre Aplicação de Reguladoras de Tensão Tipo A com Fluxo Inverso para Atender Flexibilidade ao Sistema Elétrico.	Edson Sena dos Santos | Caroline Tarangua de Mattos	ENERSUL
15:30 às 15:55	Intervalo de relacionamento					
16:00 às 16:25	2628	Técnico	Redes de distribuição até 34,5 kV	Determinação da capacidade de condução de corrente do sistema subterrâneo com modelagem do circuito térmico dos cabos.	José Henrique	CEB
16:30 às 16:55	3285	Técnico	Redes de distribuição até 34,5 kV	Novo Big Jampa - Mais Segurança, Maior Produtividade, Menor Custo e Otimização dos Serviços Operacionais de Preparação	Laudemir A. Carita Junior | Alecsandro Costa | Altino Silva | Carlos Nunes | Artur Braga Pastorelli	ELEKTRO
17:00 às 17:25	2554	Técnico	Redes de distribuição até 34,5 kV	Desenvolvimento de Metodologia e Ferramentas Para Substituição de Cabos de Baixa Tensão com Técnica de Linha Viva	Victor Salvino Borges | Dailton Pedreira Cerqueira | Kátia Cilene Falcão Xavier | Marilda Munaro | Mário Antonio Duarte Bomfim | Florami Lemos de Sá | André Luis Müller da Silva | Marcelo Antônio Ravaglio | Isaú de Oliveira Nascimento | Guilherme Rachelle Hernaski | Rafael Piros Machado	COELBA | FEERGS | LACTEC
17:30 às 17:55	2494	Técnico	Redes de distribuição até 34,5 kV	Desenvolvimento de Metodologia e Ferramental para Realização de Serviço em Linha Viva Noturna	Guilherme Rachelle Hernaski | Erival Nascimento de Almeida | Dailton Pedreira Cerqueira | Kátia Cilene Falcão Xavier | Edemir Luiz kowalski | Victor Salvino Borges | Marcelo Antônio Ravaglio | Mario Pinheiro | Rafael Piros Machado | Isaú de Oliveira Nascimento | Jair Novais Alves | Mario José Costa Pinheiro	COELBA | LACTEC

Fonte: XX SENDI

O protótipo final foi entregue a CEMIG D, apresentados aos diretores e com treinamento dos especialistas para a utilização do mesmo. A Figura 106 comprova o treinamento dos especialistas.

Figura 104 - Comprovante de Treinamento



Lista de Presença – LP

Data: 23 / 09 / 2014
Folha: 1 / 1

ASSUNTO: Treinamento/Apresentação quanto à utilização do Testador de Sanidade de Cruzetas de Madeira - protótipo do P&D 249.

Local: CRIU – Uberlândia.

Observações:

Responsável: Carlos Alberto Lopes da Silva - UFU

Início: 08:30 Término: 11:00

Participantes

Nº de Pessoal	Nome	Órgão/Empresa	Assinatura
55568	Anderson Carlos F Santos	SMUL	
49213	[illegible]	SMUL	
51086	[illegible]	SMUL	
	[illegible]	UFU	

Folha: 1/1

Participantes

Fonte: Acervo Pessoal

Após finalização do projeto a empresa CEMIG D iniciou o processo de pedido de patente em 06/05/2015, satisfazendo todas as etapas iniciais, com reconhecimento do autor deste trabalho como um dos integrantes da equipe de desenvolvimento. Os comprovantes de pedido e depósito de patente são mostrados nas Figuras 108 e 109, estando até a data de apresentação deste trabalho na fase de sigilo.

Figura 105 - Pedido de Patente

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DO DESENVOLVIMENTO, INDÚSTRIA E COMÉRCIO EXTERIOR
INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL

EXAME PRELIMINAR

N.° do Pedido: BR102015010302-6 **N.° de Depósito PCT:**
Data de Depósito: 06/05/2015

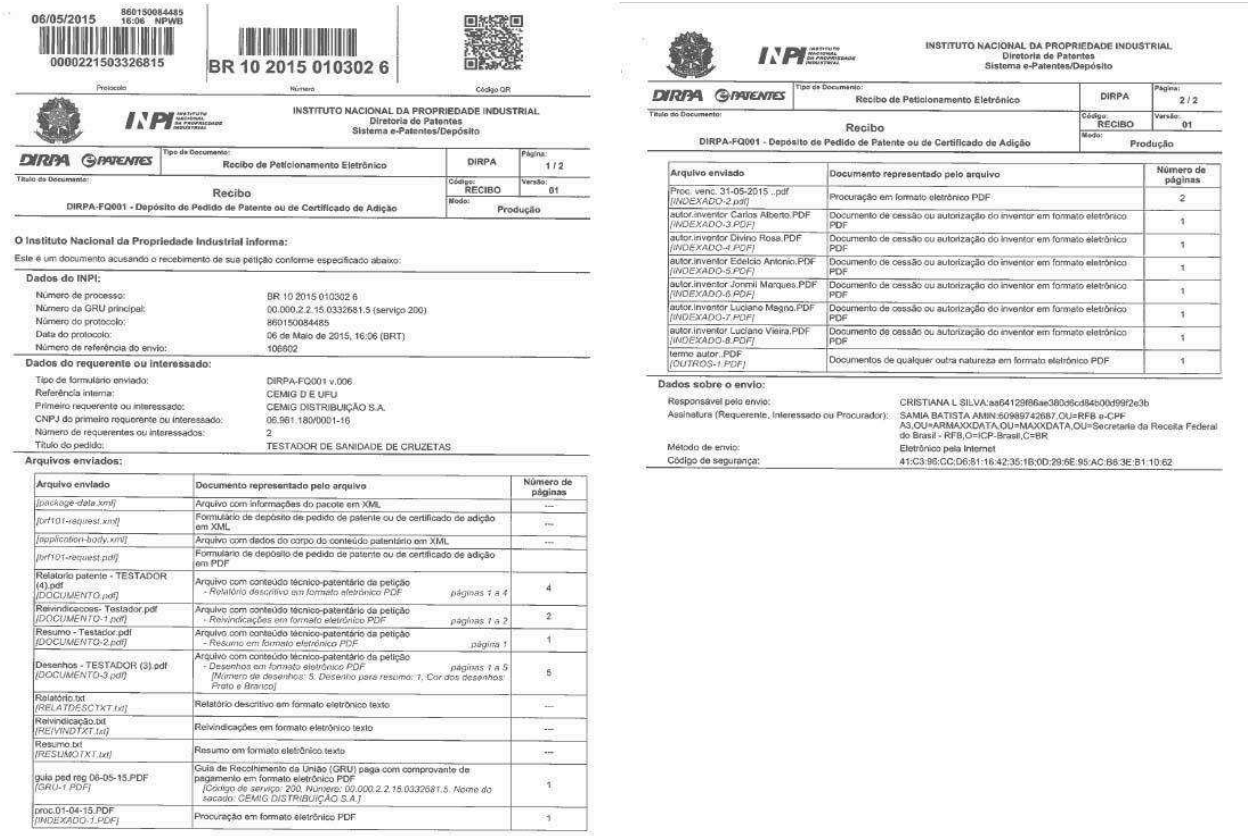
O pedido atende formalmente as disposições legais, especialmente quanto ao Art. 19 da LPI e o Instrução Normativa nº 31/2013, estando apto a ser protocolado.

Condições do Pedido	S	N
Requerimento de depósito com os campos obrigatórios preenchidos	X	
Idioma Português	X	
Relatório Descritivo	X	
Reivindicações	X	
PI e C – Apresenta desenhos citados ou não cita nem apresenta desenhos. MU – Apresenta desenhos.	X	
Resumo	X	
Formatado no padrão exigido	X	
Valor correto de Recolhimento	X	

Rio de Janeiro, 16 de junho de 2015.

Andrea Massad Fonseca Barbosa
Mat. Nº 1466814
DIRPA / COSAP/SEFOR

Figura 106 – INPI – www.inpi.gov.br



06/05/2015 860150084485 16:06 NPWB
0000221503326815
BR 10 2015 010302 6
Protocolo Número Código QR

INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL
Diretoria de Patentes
Sistema e-Patentes/Depósito

DIRPA e-PATENTES | Tipo de Documento: Recibo de Peticionamento Eletrônico | DIRPA | Página: 1 / 2
Título do Documento: Recibo | Código: RECIBO | Versão: 01
DIRPA-FQ001 - Depósito de Pedido de Patente ou de Certificado de Adição | Modo: Produção

O Instituto Nacional da Propriedade Industrial informa:

Este é um documento acusando o recebimento de sua petição conforme especificado abaixo:

Dados do INPI:

Número de processo:	BR 10 2015 010302 6
Número da GRU principal:	00.000.2.2.15.0332681.5 (serviço 200)
Número do protocolo:	860150084485
Data do protocolo:	06 de Maio de 2015, 16:06 (BRT)
Número de referência do envio:	106602

Dados do requerente ou interessado:

Tipo de formulário enviado:	DIRPA-FQ001 v.006
Referência interna:	CEMIG D E UFU
Primeiro requerente ou interessado:	CEMIG DISTRIBUIÇÃO S.A.
CNPJ do primeiro requerente ou interessado:	06.981.180/0001-16
Número de requerentes ou interessados:	2
Título do pedido:	TESTADOR DE SANIDADE DE CRUZETAS

Arquivos enviados:

Arquivo enviado	Documento representado pelo arquivo	Número de páginas
[package-data.xml]	Arquivo com informações do pacote em XML	---
[brf101-request.xml]	Formulário de depósito de pedido de patente ou de certificado de adição em XML	---
[application-body.xml]	Arquivo com dados do corpo do conteúdo patentário em XML	---
[brf101-request.pdf]	Formulário de depósito de pedido de patente ou de certificado de adição em PDF	
Relatorio patente - TESTADOR (4).pdf *[DOCUMENTO.pdf]*	Arquivo com conteúdo técnico-patentário da petição - *Relatório descritivo em formato eletrônico PDF* *páginas 1 a 4*	4
Reivindicacoes- Testador.pdf *[DOCUMENTO-1.pdf]*	Arquivo com conteúdo técnico-patentário da petição - *Reivindicações em formato eletrônico PDF* *páginas 1 a 2*	2
Resumo - Testador.pdf *[DOCUMENTO-2.pdf]*	Arquivo com conteúdo técnico-patentário da petição - *Resumo em formato eletrônico PDF* *página 1*	1
Desenhos - TESTADOR (3).pdf *[DOCUMENTO-3.pdf]*	Arquivo com conteúdo técnico-patentário da petição - *Desenhos em formato eletrônico PDF* *páginas 1 a 5* *[Número de desenhos: 5, Desenho para resumo: 1, Cor dos desenhos: Preto e Branco]*	5
Relatório.txt *[RELATDESCTXT.txt]*	Relatório descritivo em formato eletrônico texto	---
Reivindicação.txt *[REIVINDTXT.txt]*	Reivindicações em formato eletrônico texto	---
Resumo.txt *[RESUMOTXT.txt]*	Resumo em formato eletrônico texto	---
guia ped reg 06-05-15.PDF *[GRU-1.PDF]*	Guia de Recolhimento da União (GRU) paga com comprovante de pagamento em formato eletrônico PDF *[Código de serviço: 200, Número: 00.000.2.2.15.0332681.5, Nome do sacado: CEMIG DISTRIBUIÇÃO S.A.]*	1
proc.01-04-15.PDF *[INDEXADO-1.PDF]*	Procuração em formato eletrônico PDF	1

INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL
Diretoria de Patentes
Sistema e-Patentes/Depósito

DIRPA e-PATENTES | Tipo de Documento: Recibo de Peticionamento Eletrônico | DIRPA | Página: 2 / 2
Título do Documento: Recibo | Código: RECIBO | Versão: 01
DIRPA-FQ001 - Depósito de Pedido de Patente ou de Certificado de Adição | Modo: Produção

Arquivo enviado	Documento representado pelo arquivo	Número de páginas
Proc. venc. 31-05-2015 ..pdf *[INDEXADO-2.pdf]*	Procuração em formato eletrônico PDF	2
autor.inventor Carlos Alberto.PDF *[INDEXADO-3.PDF]*	Documento de cessão ou autorização do inventor em formato eletrônico PDF	1
autor.inventor Divino Rosa.PDF *[INDEXADO-4.PDF]*	Documento de cessão ou autorização do inventor em formato eletrônico PDF	1
autor.inventor Edelcio Antonio.PDF *[INDEXADO-5.PDF]*	Documento de cessão ou autorização do inventor em formato eletrônico PDF	1
autor.inventor Jonmil Marques.PDF *[INDEXADO-6.PDF]*	Documento de cessão ou autorização do inventor em formato eletrônico PDF	1
autor.inventor Luciano Magno.PDF *[INDEXADO-7.PDF]*	Documento de cessão ou autorização do inventor em formato eletrônico PDF	1
autor.inventor Luciano Vieira.PDF *[INDEXADO-8.PDF]*	Documento de cessão ou autorização do inventor em formato eletrônico PDF	1
termo autor .PDF *[OUTROS-1.PDF]*	Documentos de qualquer outra natureza em formato eletrônico PDF	1

Dados sobre o envio:

Responsável pelo envio:	CRISTIANA L SILVA:aa64129f86ae380d6cd84b00d99f2e3b
Assinatura (Requerente, Interessado ou Procurador):	SAMIA BATISTA AMIN:60989742687,OU=RFB e-CPF A3,OU=ARMAXXDATA,OU=MAXXDATA,OU=Secretaria da Receita Federal do Brasil - RFB,O=ICP-Brasil,C=BR
Método de envio:	Eletrônico pela Internet
Código de segurança:	41:C3:96:GC:D6:81:16:42:35:1B:0D:29:6E:95:AC:86:3E:B1:10:62

Fonte: INPI – www.inpi.gov.br

6 - CONCLUSÕES

Conforme já observado o sistema é eficiente e ainda não foi desenvolvido nenhum hardware que ofereça as características necessárias para satisfazer as necessidades da CEMIG D.

Os estudos já desenvolvidos e em desenvolvimento necessitam de um maior contato com a cruzeta em teste, inviabilizando os projetos neste formato, pois o que a CEMIG D busca é resolver problemas de desligamentos do fornecimento elétrico em áreas com indústrias e população que necessitam deste fornecimento constantemente.

Com a validação dos dados através do protocolo proposto pode-se obter amostras com energia suficiente para o treinamento do sistema, gerando resultados satisfatórios para o que foi proposto no projeto em conjunto com a CEMIG D.

Um dos grandes problemas enfrentados em todo o trabalho foi separar de forma eficiente os três graus de sanidades, que eram necessários para a classificação das cruzetas (1 ANO, URGENCIAL e EMERGENCIAL). O maior grau de dificuldade foi encontrado no grau URGENCIAL pois o mesmo é uma classificação bem próxima da classificação EMERGENCIAL. Isto dificultou muito as análises e diminuindo o nível de acerto do sistema. Uma solução seria separar o sistema em somente dois níveis de sanidade para se ter uma maior eficiência: BOAS e RUINS.

- Cruzetas BOAS: poderiam ficar nos postes e terem uma nova avaliação, em data prevista.
- Cruzetas RUINS: deveriam serem trocadas imediatamente. Isto considerando que o grande problema está na troca de cruzetas com tempo de vida útil boa e são trocadas sem necessidade e cruzetas que quebram por serem analisadas, visualmente pelos analistas, como sendo boas.

Outros tipos de classificação também podem ser inseridos no sistema para comparação ou mesmo para serem os critérios de análise. Podemos citar o sistema de reconhecimento de imagens (BGLearning) visto em (LeCun, Bottou, Bengio, & Haffner, 1998) onde foi obtido o reconhecimento de padrões de imagens com elevado índice de acerto (< 0.7%), para imagens de tamanho extremamente reduzido, 32 x 32 pixel, com um número de 60.000 amostra de treinamento e 10.000 amostras de testes, sendo todas diferente. Transformar um sistema de áudio em uma representação de imagens poderá ser uma boa opção. Neste caso o sinal é percorrido gerando uma imagem matricial de 32 x 32 pixel (tamanho este necessário

para compor as 1024 amostras de uma FFT, por exemplo), onde cada ponto desta matriz seria um ponto da feature desejada convertidos em um range de escalas de tons de cinzas (0-255). Para efeitos de testes iniciais e proposições futuras efetuamos o experimento com o algoritimo GBLearning, citado acima, onde todas as funções são disponibilizadas no sie do autor (http://www.cs.nyu.edu/~yann/datasets/index.html). Neste experimento foram efetuados os seguintes passos:

- Gerar um vetor com os Coeficientes FFT
- Gerar um vetor com os Coeficientes referente a forma de onda do sinal
- Gerar um vetor com os Coeficientes do Espectro de Potência
- Transformar os vetores acima gerados em uma imagem de 32x32 pixel (1024 amostras onde cada pixel desta imagem é um ponto do vetor)
- Definir um erro de treinamento para o sistema menor do que 1%
- Executar o treinamento com todas as amostras válidas
- Executar o treinamento com todas as amostras possíveis (válidas e amostras excluídas por qualquer motivo)

Os resultados obtidos para este teste simples foram:

Resultados (% acertos) com Amostras Válidas (48 Treinamento – 155 Teste):

- FFT: 76,85%
- Forma de Onda: 60,10%
- Spectro de Potência: **78,82%**

Resultados (% acertos) com Todas Amostras Possíveis (528 Treinamento – 284 Teste):

- FFT: **89,58%**
- Forma de Onda: 78,12%
- Spectro de Potência: 81,25%

Mesmo com todas as amostras possíveis (considerandos amostras descartadas estatisticamente) o índice de acerto foi muito bom (maior que 89%). Vale considerar que para este tipo de algoritimo é extremamente importante uma quantidade muito grande de

amostras de treinamento. Neste caso específico, comparado ao proposto no artigo (quantidade de 60.000 amostras) os resultados já são satisfatórios. O próprio sistema se encarregaria de ir somando amostras a medida que testes de campos forem sendo feitos, tornando o sistema ainda mais preciso a medido que o aumento de amostras chegassem no valor proposto no artigo.

A opção para o algoritmo K-NN deu-se somente pela simplicidade de implementação, velocidade de processamento, pouco uso de memória e hardware, pois o protótipo inicial seria utilizado com um minicomputador, onde não teria um hardware totalmente dedicado, com um sistema operacional reduzido. Para um equipamento final todo o sistema deverá ser encapsulado em hardware dedicado e, nestas condições, outras opções de algoritmo poderão ser incluídas para terem-se outras opções de análise.

Em conformidade com o que foi mostrado no ítem 5.3, o protótipo já foi apresentado e entregue para a CEMIG D, com treinamento do pessoal, apresentação em seminário nacional e geração de pedido de patente, atendendo a todas as especificações necessárias. Como o processo de patente é lento e moroso, no futuro teremos um equipamento totalmente inédito para ser utilizado no dia-a-dia, minimizando os problemas atuais que a CEMIG D possui, tanto em redução de gastos financeiros como com redução de acidentes causados por quebras de cruzetas.

Com a miniaturização de componentes e circuitos com softwares dedicados o protótipo gerado poderá ter ainda mais suas dimensões e pesos reduzidos. Neste protótipo é necessário um sistema de alimentação robusto, o que poderá ser eliminado no futuro, com células de carga solar, sistemas de transmissão sem fio para a captura dos sinais do martelo eletrônico e todas as vantagens que a tecnologia poderá permitir.

Comprovamos, através deste trabalho, a possibilidade e viabilidade do reconhecimento do estado de sanidade de cruzetas de madeira fixadas em postes de transmissão de energia elétrica.

REFERÊNCIAS BIBLIOGRÁFICAS

Associação Brasileira de Normas Técnicas, N. 8458. (1984a). Cruzetas de Madeira para Redes de Distribuição de Energia Elétrica.

Associação Brasileira de Normas Técnicas, N. 8459. (1984b). Cruzetas de Madeira.

Baker, K. (2013). *Singular value decomposition tutorial*. *...Value Decomposition Tutorial. pdf* (Vol. 2005).

Bradbury, J. (2000). Linear predictive coding. *Florida Institute of Technology*.

Cinnéide, A. (2008). Linear Prediction The Technique, Its Solution and Application to Speech.

CODI. (1992). Padronização e Especificação de Cruzatas. Salvador: COELBA.

Correa, S. M. B. B. (2003). *Probabilidade e Estatística* (2nd ed.). Belo Horizonte: PUC Minas Virtual.

CPFL, Unicamp, & Tecnológicas, A. S. (2012). Inspeção e Classificação de Postes e Cruzetas por Meio de Ultrassom Inteligente. *P&D Resvista Pesquisa E Desenvolvimento Da ANEEL*, *4*, 90.

Dalfré, G. M. (2007). Cruzetas De Polímeros Reciclados : Caracterização Dos Materiais, Análise Numerica E Ensaios De Modelos Reduzidos.

Davis S. and Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Signal Processing*, *28*(4), 357 – 366. http://doi.org/10.1109/TASSP.1980.1163420

El-naggar, K. M. (2011). A Dynamic Filter for Removal DC - Offset In Current and Voltage Waveforms, *3*, 389–393.

Faceli, K., Lorena, A. C., Gama, J., & Carvalho, A. (2011). *Inteligência Artificial – Uma Abordagem de Aprendizado de Máquina*. (LTC, Ed.).

Ganchev, T., Ganchev, T., Fakotakis, N., Fakotakis, N., Kokkinakis, G., & Kokkinakis, G. (2005). Comparative evaluation of various MFCC implementations on the speaker verification task. *In Proc. of the SPECOM-2005*, *1*(3), 191–194.

Gerhard, D. (2003). Audio Signal Classification : History and Current Techniques. *Department of Computer Science University of Regina Regina, Saskatchewan, CANADA*, 0–37.

Kedem, B. (1986). Spectral Analysis and Discrimination by Zero-Crossings. *Proceedings of the IEEE*, *74*(11), 1477–1493.

Khan, A., & Farooq, H. (2012). Principal Component Analysis-Linear Discriminant Analysis Feature Extractor for Pattern Recognition. *arXiv Preprint arXiv:1204.1177*, *8*(6), 267–270.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2323.

Lepage, E. S. (1986). *Manual de Preservação de Madeiras*. São Paulo: IPT São Paulo.

Machado, A. C., & Lima, L. V. (2007). *Sound Forge 5.0 - Restauração de Sons de LPs e Gravação de CDs*. Editora Érica.

Medri, W. (2011). Análise Exploratória de Dados. *Universidade Estadual de Londrina - Centro de Ciências Exatas - Departamento de Estatística*, 82.

Muniz, D. N. (2009). Estudo Sobre Reconhecimento de Áudio Repetitivo: Devolvimento de um Protótipo. *Instituto Federal de Santa Catarina*.

Shanmugapriya, D. (2010). Acoustic Signal based Feature extraction for Vehicular Classification. *Information Fusion*, 11–14.

Stojanovi, V. (2006). Fast Fourier Transform : Theory and Algorithms Discrete Fourier Transform – A review.

Vidor, F. L. R. (2003). Avaliação de Processos de Inspeção e Retratamento de Postes de Madeira.

Apêndice A - Laboratório de Campo

De posse de todas as informações acima citadas, foi necessário, então, a elaboração e a montagem de um laboratório de campo para a análise dos diversos estados de sanidade de cruzetas. A necessidade de se montar um laboratório de campo leva em consideração alguns aspectos:

a) **Facilidade de acesso:** para testes será necessário sempre estar colhendo novas amostras de sinais. O laboratório de campo torna mais fácil e rápida esta coleta de informações.

b) **Lugar Fechado:** Equipamentos estarão armazenando informações em tempo real do estado das cruzetas e, portanto, não devem ser manipulados por pessoal não autorizado no projeto. O Laboratório de Campo deverá ser fechado e somente pessoas que participem do projeto poderão ter acesso ao mesmo.

c) **Uso Intermitente:** Este Laboratório de Campo irá facilitar testes em todos os períodos do dia (manhã – tarde – noite) e, com isto, obter-se-á amostras de como as cruzetas reagem em diversos estados do tempo. Em alguns casos será necessário colher amostras na presença de pouco ruído. Para tanto, o Laboratório de Campo permite a análise em períodos noturnos onde estes ruídos são reduzidos.

A - Local de Montagem

O local definido para a montagem do Laboratório de Campo foi entre os blocos 1C e 5K, na área de campo entre os dois, conforme mostra a Figura 108.

Figura 107 - Local da Construção do Laboratório de Campo

Fonte: Acervo Pessoal

Uma equipe de montagem da CEMIG G, gerenciada pelo funcionário Anderson Carlos Ferreira dos Santos, foi deslocada até o local de montagem e foi criado o Laboratório de Campo, conforme mostra a Figura 109.

Figura 108 - Montagem do Laboratório de Campo

Fonte: Acervo Pessoal

B - Finalização do Laboratório de Campo

Implantação de uma cerca cobrindo toda a área do laboratório de campo com a colocação de portão com cadeado e acabamento final (pintura, brita e rodapés). A Figura 1001 mostra o resultado final da construção do Laboratório de Campo.

Figura 109 - Finalização do Laboratório de Campo

Fonte: Acervo Pessoal

Os postes do Laboratório de Campo foram numerados da esquerda para a direita (de quem está no prédio 5K): Poste 01, Poste 02, Poste 03, Poste 04 e Poste 05.

C - Equipamentos Utilizados no Laboratório de Campo no Seu Estado Inicial

Os equipamentos montados em cada poste (no estado inicial pois posteriormente estas configurações serão modificadas para análises) foram os seguintes:

Poste 01 – Estrutura N2 - Tipo DT – Duplo T

- 2 Cruzetas de Madeira 2400x90x112,5 mm
- 3 Parafusos M16x350 mm
- 6 Isoladores de Pino – Tipo Porcelana 15 KV
- 6 Pinos de Cruzeta 290 mm para Isolador 15 KV
- 2 Mãos Francesas Perfiladas
- 2 Parafusos M16x125 mm – Fixadores da Mão Francesa à Cruzeta
- 1 Parafuso M16x200 mm – Nivelador da Mão Francesa ao Poste
- 1 Parafuso M16x250 mm - Para Armação de 1 Estribo
- 1 Isolador Roldana de Porcelana
- 3 Suportes “L” para Cruzeta – Fixador da Chave Fusível
- 3 Chaves Fusíveis 15 KV 100 A 7,1 KA

Poste 02 – Estrutura M1 - Tipo DT – Duplo T

- 1 Cruzeta de Madeira 2400x90x112,5 mm
- 1 Mão Francesa Perfilada
- 3 Isoladores de Pino 15 KV
- 3 Pinos de Cruzeta 290 mm para Isolador 15KV
- 2 Parafusos M16x250 mm
- 1 Parafuso M16x125 mm
- 1 Armação Secundária para 1 Estribo
- 1 Isolador do Tipo Roldana de Porcelana
- 3 Amarrações de Média Tensão
- 1 Amarração de Neutro
- 1 Fita de Alumínio Protetora do Cabo CA 34 mm² - 2AWG

Poste 03 – Estrutura M4 - Tipo S/C - Seção

- 2 Cruzetas de Madeira 2400x90x112,5 mm
- 2 Mãos Francesas Perfiladas
- 4 Parafusos M16x400 mm
- 2 Parafusos M16x150 mm – Cabeça Abaulada
- 2 Selas para Fixação de Cruzetas
- 1 Par de Cintas de aço 140 mm
- 1 Par de Cintas de aço 150 mm
- 1 Par de Cintas de aço 160 mm
- 7 Olhais para parafuso 50 KN
- 12 Isoladores Disco de Vidro 175x140mm
- 6 Alças do tipo olhal para Cabo C4 – 34 mm
- 6 Ganchos do tipo olhal 50 KN
- 3 Isoladores de Pinos 15 KV
- 3 Pinos de Cruzetas 290 mm para isolador 15 KV
- 6 Parafusos M16x70 mm
- 3 Parafusos M16x45 mm
- 3 Amarrações de Média Tensão
- 2 Sapatilhas
- 2 Alças pré formadas do Tipo Olhal para Cabo 2 – 34 mm
- 8 Arruelas quadradas de 38x18x3 mm
- 8 Porcas quadradas M16x24 mm

Poste 04 – Madeira - Eucalipto leve tratado - Estrutura N1 10 L

- 1 Cruzeta de 2400x90x90 mm
- 1 Mão Francesa Perfilada
- 3 Pinos de Cruzeta de 290 mm para isolador 15KV
- 3 Isoladores de Pino 15KV

- 2 Parafusos M16x250 mm
- 1 Parafuso M16x200 mm
- 1 Parafuso 16x125 mm
- 1 Armação secundária para 1 Estribo
- 1 Isolador Roldana de Porcelana
- 3 fios em alumínio de 1,5 mm para Armações de Média Tensão
- 1 Armação de Neutro com fita de alumínio 1x10 mm p/ armação
- 2 Fitas de Alumínio Protetoras do Cabo 2, 1x10 mm para armação

Poste 05 – Estrutura N3 - Tipo S/C – Seção – 10x300 mm – Poste de Concreto Seção Circular

- 2 Cruzetas de Madeira 2400x90x112,5 mm
- 2 Mãos Francesas Perfiladas
- 2 Parafusos M16x450 mm
- 3 Parafusos M 16x150 mm – Cabeça Abaulada
- 2 Selas de Fixação para Cruzetas
- 6 Parafusos 16x70 mm
- 3 Parafusos 16x45 mm
- 1 Par de Cintas de aço 170 mm
- 1 Par de Cintas de aço 180 mm
- 1 Par de Cintas de aço 190 mm
- 3 Isoladores de Ancoragem Poliméricos 15KV
- 3 Alças do Tipo Olhal para Cabo 2 – 34 mm (AWG)
- 1 Alça de Distribuição para Cabo2
- 1 Sapatilha
- 3 Para Raios 12KV 10KA ZNO
- 3 Suportes do Tipo “L” para Cruzeta - Fixação do Para Raio

Apêndice B – Código Fonte das Principais Rotinas do Programa

Neste apêndice serão mostrados os principais códigos utilizados na programação Lush.

1 - Arquivo getfeat.lsh:

```
(addpath "$HOME/lush")
;-------------------------------------------------------------------
;     PACOTES DE FUNÇÕES BÁSICAS DO LUSH
;-------------------------------------------------------------------

(libload "todos-idx.lsh")
(libload "libc/libc")
(libload "libc/stdio.lsh")
(libload "libc/make.lsh")
(libload "libc/constants")
(libload "libnum/libnum")
(libload "libc/cparse")
(libload "libplot/plotter")
(libload "fftw/fftw")
(libload "audiofile/audiofile-config")
(libload "audio/simple-audio.lsh")
(libload "audiofile/audiofile.lsh")
(libload "mydsp.lsh")
(libload "libimage/image-io")
(libload "kmeans.lsh")
(libload "demo-v4l2.lsh")
(libload "devices/parport.lsh")


;-------------------------------------------------------------------
;  CONFIGURAÇÃO PARA O FILTRO
(defvar filterType)
(defvar filterWindow)
(setq filterType   LowPass)
(setq filterWindow Blackman)
;LowPass   0 - Passa-Baixa
;BandPass  1 - Passa-Faixa
;HighPass  2 - Passa-Alta
;BandRejec 3 - Passa-Faixa
(defvar Blackman  0)
(defvar Hamming   1)

(defvar lowerFc)    ;Frequência de Corte Inferior
(defvar upperFc)    ;Frequência de Corte Superior
(defvar sizeFilter) ;Tamanho do Filtro
(defvar filterVet)  ;Vetor para armazenar o filtro

(setq lowerFc    60)
(setq upperFc    15000)
(setq sizeFilter 64)

;Criando o filtro que será utilizado para todo o processamento
(setq filterWindow (geraFilter filterType 44100 lowerFc upperFc
sizeFilter))
;-------------------------------------------------------------------
;            FIM DA GERAÇÃO DO FILTRO
;-------------------------------------------------------------------
```

```
(setq deslocinit  1) ;deslocamento inicial de 1 segundo
(setq Tempo  2) ;quantidade de segundos de áudio a serem amostrados
(setq fsamostra 44100);taxa de amostragem do sinal a ser capturado do
martelo para 1 segundo
(setq quantamostra (* Tempo fsamostra)) ;88200);quantidade de amostras a
serem capturadas do martelo
 (defvar diruser (getenv "HOME")) ; caminho para a pasta do usuário e
programas
(defvar dirprog (concat diruser "/src/meus_programas")) ; caminho para a
pasta dos programas
(defvar dirsansadia)
(defvar dirsan1ano)
(defvar dirsanurg)
(defvar dirsanemer)
(setq   dirsansadia (concat dirprog "/sanidades/sadia")) ; caminho para a
sanidade sadia
(setq   dirsan1ano  (concat dirprog "/sanidades/1ano"))  ; caminho para a
sanidade 1 ano
(setq   dirsanurg   (concat dirprog "/sanidades/urgencial")) ; caminho
para a sanidade urgencial
(setq   dirsanemer  (concat dirprog "/sanidades/emergencial")) ; caminho
para a sanidade emergencial
(defvar listasadia);lista com as amostras de sanidade sadia
(defvar lista1ano);lista com as amostras de sanidade 1 ano
(defvar listaurg);lista com as amostras de sanidade urgencial
(defvar listaemer);lista com as amostras de sanidade emergencial
(defvar quantsadia) ; quantidade de amostras para sanidade sadia
(defvar quant1ano)  ; quantidade de amostras para sanidade 1 ano
(defvar quanturg)   ; quantidade de amostras para sanidade urgente
(defvar quantemer)  ; quantidade de amostras para sanidade emergencial
(defvar samplesize) ; quantidade de amostras do arquivo de audio
(setq samplesize 4096)
(defvar multsamplesize) ;multiplicador do tamanho da amostra para o caso
de uso da batida manual
(setq multsamplesize 4)
(defvar samplesize2)
(setq samplesize2 (/ samplesize 2)) ; metade da quantidade de amostras do
arquivo de audio
(defvar sizejanenerg)
(setq sizejanenerg 128) ; tamanho da janela para o calculo da energia
(defvar gatedata) ; limiar de detecção do sinal para eliminar sinal
indesejado no inicio do arquivo de áudio
(setq gatedata 3000)
(defvar numLPC) ; quantidade de coeficientes LPC
(setq numLPC 16)
(defvar numMFCC 10) ; quantidade de coeficientes MFCC
(defvar numSPECTRO) ; quantidade de coeficientes do espectro de potência
(setq numSPECTRO samplesize2)
(defvar numZEROS 8) ; quantidade de coeficientes do zero crossing
(defvar numENERGY 16) ; quantidade de coeficientes da energia
(defvar numFEATURES)
(defvar srate  0) ;armazena a taxa de amostragem do arquivo de audiio
(defvar winsize 100) ;define o tamanho da janela de hamming em ms
(defvar winsobrep 80) ; define o tamanho da sobreposição da janela de
hamming em ms
(defvar datawave ())           ;estrutura para armazenar temporariamente
os dados wave
(defvar melWorkingFrequencies (double-matrix)); range de frequencias para
o calculo dos mfcc's
(defvar mfcc   (double-matrix)) ; vetor para armazenar os coeficientes
mfcc
```

```
(defvar nm     (double-matrix)) ; matriz com os vetores de dados originais
(defvar mmedia (double-matrix samplesize));média dos dados
(defvar mcovar (double-matrix)) ;matriz covariância dos dados
(defvar nmnorm (double-matrix)) ;matriz com os vetores de dados
normalizados
(defvar mspect (double-matrix samplesize2)) ; vetor para armazenar o
espectro de potência
(defvar numFilters) ; quantidade de frequencias mel
(defvar melWorkingFrequencies (double-matrix))
(defvar offset 0.15) ; offset para os mfcc coeficientes
(defvar LPC (double-matrix numLPC))
(defvar MFCC (double-matrix numMFCC))
(defvar zc (double-matrix numZEROS))
(defvar energy (double-matrix numENERGY))
(defvar vetfeaturessadia (double-matrix)) ;vetor com todas as features por
sinal
(defvar vetfeatures1ano (double-matrix)) ; vetor com todas as features por
sinal
(defvar vetfeaturesurg (double-matrix)) ; vetor com todas as features por
sinal
(defvar vetfeaturesemer (double-matrix)) ; vetor com todas as features por
sinal
(defvar matfeatures (double-matrix)) ; matriz com todas as amostras de
features
(defvar matfeaturesfr (double-matrix)) ; matriz com todas as amostras de
features
(defvar ckmedias);armazena a estrutura das kmedias
(defvar listasanidades)
(defvar dircomplemento)
(defvar tipodados)
(defvar dirtipo)
(defvar nomefeatures)
(defvar tipofeatures) ;define quais featurers serão utilizadas
(defvar listacruzetas) ; lista com o nome e caminho para todas as cruzetas
analizadas
(setq listacruzetas ());
(defvar tiponormal)     ; tipo de normalização a ser usada para as
features
(defvar tiponormalwave) ; tipo de normalização a ser usada para os dados
wave
setq tiponormal     0) ; Normalização das features obtidas:  0 - sem
normalizacao / 1 - media e variancia / 2 - entre 0 e 1
(setq tiponormalwave 3) ; Normalização do sinal wave amostrado pelo
martelo: 0 - sem normalizacao / 1 - media e variancia / 2 - entre 0 e 1 /
3 - entre -1 e 1
(defvar TipoAnaliseResultado);Define como será processado o resultado para
o reconhecimento
                            ; 0 - procura a maior quantidade de
ocorrencias de um resultado no vetor de quantidade K
                            ; 1 - procura a média dos resultados de um
vetor de quantidade K
                            ; 2 - procura a mediana dos resultados de um
vetor de quantidade K

;---------------------------------------------------------------------
;   CALCULA A OBLIQUIDADE, CURTOSE, DESVIO MEDIO ABSOLUTO (AAD),
;   DESVIO MEDIANO ABSOLUTO (MAD) DE UM VETOR DE DADOS
;   obliquidade (x) = (somatoria (xi - media(x))^3)/((n -
1)*(desviopadrao^3))
;   curtose (x) = (somatoria (xi - media(x))^4)/((n - 1)*(desviopadro^4))
;   aad = (somatoria |xi - media(x)|)/ n
```

```
;   mad = mediana (|x1 - media(x)| ,... , |xn - media(x)|)
; PARÂMETROS
; vet - vetor de dados
; media - media do vetor de dados
; dp  - desvio padrao do vetor de dados
; retorna um vetor com quatro posições: [obliquidade curtose aad mad]
(defun calculaObliquidadeCurtose (vet media dp)
  (let* ((n (idx-dim vet 0))
         (obliquidade 0.0)
         (aux 0.0)
         (curtose 0.0)
         (s3 (* dp dp dp))
         (s4 (* dp dp dp dp))
         (i 0)
         (vetaux (double-matrix n))
         (aad 0)
         (mad 0)
         (resultado (double-matrix 4))
        )
     (for (i 0 (1- n))
        (setq aux          (- (vet i) media))
        (setq obliquidade (+ obliquidade (* aux aux aux)))
        (setq curtose     (+ curtose     (* aux aux aux aux)))
        (setq aad         (+ aad         (abs aux)))
        (vetaux  i        (abs aux))
     )
     (resultado 0 (/ obliquidade (* (1- n) s3)))
     (resultado 1 (/ curtose     (* (1- n) s4)))
     (resultado 2 (/ aad n))
     (resultado 3 (calculaMediana vetaux))
     resultado
  )
)

;-------------------------------------------------------------------
;   CALCULA O INTERVALO DE UM VETOR DE DADOS
;   O INTERVALO É DADO POR max(xi) - min(xi)
; PARÃMETROS:
; vet -vetor de dados
; retorna o valor, do tipo real para o calculo do intervalo
(defun calculaIntervalo (vet)
  (let* ((n (idx-dim vet 0))
         (intervalo 0.0)
         (vetaux (double-matrix n))
         (min 0.0) (max 0.0)
        )
     (setq vetaux (sortupvet vet))
     (setq min (vetaux 0))
     (setq max (vetaux (- n 1)))
     (setq intervalo (- max min))
     intervalo
  )
)

;-------------------------------------------------------------------
;   CALCULA A MEDIANA DE UM VETOR DE DADOS
; PARÂMETROS:
; vet - vetor de dados
; retorna o valor, do tipo real, para o calculo da mediana
;-------------------------------------------------------------------
(defun calculaMediana (vet)
```

```
  (let* ((n (idx-dim vet 0))
         (r 0)
         (vetaux (double-matrix n))
         (mediana 0.0)
        )
     ;ordenar o vetor em ordem crescente
        (setq vetaux (sortupvet vet))
     ;se n for PAR r = n/2 e o resultado será (xr + xr+1)/2
     ;se n for IMPAR r = (n-1)2 e o resultado será xr+1
        (if (= (mod n 2) 0)
          (progn
            (setq r (- (/ n 2) 1))
            (setq mediana (/ (+ (vetaux r) (vetaux (1+ r))) 2)))
          (progn
            (setq r (- (/ (- n 1) 2) 1))
            (setq mediana (vetaux (1+ r)))))
      mediana

  )
)

;-------------------------------------------------------------------------
; OBTEM OS DADOS DO ARQUIVO DE AMOSTRA, PULANDO UM OFFSET (EM SEGUNDOS) E
RETIRADAS O
; INÍCIO EM CONFORMIDADE COM A ENERGIA DO SINAL
; PARÂMETROS
; vet - vetor de dados
; deslocinit - deslocamento inicial a ser pulado em segundos
; fsamostra - taxa de amostragem do sinal
; janela - tamanho da janela para cálculo da energia
; quantdados - quantidados de dados a serem retornados
(defun extraidadossanidade (vet deslocinit fsamostra janela quantdados)
  (let* ((deslocinitamostras (* deslocinit fsamostra))
         (maux1 (double-matrix))
         (ec ())
         (dm ())
         (mvet (double-matrix))
         (desloctotal 0))
      (if (= deslocinit 1)
         (setq maux1 (copvet vet deslocinitamostras fsamostra))
       (setq maux1 (copvet vet deslocinitamostras 0)))
      (setq dl (limiardata maux1 gatedata))
      (setq mvet (copvet maux1  (nth dl 2) quantdados))
      (setq desloctotal (+ deslocinitamostras (nth dl 2)))
     (list mvet desloctotal)))

;-------------------------------------------------------------------------
; TRANSFORMA UMA MATRIZ M X N EM UM VETOR DE TAMANHO M*N
; PARÂMETROS:
; mat - matriz a ser transformada
(defun matvetlinha (mat)
   (let* ((m (idx-dim mat 0))
          (n (idx-dim mat 1))
          (vet (double-matrix (* m n))))
     (for (i 0 (1- m))
       (for (j 0 (1- n))
          (vet (+ (* i n) j) (mat i j))))
     vet))

;-------------------------------------------------------------------------
; TRANSFORMA UM VETOR DE TAMANHO K EM UMA MATRIZ M X N
```

```
; PARÂMETROS:
; vet - vetor
; m - quantidade de linhas da matriz
; n - quantidade de colunas da matriz
; OBS.: k = m*n
(defun vetmatmn (vet m n)
     (let* ((k (idx-dim vet 0))
            (mat (double-matrix m n)))
      (if (= k (* m n))
         (for (i 0 (1- m))
            (for (j 0 (1- n))
                (mat i j (vet (+ j (* i n)))))))
      mat))

;---------------------------------------------------------------------
;LER UM ARQUIVO DE DADOS NO FORMATO TXT
;PARÂMETROS:
; fname - nome do arquivo a ser lido
(defun loadtxt (fname)
   (let* ((lista (read-lines fname))
          (n1 (length lista))(n2 0)(aux "")(ans ())
          (n2 0)
          (mat (double-matrix))
          (vetaux (double-matrix)))
       (if (> n1 0)
        (progn
          ;obtendo o tamanho da linha para gerar a matriz de dados
          (setq n2 (length (split-words (nth lista 1))))
          (setq mat (double-matrix n1 n2))
          (setq vetaux (double-matrix n2))
          (for (i 0 (1- n1))
            (setq aux (split-words (nth lista (+ i 1))))
            (setq n2 (length aux))
            (for (j 0 (1- n2))
                (mat i j (val (nth aux (+ j 1))))))))
     mat)
)

;---------------------------------------------------------------------
; ESCREVE OS DADOS DE UMA MATRIZ EM UM ARQUIVO TEXTO
; PARÂMETROS:
; fname - nome do arquivo txt
; mat - matriz a ser gravada no arquivo texto
(defun writetxt (fname mat)
  (let* ((d (idx-dim mat)) (lin (idx-dim mat 0)) (col 1))
    (if (> (length (idx-dim mat)) 1)
     (progn
       (setq col (idx-dim mat 1))
       (writing fname
         (for (i 0 (1- lin))
           (for (j 0 (1- col))
              (printf "%10.7f " (to-double (mat i j))))
           (printf "\r"))))
     (progn
       (writing fname
         (for (i 0 (1- lin))
              (printf "%15.8f " (to-double (mat i)))
              (printf "\r")))))
   ))
```

```
;---------------------------------------------------------------------
; ESCREVE OS DADOS DE UMA MATRIZ EM UM ARQUIVO TEXTO seguido de um
comentário no final da linha
; PARÂMETROS:
; fname - nome do arquivo txt
; mat - matriz a ser gravada no arquivo texto
; listadados - lista de nomes a serem inseridos
(defun writetxt2 (fname mat listadados)
  (let* ((d (idx-dim mat)) (lin (idx-dim mat 0)) (col 1))
    (if (> (length (idx-dim mat)) 1)
     (progn
       (setq col (idx-dim mat 1))
       (writing fname
         (for (i 0 (1- lin))
           (for (j 0 (1- col))
              (printf "%10.7f " (to-double (mat i j))))
           (printf "%s" (nth listadados (1+ i)))
           (printf "\r"))))
     (progn
       (writing fname
         (for (i 0 (1- lin))
              (printf "%4.0f - %15.8f -  %s " (+ i 1) (to-double (mat i))
(nth listadados (+ i 2)))
              (printf "\r")))))
   ))

;---------------------------------------------------------------------
; ESCREVE OS DADOS DE UMA MATRIZ EM UM ARQUIVO TEXTO
; PARÂMETROS:
; fname - nome do arquivo txt
; mat - matriz a ser gravada no arquivo texto
(defun writetxtmatlab (fname mat)
  (let* ((d (idx-dim mat)) (lin (idx-dim mat 0)) (col 1))
    (if (> (length (idx-dim mat)) 1)
     (progn
       (setq col (idx-dim mat 1))
       (writing fname
         (printf "[ ")
         (for (i 0 (1- lin))
           (for (j 0 (1- col))
              (printf "%10.7f " (to-double (mat i j))))
           (printf ";")
           (printf "\r"))
         (printf " ]")
         ))
     (progn
       (writing fname
         (printf "[ ")
         (for (i 0 (1- lin))
              (printf "%15.8f " (to-double (mat i)))
              (printf ";")
              (printf "\r"))
         (printf " ]")
         )))
   ))

;---------------------------------------------------------------------
; NORMALIZA UM VETOR DE DADOS USANDO MEDIA E VARIÃNCIA
(defun normalizaDados1 (vet)
  (let* ((dimvet (idx-dim vet 0))
         (mVet   (mediaVet vet))             ; média do vetor de dados
```

```
      (mdesv  (sqrt (varianciaMat Vet)))  ;desvio padrão do vetor de dados
         (result (double-matrix dimvet)))    ;armazena o resultado
      (for (i 0 (1- dimvet))
         (result i (/ (- (vet i) mVet) mdesv)))
      result))

;---------------------------------------------------------------------
; NORMALIZA UM VETOR DE DADOS NO RANGE de rin até rout

(defun normalizaDados2 (vet rin rout)
    (let* ((dimvet (idx-dim vet 0))
           (vetout (double-matrix dimvet))
           (vmax ((idx-sup vet))) ;localizar o maximo de vet
           (vmin ((idx-inf vet))) ;localizar o mínimo de vet
          )
   (if (and (< rin rout) (<> vmax vmin))
    (for (i 0 (1- dimvet))
      (vetout i (+ rin (/ (* (- (vet i) vmin) (- rout rin)) (- vmax
vmin))))))
   vetout))

;---------------------------------------------------------------------
; CALCULA A VARIÃNCIA DE UM VETOR
(defun varianciaMat (vet)
  (let* ((dimvet (idx-dim vet 0))
         (mvet (mediaVet vet))
         (result 0.0))
    (for (i 0 (1- dimvet))
      (setq result (+ result (* (- (vet i) mVet) (- (vet i) mVet)))))
    (/ result (1- dimvet))))

;-------ORDENA O VETOR EM DOWN MODE
(defun sortdownvet (x)
  (let ((mlist (arraytolist x)))
     (setq mx (sort-list mlist <))
     (listtoarray mx)))

;----------------------------ORDENA O VETOR EM UP MODE
(defun sortupvet (x)
  (let ((mlist (arraytolist x)))
     (setq mx (sort-list mlist >))
     (listtoarray mx)))

;---------------------------------------------------------------------
; Programa que calcula uma média de uma linha de uma amostra.
; mvet - vetor de dados
(defun mediaVet (mvet)
      (let ((somaMat 0.0))
          (defvar dimMat)
          (setq dimMat (idx-dim mvet))
          (setq somaMat (idx-sum mvet))
          (/ (somaMat) (car dimMat))))

;---------------------------------------------------------------
;calcular o vetor media de uma matriz m utilizando as linhas da matriz
;para chamar a função mediaVet
(defun mediaMat (m)
      (let* ( (n 0)
                (numColMat 0)
                (mediaLinha 0.0)
                (n (idx-dim m))
```

```
                (numColMat (car n)) ;;obtem o numero de linhas
                (matMedia (double-matrix numColMat))) ;;inicializa a
matriz de media
          (for (i 1 numColMat)
             (matMedia (- i 1) (mediaVet (matvet m i)))) ;;armazena o
resultada da media da linha na posicao correspondente
          matMedia))

;---------------------------------------------------------------------
;;-----Calcula a covariancia de uma matriz
;; m - matriz de dados (cada coluna é uma amostra)
;; v - vetor de médias
(defun covarMat (m v)
 (let* ((z 0)
        (i 0)
        (j 0)
        (a1 0)
        (a2 0)
        (auxK 0)
        (K (idx-dim m 1)) ;; numero de colunas
        (D (idx-dim m 0)) ;; numero de linhas
        (N (idx-dim v 0)) ;; quantidade de elementos do vetor media
        (matC (double-matrix N N))
        (somaAux 0.0))
    ;;D - quantidade de linhas da matriz m
    ;;K - obter a quantidade de colunas da matriz m
    ;;matC - matriz que irá armazenar as covariancias
       (for (i 0 (- N 1))
         (for (j 0 (- N 1))
            (setq somaAux 0.0)
            (for (z 0 (- K 1))
               ;;calcular a somatoria
                (setq a1 (- (m i z) (v i)))
                (setq a2 (- (m j z) (v j)))
                (setq somaAux (+ somaAux (* a1 a2))))
            ;;calcular a posição da matriz covariancia
            (setq auxK (* (/ 1 (- K 1.0)) somaAux))
            (matC i j auxK)))
       matC))

;---------------------------------------------------------------------
;-- NORMALIZAÇÃO DA MATRIZ matOrig DE MEDIA vetMed e COVARIANCIA matCov
(defun normalizaMat (matOrig vetMed matCov)
 (let* ((K1 (idx-dim matOrig 0)) ;;numero de linhas da matriz de dados
        (D1 (idx-dim matOrig 1)) ;;numero de colunas da matriz de dados
        (matNormalizada (double-matrix K1 D1))                    ;;nova
matriz normalizada
        (vetCovSqrt (idx-sqrt (extraidiagonalmatriz matCov))) ;;vetor com
as raizes quadradas de vetCov
        (auxN 0.0))
   (for (k 0 (- K1 1))
    (for (d 0 (- D1 1))
       (matNormalizada k d (/ (- (matOrig k d) (vetMed d)) (vetCovSqrt
d)))))
   matNormalizada))

;-------------------------------------------------------
; CALCULA A COVARIÂNCIA DE UMA MATRIZ
; m - matriz a ser calculada a covariância
;
; OBS.: esta função já faz a chamada para o cálculo da função média
```

```
;       e também já faz a função para o calculo direto da covariância
;       através da função covarMat
(defun covariancia (m)
  (let* (;(mt (transpose m))
         ;(mMedia (mediaMat mt))
         (mMedia (mediaMat m))
         ;(mCovariancia (covarMat mt mMedia)))
         (mCovariancia (covarMat m mMedia)))
    (list mCovariancia mMedia)))

;-------------------------------------------------------------------------
;---CALCULA A PORCENTAGEM DE LAMBDA
; vetor - vetor contendo a diagonal da matriz covariancia
; comp - valor a ser comparado para ter o retorno (em porcentagem
SUBTRAIDO DE 100
; ex.: (calcLambda x 95) - vetor x e 5% de comparação
; RETORNA A DIMENSÃO REDUZIDA
(defun calcLambda (vetor comp )
   (let* ((D (idx-dim vetor 0))
          (DR 0)
          (result 100.0)
          (rr (/ comp 100.0))
          (aux1 ((idx-sum vetor)))
          (aux2 0.0)
          (teste (float-matrix D)))
     (while (AND (<= DR D) (> result rr))
         (if (< DR (1- D))
           (progn
            (setq aux2 0.0)
            (for (i (+ DR 1) (1- D))
              (setq aux2 (+ aux2 (vetor i))))
            (setq result (/ aux2 aux1))
            (teste DR result)
            (setq DR (+ DR 1))))
     )
     (setq DR (- D DR))
     DR))

;-------------------------------------------------------------------------
; CALCULA A DISTÂNCIA DOS DADOS DE UM VETOR EM RELAÇÃO A SUA MÉDIA
; PARÂMETROS:
; vet - vetor de dados
(defun distDadosMedia (vet)
  (let* ((tam (idx-dim vet 0))
         (result (double-matrix tam))
         (vmedia (mediaVet vet)))
    (for (i 0 (1- tam))
      (result i (- (vet i) vmedia)))
    result))

;-------------------------------------------------------------------
;  CALCULA A MENOR DISTÂNCIA DA MATRIZ NORMALIZADA E O VETOR DE MÉDIAS DA
;  MATRIZ NORMALIZADA
;  s - Matriz Normalizada e Transposta das features

(defun minNormMedia (s)
  (let* ((d (idx-dim s 0)) ;armazena a quantidade de linhas de s
         (k (idx-dim s 1)) ;armazena a quantidade de colunas de s
         (aux 0)
         (vmedia (mediaMat s)) ;calcula o vetor de médias da matriz s
```

```
         (auxvet (double-matrix k)) ;vetor auxiliar para armazenar as
linhas da matriz s
         (auxsub (double-matrix k)) ; armazena as subtrações por coluna
         (resultado (double-matrix d))) ;vetor par armazenar os minimos
das distâncias
    (for (i 0 (1- d))
      (progn
        ;obtem cada linha da matriz s
          (setq auxvet (extrailinhamatriz s 0))
          (for (j 0 (1- k))
            (auxsub j (abs (- (auxvet j) (vmedia i)))))
          (resultado i ((idx-inf auxsub)))))
         ;))
    resultado))

;---------------------------
; Obtem um subvetor de um vetor
; m - vetor de dados
; pos - posição de início para copia
; n - quantidade de dados a serem copiados
; pos deverá ser menor do que a quantidade de dados do vetor, caso
contrário retorna um vetor vazio
; n - nao deverá ultrapassar o tamanho total do vetor, caso acontece irá
copiar somente os dados existente. Se n = 0, copia todos os dados a partir
da posição pos
(defun copvet (m pos n)
  (let* ((T (idx-dim m 0))
         (j 0)
         (auxM 0)
         (defvar auxM)
         (posinicial pos))

   (if (or (> (-  (+ pos n) 1) T) (= n 0))
      (setq n (- T pos)))
   (setq auxM (double-matrix n))
   (if (< pos T)
    (for (i posinicial (+ posinicial (- n 1)))
           (if (<= posinicial T)
            (progn
              (if (< i T)
                 (auxM j (m i)))
              (setq j (+ j 1))))))
   auxM))

;-------------------------------------------------------------------------
; PRÉ-PROCESSAMENTO DO SINAL DE ÁUDIO
; FASES:
; 1 - Determinar início do sinal
; 2 - Obter a quantidade de amostras necessárias em potência de 2 (4096
por exemplo)
; 3 - Normalizar utilizando Variância e Média
; 4 - Aplicar janela de Hamming com sobreposição
; PARÂMETROS:
; x - sinal a ser processado
; srate - taxa de amostragem do sinal
; d - ponto de inicio do sinal a ser amostrado
; tiponormalwave - tipo de normalização a ser aplicada no sinal
; filtrar - executa filtro para os sinais ou não (0 - NÃO / 1 - SIM)
(defun preprocessasinal (x srate d tiponormalwave filtrar)
  (let* ((n (idx-dim x 0)) ; quantidade de elementos do vetor de áudio
           (desloc 0))
```

```
   (progn
    ;APLICAR FILTRO ANTES DE EXTRAI A QUANTIDADE DE AMOSTRAS NECESSARIAS
    (if (= filtrar 1)
        (setq m1 (filtraSinal srate x filterWindow))
       (setq m1 x))
     ;LOCALIZAR O PONTO DE INICIO DO SINAL
    (setq desloc (nth (extraidadossanidade m1 0 srate sizejanenerg
samplesize) 2))
    ;EXTRAIR A QUANTIDADE DE SINAIS NECESSARIOS
    (setq m1 (copvet m1 desloc samplesize))
    (if (= tiponormalwave 1)
     (setq m1 (normalizaDados1 m1))) ;normaliza os dados utilizando
variancia e media
    (if (= tiponormalwave 2)
     (setq m1 (normalizaDados2 m1 0 1))) ; normaliza os dados entre 0 e 1
    (if (= tiponormalwave 3)
     (setq m1 (normalizaDados2 m1 -1 1))) ; normaliza os dados entre -1 e
1
   )
   m1))

;-------------------------------------------------------------------------
; OBTEM TODAS AS FEATURES PARA UM DADO SINAL
; FEATURES:
; 1 - LPC (10 coeficientes)
; 2 - MFCC (10 coeficientes)
; 3 - Espectro de Potência (100 primeiros coeficientes)
; 4 - Zero Crossing (8 janelas de 512 amostras cada)
; 5 - Energia (16 janelas de 128 amostras cada)
; 6 - ARÉA abaixo do sinal
; O VETOR FINAL TERÁ DIMENSÃO DE 236
; PARÂMETROS:
; x - vetor de dados
;norm - tipo de normalização:
;  0 - sem normalização
;  1 - normalização pela media e variancia
;  2 - normalização no range 0 1
(defun obtemfeatures (x norm modo)
  (let* ((pe 0)
         (erro 0)
         (spectralData (double-matrix))
         (sData (double-matrix))
         (result (double-matrix))
         (areasinal 0.0)
         (aux 0)
         (quantfeatures 0))
   (if (= modo 0)
      (setq nomefeatures "-SEMPCA")
     (setq nomefeatures "-COMPCA"))
;obtem os coeficientes
   (setq spectralData (spectrum x 0)) ;spectro total de potencia
   (if (= (tipofeatures 0) 1)
      (setq sData (copvet spectralData 0 numSPECTRO))) ; obtem os
coeficientes do espectro de potência
   (if (= (tipofeatures 1) 1)
     (setq erro (LPCAutocorrelation x LPC pe))) ;obtem os coeficientes lpc
   (if (= (tipofeatures 2) 1)
      (setq MFCC (computeMFCC2 spectralData sRATE 48 numMFCC))) ; obtem os
coeficientes MFCC
   (if (= (tipofeatures 3) 1)
```

```
      (setq zc (listtoarray (zjanela x srate (/ samplesize numZEROS)))))  ;
obtem os coeficientes de zero-crossing
   (if (= (tipofeatures 4) 1)
      (setq energy (listtoarray (ejanela x srate (/ samplesize
numENERGY))))) ; obtem os coeficientes de energia
   (if (= (tipofeatures 5) 1)
      (setq areasinal (calculaareasinal x sRATE)))
;normaliza os coeficientes usando media e variancia
    (if (= norm 1)
     (progn
      (if (= (tipofeatures 0) 1)(setq sData  (normalizaDados1 sData  )))
      (if (= (tipofeatures 1) 1)(setq LPC    (normalizaDados1 LPC    )))
      (if (= (tipofeatures 2) 1)(setq MFCC   (normalizaDados1 MFCC   )))
      (if (= (tipofeatures 3) 1)(setq zc     (normalizaDados1 zc     )))
      (if (= (tipofeatures 4) 1)(setq energy (normalizaDados1 energy )))
     ))
    (if (= norm 2)
     (progn
;normaliza os coeficientes no range 0 a 1
     (if (= (tipofeatures 0) 1)(setq sData  (normalizaDados2 sData  0 1)))
     (if (= (tipofeatures 1) 1)(setq LPC    (normalizaDados2 LPC    0 1)))
     (if (= (tipofeatures 2) 1)(setq MFCC   (normalizaDados2 MFCC   0 1)))
     (if (= (tipofeatures 3) 1)(setq zc     (normalizaDados2 zc     0 1)))
     (if (= (tipofeatures 4) 1)(setq energy (normalizaDados2 energy 0 1)))
      ))
;---------------------------------------------------------------------
   (setq aux numSPECTRO)
   ;determina a quantidade de features
   (if (= (tipofeatures 0) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numSPECTRO))
        (setq nomefeatures (concat nomefeatures "-FFT"))))
   (if (= (tipofeatures 1) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numLPC))
        (setq nomefeatures (concat nomefeatures "-LPC"))))
   (if (= (tipofeatures 2) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numMFCC))
        (setq nomefeatures (concat nomefeatures "-MFCC"))))
   (if (= (tipofeatures 3) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numZeros))
        (setq nomefeatures (concat nomefeatures "-ZC"))))
   (if (= (tipofeatures 4) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numENERGY))
        (setq nomefeatures (concat nomefeatures "-En"))))
   (if (= (tipofeatures 5) 1)
      (progn
        (setq quantfeatures (+ quantfeatures 1))
        (setq nomefeatures (concat nomefeatures "-AREA"))))
;---------------------------------------------------------------------
   (setq result (double-matrix quantfeatures))
;define a quantidade total de features no processo de treinamento
   (setq numFEATURES quantfeatures)
;monta o vetor resultante
   (setq quantfeatures 0)
;coeficientes SPECTRO
   (if (= (tipofeatures 0) 1)
    (progn
```

```
     (for (i 0 (1- numSPECTRO)) (result (+ i quantfeatures) (sData i)))
     (setq quantfeatures (+  quantfeatures numSPECTRO))))
;coeficientes LPC
   (if (= (tipofeatures 1) 1)
    (progn
     (for (i 0 (1- numLPC))
       (result (+ i quantfeatures) (LPC i)))
     (setq quantfeatures (+  quantfeatures numLPC))))
;coeficientes mfcc
   (if (= (tipofeatures 2) 1)
    (progn
     (for (i 0 (1- numMFCC)) (result (+ i quantfeatures) (MFCC i)))
     (setq quantfeatures (+  quantfeatures numMFCC))))
;coeficientes ZeroCrossing
   (if (= (tipofeatures 3) 1)
    (progn
     (for (i 0 (1- numZeros)) (result (+ i quantfeatures) (zc i)))
     (setq quantfeatures (+  quantfeatures numZeros))))
;coeficientes ZeroCrossing
   (if (= (tipofeatures 4) 1)
    (progn
     (for (i 0 (1- numENERGY)) (result (+ i quantfeatures) (energy i)))
     (setq quantfeatures (+  quantfeatures numENERGY))))
;coeficientes da AREA
   (if (= (tipofeatures 5) 1)
    (progn
     (result quantfeatures areasinal)
     (setq quantfeatures (+  quantfeatures 1))))
;-----------------------------------------------
(setq numSPECTRO aux) ;retorna a feature spectrum
;-----------------------------------------------
   result))
;-------------------------------------------------------------------------
;---Processa todas as informações (algoritmo proposto)
(setq dircomplemento (concat "/" dirtipo "/" tipodados))
(setq listasanidades (list "1ANO" "URGENTE" "EMERGENCIAL"))
(setq gerakmedia 0) ; define se serão calculados os centroides

;-------------------------------------------------------------------------
; Gera uma matriz com todas as features propostas de todos os sinais
amostrados
; para todas as sanidades automaticamente
; Parâmetros:
; dircomplemento - diretorio base onde estão armazenados os sinais
amostrados
; deslocfile - deslocamento inicial a ser aplicado ao sinal, caso seja
necessário
; modo - sem PCA (0) ou com PCA (1)
; filtrar - aplica filtro ao sinais ou não (0 - NÃO / 1 - SIM)
(defun geramatrizfeatures(dircomplemento deslocfile modo filtrar)
  (let* ((meigen (double-matrix)); vetor com os eigen valus da matriz
covariancia
         (nametxt "")(deslocdatamat 0)
         (m1     (double-matrix samplesize)) ;armazena os dados do canal 1
         (m2     (double-matrix samplesize)) ;armazena os dados do canal 2
         (m3     (double-matrix samplesize)) ;armazena os dados do canal 3
         (nchannel 0) ;armazena a quantidade de canais do arquivo de audio
       (namefile "") ;armazena o nome do arquivo de audio a ser processado
         (vlambda 0)    ; calcula a porcentagem de lambda
 (K 0) ;armazena temporariamente a quantidade de linhas da matriz de dados
 (D 0) ;armazena temporariamente a quantidade de cols da matriz de dados
```

```
 (nlist1 0)(nlist2 0)(nlist3 0)(nlist4 0) ;núm. de elem. da lista de
sanidade
        )
    (progn
        (setq listacruzetas (list ""))
        (setq matfeatures (double-matrix))
;---------------------------------------------------------------------
      ;obter as amostras para a sanidade sadia
        (setq dirsansadia (concat dirsansadia dircomplemento))
        (setq listasadia (files dirsansadia))
      ;obter a quantidade de elementos da lista
        (setq nlist1 (length listasadia))
      ;armazenar a quantidade de amostras para a sanidade sadia
        (setq quantsadia 0);deslocfile)
      ;processar cada amostra
      ;  (for (i 3 (1- nlist1))
        (for (i 3 nlist1)
          (setq namefile (concat (concat dirsansadia "/") (nth listasadia
i))) ;obter o nome do arquivo
          (if (<> (index ".wav" namefile) ());testa se é um arquivo do
tipo WAV
           (progn
             (setq quantsadia (+ quantsadia 1))
             (setq datawave (loadwave namefile 0));leitura de todo o sinal
wave
           (setq m1 (extraicanalwave datawave canalaudio)) ;obtem os dados
do canal
           (setq srate (extraitx datawave)) ;obtem a taxa de amostragem
           ;obtem o ponto de inicio do sinal
;             (setq nametxt (concat dirsansadia "/a" (str quantsadia)
".txt"))
             (setq nametxt (concat (left namefile (index ".wav" namefile))
"txt"))
             (setq deslocdatamat (loadtxt nametxt))
           (setq m1 (preprocessasinal m1 srate (deslocdatamat 0 0)
tiponormalwave filtrar)) ; pré-processamento do sinal
           ;obtem o vetor de features
           (setq vetfeaturessadia (obtemfeatures m1 tiponormal modo))
           ;adiciona o vetor de features para a matriz de features para a
sanidade
           (setq matfeatures (addlinvetmat matfeatures vetfeaturessadia))
             (printf "================================\n")
             (printf "%s \n" namefile)
             (printf "%s - %d \n" nametxt (deslocdatamat 0 0))
             (printf "================================\n")
            ;insere o nome do arquivo na lista de nomes de cruzetas
processadas
             (list-insert listacruzetas (length listacruzetas) namefile)
           ))
         )
        (printf
"=====================================================================\n")
        (printf "QUANTIDADE DE DADOS SANIDADE SADIA: %d \n" quantsadia)
        (printf
"=====================================================================\n")
;---------------------------------------------------------------------
      ;obter as amostras para a sanidade 1ano
        (setq dirsan1ano (concat dirsan1ano dircomplemento))
        (setq lista1ano (files dirsan1ano))
      ;obter a quantidade de elementos da lista
        (setq nlist2 (length lista1ano))
```

```
      ;armazenar a quantidade de amostras para a sanidade sadia
        (setq quant1ano 0)
      ;processar cada amostra
        (for (i 3 nlist2)
          (setq namefile (concat (concat dirsan1ano "/") (nth lista1ano
i))) ;obter o nome do arquivo
          (if (<> (index ".wav" namefile) ());testa se é um arquivo do
tipo WAV
           (progn
             (setq quant1ano (+ quant1ano 1))
           (setq datawave (loadwave namefile 0));leitura de todo o sinal
wave
           (setq m1 (extraicanalwave datawave canalaudio)) ;obtem os dados
do canal 1
           (setq srate (extraitx datawave)) ;obtem a taxa de amostragem
           ;obtem o ponto de inicio do sinal
             (setq nametxt (concat (left namefile (index ".wav" namefile))
"txt"))
             (setq deslocdatamat (loadtxt nametxt))
           (setq m1 (preprocessasinal m1 srate (deslocdatamat 0 0)
tiponormalwave filtrar)) ; pré-processamento do sinal

           ;obtem o vetor de features
           (setq vetfeatures1ano (obtemfeatures m1 tiponormal modo))
           ;adiciona o vetor de features para a matriz de features para a
sanidade
           (setq matfeatures (addlinvetmat matfeatures vetfeatures1ano))
             (printf "================================\n")
             (printf "%s \n" namefile)
             (printf "%s - %d \n" nametxt (deslocdatamat 0 0))
             (printf "================================\n")
            ;insere o nome do arquivo na lista de nomes de cruzetas
processadas
             (list-insert listacruzetas (length listacruzetas) namefile)
           ))
       )
        (printf
"=====================================================================\n")
        (printf "QUANTIDADE DE DADOS SANIDADE 1 ANO: %d \n" quant1ano)
        (printf
"=====================================================================\n")
;---------------------------------------------------------------------
      ;obter as amostras para a sanidade urgente
        (setq dirsanurg (concat dirsanurg dircomplemento))
        (setq listaurg (files dirsanurg))
      ;obter a quantidade de elementos da lista
        (setq nlist3 (length listaurg))
      ;armazenar a quantidade de amostras para a sanidade sadia
        (setq quanturg 0);deslocfile)
      ;processar cada amostra
        (for (i 3 nlist3)
          (setq namefile (concat (concat dirsanurg "/") (nth listaurg i)))
;obter o nome do arquivo
          (if (<> (index ".wav" namefile) ());testa se é um arquivo do
tipo WAV
           (progn
             (setq quanturg (+ quanturg 1))
           (setq datawave (loadwave namefile 0));leitura de todo o sinal
wave
           (setq m1 (extraicanalwave datawave canalaudio)) ;obtem os dados
do canal 1
```

```
           (setq srate (extraitx datawave)) ;obtem a taxa de amostragem
           ;obtem o ponto de inicio do sinal
             (setq nametxt (concat (left namefile (index ".wav" namefile))
"txt"))
             (setq deslocdatamat (loadtxt nametxt))
           (setq m1 (preprocessasinal m1 srate (deslocdatamat 0 0)
tiponormalwave filtrar)) ; pré-processamento do sinal
           ;obtem o vetor de features
           (setq vetfeaturesurg (obtemfeatures m1 tiponormal modo))
           ;adiciona o vetor de features para a matriz de features para a
sanidade
           (setq matfeatures (addlinvetmat matfeatures vetfeaturesurg))
             (printf "================================\n")
             (printf "%s \n" namefile)
             (printf "%s - %d \n" nametxt (deslocdatamat 0 0))
             (printf "================================\n")
           ;insere o nome do arquivo na lista de nomes de cruzetas
processadas
             (list-insert listacruzetas (length listacruzetas) namefile)
           ))
         )
        (printf
"=========================================================================\n")
        (printf "QUANTIDADE DE DADOS SANIDADE URGENCIAL: %d \n" quanturg)
        (printf
"=========================================================================\n")
;-------------------------------------------------------------------------
      ;obter as amostras para a sanidade emergencial
        (setq dirsanemer (concat dirsanemer dircomplemento))
        (setq listaemer (files dirsanemer))
      ;obter a quantidade de elementos da lista
        (setq nlist4 (length listaemer))
      ;armazenar a quantidade de amostras para a sanidade sadia
        (setq quantemer 0);deslocfile)
      ;processar cada amostra
        (for (i 3 nlist4)
          (setq namefile (concat (concat dirsanemer "/") (nth listaemer
i))) ;obter o nome do arquivo
          (if (<> (index ".wav" namefile) ());testa se é um arquivo do
tipo WAV
           (progn
             (setq quantemer (+ quantemer 1))
           (setq datawave (loadwave namefile 0));leitura de todo o sinal
wave
           (setq m1 (extraicanalwave datawave canalaudio)) ;obtem os dados
do canal 1
           (setq srate (extraitx datawave)) ;obtem a taxa de amostragem
           ;obtem o ponto de inicio do sinal
             (setq nametxt (concat (left namefile (index ".wav" namefile))
"txt"))
             (setq deslocdatamat (loadtxt nametxt))
           (setq m1 (preprocessasinal m1 srate (deslocdatamat 0 0)
tiponormalwave filtrar)) ; pré-processamento do sinal
           ;obtem o vetor de features
           (setq vetfeaturesemer (obtemfeatures m1 tiponormal modo))
           ;adiciona o vetor de features para a matriz de features para a
sanidade
           (setq matfeatures (addlinvetmat matfeatures vetfeaturesemer))
             (printf "================================\n")
             (printf "%s \n" namefile)
             (printf "%s - %d \n" nametxt (deslocdatamat 0 0))
```

```
             (printf "================================\n")
            ;insere o nome do arquivo na lista de nomes de cruzetas
processadas
             (list-insert listacruzetas (length listacruzetas) namefile)
           ))
        )
       (printf
"=====================================================================\n")
       (printf "QUANTIDADE DE DADOS EMERGENCIAL: %d \n" quantemer)
       (printf
"=====================================================================\n")
    )
;retorna a matriz de features
    matfeatures))

;-----------------------------------------------------------------------
; PROCESSA TODOS OS DADOS ARMAZENADOS, CALCULANDO A MATRIZ DE FEATURES
NECESSÁRIA PARA O PROCESSAMENTO
; PARÂMETROS:
;  K - quantidade de clusters
;  modo: 0 - sem PCA / 1 - com PCA
;  filtrar : aplica ou não o filtro aos sinais (0 - NÃO / 1 - SIM)
(defun processadados (K modo filtrar)
  (let* (;(ckmedias ())
         (v (double-matrix))
         (s (double-matrix))
         (a (double-matrix))
         (u (double-matrix))
         (dr 0)
         (ur (double-matrix))
         (fr (double-matrix))
         (matdef (double-matrix 4 1))
         (vetligacao (double-matrix))
         (vetc (double-matrix))
         (offset 0)
        )
;(printf "Processando...\\n")
    (cond
         ((= tipodados "treinamento")
           (setq offset 0))
         ((= tipodados "teste")
           (setq offset 25))
         ((= tipodados "comparacao")
           (setq offset 40)))
;mudando o tamanho da amostra, caso seja necessario
    (if (= dirtipo "manual")
       (setq samplesize (* multsamplesize samplesize)))
    (setq matfeatures (geramatrizfeatures dircomplemento offset modo
filtrar));processa os vetores de dados armazenados
;armazena a matriz com as features
    (save-matrix matfeatures (concat dirprog "/sanidades/K" (str K) "-
matfeatures" nomefeatures "-" dirtipo ".dat"))
    (writetxt  (concat dirprog "/sanidades/K" (str K) "-matfeatures"
nomefeatures "-" dirtipo ".txt") matfeatures)
;lê a matriz de features (somente para testes pois a matriz já existe na
variável matfeatures)
    (load-matrix matfeatures (concat dirprog "/sanidades/K" (str K) "-
matfeatures" nomefeatures "-" dirtipo ".dat"))
  (if (= modo 1)
   (progn
```

```
    (setq matcov (covariancia (transpose matfeatures)));covariância e
media da matriz de features
    (setq v (double-matrix (idx-dim matfeatures 1) (idx-dim matfeatures
1)))
    (setq s (double-matrix (idx-dim matfeatures 1)));eight values
    (setq a (car matcov)); matriz covariância
    (setq u (svd a v s)) ; eight vector ordenados
    (setq dr (calcLambda s 5));dimensão reduzida
    (setq ur (copymat u 0 0 (- (idx-dim u 0) 1) (1- dr)));eight vectors
reduzidos
    (setq matfeaturesfr (transpose (multmat (transpose ur) (transpose
matfeatures))));features reduzidas
    (save-matrix ur               (concat dirprog "/sanidades/K" (str K)
"-dr" (str dr) "-mattranslation" nomefeatures "-" dirtipo ".dat"))
   )
   (progn
     (setq matfeaturesfr matfeatures)
     (setq DR (idx-dim matfeatures 1))))

;armazena a matriz com as novas features
     (save-matrix matfeaturesfr               (concat dirprog
"/sanidades/K" (str K) "-dr" (str dr) "-matfeaturesreduzidas" nomefeatures
"-" dirtipo ".dat"))
     (writetxt (concat dirprog "/sanidades/K" (str K) "-dr" (str dr) "-
matfeaturesreduzidas" nomefeatures "-" dirtipo ".txt") matfeaturesfr)

;;;;-----MUDANÇAS PARA CALCULAR USANDO MSE----

  (if (= gerakmedia 1)
   (progn
;calcula os centroides
    (setq ckmedias (evalkmeans matfeaturesfr K))
;salvando dados
    (save-matrix (nth ckmedias 1) (concat dirprog "/sanidades/K" (str K)
"-dr" (str dr) "-centroides" nomefeatures "-" dirtipo ".dat"))
    (save-matrix (nth ckmedias 2) (concat dirprog "/sanidades/K" (str K)
"-dr" (str dr) "-populacao" nomefeatures "-"  dirtipo ".dat"))
    (save-matrix (nth ckmedias 3) (concat dirprog "/sanidades/K" (str K)
"-dr" (str dr) "-energia" nomefeatures "-"    dirtipo ".dat"))
    (save-matrix (nth ckmedias 4) (concat dirprog "/sanidades/K" (str K)
"-dr" (str dr) "-ligacao" nomefeatures "-"    dirtipo ".dat"))
    (writetxt (concat dirprog "/sanidades/K" (str K) "-dr" (str dr) "-
populacao" nomefeatures "-" dirtipo ".txt") (nth ckmedias 2))
    (writetxt (concat dirprog "/sanidades/K" (str K) "-dr" (str dr) "-
energia"   nomefeatures "-" dirtipo ".txt") (nth ckmedias 3))
    (writetxt (concat dirprog "/sanidades/K" (str K) "-dr" (str dr) "-
ligacao"   nomefeatures "-" dirtipo ".txt") (nth ckmedias 4))
 ))
;;;;-------FIM MUDANÇAS PARA MSE
    (matdef 0 0 K)
    (matdef 1 0 dr)
    (matdef 2 0 (idx-dim matfeatures 0))
    (matdef 3 0 (idx-dim matfeatures 1))
    (writetxt  (concat dirprog "/sanidades/K" (str K) "-definicao"
nomefeatures "-"    dirtipo ".txt") matdef)

(if (= gerakmedia 1)
  (progn
;salvar a ordem dos centroides
    (setq vetligacao (copvet (nth ckmedias 4) 0 (- (idx-dim (nth ckmedias
4) 0) 1)))
```

```
    (setq vetc (selectcluster vetligacao (/ (idx-dim matfeaturesfr 0) K) K
listasanidades))
    (save-matrix vetc  (concat dirprog "/sanidades/K" (str K) "-dr" (str
dr) "-sanidadecluster" nomefeatures "-" dirtipo ".dat"))
    (writetxt (concat dirprog "/sanidades/K" (str K) "-dr" (str dr) "-
sanidadecluster" nomefeatures "-" dirtipo ".txt") vetc)
  ))
  )
)

;---------------------------------------------------------------------
; TESTA O SINAL UTILIZANDO KNN
; PARÂMETROS:
;    modo : 0 - sem pca 1 - com pca
;    knn números de proximidades
;    filtrar : executar ou não o filtro nos sinais (0 - NÃO / 1 - SIM)
(defun testasinalnewKNN (modo KNN filtrar)
   (let* ((DR 0) ;redução de features
          (K  1)   ;quantidade de centroides
          (qdados1 0)(qdados2 0)(qdados3 0)
          (matdef (double-matrix))
          (matd1 (double-matrix)) ; vetor com as distâncias euclidianas de
todas as analises
          (matd2 (double-matrix)) ; vetor com os indices da posições de
todas as analises
          (linhad1 (double-matrix))
          (linhad2 (double-matrix))
          (inifiles 0)
          (dircomplemento (concat "/" dirtipo "/" tipodados))
          (namefile "")(nametxt "")
          (deslocdatamat 0)(m1 (double-matrix))(vetfeatures (double-
matrix))
          (matfeatures (double-matrix))
          (v (double-matrix))
          (mattranslation (double-matrix))
          (s (double-matrix))(a (double-matrix))
          (u (double-matrix))(ur (double-matrix))
          (fr (double-matrix))(matcov (double-matrix))
          (clusters (double-matrix)); matriz com os clusters
          (linhafr (double-matrix))(linhafrtr (double-matrix))
          (matresult (double-matrix))(vetclusters (double-matrix))
          (clusteraux (double-matrix))(cluster1 (double-matrix))
          (cluster2 (double-matrix))(cluster3 (double-matrix))
          (matfeaturestreinamento (double-matrix))
          (quantdadosfr 0)
          (quantdadosfrtr 0)(quantfeatures 0)(nomefeatures "")
          (aux1 0.0)
          (lresult ())
          (tipoPadraoMediaMedia "")
          )

      (cond
         ((= TipoAnaliseResultado 0)
            (setq tipoPadraoMediaMedia "-Padrao"))
         ((= TipoAnaliseResultado 1)
            (setq tipoPadraoMediaMedia "-Media"))
         ((= TipoAnaliseResultado 2)
            (setq tipoPadraoMediaMedia "-Mediana"))
      )

      (cond
```

```
         ((= tipodados "treinamento")
           (setq inifiles 0)
           (setq qdados1 qdados1anotr)
           (setq qdados2 qdadosurgtr)
           (setq qdados3 qdadosemertr)
         )
         ((= tipodados "teste")
           (setq inifiles 15)
           (setq qdados1 qdados1anote)
           (setq qdados2 qdadosurgte)
           (setq qdados3 qdadosemerte)
         )
         ((= tipodados "comparacao")
           (setq inifiles 15)
           (setq qdados1 qdados1anocp)
           (setq qdados2 qdadosurgcp)
           (setq qdados3 qdadosemercp)
         ))
           (setq qdados1 qdados1anotr)
           (setq qdados2 qdadosurgtr)
           (setq qdados3 qdadosemertr)

;mudando o tamanho da amostra, caso seja necessario
    (if (= dirtipo "manual")
       (setq samplesize (* multsamplesize samplesize)))

 (if (= modo 0)
      (setq nomefeatures "-SEMPCA")
     (setq nomefeatures "-COMPCA"))

   (if (= (tipofeatures 0) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numSPECTRO))
        (setq nomefeatures (concat nomefeatures "-FFT"))))
   (if (= (tipofeatures 1) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numLPC))
        (setq nomefeatures (concat nomefeatures "-LPC"))))
   (if (= (tipofeatures 2) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numMFCC))
        (setq nomefeatures (concat nomefeatures "-MFCC"))))
   (if (= (tipofeatures 3) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numZeros))
        (setq nomefeatures (concat nomefeatures "-ZC"))))
   (if (= (tipofeatures 4) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numENERGY))
        (setq nomefeatures (concat nomefeatures "-En"))))
   (if (= (tipofeatures 5) 1)
      (progn
        (setq quantfeatures (+ quantfeatures 1))
        (setq nomefeatures (concat nomefeatures "-AREA"))))

;obtendo o valor de dr
      (setq matdef (loadtxt  (concat dirprog "/sanidades/K" (str K) "-
definicao" nomefeatures "-"    dirtipo ".txt")))
      (setq DR (matdef 1 0))
      (setq matfeatures (geramatrizfeatures dircomplemento inifiles modo
filtrar));processa os vetores de dados armazenados
```

```
      (printf "Calculando distância \n")
      (writetxt (concat dirprog "/sanidades/K" (str K) "-matfeaturesteste"
nomefeatures "-" dirtipo ".txt") matfeatures)
    (if (= modo 1)
     (progn
      (load-matrix mattranslation (concat dirprog "/sanidades/K" (str K)
"-dr" (str dr) "-mattranslation" nomefeatures "-" dirtipo ".dat"))

      (setq matfeaturesfr (transpose (multmat (transpose mattranslation)
(transpose matfeatures))));features reduzidas
      (writetxt (concat dirprog "/sanidades/K" (str K) "-dr" (str dr) "-
matfeaturesreduzidasteste" nomefeatures "-" dirtipo ".txt") matfeaturesfr)
     )
     (progn
       (setq matfeaturesfr matfeatures)
       (setq DR (idx-dim matfeatures 1))))

;obter a matriz de features reduzidas obtidas no processo de treinamento
      (load-matrix matfeaturestreinamento (concat dirprog "/sanidades/K"
(str K) "-dr" (str DR) "-matfeaturesreduzidas" nomefeatures "-" dirtipo
".dat"))

;comparar cada linha da matriz de features reduzidas de teste com toda a
matriz reduzida do treinamento
;e obter a qual linha desta matriz cada elemento da matriz reduzida de
teste tem menor distancia

      (setq quantdadosfr   (idx-dim matfeaturesfr          0))
      (setq quantdadosfrtr (idx-dim matfeaturestreinamento 0))
      (setq matd1 (double-matrix quantdadosfr quantdadosfrtr))
      (setq matd2 (double-matrix quantdadosfr quantdadosfrtr))
      (setq clusters (double-matrix quantdadosfr))
      (if (> quantdadosfr 0)
        (progn
          (for (i 0 (1- quantdadosfr))
             (setq linhafr (extrailinhamatriz matfeaturesfr i))
             (for (j 0 (1- quantdadosfrtr))
               ;calcula a distancia de um elemento da matriz de features
para cada elemento da matriz de treinametno
                 (setq linhafrtr (extrailinhamatriz matfeaturestreinamento
j))
                 (matd1 i j (disteuclidiana linhafr linhafrtr));distancia
entre o elemento de teste com cada posição da matriz fr de treinamento
                 (matd2 i j j)
              )
          )
;           (writetxt (concat dirprog "/sanidades/KNN" (str KNN) "-dr"
(str DR) "-resultadomatd1A.txt") matd1)
;           (writetxt (concat dirprog "/sanidades/KNN" (str KNN) "-dr"
(str DR) "-resultadomatd2A.txt") matd2)
          ;ordenar a matriz matd1 em ordem crescente e associar os valores
de matd2 com esta ordenação para termos os indices dos dados. O resultado
será a matriz matd2
          (for (ii 0 (1- quantdadosfr))
             (setq linhad1 (extrailinhamatriz matd1 ii));obtendo uma linha
da matriz matd1
             (setq linhad2 (extrailinhamatriz matd2 ii));obtendo uma linha
da matriz matd2
             ;ordenando a linha de matd1
             (setq lresult (ordenavetormenor linhad1))
             (for (i 0 (1- quantdadosfrtr))
```

```
                (matd1 ii i ((nth lresult 1) i))
                (matd2 ii i ((nth lresult 2) i)))
          )
         ;dados ordenados
;          (writetxt (concat dirprog "/sanidades/KNN" (str KNN) "-dr" (str
DR) "-resultadomatd1B.txt") matd1)
;          (writetxt (concat dirprog "/sanidades/KNN" (str KNN) "-dr" (str
DR) "-resultadomatd2B.txt") matd2)
         (for (ii 0 (1- quantdadosfr))
            (for (i 0 (1- quantdadosfrtr))
               ;testa se pertence a sanidade 1 ANO - menor que QDADOS1
                  (if (< (matd2 ii i) qdados1)
                      (matd2 ii i 1)
               ;testa se pertence a sanidade URGENCIAL - ENTRE QDADOS1 E
QDADOS2
                   (if (and (< (matd2 ii i) (+ qdados1 qdados2)) (>=
(matd2 ii i) qdados1))
                       (matd2 ii i 2)
               ;testa se pertence a sanidade EMERGENCIAL - MAIOR QUE
QDADOS2 E MENOR QUE QDADOS3
                    (if (and (< (matd2 ii i) (+ qdados1 qdados2 qdados3))
(>= (matd2 ii i) (+ qdados1 qdados2)))
                        (matd2 ii i 3)
                       (matd2 ii i -1))))
            )
          )
         (for (i 0 (1- quantdadosfr))
            (clusters i (procuramaiorocorrenciaNew (extrailinhamatriz
matd2 i) KNN TipoAnaliseResultado)))
         ;salvando as informações
         (writetxt2 (concat dirprog "/sanidades/KNN" (str KNN) "-dr" (str
DR) tipoPadraoMediaMedia "-resultado" nomefeatures "-" tipodados "-"
dirtipo ".txt") clusters listacruzetas)
         (writetxt (concat dirprog "/sanidades/KNN1" (str KNN)
tipoPadraoMediaMedia "-dr" (str DR) "-resultadoMATRIZORDENADA"
nomefeatures "-" tipodados "-" dirtipo ".txt") matd1)

         (writetxt (concat dirprog "/sanidades/KNN2"  (str KNN) "-dr"
(str DR) tipoPadraoMediaMedia "-resultadoMATRIZORDENADA" nomefeatures "-"
tipodados "-" dirtipo ".txt") matd2)
;;;---gerando arquivo de log para o reconhecimento (teste de resultados)
         (if (= tipodados "teste")
            (geraLOG (concat dirprog "/sanidades/KNN" (str KNN)
tipoPadraoMediaMedia "-dr" (str DR) "-resultado" nomefeatures "-"
tipodados "-" dirtipo ".log") clusters nomefeatures KNN))
      ))
))


;;=-----------------------------------------------------------
; TESTA O SINAL UTILIZANDO KNN
; PARÂMETROS:
;    modo : 0 - sem pca 1 - com pca
;    knn números de proximidades
;    filtrar : executar ou não o filtro nos sinais (0 - NÃO / 1 - SIM)
(defun testasinalnewKNN2 (modo KNN filtrar tipoSan tipoFeatures
tipoAnaliseResultado)
   (let* ((DR 0) ;redução de features
         (K  1)   ;quantidade de centroides
         (matdef (double-matrix))
```

```
          (matd1 (double-matrix)) ; vetor com as distâncias euclidianas de
todas as analises
          (matd2 (double-matrix)) ; vetor com os indices da posições de
todas as analises
          (inifiles 0)
          (dircomplemento (concat "/" dirtipo "/" tipodados))
          (namefile "")
          (dirsanidade "")
          (listasanidade ())
          (nlist1 0.0)
          (quantsanidade 0.0)
          (clusters (double-matrix)); matriz com os clusters

          (quantdados 0)
          (quantdadosfrtr 0)(quantfeatures 0)(nomefeatures "")
          (aux 0.0)
          (lresult ())
          (tipoPadraoMediaMedia "")
          )

      (cond
         ((= TipoAnaliseResultado 0)
            (setq tipoPadraoMediaMedia "-Padrao"))
         ((= TipoAnaliseResultado 1)
            (setq tipoPadraoMediaMedia "-Media"))
         ((= TipoAnaliseResultado 2)
            (setq tipoPadraoMediaMedia "-Mediana"))
      )

;mudando o tamanho da amostra, caso seja necessario
    (if (= dirtipo "manual")
       (setq samplesize (* multsamplesize samplesize)))

 (if (= modo 0)
      (setq nomefeatures "-SEMPCA")
     (setq nomefeatures "-COMPCA"))

   (if (= (tipofeatures 0) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numSPECTRO))
        (setq nomefeatures (concat nomefeatures "-FFT"))))
   (if (= (tipofeatures 1) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numLPC))
        (setq nomefeatures (concat nomefeatures "-LPC"))))
   (if (= (tipofeatures 2) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numMFCC))
        (setq nomefeatures (concat nomefeatures "-MFCC"))))
   (if (= (tipofeatures 3) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numZeros))
        (setq nomefeatures (concat nomefeatures "-ZC"))))
   (if (= (tipofeatures 4) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numENERGY))
        (setq nomefeatures (concat nomefeatures "-En"))))
   (if (= (tipofeatures 5) 1)
      (progn
        (setq quantfeatures (+ quantfeatures 1))
        (setq nomefeatures (concat nomefeatures "-AREA"))))
```

```
  (setq clusters (double-matrix (+ qdados1anote qdadosurgte
qdadosemerte)))
  (setq listacruzetas (list ""))
  (setq quantdados 0)

; entrar em cada diretório
   ;ler cada arquivo wav
   ;testar a qual sanidade pertence cada amostra
      ;obter as amostras para a sanidade sadia
        (setq dirsanidade (concat dirsan1ano dircomplemento))
        (setq listasanidade (files dirsanidade))
      ;obter a quantidade de elementos da lista
        (setq nlist1 (length listasanidade))
      ;armazenar a quantidade de amostras para a sanidade sadia
        (setq quantsanidade 0)
      ;processar cada amostra
        (for (i 3 nlist1)
          (setq namefile (concat (concat dirsanidade "/") (nth
listasanidade i))) ;obter o nome do arquivo
          (if (<> (index ".wav" namefile) ());testa se é um arquivo do
tipo WAV
           (progn
             (setq quantsanidade (+ quantsanidade 1))
;processar o sinal
             (setq dadosreconhecidos (reconheceamostraKNNNew KNN filtrar
tipoSan namefile tipoFeatures modo tipoAnaliseResultado))

           ;obtem o vetor de features
          ; (setq vetfeaturessadia (obtemfeatures m1 tiponormal modo))
           ;adiciona o vetor de features para a matriz de features para a
sanidade
          ; (setq matfeatures (addlinvetmat matfeatures vetfeaturessadia))
             (printf "================================\n")
             (printf "%d - %d - %s \n" quantdados (nth dadosreconhecidos
1) namefile)
             (printf "================================\n")
            ;insere o nome do arquivo na lista de nomes de cruzetas
processadas
             (list-insert listacruzetas (length listacruzetas) namefile)
            ;adiciona matd1 e matd2
             (setq matd1 (addlinvetmat matd1 (nth dadosreconhecidos 5)))
             (setq matd2 (addlinvetmat matd2 (nth dadosreconhecidos 6)))
            ;insere o resultado no vetor de clusters
             (clusters quantdados    (nth dadosreconhecidos 1))
             (setq quantdados (1+ quantdados))
           ))

         )
        (printf
"=====================================================================\n")
        (printf "QUANTIDADE DE DADOS SANIDADE 1 ANO: %d %d \n"
quantsanidade quantdados)
        (printf
"=====================================================================\n")
;---------------------------------------------------------------------
      ;obter as amostras para a sanidade urgente
        (setq dirsanidade (concat dirsanurg dircomplemento))
        (setq listasanidade (files dirsanidade))
      ;obter a quantidade de elementos da lista
        (setq nlist1 (length listasanidade))
```

```
      ;armazenar a quantidade de amostras para a sanidade sadia
        (setq quantsanidade 0)
      ;processar cada amostra
        (for (i 3 nlist1)
          (setq namefile (concat (concat dirsanidade "/") (nth
listasanidade i))) ;obter o nome do arquivo
          (if (<> (index ".wav" namefile) ());testa se é um arquivo do
tipo WAV
           (progn
             (setq quantsanidade (+ quantsanidade 1))
;processar o sinal
             (setq dadosreconhecidos (reconheceamostraKNNNew KNN filtrar
tipoSan namefile tipoFeatures modo tipoAnaliseResultado))

           ;obtem o vetor de features
          ; (setq vetfeaturessadia (obtemfeatures m1 tiponormal modo))
           ;adiciona o vetor de features para a matriz de features para a
sanidade
          ; (setq matfeatures (addlinvetmat matfeatures vetfeaturessadia))
             (printf "================================\n")
             (printf "%d - %d - %s \n" quantdados (nth dadosreconhecidos
1) namefile)
             (printf "================================\n")
            ;insere o nome do arquivo na lista de nomes de cruzetas
processadas
             (list-insert listacruzetas (length listacruzetas) namefile)
            ;adiciona matd1 e matd2
             (setq matd1 (addlinvetmat matd1 (nth dadosreconhecidos 5)))
             (setq matd2 (addlinvetmat matd2 (nth dadosreconhecidos 6)))
            ;insere o resultado no vetor de clusters
             (clusters quantdados    (nth dadosreconhecidos 1))
             (setq quantdados (1+ quantdados))
           ))

         )
        (printf
"=====================================================================\n")
        (printf "QUANTIDADE DE DADOS SANIDADE URGENCIAL: %d %d \n"
quantsanidade quantdados)
        (printf
"=====================================================================\n")
;---------------------------------------------------------------------------
      ;obter as amostras para a sanidade urgente
        (setq dirsanidade (concat dirsanemer dircomplemento))
        (setq listasanidade (files dirsanidade))
      ;obter a quantidade de elementos da lista
        (setq nlist1 (length listasanidade))
      ;armazenar a quantidade de amostras para a sanidade sadia
        (setq quantsanidade 0)
      ;processar cada amostra
        (for (i 3 nlist1)
          (setq namefile (concat (concat dirsanidade "/") (nth
listasanidade i))) ;obter o nome do arquivo
          (if (<> (index ".wav" namefile) ());testa se é um arquivo do
tipo WAV
           (progn
             (setq quantsanidade (+ quantsanidade 1))
;processar o sinal
             (setq dadosreconhecidos (reconheceamostraKNNNew KNN filtrar
tipoSan namefile tipoFeatures modo tipoAnaliseResultado))
```

```
          ;obtem o vetor de features
         ; (setq vetfeaturessadia (obtemfeatures m1 tiponormal modo))
          ;adiciona o vetor de features para a matriz de features para a
sanidade
         ; (setq matfeatures (addlinvetmat matfeatures vetfeaturessadia))
            (printf "================================\n")
            (printf "%d - %d - %s \n" quantdados (nth dadosreconhecidos
1) namefile)
            (printf "================================\n")
           ;insere o nome do arquivo na lista de nomes de cruzetas
processadas
            (list-insert listacruzetas (length listacruzetas) namefile)
           ;adiciona matd1 e matd2
            (setq matd1 (addlinvetmat matd1 (nth dadosreconhecidos 5)))
            (setq matd2 (addlinvetmat matd2 (nth dadosreconhecidos 6)))
           ;insere o resultado no vetor de clusters
            (clusters quantdados    (nth dadosreconhecidos 1))
            (setq quantdados (1+ quantdados))
          ))

        )
        (printf
"=====================================================================\n")
        (printf "QUANTIDADE DE DADOS SANIDADE EMERGENCIAL: %d %d \n"
quantsanidade quantdados)
        (printf
"=====================================================================\n")
;---------------------------------------------------------------------
         ;salvando as informações
          (writetxt2 (concat dirprog "/sanidades/KNN" (str KNN) "-dr" (str
DR) tipoPadraoMediaMedia "-resultado" nomefeatures "-" tipodados "-"
dirtipo ".txt") clusters listacruzetas)
          (writetxt (concat dirprog "/sanidades/KNN1" (str KNN)
tipoPadraoMediaMedia "-dr" (str DR) "-resultadoMATRIZORDENADA"
nomefeatures "-" tipodados "-" dirtipo ".txt") matd1)

          (writetxt (concat dirprog "/sanidades/KNN2"  (str KNN) "-dr"
(str DR) tipoPadraoMediaMedia "-resultadoMATRIZORDENADA" nomefeatures "-"
tipodados "-" dirtipo ".txt") matd2)

;;;---gerando arquivo de log para o reconhecimento (teste de resultados)
          (if (= tipodados "teste")
             (geraLOG (concat dirprog "/sanidades/KNN" (str KNN)
tipoPadraoMediaMedia "-dr" (str DR) "-resultado" nomefeatures "-"
tipodados "-" dirtipo ".log") clusters nomefeatures KNN))

;      ))
    (list clusters matd1 matd2)
))

;---------------------------------------------------------------------
;-----GERAÇÃO DE ARQUIVO DE LOG COM RESULTADOS DE RECONHECIMENTO----------
;---------------------------------------------------------------------
(defun geraLOG(arquivo vetordados nomefeatures KNN)
  (let* ((dadostotais (+ qdados1anote qdadosurgte qdadosemerte))
         (i 0)
         (acerto1ano 0)(acertourgente 0)(acertoemergencial 0) ;acertos das
amostras
         (reconhece1anourgente 0)(reconhece1anoemergencial 0)
;reconhecimentos como outro tipo de amostras para 1 ano
```

```
         (reconheceurgente1ano 0)(reconheceurgenteemergencial 0)
;recohecimentos como outro tipo de amostras para urgente
         (reconheceemergencial1ano 0)(reconheceemergencialurgente 0)
;reconhecimentos como outro tipo de amostas para emergencial
        )

   ;imprimindo o nome das features utilizadas

   ;calculando quantidades de erros e acertos
      (for (i 0 (1- dadostotais))
       (if (< i qdados1anote);testa os resultados para 1ANO
        (progn
         (cond
             ((= (vetordados i) 1)
                (setq acerto1ano (+ acerto1ano 1))
             )
             ((= (vetordados i) 2)
                (setq reconhece1anourgente (+ reconhece1anourgente 1))
             )
             ((= (vetordados i) 3)
                (setq reconhece1anoemergencial (+ reconhece1anoemergencial
1))
             )
         )
        ))
       (if (and (>= i qdados1anote) (< i (+ qdados1anote qdadosurgte)))
        (progn
         (cond
             ((= (vetordados i) 1)
                (setq reconheceurgente1ano (+ reconheceurgente1ano 1))
             )
             ((= (vetordados i) 2)
                (setq acertourgente (+ acertourgente 1))
             )
             ((= (vetordados i) 3)
                (setq reconheceurgenteemergencial (+
reconheceurgenteemergencial 1))
             )
         )
        ))
       (if (and (>= i (+ qdados1anote qdadosurgte)) (< i (+ qdados1anote
qdadosurgte qdadosemerte)))
        (progn
         (cond
             ((= (vetordados i) 1)
                (setq reconheceemergencial1ano (+ reconheceemergencial1ano
1))
             )
             ((= (vetordados i) 2)
                (setq reconheceemergencialurgente (+
reconheceemergencialurgente 1))
             )
             ((= (vetordados i) 3)
                (setq acertoemergencial (+ acertoemergencial 1))
             )
         )
        ))
      )
     (writing arquivo
      ;imprimindo os resultados no arquivo de log
       (printf "Feature = %s \n" nomefeatures)
```

```
       (printf "Quantidade de Vizinhos K-NN = %d\n" KNN)
      ;-------------------------------------------------------------------
       (printf "TIPO DE AMOSTRAS = 1 ANO \n")
       (printf "QUANTIDADE TOTAL DE AMOSTRAS = %d\n" (+ qdados1anotr
qdados1anote))
       (printf "QUANTIDADE DE AMOSTRAS DE TREINAMENTO = %d\n"
qdados1anotr)
       (printf "QUANTIDADE DE AMOSTRAS DE RECONHECIMENTO = %d\n"
qdados1anote)
       (printf "ACERTOS = %d - %.2f\n" acerto1ano (/ (* acerto1ano 100)
qdados1anote))
       (printf "RECONHECIDAS COMO URGENTE = %d - %.2f\n"
reconhece1anoUrgente (/ (* reconhece1anoUrgente 100) qdados1anote))
       (printf "RECONHECIDAS COMO EMERGENCIAL = %d - %.2f\n\n"
reconhece1anoEmergencial (/ (* reconhece1anoEmergencial 100)
qdados1anote))
      ;-------------------------------------------------------------------
       (printf "TIPO DE AMOSTRAS = URGENTE \n")
       (printf "QUANTIDADE TOTAL DE AMOSTRAS = %d\n" (+ qdadosurgtr
qdadosurgte))
       (printf "QUANTIDADE DE AMOSTRAS DE TREINAMENTO = %d\n" qdadosurgtr)
       (printf "QUANTIDADE DE AMOSTRAS DE RECONHECIMENTO = %d\n"
qdadosurgte)
       (printf "ACERTOS = %d - %.2f\n" acertourgente (/ (* acertourgente
100) qdadosurgte))
       (printf "RECONHECIDAS COMO 1 ANO = %d - %.2f\n"
reconheceUrgente1ano (/ (* reconheceUrgente1ano 100) qdadosurgte))
       (printf "RECONHECIDAS COMO EMERGENCIAL = %d - %.2f\n\n"
reconheceUrgenteEmergencial (/ (* reconheceUrgenteEmergencial 100)
qdadosurgte))
      ;-------------------------------------------------------------------
       (printf "TIPO DE AMOSTRAS = EMERGENCIAL \n")
       (printf "QUANTIDADE TOTAL DE AMOSTRAS = %d\n" (+ qdadosemertr
qdadosemerte))
       (printf "QUANTIDADE DE AMOSTRAS DE TREINAMENTO = %d\n"
qdadosemertr)
       (printf "QUANTIDADE DE AMOSTRAS DE RECONHECIMENTO = %d\n"
qdadosemerte)
       (printf "ACERTOS = %d - %.2f\n" acertoemergencial (/ (*
acertoemergencial 100) qdadosemerte))
       (printf "RECONHECIDAS COMO 1 ANO = %d - %.2f\n"
reconheceEmergencial1ano (/ (* reconheceEmergencial1ano 100)
qdadosemerte))
       (printf "RECONHECIDAS COMO URGENCIAL = %d - %.2f\n\n"
reconheceEmergencialUrgente (/ (* reconheceEmergencialUrgente 100)
qdadosemerte))
     )

  )
)


;-------------------------------------------------------------------------
;------RECONHECIMENTO DE UMA ÚNICA BATIDA DO MARTELO ELETRÔNICO-----------
;-------------------------------------------------------------------------
;PARÂMETROS:
;knn : números de proximidades
;filtrar : executar ou não o filtro nos sinais (0 - NÃO / 1 - SIM)
(defun reconheceamostraKNN (knn filtrar)
   (let* ((DR 0) ;redução de features
          (K 3)   ;quantidade de centroides
```

```
           (modo 0) ;análise sem PCA
           (qdados1 0)(qdados2 0)(qdados3 0)
           (matdef (double-matrix))
           (matd1 (double-matrix)) ; vetor com as distâncias euclidianas
           (matd2 (double-matrix)) ; vetor com os indices da posições
           (linhad1 (double-matrix))
           (linhad2 (double-matrix))
           (inifiles 0)
           (dircomplemento (concat "/" dirtipo "/" tipodados))
           (namefile "")(nametxt "")
           (deslocdatamat 0)(m1 (double-matrix))(vetfeatures (double-
matrix))
           (maudio (double-matrix))
           (matfeatures (double-matrix))
           (v (double-matrix))
           (mattranslation (double-matrix))
           (s (double-matrix))          (a (double-matrix))
           (u (double-matrix))          (ur (double-matrix))
           (fr (double-matrix))         (matcov (double-matrix))
           (clusters (double-matrix)); matriz com os clusters
           (linhafr (double-matrix))    (linhafrtr (double-matrix))
           (matresult (double-matrix))  (vetclusters (double-matrix))
           (clusteraux (double-matrix)) (cluster1 (double-matrix))
           (cluster2 (double-matrix))   (cluster3 (double-matrix))
           (matfeaturestreinamento (double-matrix))
           (quantdadosfr 0)
           (quantdadosfrtr 0)(quantfeatures 0)(nomefeatures "")
           (aux1 0.0)
           (lresult ())
           (imagemcamera (concat dirprog "/tmp.jpg"))
           (videocard   0)
           (devvideo (concat "/dev/video" (str videocard)))
           (tipocaptura 1) ;martelo manual
           (vetfeatures (double-matrix))
           (matfeatures (double-matrix))
           (tipodados "reconhecimentoAMOSTRA")
           (tinicio ()) (tfinal ())
           (lptcard ());fechando a porta parport0
           (lptcard (new parport lptdevice)) ;abrindo a porta parport0
           (tipofeatures [0 0 1 0 0 0])
           (ttotalreconhecimento 0)
           (ttotalgeral 0)
           (m6t ())
           (desloc 0)
           (dl 0)
           (i 0)(j 0)
           (resultadoFINAL ())
           )

(setq ttotalgeral
  (realtime ;inicia a marcação do tempo total de processamento da função
;definir a quantidade de dados de referencia para cada sanidade
   (printf "Comecando o processo \n")
           (setq qdados1 qdados1anotr)
           (setq qdados2 qdadosurgtr)
           (setq qdados3 qdadosemertr)

;mudando o tamanho da amostra, caso seja necessario
    (if (= dirtipo "manual")
       (setq samplesize (* multsamplesize samplesize)))
```

```
 (if (= modo 0)
      (setq nomefeatures "-SEMPCA")
     (setq nomefeatures "-COMPCA"))

   (if (= (tipofeatures 0) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numSPECTRO))
        (setq nomefeatures (concat nomefeatures "-FFT"))))
   (if (= (tipofeatures 1) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numLPC))
        (setq nomefeatures (concat nomefeatures "-LPC"))))
   (if (= (tipofeatures 2) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numMFCC))
        (setq nomefeatures (concat nomefeatures "-MFCC"))))
   (if (= (tipofeatures 3) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numZeros))
        (setq nomefeatures (concat nomefeatures "-ZC"))))
   (if (= (tipofeatures 4) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numENERGY))
        (setq nomefeatures (concat nomefeatures "-En"))))
   (if (= (tipofeatures 5) 1)
      (progn
        (setq quantfeatures (+ quantfeatures 1))
        (setq nomefeatures (concat nomefeatures "-AREA"))))

(printf "Lendo arquivo tmp.wav \n")

;obtendo o valor de dr
      (setq matdef (loadtxt  (concat dirprog "/sanidades/K" (str K) "-
definicao" nomefeatures "-"    dirtipo ".txt")))
      (setq DR (matdef 1 0))

;-------------------------------------------------------
;----DISPARAR O MARTELO E GRAVAR O ARQUIVO DE AUDIO
;-------------------------------------------------------
;-----------------------------------------------------------------
;ATIVAR AQUI PARA USAR PORTA PARALELA
;;alterar o volume do canal de entrada linha 0
      (if (= tipocaptura 0)
        ;;batida com marreta
        (sys (concat "amixer set Capture " nivelmartelomanual " unmute"))
        ;;batida com martelo automático
        (sys (concat "amixer set Capture " nivelmarteloautomatico "
unmute")))

        ;;deletar o arquivo tmp.wav caso ele exista
        (if (<> (filepath "tmp.wav") ())
           (sh (concat "rm " (filepath "tmp.wav"))))
        ;; Disparar a leitura de audio
        (setq fsamostra 44100); define a taxa de amostragem
        (rec-sound-2 fsamostra quantamostra devaudio)
        ;aguarda para começar a gravar
        (sleep 1)
        ;;disparar o martelo se estiver no modo automático

        (if (= tipocaptura 1)
```

```
           (==> lptcard pulse 1 timelpt 0);força a ficar um tempo em 0 e
depois em 1
           )
                  ;aguarda o tempo do arquivo de som
        (sleep (/ quantamostra fsamostra))
;-------------------------------------------------------
;-------------------------------------------------------
;-----   LER O ARQUIVO DE AUDIO
         ;ler o arquivo de som
                (setq datawave (loadwave "tmp.wav" 0))
         ;extrair o primeiro canal do arquivo de som
                (setq maudio  (extraicanalwave datawave 0))
         ;obter a taxa de amostragem do arquivo de som
                (setq srate (extraitx datawave))
         ;detectar a quantidade de amostras a serem "puladas" para obter
somente o sinal
                (setq m6t (extraidadossanidade maudio 0 srate sizejanenerg
samplesize))
                (setq deslocdatamat (nth m6t 2));quantidade de amostras a
serem deslocadas
         ;captura camera de video
                (sh (concat (concat (concat "fswebcam -r 400x300 --jpeg
95 -D 0 -q --frames 30 " imagemcamera) " -d ") devvideo))


    ;;------PROCESSA O SINAL CASO ELE SEJA ACEITÁVEL
     (if (= deslocdatamat 0)
        (progn

           (setq resultadoFINAL (list  0 0.0 0.0 maudio))
           (printf "sinal invalido\n\n")

        )
  ;
        (progn

         ;obter as features do sinal de audio
                (setq m1 (preprocessasinal maudio srate deslocdatamat
tiponormalwave filtrar)) ; pré-processamento do sinal
              (setq vetfeatures (obtemfeatures m1 tiponormal modo))
         ;armazenar as features do sinal de som na matriz de features
;-------------------------------------------------------------
;-----------FIM DA LEITURA DO ARQUIVO DE ÁUDIO--------------
;-------------------------------------------------------------
;    (printf "Calculando distância \n")

(printf "Iniciando reconhecimento \n")
(setq ttotalreconhecimento
   (realtime ;inicia a marcação do tempo de processamento somente da parte
de reconhecimento

    (if (= modo 1)
     (progn
      (load-matrix mattranslation (concat dirprog "/sanidades/K" (str K)
"-dr" (str dr) "-mattranslation" nomefeatures "-" dirtipo ".dat"))

       (setq matfeaturesfr (transpose (multmat (transpose mattranslation)
(transpose matfeatures))));features reduzidas
     )
     (progn
        (setq DR (idx-dim vetfeatures 0))))
```

```
;obter a matriz de features reduzidas obtidas no processo de treinamento
       (load-matrix matfeaturestreinamento (concat dirprog "/sanidades/K"
(str K) "-dr" (str DR) "-matfeaturesreduzidas" nomefeatures "-" dirtipo
".dat"))

;comparar cada linha da matriz de features reduzidas de teste com toda a
matriz reduzida do treinamento
;e obter a qual linha desta matriz cada elemento da matriz reduzida de
teste tem menor distancia

(printf "Processando o reconhecimento \n")
      (setq quantdadosfr 1)
      (setq quantdadosfrtr (idx-dim matfeaturestreinamento 0))
      (setq matd1 (double-matrix quantdadosfr quantdadosfrtr))
      (setq matd2 (double-matrix quantdadosfr quantdadosfrtr))
      (setq clusters (double-matrix quantdadosfr))
      (if (> quantdadosfr 0)
        (progn
              (for (j 0 (1- quantdadosfrtr))
               ;calcula a distancia de um elemento da matriz de features
para cada elemento da matriz de treinametno
                 (setq linhafrtr (extrailinhamatriz matfeaturestreinamento
j))
                 (matd1 i j (disteuclidiana vetfeatures
linhafrtr));distancia entre o elemento de teste com cada posição da matriz
fr de treinamentopreprocessasinal
                 (matd2 i j j) ;indice do elemento
              )
;ordenar a matriz matd1 em ordem crescente e associar os valores de matd2
com esta ordenação para termos os indices dos dados. O resultado será a
matriz matd2
             (setq linhad1 (extrailinhamatriz matd1 0));obtendo uma linha
da matriz matd1
             (setq linhad2 (extrailinhamatriz matd2 0));obtendo uma linha
da matriz matd2
             ;ordenando a linha de matd1
             (setq lresult (ordenavetormenor linhad1))
             (for (i 0 (1- quantdadosfrtr))
                 (matd1 ii i ((nth lresult 1) i))
                 (matd2 ii i ((nth lresult 2) i)))
          ;dados ordenados - matrizes matd1 ordenada e matd2 com os
indices referentes a ordenação de matd1
           (for (i 0 (1- quantdadosfrtr))
                ;testa se pertence a sanidade 1 ANO - menor que QDADOS1
                   (if (< (matd2 0 i) qdados1)
                       (matd2 0 i 1)
                ;testa se pertence a sanidade URGENCIAL - ENTRE QDADOS1 E
QDADOS2
                    (if (and (< (matd2 0 i) (+ qdados1 qdados2)) (>=
(matd2 0 i) qdados1))
                          (matd2 0 i 2)
                ;testa se pertence a sanidade EMERGENCIAL - MAIOR QUE
QDADOS2 E MENOR QUE QDADOS3
                     (if (and (< (matd2 0 i) (+ qdados1 qdados2 qdados3))
(>= (matd2 0 i) (+ qdados1 qdados2)))
                           (matd2 0 i 3)
                          (matd2 0 i -1))))
             )
;armazenar ao qual sanidade o sinal pertence (1 - 1 ANO / 2 - URGENTE / 3
- EMERGENCIAL
```

```
          (clusters 0 (procuramaiorocorrencia (extrailinhamatriz matd2 0)
KNN))
          ;salvando as informações

(printf "Gerando arquivo de saida \n")

          (writetxt2 (concat dirprog "/sanidades/KNN" (str KNN) "-dr" (str
DR) "-resultado" nomefeatures "-" tipodados "-" dirtipo ".txt") clusters
(list "nada" "tmp.wav")) ;listacruzetas)
      ))
 ));;--FIM ttotalreconhecimento

 (printf "Finalizando... \n")
  (setq resultadoFINAL(list (clusters 0) ttotalgeral ttotalreconhecimento
maudio));retorna a sanidade(1 - 1 Ano, 2 - URGENCIAL, 3 - EMERGENCIAL),
tempo total de processamento da função, tempo total somente da parte de
reconhecimento e o arquivo de audio gravado


  );;fim do if referente ao teste se o sinal é válido

 )

));;--FIM ttotalgeral


;  );;fim do if referente ao teste se o sinal é válido

  resultadoFINAL

))

;---------RECONHECE KNN NEW
;-------------------------------------------------------------------------
;------RECONHECIMENTO DE UMA ÚNICA BATIDA DO MARTELO ELETRÔNICO-----------
;-------------------------------------------------------------------------
;PARÂMETROS:
; knn     : números de proximidades
; filtrar : executar ou não o filtro nos sinais (0 - NÃO / 1 - SIM)
; tipoSan : tipos de sanidade a serem utilizadas
;           É um vetor do tipo [1 1 1] onde a primeira posição refere-se a
sanidade 1ANO
;           A segunda posição refere-se a sanidade URGENCIAL e a terceira
refere-se a
;           EMERGENCIAL.
;           Ex.: [1 0 1] - irá testar o sinal com as sanidades 1Ano e
Emergencial
;                [1 1 0] - irá testar o sinal com as sanidades 1Ano e
Urgencial
;                [0 1 1] - irá testar o sinal com as sanidades Urgencial e
Emergencial
;                [1 1 1] - irá testar o sinal com as três sanidades
;           OBS.: É OBRIGATÓRIO QUE PELO MENOS 2 SANIDADES SEJAM TESTADAS
; arqWave : arquivo com o sinal a ser processado (monocanal)
; tipoFeatures : vetor contendo os tipos de features a serem processados
;                [FFT LPC MFCC ZEROCROSSING ENERGIA AREA]
;                Ex.: [ 1 0 0 0 0 0] - será processada somente a FFT
; modo    : indica se será utilizada ou não PCA no reconhecimento (0 - SEM
PCA / 1 - COM PCA)
; tipoAnaliseResultado - define se: 0 - normal / 1 - Media / 2 - Mediana
;
```

```
; RETORNO:
(defun reconheceamostraKNNNEW (knn filtrar tipoSan arqWave tipoFeatures
modo tipoAnaliseResultado)
   (let* ((DR 0) ;redução de features
          (K 1)   ;quantidade de centroides

          (qdados1 0)(qdados2 0)(qdados3 0)
          (matdef (double-matrix))
          (matd1 (double-matrix)) ; vetor com as distâncias euclidianas
          (matd2 (double-matrix)) ; vetor com os indices da posições
          (linhad1 (double-matrix))
          (linhad2 (double-matrix))
          (inifiles 0)
          (dircomplemento (concat "/" dirtipo "/" tipodados))
          (namefile "")(nametxt "")
          (deslocdatamat 0)(m1 (double-matrix))(vetfeatures (double-
matrix))
          (maudio (double-matrix))
          (matfeatures (double-matrix))
          (v (double-matrix))
          (mattranslation (double-matrix))
          (s (double-matrix))          (a (double-matrix))
          (u (double-matrix))          (ur (double-matrix))
          (fr (double-matrix))         (matcov (double-matrix))
          (clusters (double-matrix)); matriz com os clusters
          (linhafr (double-matrix))    (linhafrtr (double-matrix))
          (matresult (double-matrix))  (vetclusters (double-matrix))
          (clusteraux (double-matrix)) (cluster1 (double-matrix))
          (cluster2 (double-matrix))   (cluster3 (double-matrix))
          (matfeaturestreinamento (double-matrix))
          (quantdadosfr 0)
          (quantdadosfrtr 0)(quantfeatures 0)(nomefeatures "")
          (aux1 0.0)
          (lresult ())

          (videocard   0)

          (tipocaptura 1) ;martelo manual
          (vetfeatures (double-matrix))
          (matfeatures (double-matrix))
          (tipodados "reconhecimentoAMOSTRA")
          (tinicio ()) (tfinal ())
       ;   (tipofeatures [0 0 1 0 0 0])
          (ttotalreconhecimento 0)
          (ttotalgeral 0)
          (m6t ())
          (desloc 0)
          (dl 0)
          (i 0)(j 0)
          (resultadoFINAL ())
          )

(setq ttotalgeral
  (realtime ;inicia a marcação do tempo total de processamento da função
;definir a quantidade de dados de referencia para cada sanidade
;   (printf "Comecando o processo \n")
;Definindo o range de amostras de treinamento
  (if (= (tipoSan 0) 1)
       (setq qdados1 qdados1anotr)
     (setq qdados1 0))
  (if (= (tipoSan 1) 1)
```

```
       (setq qdados2 qdadosurgtr)
     (setq qdados2 0))
  (if (= (tipoSan 2) 1)
       (setq qdados3 qdadosemertr)
     (setq qdados3 0))

;mudando o tamanho da amostra, caso seja necessario
    (if (= dirtipo "manual")
       (setq samplesize (* multsamplesize samplesize)))

 (if (= modo 0)
      (setq nomefeatures "-SEMPCA")
     (setq nomefeatures "-COMPCA"))

   (if (= (tipofeatures 0) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numSPECTRO))
        (setq nomefeatures (concat nomefeatures "-FFT"))))
   (if (= (tipofeatures 1) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numLPC))
        (setq nomefeatures (concat nomefeatures "-LPC"))))
   (if (= (tipofeatures 2) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numMFCC))
        (setq nomefeatures (concat nomefeatures "-MFCC"))))
   (if (= (tipofeatures 3) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numZeros))
        (setq nomefeatures (concat nomefeatures "-ZC"))))
   (if (= (tipofeatures 4) 1)
      (progn
        (setq quantfeatures (+ quantfeatures numENERGY))
        (setq nomefeatures (concat nomefeatures "-En"))))
   (if (= (tipofeatures 5) 1)
      (progn
        (setq quantfeatures (+ quantfeatures 1))
        (setq nomefeatures (concat nomefeatures "-AREA"))))

;obtendo o valor de dr
   (setq matdef (loadtxt  (concat dirprog "/sanidades/K" (str K) "-
definicao" nomefeatures "-"    dirtipo ".txt")))
   (setq DR (matdef 1 0))

;-------------------------------------------------------
;-----   LER O ARQUIVO DE AUDIO
;-------------------------------------------------------
         ;ler o arquivo de som
   (setq datawave (loadwave arqWave 0))
         ;extrair o primeiro canal do arquivo de som
   (setq maudio  (extraicanalwave datawave 0))
         ;obter a taxa de amostragem do arquivo de som
   (setq srate (extraitx datawave))
         ;detectar a quantidade de amostras a serem "puladas" para obter
somente o sinal
   (setq m6t (extraidadossanidade maudio 0 srate sizejanenerg samplesize))
   (setq deslocdatamat (nth m6t 2));quantidade de amostras a serem
deslocadas
    ;;------PROCESSA O SINAL CASO ELE SEJA ACEITÁVEL
   (if (= deslocdatamat 0)
      (progn ;DESCARTA O SINAL E AVISA DO ERRO
```

```
            (setq resultadoFINAL (list  0 0.0 0.0 maudio))
            (printf "sinal invalido\n\n")
         ))
;-------------------------------------------------------------
;-----------FIM DA LEITURA DO ARQUIVO DE ÁUDIO--------------
;-------------------------------------------------------------
 (if (<> deslocdatamat 0)
  (progn
 ;  (printf "Iniciando reconhecimento \n")
   (setq ttotalreconhecimento
   (realtime ;inicia a marcação do tempo de processamento somente da parte
de reconhecimento
    (setq m1 (preprocessasinal maudio srate deslocdatamat tiponormalwave
filtrar)) ; pré-processamento do sinal
    (setq vetfeatures (obtemfeatures m1 tiponormal modo))
   ; (plota2sinais maudio m1 "1" "2")
   ; (plota2sinais maudio m1 "FFT1" "FFT2" "JANELA DE COMPARAÇÃO"
"Frequências" "Amplitude" "Frequências" "Amplitudes" 1 1 0 0 0)

    (if (= modo 1)
     (progn
      (load-matrix mattranslation (concat dirprog "/sanidades/K" (str K)
"-dr" (str dr) "-mattranslation" nomefeatures "-" dirtipo ".dat"))
      (setq matfeaturesfr (transpose (multmat (transpose mattranslation)
(transpose matfeatures))));features reduzidas
     )
     (progn
        (setq DR (idx-dim vetfeatures 0))))
;obter a matriz de features reduzidas obtidas no processo de treinamento
    (load-matrix matfeaturestreinamento (concat dirprog "/sanidades/K"
(str K) "-dr" (str DR) "-matfeaturesreduzidas" nomefeatures "-" dirtipo
".dat"))
;-----------------------------------------------------------------
;;------ATUALIZAR A MATRIZ DE FEATURES EM CONFORMIDADE COM AS SANIDADES A
SEREM UTILIZADAS
    (setq matfeaturestreinamento (atualizaMatFeatures
matfeaturestreinamento qdados1 qdados2 qdados3))
;-----------------------------------------------------------------
;comparar cada linha da matriz de features reduzidas de teste com toda a
matriz reduzida do treinamento
;e obter a qual linha desta matriz cada elemento da matriz reduzida de
teste tem menor distancia
    (printf "Processando o reconhecimento \n")
    (setq quantdadosfr 1)
    (setq quantdadosfrtr (idx-dim matfeaturestreinamento 0))
    (setq matd1 (double-matrix quantdadosfrtr))
    (setq matd2 (double-matrix quantdadosfrtr))
    (setq clusters (double-matrix quantdadosfr))
    (if (> quantdadosfr 0)
      (progn
         (for (j 0 (1- quantdadosfrtr))
           ;calcula a distancia de um elemento da matriz de features para
cada elemento da matriz de treinametno
             (setq linhafrtr (extrailinhamatriz matfeaturestreinamento j))
             (matd1 j (disteuclidiana vetfeatures linhafrtr));distancia
entre o elemento de teste com cada posição da matriz fr de
treinamentopreprocessasinal
             (matd2 j j) ;indice do elemento
         )
```

```
;ordenar a matriz matd1 em ordem crescente e associar os valores de matd2
com esta ordenação para termos os indices dos dados. O resultado será a
matriz matd2
         ;(setq linhad1 matd1);obtendo uma linha da matriz matd1
         ;ordenando a linha de matd1
         (setq lresult (ordenavetormenor matd1))  ;linhad1))
         (for (i 0 (1- quantdadosfrtr))
            (matd1 i ((nth lresult 1) i))
            (matd2 i ((nth lresult 2) i)))
          ;dados ordenados - matrizes matd1 ordenada e matd2 com os
indices referentes a ordenação de matd1

   ;-----TODAS AS SANIDADES
      (if (and (> qdados1 0) (> qdados2 0) (> qdados3 0))
         (for (i 0 (1- quantdadosfrtr))
          ;testa se pertence a sanidade 1 ANO - menor que QDADOS1
            (if (< (matd2  i) qdados1)
                (matd2 i 1)
                ;testa se pertence a sanidade URGENCIAL - ENTRE QDADOS1 E
QDADOS2
                (if (and (< (matd2  i) (+ qdados1 qdados2)) (>= (matd2 i)
qdados1))
                   (matd2 i 2)
                ;testa se pertence a sanidade EMERGENCIAL - MAIOR QUE
QDADOS2 E MENOR QUE QDADOS3
                   (if (and (< (matd2 i) (+ qdados1 qdados2 qdados3)) (>=
(matd2 i) (+ qdados1 qdados2)))
                      (matd2 i 3)
                         (matd2 i -1))))
;-------------------------------------------------------------------
         )
      )
    ;------ 1 ANO E URGENCIAL
      (if (and (> qdados1 0) (> qdados2 0) (= qdados3 0))
         (for (i 0 (1- quantdadosfrtr))
          ;testa se pertence a sanidade 1 ANO - menor que QDADOS1
            (if (< (matd2 i) qdados1)
                (matd2 i 1)
                ;testa se pertence a sanidade URGENCIAL - ENTRE QDADOS1 E
QDADOS2
                (if (and (< (matd2 i) (+ qdados1 qdados2)) (>= (matd2 i)
qdados1))
                   (matd2 i 2)
                  (matd2 i -1)))
;-------------------------------------------------------------------
         )
      )
    ;------ 1 ANO E EMERGENCIAL
      (if (and (> qdados1 0) (= qdados2 0) (> qdados3 0))
         (for (i 0 (1- quantdadosfrtr))
          ;testa se pertence a sanidade 1 ANO - menor que QDADOS1
            (if (< (matd2 i) qdados1)
                (matd2 i 1)
                ;testa se pertence a sanidade EMERGENCIAL - ENTRE QDADOS1
E QDADOS3
                (if (and (< (matd2 i) (+ qdados1 qdados3)) (>= (matd2 i)
qdados1))
                   (matd2 i 3)
                 (matd2 i -1)))
;-------------------------------------------------------------------
         )
```

```
      )
    ;-----URGENCIAL E EMERGENCIAL
      (if (and (= qdados1 0) (> qdados2 0) (> qdados3 0))
         (for (i 0 (1- quantdadosfrtr))
          ;testa se pertence a sanidade URGENCIAL - menor que QDADOS2
            (if (< (matd2 i) qdados2)
                (matd2 i 2)
                ;testa se pertence a sanidade EMERGENCIAL - ENTRE QDADOS2
E QDADOS3
                (if (and (< (matd2 i) (+ qdados2 qdados3)) (>= (matd2 i)
qdados2))
                   (matd2 i 3)
                  (matd2 i -1)))
;-------------------------------------------------------------------
         )
      )

;armazenar ao qual sanidade o sinal pertence (1 - 1 ANO / 2 - URGENTE / 3
- EMERGENCIAL
         ;(clusters 0 (procuramaiorocorrenciaNew (extrailinhamatriz matd2
0) KNN tipoAnaliseResultado))
         (clusters 0 (procuramaiorocorrenciaNew matd2 KNN
tipoAnaliseResultado))
          ;salvando as informações
 ;;;        (printf "Gerando arquivo de saida \n")
 ;;;        (printf    (concat dirprog "/sanidades/KNN" (str KNN) "-dr"
(str DR) "-resultado" nomefeatures "-" tipodados "-" dirtipo ".txt\n"))
;         (writetxt2 (concat dirprog "/sanidades/KNN" (str KNN) "-dr" (str
DR) "-resultado" nomefeatures "-" tipodados "-" dirtipo ".txt") clusters
(list "nada" "tmp.wav")) ;listacruzetas)
      ))
  ));;--FIM ttotalreconhecimento
;;;  (printf "Finalizando... \n")
  (setq resultadoFINAL(list (clusters 0) ttotalgeral ttotalreconhecimento
maudio matd1 matd2));retorna a sanidade(1 - 1 Ano, 2 - URGENCIAL, 3 -
EMERGENCIAL), tempo total de processamento da função, tempo total somente
da parte de reconhecimento e o arquivo de audio gravado
  ));;fim do if referente ao teste se o sinal é válido
));;--FIM ttotalgeral
;  );;fim do if referente ao teste se o sinal é válido
  resultadoFINAL
))

;-------------------------------------------------------------------
; ATUALIZA A MATRIZ DE FEATURES PARA O TESTE DAS SANIDADES DESEJADAS
;
; PARÂMETROS:
; mat - matriz de features original
; qdados1 - quantidade de dados a serem utilizados para o treinamento de 1
Ano
; qdados2 - quantidade de dados a serem utilizados para o treinamento de
Urgencial
; qdados3 - quantidade de dados a serem utilizados para o treinamento de
Emergencial
(defun atualizaMatFeatures (mat qdados1 qdados2 qdados3)
    (let* ((m (idx-dim mat 0)) ; quantidade de linhas da matriz
           (n (idx-dim mat 1)) ; quantidade de colunas da matriz
           (newmat (double-matrix (+ qdados1 qdados2 qdados3) n)) ; nova
matriz
           (vetaux (double-matrix))
           (i 0) (j 0)(k 0)
```

```
          )
      (if (> qdados1 0);copiar a linhas da matriz original referentes a
1Ano
        (progn
          (for (i 0 (1- qdados1anotr))
              (for (j 0 (1- n))
                 (newmat k j (mat i j)))
              (setq k (1+ k)))))
     (if (> qdados2 0);copiar a linhas da matriz original referentes a
Urgencial
        (progn
          (for (i qdadosurgtr (+ qdados1anotr qdadosurgtr -1))
              (for (j 0 (1- n))
                 (newmat k j (mat i j)))
              (setq k (1+ k)))))
     (if (> qdados3 0);copiar a linhas da matriz original referentes a
emergencial
        (progn
          (for (i (+ qdados1anotr qdadosurgtr) (+ qdados1anotr qdadosurgtr
qdadosemertr -1))
              (for (j 0 (1- n))
                 (newmat k j (mat i j)))
              (setq k (1+ k)))))
    newmat
    )
)




(defun ReconheceKNN ()
  (let* ((resultado ())
         (vetorm (double-matrix))
         (vetresult [0 0 0])
         (matresult (double-matrix))
         (i 0)
        )


;testa 3 sinais
    (for (i 0 2)
      (setq resultado (reconheceamostraKNN 1))
      (setq vetorm (nth resultado 4))
      (vetresult i (nth resultado 1))
      (setq vetorm (copvet vetorm 35000 5000))
      (setq matresult (addlinvetmat matresult vetorm))
    )
      (list vetresult matresult)
 ))


;---------------------------------------------------------------------------
; LOCALIZA A MENOR OCORRÊNCIA EM UM VETOR DE DADOS
; RETORNANDO UMA LISTA COM O VALOR DA MENOR OCORRÊNCIA E O SEU ÍNDICE
; Parâmetro:
; vet - vetor a ser processado
(defun localizamenor (vet)
  (let* ((n (idx-dim vet 0))
         (pos 0)
         (valor 0.0))
```

```
      (if (> n 0)
        (progn
          (setq valor (vet 0))
          (for (i 1 (1- n))
            (if (< (vet i) valor)
              (progn
               (setq valor (vet i))
               (setq pos i))))))
      (list valor pos)))

;---------------------------------------------------------------------------
;  ORDENA UM VETOR EM ORDEM CRESCENTE
; Parâmetro:
; vet - vetor a ser ordenado
(defun ordenavetormenor (vet)
   (let* ((n (idx-dim vet 0))
          (vetaux1 (double-matrix n))
          (vetpos (double-matrix n))
          (vetaux2 (double-matrix))
          (valor1 0.0)
          (valor2 ())
          (aux 0.0)
          (pos 0)
          (result ())
         )

        (if (> n 0)
         (progn
           (setq vetaux1 (copvet vet 0 0))
           (for (i 0 (1- n))
              (vetpos i i))
           (for (i 1 (1- n))
             (setq valor1 (vetaux1 i))
             (setq vetaux2 (copvet vetaux1 i 0))
             (setq valor2 (localizamenor vetaux2))
             (if (< (nth valor2 1) (vetaux1 (1- i)))
               (progn
                  (setq aux (vetaux1 (1- i)))
                  (vetaux1 (1- i) (nth valor2 1))
                  (vetaux1 (+ (nth valor2 2) i) aux)
                  (setq aux (vetpos (1- i)))
                  (vetpos (1- i) (vetpos (+ (nth valor2 2) i)))
                  (vetpos (+ (nth valor2 2) i) aux))))
           (setq result (list vetaux1 vetpos))))
    result))

;---------------------------------------------------------------------------
; LOCALIZA A MAIOR OCORRÊNCIA EM UM VETOR DE DADOS
; GERANDO UM SUBVETOR DE ORDEM KNN
; Parâmetro:
; vet - vetor a ser processado
; knn - quantidade de dados do vetor a ser retornado
; TipoAnaliseResultado) - Define como será processado o resultado para o
reconhecimento
;             0 - procura a maior quantidade de ocorrencias de um
resultado no vetor de quantidade K
;             1 - procura a média dos resultados de um vetor de quantidade
K
;             2 - procura a mediana dos resultados de um vetor de
quantidade K
(defun procuramaiorocorrenciaNew (vet knn TipoAnaliseResultado)
```

```
  (let* ((n (idx-dim vet 0))
         (vetaux (double-matrix))
         (aux 0)
         (cont 0)
         (vcaux1 (double-matrix knn))
         (vcaux2 (double-matrix knn))
         (result 0)
         ;(result ())
        )
 (cond
  ((= TipoAnaliseResultado 0)
    (if (= knn 1)
        (setq result (vet 0))
     (progn
       (setq vetaux (copvet vet 0 knn))
;ordena o vetor
       (setq vetaux (sortdownvet vetaux))
         ;conta a quantidade de centroides existentes no vetor
       (setq qaux 1)
       (setq vaux (vetaux 0))
       (setq nc 0)
       (setq nk 0)
       (for (j 1 (1- knn))
         (if (= vaux (vetaux j))
            (progn
               (setq qaux (1+ qaux))
               (setq vaux (vetaux j)))
            (progn
               (vcaux1  nk vaux)
               (vcaux2  nk qaux)
               (setq qaux 1)
               (setq vaux (vetaux j))
               (setq nk (1+ nk))
             )))
       (vcaux1 nk vaux)
       (vcaux2 nk qaux)

       (for (i 0 (- knn 2))
         (for (j (+ i 1) (1- knn))
            (if (>= (vcaux2 j) (vcaux2 i))
              (progn
                     (setq aux1 (vcaux2 i))
                     (vcaux2 i  (vcaux2 j))
                     (vcaux2 j aux1)
                     (setq aux1 (vcaux1 i))
                     (vcaux1 i  (vcaux1 j))
                     (vcaux1 j aux1)))))

       (setq result (vcaux1 0))))
   ) ; FIM PARA TipoResultadoAnalise 0
  ((= TipoAnaliseResultado 1) ;USO DE MEDIA
     (if (= knn 1)
             (setq result (vet 0))
      (progn
            (setq vetaux (copvet vet 0 knn))
            (setq aux   (mediaVet vetaux))
            (setq result (arredonda aux))
      ))
  ); FIM PARA TipoResultadoAnalise 1
  ((= TipoAnaliseResultado 2) ;USO DE MEDIANA
      (if (= knn 1)
```

```
              (setq result (vet 0))
          (progn
              (setq vetaux (copvet vet 0 knn))
              (setq aux   (calculaMediana vetaux))
              (setq result (arredonda aux))
          ))
    ); FIM PARA TipoResultadoAnalise
  ); FIM COND 1

      result))

;-----------------------------------------------------------------------
; ASSOCIA A SANIDADE COM O NÚMERO DO CLUSTER
; PARAMETROS
; vetcluster - vetor de associação do centroide com os dados
; quantamostrasporcluster - quantidade de amostras que foram treinadas
;                           considera-se a mesma quantidade por cluster
; quantidade de clusters
; nomesK - lista com os nomes que deverão ser associados aos cluster na
ordem desejada
;       ex.: ("1ANO" "URGENTE" "EMERGENCIAL") para K = 3 e seguindo a
divisão do
;            vetor vetclusters, ou seja, os primeiros
quantamostrasporcluster estão
;            associados a sanidade 1ANO,...
(defun selectcluster (vetcluster quantamostrasporcluster K nomesK)
   (let* ((n (idx-dim vetcluster 0))
          (qc (/ n quantamostrasporcluster))
          (qaux 0)(vaux 0)(nc 0)(maior 0)(nk 0)(vetaux (double-matrix))
          (result (double-matrix qc (* 2 K)))
          (vcaux1 (double-matrix qc k));armazena o numero do cluster
temporariamente
          (vcaux2 (double-matrix qc k)));armazena a quantidade de
repetiçoes do numero do cluster temporariamente
     (if (> qc 0)
      (progn
        (for (i 0 (1- qc))
        ;copia os dados referentes a sanidade testada - numero de amostras
por cluster
           (setq vetaux (copvet vetcluster (* i quantamostrasporcluster)
quantamostrasporcluster))
        ;localiza o valor com maior quantidade de repetições
          ;ordena o vetor
           (setq vetaux (sortdownvet vetaux))
          ;conta a quantidade de centroides existentes no vetor
           (setq qaux 1)
           (setq vaux (vetaux 0))
           (setq nc 0)
           (setq nk 0)
           (for (j 1 (1- quantamostrasporcluster))
               (if (= vaux (vetaux j))
                  (progn
                     (setq qaux (1+ qaux))
                     (setq vaux (vetaux j)))
                  (progn
                     (vcaux1 i nk vaux)
                     (vcaux2 i nk qaux)
                     (setq qaux 1)
                     (setq vaux (vetaux j))
                     (setq nk (1+ nk))
                  )))
```

```
            (vcaux1 i nk vaux)
            (vcaux2 i nk qaux)
         )
           ;localiza o maior valor

         (for (i 0 (1- qc))
            (for (j 0 (1- K))
               (for (jj (1+ j) (1- K))
                  (if (> (vcaux2 i jj) (vcaux2 i j))
                    (progn
                      (setq nk   (vcaux1 i j));armazena os anteriores
                      (setq vaux (vcaux2 i j))
                      (vcaux1 i j (vcaux1 i jj));copia
                      (vcaux2 i j (vcaux2 i jj))
                      (vcaux1 i jj nk);volta o anterior
                      (vcaux2 i jj vaux))))))
         ;gerar uma unica matriz
         ;onde, para cada linha, os k primeiros elementos são a ordem dos
centroides
         ;e os proximos k elementos são a quantidade de repeticoes

         (for (i 0 (1- qc))
            (for (j 0 (1- K))
              (result i j       (vcaux1 i j))
              (result i (+ j K) (vcaux2 i j))))
    ))
    result
))

;---------------------------------------------------------------------
; DISTÂNCIA EUCLIDIANA ENTRE PONTOS
; PARÂMETROS:
; v1 - vetor 1
; v2 - vetor 2
; dist - distância euclidiana como saída da função
(defun disteuclidiana (v1 v2)
   (let* ((n1 (idx-dim v1 0))
          (n2 (idx-dim v2 0))
          (dist 0))
     (if (and (= n1 n2) (> n1 0))
        (progn
           (for (i 0 (1- n1))
              (setq dist (+ dist (* (- (v2 i) (v1 i)) (- (v2 i) (v1
i))))))
           (setq dist (sqrt dist))))
      dist))

;---------------------------------------------------------------------
; DISTÂNCIA O ERRO MÉDIO QUADRÁTICO ENTRE PONTOS
; PARÂMETROS:
; v1 - vetor 1
; v2 - vetor 2
; mse - MSE saída da função
(defun calcMSE (v1 v2)
   (let* ((n1 (idx-dim v1 0))
                  (for (i 0 (1- n1))
              (setq mse (+ mse (* (- (v2 i) (v1 i)) (- (v2 i) (v1 i))))))
           (setq mse (/ mse n1))))
      mse))

;---------------------------------------------------------------------
```

```
;   CALCULA A AŔEA DE UM SINAL DE AUDIO
;  O SINAL SERÁ REBATIDO NA SUA PARTE POSITIVA
; PARAMETROS:
; vet - vetor de dados
; tx  - taxa de amostragem
(defun calculaareasinal (vet tx)
   (let* ((n (idx-dim vet 0)) ; quantidade de amostras do sinal
          (vetnor (idx-abs vet)) ; rebate o sinal para a sua parte
positiva
          (base (/ 1 tx))     ; calcula a distância entre amostras
          (area 0.0)
          (altura 0.0)
         )
     (for (i 0 (- n 2))
        (if (<> (vetnor i) (vetnor (+ i 1)))
            (progn
              (setq altura (abs (- (vetnor i) (vetnor (+ i 1)))))
              (setq area (+ area (/ (* base altura) 2.0))))) ; area do
triangulo de aproximacao
        (if (>= (vetnor i) (vetnor (+ i 1)))
            (setq area (+ area (* base (vetnor (+ i 1)))))) ; calcula a
area usando o lado i + 1
        (if (< (vetnor i) (vetnor (+ i 1)))
            (setq area (+ area (* base (vetnor i))))) ; calcula a area
usando o lado i
     )
   area))
```

2 - Arquivo mydsp.lsh:

```
(libload "fftw/fftw")
(defvar mpi 3.14159265358979323846)
(defvar LowPass   0)
(defvar BandPass  1)
(defvar HighPass  2)
(defvar BandRejec 3)
(defvar Blackman  0)
(defvar Hamming   1)

;-------------------------------------------------------------------
; CALCULA O LOG NA BASE 10 DE UM NÚMERO x
(defun log10 (x)
 (let ((y (/ (log x) (log 10))))
   y))

;-------------------------------------------------------------------
;  OBTEM A QUANTIDADE DE AMOSTRAS NECESSÁRIAS PARA REPRESENTAR UM SINAL DE
N MILISEGUNDOS
; tx - taxa de amostragem do sinal
; n  - tempo da amostra em milissegundos

(defun numamostra (tx n)
  (let ((nm (* tx (/ n 1000))))
    (arredonda nm)))

;-------------------------------------------------------------------
; ARREDONDA UM NÚMERO
; se a parte fracionária for maior ou igual a 0.5 o número é arrendondado
para cima
; se for menor do 0.5 é arrendodado para baixo
(defun arredonda (x)
```

```
 (let* ((pint (int x))
        (pfrac (- x pint))
        (result pint))
    (if (>= pfrac 0.5)
      (setq result (1+ pint)))
    result))

;---------------------------------------------------------------------
; VETOR DE FREQUENCIA MEL
;(defvar melWorkingFrequencies [ 10.0 20.0 90.0 300.0 680.0 1270.0 2030.0
2970.0 4050.0 5250.0 6570.0])

;---------------------------------------------------------------------
; CALCULA O NÚMERO DE FILTROS DADA UMA FREQUENCIA
; nyquist - Fs/2

(defun numMelFilters (nyquist)
  (let* ((frequency nyquist)
         (delta (mel frequency))
         (numFilters 0))
      (while (> frequency 10)
        (progn
         (setq numFilters (1+ numFilters))
         (setq frequency (- frequency (/ delta 2)))
         (setq delta (mel frequency))))
      numFilters))

;---------------------------------------------------------------------
; INICIALIZA O RANGE DE FREQUENCIAS
(defun initMelFrequenciesRange(nyquist)
   (let* ((frequency nyquist)
          (delta (mel frequency))
          (numFilters (numMelFilters nyquist))
          (melWorkingFrequencies (double-matrix numFilters))
          (i 0)
          (cFreq 0))
      (while (> frequency 10)
        (progn
          (setq frequency (- frequency (/ delta 2)))
          (setq delta (mel frequency))
          (setq cFreq (arredonda frequency))
          (melWorkingFrequencies (- (- numFilters 1) i) 10)
          (while (< (melWorkingFrequencies (- (- numFilters 1) i))  (-
cFreq 10))
             (melWorkingFrequencies (- (- numFilters 1) i) (+
(melWorkingFrequencies (- (- numFilters 1) i)) 10)))
          (setq i (1+ i))))
       melWorkingFrequencies))

;---------------------------------------------------------------------
; CALCULA O VALOR DE UMA JANELA TRIANGULAR
; PARÂMETROS:
; value - valor a ser aplicada a janela
; samples - quantidade de amostras da janela
(defun wintriangular (value samples)
  (let ((aux 0.0)
        (aux1 (/ 2 (+ samples 1)))
        (aux2 (/ (+ samples 1) 2))
        (aux3 (abs (- value (/ (- samples 1) 2)))))
    (setq aux (* aux1 (- aux2 aux3)))
    aux))
```

```
;---------------------------------------------------------------------
(defun EnhanceHighFrequencies (sinal tam)
  (let ((result (double-matrix tam)))
     (for (i 1 (1- tam))
        (result i (- (sinal i) (* 0.95 (sinal (1- i))))))
     result))

;---------------------------------------------------------------------
; CALCULA O VALOR MEL PARA A FREQUENCIA DADA
; PARÂMETROS
; value - frequencia a ser calculada
(defun mel(value)
   (let* ((aux1 (+ 1.0 (/ value 700.0)))
          (aux 0.0))
     (setq aux (* 2595.0 (log10 aux1)))
     aux))

;---------------------------------------------------------------------
; CALCULA O VALOR INVERSO DE MEL PARA A FREQUENCIA DADA
; PARÂMETROS
; value - valor mel a ser invertido
(defun melinv(value)
  (let* ((aux1 (* 700.0 (- (** 10.0 (/ value 2595.0)) 1.0))))
      aux1))

;---------------------------------------------------------------------
; COMPUTE OS COEFICIENTE MFCC
; PARÂMETROS:
; sinal - SINAL DE ENTRADA QUE É O ESPECTRO DE POTÊNCIA DA FFT
; melWorkingFrequencies - range de frequencias para o calculo dos
coeficientes mfcc
(defun computeMFCC(sinal melWorkingFrequencies)
  (let* (;(melWorkingFrequencies (initMelFrequenciesRange (/ fs 2)))
         (qcoef  (idx-dim melWorkingFrequencies 0))
         (result (double-matrix qcoef))
         (mfcc   (double-matrix qcoef))
         (segment 0)
         (start   0)
         (end     0)
         (qsinal (idx-dim sinal 0)))
     (progn
        (for (i 0 (1- qcoef))
          (progn
             (result i 0)
             (setq segment (arredonda (/ (mel (melWorkingFrequencies i))
10)))
             (setq start (- segment (int (/ segment 2))))
             (setq end (+ segment (/ segment 2)))
             (if (> end qsinal)
                 (setq end qsinal))
             (for (j start (1- end))
                (result i (+ (result i) (* (sinal j) (wintriangular j
segment)))))
            (when (>  (result i) 0) (result i (log10 (abs (result i)))))
            (when (<= (result i) 0) (result i 0))))
        (for (i 0 (1- qcoef))
          (progn
            (for (j 0 (1- qcoef))
                (mfcc i (+ (mfcc i) (* (result i) (cos (* (/ (* mpi i)
qcoef) (- j 0.5)))))))))
```

```
            (mfcc i (* (mfcc i) (sqrt (/ 2.0 qcoef)))))))
       mfcc))

;---------------------------------------------------------------------
; CALCULA O ESPECTRO DE POTÊNCIA DIRETAMENTE OU NA FORMA EM decibel
; PARÂMETROS
; x - vetor de dados
; method:
; 0 - direto
; 1 - em decibel
(defun spectrum(x method)
  (let* ((xlenght (idx-dim x 0))
         (xaux (/ xlenght 2))
         (xre (double-matrix xaux))
         (xim (double-matrix xaux))
         (decibel (double-matrix xaux))
         (xx (double-matrix xlenght 2)))
   (progn
    (fftw-dft-r2c-1d x xx f) ;calcula a fft do sinal x
    ;obtem a parte real (primeira coluna de xx
    ;obtem a parte imagina (segunda coluna de xx)
    (for (i 0 (1- xaux))
       (progn
         (xre i (xx i 0))
         (xim i (xx i 1))))
    ;CALCULA O POWER SPECTRUM
    (for (i 0 (1- xaux))
           (if (> method 0)
              (decibel i (* 10 (log10 (sqrt (+ (* (xre i) (xre i)) (* (xim
i) (xim i)))))))
             (decibel i (sqrt (+ (* (xre i) (xre i)) (* (xim i) (xim
i))))))))
         decibel))

;---------------------------------------------------------------------
;  Processa o Zero Crossing - Quantidade de vezes que o sinal de áudio
muda
;  de sentido
; m - vetor de dados

(defun zerocrossing (m fs)
  (let* ((T (idx-dim m 0))
         (iT (/ 1 (- T 1)))
         (zcr 0)
         (A 0))
   (for (i 1 (- T 1))

         (if (<> (sign (m i)) (sign (m (1- 1))))
               (setq zcr (+ zcr 1))))
   zcr))

;---------------------------------------------------------------------
;  CALCULO DA ENERGIA DE UM SINAL
(defun energysignal (m)
   (let* ((N (idx-dim m 0))
          (enn 0))
    (for (i 0 (1- N))
       (setq enn (+ enn (abs (* (m i) (m i))))))
    enn))

;---------------------------------------------------------------------
```

```
; Percorre um vetor de áudio de tamanho T com uma janela de tamanho N (N <
T)
; para cada janela percorrida efetua o cálculo de Zero Crossing
;
;  m  - vetor de dados
;  fs - taxa de amostragem do vetor de dados
;  N  - tamanho (em amostras) da janela
(defun zjanela (m fs N)
  (let* ((T (idx-dim m 0))
         (posjanela 0)
         (listazcr ())
         (auxlist 0)
         (auxM (double-matrix N))
        )
     (if (< N T)
      (while (< posjanela T)
       (progn
; Obtem uma janela da posição posjanela e de tamanha N
         (setq auxM (copvet m posjanela N))
; Calcula a quantidade de zeros crossing da janela
         (setq auxlist (zerocrossing auxM fs))
; Adiciona esta quantidade ao final da lista de zeros crossing
         (setq listazcr (append listazcr (list auxlist)))
; Atualiza a posição da janela
         (setq posjanela (+ posjanela N))
        )))
;retorna a lista de zeros crossing
     listazcr))
;))

;---------------------------------------------------------------------
; Percorre um vetor de áudio de tamanho T com uma janela de tamanho N (N <
T)
; para cada janela percorrida efetua o cálculo de Energy
;
;  m  - vetor de dados
;  fs - taxa de amostragem do vetor de dados
;  N  - tamanho (em amostras) da janela

(defun ejanela (m fs N)
  (let* ((T (idx-dim m 0))
         (posjanela 0)
         (listaenn ())
         (auxlist 0)
         (auxM (double-matrix N))
        )
     (if (< N T)
      (while (< posjanela T)
       (progn
; Obtem uma janela da posição posjanela e de tamanha N
         (setq auxM (copvet m posjanela N))
; Calcula a quantidade de energia da janela
         (setq auxlist (energysignal auxM))
; Adiciona esta quantidade ao final da lista de energy
         (setq listaenn (append listaenn (list auxlist)))
; Atualiza a posição da janela
         (setq posjanela (+ posjanela N))
        )))
;retorna a lista de energia
     listaenn))
```

```
;-------------------------------------------
; EXPANDE A LISTA DE ZEROCROSSING PARA O TAMANHO DO ARQUIVO DE AUDIO
; PARA COMPARAÇÃO
; vzcr - lista de zerocrossing
; ws - tamanho da janela de varredura no arquivo de audio para gerar
;      a lista de zerocrossing
; ns - tamanho do vetor de audio a ser gerado
(defun expandezcr (vzcr ws ns)
  (let* ((vetorvzcr (listtoarray vzcr))         ; transforma a lista de
zcr em vetor
         (sizezcr   (length vzcr))              ; obtem o tamanho de zcr
      ;   (sizedados (* ws sizezcr))             ; obtem o tamanho do
vetor de dados
         (dadossaida (double-matrix ns)) ;gera um vetor do tamanho do
arquivo de som
         )
    (for (i 0 (1- sizezcr))
       (for (j 0 (1- ws))
         (if (< (+ j (* i ws))  ns)
           (dadossaida (+ j (* i ws)) (vetorvzcr i)))))
    dadossaida))

;-------------------------------------------------------------------------
;;----ELIMINA NIVEL DE AMOSTRA NO INICIO DO ARQUIVO
;; vetdados -- vetor de dados a ser retiradas as amostras do inicio
;; limiard  -- limiar minimo de dados a serem retirados
;; desloc   -- retorna a quantidade de amostras que foram puladas para
achar o limiar ou 0 caso nao seja encontrado o limiar e neste caso também
retorna o proprio vetdados como resultado
(defun limiardata (vetdados limiard)
  ;((-int-) desloc)
  (let* ((nz (car (idx-dim vetdados))) ;obtem o tamanho do vetor vetdados
         (i 0)
         (k 0)
         (matlim vetadados))
        (while (and (< i nz) (< (abs (vetdados i)) limiard))
            (setq i (+ i 1)))
        (if (< i (1- nz))
         (progn
          (setq matlim (copvet vetdados (1- i) 0))
          (setq desloc (1- i)))
         (setq desloc 0))
        (list matlim desloc)
           ))

;-------------------------------------------------------------------------
;     LEITURA DE ARQUIVO WAVE COM FUNÇÕES DO LUSH
;
;; ---RETORNA UM VETOR COM OS DADOS DE UM ARQUIVO WAV ---
;; ---nwave - arquivo wave
;; ---tamwave - quantidade de amostras a serem lidos do arquivo wave
;; ---tamwave = 0 -> retorna todo o arquivo wave
;; ---tamwave > tamanho do arquivo -> retorno todo o arquivo wave
(defun vetwave (nwave tamwave)
  (defvar waveidx) ;;irá armazenar os dados do arquivo wave em uma
estrutura idx
  (defvar vw) ;; irá armazenar o vetor com os dados do arquivo wave
  (setq waveidx (audiofile-read nwave)) ;;efetua a leitura dos dados do
arquivo wave
  (setq n (car (idx-dim waveidx))) ;; armazena a quantidade de dados do
arquivo wave
```

```
  (if (and (< tamwave n)( > tamwave 0))
      (setq n tamwave))
  (setq vw (double-matrix n))
  (for (i 0 (- n 1))
    (vw i (waveidx i)))
  vw)

(defvar rec-sound-cmd "rec -q -c %d -t wav -r %d tmp.wav trim 0 %d")
(defvar rec-sound-eca "ecasound -f:16,4,44100 -i alsahw,1,0,0 -
f:16,4,44100 -o tmp.wav -t %d -q")
(defvar play-video-cmd "xawtv  -c %c -f")

#? (play-video <c>)
;; play the video
;; using device <c>;
(de play-video (c)

  (sh (concat (concat "xawtv  -c " c)))
))

;-------------------------------------------------------------------------
; --- GRAVA UM SINAL DA PLACA DE ÁUDIO PADRÃO
; C - QUANTIDADE DE CANAIS
; R - TAXA DE AMOSTRAGEM
; T - QUANTIDADE DE AMOSTRAS A SEREM GRAVADAS
; OS DADOS SERÃO GRAVADOS EM UM ARQUIVO WAVE TEMPORÁRIO (tmp.wav) NO
DIRETÓRIO ATUAL
#? (rec-sound <c> <r> <t>)
;; rec the signal
;; using sampling rate <r>
;; <c> canais
;; and duration <t> amostras
;;retorna a estrutura de som
(de rec-sound (c r t)
  (let* ((d (/ t r))
         (fp (popen (sprintf rec-sound-cmd c r d) "rb"))
         (m (double-matrix (* c t))))
    (pclose fp)
    (setq m (loadwave "tmp.wav" 0))
    m))

;-------------------------------------------------------------------------
; --- GRAVA UM SINAL DA PLACA DE ÁUDIO PADRÃO
; C - QUANTIDADE DE CANAIS
; R - TAXA DE AMOSTRAGEM
; T - QUANTIDADE DE AMOSTRAS A SEREM GRAVADAS
; OS DADOS SERÃO GRAVADOS EM UM ARQUIVO WAVE TEMPORÁRIO (tmp.wav) NO
DIRETÓRIO ATUAL
#? (rec-sound-2 <r> <t>)
;; rec the signal
;; using sampling rate <r>
;; and duration <t> amostras
;;retorna a estrutura de som
(de rec-sound-2 (r tp devaudio)
  (let* ((d (/ tp r))
         (m (double-matrix (* 4 tp))))
     (sys (concat "ecasound -f:16,1,44100 -i " devaudio " -f:16,1,44100 -o
tmp.wav -q -t " (str d) " &" ))
    ))

;-------------------------------------------------------------------------
```

```
;LER UM ARQUIVO DE DADOS NO FORMATO TXT
;PARÂMETROS:
; fname - nome do arquivo a ser lido
(defun loadtxt (fname)
   (let* ((lista (read-lines fname))
          (n1 (length lista))(n2 0)(aux "")(ans ())
          (mat (double-matrix)))
     lista)
)

;---------------------------------------------------------------------
;GRAVA UMA MATRIZ EM UM ARQUIVO DE DADOS NO FORMATO TXT
;PARÂMETROS:
; fname - nome do arquivo a ser lido
; mat - matriz a ser gravada
(defun writetxt (fname mat)
  (let* ((d (idx-dim mat)) (lin (idx-dim mat 0)) (col 1))
    (if (> (length (idx-dim mat)) 1)
     (progn
       (setq col (idx-dim mat 1))
       (writing fname
         (for (i 0 (1- lin))
           (for (j 0 (1- col))
              (printf "%10.7f " (to-double (mat i j))))
           (printf "\r"))))
     (progn
       (writing fname
         (for (i 0 (1- lin))
              (printf "%15.8f " (to-double (mat i)))
              (printf "\r")))))
   ))

;---------------------------------------------------------------------
;leitura de arquivo WAVE com qualquer quantidade de canais
; PARÂMETRO DE ENTRADA
; fname - nome do arquivo
;
; PARÃMETROS DE SAÍDA
; Na saída teremos uma lista com dois elementos
; PRIMEIRO ELEMENTO:
; LISTA COM OS DADOS DO ARQUIVO DE ÁUDIO
; (nchannels srate brate balign bps sizedata)
; nchannels - quantidade de canais de áudio no arquivo (1, 2, 3, etc)
; srate     - taxa de amostragem do arquivo (22050, 44100, etc)
; balign    - alinhamento de bits
; bps       - quantidade de bits necessários para armazenar uma amostra
; sizedata  - quantidade de dados para um canal
; SEGUNDO ELEMENTO:
; data - matriz de nchannels x sizedata (cada linha armazena um canal de
áudio)

(defun loadwave (fname namostras)
 (let* ((fp (fopen fname "rb"))
        (riff1 (fread-byte fp))(riff2 (fread-byte fp))(riff3 (fread-byte
fp))(riff4 (fread-byte fp))
        (riff (concat (concat (chr riff1) (chr riff2))  (concat (chr
riff3) (chr riff4)))) ; RIFF
        (aux (inv4bytes (fread-int fp))) ; SIZE OF FILE
        (wav1  (fread-byte fp))(wav2  (fread-byte fp))(wav3  (fread-byte
fp))(wav4  (fread-byte fp))
```

```
        (wav (concat (concat (chr wav1) (chr wav2))  (concat (chr wav3)
(chr wav4)))) ;WAVE
        (fmt1  (fread-byte fp))(fmt2  (fread-byte fp))(fmt3  (fread-byte
fp))(fmt4  (fread-byte fp))
        (fmt (concat (concat (chr fmt1) (chr fmt2))  (concat (chr fmt3)
(chr fmt4)))) ; fmt
        (chunksize (inv4bytes (fread-int fp)))   ; chunk size
        (aformat   (inv2bytes (fread-short fp))) ; audio format
        (nchannels (inv2bytes (fread-short fp))) ; num channels
        (srate     (inv4bytes (fread-int fp)))   ; sample rate
        (brate     (inv4bytes (fread-int fp)))   ; byte rate
        (balign    (inv2bytes (fread-short fp))) ; block align
        (bps       (inv2bytes (fread-short fp))) ; bits per sample
        (data1 0)(data2 0)(data3 0)(data4 0)(dataid "")(sizedata 0)
        (deslocinicial 20)
        (fact1 0)(fact2 0)(fact3 0)(fact4 0)(factid 0)
        (factsize 0)
        (d1 0)(d2 0)(d3 0)(d4 0)
        (dados (double-matrix))
    )
      (progn

        ;POSICIONAR NO BLOCO DE DADOS
        ;se não tem compressão (audio format = 1) pular diretamente
        ;caso tenha compressão (audio format <> 1) processar o chunk FACT
        (fseek fp (+ deslocinicial chunksize))
        (if (<> aformat 1)
         (progn
            ;ir para o chunk FACT

              (setq fact1 (fread-byte fp)) (setq fact2 (fread-byte fp))
(setq fact3 (fread-byte fp)) (setq fact4 (fread-byte fp))
              (setq factid (concat (concat (chr fact1) (chr fact2))
(concat (chr fact3) (chr fact4)))) ; FACT
              (setq factsize (inv4bytes (fread-int fp)))
              (fseek-from-current fp factsize)
         ))
        ;POSICIONAR NOS DADOS
        (setq data1 (fread-byte fp)) (setq data2 (fread-byte fp)) (setq
data3 (fread-byte fp)) (setq data4 (fread-byte fp))
        (setq dataid (concat (concat (chr data1) (chr data2))  (concat
(chr data3) (chr data4))))
        (setq sizedata  (inv4bytes (fread-int fp)))
        (setq sizedata (to-int (* (/ (/ (to-float sizedata) (to-float
nchannels)) (to-float bps)) 8.0)))
        (if (and (< namostras sizedata) (> namostras 0))
          (setq sizedata namostras))
     ;ARMAZENAR OS DADOS NA MATRIZ
      ;criar a matriz de nchannels linhas x sizedata colunas
        (setq dados (double-matrix nchannels sizedata))
        (for (i 0 (1- sizedata)) ;varredura de todos as amostras
           (for (j 0 (1- nchannels)) ;varredura de todos os canais
            (progn
               (if (= bps 8)
                  (dados j i (fread-byte fp)))
               (if (= bps 16)
                   (progn
                      (setq d1 (fread-byte fp))
                      (setq d2 (fread-byte fp))
                      (dados j i (+ d1 (* d2 256)))))))
```

```
               (if (= bps 32)
                (progn
                  (setq d1 (fread-byte fp))
                  (setq d2 (fread-byte fp))
                  (setq d3 (fread-byte fp))
                  (setq d4 (fread-byte fp))
                  (dados j i (+ (+ d1 (* d2 256))  (+ (* d3 (** 2 16)) (*
d4 (** 2 32)))))))))))
    ;fechar o arquivo de som
        (fclose fp)
;montar a saidad e dados da função
        (list (list nchannels srate brate balign bps factid sizedata)
dados))))

;-------------------------------------------------------------------
; EXTRAI UM CANAL DA LISTA RETORNADA PELA LEITURA DE ARQUIVO WAVE
; PARAMETROS
; ldados - lista contendo as informações extraidas do arquivo de áudio no
formato (informaçoesdoaudio , matrizdedados)
; canal - canal que será extraído:
; 0 - canal esquerdo
; 1 - canal direito
; 2 - canal central
(defun extraicanalwave (ldados canal)
  (let ((nchannels (car (car ldados)))
        (matrizdados (car (cdr ldados))) ;extrai a matriz da lista de
resultados do canal
        (vetordados (double-matrix)))
     (if (< canal nchannels)
       (setq vetordados (extrailinhamatriz matrizdados canal)))
     vetordados))

;-------------------------------------------------------------------------
; EXTRAI O NÚMERO DE CANAIS DA LISTA RETORNADA PELA LEITURA DE ARQUIVO
WAVE
; PARAMETROS
; ldados - lista contendo as informações extraidas do arquivo de áudio no
formato (informaçoesdoaudio , matrizdedados)
(defun extraincanal (ldados)
  (let ((nchannels (car (car ldados))))
    nchannels))

;-------------------------------------------------------------------------
; EXTRAI A QUANTIDADE DE AMOSTRAS POR CANAL DA LISTA RETORNADA PELA
LEITURA DE ARQUIVO WAVE
; PARAMETROS
; ldados - lista contendo as informações extraidas do arquivo de áudio no
formato (informaçoesdoaudio , matrizdedados)
(defun extraitx (ldados)
  (let ((tx (car (cdr (car ldados)))))
    tx))

;-------------------------------------------------------------------
; EXTRAI N AMOSTRAS DO INÍCIO DE UM VETOR DE DADOS
; PARÂMETROS:
; vet - vetor de dados
; n - quantidade de amostras a serem eliminadas
; n < tamanho do vetor de dados
(defun removeamostras (vet n)
  (let* ((svet (idx-dim vet 0)) ; quantidade de amostras do vetor de dados
         (vetresult (double-matrix)))
```

```
    (if (< n svet)
      (progn
       (setq vetresult (double-matrix (- svet n)))
       (for (i 0 (- (- svet n) 1))
          (vetresult i (vet (+ i n))))))
    vetresult))

;---------------------------------------------------------------------
; NOVAS FUNÇÕES PARA O CÁLCULO DOS COEFICIENTES MFCC
;---------------------------------------------------------------------
(defun computeMFCC2 (spectralData samplingRate NumFilters m)
   (let* ((binSize (idx-dim spectralData 0))
          (mfcc (double-matrix m)))
      (for (i 0 (1- m))
        (mfcc i (GetCoefficient spectralData SamplingRate NumFilters
binSize i)))
    mfcc))

;---------------------------------------------------------------------
; * Computes the specified (mth) MFCC
; *
; * spectralData - array of doubles containing the results of FFT
computation. This data is already assumed to be purely real
; * samplingRate - the rate that the original time-series data was sampled
at (i.e 44100)
; * NumFilters - the number of filters to use in the computation.
Recommended value = 48
; * binSize - the size of the spectralData array, usually a power of 2
; * m - The mth MFCC coefficient to compute
(defun GetCoefficient (spectralData samplingRate NumFilters binSize m)
   (let ((result 0.0)
         (outerSum 0.0)
         (innerSum 0.0))
     (if (< m NumFilters)
      (progn
        (setq result (NormalizationFactor NumFilters m))
        (for (l 1 NumFilters)
            (progn
               (setq innerSum 0.0)
               (for (k 0 (1- binSize))
                  (setq innerSum (+ innerSum (* (abs (spectralData k))
(GetFilterParameter samplingRate binSize k l)))))
               (if (> innerSum 0.0)
                  (setq innerSum (log10 innerSum)))
               (setq innerSum (* innerSum (cos (* (/ (* m mpi) NumFilters)
(- l 0.5)))))
               (setq outerSum (+ outerSum innerSum))))
        (setq result (* result outerSum))))
      result))

;---------------------------------------------------------------------
;* Computes the Normalization Factor (Equation 6)
;* Used for internal computation only - not to be called directly
(defun NormalizationFactor (NumFilters m)
   (let ((normFactor 0.0))
    (if (= m 0)
       (setq normFactor (sqrt (/ 1.0 NumFilters)))
     (setq normFactor (sqrt (/ 2.0 NumFilters))))
   normFactor))

;---------------------------------------------------------------------
```

```
;* Compute the filter parameter for the specified frequency and filter
bands (Eq. 2)
;* Used for internal computation only - not the be called directly
(defun GetFilterParameter (samplingRate binSize frequencyBand filterBand)
  (let* ((filterParameter     0.0)
         (boundary            (/ (* frequencyBand samplingRate) binSize))
         (prevCenterFrequency (GetCenterFrequency (- filterBand 1)))
         (thisCenterFrequency (GetCenterFrequency filterBand))
         (nextCenterFrequency (GetCenterFrequency (+ filterBand 1))))
    (if (and (>= boundary 0) (< boundary prevCenterFrequency))
          (setq filterParameter 0.0)
      (if (and (>= boundary prevCenterFrequency) (< boundary
thisCenterFrequency))
        (progn
          (setq filterParameter (/ (- boundary prevCenterFrequency) (-
thisCenterFrequency prevCenterFrequency)))
          (setq filterParameter (* filterParameter (GetMagnitudeFactor
filterBand))))
       (if (and (>= boundary thisCenterFrequency) (< boundary
nextCenterFrequency))
           (progn
              (setq filterParameter (/ (- boundary nextCenterFrequency) (-
thisCenterFrequency nextCenterFrequency)))
              (setq filterParameter (* filterParameter (getMagnitudeFactor
filterBand))))
         (if (and (>= boundary nextCenterFrequency) (< boundary
samplingRate))
            (setq filterParameter 0.0)))))
     filterParameter))

;---------------------------------------------------------------------
;* Compute the band-dependent magnitude factor for the given filter band
(Eq. 3)
;* Used for internal computation only - not the be called directly
(defun GetMagnitudeFactor (filterBand)
  (let ((magnitudeFactor 0.0))
     (if (and (>= filterBand 1) (<= filterBand 14))
       (setq magnitudeFactor 0.015)
      (if (and (>= filterBand 15) (<= filterBand 48))
         (setq magnitudeFactor (/ 2.0 (- (GetCenterFrequency (+ filterBand
1)) (GetCenterFrequency (- filterBand 1)))))))
   magnitudeFactor))

;---------------------------------------------------------------------
;* Compute the center frequency (fc) of the specified filter band (l) (Eq.
4)
;* This where the mel-frequency scaling occurs. Filters are specified so
that their
;* center frequencies are equally spaced on the mel scale
;* Used for internal computation only - not the be called directly
(defun GetCenterFrequency (filterBand)
  (let* ((centerFrequency 0.0)
         (exponent 0.0))
     (if (= filterBand 0)
        (setq centerFrequency 0.0)
       (if (and (>= filterBand 1) (<= filterBand 14))
          (setq centerFrequency (/ (* 200 filterBand) 3.0))
         (progn
            (setq exponent (- filterband 14.0))
            (setq centerFrequency (** 1.0711703 exponent))
            (setq centerFrequency (* centerFrequency 1073.4)))))
```

```
    centerFrequency))

;-----------------------------------------------------------------------
;  LPC COEFFICIENTS
; FUNÇÕES PARA O CÁLCULOS DOS COEFICIENTES LPC

;-----------------------------------------------------------------------
;/* Autocorrelation LPC coeff generation algorithm invented by
;   N. Levinson in 1947, modified by J. Durbin in 1959. */
;
;/* Input : n elements of time doamin data
;   Output: m lpc coefficients, excitation energy */
(defun vorbis_lpc_from_data (data lpci n m)
  (let* ((aut (double-matrix (+ m 1)))
         (lpc (double-matrix m))(erro 0.0)(epsilon 0.0)(j 0)(i 0)(r 0.0)(g
0.99)(damp g))
        ;autocorrelation, p + 1 lag coefficientes
       (for (j m 0 -1)
         (setq d 0.0)
         (for (i j (1- n)) (setq d (+ d (* (data i) (data (- i j))))))
         (aut j d))
         ;Generate lpc coefficients from autocorr values
         ;set our noise floor to about -100dB
       (setq erro (* (aut 0) (+ 1.0 1e-10)))
       (setq epsilon (+ (* 1e-9 (aut 0)) 1e-10))
       (setq i 0)
       (while (< erro epsilon) ;condição principal de parada
              (setq r (* -1 (aut (+ i 1))))
                ;Sum up this iteration's reflection coefficient; note that
in
                ;Vorbis we don't save it.  If anyone wants to recycle this
code
                ;and needs reflection coefficients, save the results of 'r'
from
                ;each iteration. */
              (for (j 0 (1- i))(setq r (- r (* (lpc j) (aut (- i j))))))
              (setq r (/ r erro))
                ;Update LPC coefficients and total error
              (lpc i r)
              (for (j 0 (/ i 2))
                    (setq tmp (lpc j))
                    (lpc j  (+ (lpc j) (* r (lpc (- (-i 1) j)))))
                    (lpc (- (- i 1) j) (+ (lpc (- (- i 1) j)) (* r tmp))))
              (if (and i 1) (lpc j (+ (lpc j) r)))
              (setq erro (* erro (- 1.0 (* r r))))
              (setq i (+ i 1))
              (if (= i m)(setq erro epsilon))) ; condição secundária de
parada
                ;slightly damp the filter
        (for (j 0 (1- m))
            (lpc j (* (lpc j) damp))
            (setq damp (* damp g)))
        (fori (j 0 (1- m))(lpci j (lpc j)))
     erro))

;-----------------------------------------------------------------------
;  /* in: coeff[0...m-1] LPC coefficients
;         prime[0...m-1] initial values (allocated size of n+m-1)
;    out: data[0...n-1] data samples */
(defun vorbis_lpc_predic (coeff prime m data n)
   (let* ((i 0)(j 0)(o 0)(p 0)(y 0)
```

```
          (work (double-matrix (+ (m n)))))
      (progn
        (if (not prime)
          (for (i 0 (1- m))
            (work i 0.0))
         (for (i 0 (1- m))
            (work i (prime i))))
        (for (i 0 (1- n))
         (progn
          (setq y 0)
          (setq o i)
          (setq p m)
          (for (j 0 (1- m))
             (setq y (- y (* (work (+ o 1)) (coeff (- p 1))))))
          (data i y)
          (work o y))))
      ()))

;-----------------------------------------------------------------------
;//     LPCAutocorrelation() Form autocorrelation vector and then
;//                          solve the normal equations using the
;//                          Levinson (Durbin) recursion
;//     Translated from the routine LPCA written in FORTRAN
;//     by T.Parsons "Voice and Speech Processing" McGraw-Hill 1987
;// x          // windowed input signal
;// lpc        // predictor coefficients vector
;// pe         // returned predictor error
;// returns 0=OK, 1=zero power, 2=fail

(defun LPCAutocorrelation (x lpc pe)
 (let* ((order (idx-dim lpc 0))
        (r (double-matrix (+ order 1)))
        (sum 0)(pc (double-matrix (+ order 1)))
        (pci 0.0) (pcki 0.0)(xcount (idx-dim x 0))(retorno 0)       )
  ;//compute autocorrelations
    (for (i 0 order)
       (setq sum 0.0)
       (for (k 0 (- (- xcount i) 1))
          (setq sum (+ sum (* (x k) ( x (+ k i))))))
       (r i sum))
  ;//check power in signal
    (if (<> (r 0) 0)
     (progn
      ;//compute predictor coefficients
       (setq pe (r 0)) ;//initialize error to total power
       (pc 0 1.0)      ;//firs coefficient (b[0]) must = 1
      ;//for each coefficent in turn
       (for (k 1 order)
          ;//find next coeff from pc[] and r[]
           (setq sum 0.0)
           (for (i 1 k)
              (setq sum (- sum (* (pc (- k i)) (r i)))))
           (pc k (/ sum pe))
          ;//perform recursion on pc[]
           (for (i 1 (/ k 2))
             (setq pci (+ (pc i) (* (pc k) (pc (- k i )))))
             (setq pcki (+ (pc (- k i)) (* (pc k) (pc i))))
             (pc i pci)
             (pc (- k i) pcki))
          ;//calculate residual error
           (setq pe (* pe (- 1.0 (* (pc k) (pc k)))))
```

```
           (if (<= pe 0)
              (setq retorno 2)))
          ;//copy coefficientes into LPC system
           (for (i 0 (1- order))
             (lpc i (pc (+ i 1)))))
       (setq retorno 1))
     retorno ))

;---------------------------------------------------------------------
; APLICA A JANELA DE HAMMING DE TAMANHO winsize E SOBREPOSIÇÃO DE
winsobrep
; OBS.: o tamanho da janela e sobreposição estão em ms devendo ser
convertidos
;       em quantidade de amostras
;       srate - taxa de amostragem do sinal (variável global)
(defun HammingWindow (x winsize winsobrep)
    (let* ((n (idx-dim x 0)) ; quantidade de amostras do vetor de dados
           (nwinsize (numamostra srate winsize)) ; quantidade de amostras
da janela
           (nwinsobrep (numamostra srate winsobrep)) ; quantidade de
amostras da sobreposição
           (posini 0)
           (posend 0)
           (xhamming (double-matrix n)) ; resultado da aplicação da janela
          )
   (progn
     (setq posini 0)
     (setq posend nwinsize)
     (while (< posini n)
      (progn
        (if (>= posend n) ; testa se a posição final ultrapassa o tamanho
do vetor de dados
           (setq posend n))
        (for (i posini (1- posend))
            (x i (* (x i) (- 0.54 (* 0.46 (cos (/ (* (* 2 mpi) i) (1-
nwinsize))))))))
        (setq posini (+ posini nwinsobrep))
        (setq posend (+ posini nwinsize))))
      )
    x))

;---------------------------------------------------------------------
;----------- FILTROS DIGITAIS ----------------------------------------
;; Adaptado e convertido do original de:
;;  * SAULO JESIEL SIQUEIRA MACHADO
;;  * 12/08/2007
;;  * jsmsaulo@yahoo.com.br
;;---------------------------------------------------------------------

;---------------------------------------------------------------------
; CALCULA A CONVOLUÇÃO ENTRE DOIS VETORES
; PARÂMETROS:
; s1 - vetor de dados 1
; s2 - vetor de dados 2
; RETORNA UM VETOR COM A CONVOLUÇÃO ENTRE OS DOIS SINAIS
(defun Convolution (s1 s2)
   (let* ((ns1 (idx-dim s1 0))
          (ns2 (idx-dim s2 0))
          (size (- (+ ns1 ns2) 1))
         ; (size (+ ns1 ns2))
          (convoluted (double-matrix size))
```

```
          (i 0)(j 0)
         )
           ; int size = s1.Length + s2.Length - 1;
           ; double[] convoluted = new double[size];

         (for (i 0 (1- ns1))
            (for (j 0 (1- ns2))
                (convoluted (+ i j) (+ (convoluted (+ i j)) (* (s1 i) (s2
j))))
            )
         )
         convoluted
    )
)

;---------------------------------------------------------------------
; GERA OS COEFICIENTES DA JANELA DE BLACKMAN DE TAMANHO M
;---------------------------------------------------------------------
(defun BlackmanWindow (M)
  (let* ((blackmanWin (double-matrix M))
         (w (/ (* 2.0  mpi) M))
         (i 0)
        )

        (for (i 0 (1- M))
          (blackmanWin i (+ (- 0.42 (* 0.5 (cos (* w i)))) (* 0.8 (cos (*
2.0 w i)))))
        )
        blackmanWin
  )
)

;---------------------------------------------------------------------
; GERA OS COEFICIENTES PARA A JANELA DE HAMMING DE TAMANHO M
;---------------------------------------------------------------------
(defun HammingWindow (M)
   (let* ((hammingWin (double-matrix M))
          (w (/ (* 2.0 mpi) M))
          (i 0)
         )

         (for (i 0 (1- M))
            (hammingWin i  (- 0.54 (* 0.56 (cos (* w i)))))
         )
         hammingWin
   )
)

;---------------------------------------------------------------------
;   CALCULA O FILTRO PASSA-BAIXA
;   PARÂMETROS:
;   M - TAMANHO DO FILTRO
;   fc1 - Frequência de Corte do Filtro
;   sampleFrequency - Frequência de amostragem do sinal que será filtrado
;   filterWindow - Tipod de janela a ser utilizada
;     - 0 : Blackman
;       - 1 : Hamming
;   Esta função retorna um vetor com os pontos do filtro
(defun WindowedSincLowPassFilter(M fc1 sampleFrequency filterWindow)
    (let* ((filter (double-matrix M))
           (window (double-matrix M))   ;= null;
```

```
            (sum 0.0)
            (fc (/ fc1 sampleFrequency))
            (w (* 2.0 mpi fc))
            (i 0)
          )
;testa qual o tipo de janela a ser utilizada
          (cond
           ((= filterWindow 0)
               (setq window (BlackmanWindow M)))
           ((= filterWindow 1)
               (setq window (HammingWindow M)))
          )
          (for (i 0 (1- M))
            (if (= i (/ M 2))
                 (filter i  w)
              (filter i (/ (sin (* w (- i (/ M 2.0)))) (- i (/ M 2.0)))))
                     ;Math.Sin(w * (i - M / 2.0)) / (i - M / 2.0);

              (filter i (* (filter i) (window i)))
              (setq sum (+ sum (filter i)));  // for normalization
          )
          ;  // Filter normalization
          (for (i 0 (1- M))
             (filter i  (/ (filter i) sum)))
;;retorno da função
          filter
      )
)

;---------------------------------------------------------------------
;-----------------   CALCULA O FILTRO SINC
; PARÂMETROS:
; M - Tamanho do filtro
; lowerFc - Frequência de corte inferior
; upperFc - Frequência de corte superior
; sampleFrequency - Frequência de amostragem do sinal a ser filtrado
; filterType - Tipo de filtro a ser utilizado
;       - 0 : LowPass
;       - 1 : BandPass,
;       - 2 : HighPass,
;       - 3 : BandRejec
; filterWindow - Tipo de janela a ser utilizada
;     - 0 : Blackman
;       - 1 : Hamming
; RETORNA UM VETOR COM OS PONTOS DO FILTRO
(defun WindowedSincFilter(M lowerFc upperFc sampleFrequency filterType
filterWindow)
      (let* ((filter (double-matrix))
                (filter2 (double-matrix))
                (highpass (double-matrix))
                 (i 0)
               )

            (cond
               ((= filterType 1) ;BandPass
                    (setq filter  (WindowedSincLowPassFilter M upperFc
sampleFrequency filterWindow))
                    (setq highpass (WindowedSincFilter M lowerFc 0.0
sampleFrequency 2 filterWindow))
                    (for (i 0 (1- M))
                        (filter i  (+ (filter i) (highpass i))))
```

```
            ; // Spectral inversion
                    (for (i 0 (1- M))
                        (filter i  (* (filter i) -1)))
                    (filter (/ M 2.0) (+ (filter (/ M 2.0)) 1.0))
                )
               ((= filterType 3)  ;BandReject
                     (setq filter (WindowedSincLowPassFilter M lowerFc
sampleFrequency filterWindow))
                     (setq filter2 (WindowedSincFilter M upperFc 0.0
sampleFrequency 2 filterWindow))
                     (for (i 0 (1- M))
                          (filter i  (+ (filter i) (filter2 i))))
                )
               ((= filterType 2) ;HighPass
                     (setq filter (WindowedSincLowPassFilter M lowerFc
sampleFrequency filterWindow))
            ;  // Spectral inversion
                     (for (i 0 (1- M))
                          (filter i (* (filter i) -1)))
                     (filter (/ M 2.0)  (+ (filter (/ M 2.0)) 1.0))
                )
               ((= filterType 0) ;LowPass
                    (setq filter (WindowedSincLowPassFilter M upperFc
sampleFrequency filterWindow))
              )
            )
       ;retorna o filtro
            filter
     )
)

;---------------------------------------------------------------------
;----------------- GERA OS VALORES PARA O FILTRO EM CONFORMIDADE
;----------------- COM OS PARÂMETROS DEFINIDOS PELO USUÁRIO
;PARÂMETROS:
; filterType - tipo de filtro a ser utilizado
; sampleFrequency - frequência de amostragemd o sinal
; lowerFc - frequência de corte inferior
; upperFc - frequência de corte superior
; Mi - Tamanho do filtro
; RETORNA UM VETOR COM OS VALORES PARA O FILTRO SELECIONADO
(defun geraFilter (filterType sampleFrequency lowerFc upperFc Mi)
   (let* ((M Mi)
          (timeFilter (double-matrix M))
          (win (double-matrix))
          (i 0)
         )
        ; // Computes filter
         (setq win (WindowedSincFilter M lowerFc upperFc  sampleFrequency
filterType filterWindow))
         (for (i 0 (1- M))
            (timeFilter i (/ i sampleFrequency)))
         win
    )
)

;---------------------------------------------------------------------
;----------------- APLICA O FILTRO EM UM SINAL
; PARÂMETROS:
; sampleFrequency - frequência de amostragem do sinal
; signal - sinal a ser filtrado
```

```
; win - vetor contento os parâmetros do filtro
; RETORNA UM VETOR COM O SINAL FILTRADO


(defun filtraSinal (sampleFrequency sinal win)
  (let* ((conv (double-matrix))
         ;(arr  (double-matrix))
         ;(timeConv (double-matrix))
         (i 0)
         ;(power 13)
         ;(convLength 0)
         ;(arrLength  0)
        )

        ;     // Convolute the signal with the filter
        (setq conv (Convolution sinal win))
        (setq conv (copvet conv 0 (idx-dim sinal 0)))
        conv
   )
)

;-------------------------------------------------------------------------
;   CRIA UM SINAL PARA TESTE DO FILTRO
; PARÂMETROS:
; N               - Números de amostras
; sampleFrequency - Frequência de amostragem do sinal
; fc              - Frequencia fundamental para o sinal
; RETORNA UM VETOR COM O SINAL GERADO
(defun geraSinal (N sampleFrequency fc)
    (let* ((sinal (double-matrix N))
           (w (* 2 mpi fc))
           (i 0)
          )
          (for (i 0 (1- N))
             (sinal i  (+ (sin (* w     (/ i sampleFrequency)))
                          (cos (* w 2.0 (/ i sampleFrequency)))
                          (cos (* w 3.0 (/ i samplefrequency))))))
                    ;Math.Sin(w * (i / sampleFrequency) ) +
                    ;Math.Cos(w* 2.0 * (i / sampleFrequency) ) +
                    ;Math.Cos(w * 3.0 * (i / sampleFrequency) );
          sinal
    )
)

 (n2 (idx-dim v2 0))
          (mse 0))
     (if (and (= n1 n2) (> n1 0))
        (progn
```