UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

DEPARTAMENTO DE COMUNICAÇÕES

Técnicas de Reconstrução de Pacotes Baseadas em Transformada Wavelet e Redes Neurais Aplicadas a Codificadores de Forma de Onda em Telefonia IP

Autor:

Moisés Vidal Ribeiro

Orientador:

João Marcos Travassos Romano

Banca Examinadora:

Prof. Dr. João Marcos Travassos Romano (FEEC/UNICAMP)

Prof. Dr. Carlos Augusto Duque (FACULDADE DE ENGENHARIA/UFJF)

Prof. Dr. Fábio Violaro (FEEC/UNICAMP)

Prof. Dr. Luis Geraldo Pedroso Meloni (FEEC/UNICAMP)

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica.

Campinas, 28 de setembro de 2001

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Ribeiro, Moisés Vidal
R354t Técnicas de reconstrução de pacotes baseadas em transformada wavelet e redes neurais aplicadas a codificadores de forma de onda em telefonia IP / Moisés Vidal Ribeiro.--Campinas, SP: [s.n.], 2001.

Orientador: João Marcos Travassos Romano.
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Wavelets (Matemática). 2. Redes neurais (Computação). 3. Codificador de voz. 4. Processamento de sinais – Técnicas digitais. 5. Internet (Redes de computação). I. Romano, João Marcos Travassos. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Resumo

O presente trabalho investiga o problema da perda de pacotes em aplicações de telefonia IP ou Voz sobre IP (VoIP). Métodos de reconstrução baseados na transformada wavelet e na rede neural MLP aplicados a codificadores de forma de onda, tal como o G.722.1 da ITU-T, são apresentados. Tais métodos apresentam desempenhos superiores aos encontrados na literatura atual, em termos dos valores da SNR e MOS. Além disso, propõe-se um classificador de segmentos de voz, baseado na rede neural MLP, com baixo custo computacional para detecção e supressão de silêncio. Finalmente, uma implementação em DSP de uma solução para telefonia IP, contendo a norma G.722.1 e as técnicas propostas, é descrita.

Abstract

This work deals with the loss of packets in IP telephone or Voice over IP (VoIP) applications. Methods of reconstruction based on wavelet transform and MLP neural network, applied to waveform codec like G.722.1 from ITU-T, are presented. These methods achieve better performance than the existing techniques in terms of SNR and MOS values. Besides, we propose a low burden speech classification (voiced/unvoiced/silence), based on MLP neural network in order to provide silence detection and supression. Finally, a DSP implementation of an IP telephone solution using G.722.1 codec and the proposed techniques is described.

Agradecimentos

Primeiramente gostaria de agradecer a Deus pela sabedoria que nos tem concedido para vislumbrar, compreender e interagir com os frutos de Sua criação.

À minha esposa Patrícia pela incansável paciência, atenção, amor e carinho durante a elaboração e revisão desta tese. Sem sua presença a vida não teria sentido e as realizações passariam em vão.

À minha mãe Sara, irmãos, familiares, pais da minha esposa que sempre torceram por mim e mesmo distantes me ajudaram muito através de orações e apoio.

Ao professor João Marcos T. Romano por ter acreditado no meu potencial e ter-me oferecido apoio, orientação e os recursos necessários para realização desta tese. Sua ética na condução das situações são as lembranças mais marcantes durante este período.

Ao amigos Ricardo Suyama e Rafael Ferrari por me ajudarem a realizar a parte de implementação da presente pesquisa. Posso afirmar que o meu trabalho não seria o mesmo se a contribuição de vocês não estivesse presente.

À Texas Instruments do Brasil, na pessoa dos amigos Núncio Perella e Rafael de Souza, pela disposição em oferecer as placas DSP para realização das implementações em DSP. Graças a esta parceria nos capacitamos na área de software para DSP.

Aos professores Meloni e Fábio Violaro pela discussões durante o transcorrer do trabalho, as quais foram de grande valia e reflexão.

Aos amigos do DSP*Com* durante o período que compartilhamos o pequeníssimo, mas agradável laboratório.

Ao povo brasileiro que indiretamente custeou e custeia estes tipos de trabalhos na expectativa de que os seus resultados e/ou capacitação dos envolvidos possam proporcionar a melhoria das qualidade de vida no nosso país.

Abreviações

ADC	Analog To Digital Converter
ALU	Arithmetic Logic Unit
ATM	Asynchronous Transfer Mode
CPLD	Controlled Programmable Logic Devices
CCS	Code Composer Studio
DAC	Digital to Analog Converter
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DMA	Direct Memory Access
DSL	Digital Subscriber Line
DSK	DSP Starter Kit
DSP	Digital Signal Processor
DST	Discrete Sine Transform
DSTFT	Discrete Short Time Fourier Transform
DTMF	Dual Tome Multi-Frequency
DWT	Discrete Wavelet Transform
EDMA	Enhanced Direct Memory Access
ELT	Extended Lapped Transform
EMIF	External Memory Interface
ETSI	European Telecommunications Standards Institute
EVM	Evaluation Module
FFT	Fast Fourier Transform
HPI	Host Port Interface
IDWT	Inverse Discrete Wavelet Transform
IETF	Internet Engineering Task Force
IMLT	Inverse Modulated Lapped Transform
IP	Internet Protocol
ISDN	Integrated Services Digital Network

ITU-T	International Telecommunications Union – Telecommunication Standardization Sector
IWF	Interworing Function
LAN	Local Area Network
LOT	Lapped Orthogonal Transform
MAC	Multiply Accumulate Unit
McBSP	Multi-channel Buffered serial Port
MC	Multi-point Controller
MCU	Multi-point Control Unit
MFLOPS	Millions of Floating-Point Operations per Second
MIPS	Millions of Instructions per Second
MLT	Modulated Lapped Transform
MLPNN	Multi-layer Perceptron Neural Network
MOPS	Millions of Operations per Second
MP	Multi-point Processor
MPLS	Multi-protocol Label Switching
PCI	Peripheral Component Interface
PGA	Pin Grid Array
PLL	Phase Locked Loops
PQFP	Plastic Quad Flat Pack
PSTN	Packet Switched Telephone Network
QoS	Quality of Service
RAS	Remote Access Server
RBFNN	Radial Basis Function Neural Network
RISC	Reduced Instruction Set Computer
RTP	Real Time Protocol
SBSRAM	Synchronous Burst Static RAM
SCN	Switched Circuit Network
SDRAM	Synchronous Dynamic RAM
SQVH	Scalar Quantized Vector Huffman Coding
TDAC	Time-Domain Aliasing Cancellation
TDMVT	Time Division Multiplexed Voice Network
TQFP	Thin Quad Flat Pack

TW	Transformada Wavelet
UDP	User Datagram Protocol
VLIW	Very Long Instruction Width
VoIP	Voice over IP
WDM	Optical Wavelength Division Multiplexing

1 Introdução

A popularização da Internet, como meio de comunicação de dados à baixo custo, é um fato atual e revolucionário, pois a concepção de técnicas que possibilitam a utilização deste canal de comunicação para aplicações em tempo real, ainda que não projetadas para este fim, permitem a minimização dos custos de ligações telefônicas, videoconferências e teleconferências. Além disso, a recente implantação da Internet 2, ou rede de banda larga, tem tornado possível a utilização mais ampla da Internet como veículo de transmissão de conteúdos (dados e multimídia).

Entretanto, a utilização deste meio de comunicação, principalmente, quando baseado no protocolo IP (*Internet Protocol*), apresenta alguns problemas de ordem prática, tais como taxa de variação do atraso de chegada dos pacotes (*jitter*), congestionamentos na rede, perda de pacotes, falta de ordenação síncrona na recepção dos pacotes, falta de largura de banda disponível para transmissão de conteúdos, atrasos fim a fim, etc. Assim sendo, diferentes técnicas são necessárias para garantir a realização de uma aplicação em tempo real, pois os problemas intrínsecos às redes IP degradam a qualidade das conversações telefônicas, teleconferências, etc.

A telefonia baseada na transmissão de sinais de voz por redes IP, denominada voz sobre IP (*Voice over IP* – VoIP), contempla, além da telefonia somente em redes IP, a interoperabilidade com as redes PSTN (*Packet Switched Telephone Networks*). Como exemplo de padronização VoIP tem-se a norma H.323 para aplicações de teleconferência da ITU-T. Atualmente, a norma H.323 é utilizada pelas mega-corporações de informática e de telecomunicações, tais como IBM, Microsoft, Cisco, Intel, etc.

Uma questão muito importante e ainda em aberto para aplicações VoIP é a perda de pacotes. Estudos recentes mostram que tais valores podem chegar até 12% para redes nacionais e até 50% para redes internacionais ou intercontinentais. Desta forma, a consideração de técnicas que possibilitem a minimização destas perdas se faz necessária para garantir a melhoria da qualidade da conversação.

Diferentes métodos de reconstrução de pacotes têm sido propostos para atenuar o impacto de perda de pacotes em aplicações VoIP, todavia a maioria deles consideram o uso de maior largura de banda para aumentar a redundância dos sinais de voz

transmitidos, minimizando o efeito dos pacotes perdidos. Por outro lado há também a utilização da técnica de embaralhamento e interpolação, com a vantagem de não aumentar a banda disponível. Esta técnica de reconstrução é específica para sinais de voz submetidos à codificação por forma de onda, tais como PCM, G.722, G.727 e G.722.1 da ITU-T. No presente trabalho, novas técnicas de reconstrução de pacotes, baseada nas técnicas de embaralhamento e interpolação dos quadros de voz a partir da Transformada Wavelet e Redes Neurais Artificiais, são investigadas. Em específico, veremos que a técnica baseada na Transformada Wavelet apresenta desempenho em relação superior às demais.

Como o desempenho destas novas técnicas está diretamente atrelado à classificação do sinal de voz em sonoro, surdo ou silêncio, foi desenvolvida uma técnica de classificação de sinais de voz baseada na rede neural MLP (*Multilayer Perceptron Neural Network* – MLPNN).

O codificador a ser utilizado na presente tese é o G.722.1 da ITU-T, o qual contempla transmissão de sinais de acordo com a norma G.722, ou seja, freqüência de amostragem de 16kHz e quantizados a 16 bits. As taxas do G.722.1 são 32kbps, 24kbps e 16kbps, esta última não padronizada pela ITU-T. Este codificador, ainda em fase de normalização final pela ITU-T, é aplicável principalmente em teleconferência e telefonia *wideband* ou de alta qualidade.

A escolha deste codificador é fundamentada pela possibilidade de se propor um único codificador para aplicações em telefonia *narrowband*, telefonia *wideband* e transmissão de áudio. Um fato interessante é que este codificador foi projetado para redes com baixa taxa de erros e a utilização da técnica de reconstrução proposta permite que o mesmo seja robusto às perdas de pacotes.

Como plataformas de desenvolvimento utilizada para implementação do codificador G.722.1, da técnica de reconstrução de pacotes baseada na Transformada Wavelet e do classificador de segmentos de voz são utilizadas as placas DSP EVM TMS320C6201 e DSK TMS320C6211, da Texas Instruments, e um programa elaborado com o Visual C++ implementado em PC. A placa DSK TMS320C6211 foi usada apenas para se verificar o custo computacional da implementação neste hardware.

Para fundamentar teoricamente o trabalho proposto e suas contribuições principais, considerou-se a seguinte divisão dos capítulos:

- **Capítulo 1 ou introdução**: apresenta um panorama geral do presente trabalho.
- **Capítulo 2:** discute o atual estágio de desenvolvimento de VoIP.

- **Capítulo 3:** resume as principais técnicas de reconstrução de pacotes e apresenta novas propostas utilizando a Transformada Wavelet e Redes Neurais Artificiais.
- **Capítulo 4:** comenta a classificação dos segmentos de sinais de voz em sonoro, surdo ou silêncio e apresenta uma nova técnica, com baixo custo computacional, utilizando a Rede Neural Artificial MLP.
- **Capítulo 5:** descreve a implementação de um protótipo para telefonia IP, a partir do uso de placa DSP e de um PC.
- **Capítulo 6:** comenta, a partir dos resultados simulados e obtidos na prática, as conclusões finais.

Além dos capítulos da tese, há os apêndices A, B e C, os quais, respectivamente, reunem informações pertinentes ao trabalho sobre a Transformada Wavelet, a Transformada Lapped Modulada e o Codec G.722.1 da ITU-T.

2 Voz sobre IP (VoIP)

A disponibilidade de redes de telefonia com baixo custo e alta qualidade é uma necessidade premente na sociedade atual, que vem gerando enormes esforços de investimentos e de pesquisas. Neste contexto, um novo paradigma baseado na transmissão de voz sobre IP começou a se desenhar a partir do momento em que a comunicação de dados em forma digital, a baixo custo e via Internet, tornou-se uma realidade. A expressiva disseminação da Internet, alcançada pela implantação de infra-estrutura com diferentes arquiteturas de rede e com capacidade de interoperabilidade, tem possibilitado não só o aumento do tráfego de dados, mas também de voz, áudio e vídeo. Tais condições foram e estão sendo propiciadas pelos gigantescos avanços tecnológicos nas áreas de processamento de sinais, transmissão de dados e capacidade de processamento [HVH99].

Assim, transmissão de voz sobre redes IP, chamada Voz sobre IP (*Voice over IP – VoIP*) tem se transformado numa nova fronteira para telefonia, principalmente, pelos custos extremamente baixos nas ligações de longa distância.

Para fornecer um panorama geral sobre VoIP a seção 2.1 tece comentários sobre as características e benefícios de VoIP, a seção 2.2 trata dos possíveis cenários relativos a VoIP, a seção 2.3 discursa brevemente sobre QoS (*Quality of Service*) em redes IP, a seção 2.4 comenta sobre desenvolvimento de soluções VoIP, a seção 2.5 resume as características da qualidade de voz em aplicações VoIP, a seção 2.6 apresenta os softwares necessários para prover VoIP, a seção 2.7 descreve brevemente a norma H.323 da ITU-T e, finalmente, a seção 2.8 faz um resumo do capítulo.

2.1 Caracterização e benefícios de VoIP

A princípio, o termo Voz sobre IP pode ser definido como a capacidade de realização de chamadas telefônicas sobre uma rede de comutação de pacotes, sem garantia de qualidade de serviço e com capacidade de interoperabilidade com diferentes tipos de redes.

Na telefonia VoIP, as redes IP transmitem os sinais na forma de pacotes de dados sobre uma rede de comutação de pacotes, sem o estabelecimento de uma conexão fixa. Desta forma, os pacotes de dados dos sinais de voz são entregues por diferentes rotas ou

caminhos independentes e disponíveis, a fim de propiciar o melhor uso possível dos recursos. Os pacotes transmitidos recebem cabeçalhos especificando informações de controle [SCFJ96]. Diferentemente, a forma convencional de telefonia, executada sobre redes PSTN (*Packet Switched Telephone Networks*), realiza a transmissão dos sinais como dados binários de uma seqüência síncrona sobre uma rede de comutação de circuitos e com multiplexagem no tempo (*Time Division Multiplexed Voice Network* - TDMVT). Este tipo de rede utiliza um canal de comunicação fixo com largura de banda pré-estabelecida.

VoIP pode ser realizada de três formas distintas [TJ00]:

- **Através de um PC**: Conceitualmente é a mais simples. Basicamente faz uso dos equipamentos multimídia disponíveis nos computadores, tais como microfones, placas de som e alto-falante. As conexões se dão através de modems ligados às redes PSTN ou por meio de LANs (*Local Area Network*), utilizando softwares necessários para realizar e gerenciar as ligações para um outro PC ou para o telefone fixo da rede PSTN. A largura de banda disponível é de 64kbit/s até alguns Mbits/s.
- **Através de um dispositivo DSL** (*Digital Subscriber Line*)**:** Neste caso, o uso do PC é dispensável. O telefone é conectado à Internet através de um conector RJ-11 presente nos terminais do dispositivo DSL, e o firmware e hardware do DSL são projetados para realizar comunicação com telefones da rede PSTN.
- **Telefone IP:** são os telefones tradicionais com conexão direta à Internet e com diferentes algoritmos de processamento de voz implementados em processador digital de sinais. Como exemplo pode-se citar o telefone IP da empresa Telogy [WIT00].

Nesta tese iremos tratar da primeira técnica descrita acima, ou seja, VoIP através de um PC.

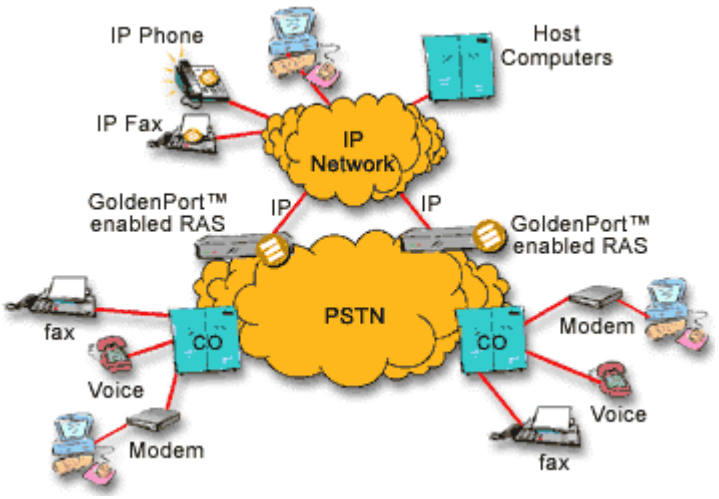
Algumas implicações de ordem prática para adequar a infra-estrutura da Internet para o tráfego de voz são importantes, principalmente as restrições intrínsecas das redes IP, bem como a interoperabilidade com as redes PSTN. Em conseqüência, muitos esforços têm sido conduzidos com intuito de desenvolver produtos que façam uso eficiente das redes IP, mesmo em condições de congestionamento. Desta forma, procura-se sobretudo minimizar os problemas de atrasos e perdas de pacotes característicos de redes, baseadas na transmissão por melhor esforço [BR00]. Assim sendo, a principal contribuição desta tese é, justamente, investigar as perdas de pacotes

em redes IP e propor uma solução que minimize estes efeitos em sinais de voz submetidos à codificação de forma de onda.

Para prover interoperabilidade entre os mundos IP e PSTN, alguns grupos de padronização propuseram normas para VoIP, englobando desde a seleção dos algoritmos de processamento de voz e vídeo até as especificações dos protocolos e dispositivos de interoperabilidade. Os grupos com as principais propostas foram da IETF (*Internet Engineering Task Force*), representando o mundo IP, e o grupo da ITU-T (*International Telecommunication Union – Telecommunication Standartization Sector*), representando as empresas de telecomunicação [ITU98a, ITU98b, ITU98c, ITU98d]. Além destes grupos, a ETSI (*European Telecommunications Standards Institute*) também iniciou esforços para especificar padrões de interoperabilidade entre a Internet e as redes PSTN [ETSI98].

Apesar do alto custo de transmissão e da pouca largura de banda disponível em redes PSTN, as mesmas não serão imediatamente substituídas e durante um certo tempo iremos conviver com sistemas PSTN, VoIP e sistemas híbridos PSTN-VoIP. Provavelmente, as substituições se darão primeiramente em redes privadas de telefonia, devido principalmente à capacidade de investimentos das empresas que necessitam de aplicações multimídia baratas, seguras e de qualidade. Assim sendo, o objetivo inicial de fornecer serviços VoIP está direcionado à reprodução das tarefas realizadas nas redes PSTN sobre as redes IP [ITU98a, ITU98b, ITU98c, ITU98d, ETSI98, IMTC98], tendo como atrativo a diminuição dos custos de operação do sistema e a interoperabilidade com as redes PSTN.

A figura 2.1 mostra genericamente a interoperabilidade entre a rede PSTN e a rede IP. Este esquema pode ser aplicado em outros tipos de redes de comutação de pacotes, tais como Frame Relay e ATM [WIT00]. Como pode ser observado na figura 2.1, dispositivos e protocolos que garantam a interoperabilidade entre as duas redes são imprescindíveis para prover as soluções híbridas (PSTN-IP). Como exemplo de norma de padronização de protocolos, dispositivos e algoritmos, tem-se a norma H.323 da ITU-T [ITU98a].



[1]**Figura 2.1**. Infra-estrutura paraVoz sobre IP (VoIP).

VoIP pode ser aplicado a quase todos os tipos de comunicação de voz, desde a simples conexão entre departamentos até teleconferências transcontinentais, com qualidades especificadas de acordo com as necessidades em questão. Tais requisitos demandam flexibilidade na configuração dos equipamentos VoIP, além das encontradas em rede PSTN.

De um modo geral os benefícios advindos da VoIP se referem à redução de custos nas chamadas de curta e longa distância, simplificação da infra-estrutura e dos protocolos envolvidos, bem como ao potencial de aplicações a longo prazo [RYA97]:

2.2 Cenários para aplicação de VoIP

Basicamente existem cinco cenários possíveis para transmissão de voz sobre redes IP, PSTN e IP-PSTN (híbridas) [HVH99], conforme descrito a seguir e mostrado na fig. 2.2:

- **Cenário 1**: representa o uso das redes PSTN como único meio de transmissão de voz.

  As redes PSTN são baseadas na comutação de circuitos com canais de comunicação de 64kbit/s e acesso através de terminais analógicos ou digitais. Uma característica principal das redes PSTN é a sua capacidade de controle, gerenciamento e estabelecimento de serviços denominada de *Intelligent Network* [ITU98].

  A possibilidade de acesso digital à rede PSTN é realizável com ISDN [KEG96].

[1] Figura Retirada do curso Voice Internet Protocol, no site www.techonline.com.

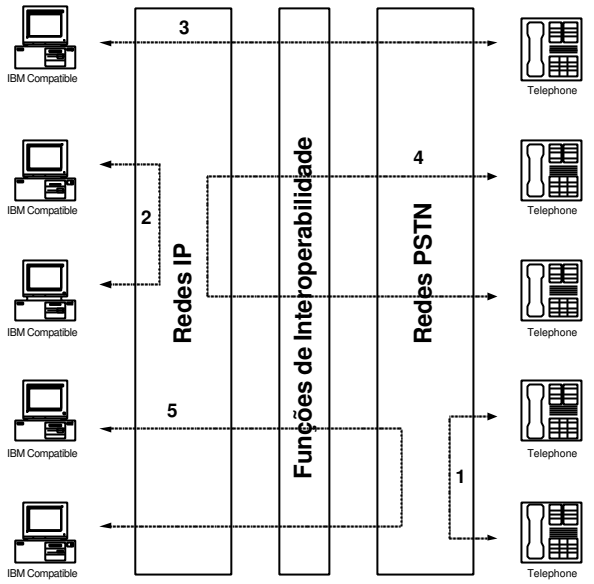


Figura 2.2. Cenários possíveis para comunicação de voz.

- **Cenário 2**: este cenário representa o futuro da telefonia, pois configura a realização de ligações telefônicas unicamente sobre redes por comutação de pacote (redes IP).

  Na telefonia em redes IP, os protocolos e processamentos de serviços específicos, tais como sinalização e codificação dos sinais de voz, são transparentes à camada de rede.

  As aplicações de telefonia sobre as redes IP podem ser realizadas a nível da camada de transporte com o protocolo UDP (*User Datagram Protocol*) [POS80] para garantir tempo real, porém sem garantia de seqüenciamento e entrega dos pacotes ou a partir do protocolo RTP (*Real Time Protocol*) [SCFJ96], implementado na camada de aplicação para garantir o seqüenciamento correto, caso haja uma entrega desordenada de pacotes via rede. Para transferência de dados de sinalização é utilizado o protocolo TCP (*Transmission Control Protocol*) [POS81].

  Os padrões H.323 da ITU-T e os esforços da ETSI [ETSI99 e ETSI00] e IETF [ADEHP99 e SR99] constituem os mais significativos avanços para padronização e, consequentemente, interoperabilidade entre diferentes tipos de equipamento para telefonia via redes IP.

- **Cenário 3**: configura a atual situação da VoIP, pois os dois terminais de telefonia envolvidos fazem uso de diferentes protocolos de comunicação para realizarem acesso às redes IP e PSTN.

Como características intrínsecas deste cenário, temos a necessidade de mapeamento entre os canais de controle de comunicação e protocolos de sinalização.

- **Cenários 4 e 5**: Os mesmos protocolos são utilizados na interface de cada terminal, enquanto outros protocolos são utilizados pelos *backbones* das redes IP e PSTN.

Os cenários 3, 4 e 5 são factíveis através das Funções de Interoperabilidade, (*Interworking Function – IWF*) que realizam a translação das sinalizações e o controle entre diferentes tipos de redes com diferentes protocolos de transporte, gerenciamento, etc. Basicamente as IWF realizam as seguintes tarefas:

- **Adaptação da Sinalização**: consiste, basicamente, no processamento e translação de mensagens de sinalização entre os diferentes protocolos.
- **Controle da Interoperabilidade**: consiste na identificação, processamento e translação de eventos de controle, gerados pelo usuário quando uma conexão é estabelecida.
- **Adaptação dos Protocolos**: consiste na adaptação dos quadros do sinal de voz codificado para transferência de uma rede IP, PSTN ou híbrida, buscando garantir as condições necessárias para estabelecimento e manutenção da qualidade de serviço especificada. Por exemplo, os provedores de serviços tentam, baseados no princípio do melhor esforço, acomodar o maior número possível de ligações telefônicas. Todavia, tal realização demanda a utilização de mecanismos de controle para garantir um uso eficiente da rede.

2.3 QoS em redes IP

A necessidade de garantir QoS para uma aplicação em tempo real especificada para Internet, tal como VoIP, deve levar em consideração os seguintes parâmetros [LU00]:

- Largura de banda sob demanda.
- Baixo atraso fim a fim.
- Variação do atraso (*Jitter*).
- Aceitáveis taxas de erro ou de perda sem retransmissão, pois em aplicações sensíveis ao atraso é inaceitável a retransmissão do sinal.

Os quatros parâmetros/aspectos citados acima são intimamente relacionados. Por exemplo, quando a largura de banda sob demanda é reservada, os outros parâmetros serão baixos, pois os pacotes não precisam permanecer um tempo excessivamente grande nos *buffers* dos roteadores e, conseqüentemente , correrem o risco de serem descartados.

Como as aplicações VoIP requerem diferentes especificações de QoS, a necessidade de prover uma classificação das mesmas é imprescindível. A seguir é apresentada uma atual classificação de QoS em comunicação multimídia (voz, áudio e vídeo)[LU00]:

- **Conversação Fim a Fim**: contempla as aplicações de telefonia e videoconferência. É caracterizada pelas restrições sobre os atrasos fim a fim. O atraso unidirecional máximo, especificado pela Recomendação G.114 da ITU-T, é de 150ms.
- **Serviços de *Broadcasting*** : a principal diferença entre este tipo de serviço e a conversação bidirecional é a sua maior tolerância a atrasos. Como exemplo, podemos ressaltar aplicações de teleconferências unidirecional e rádios via Internet.
- **Aplicações sob Demanda**: as informações são armazenadas e, posteriormente, entregues aos usuários. Podem tolerar grandes atrasos. Por Exemplo, o site www.usinadosom.com.br disponibiliza músicas no formato MP3 para *download* gratuito.

Os níveis de garantia para prover QoS na classificação acima são os seguintes:

- **Garantia Determinística**: Neste caso, a QoS especificada pelo usuário é 100% atendida, considerando a reserva dos recursos da rede para uma largura de banda de pico. Atualmente, a especificação desta garantia pode ser realizada sem a necessidade da largura de banda de pico [KZ95 e LK95]. Normalmente, este nível de garantia apresenta custo alto.
- **Garantia Estatística**: é baseada nas informações de disponibilidade ou não dos recursos de rede e é recomendada para aplicações contínuas, pois usa de maneira eficiente os recursos alocados. Neste caso, a QoS é garantida até uma certa porcentagem.

- **Melhor Esforço**: Nenhuma garantia de QoS é fornecida e a aplicação é executada com os recursos disponíveis. Estes tipos de redes apresentam menor custo. Como exemplo, podemos citar as redes IP.

As ações a serem tomadas para prover garantia de QoS são as seguintes:

- Aumento da velocidade e da largura de banda [LU94 e JJ95] usando, por exemplo, a tecnologia WDM (*Optical Wavelength Division Multiplexing*) [HOW99].
- Utilização dos modelos de arquitetura IntServ [BRA94], DiffServ [NER99] e o protocolo MPLS (*Multiprotocol label Switching*) [ROS99 e IETF01].
- Elaboração de mecanismos alternativos para garantir QoS fim a fim.
- Utilização de técnicas de codificação e reconstrução de pacotes, enfatizando o uso eficiente da largura de banda e a inteligibilidade do sinal [WSL00 e WLS00].

2.4 Desenvolvimento de produtos VoIP

O desenvolvimento atual dos produtos VoIP tem como foco a adaptação das tarefas para realização de uma chamada telefônica em redes PSTN para redes IP e a interconexão destas àquelas e às redes privadas, procurando preservar ou superar a qualidade existente nas redes PSTN.

De um modo geral, o desenvolvimento dos produtos para VoIP levam em conta quatro fatores importantes:

- A qualidade do sinal de voz comparada aos padrões disponível para redes PSTN, mesmo considerando-se redes com diferentes níveis de QoS.
- Diminuição dos efeitos de latência da rede, perda de pacotes e ruptura de conexões, mesmo durante períodos de congestionamentos ou quando múltiplos usuários compartilham os mesmos recursos.
- Interoperabilidade dos serviços fornecidos pelas redes IP e PSTN.
- Gerenciamento do sistema, segurança e forma de bilhetagem.

Os requisitos técnicos básicos para bem suportar as aplicações de VoIP são:

- Algoritmos de compressão de sinais de áudio.
- Algoritmos para supressão de silêncio.
- Garantia de QoS.

- Sinalização para tráfego de voz.
- Comutação para a rede PSTN.
- Algoritmos para controle ou cancelamento de eco acústico.

2.5 Qualidade de voz em aplicações de VoIP

O termo Qualidade de Voz (pode significar diferentes coisas, dependendo do ponto de vista adotado. Entretanto, é uma maneira de quantificar a fidelidade e a inteligibilidade do sinal.

De forma a fornecer um nível de qualidade que seja no mínimo igual ao encontrado em redes PSTN, os seguintes problemas necessitam ser investigados e atenuados:

- **Atraso**: longos atrasos fim a fim resultam em eco e em sobreposição de conversa. O eco se torna um grande problema quando os atrasos *round-trip* são maiores que 150ms e sua percepção passa a ser um problema significativo, obrigando a inclusão de sistemas de controle e cancelamento. Por outro, a sobreposição de conversas se tornam significativas se o atraso unidirecional é maior do que 250ms.
- ***Jitter* (Taxa de variação do atraso)**: O jitter é por definição a variação do tempo de chegada entre os pacotes, introduzido pela variação dos atrasos de transmissão na Internet. A utilização de *buffers* de *jitter* é recomendada para minimização deste problema, apesar do conseqüente aumento do atraso fim a fim.
- **Perda de pacotes**: As redes IP não possuem instrumentos que possam garantir a entrega e o seqüenciamento dos pacotes transmitidos. Uma das principais causas são os períodos de congestionamentos causados por falhas nos *links* de conexão e uso ineficiente dos recursos disponíveis na rede. Existem diferentes técnicas disponíveis para compensar estas perdas com diferentes metodologias de implementação. De modo geral, as perdas de pacotes acima de 10% não são toleradas. As perdas de pacotes e os métodos que possam minimizá-la, independentemente do uso de gerência de tráfego [TANE96], são os enfoques principais desta tese.

2.6 Softwares de funcionalidade para prover VoIP

Os módulos de processamento necessários à funcionalidade de um terminal VoIP são os seguintes:

Processamento da voz (*Voice Processing*): realiza a codificação, decodificação, cancelamento de eco acústico, detecção de voz, geração e detecção de tons DTMF (*Dual Tone Multi-Frequency*), aquisição e reprodução do sinal de voz. Este módulo é normalmente implementado em DSP.

- **Processamento de chamadas (*Call Processing*)**: atua como um *gateway* de sinalização, permitindo que as chamadas sejam estabelecidas através de uma rede de pacotes.
- **Processamento de pacotes (*Packet Processing*)**: processa os pacotes de sinalização e de voz, adicionando um cabeçalho de transporte apropriado antes do envio pela rede IP.
- **Gerenciamento de rede (*Network Management*)**: fornece a funcionalidade dos agentes de gerenciamento ao permitir a bilhetagem, controle remoto de falha, configurações, segurança, atendimento remoto e acesso aos diretórios de discagem.

A fig. 2.3 exemplifica a implementação dos módulos de funcionalidade para soluções VoIP no padrão H.323 da ITU-T.

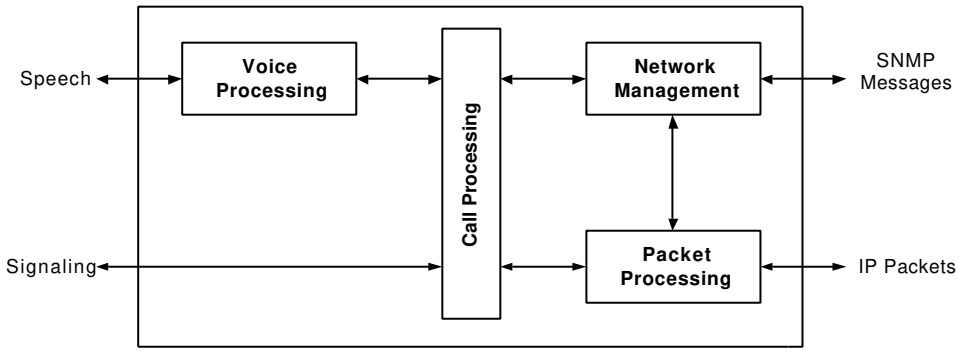


Figura 2.3. Funções nos terminais/*gateway* da norma H.323.

2.7 Norma H.323 da ITU-T

A norma H.323 da ITU-T representa o esforço das principais empresas de telecomunicações e informática para especificar um padrão que permita a interoperabilidade das redes IP com as redes PSTN. Foi inicialmente aprovada em 1996 pelo Grupo de estudo 16 da ITU-T e recebeu uma versão final em janeiro de 1998. Esta norma estabelece os fundamentos necessários a para comunicação de voz e vídeo em redes IP.

A norma H.323 contempla os algoritmos de processamento de voz, vídeo, controle de chamadas, sinalização, gerenciamento das aplicações multimídia, gerenciamento da

largura de banda para conferências entre dois usuários e multi-usuários através da especificação dos seguintes protocolos e componentes [DP00]:

- H.225: Sinalização das chamadas.
- H.245: controle do meio.
- G.711, G.722, G.723, G.728 e G.729: codecs de voz para telefonia e teleconferência.
- H.261 e H.263: codec de vídeo para videoconferência e teleconferência.
- T.120: compartilhamento de dados.
- RTP/TCP: protocolos de transporte.

Q.931: padrão para controle de chamada em redes ISDN (*Integrated Services Digital Network*).

A rápida aceitação desta norma pelo mercado está associada, principalmente, a sua adoção pelas principais empresas de tecnologia e aos seguintes fatores:

- A H.323 define padrões de multimídia para uma infra-estrutura de rede existente, permitindo aos usuários usufruir de soluções multimídia sem grande mudanças na infra-estrutura da rede instalada.
- As redes baseadas em IP estão ficando cada vez mais poderosas. As redes com arquitetura Ethernet estão migrando de 10 Mbps para 100 Mbps e a Gigabit Ethernet está fazendo progressos no mercado.
- Os PCs estão se tornando plataformas de multimídia mais poderosas devido a processadores mais rápidos e a poderosos chips dedicados à multimídia.
- A H.323 provê padrões de interoperabilidade entre LANs e outras redes.
- O fluxo de dados em redes pode ser administrado com o H.323 pelo gerente de rede, o qual pode restringir a largura de banda disponível para conferências.

2.7.1 Benefícios da Especificação da norma H.323

Os benefícios alcançados com esta normatização podem ser agrupados da seguinte maneira [DP00]:

- **Padronização dos codificadores**: a H.323 estabelece padrões para codificação de voz e vídeo.
- **Interoperabilidade**: a norma estabelece um conjunto de sinalizações para chamadas telefônicas e protocolos de controle.

- **Transparência para camada de rede**: a H.323 é projetada para ser transparente à camada de rede, pois é implementada a nível de camada de aplicação.
- **Independência de Plataforma e Aplicação**: a H.323 não está amarrada a um hardware ou sistema operacional, tal como o Windows da Microsoft.
- **Suporte Multiponto**: embora a H.323 possa apoiar conferências de três ou mais usuários sem requerer uma unidade de controle multiponto (*Multipoint Control Unit* - MCU), esta pode definir uma arquitetura mais poderosa e flexível de gerenciamento de conferências multiponto.
- **Administração de Largura de Banda**: os gerentes de redes podem limitar o número de conexões H.323 simultâneas dentro da rede, ou até mesmo a largura de banda disponível para aplicações. Estes limites asseguram que um tráfego crítico não será rompido.
- **Flexibilidade**: uma conferência H.323 pode incluir usuários com tecnologias diferentes. Por exemplo, um terminal com capacidade para voz pode participar de uma conferência com terminais que têm capacidades de dados e/ou vídeo, além de voz. Além disso, um terminal multimídia H.323 pode compartilhar dados de uma vídeoconferência com um terminal de dados T.120, enquanto compartilha voz, vídeo e dados com outros terminais H.323.
- **Conferências entre Redes**: a H.323 estabelece meios de unir sistemas baseados em LAN com sistemas baseados em ISDN. Os usuários de H.323 utilizam tecnologias de codecs comuns para padrões de vídeoconferência.

2.7.2 Componentes do H.323

Os componentes básicos de um sistema H.323 são os terminais, *getways*, *gatekeepers* e unidades de controles multiponto. A figura 2.4 mostra estes elementos e sua interoperabilidade com diferentes redes [DP00 e PIN00].

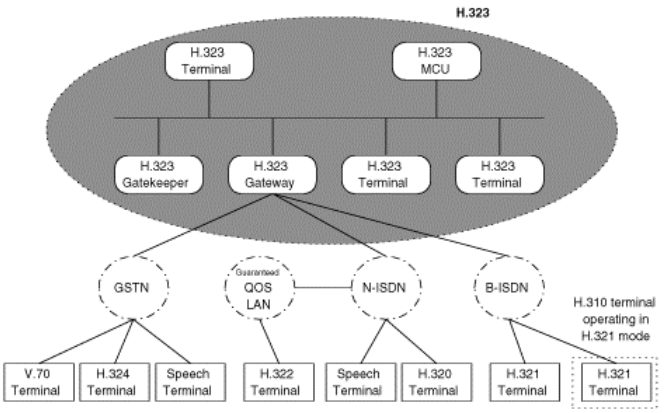


Figura 2.4. Componentes do H.323 e sua interoperabilidade com as diferentes redes.

As funções e características presentes nos diferentes componentes podem ser resumidas conforme segue:

- **Terminais**: são os dispositivos que possibilitam aos usuários a participação em conferências ponto à ponto e multiponto. De acordo com a norma H.323, os terminais devem contemplar unidade de controle, capacidade de transmissão de mídia, codec para voz e vídeo e interface para rede de comutação de pacotes. A fig. 2.5 mostra os codecs e as funcionalidades implementadas num terminal.

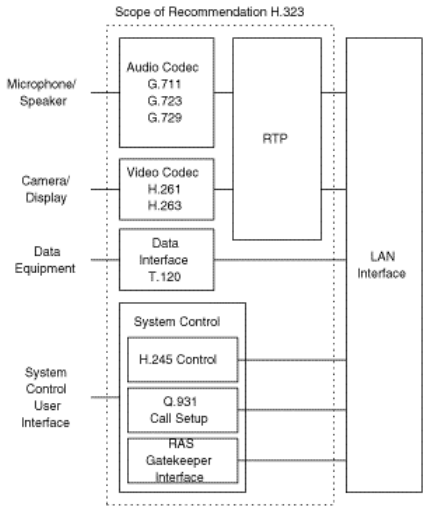


Figura 2.5. Funcionalidades implementadas nos terminais da norma H.323.

- **Gateway**: realizam as funções presentes nas redes de comutação de circuitos (*Switched Circuit Network* – SCN) em comunicações fim a fim por meio da

utilização de protocolos e dos sistemas de comunicação necessários para a interconexão entre as redes IP e PSTN, conforme mostrado na fig. 2.6.

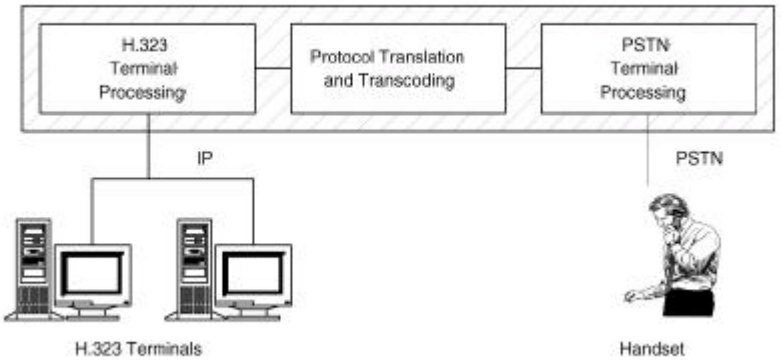


Figura 2.6. Elementos de um gateway entre as redes IP e PSTN.

- **Gatekeeper**: é logicamente separado dos outros elementos da rede, conforme mostra a fig. 2.4. Executa as funções de translação de endereço, controle de aquisição, controle da largura de banda e gerenciamento dos dispositivos presentes na rede. Além disso, pode fornecer a sinalização do controle de autorização e gerenciamento das chamadas.
- **Unidade de Controle Multiponto (MCU)**: A MCU provê conferências entre três ou mais estações. Como elemento do H.323, um MCU obrigatoriamente tem um Controlador Multiponto (*Multipoint Controller* - MC) e até alguns Processadores de Multiponto (*Multipoint Processor* - MP).

  O MC dirige negociações H.245 entre todos os terminais para determinar velocidades comuns de transmissão de voz e vídeo. Além disso, O MC controla recursos de conferência, determinando se os fluxos de voz e vídeo serão *multicast*.

  O MP trata diretamente qualquer tipo de fluxos de mídia, isto é, mistura, chaveia e processa voz, vídeo e/ou bits de dados.

  Tanto MC e como MP podem existir em um componente dedicado ou serem parte de outros componentes H.323. As conferências Multiponto são dirigidas para uma variedade de métodos e configurações H.323. A Recomendação usa os conceitos de conferências centralizadas e descentralizadas, como mostra a Figura 2.7.

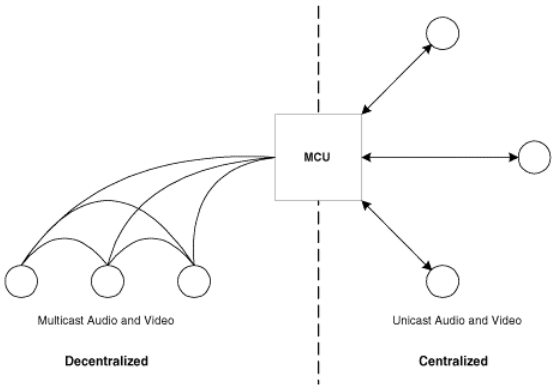


Figura 2.7. Conceitos de conferência centralizada e descentralizada presentes nas MCU.

2.8 Sumário

O presente capítulo teve como principal motivação apresentar um breve tutorial sobre uma tecnologia ainda emergente e relativamente pouco formalizada na literatura. Nele reunimos de forma resumida as principais características de VoIP, seus possíveis benefícios e cenários de aplicação. Foram discutidos aspectos relacionados à qualidade de serviço em VoIP, bem como o desenvolvimento de um produto VoIP. Uma atenção particular é dada à norma H.323, uma vez que se trata do padrão de maior penetração no mercado e, talvez, venha a ser um padrão de fato.

O capítulo seguinte aborda o problema da perda de pacotes e técnicas de reconstrução que, como vimos aqui, é crucial para garantir QoS em VoIP. As técnicas de reconstrução propostas estão entre as principais contribuições desta tese.

3 Técnicas de reconstrução de pacotes

O aumento da largura de banda disponível na Internet e da capacidade dos atuais processadores têm permitido a disseminação da transmissão de voz pela Internet (telefonia IP e VoIP). No entanto, o uso das redes IP para aplicações em tempo real, tal como observado em telefonia e demais aplicações VoIP, ainda não atingiu, o nível de QoS encontrado nas redes PSTN [SWE97]. A inferior qualidade observada na telefonia IP é devida aos grandes atrasos de transmissão [LIN98] e à perda de pacotes [CHF88].

Os atrasos observados na transmissão de dados são devidos à propagação do sinal elétrico pela rede, o processamento do sinal de voz e os *buffers* dos roteadores, os quais causam *jitter* e desordenação dos pacotes enviados. Por outro lado as perdas de pacotes são devidas, principalmente, aos congestionamentos momentâneos encontrados na Internet [LIN98], os quais provocam perdas de até 50% em redes internacionais ou intercontinentais e de até 12% em redes nacionais [LIN98 e CHE97]. Desta forma, técnicas que possibilitem reduzir os efeito de *jitter* e da desordenação, assim como a reconstrução dos pacotes perdidos, são necessárias para garantir um mínimo de garantia de QoS.

Neste capítulo serão introduzidas novas técnicas de reconstrução de pacotes baseada em embaralhamento (*interleaving*) e interpolação, à partir da transformada wavelet e da rede neural MLP. Assim sendo, as técnicas de reconstrução baseadas em embaralhamento e interpolação são discutidas na seção 3.1, enquanto na seção 3.2 são apresentadas as propostas de reconstrução de pacotes a partir da transformada wavelet e da rede neural MLP, já a seção 3.3 apresenta os resultados destas propostas e, finalmente, a seção 3.4 tece comentários finais sobre o capítulo.

3.1 Técnicas de Reconstrução de pacotes baseadas em embaralhamento (*interleaving*) e interpolação

A técnica de embaralhamento é utilizada, principalmente, para minimizar as rajadas de perdas de dados, podendo ocorrer por meio de embaralhamento das amostras ou dos pacotes.

Por outro lado, as técnicas de interpolação permitem a estimação de uma amostra a partir das amostras passadas e futuras de um sinal. Nesta técnica a qualidade da recuperação das amostras depende do algoritmo e critério de interpolação considerado.

A utilização das técnicas de embaralhamento e interpolação é bastante interessante do ponto de vista prático, pois quando ocorre congestionamento e, conseqüentemente, perda de pacotes o pior recurso a ser usado é adicionar redundâncias aos pacotes transmitidos, o que acaba por provocar congestionamentos mais severos.

Discute-se a seguir algumas características das técnicas de embaralhamento e de interpolação, nas seções 3.1.1 e 3.1.2, respectivamente.

3.1.1 Técnicas de embaralhamento

O primeiro nível de embaralhamento é o das amostras. Neste nível as rajadas de erros são distribuídas entre um conjunto de amostras de forma a tornar menos perceptíveis os erros ocorridos durante a transmissão. A separação seqüencial das amostras é especificada pelo Fator de Embaralhamento das Amostras (*Sample Interleaving Factor – SIF*) e são agrupadas em conjuntos contendo uma seqüência de amostras embaralhadas de acordo com o SIF, conforme fig. 3.1. Cada conjunto é enviado separadamente em cada pacote e, por isso, uma perda de pacote corresponderá à perda de um conjunto de amostras.

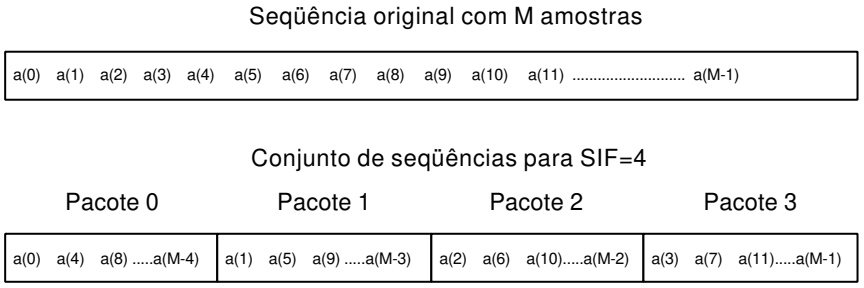


Figura 3.1. Embaralhamento de amostras para *SIF=4*.

O segundo nível de embaralhamento enfoca os pacotes enviados por um canal de comunicação (Internet, linha telefônica ou canal de comunicação móvel). Neste nível o embaralhamento, definido como embaralhamento de pacotes, possibilita a reconstrução de dados se uma rajada de erro afeta mais do que um pacote, reduzindo a probabilidade de mais de uma amostra consecutiva ser perdida. A separação sequencial dos pacotes é definida pelo Fator de Embaralhamento dos Pacotes (*Packet Interleaving Factor* – PIF), conforme mostrado na fig. 3.2.

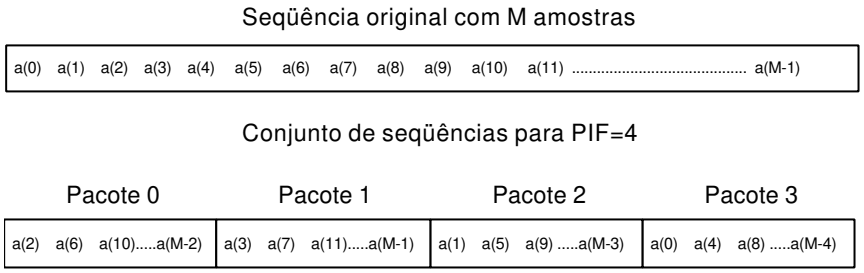


Figura 3.2. Embaralhamento de Pacotes para *PIF=4*.

As informações que devem ser contidas no cabeçalho dos pacotes após o embaralhamento são as seguintes:

- NumSeqFrame: informa qual a seqüência contida no pacote.
- NumIdent: informa a posição de cada pacote no quadro de amostras considerado.
- NumBytes: informa o número de bytes a serem recebidos pelo decodificador.
- EOS: informa ao receptor quando uma transmissão foi realizada com sucesso.

Estas informações de cabeçalho ajudam a identificar quando todos os pacotes não corrompidos pela rede chegam ao receptor. Comprovado o não recebimentos dos pacotes de amostras embaralhadas ou dos pacotes embaralhados, passa-se a usar as técnicas de interpolação.

3.1.2 Técnicas de Interpolação

Estas técnicas levam em conta as amostras adjacentes à perdida para realizar sua estimativa de acordo com algum critério pré-estabelecido. Estas técnicas são principalmente utilizadas em codificadores de voz baseados na forma de onda [LIN98].

Um método típico de interpolação, quando $SIF = 2$, é estimar as amostras perdidas como a média temporal das amostras vizinhas, conforme a fig. 3.3. Este critério é bastante razoável quando as amostras do sinal de voz são correlacionadas. Todavia, os resultados são péssimos quando as amostras do sinal de voz são descorrelacionadas.

A utilização de algoritmos adaptativos, tais como o LMS [LIN98] e o filtro de Kalman [CC97] para interpolação ou estimação das amostras perdidas, apresentam resultados melhores que os alcançados com média temporal.

Uma outra técnica baseada em transformação linear e otimização é proposta em [WL99]. Tal técnica é resumida na seção 3.1.3.

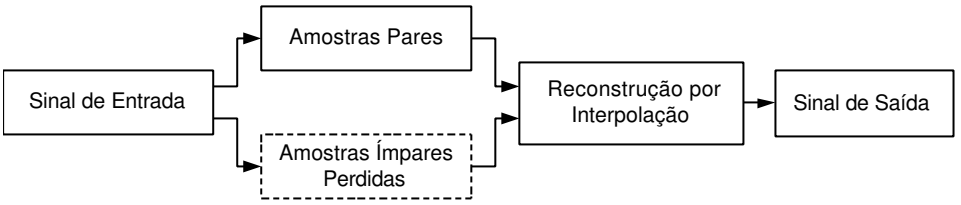


Figura 3.3. Interpolação por média temporal para *SIF=2*.

3.1.3 Técnica de reconstrução de pacotes baseada em transformação - RTRANS

Em [WL99] é proposta uma técnica de reconstrução de pacotes baseada em embaralhamento (*interleaving*) e interpolação. Esta proposta utiliza uma transformação sobre o sinal de voz para reconstrução das amostras pares ou ímpares, quando $SIF = 2$, conforme esquematizado na figura 3.4.

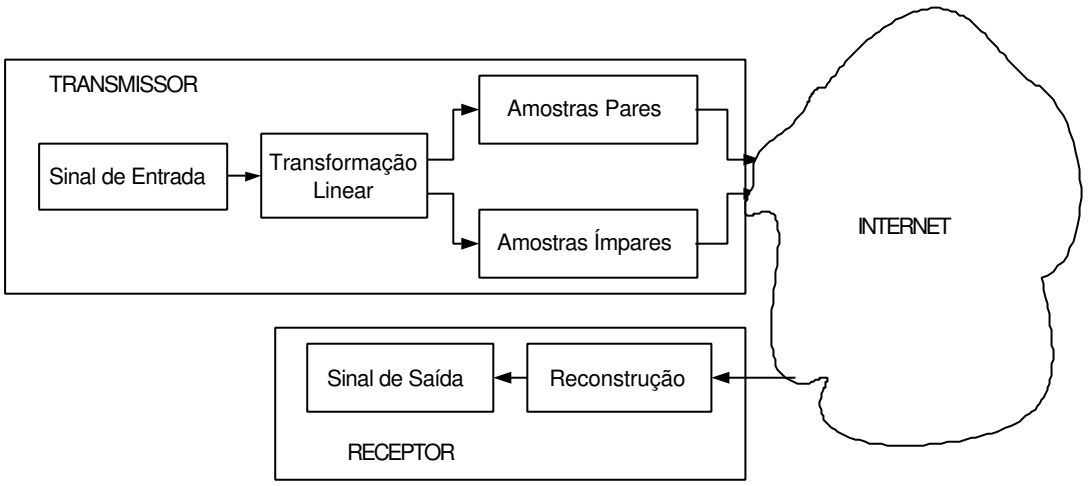


Figura 3.4. Interpolação por transformação linear.

Existem dois casos considerados para obtenção da matriz de transformação:

Caso 1: Perda de um pacote contendo amostras ímpares.

Sejam:

$\mathbf{x} = [x_0, x_1, x_2, ..., x_{2M-1}]^T$ a seqüência de amostras do sinal de voz no transmissor.

$\mathbf{y}_{par} = [y_0, y_2, ..., y_{2M-2}]^T$ a seqüência de amostras pares do sinal de voz recebida pelo receptor.

$$\hat{y}_i = \begin{cases} y_i & ,i = 0,2,...,2M-2 \\ \dfrac{y_{i-1}+y_{i+1}}{2} & ,i = 1,3,...,2M-3 \\ \dfrac{y_{2M-2}}{2} & ,i = 2M-1 \end{cases}$$ representa as amostras reconstruídas do sinal de voz.

O erro de reconstrução (*Error Reconstruction* – RE) é definido como:

$$\begin{aligned} RE &= \sum_{i=0}^{2M-1}(x_i - \hat{y}_i)^2 \\ &= \sum_{n=0}^{M-1}(x_{2n} - y_{2n})^2 + \sum_{n=0}^{M-2}\left(x_{2n+1} - \frac{y_{2n}+y_{2n+2}}{2}\right)^2 + \left(x_{2M-1} - \frac{y_{2M-2}}{2}\right)^2 \end{aligned} \quad (3.1)$$

A minimização de RE é realizada através do cálculo de $y_i, i = 0,2,...,2M-2$ a partir da equação (3.2)

$$\frac{\partial RE}{\partial y_i} = 0, \ i = 0,2,...,2M-2 \quad (3.2)$$

Caso 2: Perda dos pacotes contendo amostras pares.

Sejam:

$\mathbf{x} = [x_0, x_1, x_2, ..., x_{2M-1}]^T$ a seqüência de amostras do sinal de voz no transmissor.

$\mathbf{y}_{impar} = [y_1, y_3, ..., y_{2M-1}]^T$ a seqüência de amostras pares do sinal de voz recebida pelo receptor.

$$\hat{y}_i = \begin{cases} y_i & ,i = 1,3,...,2M-1 \\ \dfrac{y_{i-1}+y_{i+1}}{2} & ,i = 2,...,2M-2 \\ \dfrac{y_0}{2} & ,i = 0 \end{cases}$$ representa as amostras reconstruídas do sinal de voz.

O erro de reconstrução (*Error Reconstruction* – RE) é definido como:

$$\begin{aligned} RE &= \sum_{i=0}^{2M-1}(x_i - \hat{y}_i)^2 \\ &= \left(x_0 - \frac{y_1}{2}\right)^2 + \sum_{n=0}^{M-1}(x_{2n+1} - y_{2n+1})^2 + \sum_{n=0}^{M-2}\left(x_{2n+2} - \frac{y_{2n+1}+y_{2n+3}}{2}\right)^2 \end{aligned} \quad (3.3)$$

A minimização de RE é realizada através do cálculo de $y_i, i = 1,3,...,2M-1$ a partir da equação (3.4).

$$\frac{\partial RE}{\partial y_i} = 0, \;\; i = 1,3,...,2M-1 \tag{3.4}$$

A resolução das equações (3.2) e (3.4) conduz às seguintes relações:

$$\mathbf{y}_{par} = \begin{bmatrix} y_0 \\ y_2 \\ y_4 \\ \vdots \\ y_{2M-2} \end{bmatrix} = \mathbf{A}_{par}^{-1}\mathbf{B}_{par} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{2M-1} \end{bmatrix} \tag{3.5}$$

$$\mathbf{A}_{par} = \begin{bmatrix} 1 & 1/5 & 0 & \cdots & 0 & 0 & 0 \\ 1/6 & 1 & 1/6 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1/6 & 1 & 1/6 \\ 0 & 0 & 0 & \cdots & 0 & 1/6 & 1 \end{bmatrix} \tag{3.6}$$

$$\mathbf{B}_{par} = \begin{bmatrix} 2/5 & 2/5 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 2/3 & 1/3 & \cdots & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1/3 & 2/3 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1/3 & 2/3 & 1/3 \end{bmatrix} \tag{3.7}$$

$$\mathbf{y}_{impar} = \begin{bmatrix} y_1 \\ y_3 \\ y_5 \\ \vdots \\ y_{2M-1} \end{bmatrix} = \mathbf{A}_{impar}^{-1}\mathbf{B}_{impar} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{2M-1} \end{bmatrix} \tag{3.8}$$

$$\mathbf{A}_{impar} = \begin{bmatrix} 1 & 1/6 & 0 & \cdots & 0 & 0 & 0 \\ 1/6 & 1 & 1/6 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1/6 & 1 & 1/6 \\ 0 & 0 & 0 & \cdots & 0 & 1/5 & 1 \end{bmatrix} \tag{3.9}$$

$$\mathbf{B}_{impar} = \begin{bmatrix} 1/3 & 2/3 & 1/3 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 2/3 & 1/3 & \cdots & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1/3 & 2/3 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 2/5 & 2/5 \end{bmatrix} \tag{3.10}$$

As matrizes A_{par}, B_{par}, A_{impar} e B_{impar} têm dimensão MxM e $Mx2M$, respectivamente e os vetores $\mathbf{y_{par}}$ e $\mathbf{y_{impar}}$ obtidos pelas equações (3.5) e (3.8), são, então, codificados e enviados separadamente via Internet.

3.2 Técnicas propostas para reconstrução de pacotes baseadas em embaralhamento (*interleaving*) e interpolação a partir de redes neurais e transformada wavelet

As técnicas de reconstrução de pacotes baseadas em embaralhamento (*interleaving*) e interpolação são promissoras pelo fato de fazerem uso eficiente da largura de banda. Todavia, os resultados encontrados na literatura são ainda pouco animadores. Esta constatação nos levou a considerar a proposição de métodos baseados em estrutura não lineares (redes neurais) e em representação multiresolucional (transformada wavelet). Assim sendo, apresentamos a seguir métodos de reconstrução baseados na rede neural MLP e na transformada wavelet discreta.

3.2.1 Técnica baseada em rede neural artificial MLP (*Multi-Layer Perceptron Neural Network* - MLPNN)

A utilização de técnicas não lineares para resolução de diferentes problemas em engenharia tem possibilitado a obtenção de melhores resultados, principalmente, quando os sinais e sistemas estudados apresentam algum tipo de não linearidade.

Dentre as principais técnicas não lineares modernas, bastante considerada pela comunidade científica, podemos citar a lógica fuzzy [HELL97], os algoritmos genéticos [BS93] e as redes neurais artificiais [HAY99].

As redes neurais artificiais, utilizadas na presente técnica de reconstrução de pacotes, são sistemas de processamento paralelo e distribuído de informações com capacidade de aprendizado e auto-organização. São compostas de um grande número de

unidades simples de processamento chamadas de neurônios, os quais são interconectados para formar redes com capacidade de processar tarefas complexas.

Existem diferentes tipos de redes neurais artificiais. Dentre as arquiteturas mais populares, podemos citar as redes neurais perceptron multi-camadas (MLPNN), as redes neurais com funções base radial (*Radial Base Function* – RBF) e os mapas auto-organizáveis (Self-Organizing Map – SOM). Como a técnica de reconstrução de pacotes a ser apresentada é baseada nas redes neurais MLP, a seção 3.2.1.1 resume as principais característica da rede MLP e a seção 3.2.1.2 descreve, por sua vez, as metodologias propostas.

3.2.1.1 Rede neural artificial MLP

É uma das mais populares arquiteturas de rede neural. Sua unidade básica, neurônio, mostrada na fig. 3.5 é composta por uma combinação linear das entradas e de uma função de ativação. A saída de um neurônio é a combinação linear das entradas ponderadas adicionadas ao termo de polarização (*bias*), submetida à uma funçao linear ou não linear. Matematicamente, para cada neurônio tem-se:

$$x = f(\sum_{j=0}^{M-1} W_j x_j + b_j) \tag{3.11}$$

Sendo $x_j, j = 0,1,...,M-1$ a j-ésima entrada do neurônio, $W_j, j = 0,1,...,M-1$, as correspondentes ponderações ou pesos de x_j e b o termo de polarização.

A escolha da função de ativação é dependente da aplicação a ser considerada [TU97], podendo ser uma função linear ou não linear, tais como a função identidade, a função sigmoidal, a função *sign*, etc.

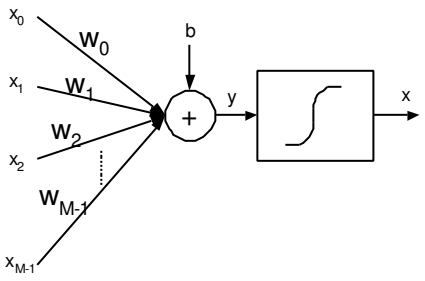


Figura 3.5. Neurônio Perceptron.

A fig. 3.6 mostra a rede neural perceptron multi-camada. Conforme pode ser observado, as informações são processadas a partir da camada de entrada (*input layer*)

para a camada de saída (*output layer*), sendo que entre elas existem camadas intermediárias de processamento. A saída x_{ik} do neurônio (i,k) é dada por:

$$x_{ik} = f(\sum_{j=0}^{N(i-1)-1} W_{ijk} x_{i-1\,j} + b_{ik}) \tag{3.12}$$

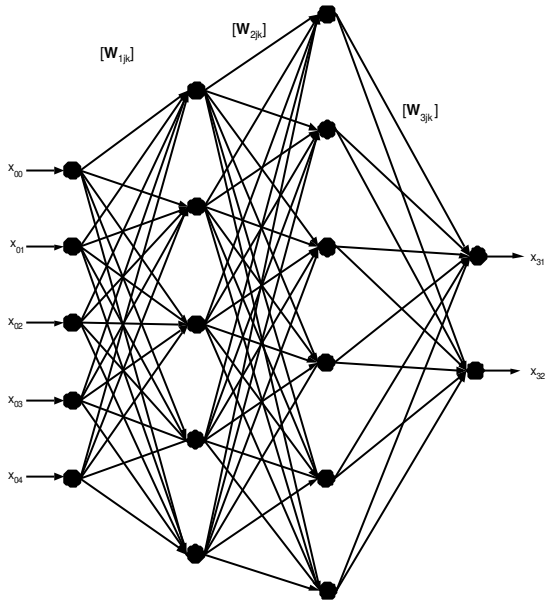


Figura 3.6. Exemplo de uma Rede Neural Perceptron Multi-Camada com 3 camadas.

Sendo i, $i = 1..., K$, a camada da rede neural MLP, x_{ik} a saída do k-ésimo neurônio da camada i, W_{ijk} a ponderação entre $x_{i-1\,j}$ e o k-ésimo neurônio camada i e $N(i)$ é o número de neurônios da camada i.

A adaptação dos parâmetros ou pesos da rede pode ser realizada de duas maneiras:

- Explorando-se a propriedade de aproximação universal da rede neural MLP [HOR89]; ou
- Por meio do algoritmo de retro-propagação (*Backpropagation Algorithm* – BPA) [LIPP87].

A MLPNN implementada para realizar interpolação das amostras perdidas, conforme a fig. 3.7, tem apenas uma camada intermediária. Os parâmetros ou pesos são atualizados pelo critério de minimização da função erro quadrático médio e a função de ativação usada é a tangente hiperbólica. O custo computacional desta implementação é de $4H(M+5)+3(H+1)N$ somas e $2H(M+4)+2(H+1)N$ multiplicações [JRRLF01], sendo H o número de neurônios, M o número de entradas e N o número de saídas da rede neural MLP.

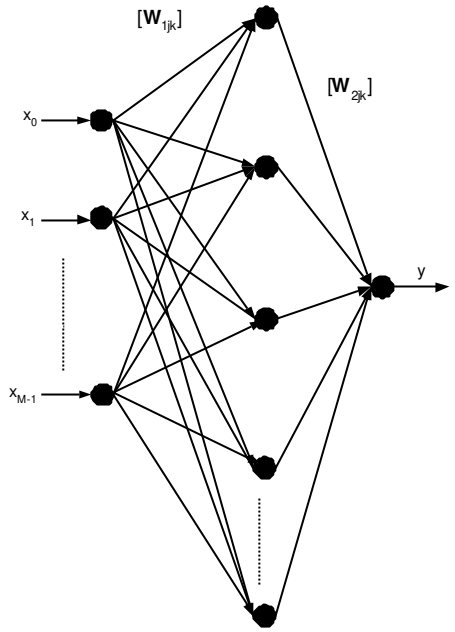


Figura 3.7. Rede Neural MLP com uma camada escondida.

3.2.1.2 Metodologias baseadas na MLPNN

As metodologias propostas neste trabalho são as seguintes:

- **Primeira Metodologia (MLP1)**: Conforme mostrado na fig. 3.8, consiste na utilização de um algoritmo não-linear, MLPNN, para realizar a interpolação das amostras perdidas (pares ou ímpares). A motivação para esta proposição é verificar se a estimação ou interpolação das amostras perdidas dos sinais de voz por uma técnica não linear apresenta resultados significativos. A consideração $SIF = 2$, escolhida de acordo com [LIN98 e WL99], é suficiente para minimizar as perdas de pacotes na Internet a nível nacional.

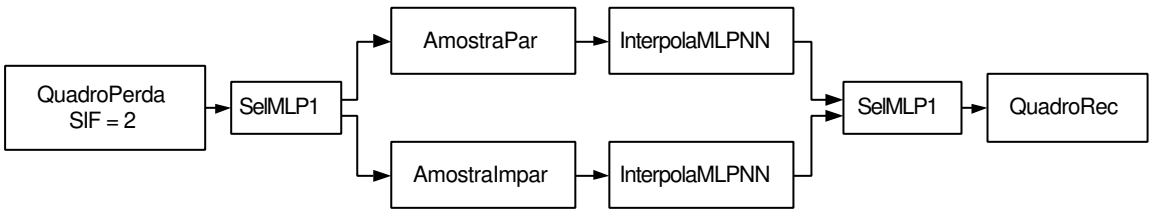


Figura 3.8. Primeira Metodologia (MLP1).

A seguir é descrita de maneira sucinta e objetiva as funções de cada bloco do diagrama correspondente à metodologia MLP1:

- **QuadroPerda**: contém o *buffer* que armazena as amostras de sinais de voz entregue via conexão à internet. As amostras armazenadas são as pares ou as ímpares.
- **SelMLP1**: verifica se a perda de amostras ocorrida corresponde às amostras pares ou ímpares do quadro de sinal de voz.
- **AmostraPar**: contém o *buffer* para armazenar as amostras pares recebidas pela conexão à Internet.
- **AmostraImpar**: contém o *buffer* para armazenar as amostras ímpares recebidas pela conexão à Internet.
- **InterpolaMLPNN**: realiza a interpolação das amostras de sinais de voz perdidas na Internet, utilizando a MLPNN.
- **QuadroRec**: contém o *buffer* para armazenar o quadro de amostras reconstruído pelo bloco InterpolaMLPNN.

• **Segunda Metodologia (MLP2)**: A diferença entre esta metodologia e a anterior está na consideração de duas MLPNN independentes para interpolar, separadamente, os sinais de voz, cujo coeficiente de correlação de atraso unitário é próximo de 1 e aqueles de coeficiente de correlação próximo a 0, conforme figura 3.9.

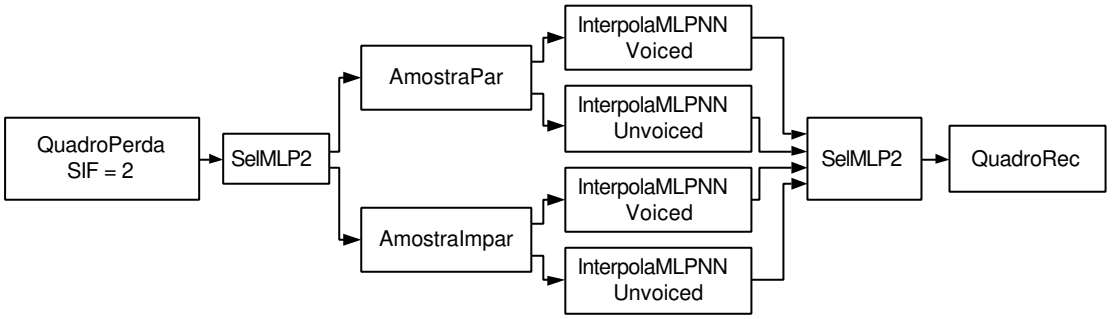


Figura 3.9. Segunda Metodologia (MLP2).

Os blocos constituintes desta metodologia realizam as seguintes funções:

- **QuadroPerda**: tem a mesma função definida na MLP1.
- **SelMLP2**: além de verificar se a perda de amostras ocorrida corresponde às amostras pares ou ímpares, também verifica se o quadro recebido corresponde a sinais de voz surda (*voiced speech*) ou sonora (*unvoiced speech*). Estas informações permitem realizar a interpolação apropriada das amostras pares ou ímpares, utilizando uma MLPNN treinada para amostras descorrelacionados e outra MLPNN para amostras correlacionados.

- **AmostraPar**: conforme especificado em MLP1.
- **AmostraImpar**: conforme especificado em MLP1.
- **InterpolaMLPNNVoiced**: realiza a interpolação das amostras correlacionadas do sinal de voz e perdidas na Internet, utilizando uma MLPNN.
- **InterpolaMLPNNUnvoiced**: realiza a interpolação das amostras descorrelacionadas do sinal de voz e perdidas na Internet, utilizando uma outra MLPNN.
- **QuadroRec**: conforme especificado em MLP1.

3.2.2 Técnica de Reconstrução baseada na Inversa da Trasnformada Wavelet – RITW

O uso da transformada wavelet para prover a reconstrução ou interpolação das amostras perdidas se dará pela exploração de sua característica multiresolucional [MAL89]. A fim de tornar claro e simples o método a ser proposto, devemos primeiramente considerar que a transformada wavelet pode ser compreendida e interpretada a partir da teoria de bancos de filtro [BGG98, DAU92 e VK95], ou seja, um sinal submetido à transformada wavelet corresponde aos processos de filtragem e sub-amostragem. Por outro lado, quando o sinal transformado é submetido a transformada inversa de wavelet tem-se como correspondente os processos de sobre-amostragem e filtragem, conforme descritos no apêndice A.

Pode-se ainda analisar a transformada wavelet como a projeção de um sinal em vários sub-espaços ortogonais [BGG98]. Estes sub-espaços podem ser visualizados pela fig. 3.10, onde $W_i, i = 0,1,2$ e $\nu_i, i = 0,1,2,3$, correspondem aos sub-espaços das funções wavelet e *scaling,* respectivamente.

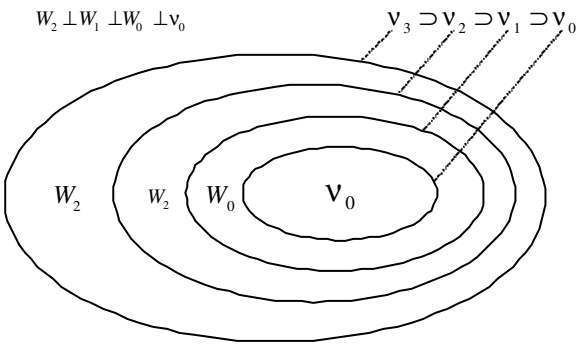


Figura 3.10. Sub-espaços das funções wavelet e *scaling*.

Analisando o esquema apresentado na figura 3.10, podemos verificar que a transformada wavelet executa uma projeção multiresolucional. Tal representação é bastante pertinente, pois permite analisar um sinal a partir de diferentes níveis de resolução.

Com base na característica multiresolucional presente na transformada wavelet chegamos a seguinte idéia intuitiva para reconstrução de pacotes:

1. Primeiramente o sinal de voz é suposto pertencer ao sub-espaço $\nu_j, j \in Z$ [BGG98] e o sinal de voz com perdas, $s_p(n) = s(2n), n = 0,1,...,M-1$, é suposto pertencer ao sub-espaço $\nu_{j-1}, j \in Z$, de forma que o erro entre o sinal original e o sinal com perda corresponde à projeção do sinal original no sub-espaço W_{j-1}. Tal consideração é bastante plausível quando o sinal corresponde a uma voz sonora, pois a projeção em W_{j-1} contém as altas freqüências do sinal de voz e, neste caso, são pouco representativas. Desta forma, o sinal $s(n)$ pode ser reconstruído a partir do sinal com perda, $s_p(n)$. Matematicamente esta estimativa de $s(n)$ é dada pela equação (3.28). O fator *G* é considerado na equação 3.28, pois o sinal $s_p(n)$ contém em média metade da energia do sinal $s(n)$.

$$\tilde{s}(n) = G\sum_k s_p(k)2^{(j-1)/2}\varphi(2^{j-1}n-k) \tag{3.28}$$

Finalmente, como o sinal $s_p(n)$ corresponde às amostras não perdidas do sinal $s(n)$, as mesmas devem substituir as suas correspondentes estimativas em $\tilde{s}(n)$.

2. Para o caso em que o sinal de voz contém informações significativas nas altas freqüências, a solução apresentada na equação (3.28) não possibilita um bom nível de recuperação. Desta forma, propomos a equação (3.29) para recuperação do sinal de voz surda.

$$\tilde{s}(n) = G_1\sum_k s_p(k)2^{(j-1)/2}\varphi(2^{j-1}n-k) + G_2\sum_k s_p(k)2^{(j-1)/2}\psi(2^{j+1}n-k) \tag{3.29}$$

Os valores de ganho G_1 e G_2 são acrescentados, pois o sinal $s_p(k)$ tem em média metade da energia do sinal $s(n)$. Assim sendo, a partir das informações contidas no

sinal com perdas, $s_p(n)$, pode-se obter uma estimativa de $s(n)$. De igual maneira ao primeiro caso, as amostras de $s_p(n)$ devem substituir as estimativas correspondentes obtidas pela equação (3.29)

As equações (3.28) e (3.29) podem ser representadas por um banco de filtro de síntese correspondentes à transformada inversa de wavelet, conforme as figs. 3.11 e 3.12, respectivamente.

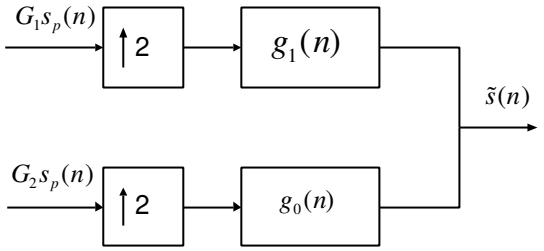


Figura 3.11. Banco de Síntese aplicado à reconstrução de pacotes de sinais de voz sonora.

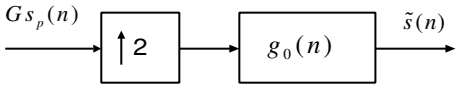


Figura 3.12. Banco de Síntese aplicado à reconstrução de pacotes de sinais de voz surda.

3.3 Resultados

Neste item comparamos os métodos propostos (RITW, MLP1 e MLP2) com o método apresentado em [WL99] (RTRANS), considerado como o mais atual método de reconstrução de pacotes para codificadores baseados na forma de onda. Serão considerados os bancos de dados de sinais de voz, amostrados a 44,1k amostras por segundo, codificados com 16 bits, obtidos diretamente do MIT Media Laboratory [MIT01] e submetidos ao conversor de taxa de amostragem, proposto pela VLSI Solution [VLSI99], para obtenção de sinais a 16k amostas por segundo e a 16 bits. Estes arquivos de voz são de domínio público e tipicamente utilizados para comparar desempenho de codificadores.

Os arquivos de voz utilizados nas simulações e obtidos no MIT Media Laboratory, amostrados a 16kHz e 16bits/amostra, são os seguintes:

- **Mlab1**: representa o arquivo spfe49_1.wav, com voz feminina em inglês.
- **Mlab2**: representa o arquivo spff51_1.wav, com voz feminina em francês.
- **Mlab3**: representa o arquivo spfg53_1.wav, com voz feminina em alemão.
- **Mlab4**: representa o arquivo spme50_1.wav, com voz masculina em inglês.
- **Mlab5**: representa o arquivo spmf51_1.wav, com voz masculina em francês.
- **Mlab6**: representa o arquivo spmg54_1.wav, com voz masculina em alemão.

Os critérios de análise utilizados são os seguintes:

- Cálculo da SNR (relação sinal-ruído – *signal-noise rate*) total do sinal reconstruído.
- Análise subjetiva com 6 pessoas distintas (três mulheres e três homens).

As condições de simulação, da mesma forma como considerados em [LIN98 e WL99], são os seguintes:

- Condição 1: Simulação das técnicas de reconstrução sem a codificação do sinal de voz, considerando-se embaralhamento *SIF=2,* perda das amostras ímpares e reconstrução das mesmas a partir das amostras pares submetidas às técnicas de reconstrução descritas e propostas nas seções anteriores. Esta situação e ilustrada na figura 3.13.

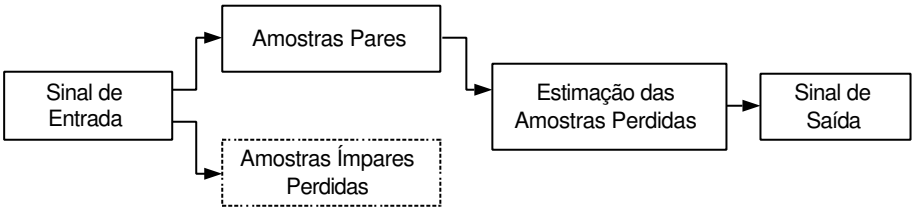


Figura 3.13. Diagrama de bloco da condição 1.

- Condição 2: Simulação das técnicas de reconstrução com a codificação do sinal de voz com o codec G.722.1 considerando-se embaralhamento *SIF=2,* perda das amostras ímpares e reconstrução das mesmas a partir das amostras pares submitidas às técnicas de reconstrução descritas e propostas nas seções anteriores. Esta figura é ilustrada na figura 3.14.

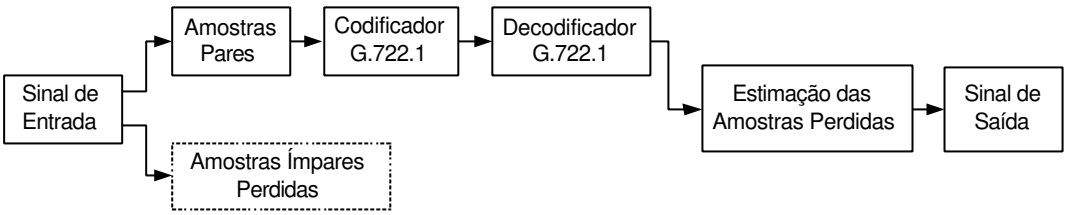


Figura 3.14. Diagrama de bloco da condição 2.

Para a técnica RITW, os seguintes parâmetros são considerados: $G = 2$, $G_1 = 1/2$ e $G_2 = 5/2$, os quais foram obtidos através de simulações.

As figs. 3.15 a 3.20 mostram o valor da SNR em função do comprimento do filtro de wavelet Daubechies para os bancos de dados **Mlab1, ..., Mlab6**, respectivamente. Nestas figuras considera-se a perda de todas as amostras ímpares e reconstrução das mesmas, a partir das amostras pares. O algoritmo de reconstrução empregado é o RITW em ponto flutuante a 64 bits e em ponto fixo a 16 bits. Estes resultados correspondem ao sinal de voz não submetido ao codec G.722.1. Como pode ser observado, as simulações em ponto fixo a 16 bits e em ponto flutuante a 64 bits não apresentam diferenças significativas. Estes resultados foram compilados com o objetivo de verificar o desempenho da técnica RITW com aritmética em ponto fixo a 16 bits, pois na implementação em DSP em tempo real a aritmética é em ponto fixo e a 16 bits.

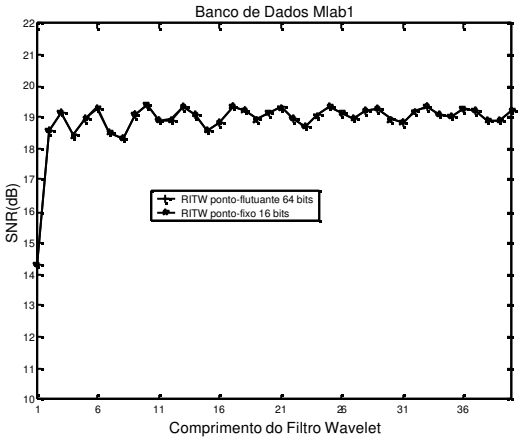


Figura 3.15. SNR x Comprimento do Filtro

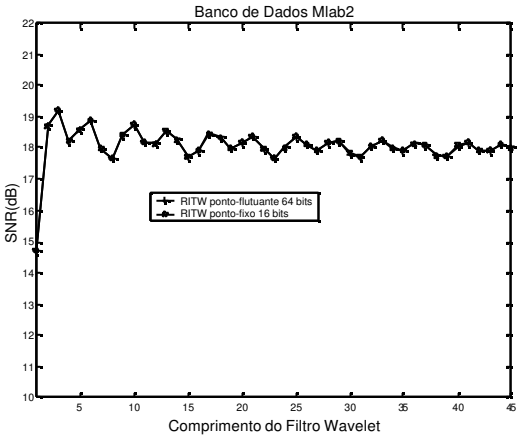


Figura 3.16. SNR x Comprimento do Filtro

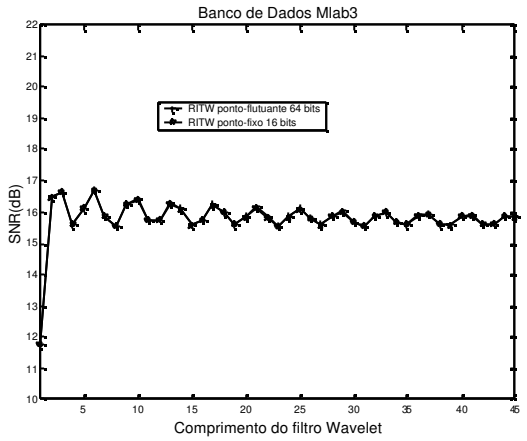


Figura 3.17. SNR x Comprimento do Filtro

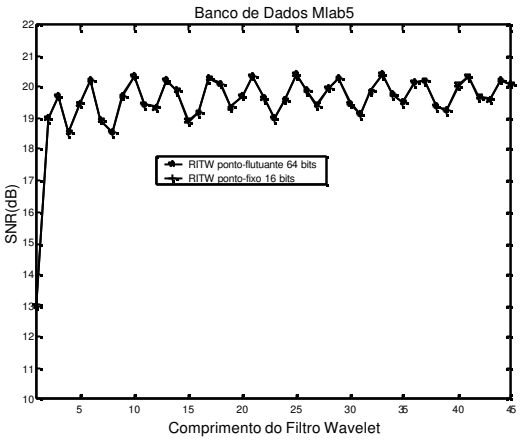


Figura 3.19. SNR x Comprimento do Filtro

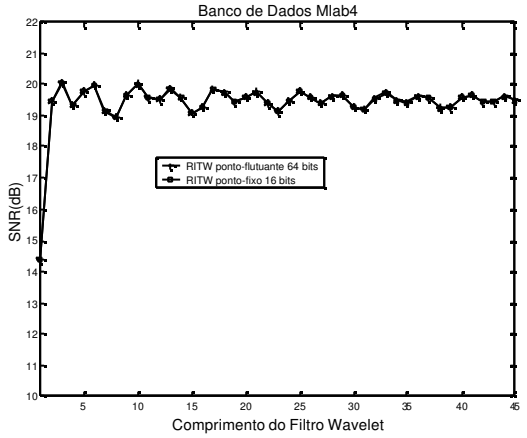


Figura 3.18. SNR x Comprimento do Filtro

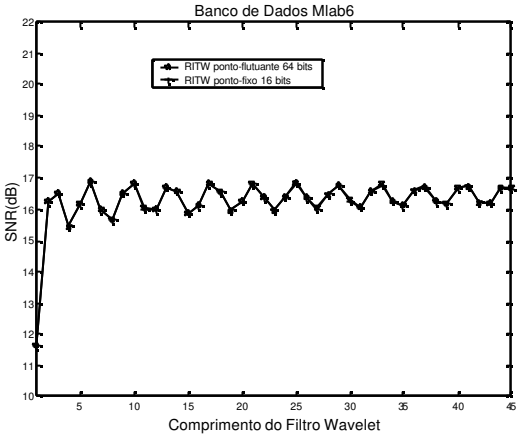


Figura 3.20. SNR x Comprimento do Filtro

As figs. 3.21 a 3.26 mostram o valor da SNR em função da wavelet Daubechies para os arquivos **Mlab1, ..., Mlab6**, respectivamente. Nestas figuras considera-se a reconstrução das amostras ímpares dos bancos de dados, a partir das amostras pares submetidas ao codec G.722.1 de voz. O algoritmo de reconstrução considerado é o RITW em ponto fixo à 16 bit. Como pode ser observado nas figuras abaixo, a SNR dos sinais codificados a 16, 24 e 32kbps apresentam valores inferiores ao obtido sem a codificação. Tais resultados já eram esperados, pois o codec G.722.1 retira as redundâncias do sinal de voz para alcançar as taxas de 16, 24 e 32kbps.

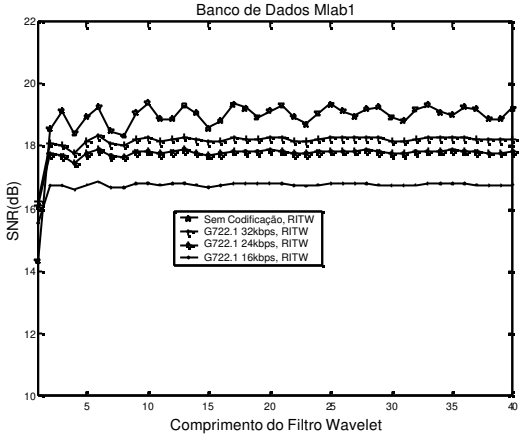


Figura 3.21. SNR x Comprimento do Filtro

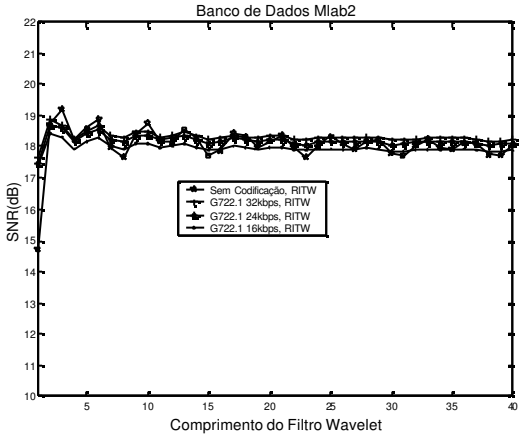


Figura 3.22. SNR x Comprimento do Filtro

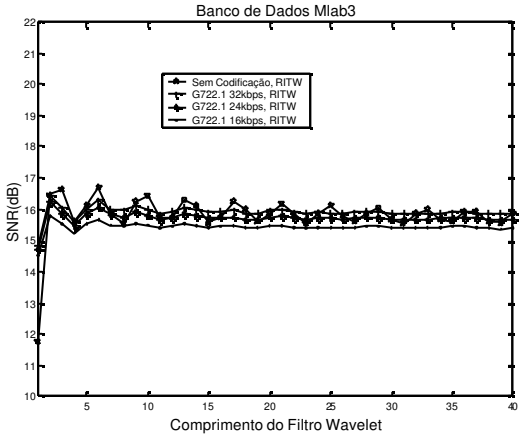


Figura 3.23. SNR x Comprimento do Filtro

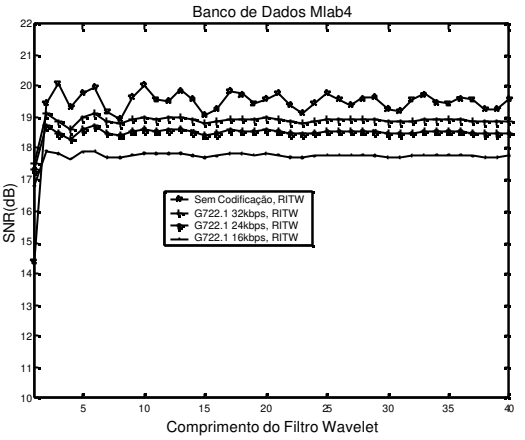


Figura 3.24. SNR x Comprimento do Filtro

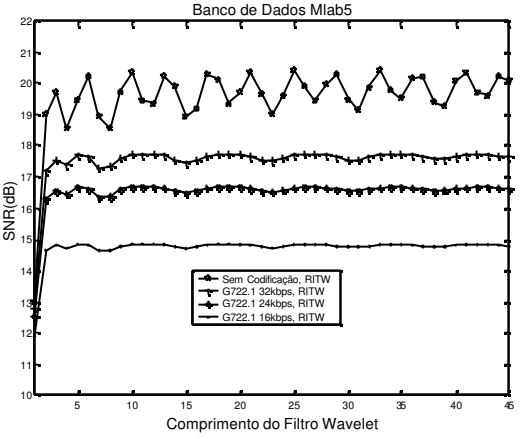


Figura 3.25. SNR x Comprimento do Filtro

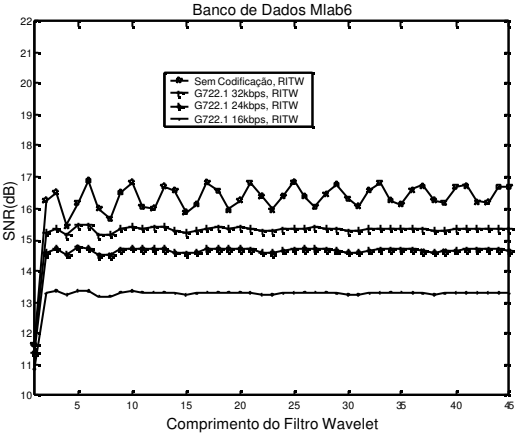


Figura 3.26. SNR x Comprimento do Filtro

As tabelas 3.1, 3.2 e 3.3 apresentam os valores da SNR observados em sinais de voz, com as seguintes características:

- Sinais de voz submetidos a condição 2.
- Uso da técnica de reconstrução RITW implementada a partir da escolha da wavelet Daubechies 10, a qual apresentou em média o melhor SNR, conforme os gráficos 3.21 a 3.26.

Como pode ser observado nas tabelas 3.1 a 3.3, alguns valores de SNR para sinais de voz com perdas e reconstruídos usando a técnica RITW são superiores ao sinais obtidos usando o codec G.722.1. Tal situação é encontrada visto que os valores de ganho G_1 e G_2 apresentam valores superiores a um ganho real igual a 1.

Tabela 3.1. Comparação do SNR para taxa de 16kbps.

Banco de Dados	SNR do G.722.1 à 16kbps	SNR do G.722.1 à 16kbps Com Perda + RITW
Mlab1	15,38	16,82
Mlab2	18,35	18,10
Mlab3	15,98	15,50
Mlab4	16,91	17,85
Mlab5	15,79	16,85
Mlab6	14,71	14,56

Tabela 3.2. Comparação do SNR para taxa de 24kbps.

Banco de Dados	SNR do G.722.1 à 24kbps	SNR do G.722.1 à 24kbps Com Perda + RITW
Mlab1	16,03	17,85
Mlab2	19,42	18,36
Mlab3	17,51	15,81
Mlab4	18,45	18,61
Mlab5	18,20	17,91
Mlab6	17,19	15,32

Tabela 3.3. Comparação do SNR para taxa de 32kbps.

Banco de Dados	SNR do G.722.1 à 32kbps	SNR do G.722.1 à 32kbps com Perda + RITW
Mlab1	16,29	18,26
Mlab2	19,84	18,46
Mlab3	18,03	16,01
Mlab4	18,83	18,99
Mlab5	18,92	18,47
Mlab6	18,08	15,63

As figuras 3.27 a 3.32 ilustram o desempenho em termos da SNR da técnica RTRANS [WL99] para a condição 1, considerando-se ponto flutuante a 64 bits, em função de M (dimensão das matrizes A_{par}, B_{par}, A_{impar}, B_{impar} definidas na seção 3.1.3). A consideração desta técnica em ponto fixo a 16 bits não foi considerada, pois a mesma não foi implementada em DSP.

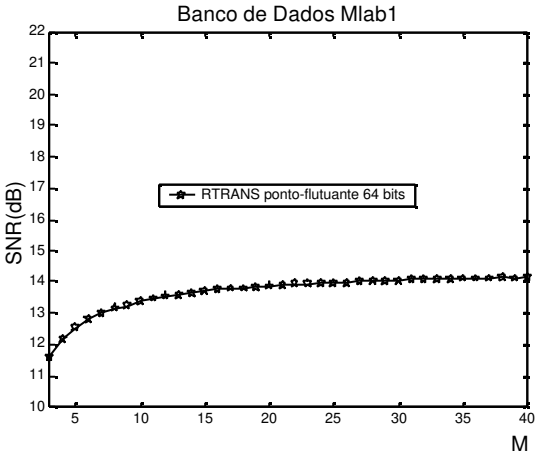


Figura 3.25. SNR x M.

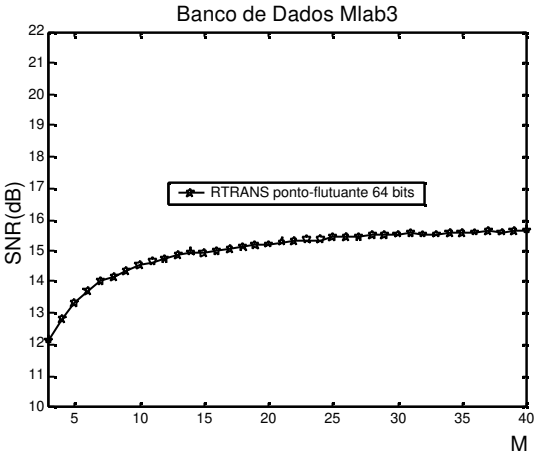


Figura 3.27. SNR x M.

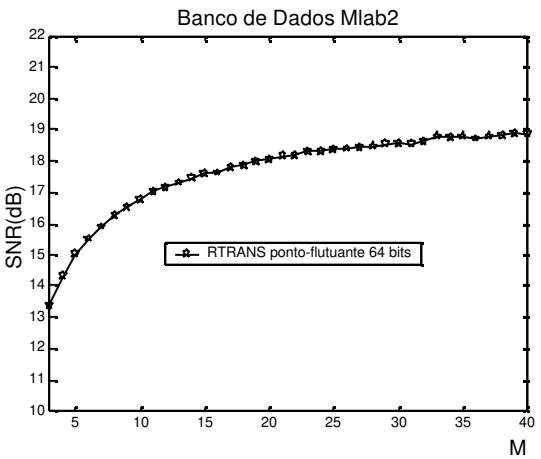


Figura 3.26. SNR x M.

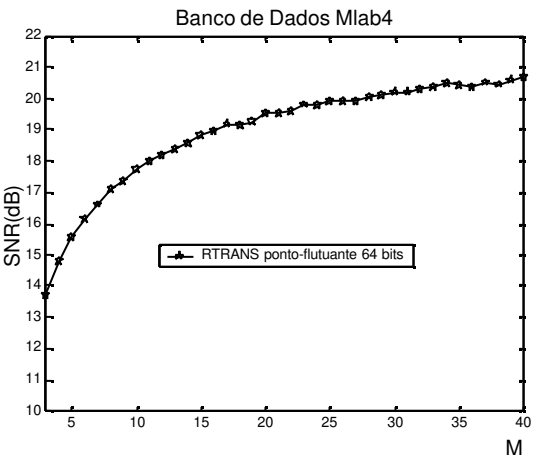


Figura 3.28. SNR x M.

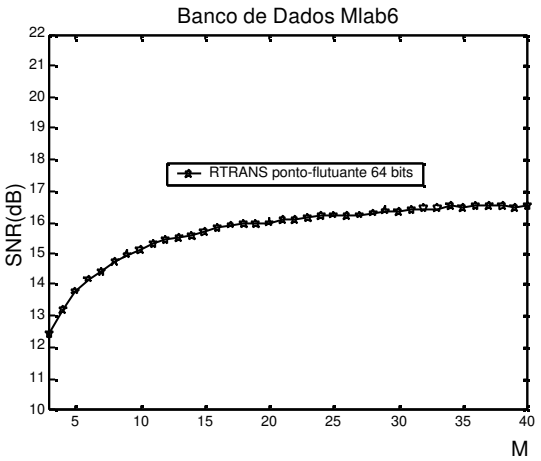


Figura 3.30. SNR x M.

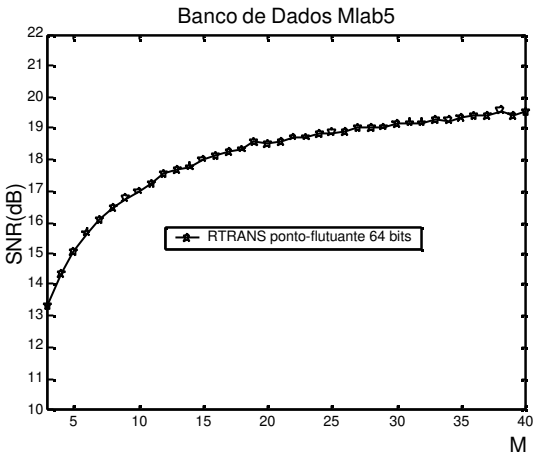


Figura 3.29. SNR x M.

Para se comparar as técnicas RITW e RTRANS [WL99], em termos da SNR total aplicadas a sinais de voz não submetidos ao codec G.722.1 são mostrados os valores máximos da SNR das respectivas técnicas na tabela 3.4. Como pode ser observado a técnica RTRANS apresenta resultados inferiores à técnica RITW, em termos de SNR. Um outro fato importante é que o ganho da técnica RTRANS, para valores $M > 40$, não mostrado nas figuras 3.27 a 3.32, é pouco significativo, oscilando entre 0,5 e 1,0 dB.

Tabela 3.4. Comparação dos valores máximos da SNR entre as técnicas RITW e RTRANS desconsiderando-se o codec G.722.1.

Banco de Dados	Máximo SNR para RITW	Máximo SNR para RTRANS
Mlab1	19.39, p/ Daub10	14,14 p/ M=38
Mlab2	19,21, p/ Daub3	18,88, p/ M=39
Mlab3	16,69, p/ Daub6	15,65, p/ M=40
Mlab4	20,07, p/ Daub3	20,68, p/ M=40
Mlab5	20,42, p/ Daub25	19,55, p/ M=38
Mlab6	16.87, p/ Daub6	16,53, p/ M=40

Para se analisar o desempenho da técnica RTRANS, quando o sinal de voz transformado é submetido ao codec G.722.1 e, posteriormente, reconstruído pela inversa da técnica RTRANS (condição 2), são mostradas nas figs. 3.33 a 3.38 as curvas de desempenho, em termos da SNR, versus M. Como pode ser notado nos gráficos, o desempenho da técnica RTRANS apresenta um nível de degradação maior à encontrada com a técnica RITW.

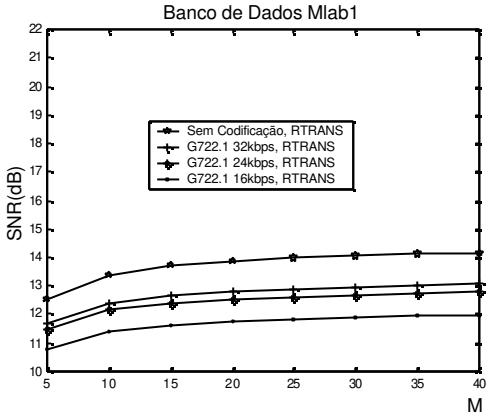


Figura 3.33. SNR x M.

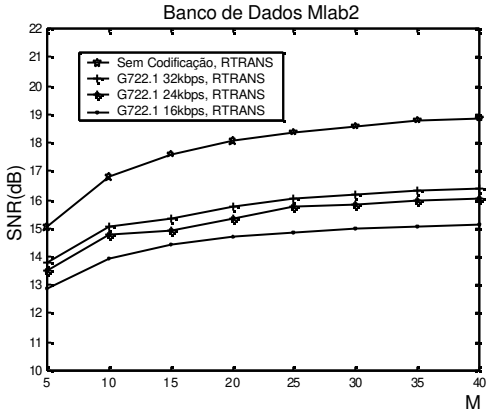


Figura 3.34. SNR x M.

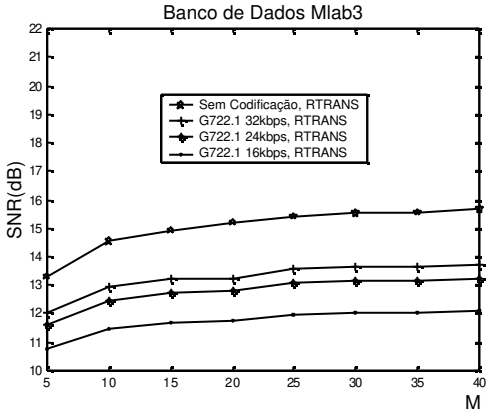


Figura 3.35. SNR x M.

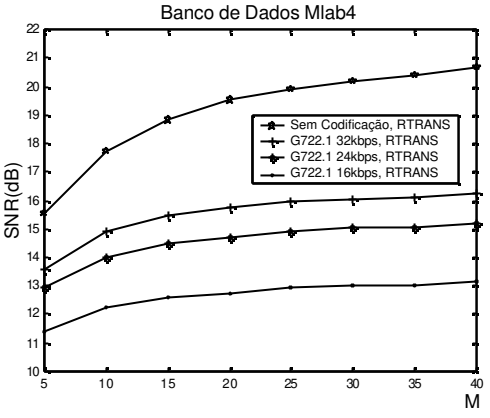


Figura 3.36. SNR x M.

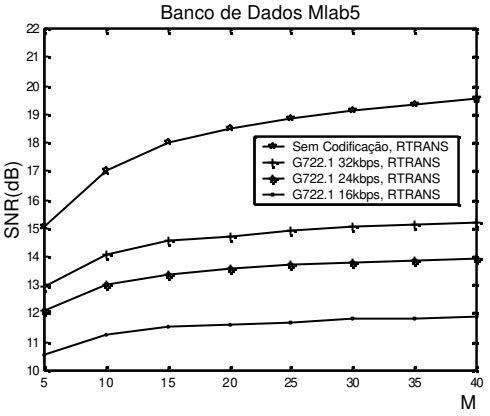


Figura 3.37. SNR x M.

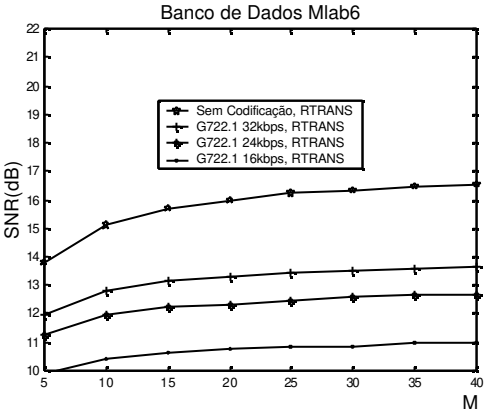


Figura 3.38. SNR x M.

As tabelas 3.5, 3.6 e 3.7 mostram o desempenho máximo, em termos da SNR, alcançado com a técnica RTRANS, da mesma forma como foi feito para a técnica RITW. Tal desempenho é alcançado para *M=40*.

Tabela 3.5. Comparação do SNR para taxa de 16kbps.

Banco de Dados	SNR do G.722.1 à 16kbps	SNR do G.722.1 à 16kbps com Perda + RTRANS
Mlab1	15,38	11,94
Mlab2	18,35	15,13
Mlab3	15,98	12,08
Mlab4	16,91	13,11
Mlab5	15,79	11,88
Mlab6	14,71	10,97

Tabela 3.6. Comparação do SNR para taxa de 24kbps.

Banco de Dados	SNR do G.722.1 à 24kbps	SNR do G.722.1 à 24kbps com Perda + RTRANS
Mlab1	16,03	12,77
Mlab2	19,42	16,02
Mlab3	17,51	13,21
Mlab4	18,45	15,21
Mlab5	18,20	13,92
Mlab6	17,19	12,65

Tabela 3.7. Comparação do SNR para taxa de 32kbps.

Banco de Dados	SNR do G.722.1 à 32kbps	SNR do G.722.1 à 32kbps Com Perda + RTRANS
Mlab1	16,29	13,04
Mlab2	19,84	16,36
Mlab3	18,03	13,72
Mlab4	18,83	16,24
Mlab5	18,92	15,21
Mlab6	18,08	13,63

A análise do desempenho da técnica MLP1 é mostrada com o objetivo de se verificar seu desempenho quando aplicada aos bancos de dados **Mlab1**, **Mlab2**, ...,

Mlab6. As considerações para análise e simulação são as mesmas empregadas para as técnicas RITW e RTRANS, ou seja, desempenho em termos da SNR e, neste caso, em função do número de neurônios presentes na camada escondida, conforme as figs. 3.39 a 3.44; operações em ponto flutuante e a 64 bits; número de entradas na MLPNN igual a 20 para a condição 1. As técnicas de interpolação baseadas em estruturas não-lineares, tais como a MLPNN apresentam desempenhos bastante variados e dependentes dos números de entradas, número de neurônios na camada intermediária e passo de adaptação, além de uma inicialização adequada aos parâmetros daquela. Sendo assim, os resultados apresentados podem ser superados a partir de uma análise minuciosa das diferentes redes neurais e de seus critérios de adaptação, formas de implementação e passo de adaptação (fixo e ajustável) e técnicas de pré e pós-processamento. Todavia os resultados apresentados a seguir são bastante ilustrativos, uma vez que nos leva a considerar que estruturas neurais mais bem elaboradas podem levar a melhores desempenhos. Apesar de terem sido realizadas simulações com até 100 neurônios na camada intermediária, apenas os resultados com até 50 neurônios são mostrados, pois as redes neurais não construtivas, tal como a MLP, apresentam resultados aleatórios para um grande número de neurônios.

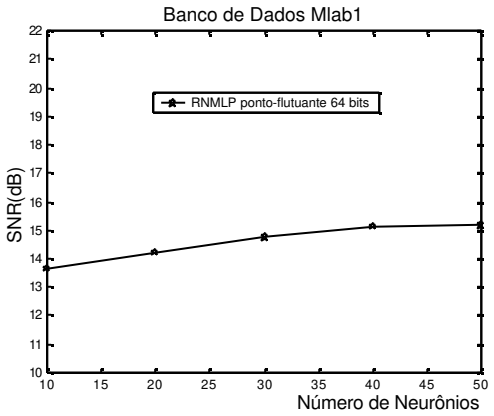


Figura 3.39. SNR x Nº de Neurônios.

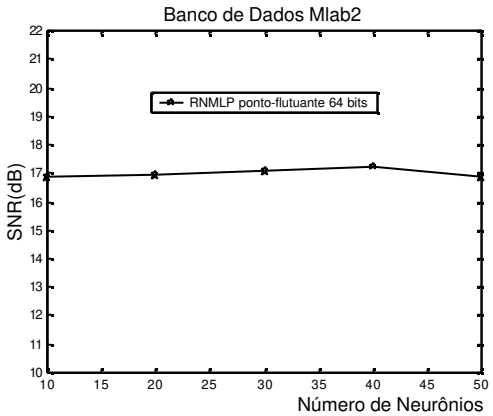


Figura 3.40. SNR x Nº de Neurônios.

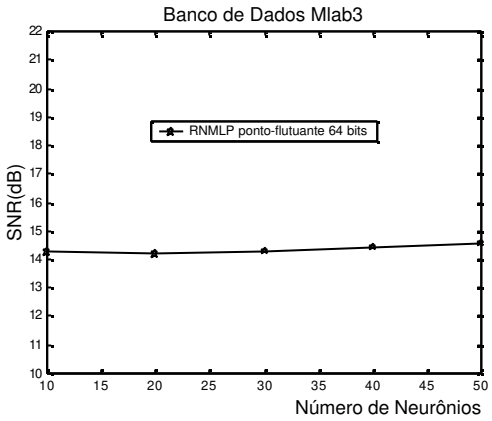


Figura 3.41. SNR x Nº de Neurônios.

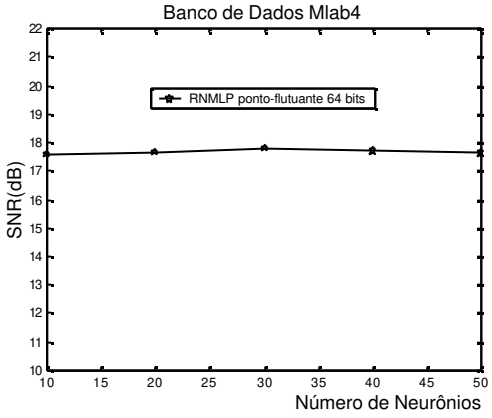


Figura 3.42. SNR x Nº de Neurônios.

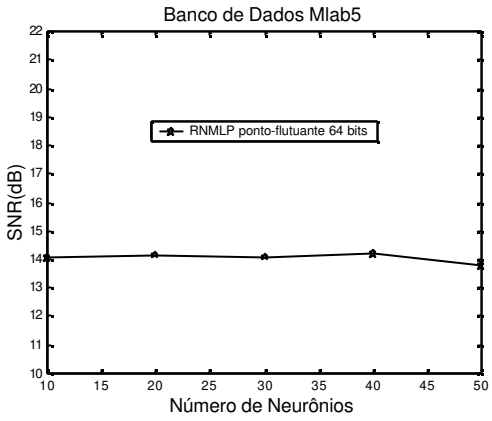


Figura 3.43. SNR x Nº de Neurônios.

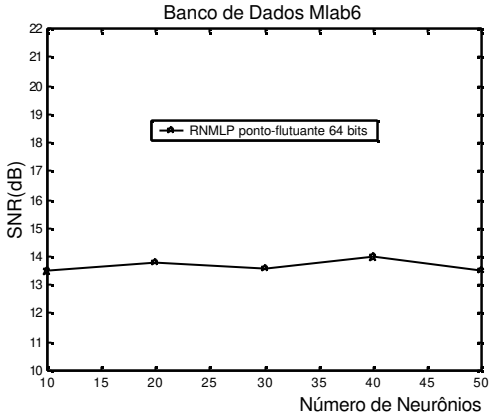


Figura 3.44. SNR x Nº de Neurônios.

As figs. 3.45 a 3.50 mostram o valor da SNR em função do número de neurônios para reconstrução das amostras ímpares com a técnica MLP1 de acordo com a condição 2.

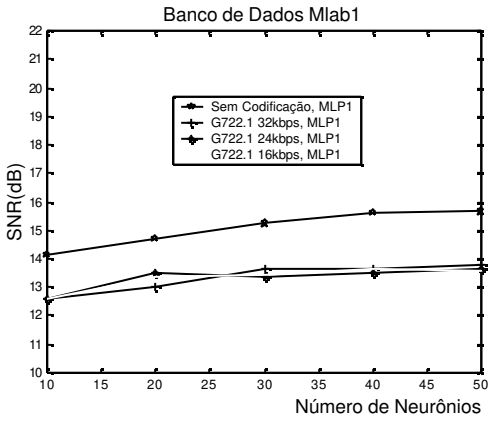


Figura 3.45. SNR x Nº de Neurônios.

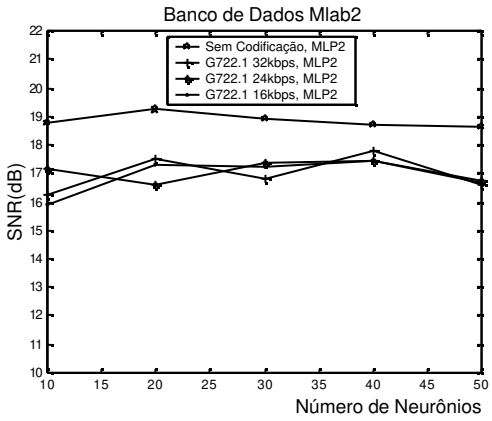


Figura 3.46. SNR x Nº de Neurônios.

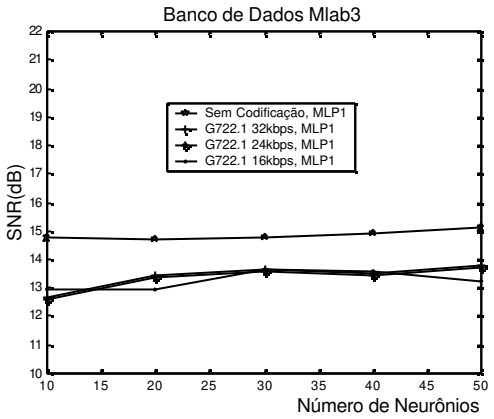


Figura 3.47. SNR x Nº de Neurônios.

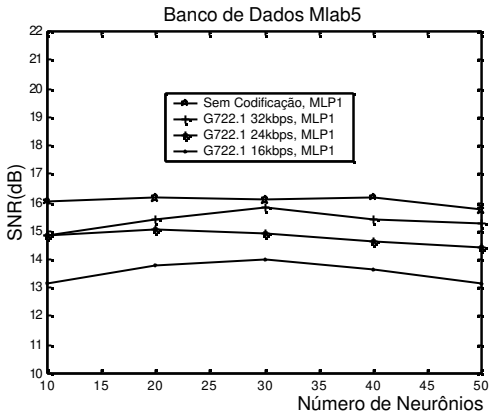


Figura 3.49. SNR x Nº de Neurônios.

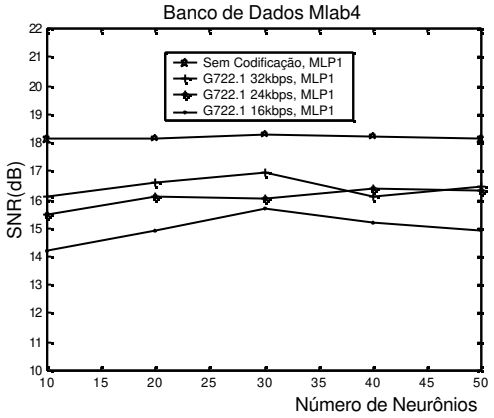


Figura 3.48. SNR x Nº de Neurônios.

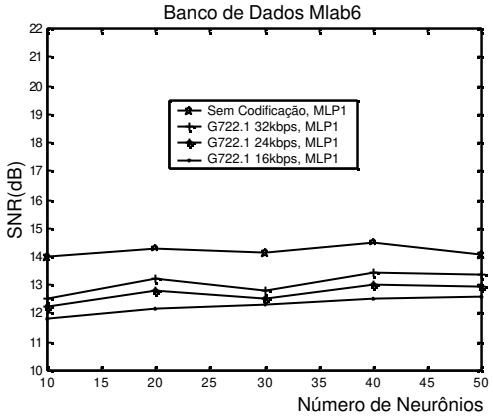


Figura 3.50. SNR x Nº de Neurônios.

As tabelas 3.8, 3.9, 3.10 apresentam os valores da SNR obtidos a partir das figs. 3.45 a 3.50, quando o número de neurônios é igual a 20. A escolha deste número de neurônios reflete a preocupação com a comprexidade computacional. Por exemplo, com 20 entradas e 20 neurônios estamos no $\boldsymbol{R}^{400}$, ou seja, o número de parâmetros utilizados para se interpolar o sinal de voz é extremamente alto e computacionalmente pesado. Estas considerações também se estendem aos resultados aferidos com a técnica MLP2.

Tabela 3.8. Comparação do SNR para taxa de 16kbps.

Banco de Dados	SNR do G.722.1 à 16kbps	SNR do G.722.1 à 16kbps com Perda + MLP1
Mlab1	15,38	13,39
Mlab2	18,35	16,48
Mlab3	15,98	12,91
Mlab4	16,91	14,90
Mlab5	15,79	13,81
Mlab6	14,71	12,14

Tabela 3.9. Comparação do SNR para taxa de 24kbps.

Banco de Dados	SNR do G.722.1 à 24kbps	SNR do G.722.1 à 24kbps com Perda + MLP1
Mlab1	16,03	13,50
Mlab2	19,42	16,62
Mlab3	17,51	13,36
Mlab4	18,45	16,12
Mlab5	18,20	15,05
Mlab6	17,19	12,81

Tabela 3.10. Comparação do SNR para taxa de 32kbps.

Banco de Dados	SNR do G.722.1 à 32kbps	SNR do G.722.1 à 32kbps com Perda + MLP1
Mlab1	16,29	12,99
Mlab2	19,84	16,67
Mlab3	18,03	13,41
Mlab4	18,83	16,62
Mlab5	18,92	15,41
Mlab6	18,08	13,20

As figs. 3.51 a 3.56 mostram o valor da SNR em função do número de neurônios para reconstrução das amostras ímpares dos bancos de dados**,** a partir das amostras pares submetidas e não submetidas ao codec G.722.1 , condição 1, e reconstrução das amostras ímpares com a técnica MLP2.

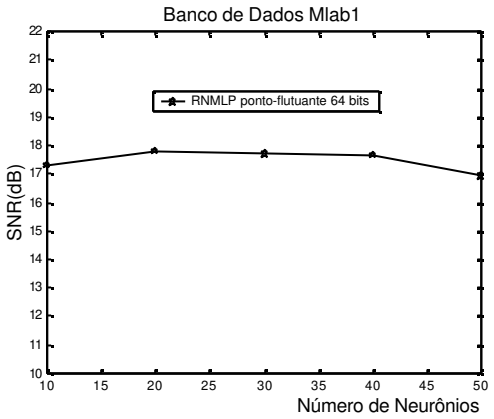


Figura 3.51. SNR x Nº de Neurônios.

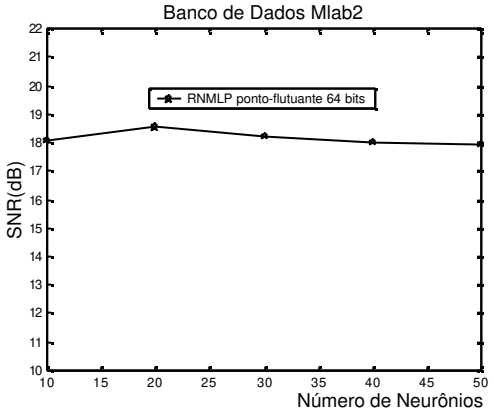


Figura 3.52. SNR x Nº de Neurônios.

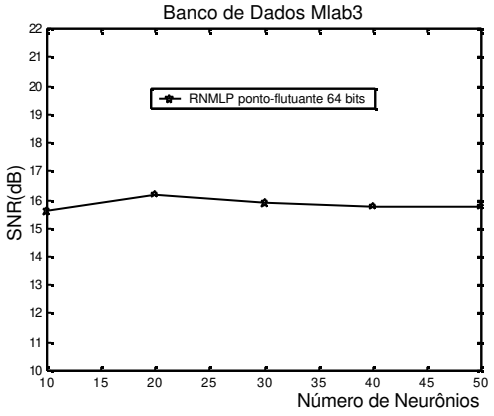


Figura 3.53. SNR x Nº de Neurônios.

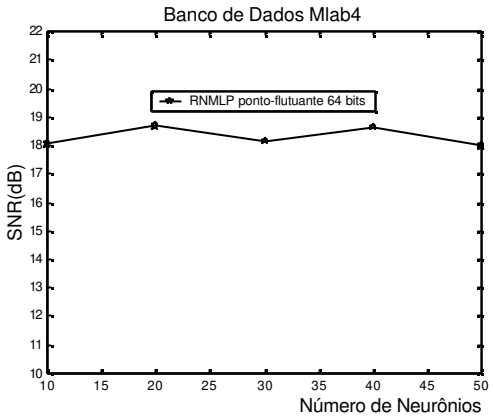


Figura 3.54. SNR x Nº de Neurônios.

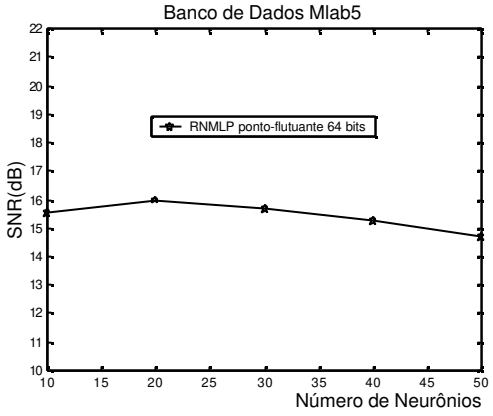


Figura 3.55. SNR x Nº de Neurônios.

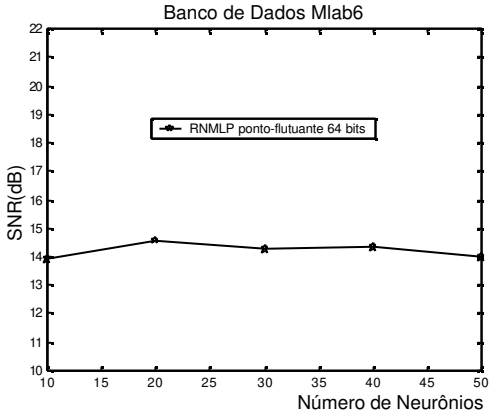


Figura 3.56. SNR x Nº de Neurônios.

As figs. 3.57 a 3.62 mostram o valor da SNR em função do número de neurônios para reconstrução das amostras ímpares dos bancos de dados para condição 2 e reconstrução das amostras ímpares com a técnica MLP2.

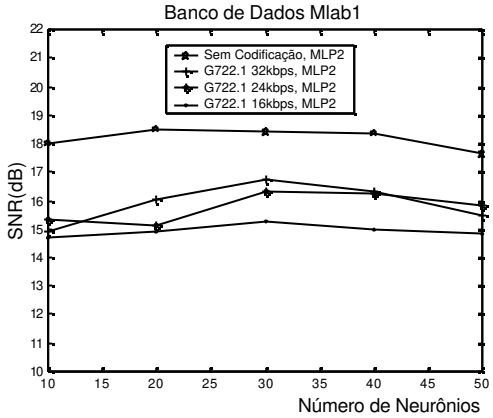


Figura 3.57. SNR x Nº de Neurônios.

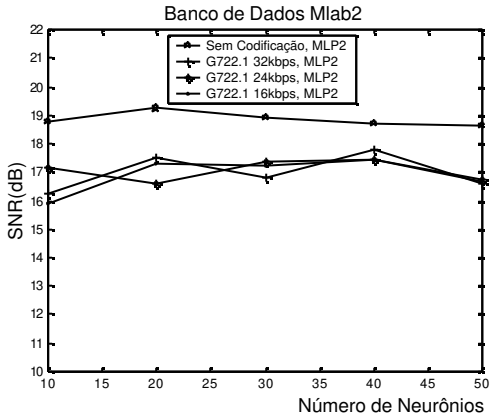


Figura 3.58. SNR x Nº de Neurônios.

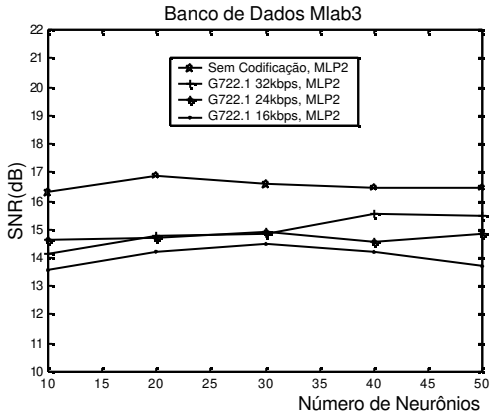


Figura 3.59. SNR x Nº de Neurônios.

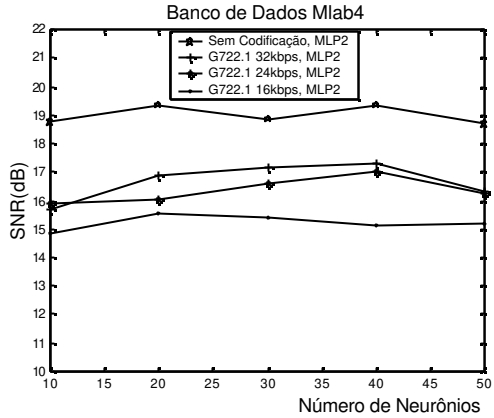


Figura 3.60. SNR x Nº de Neurônios.

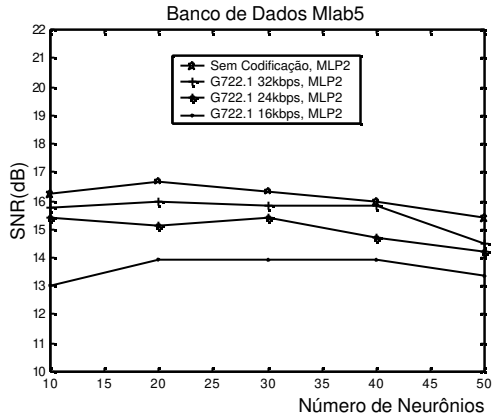


Figura 3.61. SNR x Nº de Neurônios.

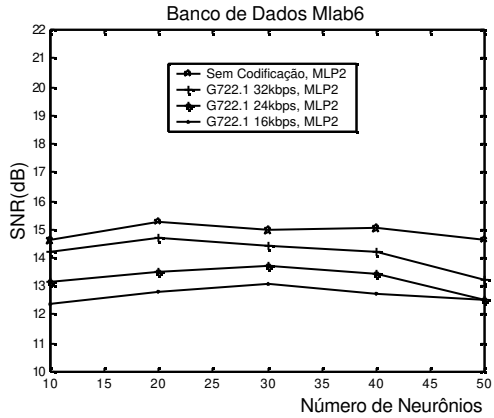


Figura 3.62. SNR x Nº de Neurônios.

As tabelas 3.11, 3.12, 3.13 apresentam os valores da SNR observados em sinais de voz, com as características e situação especificada na condição 2, utilizando-se a técnica MLP2 para reconstrução das amostras ímpares.

Tabela 3.11. Comparação do SNR para taxa de 16kbps.

Banco de Dados	SNR do G.722.1 à 16kbps	SNR do G.722.1 à 16kbps com Perda + MLP2
Mlab1	15,38	14,89
Mlab2	18,35	17,33
Mlab3	15,98	14,23
Mlab4	16,91	15,57
Mlab5	15,79	13,93
Mlab6	14,71	12,80

Tabela 3.12. Comparação do SNR para taxa de 24kbps.

Banco de Dados	SNR do G.722.1 à 24kbps	SNR do G.722.1 à 24kbps com Perda + MLP2
Mlab1	16,03	15,12
Mlab2	19,42	16,58
Mlab3	17,51	14,69
Mlab4	18,45	16,03
Mlab5	18,20	15,11
Mlab6	17,19	13,50

Tabela 3.13. Comparação do SNR para taxa de 32kbps.

Banco de Dados	SNR do G.722.1 à 32kbps	SNR do G.722.1 à 32kbps com Perda + MLP2
Mlab1	16,29	16,02
Mlab2	19,84	17,50
Mlab3	18,03	14,74
Mlab4	18,83	16,85
Mlab5	18,92	15,96
Mlab6	18,08	14,72

As simulações das técnicas MLP1 e MLP2 em ponto fixo e a 16 bits não foram consideradas neste trabalho. Entretanto, é um dos estudos a serem conduzidos.

A fim de estabelecer uma comparação entre as técnicas RITW, RTRANS, MLP1 e MLP2, as tabelas 3.14, 3.15 e 3.16 reúnem os valores de SNR total para taxas de 16kbps, 24kbps e 32kbps, respectivamente. Os valores de SNR referem-se aos sinais de voz submetidos a condição 1 e à reconstrução a partir das técnicas propostas (RITW, MLP1 e MLP2) em relação a técnica RTRANS. Observa-se que a técnica RTRANS tem pior desempenho, em termos de SNR, enquanto que as técnicas baseadas em algoritmos não lineares, MLPNN, permitem resultados melhores. No entanto, a técnica que apresenta melhor desempenho é a RITW. A técnica RITW, além de apresentar o melhor desempenho, possui o menor custo computacional, visto que a mesma é realizada através de filtros FIR.

Tabela 3.14. Comparação de SNR para a taxa de 16kbps.

Banco de Dados	G.722.1	RTRANS	MLP1	MLP2	RITW
Mlab1	15,38	11,94	13,39	14,89	16,82
Mlab2	18,35	15,13	16,48	17,33	18,10
Mlab3	15,98	12,08	12,91	14,23	15,50
Mlab4	16,91	13,11	14,90	15,57	17,85
Mlab5	15,79	11,88	13,81	13,93	16,85
Mlab6	14,71	10,97	12,14	12,80	14,56

Tabela 3.15. Comparação de SNR para a taxa de 24kbps.

Banco de Dados	G.722.1	RTRANS	MLP1	MLP2	RITW
Mlab1	16,03	12,77	13,50	15,12	17,85
Mlab2	19,42	16,02	16,62	16,58	18,36
Mlab3	17,51	13,21	13,36	14,69	15,81
Mlab4	18,45	15,21	16,12	16,03	18,61
Mlab5	18,20	13,92	15,05	15,11	17,91
Mlab6	17,19	12,65	12,81	13,50	15,32

Tabela 3.16. Comparação de SNR para a taxa de 32kbps.

Banco de Dados	G.722.1	RTRANS	MLP1	MLP2	RITW
Mlab1	16,29	13,04	12,99	16,02	18,26
Mlab2	19,84	16,36	16,67	17,50	18,46
Mlab3	18,03	13,72	13,41	14,74	16,01
Mlab4	18,83	16,24	16,62	16,85	18,99
Mlab5	18,92	15,21	15,41	15,96	18,47
Mlab6	18,08	13,63	13,20	14,72	15,63

Os resultados a partir da avaliação subjetiva realizada com seis pessoas (três mulheres e três homens) são reunidos na tabela 3.17, 318 e 3.19. Nestas tabelas são reunidos um nível subjeitvo de qualidade entre as diferentes técnicas abordas durante as simulações. Os testes foram realizados com as seguintes considerações:

- Arquivos de voz submetidos ao codec G.722.1.
- Arquivos de voz embaralhado, $SIF = 2$,submetidos ao codec G.722.1, perdas das amostras ímpares e reconstrução das mesmas a partir da técnica RTRANS (padrão) e propostas (RITW, MLP1 e MLP2), condição 2.
- As seis pessoas escutam os sinais de voz especificados anteriormente e os sinais originais de voz .
- Cada uma das pessoas concedeu uma das seguintes notas:

5 = MUITO BOM

4 = BOM

3 = RAZOÁVEL

2 = RUIM

1 = MUITO RUIM

0 = INTELIGÍVEL

Estas notas foram dada para a qualidade do sinal de voz em relação ao sinal original. Os resultados apresentados nas tabelas 3.17, 3.18 e 3.19 representam a média obitida com as análises realizadas pelos três homens e três mulheres.

A utilização análise subjetiva do sinal de voz com algumas pessoas é devido ao fato de que o valor do SNR total serve apenas como um indicativo da qualidade do sinal de voz,

além de ser a qualidade do sinal de voz algo extremamente subjetivo e dependente do tipo de pessoas escolhidas para realizar as análises. Neste contexto, pode-se dizer a análise para um correta validação das mesmas deve ser realizada com um número maior de pessoas ou por algum método objetivo baseado na análise subjetiva [BAR01].

Tabela 3.17. Análise subjetiva para a taxa de 16kbps.

Banco de Dados	G.722.1	RTRANS	MLP1	MLP2	RITW
Mlab1	4,3	2.3	2.3	2.7	**2.9**
Mlab2	4.1	2.7	2.4	2.6	**2.9**
Mlab3	4.0	2.5	2.4	2.6	**3.1**
Mlab4	4.1	2.8	3.1	3.1	**3.2**
Mlab5	4.4	2.8	3.0	3.0	**3.0**
Mlab6	4.1	2.7	3.1	3.1	**3.1**

Tabela 3.18. Análise subjetiva para a taxa de 24 kbps.

Banco de Dados	G.722.1	RTRANS	MLP1	MLP2	RITW
Mlab1	4,5	2.5	2.5	2.8	**3.0**
Mlab2	4.6	2.6	2.4	2.6	**3.1**
Mlab3	4.6	2.7	2.3	2.9	**3.3**
Mlab4	4.8	2.9	3.0	3.2	**3.2**
Mlab5	4.5	3.1	2.9	3.5	**3.5**
Mlab6	4.7	3.0	3.1	3.3	**3.4**

Tabela 3.19. Análise subjetiva para a taxa de 32kbps.

Banco de Dados	G.722.1	RTRANS	MLP1	MLP2	RITW
Mlab1	4.8	2.8	2.3	2.7	**3.1**
Mlab2	4.8	2.7	2.4	2.6	**3.4**
Mlab3	4.6	2.6	2.4	2.9	**3.1**
Mlab4	4.9	3.2	3.1	3.3	**3.3**
Mlab5	5	3.1	3.0	3.4	**3.5**
Mlab6	4.8	3.0	3.1	3.4	**3.3**

Algumas conclusões podem ser aferidas das tabelas 3.17 a 3.19

- Os resultados aferidos são melhores quando a técnica de reconstrução RITW é utilizada.
- A técnica RTRANS tem desempenho compatível com a técnica MLP1.
- A técnica MLP2 apresenta desempenho bem próximo da técnica RITW.

3.4 Sumário

Neste capítulo apresentamos três novas técnicas aplicadas a sinais de voz submetidos à codificação por forma de onda. Além disso, realizamos diversas simulações para verificar o desempenho das proposições frente a técnica mais atual encontrada na literatura.

Conforme pôde ser verificado, as técnicas propostas apresentam resultados superiores e, em especial, a técnica RITW apresenta os melhores resultados.

Como as técnicas RITW e MLP2 fazem uso de algoritmos de classificação de segmentos de voz, o capítulo 4 comenta algumas das principais técnicas de classificação de segmentos de voz e apresenta a proposta de um classificador neural com baixo custo computacional, em outras palavras, com poucos neurônios.

A fim de aferir os resultados obtidos em simulação, o capítulo 5 descreve a implementação de um protótipo de telefonia IP com a técnica RITW, utilizando-se placas DSP e um PC.

4 Técnicas de classificação de sinais de voz

A necessidade de reconhecer se um segmento de voz é surdo, sonoro ou silêncio tem duas implicações importantes no presente trabalho. Primeiramente porque os algoritmos de detecção e supressão de silêncio são essenciais em aplicações VoIP, promovendo significativa redução do uso de largura de banda, uma vez que em média 40 a 50% de uma conversação telefônica é silêncio [DMV78]. Por outro lado, a classificação do sinal de voz em surdo ou sonoro permite realizar a reconstrução dos pacotes perdidos, a partir de algoritmos que exploram tais características.

Visto que neste capítulo estaremos propondo um classificador neural para segmentos de voz, iremos comentar na seção 4.1 alguns parâmetros e técnicas utilizadas na classificação de segmentos de voz, na seção 4.2 apresentaremos a proposição de um novo classificador neural e, finalmente, a seção 4.3 faz um resumo sobre o capítulo.

4.1 Alguns parâmetros e técnicas clássicas aplicadas à classificação de segmentos de voz

Existem diferentes metodologias empregadas para classificar os segmentos de voz. Dentre as diversas técnicas disponíveis na literatura, pode-se ressaltar as que são baseadas nos seguintes parâmetros:

- **Número de cruzamentos por zeros**, N_z, no quadro de sinal de voz considerado. Esta informação indica a concentração de energia no espectro do sinal.
- **Log da energia do sinal**, E_s: definido pela equação 4.1.

$$E_s = 10\log_{10}\left(\in + \frac{1}{N}\sum_{n=1}^{N} s^2(n)\right) \tag{4.1}$$

Sendo $s(n)$ as amostras do segmento de voz e $\in << \frac{1}{N}\sum_{n=1}^{N} s^2(n)$ uma constante positiva e próxima de zero para evitar que o cálculo do $\log_{10}$ dê zero. Um valor aceitável

para $\in$ é 10^{-5}. Este parâmetro depende de vários fatores, tais como sensibilidade do microfone e dos amplificadores e da resolução do conversor A/D.

- **Coeficiente de auto-correlação normalizado de atraso unitário**, C_1, definido pela equação 4.2, determina a correlação entre amostras adjacentes no sinal de voz.

$$C_1 = \frac{\sum_{n=1}^{N} s(n)s(n-1)}{\sqrt{\left(\sum_{n=1}^{N} s^2(n)\right)\left(\sum_{n=0}^{N-1} s^2(n)\right)}} \tag{4.2}$$

- **Coeficientes do filtro de predição**, de um filtro LPC (*Linear Predictive Coding*) para análise baseada no método da covariância.
- **Erro de predição normalizado,** E_p, expressado em decibéis e definido pelas equações 4.3 e 4.4.

$$E_p = E_s - 10\log_{10}\left(10^{-6} + \left|\sum_{k=1}^{p} \alpha_k \phi(0,k) + \phi(0,0)\right|\right) \tag{4.3}$$

$$\phi(i,k) = \frac{1}{N}\sum_{n=1}^{N} s(n-i)s(n-k) \tag{4.4}$$

Sendo $\phi(i,k)$ o termo na *i*-ésima linha e *k*-ésima coluna da matriz de covariância do quadro de sinal e α_k o *k*-ésimo coeficiente do filtro preditor.

Os coeficientes do filtro preditor são obtidos pela minimização da equação 4.5.

$$E = \frac{1}{N}\sum_{n=1}^{N}\left[s(n) + \sum_{k=1}^{p} \alpha_k s(n-k)\right]^2 \tag{4.5}$$

Em [AR76] foi proposto um algoritmo de decisão, baseado em estatísticas clássicas, para classificar o segmento de voz a partir dos parâmetros acima citados. Esta proposição, tida como referência na literatura, apresenta bons resultados. No entanto, dois inconvenientes são identificados: custo computacional para implementação em tempo real e a necessidade de atualização das informações estatísticas para locutores diferentes. Este último quando não realizado de maneira correta degrada consideravelmente o desempenho da técnica. Por outro lado, técnicas baseadas em algoritmos não lineares tem possibilitado resultados interessantes sem a necessidade de informações estatísticas dos sinais de voz. Por exemplo, em [HSB00] é proposto um algoritmo para detectar o início e o fim de segmentos de voz, baseado na MLPNN com uma camada intermediária, 250 neurônios, 15 entradas e treinada com algoritmo *Backpropagation* (*Backpropagation algorithm* – BA) [HAY99]; em [WLIN01] é apresentado uma rede neuro-fuzzy recorrente para detecção de

sinais de voz na presença de ruído de ambiente com potência variável e em [QH93] é apresentado uma estrutura baseada na rede neural multicamada com realimentação (NNMFN) com uma camada escondida, 15 entradas, 15 neurônios (no mínimo) e 3 saídas para classificação de segmentos de voz.

A característica marcante dos classificadores neurais comentados acima é a complexidade computacional envolvida no treinamento e atualização das redes neurais implementadas, pois tais estruturas fazem uso de um grande número de neurônios.

4.2 Proposta de um Classificador Neural aplicado à classificação de segmentos de sinal de voz com baixo custo computacional

A figura 4.1 apresenta uma proposta para classificação de segmentos de voz. Tal proposta faz uso de apenas três parâmetros ou entradas na MLPNN: valor normalizado da energia, valor normalizado do número de cruzamento por zero e o valor absoluto do coeficiente de auto-correlação como entradas de uma MLPNN. A MLPNN, também implementada em [JRRLF01], tem apenas uma saída dada por *y*. Esta saída, durante e após o treinamento deve responder de acordo com a regra de decisão mostrada na tabela 4.1.

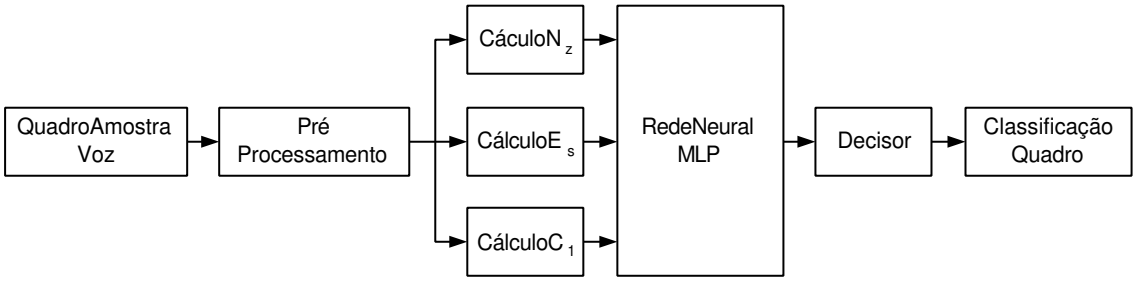


Figura 4.1. Diagrama de bloco do Classificador Neural proposto para a classificação de segmentos de sinais de voz em V-U-S (*voice-unvoice-silence*).

Para que a proposta apresentada na figura 4.1 apresente bons resultados é necessário que a MLPNN seja submetida a um treinamento exaustivo, a fim de que a sua capacidade de generalização seja melhor explorada.

Tabela 4.1. Algoritmo de decisão na saída da rede.

```
Se y < 0
    O quadro é silêncio ou surdo.
    Se Es <= max(Es de silêncio) e |C1| >= 0.50
       O quadro é silêncio.
    Senão
       O quadro é surdo.
Senão
    O quadro é sonoro.
```

Os blocos do classificador neural mostrados na figura 4.1 realizam as seguintes tarefas:

- **QuadroAmostraVoz**: Armazena a seqüência de amostras de sinal de voz, dado um comprimento qualquer.
- **Pré-Processamento**: filtra o segmento de voz para extração do sinal DC presente. O filtro implementado é de acordo com a equação (4.6) [AR76].

$$H(z) = \frac{Y(z)}{X(z)} = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^{M-1} b_k z^{-k}}{\sum_{k=0}^{N-1} a_k z^{-k}} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{a_0 + a_1 z^{-1} + a_2 z^{-2}} = \\ = \frac{1 - 2z^{-1} + z^{-2}}{1 - 2e^{-aT}\cos(bT) z^{-1} + e^{-2aT} z^{-2}} \tag{4.6}$$

Sendo

$a = 130 * 2\pi$

$b = 200 * 2\pi$

$T = \frac{10^{-4}}{1.6}$

$M = N = 3$

Substituindo-se *a*, *b* e *T* em 4.6, tem-se:

$$H(z) = \frac{1 - 2z^{-1} + z^{-2}}{1 - 1{,}89460215607491\, z^{-1} + 0{,}90293766293587\, z^{-2}} \tag{4.7}$$

Como a equação (4.7) corresponde a um filtro com coeficientes em ponto flutuante, uma adequação dos mesmos para ponto fixo à 16 bits é necessária para aplicação em tempo real em DSP e com baixo custo computacional.

- **CálculoN$_z$:** calcula e normaliza o número de cruzamentos por zeros do segmento de voz. O número de cruzamentos é normalizado pelo valor máximo de cruzamentos.

Por exemplo, para segmentos com 160 amostras ou 10ms e $f_s = 16kHz$, o valor máximo de cruzamentos estimado é 145.

- **CálculoE$_s$:** calcula a energia do segmento de voz de acordo com a equação (4.1). De forma idêntica ao item anterior, o valor da energia, em dB, também é normalizada pelo valor máximo. Para quadros de 10ms de voz amostrada a 16kHz, o valor máximo de energia estimado, em dB, é 81.
- **CálculoC$_1$:** calcula o valor absoluto o coeficiente de correlação do segmento de voz de acordo com a equação (4.2).
- **RedeNeuralMLP**: implementa a MLPNN, a qual possui uma camada intermediária, função tangente hiperbólica como função de ativação entre a camada de entrada e a camada intermediária, função identidade com a função de ativação entre a camada intermediária e a saída e os parâmetros (pesos) da rede neural são treinados e atualizados pelo algoritmo Backpropagation [HOR89].
- **Decisor**: executa a regra esboçada na tabela 4.1 para decidir se o segmento corresponde a silêncio, a uma voz sonora ou surda.
- **ClassificaçãoQuadro**: armazena a classificação do segmento de voz.

A utilização do método proposto neste item mostrou-se interessante, na medida que tornou possível a classificação de segmentos de voz por meio de apenas três parâmetros ou entradas na MLPNN.

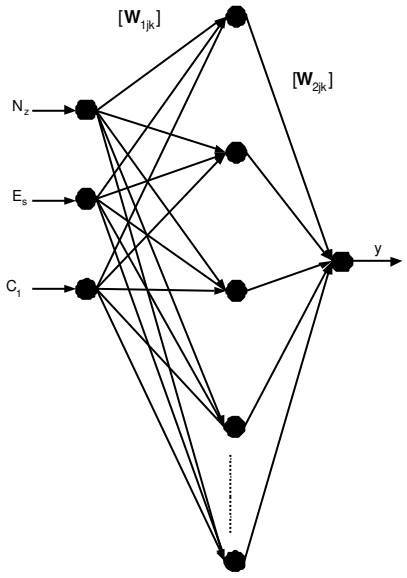


Figura 4.3. MLPNN implementada.

Para exemplificar a exatidão da classificação dos sinais de voz a figura 4.4 apresenta a taxa de classificação do classificador neural proposto em função do número de neurônios, para dados de treinamento e de teste. Nestas simulações foram utilizados os arquivos **Mlab1**, **Mlab2** e **Mlab6** para treinamento, cerca de 3200 conjuntos de dados, e os arquivos **Mlab3**, **Mlab4** e **Mlab5** para teste, cerca de 3200 conjuntos de dados. Os dados de treinamento e teste foram obtidos com o algoritmo proposto em [AR76] e refinado por uma análise visual. Os arquivos **Mlab1**, **Mlab2**,...,**Mlab6** do MIT Media Laboratory [MIT01], são os mesmos utilizados na seção 3.3. O passo de adaptação adotado para treinamento da rede é $\mu = 0{,}1$. Como pode-se observar a taxa de classificação correta obtida em condições de teste é de 95,5%.

A MLPNN foi escolhida devido a sua grande facilidade de implementação e dedução. No entanto, a possibilidade de utilização da outras redes neurais, tais como RBFNN (*Radial Basis Function Neural Network*), mapas auto-organizáveis de Kohonen e redes neural de segunda ordem são soluções que podem ser utilizadas no lugar da MLPNN treinada com algoritmo de primeira ordem (gradiente estocástico). Além disso, a regra de decisão pode ser substituída por uma regra *fuzzy*, possibilitando, desta maneira, uma decisão mais subjetiva e, conseqüentemente, melhor desempenho.

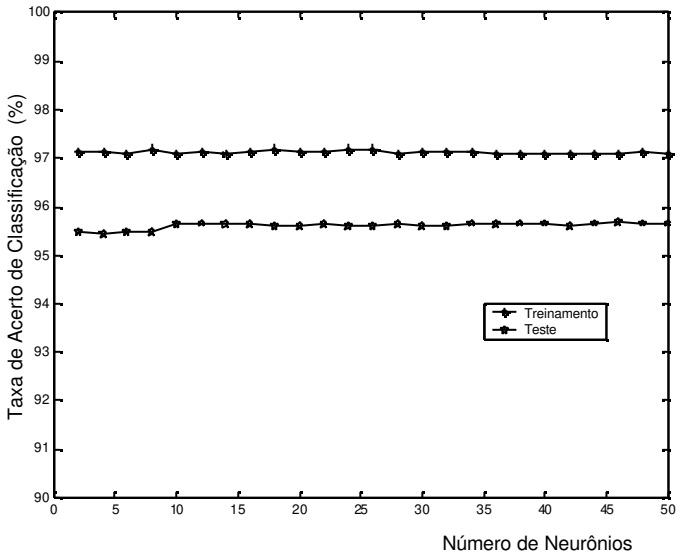


Figura 4.4. Taxa de Classificação em função do número de neurônios.

Finalmente, embora os parâmetros da MLPNN tenham valores que possibilitem a sua implementação em 16 bits, tal solução não foi considerada, pois a implementação em ponto-flutuante a 32 bits e em tempo real no DSP apresentou os resultados satisfatórios. No entanto, estudos estão sendo conduzidos com este propósito.

4.3 Sumário

Neste capítulo fizemos de maneira bastante resumida alguns comentários importantes sobre as técnicas de classificação de segmentos de voz. Além disso, propusemos um classificador neural com baixo custo computacional e bom resultado. Tal proposição é bastante pertinente, uma vez que as técnicas de reconstrução propostas no capítulo anterior, RITW e MLP2, fazem uso dessas informações para obterem melhor desempenho. Além disso, uma diminuição considerável da largura de banda utilizada é obtida pela não transmissão de silêncio.

A implementação do classificador proposto num protótipo para telefonia IP é descrita no capítulo a seguir.

5 Implementação do protótipo para telefonia IP

A implementação do protótipo para telefonia IP foi realizada de forma a se verificar os resultados obtidos através de simulação. Pode-se dividir esta implementação em duas partes:

- Implementação dos algoritmos de processamento de sinais de voz.
- Implementação dos algoritmos de interfaceamento entre o DSP e o PC.

De forma a situar a relevância desta implementação no contexto da tese, o presente capítulo é organizado em três itens distintos. Primeiramente na seção 5.1 um estudo cuja contribuição está em ordenar, de forma sistemática e nem sempre disponível na literatura, os principais critérios a serem levados em conta na escolha de um processador. Tais critérios foram considerados na presente implementação, de forma que o hardware escolhido é apresentado na seção 5.2. Finalmente a seção 5.3 descreve a solução como um todo, detalhando as diversas funções implementadas. Por fim, faz-se um sumário do capítulo.

5.1 Critérios para escolha de um processador DSP

De modo geral, a escolha de um processador depende principalmente da aplicação considerada. Assim sendo, uma série de itens devem ser pesquisados e analisados para facilitar a especificação e seleção de um dado processador DSP. Dentre os principais itens podemos ressaltar o formato aritmético do DSP, a largura do barramento, a velocidade de processamento, a organização da memória, a facilidade de uso, suporte para multiprocessadores, gerenciamento e consumo de energia e, finalmente, o item mais importante, o custo.

5.1.1 Formato Aritmético dos processadores DSP

Uma das características fundamentais dos processadores DSP é o tipo de aritmética usada. Por exemplo, a maioria faz uso de aritmética em ponto fixo, onde os números são representados por inteiros correspondendo ao intervalo entre –1.0 e +1.0. Por outro lado,

alguns processadores DSP fazem uso da aritmética em ponto flutuante, onde os valores são representados por uma *mantissa* e por um expoente ($mantissa \; x \; 2^{expoente}$), sendo a mantissa uma valor fracionário no intervalo entre –1.0 e +1.0, enquanto o expoente é um inteiro que especifica o lugar do ponto binário, análogo ao ponto decimal nos números em base 10.

A aritmética em ponto flutuante é um mecanismo mais flexível e geral do que a aritmética em ponto fixo, pois os sistemas projetados pela primeira têm acesso a uma grande faixa dinâmica de valores. Desta forma, os processadores DSP em ponto flutuante são geralmente mais fáceis de programar do que os processadores em ponto fixo. No entanto, apresentam um custo e um consumo de energia maior devido à complexidade de hardware necessária à sua implementação. Para os processadores em ponto fixo, a correta adequação numérica dos sinais nos diferentes estágios de processamento se faz fundamental para não gerar valores fora da faixa dinâmica limitada. Por exemplo, considerando um processador de 16 bits, a faixa dinâmica é de (-32768 até 32767).

A maioria da aplicações DSP em larga escala de produção faz uso dos processadores em ponto fixo, pois a prioridade é a minimização dos custos e do consumo de energia. Para se alcançar esse objetivo, os programadores de códigos determinam as faixas dinâmicas e a precisão necessária das aplicações através de simulações e realizando os ajustes necessários. Já os processadores em ponto flutuante são recomendados para aplicações DSP sensíveis à precisão, com maior demanda de faixa dinâmica ou com a prioridade residindo na facilidade de desenvolvimento, em detrimento do custo.

Os processadores DSP em ponto fixo possibilitam o uso da aritmética em ponto flutuante através de rotinas que emulam o comportamento de dispositivos em ponto flutuante, mas com custo computacional extremamente elevado.

5.1.2 Tamanho das palavras de dados

Todos os processadores DSP em ponto flutuante usam palavras de dados com 32 bits. Como exemplo pode-se ressaltar o processador TMS320C67x da Texas Instruments. Para processadores DSP em ponto fixo, o tamanho mais comum é de 16 bits, que é o caso do TMS320C62x e do TMS320C64x, da Texas Instruments. Há também os processadores

DSP com 20 ou 24 bits, como os da família ZR3800x da Zoran e da família DSP563xx da Motorola, respectivamente.

O tamanho das palavras tem um impacto maior no custo dos processadores DSP, visto que este influencia diretamente as dimensões do *chip*, o número de pinos necessários no *chip* e o tamanho dos dispositivos de memória externa conectados aos processadores DSP. Com processadores DSP de ponto fixo de 16 bits, é possível realizar operações a 32 bits, a partir de uma escolha criteriosa de algumas instruções, todavia com maior consumo de ciclos de *clock*.

Para aplicações DSP onde predominam operações aritméticas de baixa precisão, a sugestão é escolher um processador de 16 ou 20 bits. O uso de rotinas para realizar operações em 32 bits poderá ser considerado se alguma parte do programa necessitar de uma precisão maior.

É importante destacar, entretanto, que nem sempre os tamanhos das palavras de dados e de instrução são iguais. Por exemplo, os processadores DSP da família DSP ADSP-21xx, da Analog Devices, usam tamanho de 16 bits para dados e 24 bits para instruções.

5.1.3 Velocidade de processamento

Uma medida chave da adequabilidade de um processador DSP a uma aplicação particular é a velocidade de execução. Desta forma, um parâmetro fundamental para realizar esta análise é o tempo de ciclo de instrução do processador, definido como a quantidade de tempo necessária à execução das instruções rápidas.

O recíproco do tempo de ciclo de instrução dividido por um milhão e multiplicado pelo número de instruções executadas por ciclo é chamado de *processor's peak instruction execution rate* e é dado em milhões de instruções por segundo (*million instructions per second* - MIPS).

Um problema encontrado nesse parâmetro de análise de desempenho é que o custo computacional para se executar uma instrução rápida varia enormemente entre os vários tipos de processadores disponíveis. Por exemplo, os processadores DSP baseados na arquitetura VLIW (*Very Long Instruction Width*) executam múltiplas instruções por ciclo. Desta forma, a comparação de desempenho entre os processares convencionais e os baseados na arquitetura VLIW apresentam resultados fora da realidade, já que os mesmos

têm arquiteturas completamente diferentes. Até mesmo quando a comparação é entre processadores DSP convencionais, as taxas de MIPS apresentam resultados que não refletem a realidade. Dentre os problemas encontrados, os quais dificultam a análise do desempenho baseada em MIPS, temos os seguintes:

- Alguns processadores DSP possibilitam o deslocamento múltiplo de bits em uma única instrução, enquanto outros processadores executam esta tarefa utilizando um ciclo de instrução para cada bit deslocado.
- Alguns processadores DSP executam o deslocamento de dados em paralelo, ou seja, buscam os operandos e executam as instruções simultaneamente. Esta característica não é relacionada às instruções que a ALU (*Arithmetic Logic Unit*) processa. Todavia, alguns processadores fornecem o deslocamento de dados relacionado aos operandos de uma dada instrução na ALU.
- Alguns processadores DSP de última geração possuem duas ou mais MAC (*Multiply Accumulate Unit*) executando em paralelo, o que torna a comparação em MIPS sem sentido.

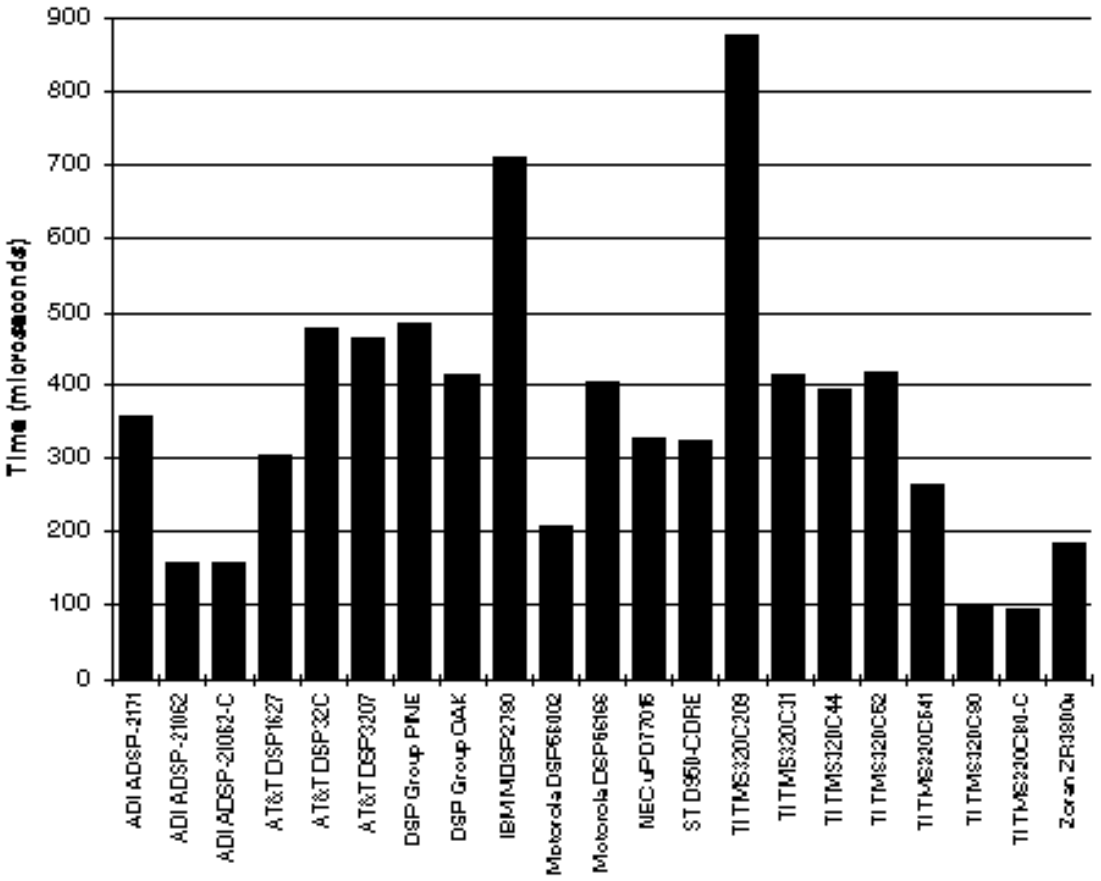
De forma a superar os problemas encontrados e tornar, assim, a análise de desempenho uma tarefa com resultados mais representativos é utilizado *benchmarks standard* [BDTI00b] para comparar a velocidade de execução de um conjunto de algoritmos, em diferentes processadores DSP. Os *benchmarks* são implementações que possibilitam a analise comparativa dos diversos processadores DSP disponíveis no mercado. Podem ser desde simples algoritmos até uma aplicação completa. Um exemplo da aplicação de *benchmarks* realizado pela empresa Berkely Design Technology Inc. é mostrado na figura 5.1.

Além das informações apresentadas e recomendadas, as seguintes considerações devem ser feitas:

- Deve-se atentar quanto ao uso dos termos: milhões de operações por segundo (*millions of operations per second* – MOPS) e milhões de operações em ponto flutuante por segundos (*millions of floating-point operations per second* – MFLOPS) , pois os fabricantes têm idéias diferentes quanto à definição do termo "operação". Por exemplo, alguns fabricantes afirmam que seus processadores em ponto flutuante tem a taxa de MFLOPS como sendo duas vezes a taxa de MIPS. A justificativa para esta afirmação é

que estes processadores são capazes de executar paralelamente as operações de multiplicação e de adição em ponto flutuante.

- A taxa de execução de instruções pode ser igual ao *clock* de entrada do processador DSP ou até um quarto de sua freqüência. Além disso, muitos dos novos processadores DSP apresentam circuitos duplicadores de *clock* ou PLL (*phase-locked loops*) que permitem o uso de *clocks* externos de baixa freqüência e geração interna das altas freqüências.



[2]Figura 5.1. Tempo, em ms, de Execução da FFT complexa com 256 amostras realizado pela Berkely Design Technology Inc., junho de 2000.

[2] Figura obtida no site da empresa Berkley Design Technology Inc, http://www.BDTI.com.

5.1.4 Organização da Memória

A organização do sistema de memória do processador tem grande impacto no seu desempenho. Por exemplo, as MACs rápidas buscam uma palavra de instrução e duas palavras de dados a cada ciclo de instrução. Para que tais tarefas sejam realizadas, existem uma variedade de recursos disponíveis, a saber:

- as memórias com múltiplos barramentos.
- memórias de dados e de programas separadas, representando a arquitetura Harvard e suas derivações encontradas, principalmente nos processadores DSP atuais.
- memórias *cache* de instrução e de dados.

As figuras 5.2 e 5.3 mostram graficamente as arquiteturas Harvard e Von Newmann.

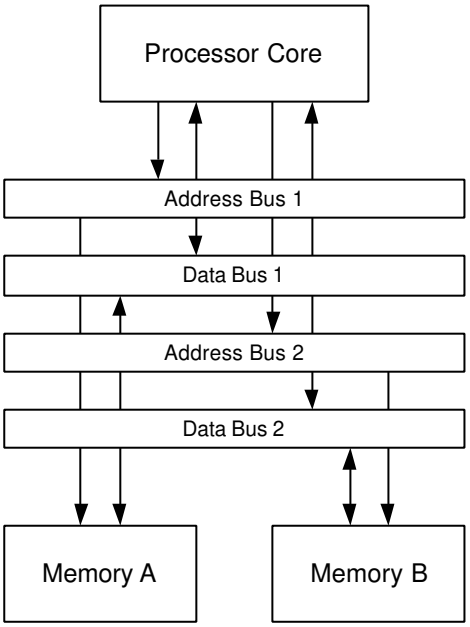


Figura 5.2. Arquitetura Harvard.

Um fator importante é o tamanho da memória *on-chip* e *off-chip* disponível. Na maioria dos processadores em ponto fixo, as memórias tendem a ser menores, apresentando tamanhos de memória interna entre 4K e 64K *words* e pequenos barramentos de acesso à memória. Muitos processadores DSP de ponto fixo apresentam barramento de endereçamento de 16 bits, o que limita o acesso à memória externa.

Alguns processadores DSP em ponto fixo fornecem uma quantidade relativamente maior de memória interna e barramento para acesso às memórias externas de tamanho

considerável. Como exemplo, podemos citar o processador DSP TMS320C30, da Texas Instruments, com 6K *words* de memória interna, barramento de 24 bits para endereçamento de memórias externas, assim como o processador DSP ADSP-21060 da Analog Devices, com 4Mbits de memória interna.

De modo geral, a melhor combinação de organização, tamanho e número de barramentos externos da memória depende da aplicação DSP.

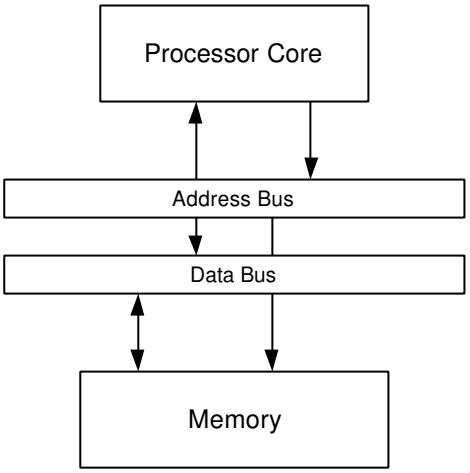


Figura 5.3. Arquitetura Von Newmann.

5.1.5 Facilidade de Desenvolvimento

O grau de facilidade de desenvolvimento de uma aplicação DSP depende exclusivamente dos objetivos a serem alcançados. Por exemplo, em pesquisas ou na elaboração de protótipos o principal requisito é a utilização de ferramentas que sejam de fácil manuseio. Por outro lado, uma companhia envolvida no desenvolvimento de um telefone celular para 3G, com restrições de custo no produto, pode abrir mão das facilidades de desenvolvimento em face a uma oferta de processadores DSP com menos recursos, mas que se enquadram na planilha de custo adotada.

Para realizar a escolha de um processador DSP alguns itens que contemplam a facilidade de desenvolvimento devem ser considerados. Os principais itens são:

- Ferramentas de software: *assemblers*, *linkers*, simuladores, depuradores, compiladores, biblioteca de códigos e sistema de operação e monitoramento em tempo real;
- Ferramentas de *hardware*: emuladores e placa de testes;

- Ferramentas de alto-nível: interface dos ambientes de programação e geração de códigos com recursos e facilidade de utilização, tal como o Code Composer Studio da Texas Instruments [DSPS99a]. A figura 5.4 mostra interação entre as diferentes ferramentas para desenvolvimento de uma aplicação DSP.

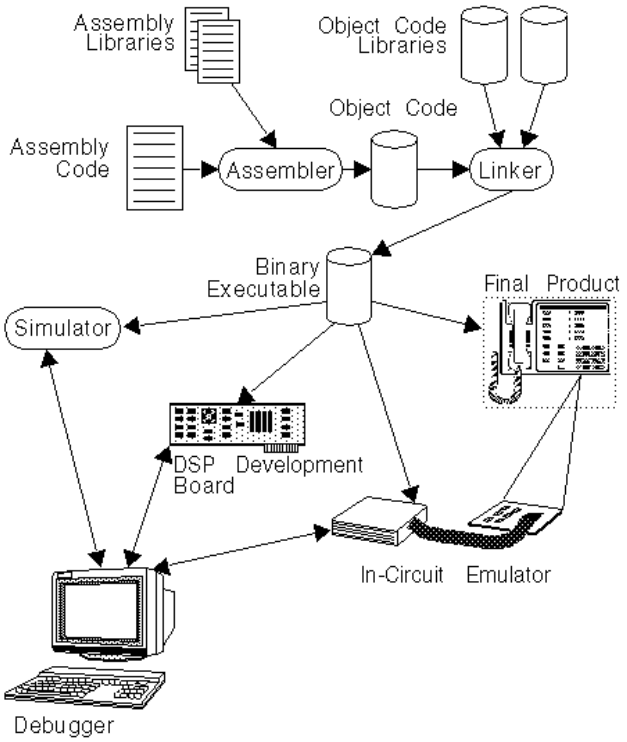


Figura 5.4. Interação das ferramentas de desenvolvimento para aplicações DSP disponíveis no mercado.

Uma questão fundamental na escolha do processador DSP é a linguagem de programação a ser utilizada. As escolhas possíveis são as linguagens *Assembly,* ANSI C e Ada [TD93, TI01, ADA95] ou a combinação das mesmas. Devidos às restrições de custo e velocidade de processamento, atualmente, a grande maioria dos programas DSP são elaborados com a linguagem *Assembly.* Desta forma, uma grande quantidade de tempo é utilizada pelo programador na otimização de códigos em *Assembly.* Esta é uma realidade bastante comum nas aplicações comerciais comuns, tais como celulares, *CD player* portáteis, telefones IP, controladores de geladeiras, entre outras, visto que são bastantes sensíveis aos acréscimos significativos no custo final dos produtos devido, principalmente, à grande concorrência existente.

Em aplicações DSP, com o uso de compiladores de linguagem em alto nível, melhores resultados são alcançados para processadores DSP em ponto flutuante do que em ponto fixo, pelas seguintes razões:

- As linguagens de alto nível não fornecem estrutura de programação otimizada para aritmética em ponto fixo;
- Os processadores DSP em ponto flutuante têm características mais regulares, menos conjuntos de instruções com restrições e, consequentemente, melhores compiladores;
- Os processadores em ponto flutuante normalmente acessam maiores espaços de memória que os processadores em ponto fixo, ou seja, possibilitam a acomodação dos códigos gerados pelo compilador, o qual tende a ser maior que o elaborado à mão, usando a linguagem *Assembly*.

Os processadores DSP baseados na arquitetura VLIW são os mais simples, pois têm um conjunto de instruções *Orthogonal RISC* e grandes registradores de arquivos, tornando possível a compilação otimizada de programas, realizados em linguagem de alto nível, tal como o ANSI C. No entanto, estes códigos ainda são ineficientes quando comparados aos códigos gerados e otimizados diretamente em *Assembly*. Normalmente, a combinação de códigos em C e em *Assembly* é uma solução adotada para estes processadores.

Alguns resultados interessantes são obtidos com os compiladores de programas em C, da Texas Instruments, aplicado aos processadores DSP TMS320C6xx. Nos *benchmark*s *standard* selecionados pela TI para teste, o desempenho alcançado foi de 80% a 95% em relação aos códigos escritos diretamente com a linguagem *Assembly*.

Finalmente, os fabricantes de processadores DSP, para a análise do desenvolvimento das aplicações DSP, disponibilizam ferramentas de emulação de *hardware*, processadores DSP com capacidade de emulação e depuração acessadas através de uma interface serial padronizada, chamada de *IEEE 1149.1 JTAG standard*. Através desta interface serial é possível inserir *breakpoints*, assim como observar e modificar o conteúdo dos registradores internos do processador. Além disso, são disponibilizados placas de desenvolvimento, tais como as placas C6201 Evaluation Module (EVM) Bundle [DSPS01] e C6211 DSP Starter Kit [TMS00b] da Texas Instruments, para a realização de simulações em tempo real, antes que o *hardware* do projeto esteja implementado, possibilitando o aumento da produtividade no desenvolvimento de novos projetos.

5.1.6 Multiprocessadores DSP

Algumas aplicações com uso computacional intensivo a alta taxa, tais como as verificadas em aplicação DSP em radar e sonar, normalmente necessitam de múltiplos processadores DSP. No entanto, a utilização de múltiplos processadores DSP não é trivial e seu uso eficiente depende da consideração dos seguintes requisitos:

- facilidade de interconexão em termos de projeto do circuito de comunicação entre os processadores DSP e custo de implementação da solução.
- Análise do desempenho da interconexão a partir do *overhead*, latência e *throughput* da comunicação entre os processadores DSP.

Como exemplo de DSP voltados a aplicações com múltiplos processadores apresentam-se os processadores da família ADSP-2106x, da Analog Devices, os quais contêm hardware de uso geral para sistemas com múltiplos processadores.

5.1.7 Gerenciamento do consumo de energia

Com o aumento explosivo do uso de processadores DSP em equipamentos portáteis, a utilização de recursos que possibilitam o gerenciamento do consumo de energia e sua consequente redução por instrução executada tem influenciado as principais implementações DSP (telefones celulares, etc.).

Para prover uma visão geral que possibilite um entendimento dos principais recursos de gerenciamento de energia presentes em alguns processadores DSP, tais como os da família TMS320C54x [KGB00] e TMS320C55x [TMS00a], é necessário se conhecer as seguintes técnicas de gerenciamento de energia:

- Utilização de baixas tensões de alimentação, tais como 3.3, 2.5 e 1.8 volts, nas diferentes versões disponíveis, possibilitando um consumo de energia muito abaixo do que os encontrados em processadores DSP alimentados com 5 volts.
- Os modos *"Sleep"* ou *"Idle"* possibilitam o desligamento do *clock* do processador em algumas partes internas, de modo a reduzir o consumo de energia quando não houver processamento em execução.
- divisor de *clock* programável configura diferentes valores de freqüências, via *software*, para se obter freqüência de *clock* mínima para uma tarefa particular.

- controle periférico capacita o *software* DSP a desabilitar dispositivos periféricos que não estão em uso e habilitá-los quando necessário.

A desconsideração das características de gerenciamento de energia torna difícil a obtenção de uma figura de consumo do processador utilizado, já que o consumo de energia pode variar de acordo com o tipo de instrução executada.

A Texas Instruments, por exemplo, fornece detalhadas informações sobre o consumo de energia associado aos tipos de instruções e configurações de seus processadores com o intuito de prover o engenheiro de projeto com informações necessárias à elaboração de um protótipo com qualidade [TMS00a].

5.1.8 Custo

O preço dos processadores DSP é o fator principal nos componente com produção em larga escala. Normalmente, as empresas usam processadores DSP de baixo custo de forma a não ultrapassar os custos dos projetos, mesmo que tal decisão implique pouca flexibilidade ou dificuldade de programação e de utilização.

Dentre as famílias disponíveis, as de menor custo tende a ser formada de processadores com menos memória interna e menor desempenho.

Um fator importante no preço dos processadores DSP é o custo do invólucro de empacotamento. Por exemplo, os invólucros PQFP (*Plastic Quad Flat Pack*) e TQFP (*Thin Quad Flat Pack*) podem ser muito mais baratos que um invólucro PGA (*Pin Grid Array*) [BDTI00a].

Finalmente, ao se considerar os preços dos processadores DSP, é importante destacar que estes estão cada vez mais baixos e são significativamente dependentes da quantidade a ser comprada.

5.2 Placa EVM e DSK 6211

A Texas Instruments, TI, atual líder mundial na fabricação de processadores DSP, tem fornecido uma gama enorme de opções para aplicações nas mais diversas áreas, tais como, controle, telecomunicações, biomédica, etc. Dentre as opções que apresentam maior desempenho, podemos ressaltar os processadores da família TMS320C6000. Estes processadores possibilitam a execução de até oito instruções em paralelo, possibilitando a

realização de 1000MIPS a 5000MIPS. Dentre os mais baratos processadores da família TMS320C6x têm-se o TMS320C6201 (ponto fixo a 16 bits) e TMS320C6211 (ponto-fixo a 16bits). Estes têm sido muito empregados em aplicações que demandam bastante custo computacional, baixo custo e não tem restrições rigorosas de consumo de energia. Por exemplo, a Texas Instruments implementou alguns codificadores de voz nos processadores C6201 e C6211 [FZ00 e FU00], de forma que vários canais fossem contemplados.

Neste presente trabalho, os processadores escolhidos para implementação do codec G.722.1, da técnica de reconstrução de pacote baseada na transformada wavelet e classificação do sinais de voz baseada em redes neurais e informações estatísticas, foram o TMS320C6201 e TMS320C6211, as informações técnicas destes processadores podem ser encontrada em [DSPS01]. Dentre as placas de desenvolvimento disponíveis no mercado que utilizam estes processadores, escolhemos as placas EVM C6201 e DSK C6211, a partir dos seguintes critérios:

1. Disponibilidade de se obter alguma placa que permitisse a implementação dos algoritmos sem uma grande preocupação de custo computacional, juntamente ao fabricante, no caso TI.
2. Escolha de um processador com aritmética em ponto fixo e 16 bits, pois os mesmos têm custo menor que os em ponto flutuante.
3. Ambiente de desenvolvimento de fácil familiarização e aprendizado. Por exemplo, o CCS (Code Composer Studio) da Texas Instruments é um excelente ambiente de desenvolvimento de algoritmos DSP.
4. Tamanho da memória interna do processador. Neste caso, a necessidade de se ter uma memória interna de tamanho razoável possibilita que não haja perda de tempo no acesso as memórias de dados e programa externos ao processador DSP.
5. Possibilidade de implementação de vários canais, assim como em [FZ00 e FU00].
6. Capacidade de processamento.

Algumas informações sobre as placas EVM C6201 e DSK 6211 são reunidas nas seções 5.2.1 e 5.2.2, respectivamente.

5.2.1 Placa EVM C6201

A placa EVM C6201 é um placa de desenvolvimento de aplicações DSP utilizada em desenvolvimento de implementações de estação rádio base para telefonia sem-fio, modems, servidores de acesso remoto (*Remote Access Servers* – RAS), sistemas xDSL (*Digital subscriber loop*), cable modens, sistema de telefonia com múltiplos canais, aplicações de imagens, sistemas de reconhecimento de voz, aplicações 3-D e etc.

A placa EVM C6201 usa o barramento PCI (*Peripheral Component Interconnect/Interface*) ou a conexão JTAG para comunicação com o PC. Pode ser alimentada via barramento PCI ou por uma fonte externa. Dentre as principais funcionalidades disponíveis pode-se ressaltar:

- Depuração do código através do Code Composer Studio.
- Interface PCI.
- Bancos de memória SBSRAM (*Synchronous Burst Static RAM*) e SDRAM (S*ynchronous Dynamic RAM*).
- ADC (*Analog to Digital Converter*) e DAC (*Digital to Analog Converter*) para sinais de voz com 16 bits e 16kamostras/s.
- Suporte para emulação JTAG.
- Barramento e sinalização para conexão com outras placa.
- Interface para memória externa (*External Memory Interface* - EMIF).

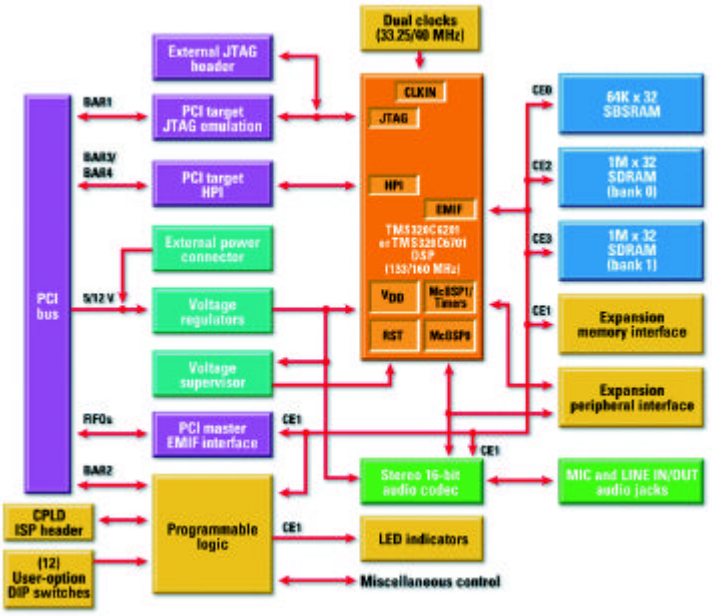
A figura 5.5 mostra o hardware da placa EVM dividido em 12 componentes, tal como é descrito a seguir:

- **Processador C6201** – a placa EVM C6201 é baseada no processador C6201, operando até 1600 MIPS para freqüência de clock de 200 MHz.
- **DSP clock** – a placa EVM C6201 possui dois geradores de clock (33.25 MHz e 40 MHz) e dois modos de operações (múltiplo de 1 e múltiplo de 4). Estes modos possibilitam ao DSP operar em quatro diferentes freqüências de clock. Por exemplo, a placa EVM C6201 utilizada nesta tese tem o DSP TMS320C6201 com clock a 133MHz.
- **External memory** – A placa EVM C6201 possui os seguintes bancos de memória:
    - um banco de memória SBSRAM, de 64K x 32, operando a133-MHz

- dois bancos de memória SDRAM, de 1M x 32, operando a 100-MHz.

Além disso, bancos de memória assíncrona podem ser adicionados pela interface de expansão de memória. De modo geral, todos os dispositivos de memória externa são endereçáveis por bytes.

- **Expansion interfaces** – A placa EVM C6201 possui interface de conexão.
- **PCI interface** – A interface PCI possibilita ao PC acessar o controlador JTAG da EVM, a interface de conexão ente o DSP e o PC (*Host Port Interface* - HPI) e os registradores de controle bem como os de sinalização da placa EVM.
- **JTAG emulation** – A emulação JTAG é mantida pelo uso do controlador de teste do barramento no DSP (Test Bus Controller - TBS) e do emulador externo XDS510.
- **Programmable logic** – A EVM tem CPLD (*Controlled Programmable Logic Devices*) para controlar e operacionalizar a lógica e os registradores de controle e sinalização da placa EVM.
- **Audio interface** – A placa EVM inclui uma interface de áudio com qualidade de CD para microfone estéreo e conexão de entrada e saída estéreo.
- **Power supplies** – Os reguladores de tensão da placa EVM fornecem os seguintes valores de voltagem:
  - 2.5V para o núcleo do processador C6201.
  - 3.3 V para as memórias de I/O do C6201, CPLD e depuradores.
  - 5 V para os componentes da interface de áudio.
- **Voltage supervision and reset control** – A placa EVM contém sistema de supervisão de voltagem para monitorar as tensões na placa e fornecer o sinal de reset da placa.
- **LED indicators** – Os *leds* de indicação sinalizam quando há alimentação na placa e são usados para sinalizações definidas pelo programador.
- **User options** – a placa EVM possui controle das suas configurações a partir de um DIP switched ou via software a partir do PC.

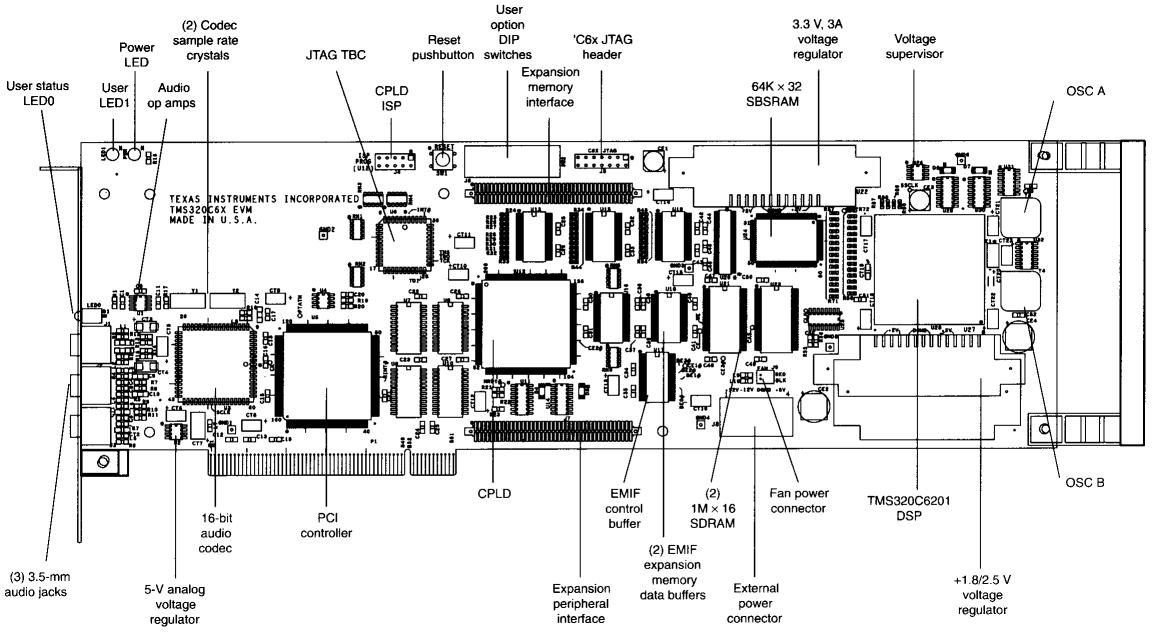


[3]**Figura 5.5**. seções do hardware da placa EVM C6201.

O software fornecido pela Texas Instruments para desenvolvimento das aplicações DSP é o Code Composer Studio (CCS), o qual possibilita o controle total das funcionalidades presentes na placa EVM e contém sofisticadas ferramentas para elaboração, compilação, otimização e depuração dos algoritmos DSP implementados. Apesar de se encontrar alguns bugs ou erros que em certos casos tornam tarefas simples numa grande dor de cabeça, o CCS é com certeza uma excelente ambiente de desenvolvimento de algoritmos DSP.

O *layout* da camada superior da placa EVM C6201 pode ser visualizado na figura 5.6.

[3] Figura obtida no site da empresa Texas Instruments, www.dspvillage.ti.com.



[4]**Figura 5.6**. Layout superior da placa EVM C6201.

5.2.2 A placa DSK C6211

A fig. 5.7 mostra a placa DSK C6211e seus principais dispositivos. As semelhanças entre os processadores C6211, presente na placa DSK 6211, e o C6201 podem ser resumidas da seguinte forma [DSPS99c]:

- A CPU do processador C6211 é idêntica à do processador C6201, possibilitando que os códigos desenvolvidos para C6201 rodem no C6211.
- A McBSP (*Multi-channel Buffered Serial Port*) é idêntica à do C6201.
- A Interface assíncrona de 16 bits para conexão com o PC, HPI (*Host Port Interface*), é a mesma presente no C6201.
- O processador C6211 possui dois temporizadores de 32 bits.
- As fontes para interromper a CPU ou enviar um evento para o controlador de DMA são semelhantes àquelas presentes no processador C6201.

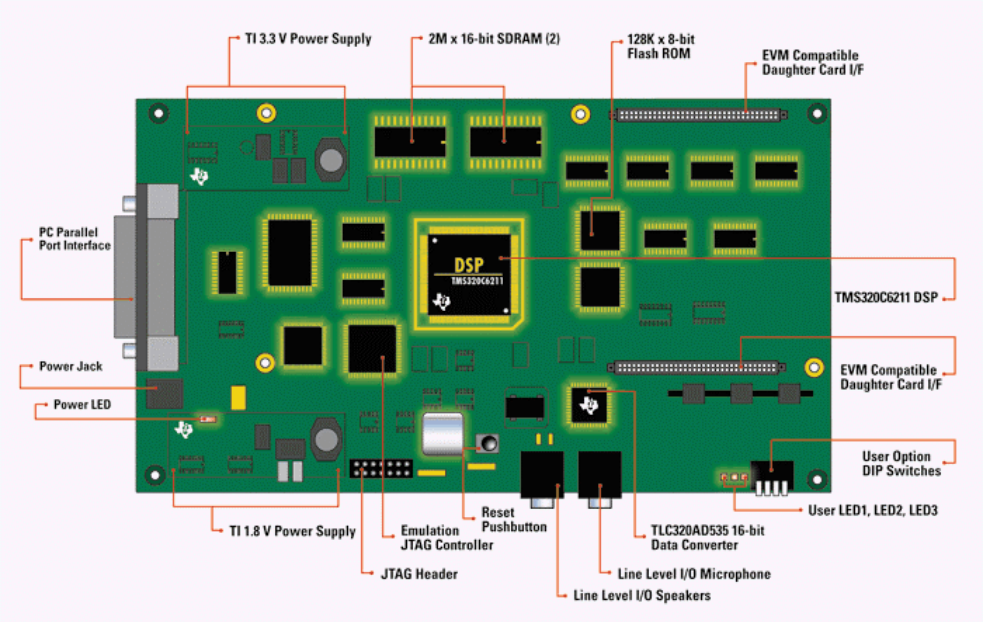
As diferenças entre os processadores C6211 e C6201 são as seguintes:

[4] Figura obtida do manual de instalação da Placa EVM 6201 da Texas Instruments, www.dspvillage.ti.com.

- Os processadores C6211 operam com freqüência de clock máxima de 150 MHz e 1200MIPS.
- Os processadores C6211 são dispositivos DSP baseados em memória *cache*. A estrutura da memória *cache* permite alto desempenho no processamento dos algoritmos DSP. Por exemplo, em média os algoritmos apresentam perda de 10 a 20% [MM00] no desempenho quando transladado de uma processador C6201 para C6211.
- No DSP C6211 é implementado um controlador de DMA, denominado *Enhanced Direct Memory Access* (EDMA) *controller* [BELL01, DSPS99b], permitindo o uso mais flexível dos canais de DMA.
- A interface de memória externa, (EMIF), foi melhorada para permitir o interfaceamento com outros tipos de memórias. Desta forma, o DSP C6211 pode interfacear com memórias assíncronas, SBSRAM, SDRAM, de 32, 16 e 8 bits.

A placa C6211 realiza a conexão com o PC através da porta paralela do PC e o uso do ambiente de desenvolvimento Code Composer Studio e de suas ferramentas são disponibilizados para desenvolvimento de aplicações DSP.

A grande vantagem da placa DSK C6211 é o seu baixo custo de aquisição, cerca de US$ 300,00, quando comparada a EVM C6201, cerca de US$ 2000,00.



[5]**Figura 5.7**. Layout superior da placa DSK C6211.

5.3 Descrição da implementação do protótipo para telefonia IP

A figura 5.8 mostra o diagrama de blocos da implementação em DSP dos algoritmos de processamento de sinais de voz durante o processo de codificação. Estes algoritmos foram implementados com a linguagem ANSI C no ambiente de programação Code Composer Studio da Texas Instruments para o DSP TMS320C6201.

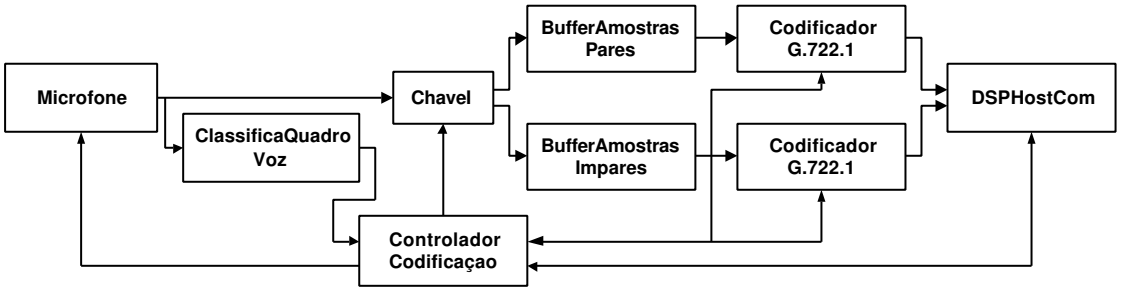


Figura 5.8. Diagrama de bloco do Codificador implementado no DSP.

Os blocos do diagrama do codificador, conforme mostrado na fig. 5.8, realizam as tarefas especificadas a seguir:

[5] Figura obtida do site da Texas Instruments, www.dspvillage.ti.com.

- **Microfone:** Este bloco realiza a aquisição dos quadros de 40ms ou 640 amostras de sinal de voz amostrado a 16kamostras/s e quantizados a 16 bits. Para realizar estas tarefas neste bloco estão implementadas as rotinas de configuração do Codec CS4231A da placa EVM C6201 [CAR98], do McBSP [ANJ98a e ANJ98b] para comunicação entre o codec de áudio, CS4231A, e o DSP, configuração dos canais de DMA 1 e 2 para realização da aquisição dos quadros de amostras de voz utilizando o conceito de ping-pong [BELL99].
- **ClassificaçãoQuadroVoz**: Neste bloco o quadro de 40 ms ou 640 amostras de sinal de voz é dividido em 4 sub-quadros de 10ms. Cada sub-quadro de 10ms é submetido ao algoritmo, baseado em redes neurais e proposto no capítulo 4, para classificação em silêncio, voz sonora ou surda. As seguintes considerações são feitas:
  - Se todos os sub-quadros são classificados como silêncio, então o quadro de 40ms não é submetido ao processo de embaralhamento, codificação e envio para o host ou PC.
  - Se no máximo três sub-quadros são classificados como silêncio, então os quadros classificados como silêncio são especificados como surdos e os processos de embaralhamento, codificação e envio dos quadros codificados para o host é realizado. A especificação dos sub-quadros de silêncio como surdos é realizada para informar ao controlador de decodificação que estes sub-quadros serão reconstruídos a partir da rotina para reconstrução de voz surda.

A fig. 5.9 mostra a estrutura da rotina dentro deste bloco, a partir da proposta apresentada no capítulo 4.

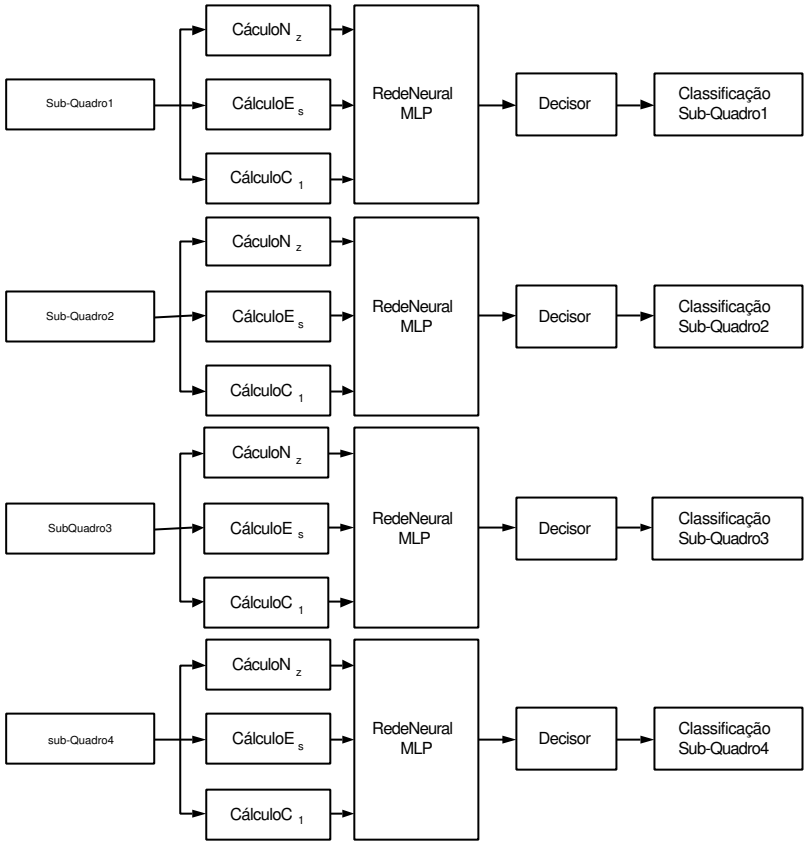


Figura 5.9. Estrutura interna do bloco **ClassificaçãoQuadroVoz**.

- **ControladorCodificação**: Neste bloco estão contidos as rotinas para habilitar o bloco **ChaveI**, a partir da informação de classificação dos sub-quadros de sinal de voz, permite o processo de embaralhamento, codificação e envio do quadro codificado para o host através de uma conexão HPI do DSP [BIS99 e NIK98], implementada no bloco **DSPHostCom**. Além disso, o **ControladorCodificação** configura o codificador G.722.1 para as taxas de 16k, 24k e 32kbits/s e controla a execução das rotinas presentes no bloco **Microfone**, a partir das informações recebida do host**.**
- **ChaveI:** De acordo com a sinalização do **ControladorCodificação** realiza o embaralhamento, $SIF = 2$, das amostras pares ou ímpares a serem submetidas ao processo de codificação executado no bloco **CodificadorG.722.1**.
- **BufferAmostrasPares:** armazena as amostras pares do quadro de 40ms ou 640 amostras a serem submetidas à codificação, ou seja, armazena 20ms ou 320 amostras do sinal de voz.

- **BufferAmostrasImpares:** armazena as amostras ímpares do quadro de 40ms ou 640 amostras a serem submetidas à codificação, ou seja, armazena 20ms ou 320 amostras do sinal de voz.
- **CodificadorG.722.1**: contém as rotinas para realizar a codificação dos buffers de amostras pares e ímpares de acordo com a norma G.722.1 da ITU-T. Os quadros codificados são entregues ao bloco **DSPHostCom**.
- **DSPHostCom**: realiza a comunicação entre o DSP e o Host através de uma conexão HPI. É responsável pelo envio dos quadros de amostras pares e ímpares codificadas para o Host. Além disso, transfere as informações e sinalizações do Host para o **ControladorCodificação**.

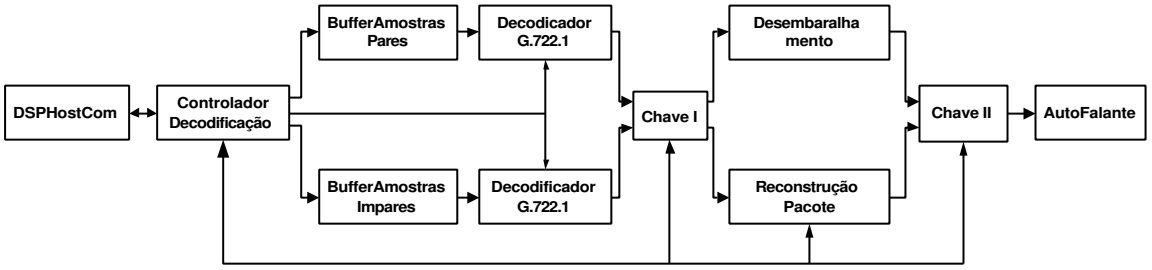


Figura 5.10. Diagrama de bloco do Decodificador implementado no DSP.

Os blocos presentes no diagrama de blocos do decodificador, mostrado na fig. 5.10, realizam as seguintes tarefas:

- **DSPHostCom**: Além das tarefas especificadas acima, para o codificador, este bloco recebe os quadros vindos do PC, através da conexão HPI, envia e recebe sinalização para garantir a decodificação dos quadros codificados e recebidos do Host.
- **ControladorDecodificação**: Recebe o quadro codificado juntamente com as informações de codificação para realizar a reconstrução dos pacotes perdidos ou o desembaralhamento dos quadros recebidos, se não houve perda.
- **BufferAmostrasPares**: realiza a mesma tarefa especificada para o codificador.
- **BufferAmostrasImpares**: realiza a mesma tarefa especificada para o codificador.
- **Decodificador G.722.1:** contém os algoritmos para realizar a decodificação dos quadros de amostras pares e/ou ímpares. A decodificação é habilitada pelo

ControladorDecodificação, a partir das informações de perda dos quadros, recebidas do Host.

- **ChaveI**: De acordo com a sinalização do ControladorDecodificação realizada o processo de desembaralhamento ou de reconstrução de pacotes. A escolha da opção de desembaralhamento significa que não houve perda de pacotes durante a transmissão e, conseqüentemente, não há necessidade de se efetuar a reconstrução de pacotes. Por outro lado, a consideração da reconstrução de pacotes é executada quando há perda de pacotes.
- **Desembaralhamento**: contém a rotina que realiza o desembaralhamento dos quadros de amostras pares e ímpares decodificada e, correspondentes ao quadro de 40ms submetido ao processo de embaralhamento e codificação.
- **ReconstruçãoPacote**: contém as rotinas para realizar a reconstrução das amostras pares ou ímpares com a técnica RITW proposta para reconstrução de pacotes. A estrutura implementada dentro deste bloco é mostrada graficamente na fig. 5.11. A reconstrução dos pacotes de amostras pares ou ímpares dar-se-á em sub-quadros de 10ms armazenados nos blocos **Sub-QuadroiComPerda**, $i = 1,...,4$. Posteriormente, os blocos **AmostrasSonoras** e **AmostrasSurdas** armazenam os quadros de amostras decodificados com perdas de acordo com a classificação efetuada pelo algoritmo de classificação de segmentos de voz no transmissor. Finalmente os sub-quadros são submetidos ao algoritmo de reconstrução contido nos blocos **Sub-QuadroiReconstruído**, $i = 1,...,4$. Os sub-quadros reconstruídos são ordenados no bloco **QuadroReconstruído**.
- **ChaveII**: A partir da sinalização do **ControladorDecodificação** seleciona o quadro de sinal de voz original na saída do bloco **Desembaralhamento**, se não houve perda, ou na saída do bloco **ReconstruçãoPocote,** se houve perda de pacotes.
- **AutoFalante**: contém as rotinas para realizar transferência do sinal de voz reconstruído ou desembaralhado, via DMA, entre a memória de dados e o auto-falante acoplado a saída do codec CS4231A.

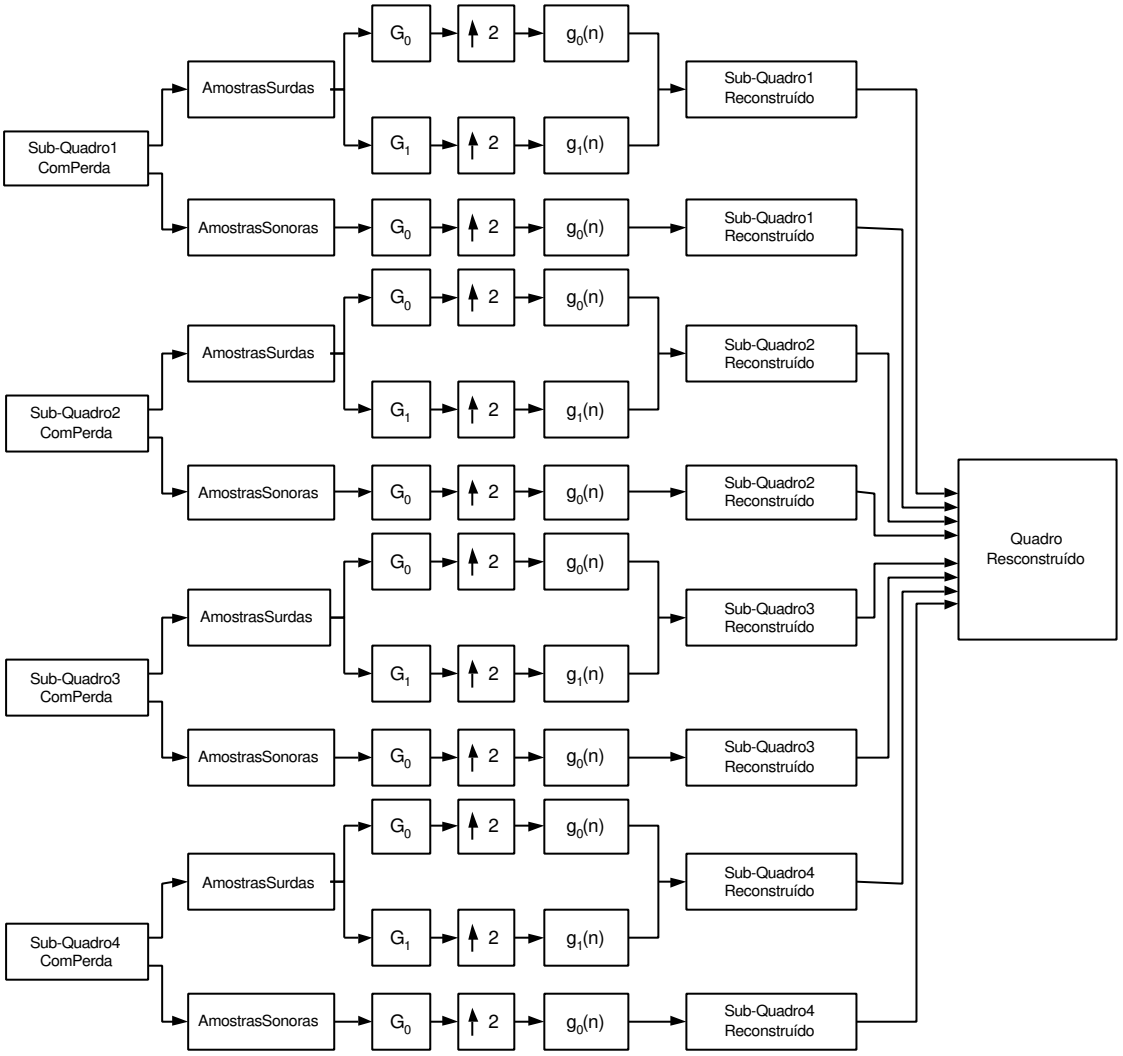


Figura 5.11. Estrutura implementada no bloco de **Reconstrução Pacote**.

A implementação das rotinas no DSP foram realizadas tendo em mente a garantia de que as mesmas fossem executadas em tempo real. Por exemplo, os códigos fonte do algoritmo do Codec G.722.1, obtidos diretamente da ITU-T, inicialmente consumiam um tempo superior a 300ms para realizar a codificação e decodificação de um quadro de 20ms. No entanto, fazendo uso das técnicas de otimização de algoritmos DSP disponíveis em [DSPS00a, DSPS00b, SCA00, HUB00, SCA99a e SCA99b] foi possível diminuir o tempo de execução desta rotina para 7.94ms, para $f_{CPU} == 133MHz$. Para se atingir os valores de otimização mencionado anteriormente foi considerada a seguinte metodologia prática para a otimização de códigos:

1º Passo: Limpar o código ou fazer o código mais simples possível.

2º Passo: Inserir *intrinsic* para operações em 32 bits.

3º Passo: Inserir adaptações da *intrinsic* de 32 bits para realizar operações a 16 bits.

4º Passo: Rearranjar os laços para realizarem várias operações em paralelo.

5º Passo: Utilizar o conjunto de opções de configuração para compilação e otimização do código no CCS, visando o equilíbrio entre velocidade e/ou tamanho do código.

Esta metodologia prática não contempla a otimização baseada na mudança da forma de implementação das rotinas, tais como a modificação da rotina de execução da MLT nos quadros de 320 amostras do sinal de voz e uso extensivo da linguagem Assembly para otimizar o código e configurar diretamente o *pipeline*. A modificação da forma de implementação das rotinas para se obter melhor desempenho é interessante, todavia a utilização da linguagem *Assembly* gera alguns problemas de ordem prática, tais como:

- As rotinas em *Assembly* são dependentes da estrutura do DSP.
- A configuração direta do *pipeline* dos processadores não é uma tarefa fácil de ser implementada, apesar de se obter os melhores resultados com esta ação.
- A elaboração de rotinas em *Assembly* requer bastante conhecimento da mesma e da estrutura interna do DSP para se obter o melhor desempenho do DSP.

A partir das considerações acima, pode-se argumentar que a utilização da linguagem *Assembly* deve ser utilizada quando o programa em questão será implementado para alguma aplicação comercial, pois, neste caso, a minimização dos custos de implementação dos equipamentos ou dispositivos é bastante importante e permite o lançamento de um produto com preço menor que o concorrente e agregando mais funcionalidades. Por exemplo, em [FZ00] é apresentada uma implementação de vários codificadores para 10 canais de telefonia, enquanto que a nossa implementação contempla no máximo 4.2 canais, considerando $f_{CPU} = 200\,MHz$, ou seja, a implementação [FZ00] faz, com certeza, uso da linguagem Assembly e, talvez, da configuração manual do *pipeline* do DSP.

A implementação das rotinas de reconstrução de pacotes não representou nenhum problema, pois as mesmas foram previamente implementadas no MatLab com aritmética de 16 bits e, posteriormente, transladada para o DSP. Todavia, o mesmo não pode ser aferido na implementação da rotina de classificação de segmentos de voz, pois esta realiza várias

operações em ponto flutuante e como o processador C6201 é de 16bits, tais operações consomem bastante ciclos de execução, como é mostrado na tabela 5.1, onde o desempenho das rotinas implementadas no C6201 (placa EVM C6201) e no C6211 (placa DSK C6211) é mostrado.

Na implementação dos algoritmos na placa DSK C6211 não foi considerado a comunicação com o Host nem a aquisição das amostras do sinal de voz, pois o objetivo foi verificar a diferença de desempenho entre os dois tipos de processadores.

Tabela 5.1. Custo Computacional.

Algoritmos	C6201 EVM (ms)	C6211 DSK (ms)
Classificador Neural	3.00	3.60
Embaralhamento	0.05	0.06
Codificador G.722.1	3.99	4.85
Decodificador G.722.1	3.86	4.78
Reconstrução de Pacotes	0.30	0.36
Desembarralhamento	0.05	0.06

Os algoritmos, implementados no PC para interfaceamento com a placa EVM C6201 utilizam a linguagem de programação Visual C++ [LAP98, HORT98, FZ00].

A largura de banda necessária para transmissão dos pacotes desconsiderando-se a técnica de reconstrução é apresentada na tabela 5.2. Equanto a tabela 5.3 mostra a largura de banda requisitada, supondo o envio dos quadros de amostras pares submetidos ao codec G.722.1 e reconstrução das amostras ímpares com a técnica RITW implementada no DSP. Como pode ser observado, o codec G.722.1, com algumas adaptações, pode ser utilizado nas situações de perdas de pacotes durante uma transmissão via Internet ou para reduzir a taxa de tranmissão até 4kbps com um nível de qualidade de acordo com as tabelas 3.17, 3.18 e 3.19. O único inconveniente desta solução é o aumento do atraso fim a fim provocado pela técnica de embaralhamento.

Tabela 5.2. Largura de banda requisitada.

Codificador G.722.1 sem supressão de silêncio	Codificador G.722.1 com supressão de silêncio
16kbits/s	8kbps
24kbtis/s	12kbps
32kbits/s	16kpbs

Tabela 5.3. Largura de banda requisitada.

Codificador G.722.1 com Perda das Amostras ímpares	Codificador G.722.1 com Perda das Amostras ímpares e supressão desilêncio
16kbits/s	8kbits/s
12kbtis/s	6kbits/s
8kbits/s	4kbits/s

5.4 Sumário

Neste capítulo procuramos reunir primeiramente os critérios para escolha de um processador DSP e a consideração dos mesmos na implementação da solução para telefonia IP descrita. Além disso, discutimos brevemente alguns informações importantes concernentes as placas DSP utilizadas e, finalmente, descrevemos como a implementação do protótipo para telefonia IP foi conduzida.

No capítulo a seguir estaremos fazendo as considerações finais sobre os resultados obtidos neste capítulo e nos anteriores.

6 Conclusão

O presente trabalho teve como objetivo o estudo das principais características de um sistema VoIP, tendo como foco de pesquisa as técnicas de reconstrução de pacotes para codificadores por forma de onda. Tal enfoque é bastante pertinente no contexto atual, pois a utilização da Internet como meio para transmissão, em tempo real, de sinais de voz e vídeo, é uma realidade. No entanto, a perda de pacotes observada em redes IP é bastante significativa, o que pode tornar problemática a manutenção de um nível aceitável de qualidade na transmissão de voz por pacotes, via Internet.

As técnicas propostas para a reconstrução de pacotes utilizando a transformada wavelet e as redes neurais representam soluções interessantes a serem consideradas em codificadores por forma de onda, pois possibilitam ganhos superiores à técnica mais atual encontrada na literatura (RTRANS). Diferentemente desta e da técnica MLP1 proposta, as técnicas de reconstrução, RITW e MLP2, consideram as características intrínsecas dos sinais de voz, tornando possível a obtenção de melhores resultados. Por exemplo, observou-se que uma única técnica para reconstrução de sinais de voz surda e sonora não é a escolha ideal, devido às características espectrais das mesmas.

A utilização de MLPNN para realizar a interpolação das amostras perdidas apresenta resultados interessantes a um custo computacional muito superior à técnica RITW e com menor desempenho (MLP1) e quase compatível (MLP2). Talvez uma solução que mereça uma investigação mais profunda seja a utilização de estruturas wavelet-neural para reconstrução de pacotes. Tal idéia constitui uma possível linha de pesquisa a ser desenvolvida.

A técnica RITW tem desempenho que supera as demais técnicas com um custo computacional bastante baixo e sua utilização para promover a diminuição da taxa do codec G.722.1 é uma das pesquisas em andamento.

As consideração implementadas no classificador neural proposto possibilitou a obtenção de uma estrutura com bom desempenho e com baixo custo. Tal solução foi possível a partir da escolha criteriosa dos dados de entrada da rede e da conjectura de que

as entradas de uma rede neural quando normalizadas permitem melhor desempenho durante o processo de aprendizagem.

A implementação dos algoritmos de processamento no DSP possibilitou abordar e investigar questões relacionadas à otimização de códigos em processadores DSP, a fim de se colocar a aplicação em tempo real e aprofundar detalhes da implementação de algoritmos em DSP, usando as linguagens "C" e *Assembly*. Tais questões são extremamente importante quando se deseja verificar o desempenho de algoritmos e soluções em aplicações concretas.

Como pesquisas futuras pretendemos analisar o uso de estruturas wavelet-neural para realizar classificação de segmentos de voz e reconstrução de pacotes. Além disso, almejamos estudar a implementação em ponto fixo em 16 bits das redes neurais.

Finalmente, devido ao aumento do atraso fim a fim provocado pela técnica de embaralhamento iremos estudar técnicas que minimizem tal efeito.

Referências

[ADA95] http://adaic.org/compilers/cpl/lists/CPL-ft-frame.html.

[ADEHP99] M. Arango, A. Dugan, I. Elliott, C. Huitema, S. Pickett. "Media Gateway Control Protocol (MGCP) Version 1 – RFC 2705." *IETF*, www.ietf.org, October, 1999.

[ANJ98a] S. Anjanaiah. "TMS320C6000 McBSP Initialization." *Texas Instruments Application Report*, SPRA488, Nov. 1998.

[ANJ98b] S. Anjanaiah. "Using the TMS320C6x McBSP as a High Speed Communication Port." *Texas Instruments Application Report,* SPRA455, May, 1998.

[AR76] B. S. Atal, L. R. Rabiner. "A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition." *IEEE Trans. On Acoustics, Speech, and Signal Processing*, vol. ASSP-24, nº 3, June, 1976.

[BAR01] J.G.A. Barbedo. "Medida Objetiva de Qualidade de Codecs de Voz na Faixa de Telefonia." *Tese de Mestrado*, FEEC/Unicamp, Campinas, 2001

[BDTI00a] Berkely Design Technology Inc. "Choosing a DSP processor White Paper", 2000.

[BDTI00b] "The BDTImark™: A measure of DSP execution speed White Paper", Berkely Design Technology Inc., 2000.

[BELL99] D. Bell. "TMS320C6000 DMA Example Applications." *Texas Instruments Application Report*, SPRA529, Mar., 1999.

[BELL01] D. Bell. "TMS320C621x/TMS320C671x EDMA Queue Management Guidelines." *Texas Instruments Application Report,* SPRA720 - February 2001.

[BGG98] C. S. Burrus, R. A. Gopinath, H. Guo. *Introduction to wavelets and wavelet transforms: A Primer*. Prentice Hall, New Jersey, 1998.

[BIS99] E. Biscondi. "Implementing the TMS320C6201/C6701/C6211 HPI Boot Process." *Texas Instruments Application Report,* SPRA512, January, 1999.

[BR00] M. Baldi, F. Risso. "Efficiency of packet voice with deterministic delay", *IEEE Communications Magazine*, pg. 170-177, May, 2000.

[BRA94] R. Braden, et al.. "Integrated Services in the Internet Architecture: an overview - RFC 1633." *IETF*, July, 1994.

[BS93] T. Back, H. P. Schwefel. "An overview of evolutionary algorithms for parameter optimization." *Evol. Comput*., 1(1), 1993.

[CAR98] B. G. Carlson. "TMS320C6000 McBSP Interface to the CS4231A Multimedia Audio Codec." *DNA Enterprises, Inc., Texas Instruments Application Report*, SPRA477, Dez., 1998.

[CC97] Y.-L. Chen, B.-S. Chen. "Model based multirate representation of speech signals and its applications to recovery of missing speech packets*." IEEE Trans. On Speech and Audio Processing*, vol. 5, nº 3, pags. 220-231, May, 1997.

[CHE97] Z. G. Chen. "Coding and transmission of digital video on the Internet." Ph.D. dissertation, *Dept. of Electrical and Computer Engineering,* University of Illinois, Urbana-Champaign, IL, 1997.

[CHF88] K. V. Chin, S. C. Hui, S. Foo. "Packet voice recovery techniques for real-time voice communication." *Proc. Of 4th Asia-Pacific Conference on Communications/6th Singapore International Conference on Communications Systems (APCC/ICC'98),* Singapore, pgs. 122-126, 1988.

[CHU97] C. K. Chui. *Wavelets: a tool for signal analysis*. SIAM, philadelphia, 1997.

[DAU92] I. Daubechies. *Ten lectures on wavelets.* SIAM, philadelphia, 1992.

[DMV78] P. G. Drago, A. M. Molinari, F. C. Vagliani. "Digital dynamic speech detectors." *IEEE Trans. On Communications*, vol. 26, pags. 140-145, January, 1978.

[DP00] J. Davidson, J. Peters. *Voice over IP fundamentals*. Cisco Press, Indianapolis, 2000.

[DRBA00] C. A. Duque, M. V. Ribeiro, H. A. C. Braga, S. Q. de Almeida. "Compressão de dados para análise de qualidade de energia elétrica utilizando transformada wavelet e filtros de Kalman." *Congresso Brasileiro de Automática (CBA2000)*, Florianópolis, Brasil, 2000.

[DSPS99a] Digital Signal processing Solutions. "Code Composer Studio White paper", *Texas Instruments Application Report,* SPRA520, maio, 1999.

[DSPS99b] Digital Signal Processing Solutions. "TMS320C6000 Peripherals – reference guide." *Texas Instruments Application Report*, SPRU190C, Abril, 1999.

[DSPS99c] Digital Signal Processing Solutions. "How to begin development today with the TMS320C6211 DSP." *Texas Instruments Application Report*, SPRA474, 1999.

[DSPS00a] Digital Signal Processing Solutions. *TMS320C6000 Programmer's Guide*. *Texas Instruments Application Report*, SPRU198D, March, 2000.

[DSPS00b] Digital Signal Processing Solutions. "*Optimizing Compiler User's Guide." Texas Instruments Application Report,* SPRU187I, April, 2000.

[DSPS01] http:// dspvillage.ti.com, 2001

[ETSI98] ETSI TIPHON Project. in progress documents: http://www.etsi.fr/tiphon, 1998.

[ETSI99] ETSI TR 101 327, V1.1.1. "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) - Guide to numbering options for public networks based on VoIP technology." *ETSI*, , November, 1999.

[ETSI00] ETSI TR 121 978. "Universal Mobile Telecommunications System (UMTS);Feasibility Technical Report - CAMEL Control of VoIP Services ." *ETSI*, , V3.0.0, Jun, 2000.

[FZ00] X. Fu, Z. Zhang. "TMS320C6000 multichannel vocoder technology demonstration kit host side design." *Texas Instruments Application Report*, SPRA558B, 2000.

[FU00] X. Fu. "TMS320C6000 multichannel vocoder technology demonstration kit target side design." *Texas Instruments Application Report*, SPRA560B, Fev., 2000.

[GMF01] B. W. Gillespie, H. S. Malvar, D. A. F. Florêncio. "Speech dereverberation via maximum-kurtosis subband adaptive filtering". *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Salt Lake City, Utah, maio 2001.

[HAA10] A Haar. "Zur Theorie der orthogonalen Funktionensystems." *Mathematische Annalen*, 69:331-371, 1910.

[HAY99] S. Haykin. *Neural Networks A Comprehensive Foundation*. Prentice Hall, New Jersey, 1999.

[HELL97] H. Hellendoorn. "After the fuzzy wave reached Europe." *European Journal of Operation Research 99*, pags. 58-71, 1997.

[HOW99] K. Howell. "Next Generation Internet*." Presentation at PublicTechnology, Inc. 21st Century Annual Member Conference*, http://www.ccic.gov/nci, April, 1999.

[HOR89] K. Hornik et. al. "Multilayer feedforward networks are universal approximators." *Neural Networks*, 2, pags. 359-366, 1989.

[HORT98] I. Horton. *Beginning with Visual C++ 6*. Wrox Press Inc., 1998.

[HSB00] A. Hussain; S. A. Samad,; L. Ban Fah. "Endpoint detection of speech signal using neural network." *Proceedings TENCON 2000*, vol. 1, pgs. 271-274, 2000.

[HUBB98] B. B. Hubbard. *The world according to wavelets.* Wellesley, A. K. Peters, Massachusetts, 1998.

[HUB00] B. Huber. "How to Write Multiplies Correctly in C Code." *Texas Instruments Application Report*, SPRA683, June, 2000.

[HVH99] M. Hamdi, O. Versheure, J. P. Hubaux. "Voice service interworking for PSTN and IP networks." *IEEE Communications Magazine*, pp.104-111, maio, 1999.

[IETF01] http://www.ietf.org/html.charters/mpls-charter.html.

[IEC00] IEC. "H.323 tutorial." http://www.iec.org/tutorials/h323/, 2000.

[IL90] H. S. Isabelle, J. S. Lim. "On modulated filter banks for image coding applications." *EEE Digital Signal Processing Workshop*, New Paltz, NY, Set. 1990.

[IMTC98] IMTC-Voice over IP working group, http://www.imtc.org, 1998.

[ITU98a] ITU-T Rec. H323. "Packet-based multimidea communications systems." 1998.

[ITU98b] ITU-T Rec. H.225.0. "Call signiling protocols and media stream packetization for packet based multimedia communication systems." 1998.

[ITU98c] ITU-T Rec. H.245. "Control Protocol for multimedia communication." 1998.

[ITU98d] ITU-T Rec H.450.1. "Generic functional protocol for the support of suplementary services in H.323." 1998.**[ITU93]** ITU-T Rec. Q.1211. "Introduction to intelligent network capability set 1." 1993.

[ITU99] ITU-T Recommendation G.722.1. *Coding at 24 and 32kbit/s for hands-free operation in systems with low frame* loss (*PREPUBLISHED RECOMMENDATION*), 09/1999.

[ITU96] ITU-T Recommendation G.192. A common digital parallel interface for speech standardization activities, Março, 1996.

[JJ95] S. Jordan, H. Jiang. "Connection establishment in high-speed networks." *IEEE Journal on Selected Areas in Communications*, vol. 13, nº 7, pgs. 1150-1161, 1995.

[JRRLF01] C. Junqueira, M. V. Ribeiro, J.M.T. Romano, C. Lima, J. B. D. Filho. "A Hybrid Algorithm Solution for GPS Antenna Arrays." *International Symposium on Kinematic Systems in Geodesy, Geomatics and Navigation (KIS2001)*, June, 2001.

[KEG96] D. Kegel. "About ISDN." http://www.alumni.caltech.edu/~dank Dez., 1996.

[KGB00] B. Kornmeier, D. Gehrke. "TPS330x Supervising DSP and Processor Applications, AAP Power Management", *Texas Instruments Application Report,* SLVA056A, agosto, 2000.

[KZ95] E. Knightly, H. Zhang. "Traffic characterization ans switch utilization using deterministic bounding interval dependent traffic models." *Proceedings of IEEE INFOCOM'95*, pgs. 1137-1145, Boston, MA, April, 1995.

[LAP98] R. C. Leinecker, T. Archer, G. Pease. *Visual C++ 6 Bible*. Hungry Minds, Inc., 1998.

[LIN98] D. Lin. "Real-time voice transmissions over the Internet." M.Sc. Thesis, *Dept. of Electrical and Computer Engineering*, Univ. of Illinois, Urbana-Champaign, Dec. 1998.

[LK95] G. Lu., C. Kang. "An efficient communication scheme for media on-demand services with hard QoS guarantees." *Journal of Network and Computer Applications*, vol. 21, nº 1, pgs. 1-15, 1995.

[LIPP87] R. P. Lippman. "An introduction to computing with neural nets." *IEEE ASSP Magazine*, pags. 4-22, April, 1987.

[LU94] G. Lu, et. al.. "Temporal synchronization support for distributed multimedia information systems." *Computer Communications*, vol. 17, nº 12, pgs. 852-862, 1994.

[LU00] G. Lu. "Issues and technologies for supporting multimedia communications over the Internet." *Computer Communications*, vol. 23, pgs. 1323-1335, 2000.

[MAL89] S. G. Mallat. "A theory for multiresolution signal decomposition: the wavelet representation." *IEEE Trans. On Pattern Recognition and Machine Intelligente*, vol. 11, nº 7, pags. 674-693, July, 1989.

[MALV90] H. S. Malvar. "Lapped transforms for efficient transform/subband coding." *IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. 38, pags. 969-978, junho 1990.

[MALV92] H. S. Malvar. *Signal processing with Lapped transforms*, Artech House, Norwood, MA, 1992.

[MALV99a] H. S. Malvar. "A modulated complex lapped transform and its applications to audio processing", *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Phoenix, Arizona, março, 1999.

[MALV99b] H. S. Malvar. "Fast progressive wavalet coding." *Proc. Data Compression Conference*, pp. 336-343, Snowbird, Utah, março, 1999.

[MIT01] MIT Media Laboratory, http://sound.media.mit.edu/mpeg4/audio/sqam/, 2001.

[MM00] J. Min, V. Markandey. "Optimizing JPEG on the TMS320C6211 2-Level Cache DSP." *Texas Instruments Application Report*, SPRA705, Dec., 2000.

[MS89] H. S. Malvar, D. H. Staelin. "The LOT; transform coding without blocking effects." *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pags. 553-559, Abril 1989.

[MT95] M. A. Muschietti, B. Torresani. "Pyramidal algorithms for Littlewood--Paley decompositions." *SIAM Journal on Mathematical Analysis,* vol. 26, nº 4, pp. 925-943, 1995.

[NER99] Y. Nernet, et al.. "A framework for differentiated services." *IETF Internet Draft*: draft-diffserv-framework-02.txt, February, 1999.

[NIK98] Z. Nikolic. "TMS320C6201/6701 DSP Host Port Interface (HPI)Performance." *Texas Instruments Application Report*, SPRA449A, April 1998.

[PEA91] W. A. Pearlman. "Performance bounds for subband coding", *Subband Image Coding* (edição Woods, J. W.), cap. 1, Boston, Kluwer, 1991.

[PIN00] C. D. B. Pinheiro. "Tutorial - Especificação ITU H.323." http://penta2.ufrgs.br/h323/indice.htm, 2000.

[PJB87] J. P. Princen, A. W. Jonhson, A. B. Bradley. "Sub-band/transform coding using filter bank designs based on time domain aliasing cancellation." *IEEE Int. Conf. Acoust., Speech, Signal Processing.* Dallas, pags. 2161-2164, Abril, 1987.

[POS80] J. Postel. "User Datagram Protocol – RFC 0768, STD0006." *IETF*, www.ietf.org, August, 1980.

[POS81] J. Postel. "Transmission Control Protocol – RFC 0793, STD0007." *IETF,* www.ietf.org, September, 1981.

[QH93] Y. Qi, B. R. Hunt. "voiced-unvoiced-silence classification of speech using hybrid features and a network classifier." *IEEE Transections on Speech and Audio Processing*, vol. 1, nº 2, April, 1993.

[RBCDMMR92] M. B. Ruskai, G. Beylkin, R. Coifman, I. Daubechies, S. Mallat, Y. Meyer, L. Raphael. *Wavelets and their applications*. Jones e Bartlett, 1992.

[RDR01] M. V. Ribeiro, C. A. Duque, J. M. T. Romano. "An enhanced data compression method for applications in power quality analysis." *27th Annual Conference of the IEEE Industrial Electronics Society*, Denver, USA, Nov., 2001.

[ROS99] E. C. Rosen. "Multiprotocol label switching architecture." *IETF Internet Draft:*draft-ietf-mpls-arch-06.txt, August, 1999.

[RYA97] J. Ryan. "Voice over IP." *The Technology Guide Series*, . http://www.tecguide.com, 1997.

[SCA00] R. Scales. "ETSI Math Operations in C for the TMS320C62x." *Texas Instruments Application Report*, SPRA617A, Nov., 2000.

[SCA99a] R. Scales. "Loop Partitioning on the TMS320C6x." *Texas Instruments Application Report,* SPRA517, February, 1999.

[SCA99b] R. Scales. "Nested Loop Optimization on the TMS320C6x." ." *Texas Instruments Application Report,* SPRA519, February, 1999.

[SCFJ96] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson. "RTP: A Transport Protocol for Real-Time Applications – RFC 1889." *Audio-Video Transport Working Group - IETF*, www.ietf.org, January, 1996.

[SH97] S. Shlien. "The modulated Lapped transform, its time-varying forms, and to audio coding." *IEEE Trans. Speech Audio Processing*, vol. 5, pp. 359-366, Julho 1997.

[SN96] G. Strang, T. Nguyen. *Wavelets and Filter Banks.* Wellesley- Cambridge Press, 1996.

[SR99] H. Schulzrine, J. Rosenberg. "Tutorial: the IETF Internet telephony architecture and protocols." http://computer.org/internet/telephony ,1999.

[SWE97] L. Sweet. "Toss your dimes-Internet video phones have arrived." *ZD Internet Magazine*, August, 1997. http://www.zdnet.com/products/content/zdim/0208/zdim0010.html.

[TANE96] A. S. Tanenbaum. *Computer Networks.* Prentice Hall, 1996.

[TEO98] A. Teolis. Computational Signal Processing with Wavelets. Birkhauser, 1998.

[TD93] T. Birus, D. Syiek, "Optimizing Ada Compilers for DSP Avionics Applications," DSP Applications, April 1993.

[TI01] http://focus.ti.com/docs/tool/toolfolder.jhtml?PartNumber=C3X-ADA-XX.

[TJ00] G. Thomsen, Y. Jani. "Internet telephony: going like crazy." *IEEE Spectrum*, pp-52-58, maio, 2000.

[TMS00a] TMS320C55x. "Technical Overview." *Texas Instruments Application Report*, SPRU393, fevereiro, 2000.

[TMS00b] TMS320C6211, http://focus.ti.com/docs/tool/toolfolder.jhtml, 2000.

[TU97] L. Tsoukalas, R. Uhrig. *Fuzzy and Neural Approaches in Engineering*. Wiley-Interscience, New York, 1997.

[VAI93] P. P. Vaidyanattan. *Multirate systems and filter banks.* Prentice Hall Signal Processing Series, Englewood Cliffs, New Jersey, 1993.

[VK95] M. Vitterli, J. Kovacevic. *Wavelets and Subband Coding*. Prentice Hall, Englewood Cliffs, New Jersey, 1995.

[VLSI99] VLSI Solution OY. The VS SRC Sample Rate Converter, http://www.mathtools.net/Applications/DSP/Converters/, 1999.

[WL99] B. W. Wah, D. Lin. "Transformation-based Reconstruction for Audio Transmissions over the Internet." *IEEE Trans. On Multimedia*, 1(4): 342-351, Dec, 1999.

[WLIN01] G.-D. Wu, C.-T. Lin. "A recurrent neural fuzzy network for word boundary detection in variable noise-level environments.' *IEEE Transections on Systems, Man, and Cybernetics – Part B: Cybernetics*, vol. 31, nº 1, Fev., 2001.

[WIT00] W. E. Witowsky. "IP telephone design and implementation issues – white paper." *Technical Officer Telogy Networks Inc*., http://www.telogy.com, 2000.

[WLS00] B. W. Wah, D. Lin, X. Su. "Streaming real-time audio and video with transformation-based error concealment and reconstruction." *Proc. First Int. Conference on Web Information Systems Engineering*, June, 2000.

[WSL00] B. W. Wah, X. Su, D. Lin. "A Survey of error-concealmente schemes for real-time Audio and Video Transmissions over the Internet." *IEEE Int. Symposium on Multimedia Software Engineering*, Dec., 2000.

A. Apêndice

A.1 Transformada Wavelet (*Wavelet Transform* – WT)

A transformada de wavelet tem se tornado uma poderosa ferramenta de análise em diversas áreas, como resolução de equações diferencias [BGG98], compressão de sinais [RDR01, DRBA00 e VK95], estimação espectral [BGG98 e VK95], análise de sinais não estacionários [SN98], etc. O fato desta ferramenta matemática ser tão utilizada está diretamente relacionada à sua generalização, da mesma forma que as funções de Fourier [SN96].

A origem do nome wavelet é devida a expressão *small wave* [BGG98].

Basicamente a transformada de wavelet permite um mapeamento bidimensional no tempo e freqüência, com alta resolução temporal para as componentes de curta duração e alta resolução espectral para as componentes de longa duração, ou seja, a representação das componentes do sinal são mapeadas na freqüência e no tempo com maior precisão do que observado com o uso da DFT e DSTFT, pois o mapeamento pode ter diferentes configurações afim de ressaltar da melhor forma possível as informações espectrais e temporais, conforme é mostrado na figura A.1.

Um dado interessante é que a partir de manipulações matemáticas das equações da transformada de Wavelet, a transformada de Fourier nas versões contínuas e discretas podem ser obtida, ou seja, a transformada de Fourier pode ser considerado como um caso particular da Transformada de Wavelet [BGG98].

De um modo geral, as wavelets padrões são baseada numa única função chamada de função de geração ou *mother wavelet*, $\psi(t)$, a qual tem algumas restrições especiais. Dentre as restrições ou características especiais, pode-se ressaltar a propriedade de oscilação e decaimento, preferenci1almente rápido. A propriedade de oscilar é expressada

matematicamente pela equação A.1. Esta é uma propriedade dependente da ortogonalidade das funções wavelets e das funções geradas.

$$\int_{-\infty}^{\infty} \psi(t)dt = 0 \qquad \text{(A.1)}$$

Algumas wavelet são, no seu projeto, consideradas como zero além de um intervalo considerado (*compactly supported),* permitindo um decaimento mais acentuado. As wavelets podem ainda ter a propriedade de decaimento rápido no domínio de Fourier, embora existe uma relação entre os decaimentos no tempo e freqüência determinado pelo princípio da incerteza de Heisenberg.

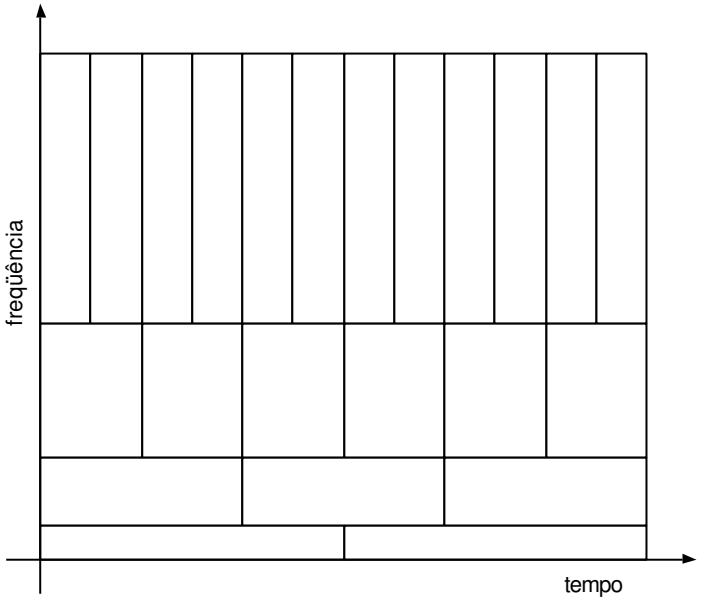


Figura A.1. Grid de Mapeamento de um sinal submetido a TW.

Outra importante restrição é relacionada aos *m vanishing moments*, representado matematicamente pela equação (A.2). Esta restrição sobre a definição de *mother wavelet* permite a representação de sinais com regiões suaves, semelhantes a polinômios de baixa ordem, com muitos coeficientes wavelet iguais a zero, ou seja, funções suaves podem ser representadas de maneira extremamente eficiente usando wavelets.

$$\int_{-\infty}^{\infty} t^l \psi(t)dt = 0, \;\; l = 0,1,2,...,m-1 \qquad \text{(A.2)}$$

As wavelets são na verdade os estreitamentos ou alongamentos e translações de uma *mother wavelet.* Considerando-se a transformada de wavelet contínua as alterações na mother wavelet são descritas matematicamente pelos parâmetros a e b reais na equação

A.3, ou seja, a wavelet $\psi_{(a,b)}(t)$ é a mother wavelet dilatada pelo parâmetro a e transladada para a posição $a^{-1}b$.

$$\psi_{(a,b)}(t) = a^{1/2}\psi(at - b) \tag{A.3}$$

Para um grande número de aplicações é assumida a seguinte restrição nos parâmetros a e b,

$$\begin{aligned} a &= 2^j, \;\; j = 0,1,2,...,J-1 \\ b &= k, \;\; k = 0,1,2,...,M-1 \end{aligned} \tag{A.4}$$

Para uma específica escolha da função *mother wavelet*, a coleção de $\{\psi_{j,k}(t)\}$, com j e k definidos pela equação (A.4), pode assumir a condição ortonormalidade, podendo, então, representar funções, da mesma forma que a série de Fourier ou uma série polinomial ortogonal, desde que, a função $f(t) \subset L^2(R)$. Matematicamente, tem-se

$$f(t) = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} d_{j,k}\psi_{j,k}(t) \tag{A.5}$$

ou seja, $f(t)$ pode ser representado em termos da combinação linear de diferentes funções wavelets. Os coeficientes de expansão, $d_{j,k}$, são obtidos a partir da equação (A.6), devido a consideração de ortogonalidade das wavelets especificadas. A existência de wavelet não ortogonais têm aplicações restritas. Por exemplo, as wavelets biortogonais têm como principal aplicação a compressão de imagens [MALV99b].

$$d_{j,k} = \left\langle f(t), \psi_{j,k}(t) \right\rangle = \int f(t)\psi_{j,k}(t)d \tag{A.6}$$

A propriedade de representação esparsa, ou seja, capacidade de representar funções que apresentam ou não descontinuidades com alguns poucos coeficientes, diferentemente da Fourier e bases polinomias, torna a wavelet uma ferramenta largamente utilizada em compressão de imagens [VK95], de áudio, de sinais aplicado a qualidade de energia [RDR01] e etc.

Para se obter a representação multiresolucional da wavelet [BGG98, MAL89b], dois tipos de funções são necessárias, sendo a primeira a funções de wavelet e a segunda chamada de função *scaling* [BGG98], $\varphi(t)$, tornando possível a representação de $f(t)$ pela equação (A.7).

$$f(t) = \sum_{k=-\infty}^{\infty} c_k \varphi_{0,k}(t) \sum_{j=0}^{\infty} \sum_{k=-\infty}^{\infty} d_{j,k} \psi_{j,k}(t) \tag{A.7}$$

sendo c_k obtido a partir da equação (A.8), dada a seguir

$$c_k = \left\langle f(t), \varphi_{0,k}(t) \right\rangle = \int_{-\infty}^{\infty} f(t) \varphi_{0,k}(t) dt \tag{A.8}$$

Para executar a transformada wavelet diferentes famílias de funções wavelet tem sido propostas tendo como restrição o decaimento rápido nos domínios espectral e temporal. Por exemplo, as wavelets Haar [HAA10], apresentadas em 1910, são *compactly supported* no tempo e decaem a taxa de $1/\omega$ no domínio da freqüência, enquanto as wavelets de Littlewood-Paley [MT95] são *compactly supported* na freqüência e decaem a taxa de $1/t$ no tempo. Entre estas duas famílias de wavelets existem outras que contemplam rápidos decaimentos na freqüência e no tempo, concomitantemente. Dentre estas famílias pode-se ressaltar a família de wavelet desenvolvidas por Ingrid Daubechies [DAU92], popularizada pela suas características intrínsecas que possibilitam a análise de diferentes tipos de sinais.

Maiores detalhes sobre a transformada de wavelet pode ser encontrado em [BGG98, VAI93, HUBB98, DAU92, VK95, CHU97 e TEO98].

Um dos aspectos importantes da transformada de wavelet é seu cálculo rápido executado por algoritmos rápidos. Por exemplo, a transformada Wavelet discreta de (*Discrete Wavelet Transform* – DWT) com decimação *dyadic*, tem complexidade computacional proporcional a $O(N)$, sendo N o comprimento do filtro utilizado, enquanto a *Packet Wavelet Transform e a Overcomplete Wavelet transform* têm complexidade computacional proporcional a $O(N \log N)$ [BGG98 e TEO98], ou seja, a DWT com maior complexidade computaticional é comparável aos algoritmos rápidos da Transformada de Fourier.

A análise multiresolucional que proporciona a utilização de bancos de filtros em algoritmos rápidos da DWT é devida, primeiramente e principalmente, às pesquisas de Stephane Mallat e Yves Meyer [RBCDMMR92].

A análise de sinais a partir da DWT baseada em banco de filtros é como segue.

Seja a função *scaling* [BGG98], $\varphi(t)$, definida na equação (A.9) e considerando-se a equação (A.3), tem-se

$$\varphi(t) = \sum_n h(n)\sqrt{2}\varphi(2t-n) \tag{A.9}$$

e a função $\psi(t)$, dada por

$$\psi(t) = \sum_n h_1(n)\sqrt{2}\psi(2t-n) \tag{A.10}$$

Com a seguinte condição de ortogonalidade

$$h_1(n) = (-1)^n h(1-n) \tag{A.11}$$

Assumindo que uma única solução existe para a equação (A.9), a translação e ponderação da variável t é obtida pela seguinte equação

$$\varphi(2^j t-k) = \sum_n h(n)\sqrt{2}\varphi(2(2^j t-k)-n) = \sum_n h(n)\sqrt{2}\varphi(2^{j+1}t-2k-n) \tag{A.12}$$

fazendo $m = 2k+n$, tem-se

$$\varphi(2^j t-k) = \sum_m h(m-2k)\sqrt{2}\varphi(2^{j+1}t-m) \tag{A.13}$$

Considerando $\nu_j \subset L^2(R)$ definido por

$$\nu_j = \underset{k}{Span}\left\{2^{j/2}\varphi(2^j t-k)\right\} \tag{A.14}$$

tem-se

$$f(t) \in \nu_{j+1} \Rightarrow f(t) = \sum_k c_{j+1}(k)2^{(j+1)/2}\varphi(2^{j+1}t-k) \tag{A.15}$$

A equação (A.15) expressa a função $f(t)$ somente em termos das funções *scaling*. Para mostrar os detalhes da função $f(t)$, pode-se escrever a seguinte equação

$$f(t) = \sum_k c_j(k)2^{j/2}\varphi(2^j t-k) + \sum_k d_j(k)2^{j/2}\varphi(2^j t-k) \tag{A.16}$$

Sendo que o termo $2^{j/2}$ mantém a norma das funções *scaling* e wavelet com valor unitário para os diferentes valores de j. Considerando-se $\varphi_{j,k}(t)$ e $\psi_{j,k}(t)$ ortonormais, tem-se as seguintes equações para cálculo dos coeficientes $c_j(k)$ e $d_j(k)$.

$$c_j(k) = \left\langle f(t), \varphi_{j,k}(t)\right\rangle = \int f(t)2^{j/2}\varphi(2^j t-k)dt \tag{A.17}$$

substituindo a equação A.13 na equação (A.17), tem-se

$$c_j(k) = \sum_m h(m-2k) \int f(t) 2^{(j+1)/2} \varphi(2^{j+1}t - k) dt \tag{A.18}$$

A integral na equação (A.18) é similar a integral da equação (A.17). Assim, chegamos a seguinte equação

$$c_j(k) = \sum_m h(m-2k) c_{j+1}(m) \tag{A.19}$$

De maneira análoga, tem-se

$$d_j(k) = \sum_m h_1(m-2k) d_{j+1}(m) \tag{A.20}$$

A reconstrução ou síntese da função $f(t)$, a partir dos coeficientes $c_j(k)$ e $d_j(k)$ é obtido com a seguinte equação

$$c_{j+1} = \sum_m c_j(m) h_0(k-2m) + \sum_m d_j(m) h_1(k-2m) \tag{A.21}$$

As figuras A.2 e A.3, apresentam graficamente a análise e síntese de sinais, baseadas na DWT. Para estas figuras considera-se $g_0(n) = h_0(n)$ e $g_1(n) = h_1(n)$.

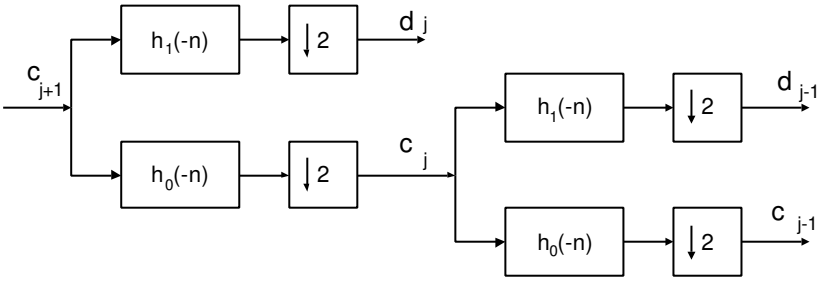


Figura A.2. Banco de Filtro de análise para aplicação da DWT com dois estágios.

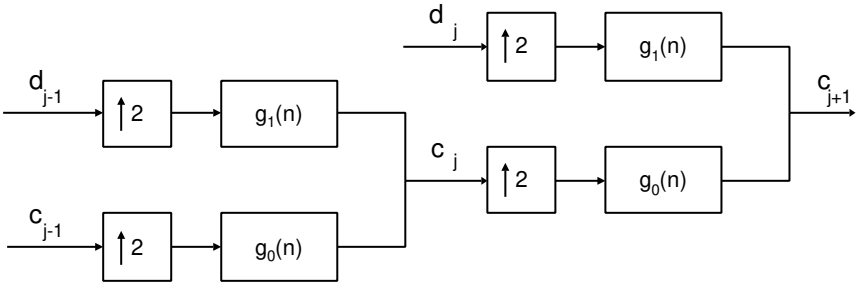


Figura A.3. Banco de Filtro de síntese ou IDWT (Inverse Discrete Wavelet Transform) com dois estágios.

B. Apêndice

B.1 A transformada Lapped Modulada – MLT

A transformada Lapped Modulada (*Modulation Lapped Transform – MLT*) é uma componente da família das transformadas Lapped ortogonais (LOT) [MS89], obtida a partir da modulação senoidal de protótipos de filtros passa-baixa.

Esta transformada é similar a transformada Lapped ortogonal (LOT) no sentido de que as funções de bases tem comprimento igual a $L = 2M$, sendo M o número de sub-bandas do banco de filtro [PJB87].

A principal vantagem da transformada MLT reside no fato de que a mesma pode ser calculada de maneira eficiente, possibilitando um custo computacional compatível às tradicionais transformadas DCT, DCT IV, DST e com melhor desempenho [MALV92]. Além disso, esta transformada é tida como uma ferramenta matemática eficiente na decomposição de sinais no domínio da freqüência e processamento de sinais em sub-bandas e, devido as suas propriedades de reconstrução perfeita, sem *efeitos de blocagem* e quase sempre ótima performance na codificação por transformada para uma grande variedade de sinais. É usada nos mais modernos sistemas de codificação de áudio e voz, tais como, Dolby AC-3, MPEG-2 Layer III [SH97].

Uma desvantagem da MLT é que os seus coeficientes são reais e, a partir deles não é possível obter as informações de fases presentes em vários sinais. Por exemplo, em aplicações de redução de ruído via subtração espectral e cancelamento de *echo* acústico fazem-se necessárias as informações de componentes complexas do sinal [MALV99a], [GMF01].

A descrição sucinta desta transformada será discutida através da demonstração de sua obtenção e cálculo rápido nas seções B.1.1 e B.1.2, respectivamente.

B.2 Definição da Transformada MLT

A partir da consideração do comprimento do filtro igual a $L = 2M$, ou seja, duas vezes o número de sub-bandas, o banco de filtros baseado no critério TDAC [PJB87] (*time-domain aliasing cancellation*) passa a ter a propriedade da reconstrução perfeita.

Em [MALV90] foi reconhecido que o banco de filtro *oddly-stacked* corresponde a transformada Lapped (LT) [MALV92], visto que todos os bancos de filtro com reconstrução perfeita e os filtros de análise e síntese idênticos (com *time reversal*) formam uma transformada Lapped. Este banco de filtros *oddly-stacked* representa matematicamente a Transformada Lapped Modulada (Modulation Lapped Transform - MLT), que satisfaz as mesmas condições de ortogonalidade observadas na transformada LOT.

As funções de bases de uma transformada MLT são as respostas impulsivas do correspondente banco de filtro de síntese e definidas por

$$p_{n,k} = f_k(n) = h(n)\cos\left[\left(k+\frac{1}{2}\right)\left(n-\frac{L-1}{2}\right)\frac{\pi}{M}+\phi_k\right] \tag{B.1}$$

para $k = 0,1,...,M-1$ e $n = 0,1,...,L-1$. De forma usual, M é o número de sub-bandas (o número de coeficientes da transformada) e L é o comprimento do filtro. Fazendo $L = NM$, a escolha das fases da função de modulação que permite a reconstrução perfeita é

$$\phi_k = \left(k+\frac{1}{2}\right)(N+1)\frac{\pi}{2} \tag{B.2}$$

As fases em [PJB87] não são iguais as da equação B.2, pois diferem de π nos ângulos. No entanto, tal diferença é irrelevante, já que não é afetada a forma das funções de base.

A partir da definição de ϕ_k e $N = 2$ na equação B.2, as funções de base da transformada MLT podem ser escritas de acordo coma equação 3 abaixo.

$$p_{n,k} = h(n)\sqrt{\frac{2}{M}}\cos\left[\left(n+\frac{M+1}{2}\right)\left(k+\frac{1}{2}\right)\frac{\pi}{M}\right],\ k = 0,1,...,M-1\ \ n = 0,1,...,2M-1 \tag{B.3}$$

A introdução do fator $\sqrt{\frac{2}{M}}$ na equação B.3 é para normalização das funções de base. Em [PJB87] foi demonstrado que se o protótipo do filtro passa-baixa $h(n)$ contempla a restrição esboçada na equação B.4 os bancos de filtros definidos como na equação B.3 têm reconstrução perfeita.

$$h(L-1-n) = h(n) \tag{B.4}$$

Tal restrição impõe a condição de simetria para a janela e

$$h(n)^2 + h(n+M)^2 = 1 \tag{B.5}$$

Uma possível escolha para a janela é dado em [IL90] e mostrada na equação B.6.

$$h(n) = \pm sin\left[\frac{n\pi}{2(M-1)}\right] \tag{B.6}$$

Tal janela satisfaz as condições de reconstrução perfeita, no entanto, o filtro passa baixa da sub-banda não é polifásico normalizado.

A propriedade Normalização polifásica [PEA91] diz que um sinal DC pode ser perfeitamente reconstruído com somente a sub-banda passa-baixa. Por exemplo, se

$x(n) = 1$

então

$$|X_k(m)| = \begin{Bmatrix} 1/\sqrt{M}, & k = 0 \\ 0, & k = 1,2,...,M-1 \end{Bmatrix}$$

Esta propriedade é importante em codificação de imagem, pois a falta de normalização polifásica gera artefatos na reconstrução da imagem [PEA91].

Em [PEA91] foi mostrado que a única janela que satisfaz ambas normalização polifásica e reconstrução perfeita é dada pela equação B.7.

$$h(n) = \pm sin\left[\left(n+\frac{1}{2}\right)\frac{\pi}{2M}\right] \tag{B.7}$$

A escolha do sinal na equação B.7 é irrelevante, visto que o mesmo somente determina o sinal da sub-banda passa-baixa. A consideração do sinal negativo acarreta que se o sinal $x(n), n = 0,1,...,2M-1$ é dc e positivo, então, $X(m), m = 0,1,...,L-1$ tem também sinal positivo.

O protótipo do filtro na equação B.7 foi introduzido em [MALV90] ao se demonstrar a reconstrução perfeita do sinal considerando-se a sub-banda passa baixa.

Diferentemente da transformada LOT, as funções de base da transformada MLT não têm simetria, ou seja, as funções de transferência por elas representadas não tem fase linear. Em aplicações, tais como: codificação de voz e filtragem adaptativa, isso não é uma

questão importante. No entanto, quando se está processando sinais de comprimento finito, como em codificação de imagens, esta falta de fase linear pode conduzir a efeitos de borda.

É possível a utilização de algoritmos rápidos para cálculo da mesma, conforme é mostrado na seção seguinte, devido à estrutura de modulação das funções de base da transformada MLT.

B.3 Algoritmo rápido para a MLT

Ao comparar-se a definição da transformada MLT na equação B.3 com a definição da transformada DCT-IV, pode-se observar que os cosenos modulados das duas transformadas são os mesmos. Desta forma, o desenvolvimento matemático a seguir mostra como implementar a MLT, baseado na DCT-IV.

O valor da k-ésima sub-banda do sinal, ou k-ésimo coeficiente da transformada no m-ésimo bloco é dado por

$$\begin{aligned} X_k(m) &= h_k(n) * x(n)\big|_{n=mM} \\ &= \sum_{r=0}^{2M-1} h_k(r)\, x(mM-r) \\ &= \sum_{r=0}^{2M-1} h_k(2M-1-r)\, x(mM-2M+1+r) \\ &= \sum_{r=0}^{2M-1} f_k(r)\, x(mM-2M+1+r) \end{aligned} \tag{B.8}$$

Definindo a seqüência do bloco de entrada por

$$x_m(r) = x(mM-2M+1+r) \tag{B.9}$$

e usando a definição de ELT [MALV92], temos

$$\begin{aligned} X_k(m) &= \sum_{r=0}^{2M-1} f_k(r)\, x_m(r) \\ &= \sqrt{\frac{2}{M}} \sum_{r=0}^{2M-1} x_m(r) h(r) \cos\left[\left(n+\frac{M+1}{2}\right)\left(k+\frac{1}{2}\right)\frac{\pi}{M}\right] \end{aligned} \tag{B.10}$$

Usando a mudança de variável $r = s + 3M/2$ e definindo

$$g(s) = x_m(s+3M/2)h(s+3M/2),\ \ g(s) = 0 \ \ se \ \ s \notin [-3M/2, M/2-1] \tag{B.11}$$

A equação B.10 pode ser reescrita para

$$X_k(m) = \sqrt{\frac{2}{M}} \sum_{s=-3M/2}^{5M/2-1} g(s) \cos\left[\left(s+\frac{1}{2}\right)\left(k+\frac{1}{2}\right)\frac{\pi}{M}\right] \tag{B.12}$$

Fazendo algumas manipulações algébricas [MALV92], obtém-se a seguinte seqüência

$$X_k(m) = \sum_{v=0}^{M-1} u(v)\phi_k(v) \tag{B.13}$$

sendo

$$u(v) = \sum_{r=0}^{1} (-1)^r g(i+(2r-2)M) - g(-1-i) \tag{B.14}$$

e

$$\phi_k(v) = \sqrt{\frac{2}{M}} \cos\left[\left(s+\frac{1}{2}\right)\left(k+\frac{1}{2}\right)\frac{\pi}{M}\right] \tag{B.15}$$

representando as funções de base da DCT-IV.

A partir da equação B.14, temos

$$u(i+M/2) = g(i-3M/2) - g(-M/2-1-i),\ i = 0,1,...,M/2-1 \tag{B.16}$$

$$u(M/2-1-i) = -g(M/2-1-i) - g(i-M/2),\ i = 0,1,...,M/2-1 \tag{B.17}$$

Usando a equação B.11 e a simetria da janela, as equações B.16 e B.17 podem ser reescritas da seguinte forma

$$u(i+M/2) = x_m(i)h(i) - x_m(M-1-i)h(M-1-i),\ i = 0,1,...,M/2-1 \tag{B.18}$$

$$u(M/2-1-i) = -x_m(2M-1-i)h(i) - x_m(i+M)h(M-1-i),\ i = 0,1,...,M/2-1 \tag{B.19}$$

Seja a definição do operador de atraso para t amostras definido por

$$D_t\{y(n)\} \equiv y(n-t) \tag{B.20}$$

Então, as equações B.18 e B.19 podem ser rescritas para

$$u(i+M/2) = D_M\{x_m(i+M)\}h(i) - D_M\{x_m(2M-1-i)\}h(M-1-i),\ i = 0,1,...,M/2-1$$

(B.21)

$$u(M/2-1-i) = -x_m(2M-1-i)h(i) - x_m(i+M)h(M-1-i),\ i = 0,1,...,M/2-1 \tag{B.22}$$

As equações B.13, B.18 e B.19 conduzem a implementação da MLT mostrado na figuras B.1 e B.2 .

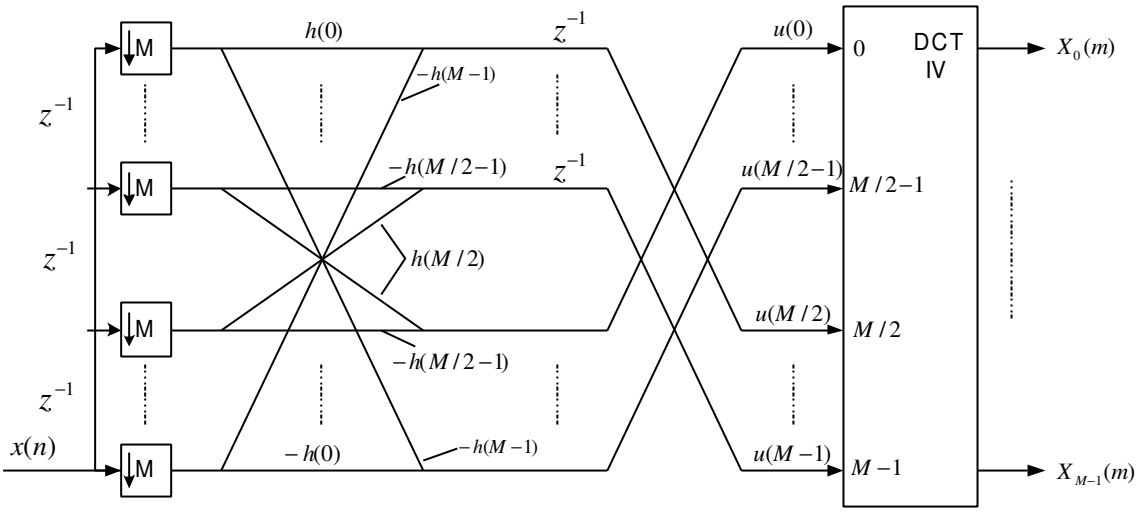


Figura B.1. Estrutura em blocos do algoritmo rápido da Transformada MLT ou banco de filtro de análise.

A complexidade computacional da estrutura da transformada MLT baseada na DCT-IV e com um estágio *butterfly* [MALV92], mostrada graficamente nas figuras B.1 e B.2 , é dada por $M + \log_2 M + 3$ adições e $M + 3\log_2 M + 1$ multiplicações.

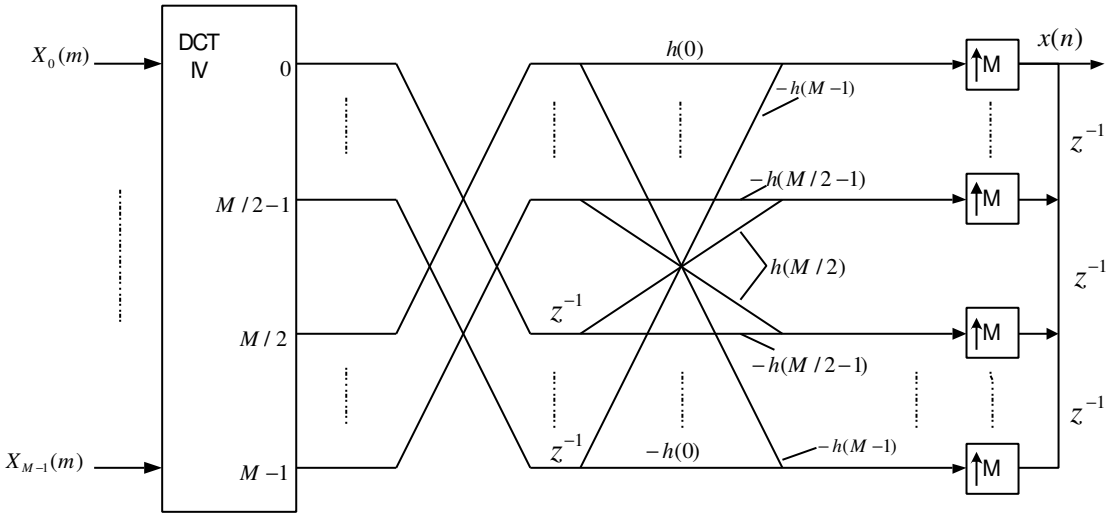


Figura B.2. Estrutura em blocos do algoritmo rápido da Transformada MLT inversa ou banco de filtro de síntese.

C. Apêndice

Norma ITU-T G722.1

A norma ITU-T G722.1 [ITU99] descreve um codec, baseado na forma onda e de baixa complexidade, aplicado a sinais de voz com largura de banda entre 5kHz e 7kHz, operando a taxa de bit de 24kbits/s (480 bits para cada quadro de 20ms) e 32kbit/s (640 bits para cada quadro de 20ms). A entrada digital do codificador pode ser no formato complemento a 2 com 14, 15, 16 bits e taxa de amostragem de 16kHz, conforme é sugerido na recomendação G.722 da ITU-T. Os circuitos de interface digital e analógica para a entrada do codificador e saída do decodificador podem seguir as especificações sugeridas em [ITU96].

O algoritmo de codificação por forma de onda usa a Transformada Lapped Modulada (MLT) [MALV92] aplicada a quadros de 20ms ou 320 amostras de voz. O comprimento da transformada MLT é de 640 amostras e uma sobreposição de 50% (320 amostras) é considerada entre os quadros de amostras, gerando um tamanho de buffer de 20ms. De um modo geral, o atraso total do algoritmo é de 40ms e os demais atrasos são inerentes ao custo computational do codec e aos atrasos de transmissão, através de uma rede PSTN ou pela Internet.

O algoritmo do codec G.722.1 é implementado considerando-se operações matemáticas em ponto fixo e uma implementação em ANSI C como referência é disponibilizado como constituinte da recomendação [ITU99].

Para prover uma descrição clara e suscinta sobre o codec G.722.1, as seção C.1. e C.2 descrevem o funcionamento do codificador e do decodificador, respectivamente.

C.1 Codificador

No processo de codificação, os sinais de voz digitalizados são submetido a MLT em quadros de 640 amostras, sendo 320 amostras do quadro atual e 320 amostras do quadro anterior. O resultado dessa transformação é um quadro de 320 coeficientes MLT.

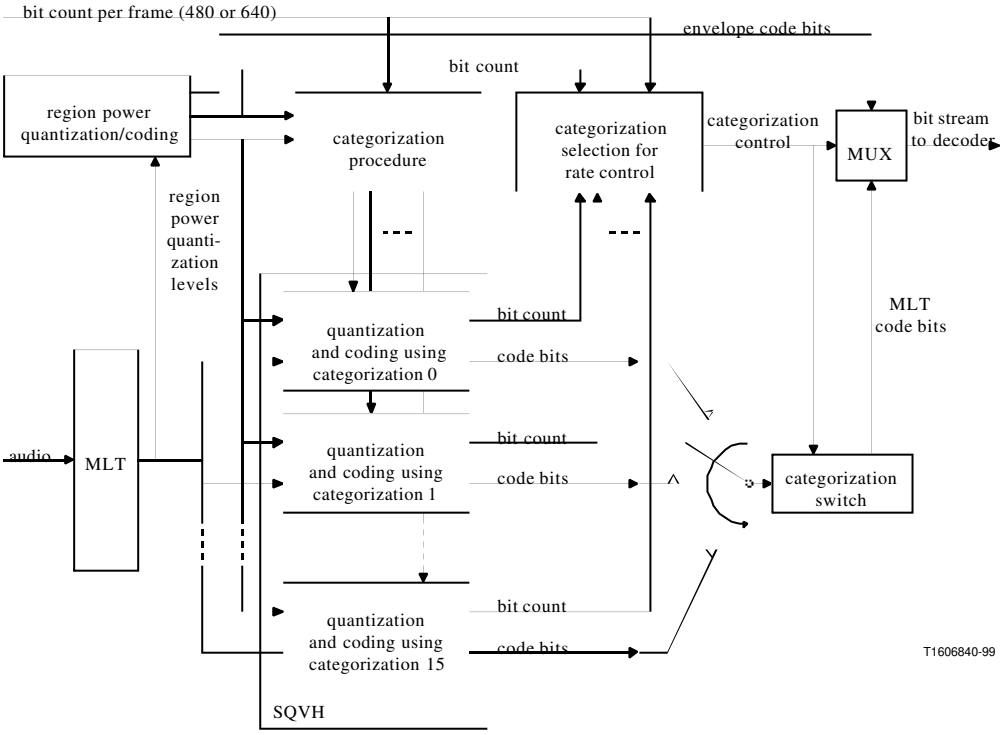


Figura C.1. Diagrama de Blocos do Codificador G.722.1.

Os coeficientes MLT são, então, submetidos a um módulo que calcula o envelope da amplitude para cada conjunto de 20 coeficientes MLT.

O envelope da amplitude é uma representação aproximada do espectro da MLT e fornece o valor *rms* do sinal de voz para cada conjunto de 20 coeficientes MLT.

A seguir, os coeficientes no domínio da transformada MLT são divididos em blocos de 20 amostras, aqui chamados de regiões. Cada região representa uma largura de banda de 500Hz. O número de regiões considerado na norma é 14, contemplando uma largura de banda do sinal voz até 7kHz. A figura C.2 mostra um exemplo do espectro de freqüência das regiões e os envelopes da amplitude dos coeficientes MLT nestas regiões.

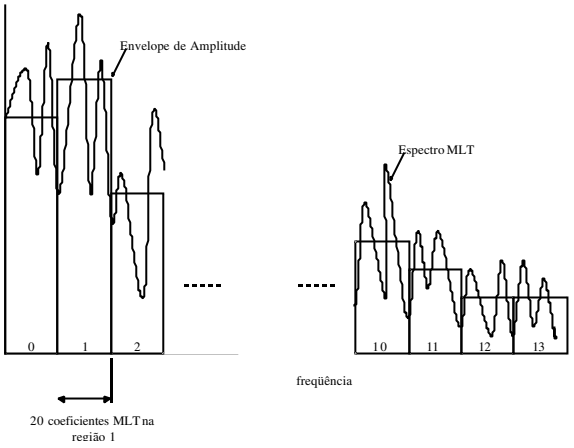


Figura C.2. Exemplo do espectro dos coeficientes MLT e envelopes da amplitude.

A partir dos envelopes de amplitude quantizados e do número de bits restantes no quadro de saída do codificador (para codificação do envelope da amplitude - 4 bits são alocados), o procedimento de categorização é realizado e são geradas 16 categorizações (de 0 a 15) com diferentes números de bits para codificar os coeficientes MLT obtidos na transformação.

Essas categorizações consistem em um conjunto de categorias de atribuições para as 14 regiões especificadas. Sendo que cada categoria define um conjunto de quantizações pré-determinadas e parâmetros de codificação para cada uma região.

A partir da escolha de uma das 16 categorizações possíveis, as atribuições de categoria de cada região são obtidas e juntamente com a informação de envelope de sua amplitude são determinados os parâmetros pertinentes à quantização e codificação dos coeficientes MLT em cada uma das regiões r, $0 \le r \le 13$.

Os coeficientes MLT de cada região são primeiramente normalizados pelo envelope da amplitude desta região e submetidos a uma quantização escalar. Os índices obtidos a partir da quantização escalar são combinados num vetor de índices, o qual é codificado com um número variável de bits, usando a codificação de Huffman de comprimento variável.

A adequação da taxa codec G.722.1 é realizada através da seleção de uma das categorizações.

Finalmente, os bits de código representando os coeficientes MLT quantizados, produzidos a partir de uma categorização selecionada, são enviados para o MUX de transmissão, conforme figura C.1.

Para sistematizar a compreensão do codificador, a seção C.1.1 descreve a MLT, a seção C.1.2 comenta sobre o envelope de amplitude, a seção C.1.3 descreve o procedimento de categorização, a seção C.1.4 mostra a codificação vetorial de Huffman de comprimento variável e quantização escalar, a seção C.1.5 apresenta o controle da taxa de bit do G.722.1 e, finalmente, a seção C.1.6 trata da formatação do quadro codificado.

C.1.1 A Transformada Lapped Modulada (Modulated Lapped Transform – MLT)

A MLT implementada na norma G.722.1 tem como entrada as mais recentes 640 amostras de voz, $x(n), 0 \le n \le 639$, onde $x(0)$ é a amostra mais atrasada e os 320 coeficientes na saída da transformada MLT, $mlt(m), \quad 0 \le m \le 319$ são dados por

$$mlt(m) = \sum_{n=0}^{639} \sqrt{\frac{2}{320}} sin\left(\frac{\pi}{640}(n+0.5)\right) \cos\left(\frac{\pi}{320}(n-159.5)(m+0.5)\right) x(n) \qquad \text{(C.1)}$$

A equação C.1 pode ser reescrita utilizando operações de adição, sobreposição e janelamento seguidas pela Transformada Discreta do Coseno IV (DCT-IV) [MALV92], conforme segue:

$$v(n) = w(159-n)x(159-n) + w(160+n)x(160+n), \; 0 \le n \le 159 \qquad \text{(C.2)}$$

$$v(n+160) = w(319-n)x(320+n) - w(n)x(639-n), \; 0 \le n \le 159 \qquad \text{(C.3)}$$

sendo

$$w(n) = sin\left(\frac{\pi}{640}(n+0.5\right), \; 0 \le n \le 319 \qquad \text{(C.4)}$$

Combinando as equações C.2 e C.3 com uma DCT VI, a expressão para a MLT é:

$$mlt(m) = \sum_{n=0}^{639} \sqrt{\frac{2}{320}} \cos\left(\frac{\pi}{320}(n+0.5)(m+0.5)\right) v(n) \qquad \text{(C.5)}$$

C.1.2 Cálculo, Quantização e codificação do envelope de amplitude

Considerando-se o número de regiões igual a 14, temos que a região r inclui os coeficientes MLT no intervalo $[20r, 20r+19], \; 0 \le r \le 13$.

O valor do envelope da amplitude na região r é definido como o valor *RMS (root-mean-square*) dos coeficientes MLT na região e calculado a partir da seguinte equação

$$rms(r) = \sqrt{\frac{1}{20}\sum_{n=0}^{19} mlt(20r+n)mlt(20r+n)} \qquad \text{(C.6)}$$

Os valores $rms(r),\ 0 \le r \le 13$ são, então, submetidos a um quantizador e a saída deste é um índice, $i,\ -8 \le i \le 31$ a ser armazenado no vetor $rms\_index(r),\ 0 \le r \le 13$. Os índices $i,\ -8 \le i \le 31$ são obtidos a partir da equação C.7.

$$2^{\frac{(i+2)}{2}},\ -8 \le i \le 31 \qquad \text{(C.7)}$$

para verificar quais valores podem ser quantizados para $2^{\frac{(i+2)}{2}}$, considerando o intervalo entre $2^{\frac{(i-0.5+2)}{2}}$ e $2^{\frac{(i+0.5+2)}{2}}$ uma medida logarítmica é considerada.

Por exemplo, se $rms(r) = 310$, então, o correspondente nível de quantização é $2^{\frac{(15+2)}{2}}$ ou 362.04 e $rms\_index(r) = 15$, pois $2^{\frac{(-0.5+2)}{2}} = 304.43$.

O valor de $rms\_index(0)$ é restringido ao intervalo entre 1 e 31 e utiliza 5 bits para sua codificação.

As diferenças dos índices das outras 13 regiões são codificados, conforme o algoritmo na figura C.3, e, então, submetidas à codificação de Huffman de comprimento variável para transmissão. Os valores que as diferenças entre os índices podem assumir estão entre -11 e +12, assim sendo um ajuste é realizado para restringir as diferenças neste intervalo.

```
for(r = number_of_regions − 2; r >= 0; r − −)
{
        if (rms_index[r] < rms_index[r +1] − 11)
                rms_index[r] = rms_index[r + 1] − 11;
}
for(r = 1; r < number_of_regions; r + +)
{
        j = rms_index[r] − rms_index[r − 1];
        if ( j < −12)
        {
                j = −12;
                rms_index[r] = rms_index[r − 1] + j;
        }
        differential_rms_index[r] = j;
}
```

Figura C.3. Algoritmo para codificação dos índices.

As diferenças armazenadas no vetor $differential\_rms\_index[r]$, $0 \le r \le 13$ são codificadas de acordo com o código de Huffman de comprimento variável definido na tabela $defferential\_region\_power\_codes[r][j+12]$. O número de bits para cada código de Huffman é obtido a partir da tabela $differential\_region\_power\_bits[r][j+12]$ e transmitidos na ordem de especificação das regiões, considerando sempre que o bit mais significativo é transferido primeiro. As tabelas citadas acima, assim como outras que serão citadas posteriormente, fazem parte da norma G.722.1.

C.1.3 Procedimento de Categorização

O procedimento de categorização determina os *step-size* e outros parâmetros usados na determinação da categoria para cada uma das regiões e, consequentemente, na execução da quantização dos coeficientes MLT.

Existem 8 possíveis categorias numeradas de 0 a 7, cujos números de bits são fornecidos na tabela C.1. Estas categorias são disponíveis para cada uma das 16 categorização calculadas.

As informações que possibilitam a obtenção das 16 categoriações são as seguintes:

- $number\_of\_available\_bits$: o número de bits disponíveis no quadro atual após a contabilização dos bits de controle e do envelope de amplitude;
- $rms\_index()$: conjunto dos valores $rms(r),\ \ 0 \le r \le 13,$ quantizados.

A categoria atribuída para uma região $r,\ \ 0 \le r \le 13,$ determina os parâmetros de quantização e o número de bits necessário à representação dos coeficientes MLT.

O número de bits necessário para as regiões $r, 0 \le r \le 13$ varia de acordo com as estatísticas dos coeficientes MLT.

Finalmente, seleciona-se a melhor caracterização que se ajuste a taxa de transmissão considerada com base no critério descrito no item C.1.5.

Tabela C.1 – Número de bits para cada categoria.

Categoria	expected_bits_table = bits de código por região como uma função da categoria
0	52
1	47
2	43
3	37
4	29
5	22
6	16
7	0

O mesmo procedimento de categorização é empregado no decodificador. Por isso, para garantir a interoperabilidade é importante que diferentes implementações deste procedimento produzam idênticas caracterizações.

C.1.3.1 Ajuste do número de bits disponíveis

Baseados no atual número de bits disponíveis, o cálculo da estimação do número de bits disponíveis é mostrado na figura C.4.

if

$$number\_of\_available\_bits > 320,$$

then

$$estimated\_number\_of\_available\_bits = 320 + ((number\_of\_available - 320) * 5/8)$$

Figura C.4 – Algoritmo para cálculo do número de bit estimado disponível.

C.1.3.2 Cálculo da categorização inicial

Para qualquer *offset* inteiro no intervalo de –32 e 31, a atribuição das categorias é dada por

$$category(r) = \max\{0, \min\{7, (offset - rms\_index(r))/2\}, \quad 0 \le r \le 13 \tag{C.8}$$

O mesmo *offset* é usado em todas as regiões e o número total de bits a ser utilizado na codificação dos coeficientes MLT é dado por

$$\exp ected\_number\_of\_code\_bits = \sum_{r=0}^{13} \exp ected\_bits\_table(category(r)) \tag{C.9}$$

O ajuste no valor do *offset* é calculado até que um valor maior do que o *offset* satisfaça a seguinte relação:

$$\exp ected\_number\_of\_code\_bits \ge estimated\_number\_o\_available\_bits - 32 \tag{C.10}$$

C.1.3.3 Geração das 15 categorias restantes

Uma vez que a categorização inicial tenha sido calculada, as outras 15 categorizações restantes são também calculadas. Para cada uma nova categorização calculada é realizada uma ajustagem na categoria de uma única região em relação a categorização anterior, de forma que cada uma das categorizações difiram de apenas uma mudança.

O procedimento para determinação das 15 categorizações restantes é o seguinte:

Primeiro Passo: Calcular a categorização inicial, a partir da equação abaixo:

$$initial\_categorization(r) = \max\{0, \min\{offset - rms\_index(r)/2\}\}, \quad 0 \le r \le 13 \tag{C.11}$$

Segundo Passo: criar as variáveis temporárias e inicializá-las:

$\max\_category(r) = initial\_categorization(r),\ 0 \le r \le 13$

$\min\_category(r) = initial\_categorization(r),\ 0 \le r \le 13$

$\max\_bits = \exp ected\_of\_code\_bits$

$\min\_bits = \exp ected\_of\_code\_bits$

Terceiro Passo: Para cada uma das 15 categorizações restantes executar o seguinte comparação

if $(\max\_bits + \min\_bits) \le 2 * estimated\_number\_of\_available\_bits$

- Nova categorização é requisitada com maior número de bits.

if $\max\_category(r) > 0,\ 0 \le r \le 13$

- Encontrar a região que minimiza a seguinte função

$$f_{\max}(r) = offset - rms\_index(r) - 2 * \max\_category(r) \tag{C.12}$$

if *diferentes regiões* $\min$ *imizam* $f(r)$

- Considere a região de menor valor de r ou de mais baixa freqüência.
- A categoria desta região no vetor $\max\_categoy(r),\ 0 \le r \le 13$ recebe decremento de valor unitário.

- número de bits da nova categorização é recalculado e o valor de $\max\_bits$ recebe um novo valor.

else

- Uma nova categorização com um número menor de bits é requisitado.

if $\min\_category(r) < 7,\ 0 \le r \le 13$

Encontrar a região que maxmiza a seguinte função:

$$f_{\min}(r) = offset - rms\_index(r) - 2 * \min\_category(r) \tag{C.13}$$

if *diferentes regiões* $\max$ *mizam* $f(r)$

- Considere a região de maior valor de r ou de mais alta freqüência;
- A categoria desta região no vetor $\min\_categoy(r),\ 0 \le r \le 13$ recebe incremento de valor unitário.

- número de bits da nova categorização é recalculado e o valor de min_ *bits* recebe aquele valor.

As 16 categorizações obtidas são ordenadas de acordo com o número de bits de codificação dos coeficientes MLT, de forma que a categorização 0 contém o maior número de bits e a categorização 15 contém o menor número de bits.

C.1.4. Codificação vetorial de Huffman com quantização escalar (SQVH)

Para regiões com atribuição de categoria entre 0 e 6, os coeficientes MLT são separados em sinal e magnitude. Os valores que representam a magnitude são normalizados pelos valores $rms(r),\ 0 \le r \le 13,$ e a seguir submetidos a quantização escalar e, posteriormente, à codificação de Huffman de comprimento variável.

De forma geral, considerando as regiões, $r, 0 \le r \le 13$, o codificador realizar as seguintes tarefas:

Tarefa 1: O codificador normaliza e quantiza os valores absolutos de cada coeficientes MLT, $mlt(20r+i), 0 \le r \le 13\ e\ 0 \le i \le 19$, para produzir o índice de quantização, $k(20r+i),\ 0 \le r \le 13 e\ 0 \le i \le 19$ utilizando a equação C.14.

$$k(20r+i) = \min\{\lfloor (x * abs(mlt(20r+i)) + deadzone\_rounding)\rfloor, k\max\}, \quad 0 \le r \le 13\ e\ 0 \le i \le 19 \qquad \text{(C.14)}$$

Para $x = \dfrac{1}{stepsize * (quantized\_value\_of\_rms(r))}$ e valores de *stepsize*, deadzone_*rounding* e *kmax* dados pela tabela C.2.

Tabela C.2 – Constantes utilizadas pela técnica SQVH *(Scalar Quantized Vector Huffman Coding*)

categoria	stepsize	deadzone_rounding	kmax
0	$2^{-1.5}$	0.3	13
1	$2^{-1.0}$	0.33	9
2	$2^{-.5}$	0.36	6
3	$2^{0.0}$	0.39	4
4	$2^{.5}$	0.42	3
5	$2^{1.0}$	0.45	2
6	$2^{1.5}$	0.5	1

Os índices $k(20r+i),\ 0 \le r \le 13 e\ 0 \le i \le 19$ são combinados num vetor de índices a partir da equação C.15, considerando os valores predeterminados de vpr , cuja dimensão é vd. Os valores de vpr e vd são mostrados na tabela C.3. A figura figura C.5 ilustra este processo

$$vector\_index(n) = \sum_{j=0}^{vd-1} k(n*vd+j)(k\max+1)^{(vd-(j+1))},\ 0 \le n < vpr \qquad \text{(C.15)}$$

onde

n representa o n-*ésimo* vetor na região $r,\ 0 \le r \le 13$.

j é o índice do j-*ésimo* valor de $k()$ num dado vetor de numa dada região.

vd é a dimensão do vetor para uma dada categoria.

vpr é o número de vetores por região para uma dada categoria.

k max é o valor máximo de $k()$ para uma dada categoria.

$u = (k\max+1)^{vd}$ representa o número de valores distintos que um vetor pode ter numa categoria

.

Table C.3. Definição das constantes *vd*, *vpr* e *u.*

categoria	vd	vpr	u
0	2	10	196
1	2	10	100
2	2	10	49
3	4	5	625
4	4	5	256
5	5	4	243
6	5	4	32

O número de bits necessário para representar um vetor de índices, $vector\_index(n)$, para uma dada categoria é fornecido pelas tabelas $mlt\_sqvh\_bitcount\_category\_x[]$, $0 \le x \le 6$. Estas tabelas também fornecem o número de bits das palavras de código contidas nas tabelas $\_svqh\_code\_category\_x[]$, $0 \le x \le 6$. Esta contagem de bit não inclui os bits de sinais.

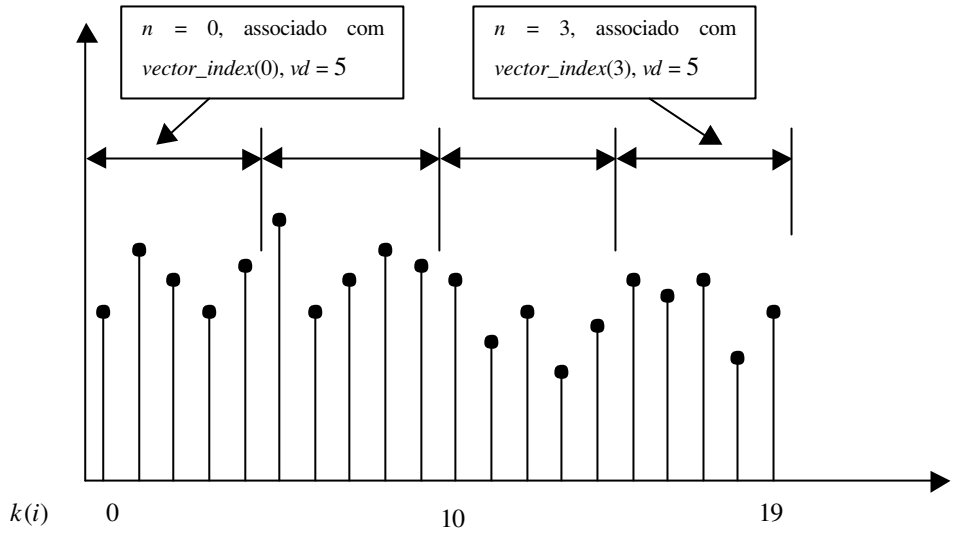


Figura C.5. Exemplo da utilização dos parâmetros vd, vpr e u.

O número de bits, incluindo os bits de sinais, necessário para representar os coeficientes MLT para uma região r, $0 \le r \le 13$, para uma categoria x, $0 \le x \le 6$, é dado por:

$$number\_of\_regions\_bits(r) = \sum_{n=0}^{vpr-1} mlt\_svqh\_bitcount\_category(vector\_index(n)) + (num\_sign\_bits\_n^{th}\_vec) \quad \text{(C.16)}$$

C.1.5. Controle da Taxa de Bit do codificador

O processo de seleção da melhor categorização para a transmissão é realizado da seguinte forma:

- Primeiramente, as categorizações com total de bits em excesso são desconsideradas e a seguir, dentre as categorizações restantes, é escolhida uma com o índice mais baixo. Por exemplo, quando a categorização 0 até 3 usa muitos bits e a categorização 4 contempla as restrições de transmissão, é selecionada a mesma.
- Se nenhuma categorização produz um total de bits que se ajustam dentro das restrições de transmissão. Neste caso a categorização 15 é escolhida.
- Se o número de bits necessários para o codificador representar o quadro de 20ms de sinais de voz é menor do que o número de bits permitido por quadro (480 ou 640), os bits disponíveis não usados são preenchidos com o valor 1 binário.

C.1.6. Transmissão dos índices do vetor de coeficientes MLT no formato de uma seqüência de bits

Os vetores de índices codificados de acordo com as tabelas $mlt\_svqh\_btcount\_category\_x[]$, $0 \le x \le 6$ e $mlt\_svqh\_code\_category\_x[]$, $0 \le x \le 6$ são transmitidos considerando, primeiramente, os que representam as baixas freqüências e, posteriormente, os que representam alta freqüência.

Os bits de sinais relacionados aos coeficientes MLT são transmitidos após cada código do vetor de índice. Para valores positivos o bit de sinal assume o valor 1 binário.

Finalmente, o sinal de voz codificado é armazenado em quadros de 480 e 640 bits para se obter taxa de transmissão de 24kbit/s e 32kbit/s. A figura C.6, mostra o quadro de dados a serem transmitidos.

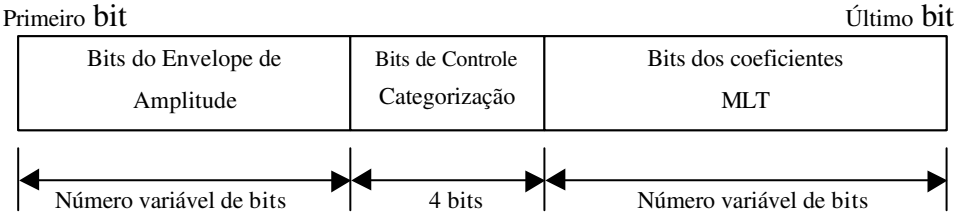


Figura C.6. Formato do Quadro a ser transmitido.

C.2. Decodificador

Os primeiros cinco bits, representando o envelope de amplitude da região 0, são, primeiramente, decodificados e, em seguida os bits correspondentes aos envelopes de amplitude das regiões r, $1 \leq r \leq 13$.

A seguir, os quatro bits de controle de categorização são decodificados para se determinar quais das 16 categorizações disponíveis foi selecionada e transmitida pelo codificador.

De forma análoga ao codificador, o decodificador faz uso do envelope de amplitude associado ao número de bits da seqüência de códigos que representam os coeficientes MLT para calcular as 16 possíveis categorizações. A obtenção dos parâmetros da categorização escolhida pelo codificador é feita considerando-se as 16 possíveis categorizações geradas e a informação de controle de categorização contida na seqüência de códigos.

De posse dos parâmetros (atribuição de categoria das regiões r, $0 \leq r \leq 13$), o decodificador executa a decodificação dos coeficientes MLT.

Se alguma região r, $0 \leq r \leq 13$ recebeu a atribuição da categoria 7 nenhum coeficiente MLT foi transmitido. Para se executar a reconstrução desta região, o decodificador faz uso de uma técnica chamada de *noise-fill.* Os coeficientes MLT que representam freqüências acima de 7kHz são considerados como zero.

Quando as categorias 5 ou 6 são atribuídas para uma região r, $0 \leq r \leq 13$, novamente o decodificador faz uso da técnica *noise-fill*, pois tais categorias têm muitos dos seus coeficientes quantizados para zero.

Após a reconstrução dos coeficientes MLT, a transformada Lapped Modulada inversa é aplicada aos coeficientes MLT e, então, gera-se um conjunto de 320 amostras no domínio do tempo.

Para descrever sucintamente as operações executadas pelo decodificador é apresentado na seção C.2.1 a decodificação do envelope de amplitude, enquanto que a determinação da categorização é descrita na seção C.2.2, a decodificação dos coeficientes MLT é apresentada na seção C.2.3, a técnica *noise-fill* é comentada em C.2.4 e, finalmente, a aplicação IMLT (*Inverse Modulated Lapped Transform*) é resumida na seção C.2.5.

C.2.1. Decodificação do Envelope de Amplitude

A decodificação do envelope de amplitude se dá, primeiramente, a partir da decodificação dos cinco primeiros bits do quadro, os quais correspondem a $rms\_index(0)$. A seguir, para as regiões restantes, os códigos de comprimento variável do vetor $differential\_rms\_index(r),\ 1 \le r \le 13$ são decodificados de acordo com as tabelas $differential\_region\_power\_bits[][]$ e $differential\_region\_power\_codes[][]$ da norma G.722.1.

Os índices para estas regiões $r,\ 1 \le r \le 13$ são reconstruídos a partir da equação (C.17).

$$rms\_index(r) = rms\_index(r-1) + differential\_rms\_index(r), \quad 1 \le r \le 13 \tag{C.17}$$

C.2.2. Determinação da categorização

Após realizada a decodificação do envelope de amplitude, o decodificador determina o número de bits restantes para os coeficientes MLT, utilizando a equação C.18.

$$bits\_available = bits\_per\_frame - amplitude\_envelope\_bits - categorization\_control\_bits \tag{C.18}$$

A seguir, usando o mesmo procedimento de categorização utilizado pelo codificador, o decodificador gera as 16 categorizações possíveis.

C.2.3. Decodificação dos coeficientes MLT

O código de comprimento variável que representa os vetores MLT em cada uma das regiões $r, 0 \le r \le 13$, são decodificados de acordo com as tabelas $mlt\_svqh\_bitcount\_category\_x[], 0 \le x \le 6$ e $mlt\_svqh\_code\_category\_x[], 0 \le x \le 6$. Os índices de quantização dos coeficientes MLT, $k(i)$ são recuperados a partir da equação (C.19).

$$k(i) = \left\lfloor \frac{vector\_index(n)}{(k\max+1)^j} \right\rfloor MOD(k\max+1) \qquad \text{(C.19)}$$

sendo

$\lfloor z \rfloor$ indica o maior valor inteiro menor ou igual a z.

$i = (n+1)vd - j - 1,\ 0 \le j \le vd-1\ \ 0 \le n \le vpr-1$ representa o n-ésimo vetor da região $r, 0 \le r \le 13$.

vd é a dimensão do vetor para uma dada categoria.

k max é o máximo valor de $k()$ para uma dada categoria, conforme a tabela 2.

A reconstrução dos coeficientes MLT é executada pelo uso das tabelas $mlt\_quant\_centroid[][]$ com as amplitudes dos coeficientes MLT reconstruídos pelo cálculo do produto de $rms(r)$ e pelo valor do centróide das tabelas $mlt\_quant\_centroid[][]$.

C.2.4. Técnica noise-fill

Esta técnica consiste em atribuir valores aos coeficientes MLT proporcionais a amplitude média dos coeficientes MLT, $rms(r), 0 \le r \le 13$ não transmitidos pelo codificador (atribuição de categoria 7) para uma dada região $r, 0 \le r \le 13$ considerada. O sinal de cada um destes coeficiente é aleatoriamente especificado utilizando um gerador de número pseudo-aleatório e os valores de proporcionalidade definidos para as categorias 5, 6 e 7 são reunidos a tabela C.4.

Tabela C.4 – Constantes de Proporcionalidade da técnica *noise-fill.*

Categoria	Constante de Proporcionalidade
5	.176777
6	.25
7	.707107

C.2.5. A Transformada Lapped Modulada inversa (Inverse Modulated Lapped Transform – IMLT)

A IMLT implementada na norma G.722.1 é dada pela equação C.20

$$u(n) = \sum_{m=0}^{319} \sqrt{\frac{2}{320}} \cos\left(\frac{\pi}{320}(m+0.5)(n+0.5))mlt(m) \right),\ 0 \le n \le 319 \qquad \text{(C.20)}$$

As operações de janelamento, sobreposição e adição fazem uso de 50% das amostras na saída da DCT do quadro atual com 50% das amostras na saída da DCT no quadro anterior, conforme segue:

$$y(n) = w(n)u(159-n) + w(319-n)u\_old(n),\ 0 \le n \le 159 \qquad \text{(C.21)}$$

$$y(n+160) = w(160+n)u(n) - w(159-n)u\_old(159-n),\ 0 \le n \le 159 \qquad \text{(C.22)}$$

$$w(n) = sin\left(\frac{\pi}{640}(n+0.5) \right),\ 0 \le n \le 319 \qquad \text{(C.23)}$$

A parte de $u(n)$ não utilizada é armazenada como $u\_old()$ para uso no próximo quadro. A equação (C.24) é a representação matemática desta consideração.

$$u\_old(n) = u(n+160),\ 0 \le n \le 159 \qquad \text{(C.24)}$$