COPPE/UFRJ

APLICAÇÕES EM CODIFICAÇÃO DE SINAIS: O CASAMENTO APROXIMADO DE PADRÕES MULTIESCALAS E A CODIFICAÇÃO DISTRIBUÍDA DE ELETROCARDIOGRAMA

Eddie Batista de Lima Filho

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Elétrica.

Orientadores: Eduardo Antônio Barros da Silva
Murilo Bresciani de Carvalho

Rio de Janeiro
Novembro de 2008

APLICAÇÕES EM CODIFICAÇÃO DE SINAIS: O CASAMENTO APROXIMADO DE PADRÕES MULTIESCALAS E A CODIFICAÇÃO DISTRIBUÍDA DE ELETROCARDIOGRAMA

Eddie Batista de Lima Filho

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

Prof. Eduardo Antônio Barros da Silva, Ph.D.

Prof. Murilo Bresciani de Carvalho, D.Sc.

Prof. Sergio Lima Netto, Ph.D.

Prof. Gelson Vieira Mendonça, Ph.D.

Prof. Carla Liberal Pagliari, Ph.D.

Prof. Jacob Scharcanski, Ph.D.

RIO DE JANEIRO, RJ – BRASIL

NOVEMBRO DE 2008

de Lima Filho, Eddie Batista

Aplicações em Codificação de Sinais: o Casamento Aproximado de Padrões Multiescalas e a Codificação Distribuída de Eletrocardiograma/Eddie Batista de Lima Filho. – Rio de Janeiro: UFRJ/COPPE, 2008.

XVI, 190 p.: il.; 29, 7cm.

Orientadores: Eduardo Antônio Barros da Silva

Murilo Bresciani de Carvalho

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2008.

Referências Bibliográficas: p. 172 – 190.

1. Recorrência de Padrões Multiescalas. 2. Quantização Vetorial. 3. Casamento Lateral Generalizado. 4. Contextos Adaptativos. 5. Eletrocardiograma. 6. Pré-processamento de ECG. 7. Eletromiograma. 8. Compressão Distribuída de Sinais. I. da Silva, Eduardo Antônio Barros *et al.*. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

À minha mãe, pelos conselhos sempre contundentes e conversas engraçadas.

À minha noiva, por suportar as minhas ausências.

Agradecimentos

Agradeço ao meu amigo e orientador Eduardo Antônio Barros da Silva, pela orientação sempre honesta, coerente, acertada e direta, ao meu amigo Aguinaldo Silva, pela incrível ajuda que me permitiu iniciar o doutorado, e ao meu amigo Waldir Sabino, por sempre me ajudar nos momentos difíceis.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

APLICAÇÕES EM CODIFICAÇÃO DE SINAIS: O CASAMENTO APROXIMADO DE PADRÕES MULTIESCALAS E A CODIFICAÇÃO DISTRIBUÍDA DE ELETROCARDIOGRAMA

Eddie Batista de Lima Filho

Novembro/2008

Orientadores: Eduardo Antônio Barros da Silva
Murilo Bresciani de Carvalho

Programa: Engenharia Elétrica

Este trabalho aborda dois paradigmas de compressão distintos: o casamento aproximado de padrões multiescalas, representado pelo algoritmo *Multidimensional Multiscale Parser* (MMP), e a Codificação Distribuída de Sinais (*Distributed Source Coding* - DSC). O MMP é um método de codificação que se mostrou eficaz na compressão de imagens em níveis de cinza, apresentando um comportamento universal. Com uma estrutura flexível, esse algoritmo permite a adição de extensões específicas aos sinais a serem codificados, proporcionando um alto grau de adaptabilidade. Devido a isso, apresenta-se uma investigação das capacidades de codificação do MMP, visando o seu desenvolvimento e a sua utilização na compressão de outros tipos de sinais, como o eletrocardiograma (ECG) e o eletromiograma (EMG), identificando as suas características e desenvolvendo e aprimorando ferramentas de codificação. Após essa primeira etapa, a DSC é também investigada. Apesar da literatura atual se concentrar em sinais de vídeo, mostra-se que um outro tipo de sinal, como o ECG, também pode se beneficiar da abordagem distribuída, dando origem a novas classes de codificadores.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

APPLICATIONS IN SOURCE CODING: THE APPROXIMATE MULTISCALE PATTERN MATCHING AND THE DISTRIBUTED ELECTROCARDIOGRAM CODING

Eddie Batista de Lima Filho

November/2008

Advisors: Eduardo Antônio Barros da Silva
Murilo Bresciani de Carvalho

Department: Electrical Engineering

This work tackles two distinct compression paradigms: the approximate multiscale pattern matching, represented by the Multidimensional Multiscale Parser (MMP) algorithm, and Distributed Source Coding (DSC). The MMP is a coding method that proved to be effective in compressing grayscale images, presenting a universal behavior. Having a flexible structure, that algorithm allows the addition of extensions specific to the signals to be encoded, providing a high degree of adaptability. Due to that, an investigation of the coding capabilities of the MMP is presented; it aims at its further development and use for compressing other kinds of signals, such as electrocardiograms (ECG) and electromyograms (EMG), as well as identifying their characteristics and developing and improving coding tools. After this first step, the DSC is also investigated. Despite the fact that most works in the DSC literature concentrate on video signals, it is shown that other kinds of signals, like ECG signals, can also benefit from the distributed approach, giving rise to new classes of encoders.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A compressão de dados é uma das subáreas do processamento de sinais mais exploradas no meio científico, pois permite que aplicações como vídeo, áudio, textos e sinais biológicos, que quase sempre constituem imensas quantidades de informação, sejam transmitidas através de infra-estruturas com larguras de banda bastante reduzidas ou armazenadas de forma eficiente em bancos de dados. Tal feito é possível através da exploração da redundância intrínseca existente nesses conjuntos de dados [8], permitindo ao codificador gerar uma versão compacta dos mesmos, a qual necessita de uma quantidade de bits menor para a sua representação. O decodificador, por sua vez, reconstrói uma versão aproximada ou exata dos dados originais, através do processamento da versão compacta e posterior recuperação da representação inicial, sendo que geralmente está é uma tarefa mais simples e de baixo custo computacional. Essa representação compacta pode ser gerada com base apenas no conjunto de dados atual ou também em outros conjuntos previamente codificados, calculando-se as diferenças e utilizando-se as mesmas na nova descrição, juntamente com a informação auxiliar.

No paradigma de compressão tradicional, a correlação entre os dados é explorada no codificador, que recebe os sinais originais, identifica/processa as estruturas redundantes e armazena/envia a nova representação. Normalmente, essa tarefa é computacionalmente intensa, principalmente para sinais de vídeo, onde há necessidade de se explorar tanto a redundância espacial quanto a temporal [9]. Um representante da classe de codificadores tradicionais é o *Multidimensional Multiscale Parser* (MMP) [10], que consiste em um algoritmo de comportamento universal utilizado

inicialmente para a codificação de imagens.

O MMP é um algoritmo de compressão baseado no casamento aproximado de padrões multiescalas, que codifica segmentos de um dado sinal utilizando versões contraídas e expandidas de padrões armazenados em um dicionário, que é continuamente atualizado durante o próprio processo de codificação. Os segmentos podem ter dimensões variadas, sendo que o casamento é realizado com o auxílio de uma transformação de escala [10–12]. O MMP pode ser estendido para n dimensões e é adaptativo, o que pode ser evidenciado analisando-se o modo como o dicionário é construído. Dadas estas constatações, é possível aplicar o MMP a vários tipos de sinais, desde que as devidas extensões sejam incorporadas. Os seus resultados iniciais com imagens individuais [10, 12], imagens estéreo [13] e resíduos [14] indicam um grande potencial para a compressão de dados, que pode ser explorado de diversas maneiras. Por exemplo, sinais de eletrocardiograma (ECG), que possuem uma estrutura aproximadamente periódica, são candidatos naturais para a compressão com padrões recorrentes, desde que o conhecimento sobre a sua estrutura seja incorporado ao algoritmo base.

Dependendo de qual característica é enfocada, o MMP pode ser visto de três maneiras: como transformador, pois a sua codificação pode ser descrita como uma decomposição em um conjunto de funções-base ortogonais (o dicionário inicial utiliza pulsos de amplitudes variadas, que correspondem à função de escalonamento da *wavelet* de Haar) [15], quantizador, devido ao fato das seqüências de entrada serem aproximadas por outras armazenadas em um dicionário, ou codificador de entropia, dado que as aproximações utilizam "seqüências típicas" [16] retiradas do próprio sinal.

Sendo assim, o MMP pode ser considerado um esquema geral para a compressão de sinais, com adaptações específicas para cada dado a ser codificado. Isto indica que há uma vasta gama de sinais, ainda inexplorados, que podem ter o MMP como um bom codificador. Numa abordagem unificada, haveria um algoritmo base para todas as aplicações e, dependendo do problema em questão, um subconjunto das ferramentas específicas seria empregado. No presente trabalho, duas classes de sinais serão abordadas: imagens em níveis de cinza e sinais biológicos, utilizando-se especificamente sinais de eletrocardiograma e eletromiograma.

Nos esquemas de compressão de dados tradicionais, como já foi mencionado, é fácil perceber que a parte computacionalmente intensa do processo executado pelo CODEC (*i.e.* enCOder/DECoder) está localizada no codificador, o que é extremamente conveniente para transmissões em redes ponto-multiponto, como radiodifusão de sons e imagens (rede de TV comercial). Nesse caso, o elemento de maior complexidade da rede está presente apenas na cabeça de rede (conhecida em inglês como *Headend*) e atende a inúmeros elementos de baixa complexidade e baixo custo, localizados nas dependências do usuário (receptores). Entretanto, em redes multiponto-ponto, como as redes de sensores [17, 18], tal abordagem resultaria em um custo de implantação extremamente elevado, além de um grande consumo de energia por ponto de rede, inviabilizando o projeto.

Nos últimos anos, a compressão distribuída de sinais (*Distributed Source Coding* - DSC) [19–23] tem despertado grande interesse na comunidade científica internacional, o que gerou um número expressivo de publicações em diversos periódicos e congressos [1, 2, 24–65]. Tal fato deve-se principalmente à sua aplicação em redes distribuídas, podendo ocasionar uma reversão no paradigma de compressão tradicional: vários codificadores simples e um decodificador complexo, que explora as dependências estatísticas dos vários feixes de bits transmitidos. Nesse caso, a informação auxiliar, anteriormente explorada apenas no codificador através de um processo computacionalmente complexo, é explorada também no decodificador. O codificador, por sua vez, teria acesso apenas ao conhecimento sobre as estatísticas conjuntas dos dados auxiliar e de referência, comumente chamadas de "canal de correlação". O fundamento da compressão distribuída reside no conceito de *cosets* [66, 67], que divide o espaço de sinal de forma a permitir a reconstrução do dado com informação auxiliar no decodificador.

Os dois paradigmas de compressão apresentados, ou seja, o algoritmo MMP (compressão tradicional baseada em dicionários) e a compressão distribuída de sinais, não são mutuamente exclusivos e têm o potencial para resolver problemas complementares ou distintos na área da compressão de sinais. Sendo assim, é interessante explorar as capacidades de cada um, mostrando novas aplicações e desenvolvendo esquemas práticos de codificação.

Os objetivos do presente trabalho são explorar as características e capacidades

de compressão do MMP, através de uma extensa investigação e posterior aplicação a alguns tipos de dados, e delinear novas aplicações para esquemas de compressão distribuída de sinais, fazendo com que as mesmas se beneficiem das principais características deste novo paradigma. Os resultados das simulações realizadas são comparados com soluções existentes na literatura, visando validar as soluções propostas. Em muitos dos casos apresentados, as soluções obtidas com algoritmos baseados no MMP são competitivas ou chegam mesmo a superar o estado da arte em compressão de imagens e sinais biológicos. Com relação à compressão distribuída, cuja aplicação alvo é a compressão de sinais de ECG, os resultados obtidos não possuem uma referência para comparação. Entretanto, levando-se em consideração os resultados para algoritmos tradicionais, as soluções desenvolvidas mostram-se promissoras (não parece haver uma grande distância com relação à compressão tradicional, como geralmente ocorre), apresentando um grande potencial para desenvolvimentos futuros.

O restante deste trabalho está dividido como segue. O algoritmo MMP está descrito no capítulo 2, apresentando-se a máquina de codificação e o feixe de bits gerado. O problema da compressão distribuída de sinais é abordado no capítulo 3, através de uma descrição das suas principais características. No capítulo 4, são apresentados os esquemas desenvolvidos, com base no MMP, para a compressão de imagens, sinais de Eletrocardiograma (ECG) e sinais de Eletromiograma (EMG), bem como resultados de simulações. Em cada um deles, o algoritmo MMP é extendido com novas técnicas de modelagem ou pré-processamento de sinal, que contribuem para o aumento de desempenho global; no caso do MMP para imagens, assume-se um modelo para a fonte e, na compressão de sinais de ECG, processam-se os batimentos cardíacos diretamente. Os principais códigos corretores de erros empregados na compressão distribuída de sinais são apresentados no capítulo 5, e uma nova aplicação para a compressão distribuída, envolvendo a compressão de sinais de ECG, é introduzida no capítulo 6, onde desenvolvem-se dois compressores distintos: um no domínio da freqüência e outro no domínio da amostra. Finalmente, no capítulo 7, as conclusões do trabalho são expostas e implementações futuras são sugeridas.

Capítulo 2

O algoritmo MMP

O algoritmo MMP é um esquema de compressão de dados com perdas baseado no *casamento aproximado de padrões multiescalas*, que foi recentemente desenvolvido em [10]. No casamento de padrões comum, dois vetores $\mathbf{u}$ e $\mathbf{v}$ de comprimentos iguais podem ser casados se são bastante semelhantes, de acordo com uma dada métrica de proximidade. Por outro lado, no casamento de padrões multiescalas, estes dois sinais podem ter comprimentos diferentes, ou seja, $\ell(\mathbf{u}) \neq \ell(\mathbf{v})$. Para que o casamento ocorra, é preciso mudar os comprimentos dos vetores com uma *transformação de escala* $T_N(\mathbf{x}) : \mathbb{R}^{\ell(\mathbf{x})} \mapsto \mathbb{R}^N$ [11]. O casamento multiescalas está ilustrado na Figura 2.1, onde um vetor $\mathbf{u}$ de comprimento 2 é expandido para o comprimento 5, de modo a ser comparado com um vetor $\mathbf{v}$, também de comprimento 5.

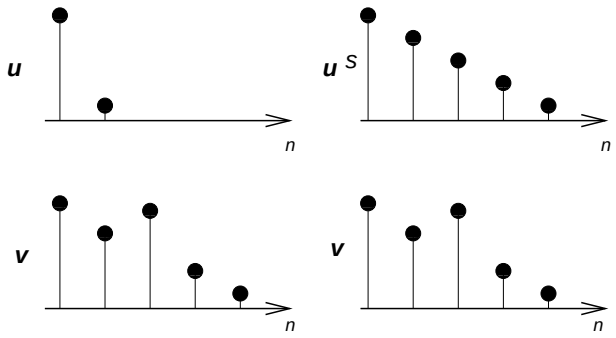


Figura 2.1: O casamento aproximado de padrões multiescalas.

As simulações realizadas mostraram que o MMP é bastante eficaz na compressão de diversos tipos de imagens, especialmente aquelas contendo textos ou gráficos e as compostas por textos, gráficos e outras imagens em níveis de cinza. Nesse último caso, versões anteriores do algoritmo já conseguiam até superar outros codificadores

considerados como o estado da arte em compressão de imagens [10–12]. Uma versão mais recente do MMP, conhecida como MMP-INTRA-FT [68], conseguiu superar o H.264 intra-quadros [69] e o JPEG2000 [70] para todas as imagens testadas.

2.1 O algoritmo MMP bidimensional

No MMP bidimensional [10], existe um dicionário $\mathcal{D} = \{\mathbf{b}_0, \mathbf{b}_1, \ldots, \mathbf{b}_{L-1}\}$ com L elementos $\mathbf{b}_i$ de tamanho variável, que são utilizados para aproximar sub-blocos de tamanho variável de um dado bloco de entrada

$$\mathbf{X}^0 = \begin{pmatrix} x(0,0) & x(0,1) & \ldots & x(0,N_0-1) \\ \vdots & \vdots & \ddots & \vdots \\ x(M_0-1,0) & x(M_0-1,1) & \ldots & x(M_0-1,N_0-1) \end{pmatrix},$$

onde $M_0 = N_0$ são potências de dois. Na tentativa de codificar $\mathbf{X}^0$, o MMP procura, no dicionário $\mathcal{D}$, o melhor bloco $\mathbf{b}_{i_0}$ para substituir $\mathbf{X}^0$, sem se importar com as dimensões de cada $\mathbf{b}_i$. Se um elemento $\mathbf{b}_{i_0}$ adequado é encontrado, o algoritmo gera um *flag* '1', indicando um casamento bem sucedido (o critério de distorção é atendido), e o índice i_0 referente ao elemento escolhido; caso contrário, o bloco de entrada é particionado em dois outros blocos iguais. Se $M_0 > N_0$, as partições são

$$\mathbf{X}^1 = \begin{pmatrix} x(0,0) & \cdots & x(0,N_0-1) \\ \vdots & \ddots & \vdots \\ x(M_0/2-1,0) & \ldots & x(M_0/2-1,N_0-1) \end{pmatrix}$$

e

$$\mathbf{X}^2 = \begin{pmatrix} x(M_0/2,0) & \cdots & x(M_0/2,N_0-1) \\ \vdots & \ddots & \vdots \\ x(M_0-1,0) & \ldots & x(M_0-1,N_0-1) \end{pmatrix}.$$

Senão, se $M_0 \leq N_0$, as partições ficam

$$\mathbf{X}^1 = \begin{pmatrix} x(0,0) & \cdots & x(0,N_0/2-1) \\ \vdots & \ddots & \vdots \\ x(M_0-1,0) & \ldots & x(M_0-1,N_0/2-1) \end{pmatrix}$$

e

$$\mathbf{X}^2 = \begin{pmatrix} x(0, N_0/2) & \cdots & x(0, N_0 - 1) \\ \vdots & \ddots & \vdots \\ x(M_0 - 1, N_0/2) & \ldots & x(M_0 - 1, N_0 - 1) \end{pmatrix}.$$

Quando isto ocorre, o algoritmo gera um *flag* '0', indicando que houve uma divisão do bloco inicial, e repete o procedimento para $\mathbf{X}^1$. Se um bloco semelhante a $\mathbf{X}^1$ é encontrado no dicionário $\mathcal{D}$, o algoritmo gera um *flag* '1' e um índice i_1, passando então a processar $\mathbf{X}^2$; caso contrário, um *flag* '0' é gerado e $\mathbf{X}^1$ é particionado.

Assim, ao tentar codificar $\mathbf{X}^0$, o MMP constrói uma árvore de segmentação binária $\mathcal{S} = \{n_j\}$, na qual cada nó n_j corresponde a um sub-bloco $\mathbf{X}^j$ em $\mathbf{X}^0$. Os nós estão associados aos sub-blocos de acordo com:

i) n_0 corresponde a $\mathbf{X}^0$.

ii) Para cada nó n_j associado a um sub-bloco $\mathbf{X}^j$ de tamanho $M_j \times N_j$, tem-se: se $M_j > N_j$, então n_{2j+1} e n_{2j+2} estão associados a dois sub-blocos de dimensões $M_j/2 \times N_j$, tal que a sua concatenação é igual a $\mathbf{X}^j$. Senão, se $M_j \leq N_j$, então n_{2j+1} e n_{2j+2} estão associados a dois sub-blocos $\mathbf{X}^{2j+1}$ e $\mathbf{X}^{2j+2}$, de tamanho $M_j \times N_j/2$.

iii) Esta segmentação é recursivamente aplicada a todos os sub-blocos, começando no nó n_0, até que a escala de dimensões 1×1 seja alcançada, não sendo mais possível dividir um sub-bloco.

Este processo está ilustrado na Figura 2.2 para $N_0 = M_0 = 4$. Uma das características marcantes do MMP reside no uso de uma árvore de segmentação binária para dados bidimensionais (imagens), quando a maioria dos algoritmos utiliza *quadtrees* [71].

Para que a árvore de segmentação gerada seja ótima em termos de taxa-distorção, a escolha do bloco deve ser baseada na minimização do custo *lagrangeano* [72] $J(n_j)$, definido como

$$D(n_j) = d\left(\mathbf{X}^j, \mathbf{b}_{i_j}^{s_j}\right) \tag{2.1}$$

$$R(n_j) = -\log_2\left(\Pr\left\{\mathbf{b}_{i_j}\right\}\right) \tag{2.2}$$

$$J(n_j) = D(n_j) + \lambda R(n_j) \tag{2.3}$$

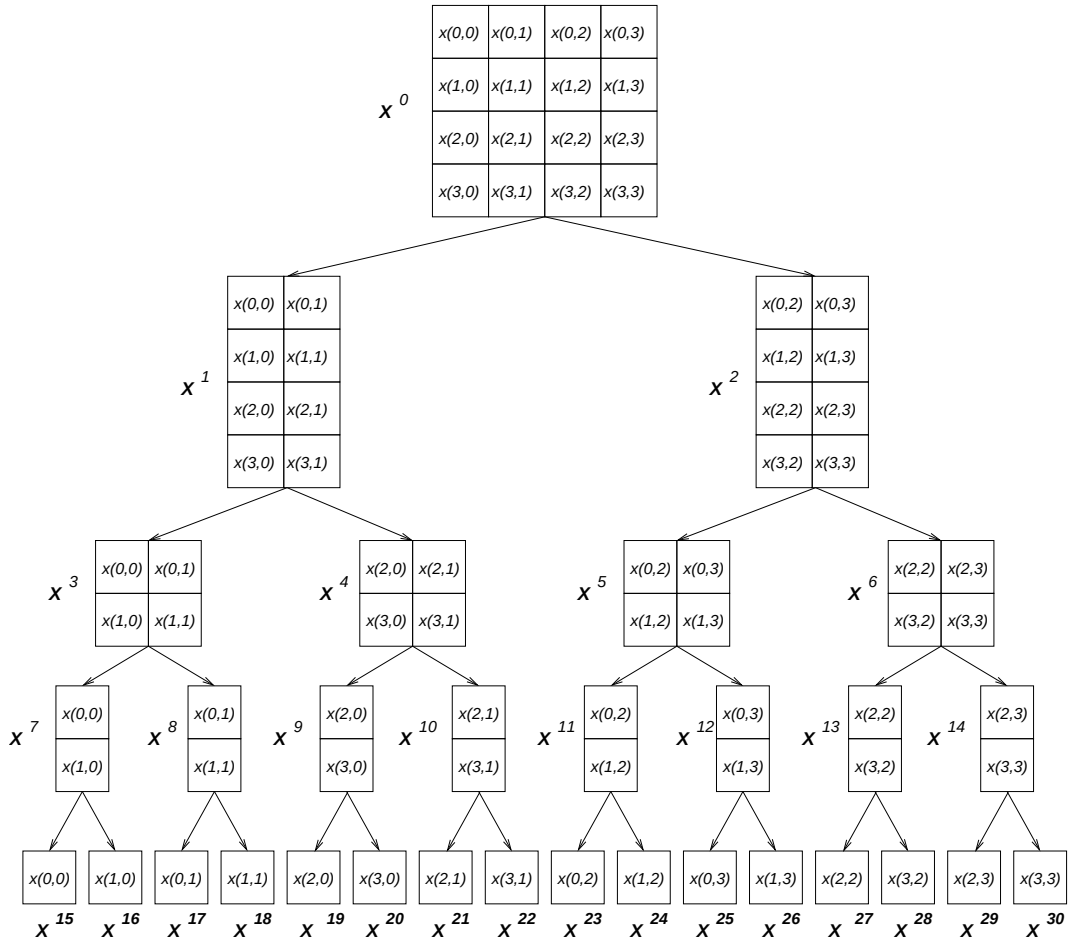


Figura 2.2: A árvore de segmentação de um bloco de entrada $\mathbf{X}^0$.

onde $\mathbf{b}_{i_j}^{s_j}$ é uma versão escalonada de $\mathbf{b}_{i_j}$, $d(\mathbf{u}, \mathbf{b})$ é a métrica de distorção (geralmente o erro quadrático) e $\Pr\left\{\mathbf{b}_{i_j}\right\}$ é a probabilidade do índice i_j ser escolhido para a codificação.

A escolha do melhor vetor para codificar $\mathbf{X}^j$ pode ser realizada através de uma busca exaustiva, onde o dicionário $\mathcal{D}_{s_j}$ é uma cópia de $\mathcal{D}$ com todos os seus vetores contraídos ou expandidos para a escala s_j. Se as dimensões do vetor de entrada são $M_0 \times N_0$, com $M_0 = N_0$, a árvore de segmentação está associada a blocos de dimensões $\{M_0 \times M_0/2, M_0/2 \times M_0/2, M_0/2 \times M_0/4, \ldots, 1 \times 1\}$. Assim, para se evitar a realização da transformação de escala a cada tentativa de casamento, mantém-se $(1 + 2\log_2(M_0))$ cópias do dicionário (estratégia multidicionários), tornando o algoritmo muito mais rápido; a única penalidade reside no uso de memória.

A transformação de escala bidimensional é separável e foi implementada através de operações clássicas de mudança de taxa de amostragem. Inicialmente, todas as linhas são transformadas e depois todas as colunas, utilizando-se o seguinte proce-

dimento:

$$\begin{aligned}
&\mathbf{Y}_i^{K\times N} = T_N\left[\mathbf{X}_i^{K\times L}\right],\ i = 0,1,\ldots,K-1,\\
&\mathbf{Z}_j^{N\times M} = T_M\left[\left(\left(\mathbf{Y}^{K\times N}\right)^T\right)_j\right],\ j = 0,1,\ldots,N-1,\\
&\mathbf{X}_l^{M\times N} = \left(\left(\mathbf{Z}^{N\times M}\right)^T\right)_l,\ l = 0,1,\ldots,M-1.
\end{aligned} \tag{2.4}$$

As transformações componentes foram implementadas como segue. Ao se transformar $\mathbf{v}$, com tamanho N_0, em $\mathbf{v}^S$, com tamanho $N > N_0$, $v^S(n)$ ($n = 0,1,\ldots,N-1$) será dado por

$$\begin{aligned}
m^0(n) &= \left\lfloor \frac{n\,(N_0-1)}{N} \right\rfloor\\
m^1(n) &= \begin{cases} m^0(n)+1, & m^0(n) < N_0-1\\ m^0(n), & m^0(n) = N_0-1 \end{cases}\\
\alpha(n) &= n\,(N_0-1) - N m^0(n),\\
v^s(n) &= \left\lfloor \frac{\alpha(n)\left(v\left(m^1(n)\right) - v\left(m^0(n)\right)\right)}{N} \right\rfloor + v(m^0(n)).
\end{aligned} \tag{2.5}$$

Ao se modificar o tamanho N_0 para $N < N_0$, $v^S(n)$ ($n = 0,1,\ldots,N-1$) será dado por

$$\begin{aligned}
m^0(n,k) &= \left\lfloor \frac{n\,(N_0-1)+k}{N} \right\rfloor,\\
m^0(n,k) &= \begin{cases} m^0(n,k), & m^0(n,k) < N_0,\\ N_0-1, & m^0(n,k) = N_0, \end{cases}\\
m^1(n,k) &= \begin{cases} m^0(n,k)+1, & m^0(n,k) < N_0-1,\\ m^0(n,k), & m^0(n,k) = N_0-1, \end{cases}\\
\alpha(n,k) &= n\,(N_0-1)+k - N m^0(n,k),\\
v^s(n) &= \frac{1}{N_0+1}\sum_{k=0}^{N_0}\left(\left\lfloor \frac{\alpha(n,k)\left(v\left(m^1(n,k)\right) - v\left(m^0(n,k)\right)\right)}{N} \right\rfloor + v\left(m^0(n,k)\right)\right).
\end{aligned} \tag{2.6}$$

Para que o decodificador seja capaz de recuperar os dados, a informação associada aos nós-folha da árvore de segmentação deve ser codificada e adicionada ao arquivo compactado. Com isso, o custo associado à árvore de segmentação completa, conforme ilustrado na Figura 2.2, consiste na soma dos custos de todos os

nós-folha. Entretanto, a árvore de segmentação completa não é necessariamente a melhor instância de partição dos blocos (considerando-se a taxa de codificação final). A Figura 2.3, por exemplo, ilustra uma outra forma da árvore de segmentação, que pode até mesmo apresentar um custo global menor. O custo total para se codificar $\mathbf{X}_0$, utilizando-se uma árvore de segmentação arbitrária $\mathcal{S}$, é calculado como:

$$J(\mathcal{S}) = \sum_{n_j \in \mathcal{S}_\mathcal{L}} D(n_j) + \lambda \left(R_t(\mathcal{S}) + \sum_{n_j \in \mathcal{S}_\mathcal{L}} R(n_j) \right), \tag{2.7}$$

onde $\mathcal{S}_\mathcal{L}$ é o conjunto de nós-folha de $\mathcal{S}$ e $R_t(\mathcal{S})$ é a taxa necessária para se especificar $\mathcal{S}$.

O MMP tenta, então, encontrar a melhor árvore de segmentação através da poda da árvore de segmentação completa, utilizando um algoritmo de busca similar aos encontrados em [71, 73]. Define-se, então, a sub-árvore $\mathcal{S}(n_j)$ como um subconjunto de $\mathcal{S}$ que tem n_j como nó-raiz. Para se encontrar a árvore ótima, se $J(n_l) < J(\mathcal{S}(n_{2l+1})) + J(\mathcal{S}(n_{2l+2}))$, as sub-árvores $\mathcal{S}(n_{2l+1})$ e $\mathcal{S}(n_{2l+2})$ devem ser podadas de $\mathcal{S}$, ocasionando a diminuição do custo total.

No exemplo ilustrado na Figura 2.3, a saída gerada pelo MMP seria $0, 0, 1, i_3, 0, 0, i_{19}, i_{20}, 1, i_{10}, 1, i_2$, que são posteriormente codificados com a ajuda de um codificador aritmético [74]. Como i_{19} e i_{20} localizam-se na última escala possível, o *flag* '1' não é necessário. A seqüência de *flags* 0010011 define completamente a árvore de segmentação escolhida pelo codificador. Como os blocos associados aos indices i_j são conhecidos (tanto o codificador quanto o decodificador têm o mesmo dicionário $\mathcal{D}$), a saída gerada pelo MMP pode ser facilmente decodificada. Com os *flags*, o decodificador é capaz de reconstruir completamente a árvore de segmentação. Dado que a profundidade de cada nó n_j associado ao índice i_j define as dimensões do bloco, o MMP é então capaz de substituir cada nó n_j por uma versão corretamente escalonada de $\mathbf{b}_{i_j}$ no dicionário $\mathcal{D}$.

Uma das características mais importantes do MMP, que também proporciona um aumento no desempenho no algoritmo, é a capacidade do dicionário se adaptar ao dado em codificação. O MMP começa com um dicionário inicial muito simples, que na escala de dimensões 1×1 é composto por inteiros entre os valores mínimo e máximo da imagem, crescendo com o decorrer da codificação; os dicionários em todas as outras escalas são compostos por versões expandidas dos elementos da

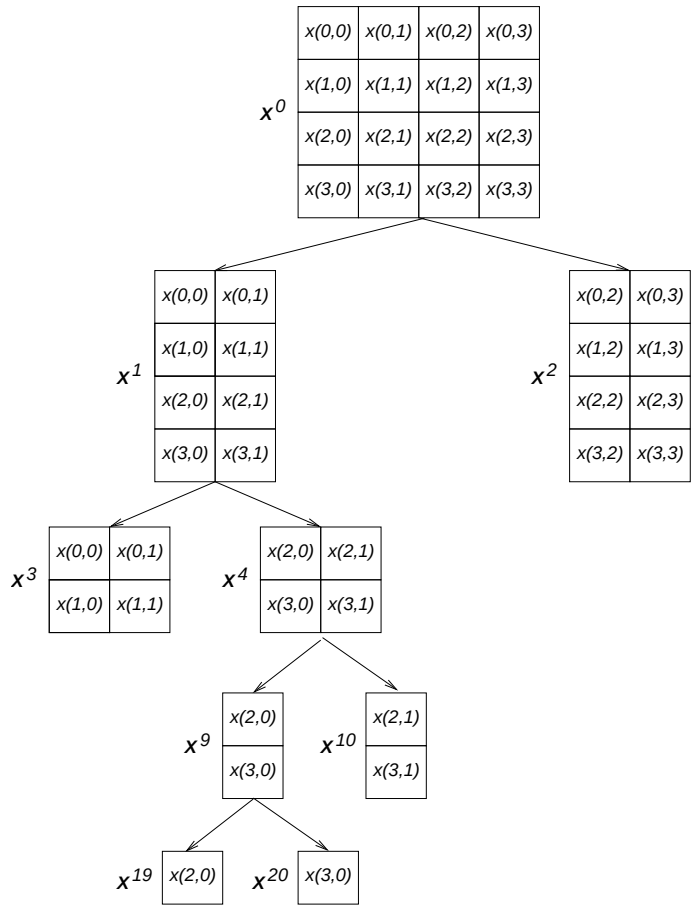


Figura 2.3: A árvore de segmentação podada de um bloco $\mathbf{X}^0$.

escala de dimensões 1×1. A adaptabilidade do dicionário ocorre devido à sua atualização com versões contraídas e expandidas de blocos previamente codificados (seqüências típicas), de uma maneira similar ao que é feito nos esquemas Lempel-Ziv sem perdas [75]. Por exemplo, logo que os índices i_{19} e i_{20}, na Figura 2.3, são determinados, podem-se construir $\hat{\mathbf{X}}^{19}$ e $\hat{\mathbf{X}}^{20}$, que são as versões reconstruídas de $\mathbf{X}^{19}$ e $\mathbf{X}^{20}$, escalonando-se os blocos $\mathbf{b}_{i_{19}}$ e $\mathbf{b}_{i_{20}}$. A versão reconstruída de $\mathbf{X}^9$, formada pela concatenação de $\hat{\mathbf{X}}^{19}$ e $\hat{\mathbf{X}}^{20}$, é determinada como

$$\hat{\mathbf{X}}^9 = \begin{pmatrix} \hat{x}(2,0) \\ \hat{x}(3,0) \end{pmatrix}.$$

$\hat{\mathbf{X}}^9$ pode então ser incluído em $\mathcal{D}$, dado que o mesmo não existe no dicionário e os seus componentes são conhecidos tanto pelo codificador quanto pelo decodificador, o que ocasiona a correta sincronização dos dicionários. Do mesmo modo, a próxima atualização será realizada incluindo-se no dicionário a concatenação de $\hat{\mathbf{X}}^9$ e $\hat{\mathbf{X}}^{10}$, logo que os dois segmentos estejam disponíveis. As demais atualizações são

realizadas da mesma maneira.

Uma cópia do dicionário $\mathcal{D}$ na escala $2^{-p}M_0 \times 2^{-q}N_0$ é definida como $\mathcal{D}_{p+q}$. Como já foi dito, se cópias de $\mathcal{D}$ forem mantidas para todas as escalas, evita-se a realização da transformação de escala ao se procurar um elemento no dicionário. Tal abordagem, conhecida como *estratégia multidicionários*, acelera bastante a busca de aproximações para o bloco de entrada, diminuindo o tempo de codificação.

Se a estratégia multidicionários for utilizada, ao se incluir um novo bloco $\hat{\mathbf{X}}^j$ no dicionário, na verdade é necessário incluir $T_{2^{-p}M_0,2^{-q}N_0}[\hat{\mathbf{X}}^j]$ em $\mathcal{D}_{p+q}$ para $(p,q) = \{(0,0),(0,1),(1,1),(1,2),\ldots,(\log_2 M_0, \log_2 N_0)\}$. É importante notar que, durante a codificação, as diferentes cópias de dicionário deixam de ser apenas versões escalonadas de $\mathcal{D}$, pois novos elementos só serão incluídos se não existirem no dicionário. Dependendo da escala em questão, o crescimento do dicionário pode ser mais rápido ou mais lento, pois a repetição de elementos acaba sendo mais provável. Isso ocorre, por exemplo, quando se tenta introduzir um elemento com dimensões 8×8 em uma escala de dimensões 2×1, pois o bloco transformado tem grande possibilidade de já existir. Os dicionários em escalas pequenas também tendem a saturar mais rapidamente, pois o número de elementos possíveis também é menor. Além disso, o dicionário não pode crescer indefinidamente, pois geralmente há restrições de memória; nas implementações práticas do algoritmo MMP utilizadas neste trabalho, o tamanho máximo do dicionário é sempre fixado em $L = 400.000$ elementos. Sendo assim, os vários dicionários acabam sendo podados de forma diferente (o elemento mais antigo dentre os menos utilizados é retirado a cada novo elemento). Para que isto fique bem claro, uma nova notação é adotada: ao invés de se utilizar $\mathbf{b}_i^s$, que representa a transformação para a escala s do elemento i do dicionário $\mathcal{D}$, será utilizado $\mathbf{b}_{i,s}$, que representa o elemento i de $\mathcal{D}_s$, o dicionário na escala s.

Considerando que os custos lagrangeanos $J(n_l)$ sejam independentes, um procedimento de otimização da árvore de segmentação $\mathcal{S}$ pode ser incorporado ao algoritmo de busca do elemento. Tal suposição é possível se o dicionário for suficientemente grande, pois assim não haveria variações significativas nos elementos disponíveis para a codificação, ou seja, o dicionário não apresentaria grandes diferenças durante a codificação de nós consecutivos. Conseqüentemente, a remoção de um nó não afetaria os elementos mais aptos à codificação de um outro nó, tornando os custos

independentes. Entretanto, dado que o dicionário é atualizado enquanto o dado é codificado, a poda de uma sub-árvore $\mathcal{S}(n_j)$ pode ocasionar a remoção de muitas atualizações do dicionário, o que afeta todos os nós à direita do atual. Sendo assim, se uma sub-árvore é podada, é necessário remover do dicionário um elemento que poderia mais tarde ser utilizado para aproximar um bloco de entrada.

Como já foi dito, a árvore de segmentação $\mathcal{S}$ pode ser otimizada num sentido taxa-distorção, permitindo uma distribuição dos bits para a codificação que leva em consideração as necessidades globais de cada bloco de entrada. A otimização começa a partir do bloco $\mathbf{X}^0$ e particiona recursivamente os blocos $\mathbf{X}^{2j+1}$, calculando os custos lagrangeanos $J(n_{2j+1})$ [72] associados aos mesmos. A ordem de varredura dos elementos está representada na Figura 2.4. Quando o algoritmo chega à última escala, ou seja, a de dimensões 1×1, o mesmo retorna analisando os custos das sub-árvores com início nos nós-filhos $\mathcal{S}(n_{2j+1})$ e $\mathcal{S}(n_{2j+2})$ e o custo do nó-pai n_j, para decidir sobre a remoção das sub-árvores, o que acontece se $J(n_j) \leq J(\mathcal{S}(n_{2j+1})) + J(\mathcal{S}(n_{2j+2}))$. As estatísticas dos elementos $\mathbf{b}_{i_{2k+1},s_{2k+1}}$ e dos *flags* descritores da árvore de segmentação utilizados, para todas as escalas $(p,q) = (0,0),(0,1),(1,1),(1,2),...,(\log_2 M_0, \log_2 N_0)$, são atualizadas a cada nó codificado, levando-se em consideração a utilização ou exclusão de um dado elemento. Se o algoritmo decide pela remoção, todas as atualizações devidas a $\mathcal{S}(n_{2j+1})$ e $\mathcal{S}(n_{2j+2})$ são removidas do dicionário $\mathcal{D}$ e as estatísticas dos elementos e dos *flags* são corrigidas. O procedimento continua através da partição e análise dos outros sub-blocos $\mathbf{X}^{2j+2}$ da partição atual (o bloco à direita ou abaixo) e então para os blocos seguintes à esquerda e acima, até que não haja mais elementos.

É importante ressaltar que as probabilidades de ocorrência de cada elemento do dicionário e de cada *flag* em cada profundidade são necessárias no cálculo dos custos lagrangeanos. Sendo assim, mantém-se um registro do número de vezes que cada um é utilizado, que também é corrigido quando ocorrem atualizações ou podas.

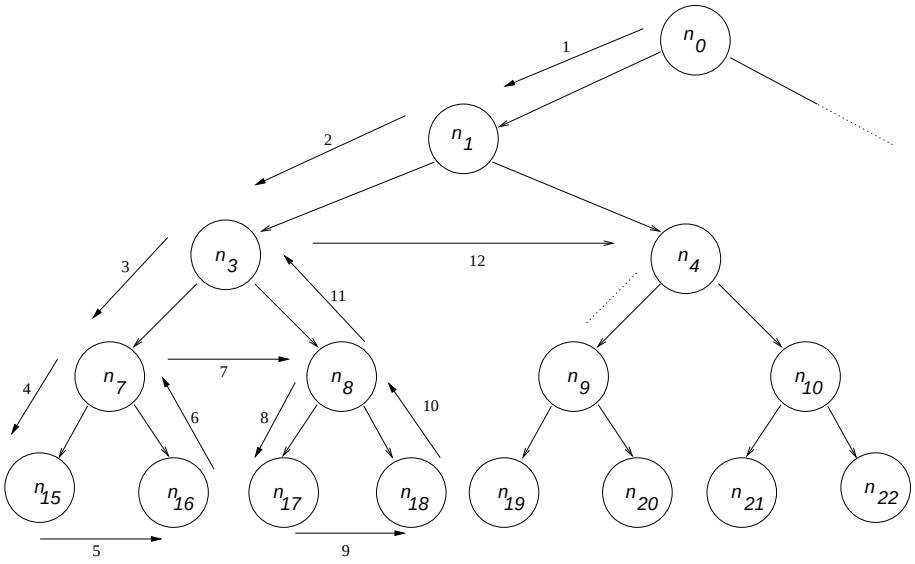


Figura 2.4: Ordem de processamento dos nós durante a otimização no MMP.

Capítulo 3

A codificação distribuída de sinais

A codificação distribuída consiste no fato de duas ou mais fontes aleatórias serem codificadas independentemente e decodificadas conjuntamente, explorando-se suas dependências estatísticas no codificador, mas com acesso à informação auxiliar também no decodificador. Os seus princípios foram fundamentados nos anos 70 por Slepian e Wolf [19], para a compressão sem perdas, sendo mais tarde estendidos para a compressão com perdas por Wyner e Ziv [20]. Na época, Wyner e Ziv conseguiram mostrar que a compressão (com perdas) com exploração da informação auxiliar no decodificador poderia ser obtida com um aumento de taxa maior ou igual a zero, sendo que a igualdade era verificada em alguns casos específicos, como no das fontes gaussianas sem memória.

Essa constatação tinha o potencial de provocar o surgimento de uma ramificação completamente nova na área de codificação de sinais, o que só aconteceu recentemente, pois o paradigma de processamento poderia mudar drasticamente: um ou vários codificadores simplificados e apenas um decodificador extremamente complexo, que exploraria a correlação existente entre os dados.

3.1 A codificação distribuída sem perdas

Supondo-se duas seqüências i.i.d. X e Y, é possível codificar as mesmas sem erros ($\tilde{X} = X$ e $\tilde{Y} = Y$), atingindo-se as taxas $R_X \geq H(X)$ e $R_Y \geq H(Y)$, sendo $H(X)$ a entropia de X e $H(Y)$ a entropia de Y [8]. Obviamente, tal resultado não é construtivo, pois ainda é necessário encontrar códigos que possibilitem alcançar

esses patamares. Se as fontes forem codificadas conjuntamente, uma taxa igual a $H(X,Y)$ (entropia conjunta de X e Y) também se mostra suficiente para a sua reconstrução sem perdas, como mostrado na Figura 3.1. Este é o processamento realizado pelos codificadores tradicionais atualmente utilizados.

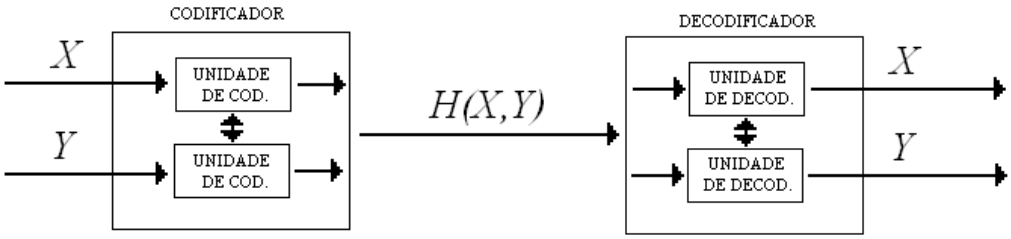


Figura 3.1: Abordagem utilizada na compressão tradicional de dados.

Entretanto, Slepian e Wolf demonstraram que há uma região de taxas atingíveis para a codificação com informação auxiliar no decodificador, definida por

$$R_X + R_Y \geq H(X,Y) \tag{3.1}$$

com $R_X \geq H(X|Y)$ e $R_Y \geq H(Y|X)$, o que está representado na Figura 3.2. Surgiu, assim, o teorema de Slepian-Wolf. Isto significa que uma taxa igual à entropia conjunta $H(X,Y)$ também pode sempre ser alcançada para a codificação com informação auxiliar no decodificador, desde que se lide com uma probabilidade de erro residual na recuperação de X e Y (a recuperação sem erros é válida para casos assintóticos), seja explorando-se parte das dependências no codificador e parte no decodificador ou totalmente no decodificador. Sendo assim, dado que a seqüência Y pode ser codificada com taxa $R_Y = H(Y)$, X pode ser codificada com $H(X|Y)$ (utilizando-se o conhecimento sobre as estatísticas conjuntas, mas sem de fato utilizar as amostras y) e decodificada completamente tendo-se acesso a Y no decodificador. A prova do teorema de Slepian-Wolf é baseada em *binning* aleatório.

De fato, podem-se desenvolver codificadores de modo que toda a região de Slepian-Wolf seja atingida; o maior problema está no projeto de códigos práticos, que possibilitem tal objetivo. A compressão com informação auxiliar apenas no decodificador é identificada como os pontos A e B da Figura 3.2, ou seja, a abordagem conhecida como assimétrica.

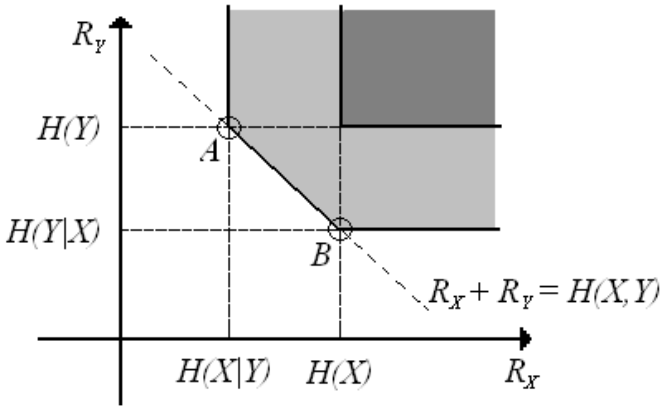


Figura 3.2: Região de taxas atingíveis para a codificação de Slepian-Wolf.

Como exemplo prático de como funciona a codificação com informação auxiliar apenas no decodificador [1], suponhamos X e Y como palavras de três bits equiprováveis, ou seja, $H(X) = 3$ bits e $H(Y) = 3$ bits, cuja diferença esteja, no máximo, em um bit ($d_{Hamming}(X, Y) \leq 1$). Assim, se $X = 011$, Y pode ser 011, 111, 001 ou 010. A informação mais importante para a compressão distribuída reside no conhecimento da estatística conjunta de X e Y, ou seja, $d_{Hamming}(X, Y) \leq 1$, que permitirá a codificação de $X = \{x_1, x_2, x_2, \ldots, x_n\}$ sem se ter acesso direto a $Y = \{y_1, y_2, y_2, \ldots, y_n\}$. Para que isso seja possível, o espaço amostral de X, ou seja $\{000, 001, 010, 011, 100, 101, 110, 111\}$, é particionado em quatro conjuntos disjuntos $\{000, 111\}$, $\{001, 110\}$, $\{010, 101\}$ e $\{011, 100\}$, onde as distâncias entre palavras de um mesmo conjunto é de 3 bits. Ao se processar X, verifica-se a que conjunto o mesmo pertence e envia-se apenas o índice desse conjunto, que pode ser codificado com apenas dois bits (são apenas quatro conjuntos). Na Figura 3.3, por exemplo, para o primeiro conjunto de bits de cada fonte, envia-se o código relacionado ao valor de Y (por exemplo, resultante do codificador de entropia) ou os próprios bits (valor bruto), ou seja, $C(001)$, e apenas o conjunto ao qual o valor de X pertence, o que resulta em 11. No receptor, $Y = \{y_1, y_2, y_2, \ldots, y_n\}$ é então utilizado para se decidir entre as palavras de um dado conjunto, conforme representado na Figura 3.3. Como $H(X, Y) = H(Y) + H(X|Y) = 3 + 2 = 5$ bits e utilizam-se 5 bits para cada transmissão (3 para cada elemento de Y e 2 para cada índice), realmente alcança-se um dos cantos (A ou B) da região de Slepian-Wolf.

Este exemplo, apesar de simples, revela dois pontos interessantes:

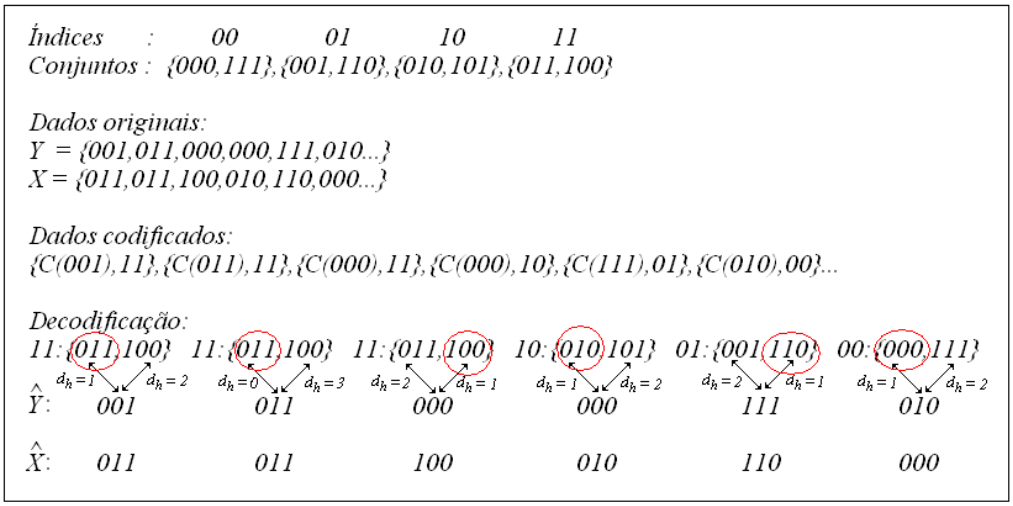


Figura 3.3: Exemplo de codificação distribuída.

- O conjunto de índice 00 é um código de repetição de distância 3, sendo os outros *cosets* [66, 67] deste;
- Os conjuntos foram montados com base na estatística condicional (também conhecida como canal de correlação) de X e Y.

Apesar de a teoria fornecer a base para que se processe o dado com informação auxiliar no decodificador, a abordagem de *binning* não é suficiente para se desenvolverem aplicações práticas. Entretanto, observando-se o que foi feito no exemplo da Figura 3.3 e os pontos levantados, percebe-se que, na verdade, foi criado um código corretor de erros casado com o "ruído de correlação", que corrige a informação de X baseado nas estatísticas conjuntas com Y. Pode-se, inclusive, mostrar que se m é a palavra código ou mensagem, G é a matriz geradora, x é o dado a ser codificado, H é a matriz de checagem de paridade e s é a síndrome da palavra código (também conhecido como padrão de erro), então

$$m = Gx = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} x, \text{ com } G = \begin{bmatrix} I_k \\ P \end{bmatrix}$$

$$G^T H = H^T G = 0 \Rightarrow H = \begin{bmatrix} P^T \\ I_{n-k} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$s = mH$$

$$p/\ m = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix} \Rightarrow s = \begin{bmatrix} 1 & 1 \end{bmatrix}$$
$$p/\ m = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \Rightarrow s = \begin{bmatrix} 1 & 1 \end{bmatrix}$$
$$p/\ m = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \Rightarrow s = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

De fato, é fácil notar que os índices dos conjuntos são, na verdade, as síndromes do código projetado para o canal de correlação dos dados da Figura 3.3.

Sendo assim, uma abordagem construtiva direta para o problema da codificação distribuída consiste em se utilizar códigos corretores de erros, o que foi primeiramente sugerido por Wyner [23]. A idéia consiste em se particionar o espaço amostral em conjuntos disjuntos, que são *cosets* de um dado código corretor de erros com uma estrutura adequada ao canal de correlação. Esta abordagem está representada na Figura 3.4. De fato, se a correlação entre X e Y puder ser modelada por um canal binário (*e.g.* canal binário simétrico), códigos turbo [76] e LDPC [77] podem ser utilizados para se obterem desempenhos próximos ao limite de Slepian-Wolf.

Figura 3.4: Codificação distribuída prática.

Projetos de codificadores de Slepian-Wolf práticos podem ser encontrados em [1, 24–26, 42]. Algumas dessas abordagens utilizam códigos LDPC ou turbo e outras utilizam códigos em treliça, com bons resultados. Entretanto, vários trabalhos não possuem uma relação mais direta com o conceito estabelecido por Slepian e Wolf, enviando bits de paridade ao invés de síndromes [1, 24].

Codificadores baseados em códigos LDPC e Turbo têm-se tornado cada vez mais comuns, devido principalmente ao bom desempenho apresentado pelos mesmos em diversos tipos de canais e situações de erro. O codificador em [1], por exemplo, utiliza códigos turbo perfurados de taxa adaptável [78], com um canal de *feedback*

para mudar a taxa de codificação caso o canal de correlação sofra variações, o que é estimado pelo decodificador através da probabilidade de erro de símbolo. Este *codec* está representado na Figura 3.5.

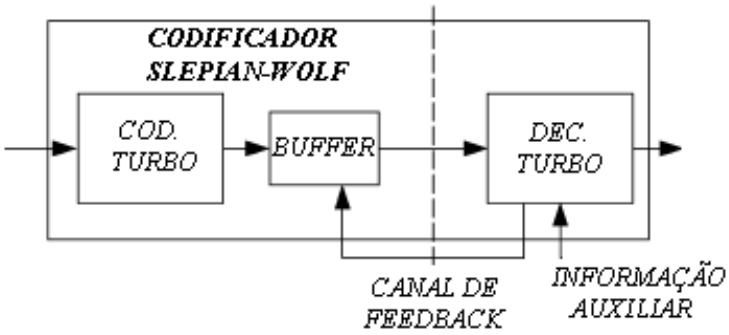


Figura 3.5: Codificador de Slepian-Wolf com canal de *feedback* apresentado em [1].

Já o codificador em [25] é baseado em códigos LDPC, utilizando diretamente o conceito de Slepian-Wolf através do envio de síndromes. A entrada x é multiplicada pela matriz de checagem de paridade H do código LDPC, encontrando-se a síndrome s correspondente. O decodificador, então, estima o dado enviado através da síndrome recebida e da informação auxiliar Y.

Em [42], o espaço das palavras de dados é particionado utilizando-se um código em treliça com constante de restrição $k = 8$ e taxa $1/2$ [66], gerando-se as síndromes através da convolução entre a palavra de dados e a matriz de checagem de paridade do código. A adequação ao canal de correlação é realizada através da classificação dos blocos de dados em conjuntos, cada um com um código em treliça apropriado. Alguns autores também desenvolveram trabalhos sobre casos simétricos (ponto médio entre A e B na Figura 3.2) [23], com bons resultados. É possível perceber em [17] que quanto maior é a correlação entre X e Y, menor é a probabilidade de erro para um dado código de Slepian-Wolf e, quanto maior é a capacidade de correção de erros do código, menor é a correlação exigida para se atingir uma dada probabilidade de erro. Isso mostra que, de fato, bons códigos corretores de erros são sérios candidatos a codificadores de Slepian-Wolf. Mais genericamente, pode-se inferir que códigos casados com o canal de correlação podem ser utilizados como códigos de Slepian-Wolf e têm o potencial para proporcionar desempenhos próximos aos limites apresentados.

3.2 A codificação distribuída com perdas

Como já foi dito, o trabalho de Slepian e Wolf contemplava apenas codificação sem perdas. Entretanto, a codificação com perdas também era um tópico muito interessante e de aplicação direta a diversos casos práticos, levando à necessidade de contemplação da mesma no esquema de compressão distribuída, o que atraiu o interesse de Wyner e Ziv [20–23].

Logo depois de [19], Wyner e Ziv [20] estenderam o trabalho de Slepian e Wolf através da criação de uma "teoria taxa-distorção para compressão com perdas e informação auxiliar no decodificador", apresentando resultados semelhantes aos de Slepian e Wolf. Supondo-se X e Y fontes aleatórias, com Y sendo a informação auxiliar, e que uma distorção $D = E[d(X, \tilde{X})]$ pode ocorrer na reconstrução de X, Wyner e Ziv provaram que

$$R_{X|Y}^{WZ}(D) \geq R_{X|Y}(D), \tag{3.2}$$

onde $R_{X|Y}^{WZ}(D)$ é a função taxa-distorção de Wyner e Ziv e $R_{X|Y}(D)$ é a função taxa distorção para Y disponível apenas no codificador. Logo, se o codificador não tiver acesso a Y, uma perda de taxa maior ou igual a zero pode ocorrer, o que consiste no dual do teorema apresentado em [79]. No caso de fontes gaussianas sem memória, a perda de taxa é zero [20]. Por exemplo, para o canal binário simétrico (*Binary Symmetric Channel* - BSC), tem-se:

$$R_{X|Y}(D) = \begin{cases} H(p) - H(D), & 0 \leq D \leq \min\{p, 1-p\} \\ 0, & D > \min\{p, 1-p\} \end{cases} \tag{3.3}$$

$$R_{X|Y}^{WZ}(D) = l.c.e.\{H(p * D) - H(D), (p, 0)\},\ 0 \leq D \leq p \tag{3.4}$$

onde p é a probabilidade de interpretação de 1 como 0 e vice-versa e $l.c.e.$ é o fecho convexo inferior de $H(p * D) - H(D)$ e o ponto $(p, 0)$, com a operação $p * D = (1-p)D + (1-D)p$.

Embora haja uma perda de taxa na codificação de Wyner-Ziv, a mesma ainda é extremamente importante, principalmente em redes de sensores, pois muitas fontes de vídeo e imagens podem ser modeladas como fontes gaussianas conjuntas [17]. Além disso, o projeto de códigos de Wyner-Ziv mais adequados tende a resultar em desempenhos próximos aos limites apresentados.

Na codificação de Wyner-Ziv, assim como em qualquer esquema de codificação tradicional, a fonte X sofre perdas (quantização) para facilitar sua representação e a correlação resultante com Y é explorada para uma redução ainda maior de taxa. Como a codificação de Slepian-Wolf é, na verdade, um problema de codificação de canal, a codificação de Wyner-Ziv torna-se um problema de codificação conjunta fonte-canal, como na quantização codificada por treliças [80], e a informação auxiliar Y pode ser utilizada tanto para decodificar X quanto para estimar o mesmo. Um codificador de Wyner-Ziv genérico está ilustrado na Figura 3.6. O quantizador divide o espaço de sinal em células, identificadas por um índice Q. Um algoritmo eficiente para gerar quantizadores ótimos pode ser encontrado em [81] e uma extensão mais geral do algoritmo de Loyd em [82]. Em [55], Zamir et. al propuseram a utilização de reticulados que, sob certas condições, podem resultar em desempenhos próximos do limite de Wyner-Ziv. O codificador de Slepian-Wolf funciona conforme explicado na seção 3.1 e transmite os bits de síndrome através do canal (considerado aqui um canal ideal). No decodificador, a informação auxiliar Y é utilizada para se decidir sobre $\tilde{X}$ e, posteriormente, refinar a informação reconstruída.

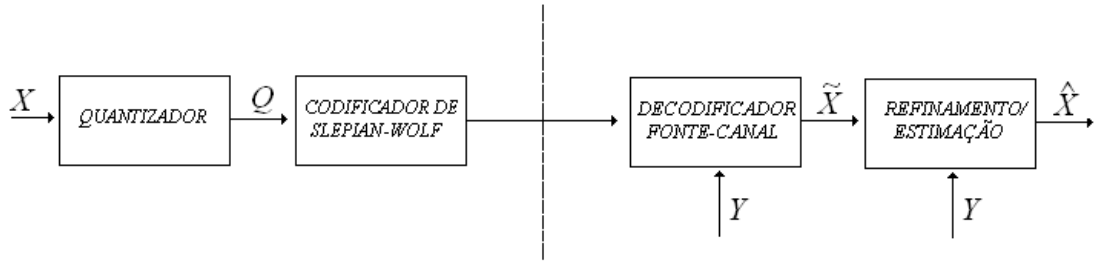


Figura 3.6: Codificador de Wyner-Ziv.

Há vários codificadores de Wyner-Ziv na literatura, sendo um exemplo interessante o apresentado em [2]. Nesse paradigma, representado Figura 3.7, um subconjunto dos quadros da seqüência de vídeo são designados como quadros-chave, que são codificados e decodificados com um *codec* intra-quadros tradicional. Entre os quadros-chave estão os quadros de Wyner-Ziv, que são codificados segundo uma abordagem intra-quadros e decodificados utilizando-se a informação auxiliar, ou seja, os quadros-chave, através de uma predição inter-quadros. Uma transformada discreta do cosseno (*Discrete Cosine Transform* - DCT) em blocos é aplicada a cada quadro de vídeo Wyner-Ziv, sendo os coeficientes quantizados independentemente,

agrupados e comprimidos por um codificador de Slepian-Wolf como o mostrado na Figura 3.5. A estimação de movimento no decodificador é auxiliada por um código *hash* enviado juntamente com os dados, consistindo em um pequeno conjunto de coeficientes de DCT quantizados.

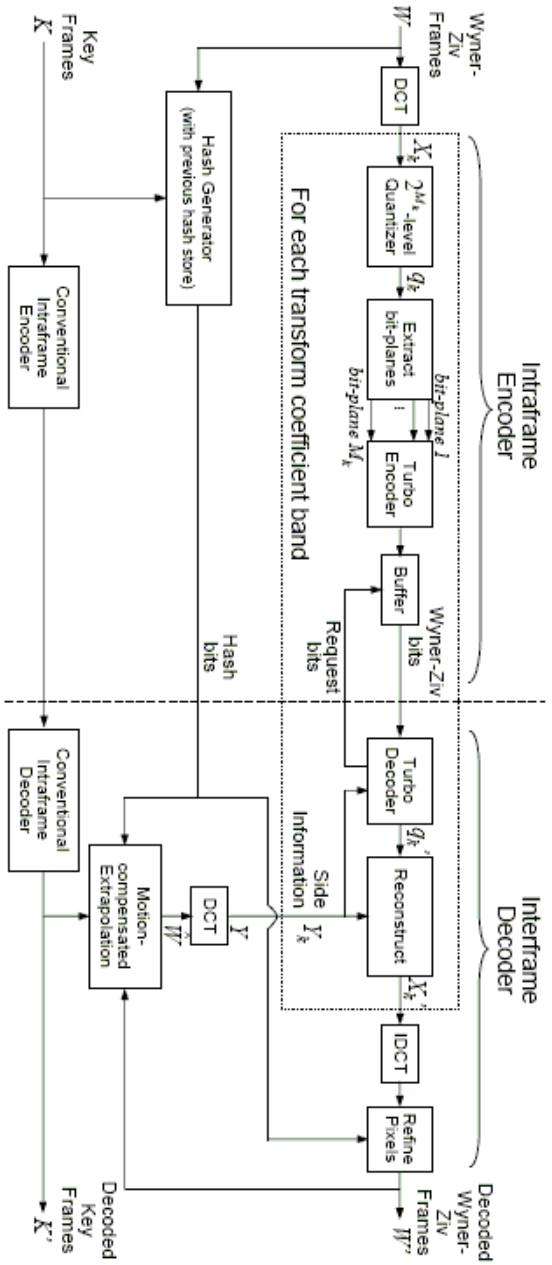


Figura 3.7: Codificador de Wyner-Ziv em [2].

3.3 Discussão sobre codificação distribuída

Como já foi mencionado, a codificação de Wyner-Ziv é, na verdade, um problema de codificação conjunta fonte-canal, com quantização para a adequação dos dados e codificação de canal para a criação de *cosets*/compressão. Sendo assim, é necessário expor as características de cada problema e encontrar os pontos de intersecção, que podem ser explorados para uma otimização do conjunto. Na codificação de sinal-fonte, lida-se basicamente com dois tipos de ganho [83]:

- **Ganho granular**: Ganho de desempenho resultante de uma cobertura (espacial) local mais eficiente durante a quantização. Por exemplo, em duas dimensões, regiões hexagonais são mais eficientes que regiões quadradas. O ganho granular é dependente da medida de distorção [84];
- **Ganho de Fronteira**: Ganho de desempenho resultante de uma cobertura (espacial) global mais eficiente, ou seja, da capacidade do quantizador em adaptar a distribuição dos seus símbolos à distribuição do sinal. Por exemplo, para fontes não uniformes, o ganho de fronteira pode ser muito maior que o ganho granular. Um meio de se medir o desempenho na exploração do ganho de fronteira reside no cálculo da entropia. O ganho de fronteira também é chamado de ganho de cobertura [84].

Na codificação de canal, por sua vez, há os seguintes ganhos [83]:

- **Ganho de Formatação**: Em esquemas de modulação codificada, o ganho de formatação resulta da distribuição global das palavras-código no espaço, com o objetivo de gerar uma constelação não uniforme e adaptada à distribuição do sinal, reduzindo a energia média da representação. Logo, a principal conseqüência deste ganho é uma economia de energia, representada pelo aumento do *Peak-to-Average Power Ratio* (PAPR) [85], ou seja, a razão entre a potência de pico e a potência média. Diz-se, por exemplo, que o ganho de formatação máximo é atingido com a codificação de entropia (sinais com grande norma são utilizados menos freqüentemente que os com pequena norma) [86];
- **Ganho de Empacotamento**: Ganho resultante da distribuição local das palavras-código, de modo a se manter uma distância máxima entre as mesmas e

permitir uma correção de erros mais robusta [87]. O ganho de empacotamento é o principal objetivo de um código corretor de erros.

Como na DSC o código corretor de erros é utilizado apenas para se realizar a compressão, fazendo uso da informação auxiliar Y para recuperar X através da "correção dos erros resultantes do canal de correlação" (o que significa que o único objetivo é espaçar as palavras-código criando os *cosets*), o principal objetivo da codificação de canal é o ganho de empacotamento. Além disso, existe algum relacionamento entre os ganhos apresentados: por exemplo, parte do ganho granular pode ser sacrificado para se obter um ganho de empacotamento maior [88] e, em geral, os problemas de empacotamento e cobertura não possuem a mesma solução [87]. Como um codificador de Wyner-Ziv é, acima de tudo, um codificador de sinal-fonte, é interessante estabelecer uma equivalência entre os ganhos de empacotamento e fronteira, de modo que o segundo possa ser atingido na etapa de codificação de canal, o que proporcionaria um desempenho ótimo.

Uma pergunta interessante, que surge com o que foi apresentado, é se a compressão distribuída é capaz de alcançar o mesmo desempenho dos esquemas de compressão tradicionais, principalmente, por exemplo, numa rede de sensores onde não haja comunicação entre os nós (um receptor significaria mais *hardware* e um maior consumo de energia, além de maior custo). Em princípio, isso é possível, pois existe uma forte correlação entre as imagens capturadas por sensores vizinhos e a compressão distribuída pode explorá-la do mesmo modo que a codificação tradicional [17]. Além disso, mesmo que ainda haja uma diferença, a compressão distribuída ainda é necessária em redes com reversão de complexidade, onde o codificador precisa apresentar baixo custo. O maior problema, entretanto, reside na fundamentação e compreensão da relação entre os componentes do *codec* (quantizador e codificador de Slepian-Wolf), o que permitiria o desenvolvimento de esquemas mais robustos e adaptados aos dados. Além disso, há também a necessidade de uma modelagem mais concisa do canal de correlação, resultando num código específico e de alto rendimento para uma dada aplicação.

Capítulo 4

Aplicações do algoritmo MMP

Este capítulo apresenta uma investigação desenvolvida sobre as aplicações do algoritmo MMP, com os objetivos de identificar outros sinais que possam se beneficiar da abordagem baseada em padrões multiescalas recorrentes e desenvolver novas ferramentas de codificação para se estender o algoritmo base. Três tipos de sinais foram utilizados: imagens naturais, ECGs e EMGs. Dado que os mesmos são completamente diferentes, representando classes de dados distintas, é possível uma análise bastante abrangente do desempenho do MMP em compressão genérica de dados.

4.1 O MMP aplicado à compressão de imagens

A primeira aplicação do algoritmo MMP foi a compressão de imagens, inicialmente abordada em [10–12]. Nas seções seguintes, serão propostas novas extensões aos algoritmos em [10–12], através da adoção de modelos para a fonte e do desenvolvimento de novas estratégias de dicionário.

4.1.1 O MMP com modelos de probabilidade adaptativos: APM-MMP

O algoritmo MMP básico, conforme apresentado no capítulo 2, proporcionou um bom desempenho para imagens naturais, como a *Lena* e a *Barbara* (ver o apêndice B), e um excelente desempenho para imagens mistas e de texto, como a *PP1205* e a *PP1209* (ver o apêndice B). Entretanto, principalmente para imagens naturais, sempre ocorria o efeito de blocagem, fazendo com que as transições entre blocos de

imagem ficassem bem evidentes. Esse efeito é devido ao fato do MMP processar cada bloco de imagem independentemente, sem qualquer análise dos elementos já codificados. Nesta seção, apresenta-se uma extensão do MMP que procura explorar de modo mais eficaz a dependência estatística inter-blocos.

O modelo de probabilidades adaptativo

O MMP original foi concebido como um esquema de compressão universal, ou seja, ele não assume qualquer conhecimento *a priori* sobre a natureza estatística do sinal a ser comprimido. Ao tentar codificar um sinal qualquer, o MMP deve aprender as estatísticas a partir dos próprios dados em codificação, o que é realizado pela atualização de dicionário e pela codificação aritmética de índices e *flags*. A eficiência desse processo depende da adaptação proporcionada por essas duas técnicas, o que aumenta com a codificação de grandes conjuntos de dados. Entretanto, os primeiros segmentos de sinal sempre são codificados com baixa eficiência, dado que, no início do processo, o modelo estatístico não representa com fidelidade a fonte em questão. Isso poderia ser minimizado através de treinamento, porém, se a imagem não fizer parte ou for muito diferente do conjunto de treinamento, o início da codificação ainda será ineficiente.

Durante a codificação de um dado bloco, todos os elementos do dicionário adicionados devido aos blocos anteriores são utilizados, permitindo ao MMP explorar levemente a dependência estatística entre os mesmos. Apesar disso, o MMP não é capaz de aprender ou explorar diretamente as estatísticas conjuntas dos blocos de imagem, resultando em escolhas independentes para cada bloco codificado. Por exemplo, um bloco $\hat{\mathbf{X}}^0$ codificado pelo MMP pode apresentar descontinuidades severas nas fronteiras dos blocos $\hat{\mathbf{X}}^j$ gerados pelo procedimento de segmentação, mesmo que o bloco original $\mathbf{X}^0$ seja suave, fato este ilustrado na Figura 4.1.

Estas descontinuidades ocorrem porque a distorção na equação (2.1), utilizada para avaliar o custo do nó n_j, não depende das escolhas feitas para os blocos vizinhos de $\hat{\mathbf{X}}^j$. Portanto, o MMP não tem controle sobre a suavidade nas fronteiras dos mesmos. A maneira mais interessante de se corrigir esse problema, evitando a transmissão de informação auxiliar, é definir um modelo de probabilidades adaptativo, com o potencial de reduzir a taxa através da atribuição de probabilidades

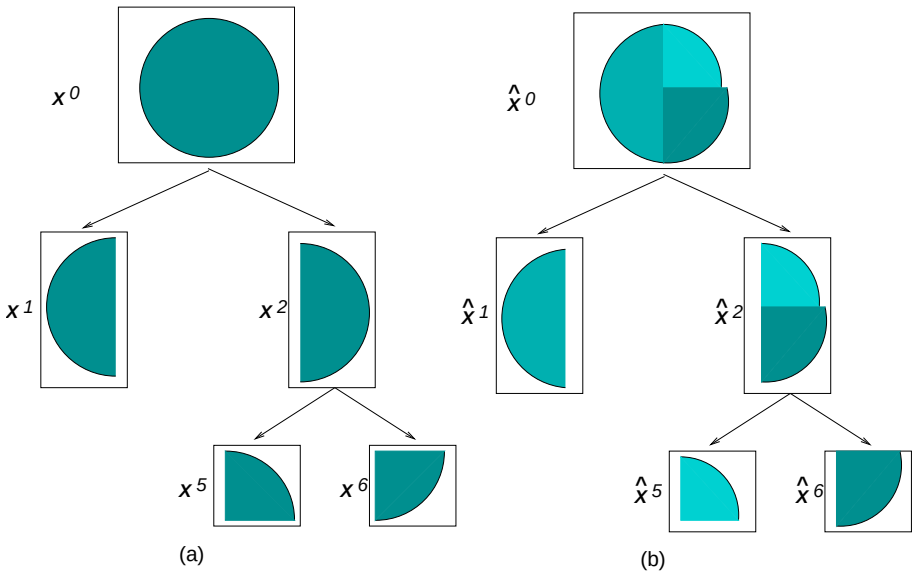


Figura 4.1: Descontinuidades causadas pelo processo de segmentação do MMP: (a) Bloco original; (b) Bloco reconstruído.

de ocorrência mais altas para elementos semelhantes aos seus vizinhos causais, de acordo com uma dada métrica de distorção.

A extensão ao algoritmo MMP proposta nesta seção tem o objetivo de implementar tal modelo de probabilidades, baseando-se nas características dos vizinhos causais do bloco em questão.

No MMP, a função densidade de probabilidade é estimada mantendo-se registros do número de vezes que cada elemento $\mathbf{b}_i$ de $\mathcal{D}$ é utilizado. O MMP básico utiliza essas probabilidades para excitar os modelos do codificador aritmético durante a otimização da árvore de segmentação e a codificação. Na nova abordagem, as probabilidades são ponderadas de acordo com a vizinhança causal, através de uma função de ponderação, influenciando as escolhas feitas pelo MMP para cada bloco. Tal processamento é equivalente a se codificar um bloco através de contextos [89–91], tendo como métrica a continuidade através da vizinhança causal.

Para que a presente abordagem seja descrita com clareza, é necessário fazer as seguintes definições. Para um bloco $\mathbf{X}^j$, de dimensões $M_j \times N_j$, associado ao nó n_j, definem-se o *vizinho superior* $\mathbf{U}^j$ como o bloco, de dimensões $M_j \times N_j$, imediatamente acima de $\mathbf{X}^j$, e o *vizinho esquerdo* $\mathbf{L}^j$ como o bloco, de dimensões $M_j \times N_j$, imediatamente à esquerda de $\mathbf{X}^j$, o que está ilustrado na Figura 4.2. Definem-se também o *vizinho superior reconstruído* $\hat{\mathbf{U}}^j$ e o *vizinho esquerdo reconstruído* $\hat{\mathbf{L}}^j$, como os

blocos utilizados pelo MMP para representar $\mathbf{U}^j$ e $\mathbf{L}^j$, respectivamente. Tanto $\hat{\mathbf{U}}^j$ quanto $\hat{\mathbf{L}}^j$ estão normalmente disponíveis antes da codificação de $\mathbf{X}^j$ (com exceção dos blocos localizados nas bordas da imagem), devido à ordem de varredura dos blocos, que ocorre no sentido de leitura (da esquerda para a direita e de cima para baixo). Se o bloco de entrada estiver localizado na primeira linha ou na primeira coluna da imagem, $\hat{\mathbf{U}}^j$ ou $\hat{\mathbf{L}}^j$, respectivamente, podem não existir. De qualquer modo, sempre haverá pelo menos um vizinho, o que valida a análise feita (com exceção do primeiro bloco).

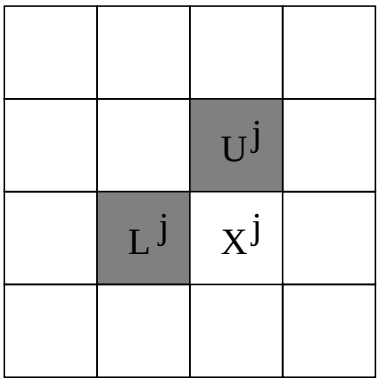


Figura 4.2: Vizinhos superior e esquerdo.

Por exemplo, com relação à Figura 2.3, quando o MMP tenta codificar $\mathbf{X}^{10}$, as representações de $\hat{\mathbf{X}}^3$, $\hat{\mathbf{X}}^{19}$ e $\hat{\mathbf{X}}^{20}$ já estão disponíveis. Portanto, conhece-se exatamente a primeira coluna e até a segunda coluna das duas primeiras linhas de $\hat{\mathbf{X}}^0$, ou seja

$$\hat{\mathbf{X}}^0 = \begin{pmatrix} \hat{x}(0,0) & \hat{x}(0,1) & ? & ? \\ \hat{x}(1,0) & \hat{x}(1,1) & ? & ? \\ \hat{x}(2,0) & ? & ? & ? \\ \hat{x}(3,0) & ? & ? & ? \end{pmatrix}.$$

Os vizinhos superior e esquerdo reconstruídos de $\mathbf{X}^{10}$, respectivamente, são determinados como

$$\hat{\mathbf{L}}^2 = \begin{pmatrix} \hat{x}(2,0) \\ \hat{x}(3,0) \end{pmatrix} \quad \text{and} \quad \hat{\mathbf{U}}^2 = \begin{pmatrix} \hat{x}(0,1) \\ \hat{x}(1,1) \end{pmatrix}.$$

Para que as dependências estatísticas de um dado bloco $\mathbf{X}^j$ com relação aos seus vizinhos sejam exploradas, precisa-se definir um modo de estimar a probabilidade condicional de $\mathbf{X}^j$, dado que os seus vizinhos superior e esquerdo ocorreram. Se $\mathbf{x}^j$

for um vetor aleatório representando o bloco de imagem j e $\mathbf{x}^{j_U}$ e $\mathbf{x}^{j_L}$ forem vetores representando os seus vizinhos causais, pode-se escrever

$$\begin{aligned}
&\Pr \left\{\mathbf{x}^j = \mathbf{X}^j \left| \mathbf{x}^{j_U} = \mathbf{U}^j, \mathbf{x}^{j_L} = \mathbf{L}^j \right.\right\} = \\
&= \frac{\Pr\left\{\mathbf{x}^j = \mathbf{X}^j, \mathbf{x}^{j_U} = \mathbf{U}^j, \mathbf{x}^{j_L} = \mathbf{L}^j\right\}}{\Pr\left\{\mathbf{x}^{j_U} = \mathbf{U}^j, \mathbf{x}^{j_L} = \mathbf{L}^j\right\}} \\
&= \frac{\Pr\left\{\mathbf{x}^{j_U} = \mathbf{U}^j, \mathbf{x}^{j_L} = \mathbf{L}^j \left| \mathbf{x}^j = \mathbf{X}^j \right.\right\} \Pr\left\{\mathbf{x}^j = \mathbf{X}^j\right\}}{\Pr\left\{\mathbf{x}^{j_U} = \mathbf{U}^j, \mathbf{x}^{j_L} = \mathbf{L}^j\right\}} \\
&= \mathcal{W}\left(\mathbf{X}^j, \mathbf{U}^j, \mathbf{L}^j\right) \Pr\left\{\mathbf{x}^j = \mathbf{X}^j\right\}. \qquad (4.1)
\end{aligned}$$

Os vizinhos superior e esquerdo foram escolhidos porque são a vizinhança causal mais próxima de $\mathbf{X}^j$ e estão diretamente disponíveis para o decodificador, o que evita a transmissão de informação auxiliar. Além disso, eles proporcionam uma maneira simples de se prever o comportamento do próximo bloco, dado que a suposição de suavidade é válida para a imagem.

O algoritmo MMP básico utiliza o número de ocorrências de cada elemento de dicionário para estimar a probabilidade $\Pr\{\mathbf{b}_{i,s}\}$ de que cada candidato $\mathbf{b}_{i,s}$ seja utilizado para representar $\mathbf{X}^j$. Esta informação é utilizada na otimização da árvore de segmentação e na codificação de cada índice. Como o dicionário é construído a partir da própria entrada, espera-se que $\Pr\{\mathbf{b}_{i,s}\}$ também possa ser utilizado como estimativa de $\Pr\{\mathbf{x}^j = \mathbf{b}_{i,s}\}$. Entretanto, se a vizinhança é conhecida, a taxa para se codificar o índice do elemento utilizando-se $\Pr\{\mathbf{x}^j = \mathbf{X}^j | \mathbf{x}^{j_U} = \mathbf{U}^j, \mathbf{x}^{j_L} = \mathbf{L}^j\}$ como modelo de probabilidade é, no máximo, a mesma (e normalmente é menor) que a obtida utilizando-se $\Pr\{\mathbf{x}^j = \mathbf{X}^j\}$, dado que a entropia condicional $H(X|Y) = H(X) - H(X,Y)$ não pode ser maior que a entropia $H(X)$ [92].

Sendo assim, estas constatações podem ser utilizadas para melhorar a eficiência de codificação do MMP, desde que se utilize o seguinte modelo de probabilidades, derivado da equação (4.1):

$$\Pr\left\{\mathbf{b}_{i,s} \left| \hat{\mathbf{U}}^j, \hat{\mathbf{L}}^j \right.\right\} = \mathcal{W}\left(\mathbf{b}_{i,s}, \hat{\mathbf{U}}^j, \hat{\mathbf{L}}^j\right) \Pr\{\mathbf{b}_{i,s}\}, \qquad (4.2)$$

onde $\Pr\{\mathbf{b}_{i,s}\}$ são as probabilidades utilizadas no MMP original. É importante perceber que utilizam-se $\hat{\mathbf{U}}^j$ e $\hat{\mathbf{L}}^j$ ao invés de $\mathbf{U}^j$ e $\mathbf{L}^j$, respectivamente, o que evita a transmissão de informação auxiliar.

O próximo passo é definir a função de ponderação $\mathcal{W}(.)$. Idealmente, a equação (4.1) deveria refletir a probabilidade conjunta dos vizinhos superior e esquerdo para

cada $\mathbf{b}_{i,s}$. Entretanto, tal estimação não é prática e deve-se procurar por simplificações adequadas. Assim, associa-se $\mathcal{W}\left(\mathbf{b}_{i,s}, \hat{\mathbf{U}}^j, \hat{\mathbf{L}}^j\right)$ a uma medida de continuidade de cada bloco $\mathbf{b}_{i,s}$, com relação aos vizinhos causais $\hat{\mathbf{U}}^j$ e $\hat{\mathbf{L}}^j$.

Dado que a posição da primeira amostra de $\mathbf{X}^j$ (de dimensões $M_j \times N_j$) dentro de $\mathbf{X}^0$ é $x(k_j, l_j)$, a posição da primeira amostra de $\hat{\mathbf{U}}^j$ e $\hat{\mathbf{L}}^j$ são $\hat{x}(k_j - M_j, l_j)$ e $\hat{x}(k_j, l_j - N_j)$, respectivamente, sendo o canto superior esquerdo considerado como a origem do bloco.

Para que a suavidade da aproximação seja medida, definem-se quatro parâmetros de comparação para cada bloco $\mathbf{b}_{i,s} \in \mathcal{D}_s$, onde $\mathcal{D}_s$ é a cópia do dicionário $\mathcal{D}$ na escala s (o canto superior esquerdo do bloco $\mathbf{b}_{i,s}$ corresponde à coordenada $(0,0)$):

i) As *descontinuidades de ordem zero*:

$$D_v^0\left(j, \mathbf{b}_{i,s}, n\right) = \hat{x}(k_j - 1, l_j + n) - b_{i,s}(0, n),\ n = 0, 1, \ldots, N_j - 1 \tag{4.3}$$

$$D_h^0\left(j, \mathbf{b}_{i,s}, m\right) = \hat{x}(k_j + m, l_j - 1) - b_{i,s}(m, 0),\ m = 0, 1, \ldots, M_j - 1 \tag{4.4}$$

ii) As *descontinuidades de primeira ordem*:

$$\begin{aligned} D_v^1\left(j, \mathbf{b}_{i,s}, n\right) \ = \ & \hat{x}(k_j - 2, n + l_j) - \hat{x}(k_j - 1, n + l_j) + \\ & b_{i,s}(0, n) - b_{i,s}(1, n),\ n = 0, 1, \ldots, N_j - 1 \end{aligned} \tag{4.5}$$

$$\begin{aligned} D_h^1\left(j, \mathbf{b}_{i,s}, m\right) \ = \ & \hat{x}(m + k_j, l_j - 2) - \hat{x}(m + k_j, l_j - 1) + \\ & b_{i,s}(m, 0) - b_{i,s}(m, 1),\ m = 0, 1, \ldots, M_j - 1 \end{aligned} \tag{4.6}$$

Estas definições são, na verdade, aproximações para as derivadas de primeira e segunda ordens, respectivamente.

Define-se, então, a métrica de *rugosidade* [12] como

$$\begin{aligned} R(\mathbf{b}_{i,s}, j) = \quad & \sum_{n=0}^{N_j - 1} \left| \alpha D_v^0\left(j, \mathbf{b}_{i,s}, n\right) + \beta D_v^1\left(j, \mathbf{b}_{i,s}, n\right) \right| + \\ & \sum_{m=0}^{M_j - 1} \left| \gamma D_h^0\left(j, \mathbf{b}_{i,s}, m\right) + \theta D_h^1\left(j, \mathbf{b}_{i,s}, m\right) \right| \end{aligned} \tag{4.7}$$

onde $\mathbf{b}_{i,s}$ é o i-ésimo elemento do dicionário na escala s. A métrica específica uti-

lizada é

$$R(\mathbf{b}_{i,s}, j) = \sum_{n=0}^{N_j-1} \left| \left\lfloor \frac{\hat{U}^j(M_j-2,n) - \hat{U}^j(M_j-1,n) + b_{i,s}(0,n) - b_{i,s}(1,n)}{2} \right\rfloor + b_{i,s}(0,n) - \hat{U}^j(M_j-1,n) \right| + \sum_{m=0}^{M_j-1} \left| \left\lfloor \frac{\hat{L}^j(m,N_j-2) - \hat{L}^j(m,N_j-1) + b_{i,s}(m,0) - b_{i,s}(m,1)}{2} \right\rfloor + b_{i,s}(m,0) - \hat{L}^j(m,N_j-1) \right|. \quad (4.8)$$

A rugosidade foi concebida observando-se que o algoritmo deveria preservar a continuidade das estruturas básicas através dos blocos, tais como regiões constantes, retas e superfícies com comportamento modelado por funções quadráticas. Os fatores multiplicativos α, β, γ e θ foram escolhidos para uma melhor adaptação a essas estruturas, levando-se em consideração seu comportamento em imagens reais. Normalmente, supondo-se estruturas simétricas, $\alpha = \gamma$ e $\beta = \theta$. Nos resultados apresentados neste trabalho, $\alpha = \gamma = -1$ e $\beta = \theta = 1/2$.

Portanto, quanto menor é a rugosidade de um elemento, maior é a sua probabilidade num dado contexto, desde que o dado em questão seja suave (assume-se este comportamento para a fonte, que é algo razoável quando se trata de imagens naturais). Assim, utilizam-se as colunas na esquerda de $\hat{\mathbf{L}}^j$ e as linhas inferiores de $\hat{\mathbf{U}}^j$ para se avaliar os "melhores" blocos de $\mathcal{D}_s$ para a codificação de $\mathbf{X}^j$, ou seja, aqueles que proporcionam a melhor continuidade com relação à vizinhança causal. O contexto é escolhido de acordo com estas regras e antes de cada tentativa de casamento. Nos casos em que o sub-bloco em questão estiver localizado nas fronteiras de $\mathbf{X}^0$, $\hat{\mathbf{L}}^j$ e $\hat{\mathbf{U}}^j$ deverão ser obtidos dos blocos codificados anteriormente, se os mesmos existirem.

A rugosidade não é a única métrica utilizada para se montar os contextos. Observa-se que, em geral, quanto maior a rugosidade de um bloco, menor tende a ser sua quantidade de detalhes (bordas ou transições dentro do bloco). Dada esta constatação, se houver dois blocos, um com baixo e outro com alto nível de detalhes, com aproximadamente as mesmas rugosidades para os elementos de $\mathcal{D}$, o bloco com menor nível de detalhes tende a utilizar um dicionário com elementos de baixo conteúdo de detalhes, o que corresponderia a um elemento de menor rugosidade.

Além disso, o número de elementos em um dicionário com baixo nível de detalhes é normalmente reduzido. Desta maneira, para se codificar este elemento com uma taxa baixa, é necessário um dicionário com poucos elementos e altas probabilidades de ocorrência. Já para elementos com um alto nível de detalhes, é melhor um dicionário com alto nível de detalhes, o que corresponde a muitos elementos de alta rugosidade. Por outro lado, para se codificar um bloco com alto nível de detalhes, é interessante escolher um elemento que também contenha um alto nível de detalhes, o que corresponderia a um elemento de alta rugosidade. Entretanto, o número de elementos com alto nível de detalhes no dicionário tende a ser grande. Portanto, para se codificar um elemento com grande quantidade de detalhes, é necessário um dicionário com alto nível de detalhes, o que corresponde a muitos elementos de alta rugosidade e apresentando altas probabilidades de ocorrência. Isto sugere que para se determinar as probabilidades dos elementos do dicionário em cada contexto, necessita-se, além da rugosidade, de outra métrica para se medir o nível de detalhes de um bloco. Uma possível solução seria medir a atividade no domínio da freqüência e transmitir a mesma como informação auxiliar, para cada bloco codificado. Apesar de factível, esta solução geraria um *overhead* inaceitável. Para que isto seja possível sem a necessidade de informação auxiliar, ou seja, baseando-se na vizinhança causal, desenvolveu-se a métrica de *atividade*, como definido na equação (4.9) para vetores de duas dimensões. A atividade foi escolhida por levar em consideração a quantidade de transições através da vizinhança causal de um bloco, consistindo numa medida razoável da complexidade de um sinal. Um bloco com alta complexidade necessita de um contexto no qual elementos de alta rugosidade tenham probabilidades de ocorrência suficientemente altas, ocasionando uma codificação correta do bloco. Para um bloco de imagem $\mathbf{X}$ de dimensões $M \times N$, a atividade é calculada como

$$A\left(\mathbf{X}\right) = \max_{m,n} \left\{ \left(\sum_{k=0}^{N-2} |x(m,k+1) – x(m,k)| \right), \left(\sum_{p=0}^{M-2} |x(p+1,n) – x(p,n)| \right) \right\}. \tag{4.9}$$

Dada a discussão acima, a probabilidade de um elemento do dicionário, em um dado contexto, deve ser uma função não crescente da rugosidade e uma função não decrescente da atividade dos vizinhos causais de um bloco. Em outras palavras, a função $\mathcal{W}(\cdot)$ deve ser uma função não crescente da rugosidade $R(\mathbf{b}_{i,s}, j)$ e uma

função não decrescente das atividades $A(\hat{\mathbf{U}}^j)$ e $A(\hat{\mathbf{L}}^j)$. Sendo assim, a equação (4.2) pode ser aproximada por

$$\Pr\left\{\mathbf{b}_{i,s}|\hat{\mathbf{U}}^j,\hat{\mathbf{L}}^j\right\} = \mathcal{W}\left(R(\mathbf{b}_{i,s},j), A(\hat{\mathbf{U}}^j), A(\hat{\mathbf{L}}^j)\right)\Pr\left\{\mathbf{b}_{i,s}\right\}. \tag{4.10}$$

É importante salientar que, em imagens suaves, a função $\mathcal{W}(\cdot)$ tenderá a diminuir muito as probabilidades de elementos com alta rugosidade. Além disso, o número desses elementos de baixa probabilidade será dependente da atividade da vizinhança causal - quanto maior a atividade, menor o número de elementos com probabilidade baixa. Dado que esses elementos com alta rugosidade têm um custo alto, dificilmente os mesmos serão escolhidos para se codificar algum bloco. Portanto, não haveria uma perda de desempenho significativa se as suas probabilidades fossem aproximadas para zero, o que equivale a retirá-los do dicionário $\mathcal{D}_s$. Assim, a aproximação de probabilidades muito baixas para zero significa que o algoritmo está utilizando um dicionário reduzido $\mathcal{D}_s^j \subset \mathcal{D}_s$ para a codificação de $\mathbf{X}^j$.

O número de elementos de $\mathcal{D}_s^j$, definido como N_s^j, deve ser uma função não crescente das atividades da vizinhança causal. A principal vantagem dessa abordagem, ou seja, utilizar $\mathcal{D}_s^j$ ao invés de $\mathcal{D}_s$, reside no fato de que um número menor de vetores será varrido a cada tentativa de codificação, reduzindo a complexidade global do processo.

Mais especificamente, o dicionário reduzido $\mathcal{D}_s^j$ é composto pelos N_s^j elementos menos rugosos, onde N_s^j é uma função não decrescente das atividades dos blocos $\hat{\mathbf{L}}^j$ e $\hat{\mathbf{U}}^j$. Logo, partindo-se da equação (4.10), o modelo de probabilidades para a codificação de $\mathbf{X}^j$ torna-se

$$\Pr\left\{\mathbf{b}_{i,s}|\hat{\mathbf{U}}^j,\hat{\mathbf{L}}^j\right\} = \begin{cases} \mathcal{W}\left(R(\mathbf{b}_{i,s},j), A(\hat{\mathbf{U}}^j), A(\hat{\mathbf{L}}^j)\right)\Pr\left\{\mathbf{b}_{i,s}\right\} & ,\mathbf{b}_{i,s} \in \mathcal{D}_s^j \\ 0 & ,\mathbf{b}_{i,s} \notin \mathcal{D}_s^j. \end{cases} \tag{4.11}$$

Há várias alternativas para a função de ponderação $\mathcal{W}(\cdot)$ na equação (4.11), sendo a mais simples considerá-la uma constante. Isso seria equivalente a utilizar um dicionário com os N_s^j elementos menos rugosos de $\mathcal{D}_s$ com probabilidades proporcionais a $P_{\mathcal{D}_s}(\mathbf{b}_{i,s})$, como na equação (4.12) abaixo, onde $C = \sum_{\mathbf{b}_{i,s}\in\mathcal{D}_s^j} \Pr\left\{\mathbf{b}_{i,s}\right\}$.

$$\Pr\left\{\mathbf{b}_{i,s}|\hat{\mathbf{U}}^j,\hat{\mathbf{L}}^j\right\} = \begin{cases} C^{-1}\Pr\left\{\mathbf{b}_{i,s}\right\} & ,\mathbf{b}_{i,s} \in \mathcal{D}_s^j \\ 0 & ,\mathbf{b}_{i,s} \notin \mathcal{D}_s^j \end{cases} \tag{4.12}$$

O número de elementos N_s^j do dicionário reduzido $\mathcal{D}_s^j$ é dado por

$$N_s^j = \max\left\{ N_{\max}\left(\frac{A(\hat{\mathbf{U}}^j) + A(\hat{\mathbf{L}}^j)}{2}\right) A_{\max}^{-1}, 16 \right\} \tag{4.13}$$

onde $N_{\max}$ é o tamanho máximo permitido para o dicionário reduzido, $A(\hat{\mathbf{L}}^j)$ e $A(\hat{\mathbf{U}}^j)$ são dados por (4.9), e $A_{\max} = \max_j \left\{A(\mathbf{X}^j)\right\}$, ou seja, o valor máximo de $A(\mathbf{X}^j)$ para toda a imagem (calculado com blocos de mesmas dimensões de $\mathbf{X}^0$). $N_{\max}$, o máximo tamanho para o dicionário reduzido, é calculado como

$$N_{\max} = \begin{cases} \left\lfloor \dfrac{A_{\text{final}}.4999}{285} \right\rfloor + 1, & A_{\text{final}} < 285 \\ \left\lfloor \dfrac{(A_{\text{final}} - 285)\,.6000}{39} \right\rfloor + 5000, & A_{\text{final}} \geq 285, \end{cases} \tag{4.14}$$

onde $A_{\text{final}} = \left\lfloor \frac{A_{\max}}{4} \right\rfloor + A_{\text{avg}}$. $A_{\text{avg}} = E\left[A(\mathbf{X}^j)\right]$ é o valor médio para todas as atividades da imagem, calculado do mesmo modo que $A_{\max}$. Tais fórmulas foram obtidas através de simulações com as imagens de teste e refletem o seu comportamento com relação às medidas adotadas (ver [93]). É importante ressaltar que embora $A_{\max}$ e A_{final} devam ser transmitidos como informação auxiliar, o *overhead* oriundo dos mesmos é desprezível, desde que sejam calculados uma única vez para a imagem.

O procedimento empregado acima para a escolha dos elementos formadores de $\mathcal{D}_s^j$, para a codificação de $\mathbf{X}^j$, é essencialmente o mesmo empregado em vários esquemas de *quantização vetorial com casamento lateral* presentes na literatura [94–97]. O *Side-Match MMP* (SM-MMP) descrito em [12, 93], por exemplo, corresponde ao caso particular descrito na equação (4.12).

Desde que o dado codificado seja suave, outras boas alternativas para $\mathcal{W}(\cdot)$ são aquelas que aumentam as probabilidades dos elementos menos rugosos em relação aos mais rugosos. Uma opção, que foi utilizada nas simulações desta seção e resultou em um bom desempenho taxa-distorção, consiste em

$$\Pr\left\{\mathbf{b}_{i,s} | \hat{\mathbf{U}}^j, \hat{\mathbf{L}}^j\right\} = \begin{cases} C^{-1} e^{-\frac{1}{2}\left(\frac{R(\mathbf{b}_{i,s},j)}{\frac{A(\hat{\mathbf{U}}^j)+A(\hat{\mathbf{L}}^j)}{2}}\right)^2} \Pr\left\{\mathbf{b}_{i,s}\right\} & , \mathbf{b}_{i,s} \in \mathcal{D}_s^j \\ 0 & , \mathbf{b}_{i,s} \notin \mathcal{D}_s^j, \end{cases} \tag{4.15}$$

onde C é escolhido tal que $\sum_{\mathbf{b}_{i,s} \in \mathcal{D}_s^j} \Pr\left\{\mathbf{b}_{i,s} | \hat{\mathbf{U}}^j, \hat{\mathbf{L}}^j\right\} = 1$.

Novamente, o número de elementos N_s^j do dicionário reduzido é dado por (4.13). A fórmula final para a função de ponderação é baseada na observação de que elementos menos rugosos são utilizados com maior probabilidade e essa dependência é

bem modelada com uma função gaussiana. Além disso, a atividade pode ser vista com uma medida de espalhamento, como a variância, definindo a complexidade em se encontrar uma aproximação conveniente para o bloco atual.

A função de ponderação descrita pela equação (4.15) satisfaz as exigências básicas de uma boa função de ponderação, ou seja, ela é tanto uma função não crescente com a rugosidade quanto não decrescente com as atividades.

A abordagem apresentada não se restringe apenas a imagens suaves, podendo incorporar qualquer modelo de fonte, substituindo-se as rugosidades e as atividades por algo mais conveniente.

As partições de dicionário

O algoritmo MMP atualiza o dicionário enquanto os dados de entrada são codificados, com versões expandidas e contraídas de concatenações de blocos previamente processados. Mais especificamente, supondo-se dois nós $\mathbf{X}^{2k+1}$ e $\mathbf{X}^{2k+2}$, da Figura 2.3 percebe-se que a sua escala é $s_i = \lfloor \log_2(k+1) \rfloor$, onde $\lfloor a \rfloor$ é o maior inteiro menor ou igual a a. Deste modo, sempre que $\mathbf{X}^{2k+1}$ for codificado com $\mathbf{b}_{i_{2k+1},s_i}$ e $\mathbf{X}^{2k+2}$ com $\mathbf{b}_{i_{2k+2},s_i}$, então o dicionário será atualizado com $\mathbf{b}_{i_k,s_i-1}$, a concatenação de $\mathbf{b}_{i_{2k+1},s_i}$ e $\mathbf{b}_{i_{2k+2},s_i}$ (esse novo elemento pertencerá a escala $s_i - 1$).

Como já foi mencionado no capítulo 2, se a abordagem multidicionários for utilizada e o bloco de entrada tiver dimensões $M_0 \times M_0$, haverá $(1+2\log_2 M_0)$ cópias do dicionário, uma para cada escala s (ver a discussão no final da seção 2.1). Sempre que um novo elemento $\mathbf{b}_{i_j,s_j}$ precisa ser incluído nos dicionários $\mathcal{D}_s$, $s = 0, 1, \ldots, \log_2 M_0$, a sua escala é mudada para s, utilizando-se a transformação de escala (ver a seção 2.1). Isto significa que cada elemento em qualquer escala s é, na verdade, uma transformação de um elemento de qualquer das escalas $(1 + 2\log_2 M_0)$. Essa informação, indicando para cada elemento a sua escala de origem, está disponível tanto para o codificador quanto para o decodificador.

Assim, pode-se inferir que a atualização de dicionário no MMP impõe uma estrutura bem definida a todos os dicionários, como se cada um deles pudesse ser dividido em sub-dicionários contendo elementos que foram transformados de uma

certa escala. Define-se, então, uma *partição* do dicionário na escala s, $\mathcal{D}_s$, como:

$$\mathcal{D}_s = \bigcup_{k=0}^{2\log_2 M} \mathcal{D}_{s,k}, \quad \text{com} \quad \mathcal{D}_{s,k} \cap \mathcal{D}_{s,l} = \emptyset, \quad k \neq l \tag{4.16}$$

onde cada sub-dicionário $\mathcal{D}_{s,k}$ é composto pelos elementos de $\mathcal{D}_s$ que foram obtidos através do escalonamento de elementos originários da escala k, formada pela concatenação de dois elementos na escala $k+1$.

Se as estatísticas de imagens naturais, como ilustrado na Tabela 4.1 para a imagem *Lena* (ver o apêndice B), forem analisadas, para cada escala, é possível se identificar um padrão específico na escolha dos elementos para a codificação de cada bloco $\mathbf{X}^j$. Por exemplo, a Tabela 4.1 mostra que 52,88% dos blocos de dimensões 2×2 que foram utilizados na codificação da imagem *Lena* foram obtidos pela contração de blocos de dimensões 4×2. Sendo assim, fica claro que a maioria das escalas apresenta uma origem preferencial para os blocos escolhidos.

Tabela 4.1: Porcentagem (%) de blocos utilizados na escala s do dicionário $\mathcal{D}_s$ com origem na escala k, para blocos de entrada de dimensões 16×16 da imagem Lena.

Escala	Escala s								
k	1×1	2×1	2×2	4×2	4×4	8×4	8×8	16×8	16×16
1×1	100,00	30,38	0,18	3,75	6,32	14,55	13,69	15,95	9,17
2×1	0,00	45,85	0,67	1,32	0,92	0,76	3,38	1,90	0,83
2×2	0,00	12,85	17,12	4,37	10,24	15,43	9,34	8,10	16,67
4×2	0,00	0,14	52,88	7,47	6,41	10,83	11,43	7,38	5,00
4×4	0,00	0,00	21,98	53,56	23,34	15,54	11,27	13,10	19,16
8×4	0,00	1,94	1,85	17,69	14,81	11,87	11,75	6,43	9,17
8×8	0,00	8,56	2,52	5,53	29,98	23,57	23,35	29,05	10,84
16×8	0,00	0,14	2,02	2,44	4,15	2,68	7,09	7,38	15,83
16×16	0,00	0,14	0,78	3,87	3,83	4,77	8,70	10,71	13,33

Uma possível maneira de se explorar este comportamento consiste em separar a representação de um elemento no dicionário $\mathcal{D}_s$ em duas componentes: *escala de origem* e *índice*, o que pode levar a uma redução na taxa necessária para a sua codificação. Considerando a partição na equação (4.16), os elementos de dicionário são

então indexados em duas etapas: primeiramente o sub-dicionário $\mathcal{D}_{s,k}$ é escolhido, definindo a escala de origem como k, e depois o índice i_j do melhor elemento em $\mathcal{D}_{s,k}$ para representar o bloco de entreada $\mathbf{X}^j$ é identificado. Devido à partição na equação (4.16), cada dicionário terá um número de elementos menor, fazendo com que os mesmos sejam endereçados com um número menor de bits. Além disso, se em alguma escala s alguns sub-dicionários forem utilizados mais freqüentemente que outros, a escolha poderá ser realizada com um número de bits menor que o logaritmo do número de sub-dicionários. Isso pode levar a uma redução global na taxa necessária para se especificar os nós n_j da árvore de segmentação $\mathcal{S}$, desde que se utilizem contextos adaptativos.

De modo a especificar a taxa para se codificar $\mathbf{b}_{i_j,s_j}$, o índice do i_j-ésimo elemento do dicionário na escala s_j, dentro de $\mathcal{D}_{s,k}$, a expressão dada na equação (2.2) torna-se

$$R(n_j) = -\left[\log_2\left(\Pr\left\{\mathcal{D}_{s_j,k}\right\}\right) + \log_2\left(\Pr\left\{\mathbf{b}_{i_j,s_j}|k\right\}\right)\right], \tag{4.17}$$

onde $\Pr\left\{\mathcal{D}_{s_j,k}\right\}$ é a probabilidade de que o elemento escolhido para codificar $\mathbf{X}^j$ pertença ao sub-dicionário na escala s_j com elementos transformados da escala k e $\Pr\left\{\mathbf{b}_{i_j,s_j}|k\right\}$ é a probabilidade do vetor $\mathbf{b}_{i_j}$ de $\mathcal{D}_{s,k}$ ser escolhido para codificar uma dado bloco de sinal. Seguindo o conceito apresentado anteriormente, pode-se também gerar contextos adaptativos para as partições de dicionário, utilizando-se um modelo multiplicativo similar ao da equação (4.11), mas agora para as probabilidades na equação (4.17). Definindo-se $\Pr\left\{\mathbf{b}_{i,s}|\hat{\mathbf{U}}^j,\hat{\mathbf{L}}^j,k\right\}$ como a probabilidade de que o vetor $\mathbf{b}_{i,s}$ seja escolhido do sub-dicionário k na escala s, durante a codificação do bloco $\mathbf{X}_j$, e $\Pr\left\{\mathcal{D}_{s,k}|\hat{\mathbf{U}}^j,\hat{\mathbf{L}}^j\right\}$ como a probabilidade de que o sub-dicionário $\mathcal{D}_{s,k}$ seja escolhido durante a codificação de $\mathbf{X}_j$, têm-se os seguintes contextos:

$$\Pr\left\{\mathbf{b}_{i,s}|\hat{\mathbf{U}}^j,\hat{\mathbf{L}}^j,k\right\} = \begin{cases} \mathcal{W}\left(R(\mathbf{b}_{i,s},j),A(\hat{\mathbf{U}}^j),A(\hat{\mathbf{L}}^j)\right)\Pr\left\{\mathbf{b}_{i,s}|k\right\} & ,\mathbf{b}_{i,s}\in\mathcal{D}^j_{s,k} \\ 0 & ,\mathbf{b}_{i,s}\in\mathcal{D}^j_{s,k}. \end{cases} \tag{4.18}$$

$$\Pr\left\{\mathcal{D}_{s,k}|\hat{\mathbf{U}}^j,\hat{\mathbf{L}}^j\right\} = \begin{cases} C\Pr\left\{\mathcal{D}_{s,k}\right\} & ,\mathcal{D}^j_{s,k}\neq\emptyset \\ 0 & ,\mathcal{D}^j_{s,k}=\emptyset. \end{cases} \tag{4.19}$$

onde C é escolhido tal que $\sum_k \Pr\left\{\mathcal{D}_{s,k}|\hat{\mathbf{U}}^j,\hat{\mathbf{L}}^j\right\} = 1$.

Melhorias no algoritmo base

Foi observado que, em geral, quanto mais seqüências típicas são incluídas no dicionário, melhor é o desempenho do MMP [13,93]. Em outras palavras, quanto mais rapidamente o dicionário cresce, melhor é o resultado obtido, desde que o mesmo cresça com "elementos úteis". Com base nessas observações, para que o desempenho do algoritmo proposto seja melhorado, deve-se modificar o procedimento de atualização, fazendo com que o algoritmo aprenda não apenas as concatenações dos blocos, mas também versões deslocadas dos mesmos. Isto permite ao dicionário $\mathcal{D}_s$ crescer mais rapidamente, reduzindo a distorção média. Ao se utilizarem os dicionários reduzidos, a taxa média não é afetada de modo significativo, pois o tamanho N_s^j do dicionário $\mathcal{D}_s^j$ é independente do tamanho de $\mathcal{D}_s$ (ver a equação (4.13)).

Com a nova abordagem, até 10 novos elementos podem ser incluídos, através de deslocamentos para a esquerda e para cima (vizinhança causal). Supondo-se que o dicionário seja atualizado com a inclusão da concatenação de $\hat{\mathbf{X}}^{13}$ e $\hat{\mathbf{X}}^{14}$, como mostrado na Figura 4.3, no algoritmo original a única atualização seria o bloco quadrado com o canto superior esquerdo em $\hat{\mathbf{x}}(4,4)$ e o conto inferior direito em $\hat{\mathbf{x}}(7,7)$, representado por $\{\hat{\mathbf{x}}(4,4), \hat{\mathbf{x}}(7,7)\}$. Entretanto, outros elementos podem ser incluídos, tais como $\{\hat{\mathbf{x}}(3,4), \hat{\mathbf{x}}(6,7)\}$, $\{\hat{\mathbf{x}}(2,4), \hat{\mathbf{x}}(5,7)\}$ e $\{\hat{\mathbf{x}}(1,4), \hat{\mathbf{x}}(4,7)\}$, que correspondem a deslocamentos verticais de $\frac{M}{4}$, $\frac{M}{2}$ e $\frac{3M}{4}$. Deste modo, o algoritmo pode aprender muitas versões deslocadas de um novo elemento, proporcionando uma adaptação mais rápida ao sinal em codificação, que pode levar diretamente a uma redução de custo. O mesmo procedimento pode ser aplicado a deslocamentos horizontais e diagonais, resultando em até 10 novos elementos. Se a concatenação a ser incluída tiver dimensões $M \times N$ e for representada por $\{\hat{\mathbf{x}}(k,l), \hat{\mathbf{x}}(k+M-1, l+N-1)\}$, os blocos deslocados podem ser representados por $\{\hat{\mathbf{x}}(k-\delta_v, l-\delta_h), \hat{\mathbf{x}}(k-\delta_v+M-1, l-\delta_h+N-1)\}$, onde os deslocamentos vertical e horizontal δ_v e δ_h, respectivamente, são dados na Tabela 4.2 ($\lfloor x \rfloor$ significa o maior inteiro menor ou igual a x).

Tabela 4.2: Deslocamentos a serem incluídos no dicionário.

δ_v	0	$\lfloor \frac{M}{4} \rfloor$	$\lfloor \frac{M}{2} \rfloor$	$\lfloor \frac{3M}{4} \rfloor$	0	0	0	$\lfloor \frac{M}{4} \rfloor$	$\lfloor \frac{M}{2} \rfloor$	$\lfloor \frac{3M}{4} \rfloor$
δ_h	0	0	0	0	$\lfloor \frac{N}{4} \rfloor$	$\lfloor \frac{N}{2} \rfloor$	$\lfloor \frac{3N}{4} \rfloor$	$\lfloor \frac{N}{4} \rfloor$	$\lfloor \frac{N}{2} \rfloor$	$\lfloor \frac{3N}{4} \rfloor$

É importante ressaltar que para pequenos blocos o número de novos elementos adicionados pode ser inferior a 10. Por exemplo, na escala de dimensões 2×1 há apenas os deslocamentos $(0, 0)$ e $(1, 0)$ (significam (δ_v, δ_h)). Sempre que o sub-bloco estiver localizado nas fronteiras do bloco de entrada, algumas partes dos deslocamentos deverão ser obtidas dos blocos vizinhos anteriormente codificados, se os mesmos existirem.

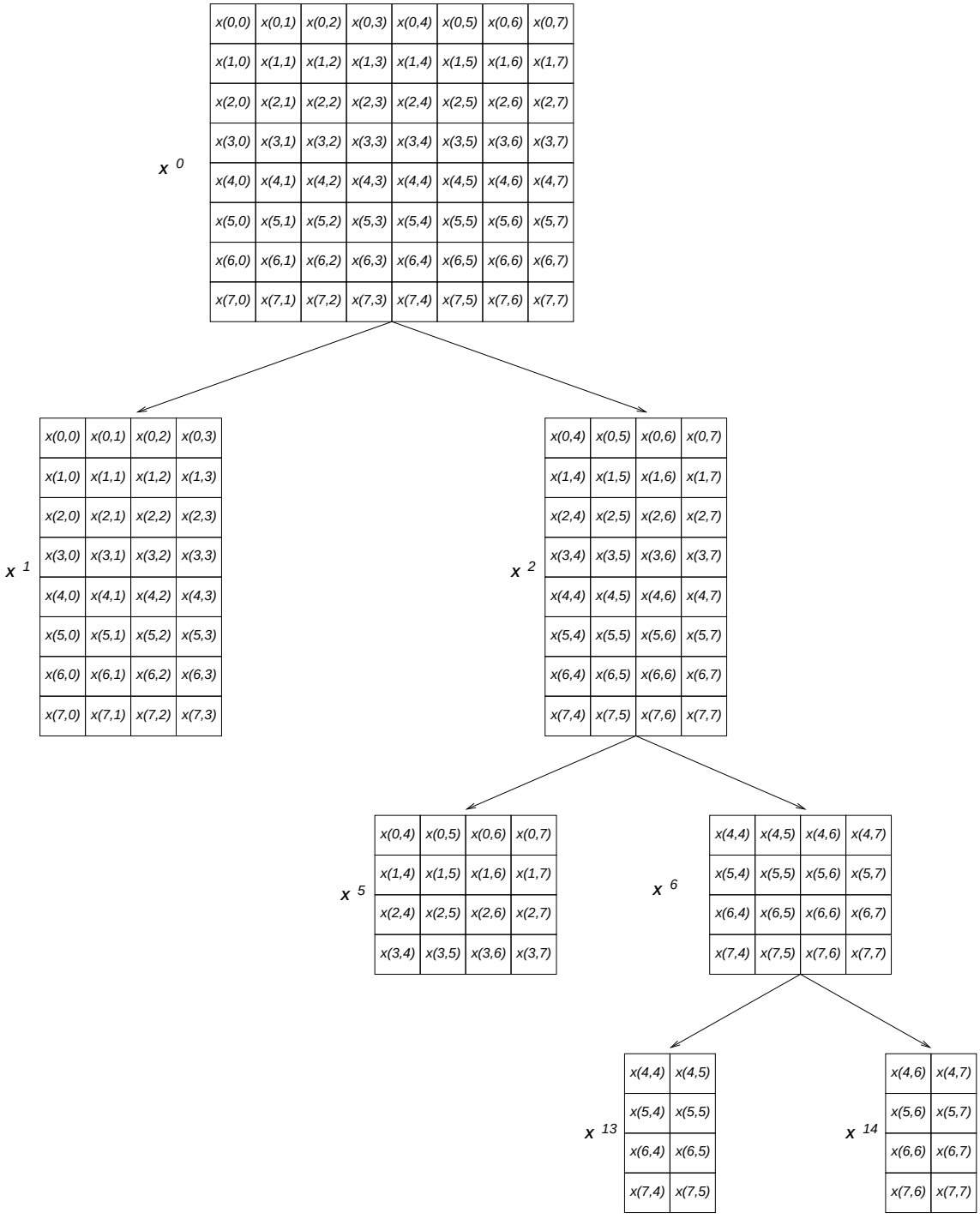


Figura 4.3: Deslocamentos para atualização do dicionário.

Resultados de simulações

A extensões propostas foram implementadas e identificadas como o algoritmo *Adaptive Probability Model MMP* (APM-MMP), com o objetivo de comprimir imagens em níveis de cinza. As imagens utilizadas foram: *Lena*, *Peppers*, *PP1205* e *PP1209*, de dimensões 512×512, e *Einstein* e *Cameraman*, de dimensões 256×256. Todas podem ser encontradas no apêndice B. As simulações foram realizadas com o MMP [10], o APM-MMP e o JPEG2000 [70], para possibilitar uma comparação direta dos resultados. Algumas imagens também apresentam resultados para o SM-MMP, introduzido em [93].

As imagens de entrada $x(m,n) : m,n = 0,1,2,3 \ldots 2^k - 1$ são segmentadas em blocos de dimensões $M \times M$, com $M = 16$. Assim, os sinal de entrada $x(m,n)$ é primeiramente processado como uma seqüência de blocos de dimensões 16×16

$$x(m,n) = \begin{pmatrix} \mathbf{X}^0_{0,0} & \cdots & \mathbf{X}^0_{0,2^k/16-1} \\ \vdots & \ddots & \vdots \\ \mathbf{X}^0_{2^k/16-1,0} & \cdots & \mathbf{X}^0_{2^k/16-1,2^k/16-1} \end{pmatrix}, \tag{4.20}$$

onde

$$\mathbf{X}^0_{m,n} = \begin{pmatrix} x(16m,16n) & \ldots & x(16m,16n+15) \\ \vdots & \ddots & \vdots \\ x(16m+15,16n) & \ldots & x(16m+15,16n+15) \end{pmatrix}. \tag{4.21}$$

Cada bloco $\mathbf{X}^0_{m,n} : m,n = 0,1,2,\ldots 2^k/16 - 1$ é codificado independentemente, utilizando-se um ou mais elementos do dicionário $\mathcal{D}$, que é inicializado com $\mathcal{D}^0 = \{x_{min}, x_{min}+2, x_{min}+4, \ldots, x_{max}\}$, onde x_{min} e x_{max} são os valores mínimo e máximo das amostras de $x(n)$, respectivamente. Após a codificação de cada bloco $\mathbf{X}^0_{m,n}$, o dicionário resultante $\mathcal{D}^{m,n}$ é mantido e utilizado como dicionário inicial para o próximo bloco $\mathbf{X}^0_{m,n+1}$ ou $\mathbf{X}^0_{m+1,0}$. Se $x(m,n)$ for muito grande, o dicionário resultante também o será. De modo a lidar com limitações físicas dos recursos computacionais disponíveis, o tamanho máximo do dicionário é limitado em 400000 elementos. Quando este limite é atingido, o elemento mais antigo do dicionário é descartado, permitindo a inclusão de um mais novo (o elemento mais antigo é aquele que não foi utilizado pelo maior período de tempo). Devido ao critério de continuidade, ao se tentar codificar o bloco de entrada $\mathbf{X}^0_{m,n}$, os blocos previamente

codificados $\hat{\mathbf{X}}^0_{m-1,n}$ e $\hat{\mathbf{X}}^0_{m,n-1}$ podem ser utilizados na definição do dicionário reduzido dos sub-blocos localizados nas bordas superior e esquerda, desde que os vizinhos estejam nos blocos de entrada anteriores.

As Figuras 4.4 a 4.6 mostram o desempenho taxa-distorção dos três algoritmos. Após uma breve análise, pode-se verificar que:

i) O APM-MMP supera o MMP original para todas as imagens.

ii) O APM-MMP supera o JPEG2000 para a imagem *Cameraman* por $\approx$ 1,0 dB a 0,6 bits/pixel (bpp).

iii) O APM-MMP supera o JPEG2000 por $\approx$ 2,5dB para a imagem *PP1209* e por $\approx$ 4,5 dB para a imagem *PP1205*, ambas a 0,6 bpp.

iv) O APM-MMP supera o JPEG2000 para a imagem *Einstein* por $\approx$ 0,2 dB a 0,5 bpp.

v) O JPEG2000 supera o APM-MMP para a imagem *Peppers* por $\approx$ 0,15 dB a 0,6 bpp.

vi) Com relação à imagem *Lena*, o APM-MMP não apresentou o mesmo desempenho que o JPEG2000.

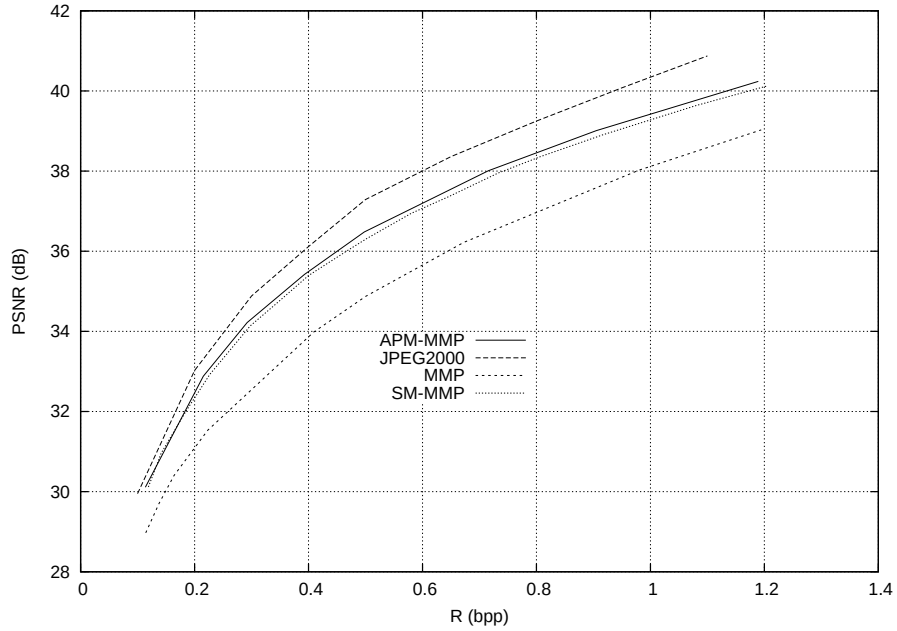
As figuras mostram que o APM-MMP proporciona um ganho expressivo sobre o MMP original para imagens suaves. Isto sugere que os contextos introduzidos nesta seção foram eficazes, melhorando o desempenho do algoritmo para essa classe de imagens. Por exemplo, o ganho em PSNR para a imagem *Lena*, a 0,5 bpp, ficou em torno de $\approx$ 1,6dB. Para a imagem *PP1209*, também são apresentados resultados com os esquemas de compressão de documentos em [98] e [99], conhecidos como AVC-C e MRC com AVC-I/JBIG2, respectivamente, assim como com o H.264 intra-quadros [69]. As figuras mostram que o MMP é competitivo em compressão de documentos, superando codificadores que representam o estado da arte.

É importante notar que este ganho de desempenho apresentado pelo APM-MMP para imagens suaves não ocasiona perdas em imagens com textos ou gráficos, apesar do modelo utilizado ser adequado às primeiras, o que reforça o comportamento universal do MMP. Na verdade, o desempenho para as imagens *PP1205* e *PP1209*

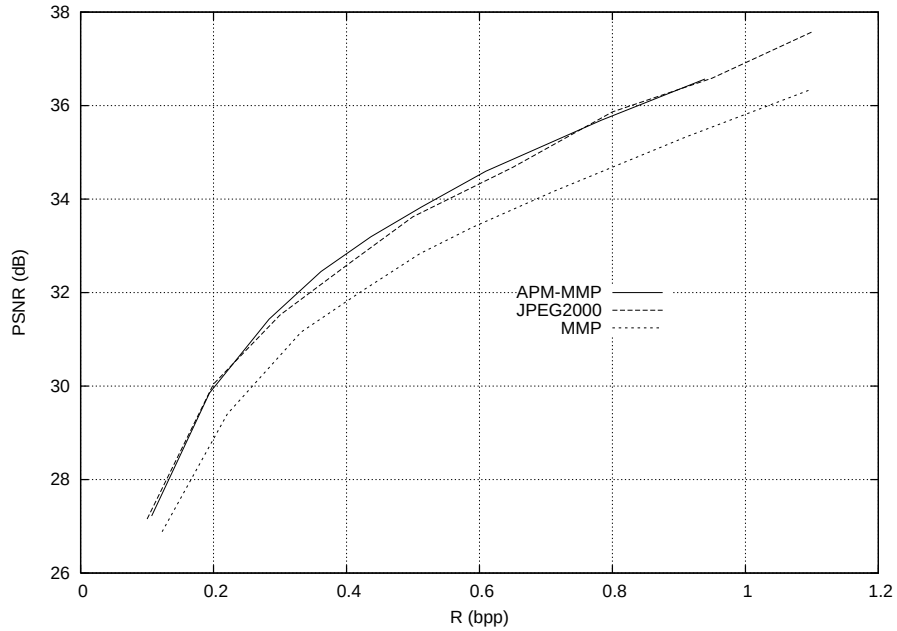
chegou até a aumentar. Pode-se argumentar que este modelo de probabilidades não é adequado a essas imagens, porém, nas áreas não suaves os valores das atividades são normalmente maiores, o que leva a dicionários maiores, compostos também por elementos de alta rugosidade (ver a discussão apresentada nesta seção, ao longo da introdução do modelo de probabilidades). Assim, o desempenho do MMP não é prejudicado pelo modelo de probabilidades. É importante notar que o desempenho obtido para as imagens *PP1205* e *PP1209* é, na maior parte, devido às partições de dicionário, que aumentam a eficácia dos contextos.

A Figura 4.7 mostra um detalhe da imagem *Lena* codificada pelo MMP original (na esquerda), APM-MMP (no centro), e JPEG2000 (na direita), todos a 0,3 bpp. A imagem codificada pelo APM-MMP apresenta qualidade superior quando comparada à codificada pelo MMP, o que é esperado devido ao grande aumento na PSNR ($\approx$ 1,3 dB nessa taxa). É possível também verificar que a blocagem foi bastante reduzida com o APM-MMP, indicando a eficácia do modelo proposto. A qualidade subjetiva da imagem Lena codificada pelo APM-MMP não é tão boa quando a apresentada pela versão codificada com o JPEG2000. Isto ocorre devido à ausência de grandes áreas homogêneas e transições agudas na imagem, que é basicamente composta por inúmeras transições suaves. Estas últimas são codificadas por esquemas baseados em *wavelets* com grande eficácia, além de serem um desafio para o MMP, dado que diversos gradientes diferentes devem ser aprendidos. Uma solução natural para este problema consiste em proporcionar um dicionário inicial mais rico para o MMP, com muitos blocos representando diversos gradientes diferentes.

A Figura 4.8 mostra um detalhe da imagem *PP1209* codificada pelo APM-MMP (no centro) e pelo JPEG2000 (na direita), ambos a 0,5 bpp. No MMP, a taxa de bits desejada é obtida variando-se o parâmetro λ, na equação (2.7). É possível perceber que a imagem codificada pelo APM-MMP também apresenta alta qualidade subjetiva, preservado as bordas das letras e dos gráficos. Para as duas versões comprimidas da imagem *Lena*, pode-se argumentar que o JPEG2000 proporciona uma melhor reconstrução, dado a ausência de blocagem. Entretanto, esse efeito já está presente na versão original desta imagem; na verdade, a codificação do JPEG2000 acaba mascarando tal comportamento.

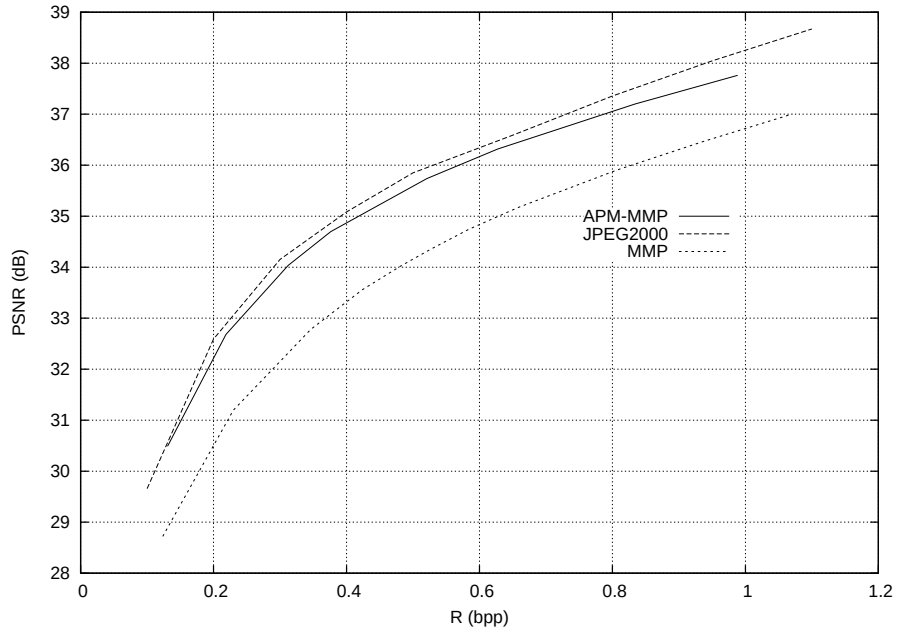


(a)

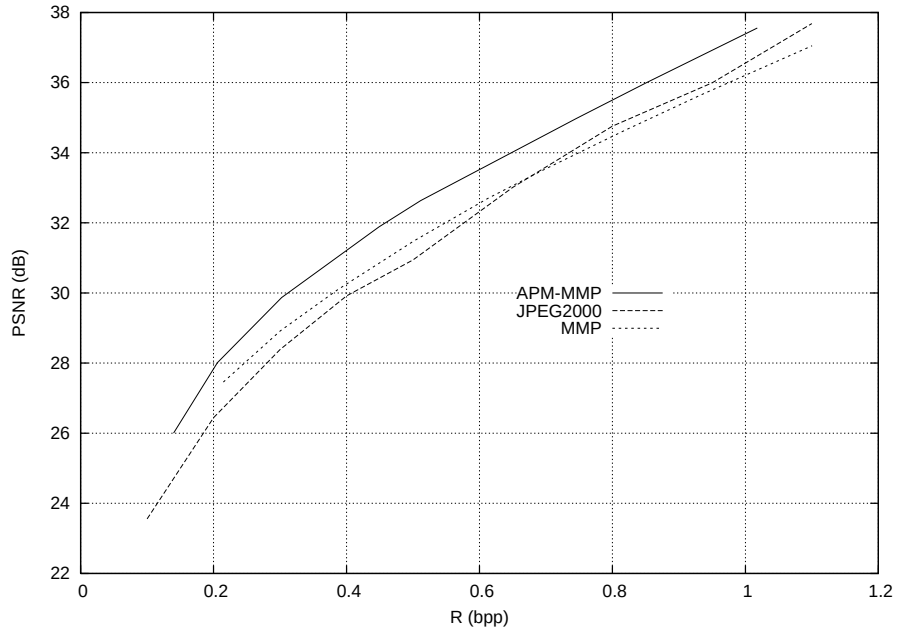


(b)

Figura 4.4: Desempenho taxa-distorção do APM-MMP, SM-MMP, MMP e JPEG2000 para: (a) *Lena* 512×512; (b) *Einstein* 256×256.

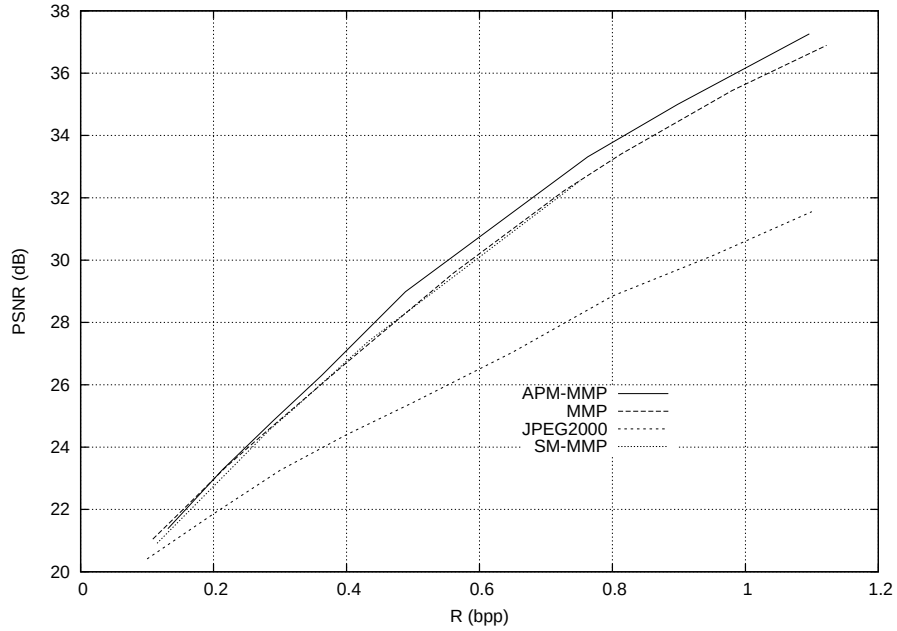


(a)

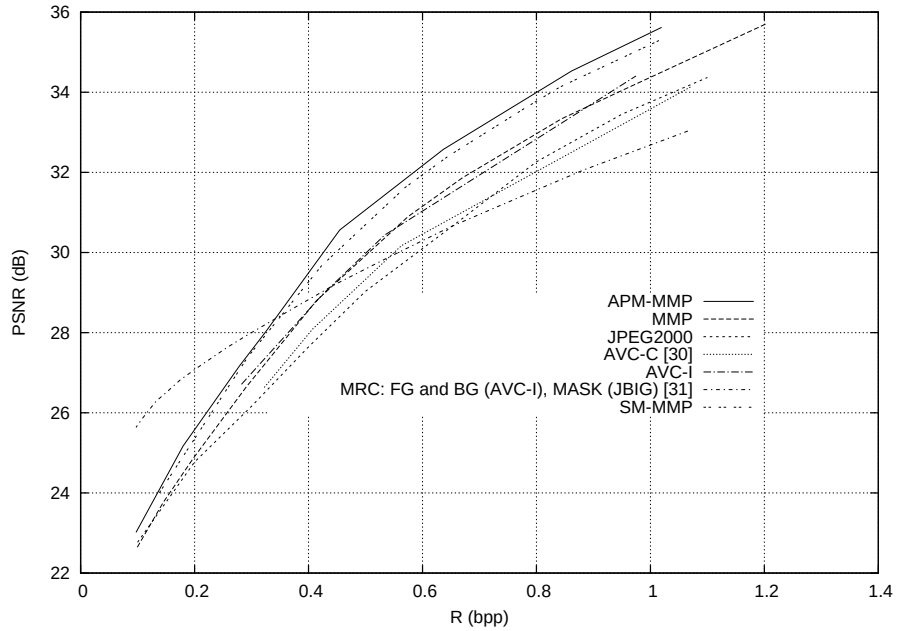


(b)

Figura 4.5: Desempenho taxa-distorção do APM-MMP, SM-MMP, MMP e JPEG2000 para: (a) *Peppers* 512×512; (b) *Cameraman* 256×256.



(a)



(b)

Figura 4.6: Desempenho taxa-distorção do APM-MMP, SM-MMP, MMP e JPEG2000 para: (a) *PP1205* 512×512; (b) *PP1209* 512×512.

Figura 4.7: *Lena* codificada a 0,3 bpp. MMP original (esquerda), APM-MMP (centro) e JPEG2000 (direita).

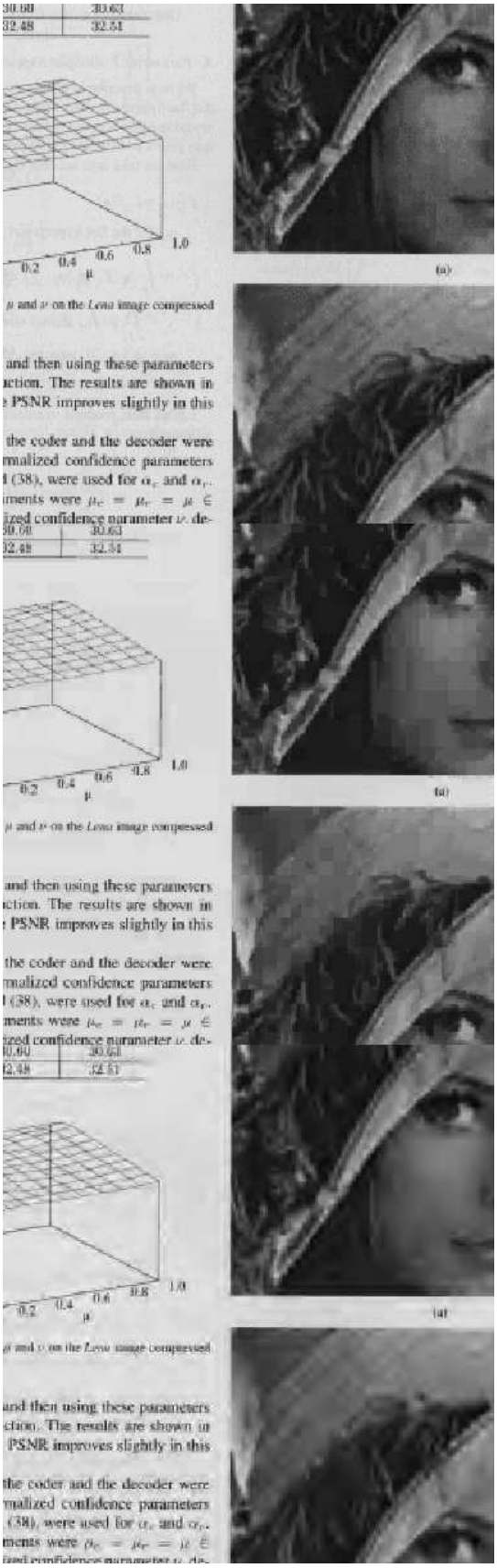


Figura 4.8: *PP1209* codificada a 0,5 bpp. Imagem original (esquerda), APM-MMP (centro) e JPEG2000 (direita).

4.1.2 O MMP com casamento lateral generalizado: GSM-MMP

No APM-MMP descrito na seção 4.1.1 com o modelo de contextos dado pela equação (4.12), daqui em diante chamado de *Adaptive Probability Model - Constant Model* MMP (APM-CM-MMP), cada bloco de sinal passa a ser codificado dependentemente dos seus vizinhos causais. Como os blocos de imagem $\mathbf{X}^0$ são varridos no sentido de leitura, ou seja, da esquerda para a direita e de cima para baixo, há apenas dois vizinhos causais disponíveis, conforme explicado na seção 4.1.1: $\mathbf{U}^0$ e $\mathbf{L}^0$. Entretanto, se o modo de varredura for modificado [94], mais vizinhos podem ser utilizados, como ilustrado na Figura 4.9. Esta é a base do novo esquema apresentado nesta seção, batizado de *Generalized Side-Match MMP* (GSM-MMP).

O casamento lateral generalizado

Dado que agora a vizinhança causal será estendida, faz-se necessário definir os novos vizinhos. Considera-se como o *vizinho inferior* $\mathbf{Lo}^j$ o bloco de dimensões $M_j \times N_j$ imediatamente abaixo de $\mathbf{X}^j$, e como o *vizinho direito* $\mathbf{R}^j$ o bloco de dimensões $M_j \times N_j$ imediatamente à direita de $\mathbf{X}^j$.

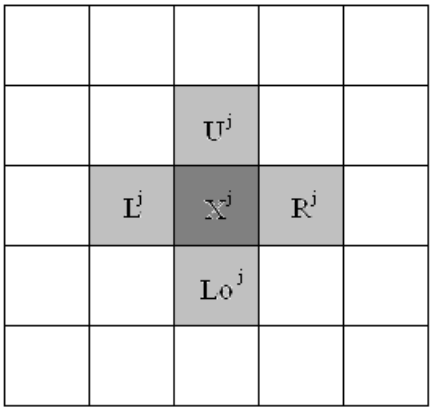


Figura 4.9: Novos vizinhos do bloco em processo de codificação.

Para que os novos vizinhos sejam utilizados, o presente esquema passará a classificar os blocos de imagem em dois grupos: blocos de estado inicial $\mathcal{X}^0$ e blocos com casamento lateral generalizado $\mathcal{X}^{0(GSM)}$. O primeiro grupo consiste nos primeiros blocos $\mathbf{X}^0$ a serem codificados, que são organizados num esquema similar a uma subamostragem de 2:1 na horizontal e na vertical, conforme ilustra a Figura 4.10.

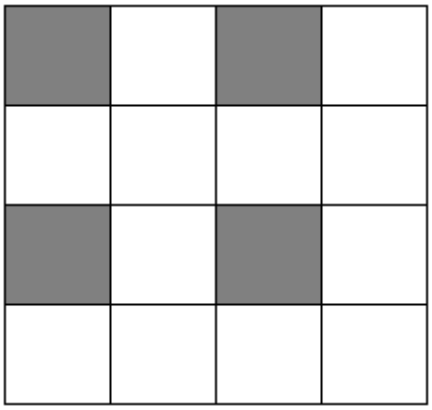


Figura 4.10: Organização dos blocos de estado inicial.

Estes blocos podem ser codificados com o MMP original ou o APM-CM-MMP. Entretanto, para melhorar o desempenho em taxas mais baixas e criar uma hierarquia de casamento lateral, todos os blocos de estado inicial são codificados utilizando-se o APM-CM-MMP (o tamanho de cada $\mathcal{D}_s^j$ é quadruplicado, devido ao aumento da incerteza), com exceção do bloco no canto superior esquerdo, que é o bloco fundamental e não possui qualquer vizinho, sendo necessária a utilização do MMP. Deste modo, o casamento lateral não é mais realizado com os vizinhos imediatamente acima e à esquerda, mas sim com $\mathcal{U}^0$ e $\mathcal{L}^0$, conforme ilustrado na Figura 4.11. Esta estratégia melhora o desempenho do esquema em taxas mais baixas ($<$ 0,3 bpp). Isto pode ser entendido levando-se em consideração o fato de que, se apenas o MMP fosse empregado na codificação destes blocos, dado que o mesmo não utiliza contextos adaptativos, necessitaria de taxas R_{i_j,s_j} para codificar os índices $\mathbf{b}_{i_j,s_j}$ e R_t para codificar os *flags* descritores da árvore de segmentação muito maiores. Como pode ser facilmente observado, os blocos de estado inicial correspondem a 25% do total de blocos da imagem.

Após a codificação dos blocos $\mathcal{X}^0$, o *grid* inicial está gerado, com blocos-chave periodicamente espalhados pela imagem. Esses blocos podem ser utilizados como novos vizinhos para o casamento lateral, consistindo em uma nova formação de contextos para o processo de codificação. O próximo grupo de blocos, conhecido como blocos com casamento lateral generalizado $\mathcal{X}^{0(GSM)}$, são codificados sob o conceito do casamento lateral generalizado, onde pode haver mais de dois vizinhos. Com a organização sugerida na Figura 4.10, podem ocorrer três situações (sem contar com

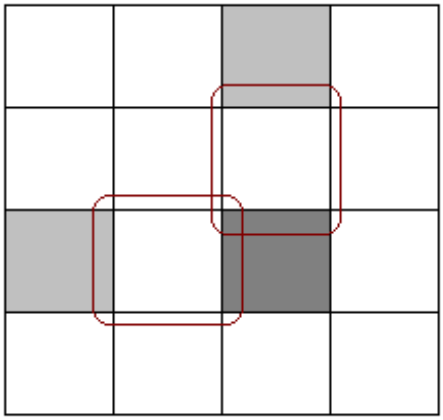


Figura 4.11: Casamento lateral realizado nos blocos de estado inicial.

as restrições dos blocos nas bordas da imagem, que são casos especiais), mostradas na Figura 4.12. Observa-se, então, que agora é possível haver casamento com até três vizinhos e de diferentes formas. A rugosidade e os contextos são calculados de modo similar ao mostrado na seção 4.1.1, equações (4.8) e (4.12), mas com a adição de mais um elemento. Por exemplo, para o casamento tipo 3 mostrado na Figura 4.12, a rugosidade e o modelo de contextos seriam

$$
\begin{aligned}
R\left(\mathbf{U}^j, \mathbf{L}^j, \mathbf{R}^j, \mathbf{b}_i\right) = \sum_{k=0}^{N-1} & \left| \left\lfloor \frac{U^j(M-2,k) - U^j(M-1,k) + b_i(0,k) - b_i(1,k)}{2} \right\rfloor + \right. \\
& \left. b_i(0,k) - U^j(M-1,k) \right| + \\
\sum_{p=0}^{M-1} & \left| \left\lfloor \frac{L^j(p,N-2) - L^j(p,N-1) + b_i(p,0) - b_i(p,1)}{2} \right\rfloor + \right. \\
& \left. b_i(p,0) - U^j(p,N-1) \right| \\
\sum_{p=0}^{M-1} & \left| \left\lfloor \frac{b_i(p,N-2) - b_i(p,N-1) + R^j(p,0) - R^j(p,1)}{2} \right\rfloor + \right. \\
& \left. R^j(p,0) - b_i(p,N-1) \right|
\end{aligned}
\tag{4.22}
$$

e

$$
\Pr\left\{\mathbf{b}_{i,s} | \hat{\mathbf{U}}^j, \hat{\mathbf{L}}^j, \hat{\mathbf{R}}^j\right\} = \begin{cases} C^{-1} \Pr\left\{\mathbf{b}_{i,s}\right\} & , \mathbf{b}_{i,s} \in \mathcal{D}_s^j \\ 0 & , \mathbf{b}_{i,s} \notin \mathcal{D}_s^j, \end{cases} \tag{4.23}
$$

respectivamente. O número de elementos N_s^j do dicionário reduzido $\mathcal{D}_s^j$ é dado por

$$
N_s^j = \max\left\{ N_{\max}\left(\frac{A(\hat{\mathbf{U}}^j) + A(\hat{\mathbf{R}}^j) + A(\hat{\mathbf{L}}^j)}{2} \right) A_{\max}^{-1}, 16 \right\}, \tag{4.24}
$$

o que é apenas uma extensão da equação (4.13). $N_{\max}$ é calculado de acordo com (4.14). Extensões similares podem ser feitas para o casamento tipo 1. O dicionário $\mathcal{D}$, no GSM-MMP, é montado e atualizado da mesma maneira apresentada para o APM-MMP (ver a seção 4.1.1).

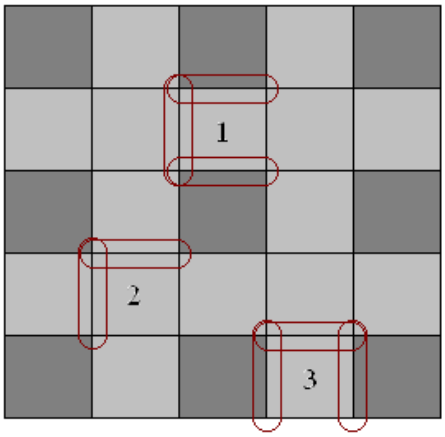


Figura 4.12: Tipos de casamento encontrados em blocos com casamento lateral generalizado.

Com isso, os blocos utilizados na criação de $\mathcal{D}_s^j$ são escolhidos com uma maior restrição e são mais casados com os seus vizinhos causais. Tanto o MMP original quanto o APM-MMP podem ser considerados casos especiais deste esquema.

O dicionário de vizinhança

Apesar do GSM-MMP já utilizar um esquema de atualização que leva em consideração elementos da vizinhança causal do bloco (ver as melhorias do algoritmo base apresentadas na seção 4.1.1), nem todos os deslocamentos possíveis são considerados. Além disso, o contexto calculado pode não incluir em $\mathcal{D}_s^j$ algum elemento interessante para a codificação de $\mathbf{X}^j$, dada a incerteza com respeito aos *pixels* que não fazem parte da borda dos blocos [93]. Para propiciar um dicionário mais rico e capturar estruturas periódicas no sinal, resultando numa maior qualidade da imagem reconstruída, utiliza-se também o conceito de dicionário de vizinhança $\mathcal{D}^N$. Tal dicionário consiste em todos os deslocamentos possíveis, pixel a pixel, na vizinhança causal de $\mathbf{X}^j$, como ilustrado na Figura 4.13. Para cada $\mathbf{X}^j$, uma janela com as mesmas dimensões é deslocada na sua vizinhança causal, onde cada deslocamento corresponde a um índice de dicionário $\mathbf{b}_i^N$, e procura-se o elemento que

melhor aproxima $\mathbf{X}^j$. Como resultado, se certos blocos se repetem na imagem de forma determinística, alguns elementos $\mathbf{b}_i^N$ serão mais utilizados que outros, o que proporciona ganhos significativos quando do uso de um codificador aritmético [74]. É interessante observar que não há uma formação de contextos para $\mathcal{D}^N$, evitando-se qualquer restrição quanto aos elementos disponíveis.

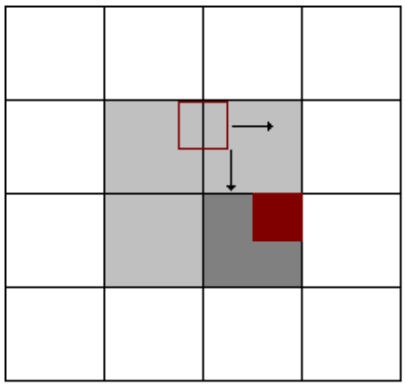


Figura 4.13: Exemplificação do dicionário de vizinhança.

Resultados de simulações

Para se avaliar a eficácia do método, o GSM-MMP foi utilizado para comprimir imagens em níveis de cinza, inicialmente divididas em blocos de dimensões 16×16, juntamente com o APM-CM-MMP e o JPEG2000 [70]. Os blocos de entrada foram processados em seqüência pelos algoritmos, no sentido de leitura, ou seja, da esquerda para a direita e de cima para baixo, como descrito na seção 4.1.1.

O dicionário inicial e a transformação de escala também são como como descrito na seção 4.1.1 . Com blocos iniciais de dimensões 16×16, 9 escalas diferentes podem ser obtidas

As Figuras 4.14 a 4.16 mostram o desempenho taxa-distorção dos algoritmos para as imagens *Cameraman*, *Einstein*, *House*, *Lena*, *F-16* e *PP1209*, sendo as três primeiras de dimensões 256×256 e as outras de dimensões 512×512 pixels (ver o apêndice B). As Figuras mostram que:

i) O APM-CM-MMP apresentou desempenhos melhores que os do JPEG2000 para as imagens *Cameraman* e *PP1209*.

ii) O APM-CM-MMP apresentou desempenhos comparáveis aos do JPEG2000 para as imagens *Eintein* e *F-16*.

iii) O APM-CM-MMP apresentou desempenho superior ao do JPEG2000, em taxas mais baixas, para a imagem *House*.

iv) O GSM-MMP apresentou desempenhos superiores aos do APM-CM-MMP para todas as imagens.

v) O GSM-MMP apresentou melhores desempenhos que o JPEG2000 para quatro das imagens, com vantagens da ordem de $\approx$ 2,2 dB a 0,5 bpp para a *PP1209* (misto de imagens em níveis de cinza, texto e gráficos), $\approx$ 1,8 dB a 0,5 bpp para a *Cameraman*, $\approx$ 0,35 dB a 0,4 bpp para a *Einstein* e $\approx$ 0,9 dB a 0,4 bpp para a *House*.

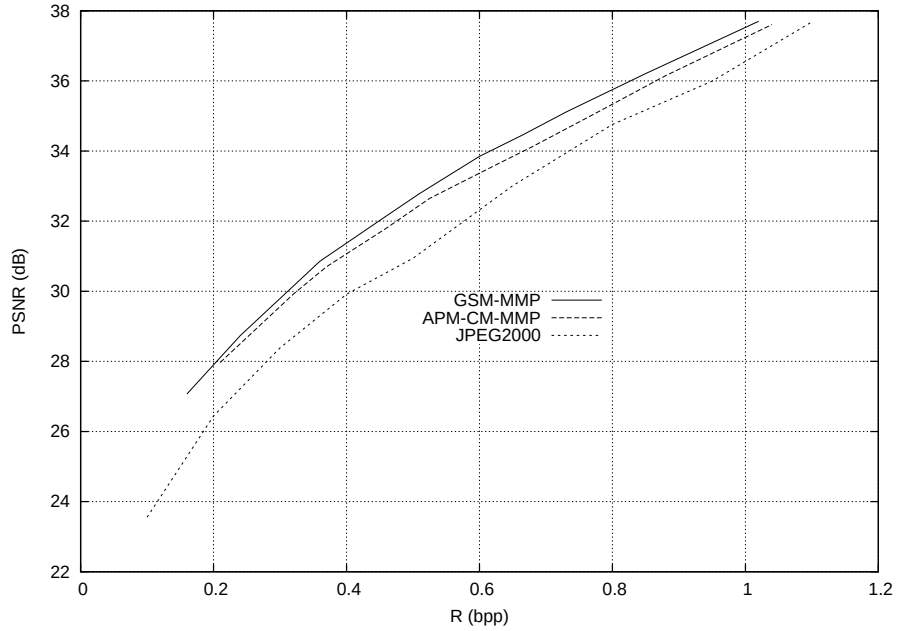
vi) O GSM-MMP apresentou desempenho comparável ao do JPEG2000 para a imagem *F-16*, ganhando em taxas mais baixas.

vii) O GSM-MMP não apresentou o mesmo desempenho que o JPEG2000 para a imagem *Lena*.

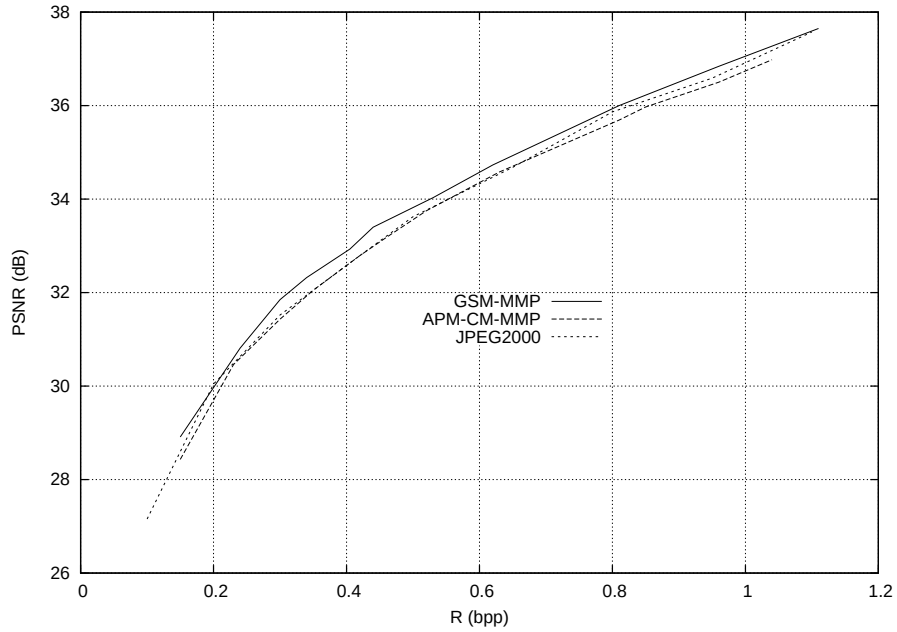
A Figura 4.17 mostra a imagem *Cameraman* original e as versões reconstruídas pelo GSM-MMP e pelo JPEG2000, a 0,3 bpp. É fácil perceber que os detalhes da imagem são mais preservados pelo GSM-MMP, inclusive os edifícios no lado direito, que não podem ser completamente identificados na versão do JPEG2000. O rosto do *cameraman* também está mais nítido.

4.2 O MMP aplicado à compressão de sinais de eletrocardiograma

Os sinais de Eletrocardiograma (ECG) [100, 101] são ferramentas importantes para o diagnóstico de doenças cardiovasculares, sendo muito comuns em sistemas de monitoramento (*e.g.* ECG *Holter*). Um exame completo, que pode ter várias derivações (*e.g.* 12 canais), constitui uma grande quantidade de informações, o que dificulta o seu armazenamento ou a sua transmissão. Com um esquema de

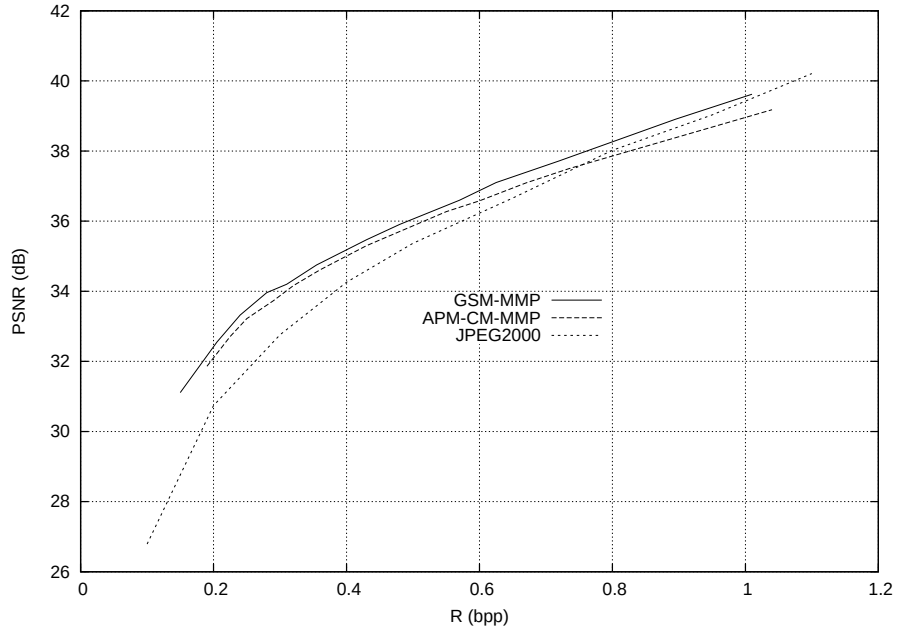


(a)

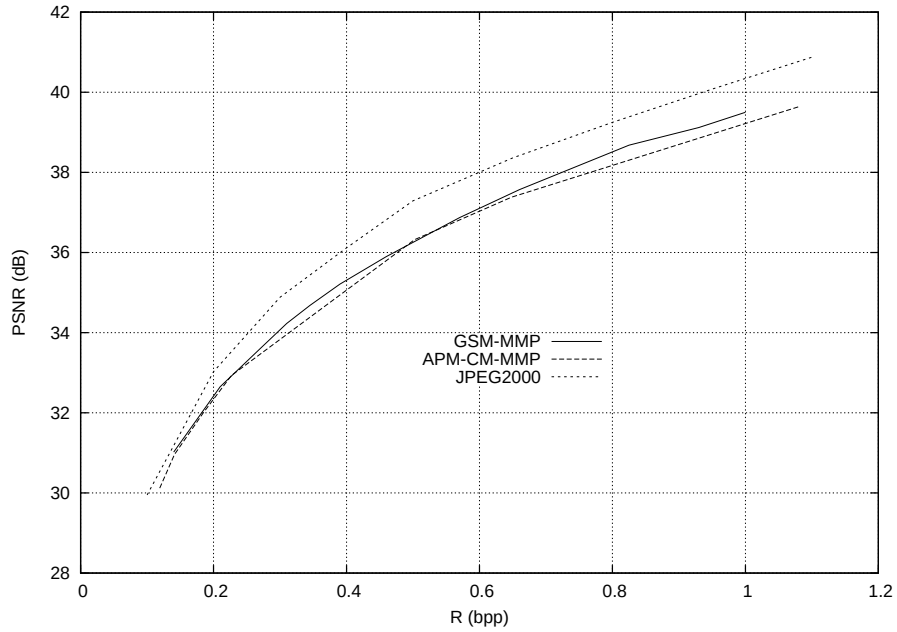


(b)

Figura 4.14: Desempenho taxa-distorção do GSM-MMP, APM-CM-MMP e JPEG2000 para: (a) *Cameraman* 256×256; (b) *Einstein* 256×256.

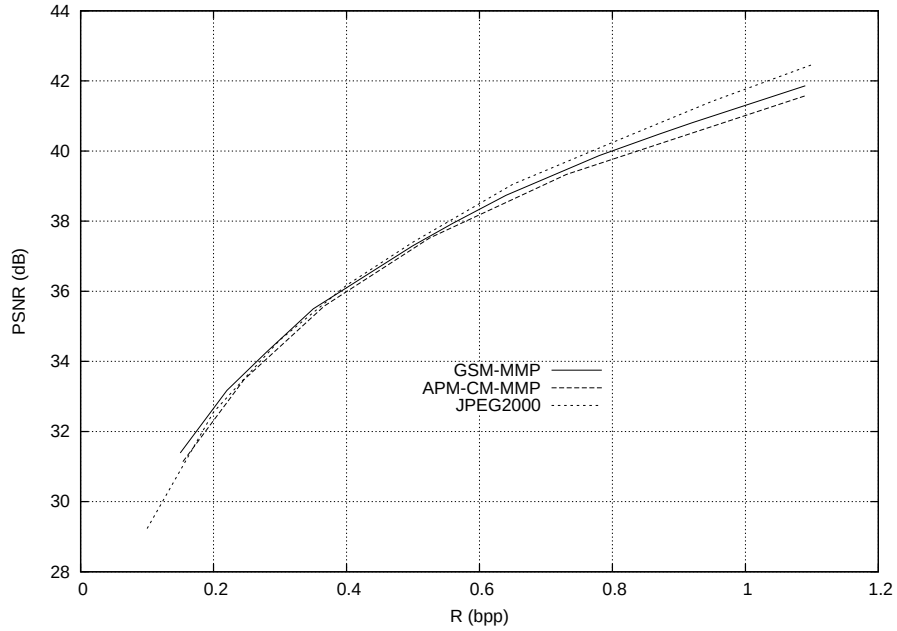


(a)

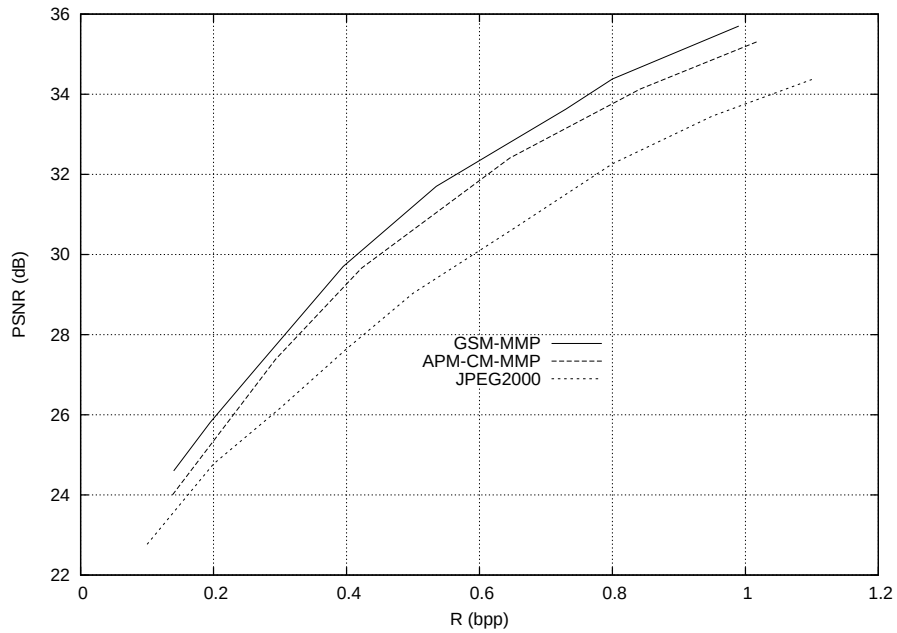


(b)

Figura 4.15: Desempenho taxa-distorção do GSM-MMP, APM-CM-MMP e JPEG2000 para: (a) *House* 256×256; (b) *Lena* 512×512.



(a)



(b)

Figura 4.16: Desempenho taxa-distorção do GSM-MMP, APM-CM-MMP e JPEG2000 para: (a) *F-16* 512×512; (b) *PP1209* 512×512.

Figura 4.17: *Cameraman* codificada. Imagem original (esquerda), GSM-MMP a 0,30 bpp com PSNR = 29,95 dB (centro) e JPEG2000 a 0,30 bpp com PSNR = 28,30 dB (direita).

compressão eficiente, aplicações como transmissão por linha telefônica (o que viabiliza o diagnóstico remoto) e bancos de dados de ECG seriam bastante facilitadas, permitindo um acompanhamento detalhado de cada paciente e diagnósticos mais rápidos. Um sinal de ECG típico está mostrado na Figura 4.18.

As próximas seções apresentarão novos desenvolvimentos em compressão de ECG, principalmente no que diz respeito às técnicas de pré-processamento, que aumentam o desempenho dos algoritmos de codificação. Uma versão unidimensional do MMP será empregada, bem como codificadores de imagem como o JPEG2000 [70] e o H.264 intra-quadros [69].

Figura 4.18: Sinal de ECG típico extraído da base de dados MIT/BIH.

4.2.1 A estrutura do sinal de ECG: uma breve análise

O diagnóstico proporcionado pelo médico é diretamente baseado na observação da estrutura do sinal de ECG; algumas doenças podem ser identificadas com muita precisão e outras apenas estimadas. Sendo assim, prática e experiência são fatores-chave ao se realizar o diagnóstico. De modo a entender esse processo, esta seção proporcionará uma breve discussão sobre a morfologia do sinal de ECG e sua análise. Uma descrição mais detalhada pode ser encontrada em [100, 101].

Como já foi dito, os sinais de ECG constituem uma das principais ferramentas para o diagnóstico de doenças cardiovasculares, que podem ser diretamente inferidas através da análise da sua estrutura. Portanto, o conhecimento detalhado da formação do sinal de ECG é preponderante para um diagnóstico confiável, e todo e qualquer processamento aplicado ao mesmo deve manter a sua estrutura original.

O coração é uma bomba muscular que envia sangue através do sistema circulatório e é composta por dois (direito e esquerdo) átrios e dois (direito e esquerdo) ventrículos. O ventrículo direito e o átrio esquerdo são conectados aos pulmões, onde o dióxido de carbono é removido do sangue e oxigênio é adicionado; o ventrículo esquerdo e o átrio direito proporcionam a conexão para o resto do corpo. Essas quatro câmaras são contraídas e relaxadas de forma coordenada, através de estímulos elétricos enviados por um grupo de células chamado *nó de sinus*.

O sinal de ECG consiste na medição desses estímulos elétricos que comandam a ativação/desativação das câmaras, dados por campos elétricos na superfície do corpo humano. A Figura 4.19 mostra uma porção completa de um sinal de ECG, correspondendo a um ciclo do batimento cardíaco, que é composta pelas ondas **P**, **Q**, **R**, **S** e **T**. A onda **P** representa a despolarização do átrio, geralmente com amplitude de 0,1mV, e o período que vai do começo da onda **P** até o começo do complexo **QRS** é chamado de intervalo **P-R**. Este último representa o espaço de tempo entre o início da despolarização atrial e o início da despolarização ventricular. O complexo **QRS** representa a despolarização ventricular e possui amplitude média de 1,0 mV. O período isoelétrico (segmento **ST**) logo após o complexo **QRS** é o intervalo de tempo durante o qual o ventrículo está despolarizado e corresponde à paralisação da atividade ventricular. A onda **T** representa a repolarização ventricular e é mais longa que a despolarização; a sua amplitude apresenta um valor médio de 0,2 mV. O intervalo **Q-T** representa o tempo necessário para a despolarização/repolarização ventricular, sendo portanto uma estimativa da duração média da atividade ventricular.

A observação do sinal de ECG pode revelar muitas doenças cardíacas, como desordens na seqüência de ativação, arritmias, aumento nos tamanhos dos átrios e ventrículos, isquemias e cardites, além de proporcionar monitoramento de aparelhos de marca-passo e ação de drogas.

A arritmia ventricular pode ser identificada pela observação de complexos QRS de longa duração, normalmente superior a 0,1 s. Isto significa que a ativação dos ventrículos não está ocorrendo normalmente. Outra forma de arritmia é a fibrilação ventricular, na qual os ventrículos são despolarizados de forma caótica, gerando ondulações irregulares sem complexos QRS. Bloqueios atrioventriculares, que são

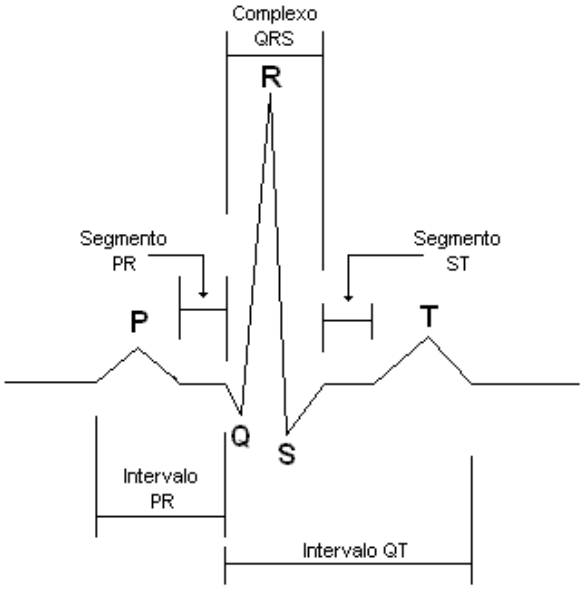


Figura 4.19: ECG de um ciclo do batimento cardíaco.

distúrbios na condução do estímulo elétrico, refletem no intervalo P-R, fazendo-o mais longo que o normal ($> 0{,}2$ s). Isquemias, que são oclusões na artéria coronária que diminuem o transporte de oxigênio para o músculo cardíaco, são diagnosticadas através de mudanças na onda T. A hipertrofia do átrio direito ocasiona uma onda P de amplitude muito alta ($> 0{,}25$ mV).

Com base nestas observações, pode-se concluir que um esquema de compressão de ECG eficiente deve imperativamente manter a largura do complexo QRS, a distância entre todas as ondas componentes e o formato das ondas P e T. Se isto não for observado, o médico analisará estruturas errôneas ou incompletas, comprometendo o diagnóstico.

4.2.2 A compressão de sinais de ECG

Nos últimos anos, os meios de armazenamento (*e.g.* memórias e discos rígidos) ficaram muito mais baratos e a sua capacidade aumentou consideravelmente. Apesar disso, a compressão de ECG ainda é uma aplicação importante, principalmente devido à necessidade de transmissão através de linhas telefônicas ou redes de comunicação com bandas restritas. Dado estes fatos, um esforço considerável tem sido feito para se alcançarem altas taxas de compressão com baixa distorção, não comprometendo a efetividade do diagnóstico.

Os algoritmos de compressão existentes na literatura podem ser classificados em três grupos distintos: o direto, o paramétrico e o baseado em transformadas. Na classe direta, os algoritmos empregam técnicas de predição para estimar uma amostra ou grupo de amostras com base nos dados previamente codificados; o resíduo é então gerado, subtraindo-se as amostras dos valores preditos, sendo o mesmo posteriormente quantizado e codificado [102]. Na classe paramétrica, os algoritmos tentam extrair características do sinal de entrada, que são posteriormente utilizadas para se comandar um sintetizador, que por sua vez reconstrói o sinal original [103]. Na classe baseada em transformadas, o sinal de entrada é primeiramente transformado para outro domínio, normalmente através de uma transformação linear. Os coeficientes da transformada são então comprimidos, utilizando-se uma combinação de quantização e codificação de entropia. Uma das vantagens dessa abordagem consiste no fato de que é normalmente mais fácil comprimir o sinal no domínio da transformada, desde que uma operação adequada seja escolhida. Alguns dos melhores codificadores pertencem a essa classe [4, 104–106]. Além disso, muitos dos esquemas de codificação que representam o estado da arte, desenvolvidos até agora, são baseados no paradigma de compressão em três passos: uma primeira etapa de transformação seguida por um esquema de quantização e também por um codificador de entropia.

A comparação entre algoritmos de compressão de ECG é feita através dos sinais existentes na dase de dados MIT/BIH (*Massachusetts Institute of Technology/Beth Israel Hospital*). Esse acervo contém partes de exames de ECG de 48 indivíduos diferentes, cada um com duas derivações; cada canal tem a duração aproximada de 30 minutos. Os sinais foram amostrados a 360 Hz e quantizados com 11 bits. Um desses sinais (os primeiros três segundos) já foi mostrado na Figura 4.18.

Os resultados da codificação de sinais de ECG são normalmente avaliados utilizando-se a métrica de *Percent Root mean square Difference* (PRD), que mede distorção, o *Compressed Data Rate* (CDR), que é a taxa de bits da representação comprimida, e o *Compression Ratio* (CR), que representa a taxa de compressão,

definidos como

$$PRD = 100\sqrt{\frac{\sum_{n=0}^{N-1}\left(x(n)-\hat{x}(n)\right)^2}{\sum_{n=0}^{N-1}\left(x(n)-\mu\right)^2}} \tag{4.25}$$

$$CDR = \frac{B_c}{T} \tag{4.26}$$

$$CR = \frac{B_o}{B_c}, \tag{4.27}$$

onde μ é a linha base do conversor analógico-digital utilizado para a aquisição dos dados $x(n)$ (na MIT/BIH $\mu = 1024$), B_c é o número total de bits do dado comprimido, incluindo a informação auxiliar, T é a duração do sinal original, em segundos, e B_o é o número total de bits do sinal original.

4.2.3 O MMP adaptado para a compressão de sinais de ECG

O algoritmo MMP base para sinais de ECG é o mesmo apresentado na seção 2.1, com a diferença de que a segmentação é aplicada a um sinal unidimensional, conforme ilustra a Figura 4.20. Além disso, três extensões são utilizadas: uma nova estratégia de otimização da árvore de segmentação, a criação de contextos com base na vizinhança causal e a partição de dicionários, de forma semelhante ao que é feito na seção 4.1.1. Na verdade, o algoritmo descrito nesta seção pode ser visto como um APM-CM-MMP com uma estratégia de otimização avançada.

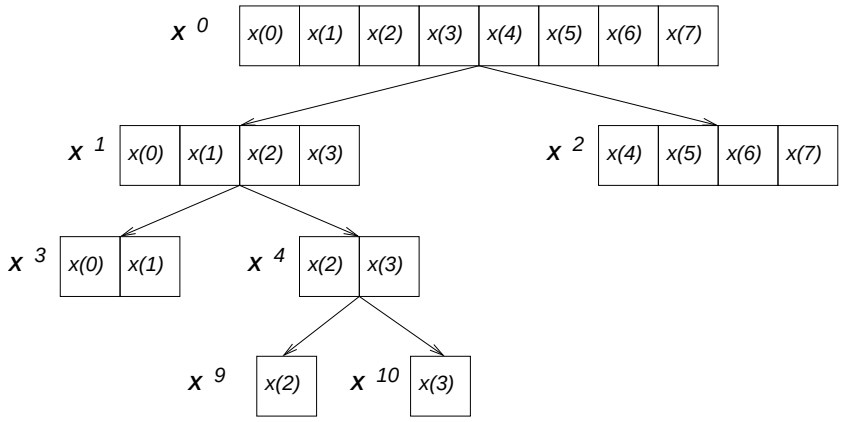


Figura 4.20: Segmentação de um sinal unidimensional no MMP.

O algoritmo MMP unidimensional

No MMP unidimensional, o processamento do sinal é feito de forma similar ao que foi apresentado em 2.1 para duas dimensões. Entretanto, o sinal de entrada é

dividido em segmentos lineares com comprimentos iguais, cujo valor é uma potência de 2 (*e.g.* 64 amostras), que são posteriormente recursivamente particionados ao meio, na tentativa de se encontrar um elemento adequado no dicionário.

Novamente, elementos de um dicionário $\mathcal{D} = \{\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{L-1}\}$ são utilizados para aproximar segmentos de um dado vetor de entrada

$$\mathbf{X}^0 = \begin{pmatrix} x(0,0) & x(0) & \ldots & x(N_0-1) \end{pmatrix}, \tag{4.28}$$

onde N_0 é uma potência de dois. O MMP procura então, no dicionário $\mathcal{D}$, o melhor vetor $\mathbf{v}_{i_0}$ para substituir $\mathbf{X}^0$. Se um elemento $\mathbf{v}_{i_0}$ adequado é encontrado, o algoritmo gera um *flag* '1', indicando um casamento bem sucedido, e o índice i_0 referente ao elemento escolhido; caso contrário, o vetor de entrada é particionado em dois outros segmentos iguais

$$\mathbf{X}^1 = \begin{pmatrix} x(0) & \cdots & x(0, N_0/2-1) \end{pmatrix}$$

e

$$\mathbf{X}^2 = \begin{pmatrix} x(N_0/2) & \cdots & x(N_0-1) \end{pmatrix}.$$

Quando isto ocorre, o algoritmo gera um *flag* '0', indicando que houve uma divisão do vetor, e repete o procedimento para $\mathbf{X}^1$. O restante do processo de codificação segue a mesma lógica já apresentada na seção 2.1 para o caso bidimensional.

Com relação à árvore de segmentação binária $\mathcal{S} = \{n_j\}$ resultante, cada nó n_j corresponde a um segmento $\mathbf{X}^j$ em $\mathbf{X}^0$, de acordo com:

i) n_0 corresponde a $\mathbf{X}^0$.

ii) Para cada nó n_j associado a um segmento $\mathbf{X}^j$ de tamanho N_j, tem-se que n_{2j+1} e n_{2j+2} estão associados a dois segmentos de comprimento $N_j/2$, tal que a sua concatenação seja igual a $\mathbf{X}^j$.

iii) Esta segmentação é recursivamente aplicada a todos os segmentos, começando no nó n_0, até que a escala de dimensões 1×1 seja alcançada.

Um exemplo de segmentação está ilustrado na Figura 4.20. As transfomações de escala são as mesmas apresentadas na seção 2.1, com a ressalva de que são aplicadas uma única vez (sinal unidimensional), ou seja,

$$\mathbf{Y}^N = T_N\left[\mathbf{X}^L\right]. \tag{4.29}$$

A estratégia multidicionários é utilizada, aumentando-se a velocidade de codificação, e o dicionário é atualizado da mesma maneira apresentada na seção 4.20, utilizando-se concatenações de segmentos previamente codificados.

Conforme apresentado na seção 2.1, a árvore de segmentação pode ser otimizada, podando-se os nós-filhos sempre que o seu custo lagrangeano for maior que o do nó-pai. Inicialmente, isso poderia sugerir que não há mais possibilidade de melhoria com relação à segmentação, porém, ocorre exatamente o contrário. Todas as possibilidades de segmentação geradas para uma árvore binária, para um vetor de entrada de comprimento 8, estão ilustradas na Figura 4.21(a), sendo que as Figuras 4.21(b-c) mostram outras possibilidades de segmentação. Uma breve análise destas figuras indica que a estrutura da árvore binária restringe as opções de segmentação, pois não há como avaliar custos de nós que não tenham o mesmo nó-pai.

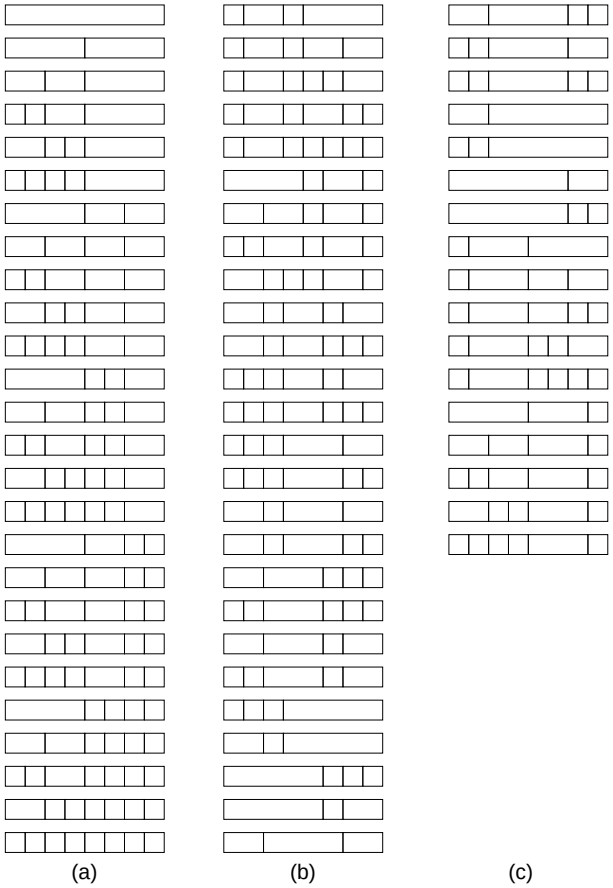


Figura 4.21: Segmentação no MMP: (a) Permitido, (b-c) Não permitido.

Em [107], é apresentado um algoritmo que extende as opções de segmentação de uma árvore binária, chamado de algoritmo *prune-join* (poda-junta). Nesse al-

goritmo, um vetor de entrada é particionado, obedecendo-se a um critério taxa-distorção [71], e cada segmento é aproximado por uma função polinomial. A otimização é realizada iterativamente, podando-se a árvore binária como descrito em [71]. A primeira etapa consiste na já conhecida *poda* dos nós. A segunda, a *união*, é realizada testando-se todos os nós vizinhos que não são descendentes de um mesmo nó-pai, para verificar se a sua codificação como um único elemento reduz o custo lagrangeano global. O passo de união pode ser repetido recursivamente, possibilitando uniões de uniões de nós. Então, os coeficientes dos polinômios associados aos nós restantes são quantizados e codificados. Em [107], provou-se que o algoritmo assim descrito supera a otimização baseada somente na poda dos nós, proporcionado um desempenho taxa-distorção aproximadamente ótimo (pelo menos para sinais que são aproximadamente polinomiais por partes). Esses resultados teóricos não podem ser diretamente aplicados ao MMP, pois a utilização do dicionário e a abordagem baseada em recorrência de padrões multiescalas são muito diferentes da codificação escalar de polinômios utilizada no algoritmo *prune-join* original. Entretanto, espera-se um aumento no desempenho ao se estender as possibilidades de segmentação. Todas as opções de segmentação mostradas na Figura 4.21(b-c) podem ser obtidas de uma segmentação binária, seguida de uma união entre nós com nós-pais diferentes (nós-primos).

Para se incorporar o conceito *prune-join* ao MMP, aplica-se a segmentação original a um vetor de entrada $\mathbf{X}^0$, como o mostrado na Figura 4.20, obtendo-se uma árvore de segmentação $\mathcal{S}$ otimizada no sentido taxa-distorção. A otimização descrita na seção 2.1 realiza a parte de poda do algoritmo. Depois disso, emprega-se uma análise para verificar se dois nós-primos podem ser unidos, resultando numa redução do custo lagrangeano. A Figura 4.22(a) ilustra um exemplo de árvore de segmentação após a poda. A Figura 4.22(b) ilustra uma possível operação de união.

Define-se $U(n_j, n_l)$ como a união de dois nós-primos n_j e n_l. Em cada par destes, verifica-se se o custo de sua codificação de forma independente (isto é, $J(n_j) + J(n_l) + \lambda R_{nu}(i,j)$) é maior que o custo da codificação da sua união ($R_{nu}(i,j)$ é a taxa associada à codificação dos *flags* indicando que os nós não devem ser unidos). O custo lagrangeano da união é dado por

$$J\left(U\left(n_j, n_l\right)\right) = d\left(\left(\begin{array}{c} \mathbf{X}^j \mathbf{X}^l \end{array}\right), \mathbf{v}_{i_{jl}, s_{jl}}\right) + \lambda\left(R_u(j,l) - \log_2(\Pr(i_{jl}))\right), \qquad (4.30)$$

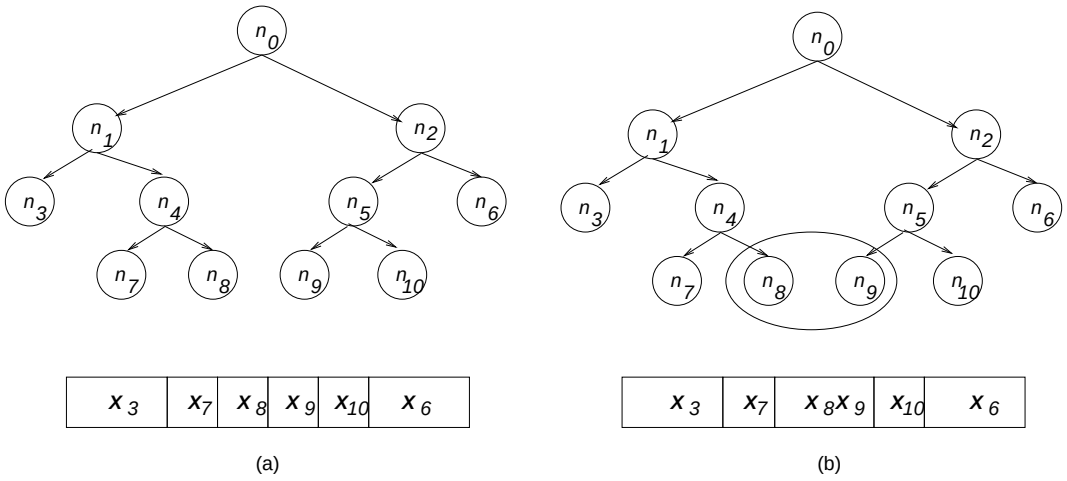


Figura 4.22: O processo de *união* dos nós.

onde $R_u(j,l)$ é a taxa gasta para se codificar o *flag* de *união*, $\mathbf{v}_{i_{j_l},s_{j_l}}$ é a melhor aproximação do dicionário para a união de $\mathbf{X}^j$ e $\mathbf{X}^l$ e i_{j_l} representa o índice do vetor escolhido.

Quando a codificação conjunta de dois nós n_j e n_l ocorre, o dicionário não é mais atualizado com deslocamentos do resultado da concatenação das versões reconstruídas de vetores associados aos mesmos, como descrito na seção 4.1.1. Logo, toda vez que uma união ocorre, as atualizações de dicionário devidas aos blocos isolados são removidas e os registros de estatísticas de uso de elementos e *flags* são atualizados. Além disso, o custo da união deve ser computado juntamente com o primeiro bloco da união analisado, prevenindo o uso de um elemento que seria podado posteriormente. É interessante ressaltar que se forem utilizadas múltiplas cópias do dicionário, não haverá mais apenas $(1+\log_2(N))$ escalas diferentes, pois o procedimento de união adicionará muitas escalas extras.

O modelo de probabilidades

Como já foi mencionado na seção 4.1.1, a representação $\hat{\mathbf{X}}^0$ que o MMP produz pode apresentar descontinuidades nas bordas dos blocos $\hat{\mathbf{X}}^j$ gerados pelo procedimento de segmentação, mesmo que o vetor original $\mathbf{X}^0$ seja suave, fato este que está ilustrado na Figura 4.23 para um sinal unidimensional.

Do modo similar ao que foi proposto na seção 4.1.1, pode-se controlar a suavidade das representações geradas pelo MMP unidimensional através da definição de um modelo de probabilidades. A idéia é escolher um subconjunto do dicionário

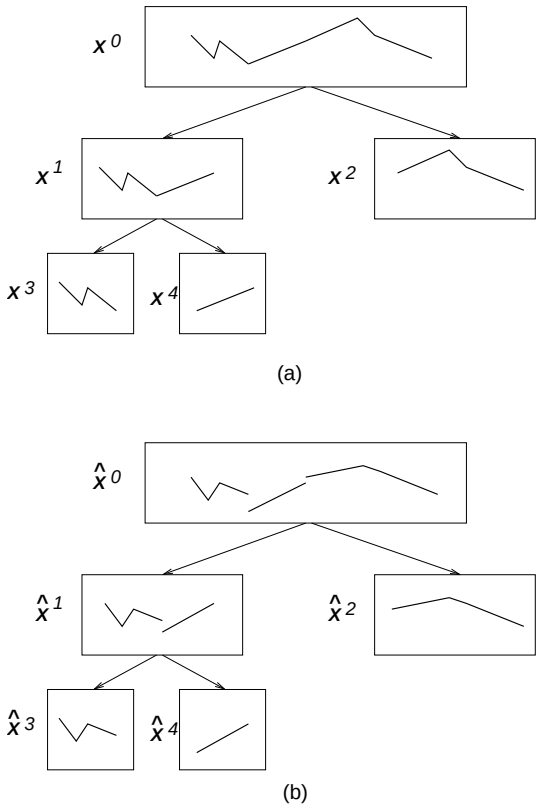


Figura 4.23: Descontinuidades no MMP unidimensional: (a) Sinal original; (b) Sinal reconstruído.

$\mathcal{D}$, ou seja, $\mathcal{D}_S^j$, composto pelos N_S^j melhores vetores $\mathbf{v}_{i,s} \in \mathcal{D}$ dados pelo contexto calculado. Para se compreender melhor a formação dos contextos no MMP unidimensional, são necessárias algumas definições.

A posição da primeira amostra de $\mathbf{X}^j$ dentro de $\mathbf{X}^0$ é dada por:

$$Fp(j) = Q\left((j+1)\,2^{-\lfloor \log_2(j+1) \rfloor} - 1\right) \tag{4.31}$$

onde Q é o comprimento de $\mathbf{X}^0$. Por exemplo, com relação à Figura 4.21 tem-se: $Fp(0) = 0$ (a primeira amostra de $\mathbf{X}^0$ é x(0)), $Fp(1) = 0$ (a primeira amostra de $\mathbf{X}^1$ é x(0)), $Fp(2) = 4$ (a primeira amostra de $\mathbf{X}^2$ é x(4)) e assim por diante.

O comprimento Q^j de $\mathbf{X}^j$, pode ser calculado como

$$Q^j = Q2^{-\lfloor \log_2(j+1) \rfloor} \tag{4.32}$$

e o *vizinho esquerdo* de $\mathbf{X}^j$ é dado por

$$\mathbf{L}^j = \left(\begin{array}{ccc} \hat{x}(Fp(j) - Q^j) & \ldots & \hat{x}(Fp(j) - 1) \end{array} \right),\ Fp(j) \geq 1 \tag{4.33}$$

De acordo com a equação (4.33), o vizinho esquerdo $\mathbf{L}^j$ é um vetor de mesmo comprimento que $\mathbf{X}^j$, cujos componentes são as amostras reconstruídas à esquerda de $\mathbf{X}^j$.

Para se medir a suavidade da aproximação, definem-se três parâmetros:

i) A *descontinuidade de ordem zero*:

$$D^0\left(\mathbf{v}_{i,s}, j\right) = L^j(Q^j - 1) - v_{i,s}(0) \tag{4.34}$$

ii) A *descontinuidade de primeira ordem*:

$$D^1\left(\mathbf{v}_{i,s}, j\right) = L^j(Q^j - 2) - L^j(Q^j - 1) + v_{i,s}(0) - v_{i,s}(1) \tag{4.35}$$

iii) A *descontinuidade de segunda ordem*:

$$D^2\left(\mathbf{v}_{i,s}, j\right) = L^j(Q^j - 3) - 2L^j(Q^j - 2) + L^j(Q^j - 1) - v_{i,s}(0) + 2v_{i,s}(1) - v_{i,s}(2) \tag{4.36}$$

Estas definições são baseadas nas aproximações para as suas respectivas derivadas. Define-se, então, a métrica de *rugosidade* como:

$$R(\mathbf{v}_{i,s}, j) = \left|\alpha D^0\left(\mathbf{v}_{i,s}, j\right) + \beta D^1\left(\mathbf{v}_{i,s}, j\right)\right| + \left|\gamma D^1\left(\mathbf{v}_{i,s}, j\right) + \theta D^2\left(\mathbf{v}_{i,s}, j\right)\right|$$

onde $\mathbf{v}_{i,s}$ é um vetor do dicionário na escala s. Assim como feito na seção 4.1.1, estas funções tornam possível a identificação de transições através dos blocos, mantendo assim a continuidade do sinal e uma escolha adequada dos contextos. Consegue-se então manter a continuidade de valores constantes, retas e parabolóides através dos segmentos de sinal. Por exemplo, na Figura 4.24, a descontinuidade de primeira ordem tenderá a favorecer a utilização do bloco com um círculo vermelho, pois a manutenção da continuidade exigiria uma reta também crescente no outro bloco. A métrica de *rugosidade* específica utilizada foi:

$$\begin{aligned} R(\mathbf{v}_{i,s}, j) = \Bigg| & \left|L^j(Q^j - 3) - L^j(Q^j - 1) + v_{i,s}(0) - v_{i,s}(2)\right| \\ & - \left\lfloor \frac{4}{3}\left|L^j(Q^j - 2) - v_{i,s}(1)\right| \right\rfloor \Bigg|, \end{aligned} \tag{4.37}$$

onde $\alpha = \beta = \frac{4}{3}$, $\gamma = 2$ e $\theta = 1$.

Sendo assim, com relação à Figura 4.21, quando o MMP tenta codificar $\mathbf{X}^2$, já estão disponíveis $\hat{\mathbf{X}}^3$, $\hat{\mathbf{X}}^9$ e $\hat{\mathbf{X}}^{10}$. Portanto, conhece-se $\hat{\mathbf{X}}^0$ até a quarta amostra

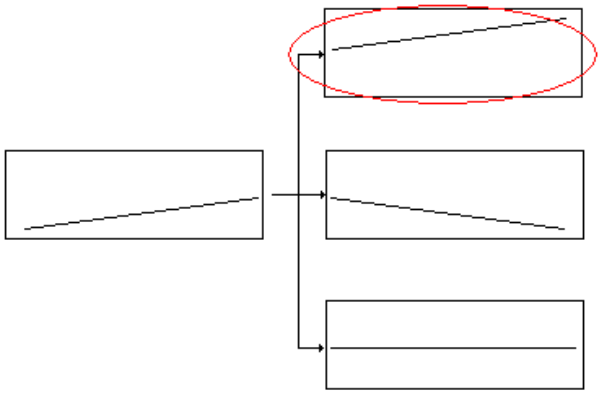


Figura 4.24: Exemplo de escolha de bloco para continuidade.

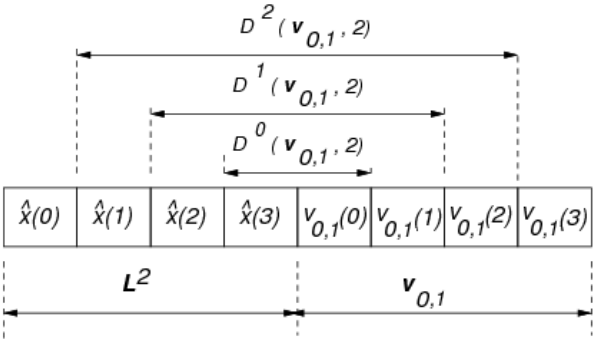


Figura 4.25: Cálculo da rugosidade

($Fp(2) = 4$), ou seja, $\hat{\mathbf{X}}^0 = \begin{pmatrix} \hat{x}(0) & \hat{x}(1) & \hat{x}(2) & \hat{x}(3) & ? & ? & ? & ? \end{pmatrix}$. O vizinho esquerdo de $\mathbf{X}^2$ é então determinado como $\mathbf{L}^2 = \begin{pmatrix} \hat{x}(0) & \hat{x}(1) & \hat{x}(2) & \hat{x}(3) \end{pmatrix}$. Se não houver um vizinho esquerdo, o que acontece com o primeiro bloco, $\mathcal{D}_S^j = \mathcal{D}^j$.

Assim, o MMP constrói um dicionário $\mathcal{D}_S^j$ contendo os N_S^j vetores que mais se adéquam ao contexto atual, de acordo com o modelo

$$\Pr\left\{\mathbf{v}_{i,s}|\hat{\mathbf{L}}^j\right\} = \begin{cases} C^{-1}\Pr\left\{\mathbf{v}_{i,s}\right\} & , \mathbf{v}_{i,s} \in \mathcal{D}_s^j \\ 0 & , \mathbf{v}_{i,s} \notin \mathcal{D}_s^j \end{cases} \tag{4.38}$$

A Figura 4.25 ilustra como calcular $R(\mathbf{v}_{0,1}, 2)$, onde três pixels de borda de cada bloco são considerados no cálculo das continuidades.

Baseando-se na asserção de que sinais mais simples exigem um número menor de vetores para serem bem representados (ver discussão na seção 4.1.1), utiliza-se a métrica de *atividade*, como definida na equação (4.39) para um vetor unidimensional, para se estimar o número de vetores N_S^j necessários em $\mathcal{D}_S^j$ para se codificar o segmento de entrada. A atividade tem a capacidade de medir a complexidade do

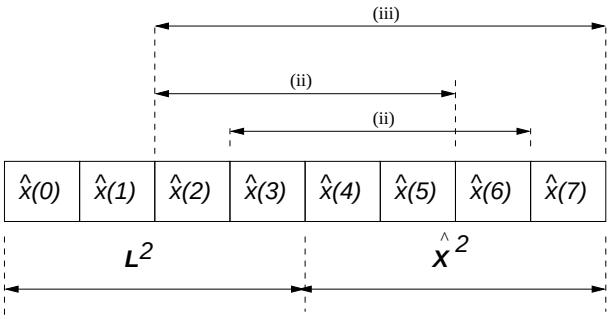


Figura 4.26: Novas atualizações do MMP unidimensional ($j = 2$).

sinal, proporcionando uma estimativa de dificuldade na codificação de um dado elemento (ver discussão na seção 4.1.1).

$$A\left(\mathbf{L}^j\right) = \sum_{n=1}^{N^j-1} \left|L^J(n-1) - L^j(n)\right| \tag{4.39}$$

O tamanho final do dicionário reduzido é proporcional à atividade do vizinho esquerdo, avaliada pela equação (4.39). O número de elementos N_S^j do dicionário $\mathcal{D}_S^j$ é calculado como

$$N_S^j = \max\left(16\lfloor A(L^j)\rfloor, 200\right). \tag{4.40}$$

Assim como no APM-CM-MMP, as partições de dicionário e as melhorias com mais blocos de atualização (apenas deslocamentos para a esquerda e uma união) também são utilizadas. As novas inclusões no dicionário estão indicadas na Figura 4.26. A inclusão (i) corresponderia (não está explicitamente indicada) a atualização padrão, a (ii) indica os deslocamentos de $\frac{Q^j}{4}$ e $\frac{Q^j}{2}$ e a (iii) a união com um bloco de comprimento $\frac{Q^j}{2}$. O dicionário inicial é composto como descrito na seção 4.1.1.

O dicionário de deslocamentos

Como pode ser visto na Figura 4.18, um sinal de ECG típico é aproximadamente periódico. O algoritmo MMP, por sua vez, adapta o seu dicionário ao sinal de entrada e tem a capacidade de aprender o padrão em um período. Entretanto, haja vista que os comprimentos dos segmentos podem não ser múltiplos do período do sinal, também é necessário aprender várias versões deslocadas de um período para uma representação eficiente do sinal. De modo a melhorar o desempenho para sinais aproximadamente periódicos, pode-se utilizar um *dicionário de deslocamentos* $\mathcal{D}^D$, o qual inclui versões deslocadas de aproximações de segmentos já codificados,

aumentando a adaptabilidade do algoritmo. Tal dicionário pode ser implementado mantendo-se as M últimas amostras do sinal reconstruído em um vetor $\mathbf{V}_D^j$, ou seja,

$$\mathbf{V}_D^j = \begin{pmatrix} \hat{x}(Fp(j)-M) & \hat{x}(Fp(j)-M+1) & \dots & \hat{x}(Fp(j)-1) \end{pmatrix}. \tag{4.41}$$

O dicionário de deslocamentos é então definido como:

$$\begin{aligned} \mathbf{u}(p) &= \begin{pmatrix} V_D^j(p) & V_D^j(p+1) & \dots & V_D^j(p+Q^j-1) \end{pmatrix} \\ \mathcal{D}^D &= \{\mathbf{u}(p)\}, p = 0, 1, \dots, M - Q^j. \end{aligned} \tag{4.42}$$

Assim como feito para o dicionário principal, um dicionário de deslocamentos reduzido $\mathcal{D}_S^D$, composto pelos N_S^D vetores menos rugosos em $\mathcal{D}^D$ (como definido em (4.7)) é também criado durante a codificação de cada bloco. O tamanho N_S^D é fixo em $\frac{Q^j}{4}$.

Ao tentar codificar $\mathbf{X}^j$, o MMP procura em $\mathcal{D}_S^D$ pelo melhor elemento para codificar $\mathbf{X}^j$, levado em consideração o compromisso taxa-distorção. Define-se então

$$\begin{aligned} J_D^j(p) &= d\left(\mathbf{u}(p), \mathbf{X}^j\right) + \lambda R(p), \mathbf{u}(p) \in \mathcal{D}_S^D \\ p_j &= \min_p J_D^j(p) \\ \mathbf{v}_D^j &= \mathbf{u}(p_j), \end{aligned} \tag{4.43}$$

onde $R(p)$ é a taxa necessária para se codificar p. Em outras palavras, o MMP procura pelo elemento em $\mathcal{D}_S^D$ que minimiza o custo lagrangeano $J_D^j(p)$, como definido na seção 2.1.

Durante o procedimento de otimização, o mínimo custo lagrangeano do dicionário de deslocamentos reduzido $J_D^j(p_j)$ é comparado ao custo lagrangeano mínimo proporcionado por $\mathcal{D}_S$, selecionando-se o menor valor. Além disso, é necessário mais um *flag* para identificar a origem do elemento ($\mathcal{D}^D$ ou $\mathcal{D}$). O dicionário de deslocamentos $\mathcal{D}^D$ apresenta o mesmo conceito do dicionário de vizinhança $\mathcal{D}^N$ da seção 4.1.2, com a diferença de que os deslocamentos são apenas na horizontal. Neste tipo de abordagem, o parâmetro mais importante é o tamanho da vizinhança, pois o mesmo define a taxa utilizada para a codificação dos seus índices. Para sinais de ECG, uma vizinhança de 1024 amostras se mostrou suficiente.

Resultados de simulações

O MMP unidimensional descrito nesta seção foi implementado e aplicado à compressão de sinais de ECG extraídos da base de dados MIT/BIH. Para que fossem feitas comparações com os trabalhos em [104], [105], [4] e [3], que representavam o estado da arte na época em que o presente trabalho foi desenvolvido/publicado (com exceção do último, que é mais recente), utilizaram-se os dois canais dos seguintes registros (11 no total): 100, 101, 102, 103, 107, 109, 111, 115, 117, 118 e 119.

A Figura 4.27 mostra os resultados de simulação (PRD versus taxa de bits) para todos os 11 registros (cada registro foi completamente processado).

A Figura 4.28 mostra uma comparação com os resultados médios apresentados em [105] para o SPIHT [104] e o WT-DCCR-TV-VQ [105]. É importante mencionar que os resultados apresentados para o SPIHT foram retirados de [105] e não de [104]. Isto foi feito porque os resultados em [105] correspondem à compressão do registro completo (em torno de 30 minutos), diferentemente dos presentes em [104], que correspondem apenas aos primeiros 10 minutos (isto é justo porque resultados para 30 minutos são, em geral, levemente melhores que os para 10 minutos). Os dados para a Figura 4.28 foram obtidos realizando-se a média dos valores de PRD obtidos para ambos os canais, dos 11 registros, em cada taxa. Desta Figura, percebe-se que o MMP apresenta desempenhos superiores aos do SPIHT e do WT-DCCR-TV-VQ para todas as taxas. Para uma comparação justa do método proposto com o apresentado em [4], utilizou-se a Tabela 4.3, que contém resultados para os primeiros 10 minutos dos registros apresentados na Figura 4.28. Isso foi necessário porque os resultados em [4] correspondem a 10 minutos e não aos sinais completos. Comparações com os métodos apresentados em [3], [6], [5] e outros resultados de [4], também com os primeiros 10 minutos de alguns registros, estão disponíveis na Tabela 4.4.

Embora o MMP apresente desempenhos superiores ao SPIHT [104] e ao WT-DCCR-TV-VQ [105], o mesmo não acontece para os métodos baseados no JPEG2000 em [4] e [3]. Uma característica importante desses métodos é a normalização de período, na qual uma imagem de ECG é montada, com um período por linha, o que proporciona uma exploração eficaz da redundância inter-batimento do sinal de ECG. Se este método for incorporado ao MMP, o que será feito na próxima seção, o

dicionário de deslocamentos, por exemplo, seria utilizado de maneira muito eficaz, pois haveria uma tendência no uso de índices com deslocamento de um período completo.

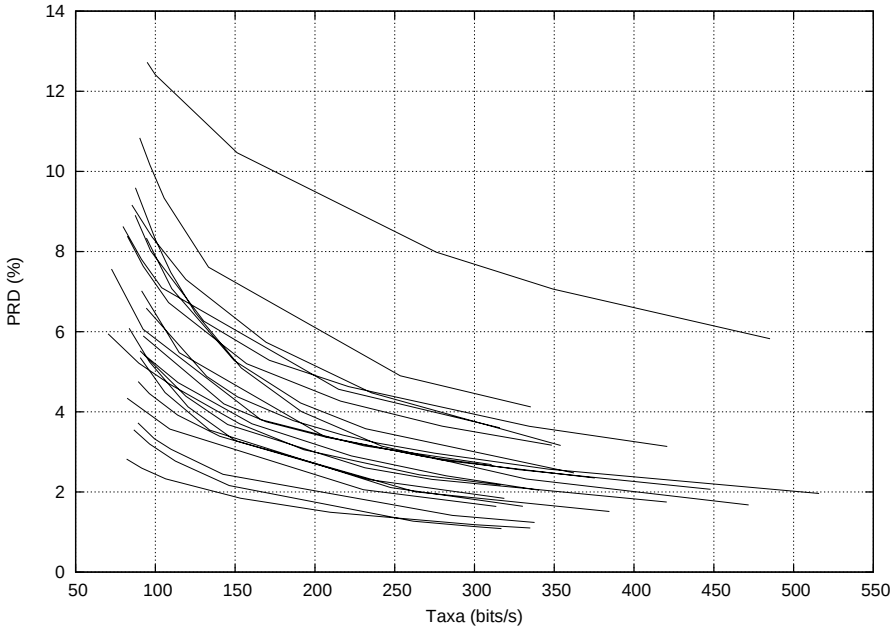


Figura 4.27: Desempenho do MMP para os sinais da base de dados MIT/BIH, comprimento total dos registros ($PRD \times CDR$).

Um fato interessante reside no conjunto de sinais de ECG a se utilizar para a comparação com outros algoritmos. Apesar de em extenso conjunto de dados ter sido utilizado, com vários registros, é possível restringir bastante os sinais de teste, escolhendo-se apenas os mais significativos. Os registros mais interessantes para testes de desempenho, dado o que é mais utilizado na literatura e também com base no que foi visto durante este trabalho, são os seguintes: 100, 117 e 119. Se mais sinais forem necessários, também podem ser utilizados: 102, 107 e 115.

As Figuras 4.29, 4.30, 4.31 e 4.32 contém 1 segundo do canal 0 dos registros 100, 107, 117 e 119 respectivamente, em diferentes valores de PRD e taxas.

Análise dois sinais reconstruídos

Como mostrado na última seção, o desempenho do algoritmo proposto é bom com relação aos resultados de PRD. Entretanto, a utilidade dos sinais reconstruídos para diagnóstico ainda deve ser verificada.

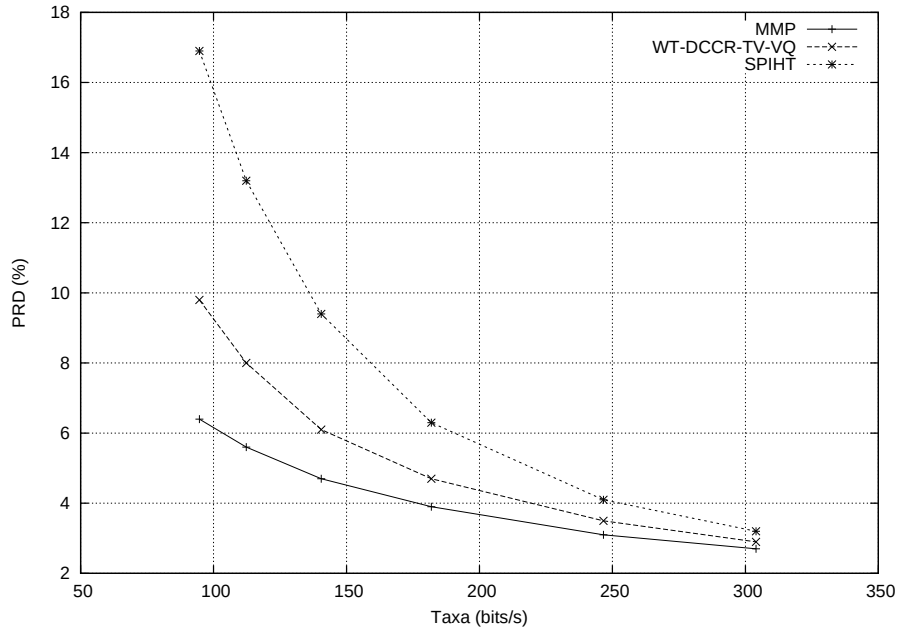


Figura 4.28: Desempenho médio do MMP comparado ao SPIHT e ao WT-DCCR-TV-VQ for para o conjunto de teste (comprimento total).

Tabela 4.3: Comparação de desempenho ($PRD \times CR$) entre o MMP e o JPEG2000, primeiros 10 minutos dos registros.

CR	**JPEG2000** [4]	**MMP**
8:1	1.52	1.96
10:1	1.86	2.34
16:1	2.74	3.29
20:1	3.26	3.86

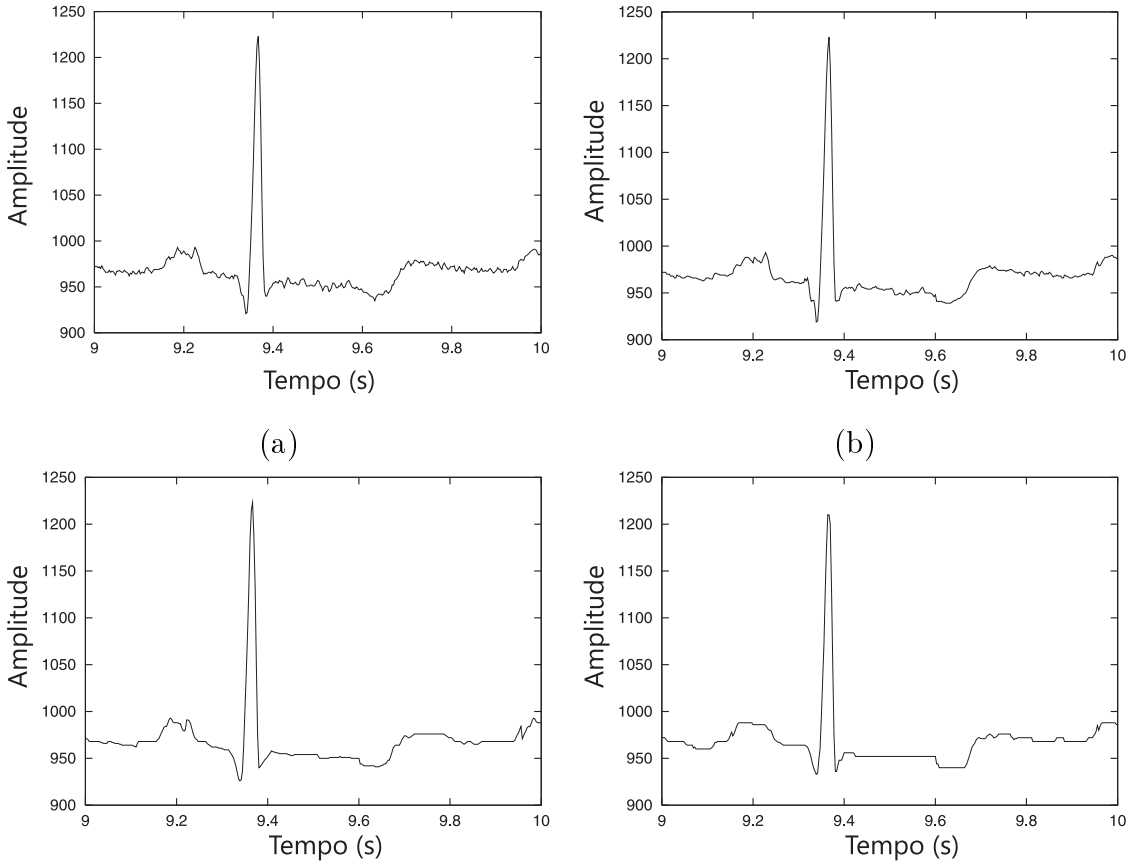


Figura 4.29: Registro 100 reconstruído com o MMP: (a) Sinal original; (b) Sinal reconstruído com $PRD = 2{,}67$ e $CDR = 305{,}41$; (c) Sinal reconstruído com $PRD = 3{,}84$ e $CDR = 164{,}97$; (d) Sinal reconstruído com $PRD = 6{,}04$ e $CDR = 89{,}57$.

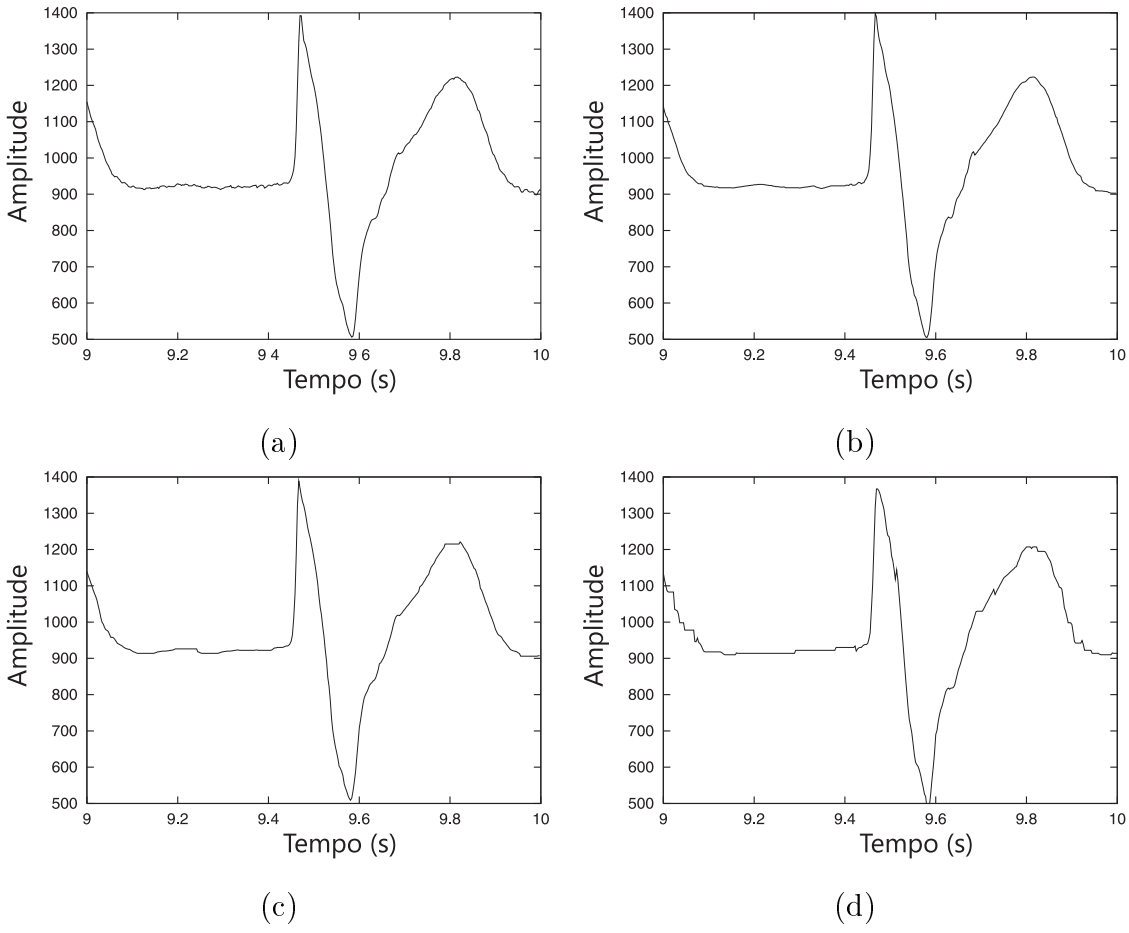


Figura 4.30: Registro 107 reconstruído com o MMP: (a) Sinal original; (b) Sinal reconstruído com $PRD = 1{,}68$ e $CDR = 471{,}84$; (c) Sinal reconstruído com $PRD = 3{,}15$ e $CDR = 242{,}10$; (d) Sinal reconstruído com $PRD = 8{,}25$ e $CDR = 100{,}25$.

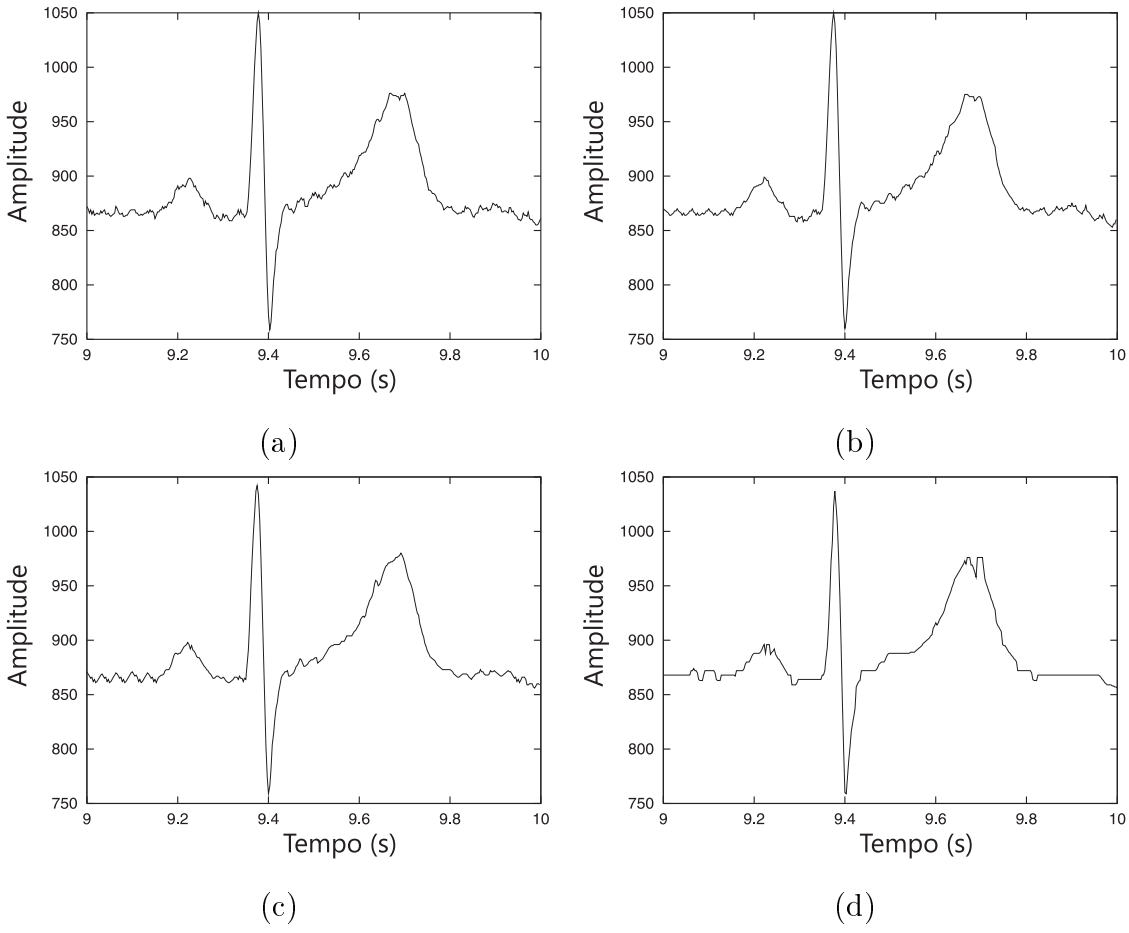


Figura 4.31: Registro 117 reconstruído com o MMP: (a) Sinal original; (b) Sinal reconstruído com $PRD = 0{,}88$ e $CDR = 469{,}91$; (c) Sinal reconstruído com $PRD = 1{,}38$ e $CDR = 236{,}15$; (d) Sinal reconstruído com $PRD = 2{,}40$ e $CDR = 100{,}65$.

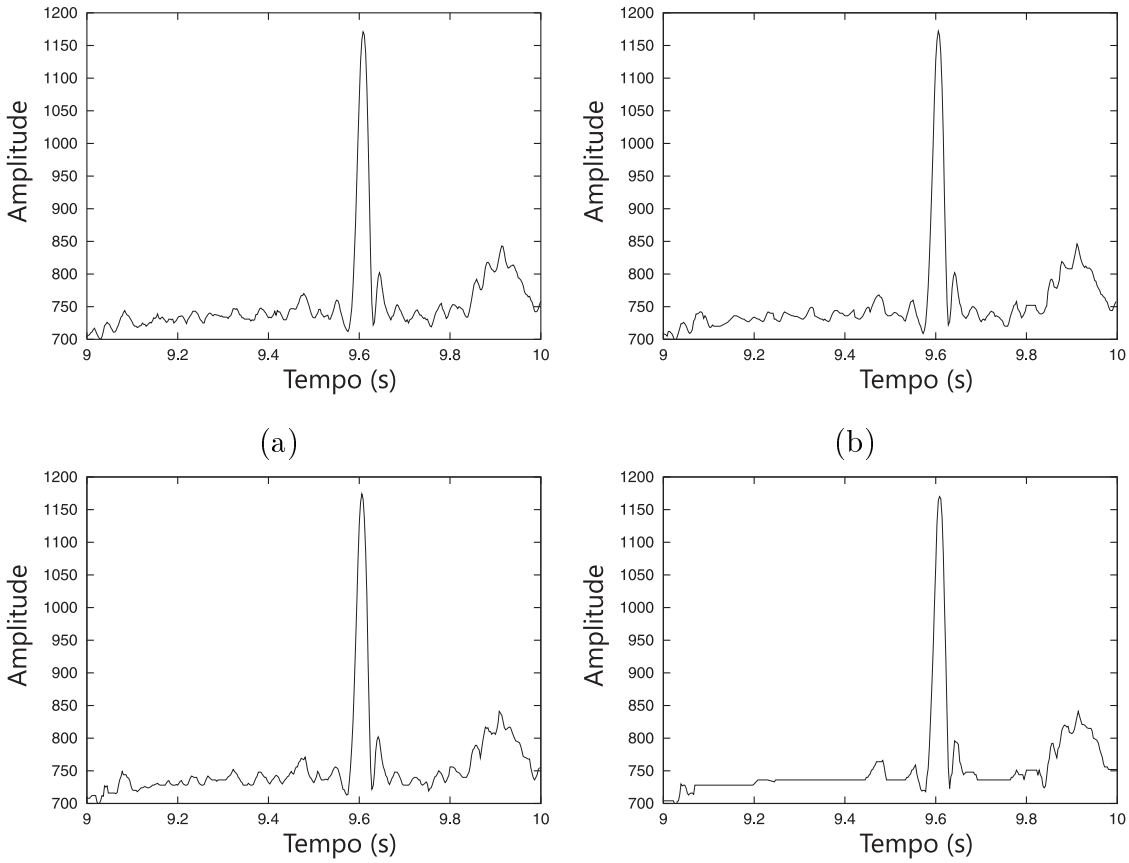


Figura 4.32: Registro 119 reconstruído com o MMP: (a) Sinal original; (b) Sinal reconstruído com $PRD = 1{,}10$ e $CDR = 390{,}30$; (c) Sinal reconstruído com $PRD = 1{,}41$ e $CDR = 286{,}77$; (d) Sinal reconstruído com $PRD = 2{,}36$ e $CDR = 147{,}79$.

Utilizando-se os conceitos básicos sobre ECG apresentados anteriormente, a informação de diagnóstico presente nos sinais reconstruídos das Figuras 4.29 até 4.32 será discutida[1]. As Figuras 4.29(b) até 4.29(d) mostram um sinal de ECG comprimido a 305,41, 164,97 e 89,57 bps, respectivamente. Os dois primeiros sinais reconstruídos apresentam distorção moderada nas ondas P e T, o que os torna adequados para a determinação de um diagnóstico. Entretanto, o sinal reconstruído a 89,57 bps (Figura 4.29(d)) apresenta muita distorção na sua onda T e no segmento ST, o que poderia comprometer o diagnóstico de desordens da artéria coronária como isquemia e angina (se existirem). É importante ressaltar também que, na prática médica, o sinal de ECG é normalmente filtrado antes da análise (para que o ruído da rede elétrica e o tremor muscular sejam removidos) e seria muito parecido com os sinais das Figuras 4.29(b) a 4.29(c).

O ECG na Figura 4.30 apresenta bloqueio de ramo nos ventrículos. Isto pode ser deduzido da duração alongada do complexo QRS. Essa característica é preservada até mesmo no sinal reconstruído a taxa mais baixa (Figura 4.30(d)), significando que todos os sinais comprimidos poderiam ser utilizados para análise e diagnóstico.

Outro sinal de ECG é mostrado na Figura 4.31. Embora os sinais reconstruídos não estejam muito deformados, os dois nas taxas mais baixas (Figuras 4.31(c) – 236,15 bps e 4.31(d) – 100,65 bps) apresentam alterações no topo da onda T. Apesar disso, tais problemas não afetariam o diagnóstico médico, pois as estruturas mais importantes não foram comprometidas.

O último sinal, mostrado na Figura 4.32, é muito ruidoso. Portanto, seria difícil detectar a onda P até mesmo no sinal original. Na verdade, cada segmento num sinal de ECG típico está mais relacionado a um tipo de derivação (canal). Para uma análise segura da onda P deste sinal, outra derivação deveria ser observada. Os dois primeiros sinais reconstruídos (Figuras 4.32(b) – 390,30 bps e 4.32(c) – 286,77 bps) preservam a maior parte das características do sinal original e poderiam ser utilizados para diagnóstico. O último sinal reconstruído (Figura 4.32(d) – 147,79 bps) apresenta uma deformação moderada em todos os segmentos e ainda poderia ser utilizado para diagnóstico.

[1] A análise dos sinais reconstruídos foi possível devido a uma grande contribuição da Dra. Katia do Nascimento Couceiro, professora adjunta da Universidade Federal do Amazonas.

Tabela 4.4: Comparação de desempenho ($PRD \times CR$) entre o MMP, os algoritmos baseados no JPEG2000 em [3, 4] e os algoritmos em [5, 6].

Algoritmo	**Registro**	**CR**	**PRD**
Lee et. al [5]	100	24 : 1	8.10
Chou et. al app. 2 [3]	100	24 : 1	4.06
Norm - JPEG2000	100	24 : 1	6.01
MMP	100	24 : 1	3.82
Chou et. al app. 2 [3]	117	13 : 1	1.18
MMP	117	13 : 1	1.19
Wei et. al [6]	117	10 : 1	1.18
Bilgin et. al [4]	117	10 : 1	1.03
Chou et. al app. 1 [3]	117	10 : 1	0.98
MMP	117	10 : 1	1.00
Bilgin et. al [4]	117	8 : 1	0.86
MMP	117	8 : 1	0.86
Lee et. al [5]	119	24 : 1	10.5
Bilgin et. al [4]	119	21.6 : 1	3.76
Chou et. al app. 2 [3]	119	20.9 : 1	1.81
Norm - JPEG2000	119	20.9 : 1	3.90
MMP	119	20.9 : 1	2.04
Chou et. al app. 2 [3]	119	10 : 1	1.03
Norm - JPEG2000 [3]	119	10 : 1	1.37
MMP	119	10 : 1	1.12

Como pode ser entendido das análises acima, sinais de ECG codificados com o MMP preservam a maior parte da informação utilizada em diagnósticos, mesmo em baixas taxas. Entretanto, para um diagnóstico seguro, taxas acima de 150,00 bps devem ser utilizadas. É importante ressaltar que o MMP sempre preserva, com grande precisão, o complexo QRS.

4.2.4 O MMP com normalização de período

Sabe-se que, em exames onde o sinal de eletrocardiograma revela alguma desordem, há uma grande variação nos ciclos de ECG durante o mesmo. O período dos batimentos cardíacos e a forma das sub-ondas mudam ao longo do sinal e o MMP acaba tendo que aprender diferentes padrões para cada período de batimento. Essa característica não permite ao MMP utilizar o dicionário de maneira eficaz, pois os padrões anteriormente aprendidos não se repetem com muita freqüência nos períodos subseqüentes.

Para que não haja perda de desempenho no MMP devido à variação do comprimento de cada batimento, pode-se utilizar a normalização de período empregada por [4] e [3], mencionada na seção 4.2.3. Essa extensão, além de reduzir a necessidade de aprendizado e adaptação por parte do MMP, tem o potencial de maximizar a utilização de índices específicos do dicionário de deslocamentos, reduzindo a taxa necessária para a codificação do sinal.

A normalização de período

Para que a exploração das dependências intra-batimento seja realizada de forma eficaz, incorpora-se uma normalização de período ao algoritmo da seção 4.2.3, conforme descrito em [4]. O procedimento começa com a segmentação do sinal de ECG, na qual cada período do batimento cardíaco é identificado e separado, como ilustrado na Figura 4.33. Os algoritmos utilizados para a detecção dos períodos foram os disponibilizados em [108]. Dado que cada período de ECG pode apresentar uma duração diferente, os mesmos foram normalizados para um valor comum, através do método descrito em [4]. Seguindo este procedimento, um período do sinal de ECG original $X = [x(0)\ x(1)\ \ldots\ x(N_o - 1)]$ pode ser convertido em um período

normalizado $X_n = [x_n(0)\ x_n(1)\ \ldots\ x_n(N_n - 1)]$, que é obtido utilizando-se

$$\begin{aligned} X_n(m) =& \hat{X}(h^*) \\ h^* =& \frac{m.(N_o - 1)}{(N_n - 1)}, \end{aligned} \tag{4.44}$$

onde $\hat{X}(h^*)$ é a versão interpolada de $X(n)$, N_o é o comprimento do período, N_n é o comprimento do período normalizado e $m = 0, 1,\ \ldots\ N_n - 1$.

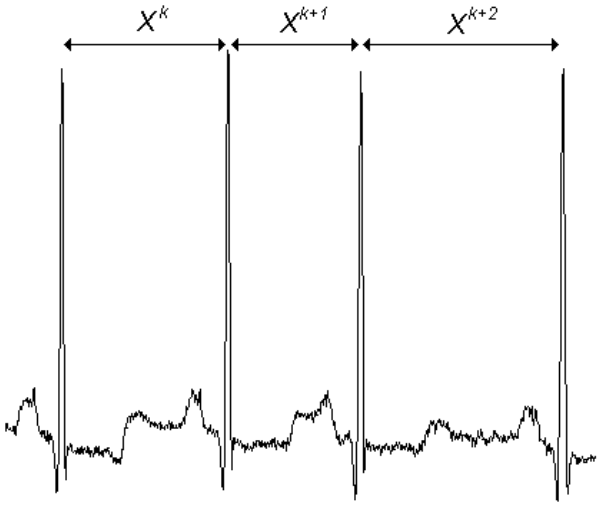


Figura 4.33: Detecção de períodos de ECG.

A interpolação [109] é obtida utilizando-se os métodos de interpolação por *spline cúbica* descritos em [110]. Ao reconstruir o sinal original, o decodificador necessita dos períodos originais como informação auxiliar, que foram codificados com um codificador aritmético diferencial e adicionados ao cabeçalho do arquivo comprimido. Os períodos originais podem então ser recuperados com a mesma transformação descrita em (4.44).

N_n é escolhido com sendo a parte inteira da média dos comprimentos de todos os períodos detectados, mantendo-se o sinal normalizado com um comprimento final menor ou igual ao seu comprimento original.

É interessante observar que a interpolação já introduz alguma distorção e a sua escolha é decisiva para um bom desempenho. Geralmente, *splines* cúbicas são preferidas devido a sua habilidade em aproximar *kernels* cúbicos to tipo C2-contínuo [110].

A redução de faixa dinâmica

Sabe-se que o desempenho do algoritmo MMP é sensível à faixa dinâmica do sinal a ser codificado, sendo que a sua redução pode levar a uma melhoria de desempenho. Além disso, o algoritmo já se mostrou eficiente na codificação de sinais gaussianos [10] e resíduos [14], evidenciando uma grande capacidade para o processamento de sinais com característica descorrelacionada.

Dado que já se possuem segmentos com o mesmo comprimento, a faixa dinâmica é facilmente reduzida através do cálculo do período normalizado médio (*Average Normalized Period Segment*), definido como

$$X_{n_{avg}}(m) = \frac{1}{M} \sum_{k=0}^{M-1} X_n^k(m), \tag{4.45}$$

no qual M é o número de períodos, X_n^k é o k-ésimo período normalizado e $m = 0, 1, \ \ldots \ N_n - 1$. Cada período normalizado e com faixa dinâmica reduzida (*Reduced Dinamic Range*) é computado através da subtração do período normalizado médio de cada período normalizado, ou seja,

$$X_{n_{rdr}}^k(m) = X_n^k(m) - X_{n_{avg}}(m), \tag{4.46}$$

onde $m = 0, 1, \ \ldots \ N_n - 1$.

O período normalizado médio é também codificado com um codificador aritmético diferencial e adicionado ao cabeçalho do arquivo comprimido. O sinal processado pelo MMP é então composto por períodos de ECG originais normalizados e com faixa dinâmica reduzida. Os passos expostos estão ilustrados na Figura 4.34.

Resultados de simulações

Novamente, o MMP foi aplicado aos registros 100, 101, 102, 103, 107, 109, 111, 115, 117, 118 e 119 da base de dados de arritmia MIT/BIH e comparado com o primeiro algoritmo desenvolvido na seção 4.2.3 e com os trabalhos em [3–6, 104]. Agora, apenas os primeiros 10 minutos de cada registro foram utilizados, proporcionando resultados compatíveis com os métodos em comparação. Os resultados estão resumidos nas Tabelas 4.5 e 4.6. A primeira mostra os resultados de simulação (PRD versus taxa de bits) para todos os 11 registros, realizando-se a média dos valores de PRD obtidos para ambos os canais; a segunda apresenta resultados isolados

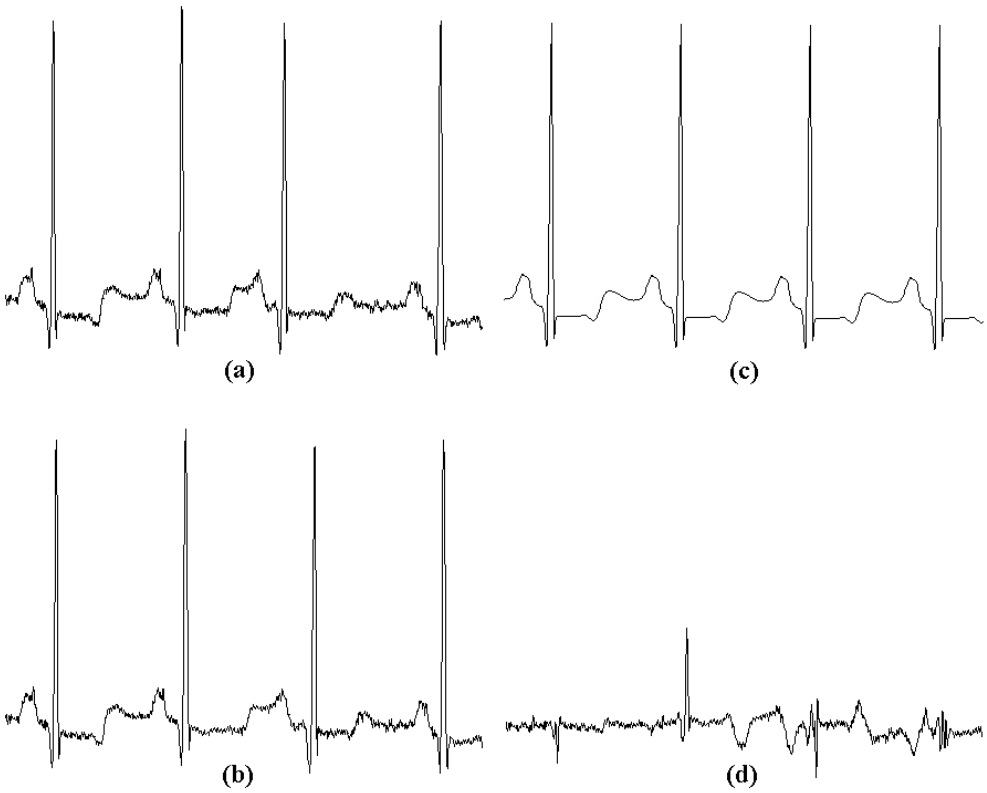


Figura 4.34: Etapas de processamento: (a) Sinal original; (b) Sinal normalizado; (c) Concatenação de períodos normalizados médios; (d) Sinal normalizado e com faixa dinâmica reduzida.

para os registros mais significativos. Percebe-se que as modificações propostas melhoraram o desempenho do MMP em todas as taxas de compressão testadas. Além disso, o método proposto agora é competitivo com o método baseado no JPEG2000, apresentado em [4].

É importante notar que este desenvolvimento mostrou uma nova área de pesquisa para a compressão de sinais de ECG: técnicas de pré-processamento. Tais ferramentas podem ser agregadas a praticamente qualquer esquema de compressão, com o potencial de melhorar o desempenho do mesmo. Dada esta constatação, as próximas seções relacionadas a ECG tentarão explorar tais técnicas, aumentando a correlação do sinal apresentado ao codificador.

Tabela 4.5: Comparação de desempenho em termos de PRD entre MMP, JPEG2000 e SPIHT, para vários valores de taxa de compressão(CR).

CR	**JPEG2000 [4]**	**MMP norm. per.**	**MMP seção 4.2.3**	**SPIHT [104]**
4:1	0.78	1.02	1.05	1.19
8:1	1.52	1.83	1.96	2.46
10:1	1.86	2.14	2.34	2.96
16:1	2.74	2.95	3.29	4.85
20:1	3.26	3.41	3.86	6.49

4.2.5 Novas ferramentas para a compressão de sinais de ECG

Alguns autores têm desenvolvido métodos de compressão de ECG como imagem [3–7], o que facilita a exploração das dependências intra e inter-batimento, principalmente devido à estrutura bidimensional. Tais métodos podem ser divididos em três etapas: detecção de período, pré-processamento e transformação. O segundo passo já foi investigado por alguns autores e consiste basicamente nas técnicas de detecção e normalização de período [3,4,6] apresentadas na seção anterior. Entretanto, há fatores que evitam que os períodos normalizados apresentem alta correlação, comprometendo o desempenho do decodificador.

Nesta seção, propõem-se duas novas técnicas a serem incorporadas ao passo de pré-processamento, que exploram a estrutura do sinal de maneira mais eficiente, através do aumento da correlação entre períodos de ECG adjacentes: a equalização DC, que grampeia todos os segmentos de período no mesmo nível DC, e a ordenação por complexidade (*Complexity Sorting* - CS), que rearranja os períodos em ordem decrescente de semelhança com o de menor variância. Como resultado, a imagem de ECG fica com áreas de períodos semelhantes, onde aparecem regiões de valores constantes, facilitando a exploração das dependências.

Há métodos de compressão que utilizam o JPEG2000 como codificador para o ECG bidimensional, geralmente apresentando bons resultados. Esta seção também apresenta resultados com o H.264/AVC intra-quadros, validando a eficácia das técnicas apresentadas.

Tabela 4.6: Comparação de desempenho ($PRD \times CR$) entre o MMP com normalização de período, o MMP da seção 4.2.3, os algoritmos baseados no JPEG2000 em [3,4] e os algoritmos em [5,6].

Algoritmo	**Registro**	**CR**	**PRD**
Lee et. al [5]	100	24 : 1	8.10
Chou et. al app. 2 [3]	100	24 : 1	4.06
Norm - JPEG2000	100	24 : 1	6.01
MMP seção 4.2.3	100	24 : 1	3.82
MMP norm. per.	100	24 : 1	3.41
Chou et. al app. 2 [3]	117	13 : 1	1.18
MMP seção 4.2.3	117	13 : 1	1.19
MMP norm. per.	117	13 : 1	0.98
Wei et. al [6]	117	10 : 1	1.18
Bilgin et. al [4]	117	10 : 1	1.03
Chou et. al app. 1 [3]	117	10 : 1	0.98
MMP seção 4.2.3	117	10 : 1	1.00
MMP norm. per.	117	10 : 1	0.85
Bilgin et. al [4]	117	8 : 1	0.86
MMP seção 4.2.3	117	8 : 1	0.86
MMP norm. per.	117	8 : 1	0.75
Lee et. al [5]	119	24 : 1	10.5
Bilgin et. al [4]	119	21.6 : 1	3.76
Chou et. al app. 2 [3]	119	20.9 : 1	1.81
Norm - JPEG2000	119	20.9 : 1	3.90
MMP seção 4.2.3	119	20.9 : 1	2.04
MMP norm. per.	119	20.9 : 1	2.00
Chou et. al app. 2 [3]	119	10 : 1	1.03
Norm - JPEG2000 [3]	119	10 : 1	1.37
MMP seção 4.2.3	119	10 : 1	1.12
MMP norm. per.	119	10 : 1	1.10

Técnicas existentes na literatura

O processamento do sinal ECG como imagem exige a transformação do registro original em um sinal bidimensional. Isso é obtido através da detecção de cada período, seguida por uma segmentação e pela montagem de cada período em uma linha da matriz. Nesta seção, novamente a detecção do complexo QRS é executada com os algoritmos em [108]. De fato, existem muitos outros algoritmos de detecção presentes na literatura [111–113]; tanto os algoritmos em [108] quanto em [111] apresentaram bons resultados e, dado que o desempenho do esquema de compressão afeta bastante a qualidade do ECG reconstruído [3], a escolha final deve ser feita cuidadosamente; tanto os algoritmos em [108] quanto em [111] proporcionam bons resultados.

Depois, o sinal é segmentado e montado novamente como imagem, escolhendo-se o valor máximo do complexo QRS como fronteira de segmentação, deixando meio pico em cada extremo da linha. Entretanto, em [3] os autores argumentam que isso criaria descontinuidades, tornando a compressão do sinal mais difícil. Na verdade, tal condição é muito dependente do compressor adotado e o deslocamento de amostra sugerido em [3] pode até mesmo piorar os resultados, devido à perda de simetria. A matriz resultante desse processo pode ser vista nas Figuras 4.35(a) e 4.36(a) para os registros 100 e 119 da base de dados MIT/BIH, respectivamente. Embora todos os períodos estejam alinhados na esquerda, os mesmos não apresentam o mesmo comprimento, normalmente devido à presença de alguma desordem ou variações na condição do paciente.

Para se corrigir este comportamento e proporcionar uma melhor exploração das dependências intra-batimento, Wei et. al em [6] propuseram a normalização de período, também adotada por [4] e utilizada na seção 4.2.4, que escalona o comprimento de todos os períodos para um mesmo valor.

Em [3], outra técnica de pré-processamento foi introduzida, chamada de ordenação de período, que consiste no rearranjo de todos os períodos com base nos seus comprimentos. É interassante notar que este processamento supõe que períodos com comprimentos similares são muito correlacionados, o que pode não ser válido para muitos tipos de sinais de ECG (*e.g.*, quando o paciente sofre de alguma doença).

Novas técnicas de pré-processamento

Ao se analisarem as imagens de ECG com normalização de período mostradas nas Figuras 4.35(b) e 4.36(b), percebe-se que linhas adjacentes não apresentam o mesmo nível DC. Isso é normal ao longo de um exame, pois as condições do paciente mudam, porém, prejudica o desempenho do compressor utilizado. A equalização DC foi desenvolvida para tratar este problema. Os níveis DC originais podem ser calculados como

$$DC_k = \left\lfloor \frac{1}{L_N} \sum_{i=1}^{L_N} x_k(i) \right\rfloor, \tag{4.47}$$

onde $x_k(i)$ são as amostras de sinal do k-ésimo período e L_N é o comprimento normalizado. Com isso, todos os períodos são grampeados no mínimo valor DC possível através de

$$x_k^{DC}(i) = x_k(i) - (DC_k - DC_{min}) + \delta, \tag{4.48}$$

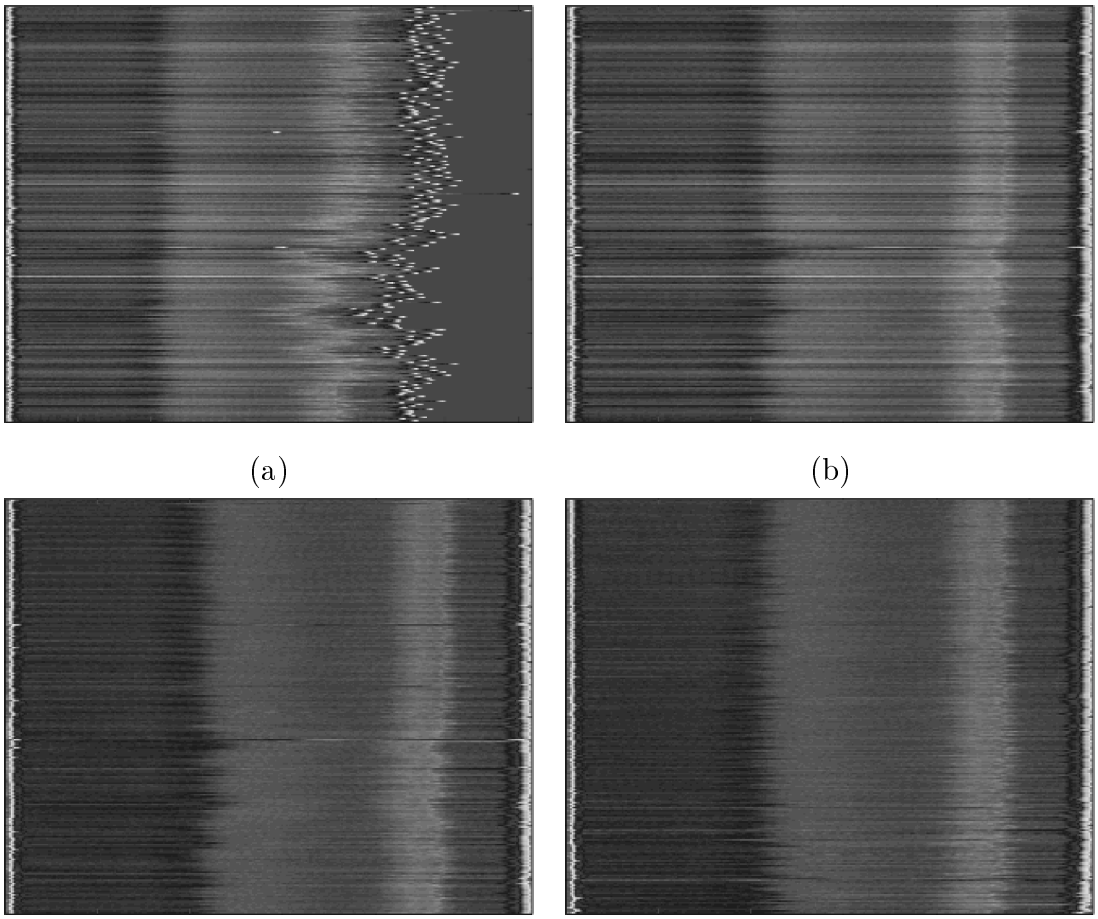
onde $x_k^{DC}(i)$ são as amostras de sinal dos períodos equalizados, DC_{min} é o nível DC mínimo encontrado e δ é uma constante para garantir que o valor mínimo de qualquer amostra seja maior ou igual a zero. O resultado deste pré-processamento está ilustrado na Figura 4.35(c).

Após a conclusão da equalização, um problema persiste: linhas adjacentes ainda podem ser bem diferentes, o que é visto nas Figuras 4.35(c) e 4.36(c) como descontinuidades agudas ao longo da direção vertical. Este problema pode ser minimizado com a ordenação de período descrita em [3], porém, a mesma pode falhar, pois nem sempre períodos com comprimentos próximos são semelhantes.

Nesta seção, propõe-se uma abordagem simples e direta para a solução deste problema: reorganizar os períodos com base em uma métrica confiável de proximidade entre os mesmos. Ao término da equalização DC, o elemento de menor variância é posicionado na primeira linha da matriz de imagem e os períodos restantes são rearranjados em ordem decrescente de proximidade com o primeiro. Verificou-se que o erro médio quadrático é um bom critério de semelhança, sendo dado por

$$MSE_k = \frac{1}{L_N} \sum_{i=1}^{L_N} (x_k(i) - x_{\sigma_{\min}}(i))^2, \tag{4.49}$$

onde $x_{\sigma_{\min}}(i)$ são as amostras do período com a menor variância e $x_k(i)$ são as amostras do k-ésimo período de ECG. Este processamento é chamado de ordenação



(a) (b)

(c) (d)

Figura 4.35: Efeitos das técnicas de pré-processamento propostas no registro 100 da base de dados de ECGs MIT/BIH. (a) Períodos originais. (b) Períodos normalizados. (c) Equalização DC aplicada a (b). (d) Ordenação por complexidade aplicada a (c).

por complexidade e o seu resultado pode ser visto na Figura 4.35(d). A imagem gerada é bem mais suave e pode ser comprimida com grande eficiência por um bom compressor de imagens.

Quando a matriz resultante das técnicas de equalização DC e ordenação por complexidade é comparada com as matrizes geradas pelas outras técnicas de pré-processamento (Figuras 4.35(a) e 4.36(a), 4.36(b) e 4.36(b) and 4.35(c) e 4.36(c)), percebe-se que a correlação entre linhas adjacentes tende a aumentar, favorecendo o desempenho do codificador bidimensional.

Um codificador completo, utilizando as técnicas propostas, está ilustrado na Figura 4.37. Primeiramente, os picos dos complexos QRS detectados, com os algoritmos em [108]; cada período é delimitado em ambos os lados com meio pico de

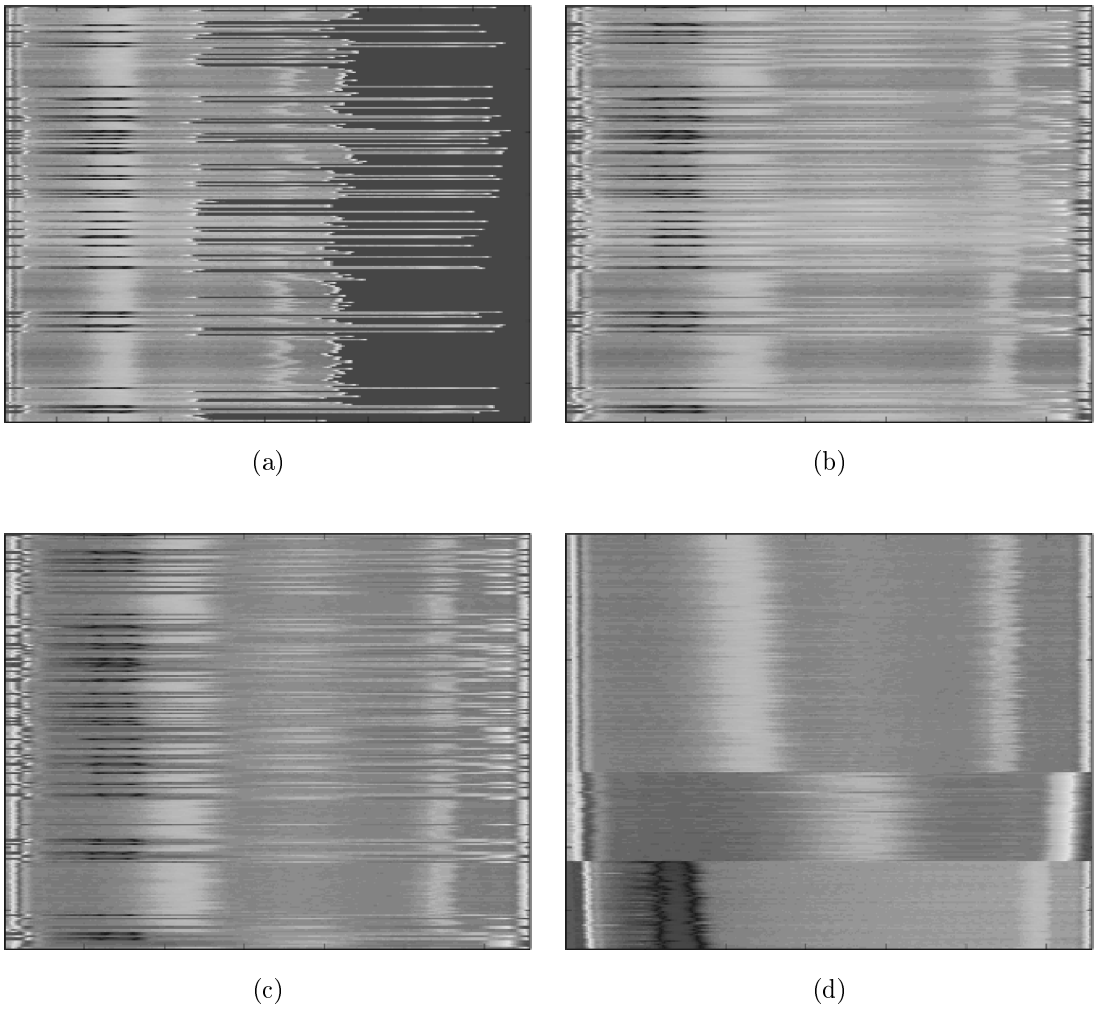


Figura 4.36: Efeitos das técnicas de pré-processamento propostas no registro 119 da base de dados de ECGs MIT/BIH. (a) Períodos originais. (b) Períodos normalizados. (c) Equalização DC aplicada a (b). (d) Ordenação por complexidade aplicada a (c).

complexo QRS e ocupa uma linha inteira na imagem de ECG. Após este procedimento, os períodos são normalizados para um comprimento comum, igual ao valor médio de todos os períodos detectados, através do método descrito na seção 4.2.4. O sinal resultante sofre então a equalização DC, fixando o nível de cada período no menor valor maior ou igual a zero, e a ordenação de complexidade, conforme apresentado nesta seção. A imagem resultante é então passada para o compressor de imagens que, neste trabalho, tem duas opções: o JPEG2000 [70] e o H.264 intra-quadros [69].

Também envia-se a informação auxiliar, juntamente com os dados comprimidos, que é utilizada na recuperação dos comprimentos originais dos períodos, dos níveis

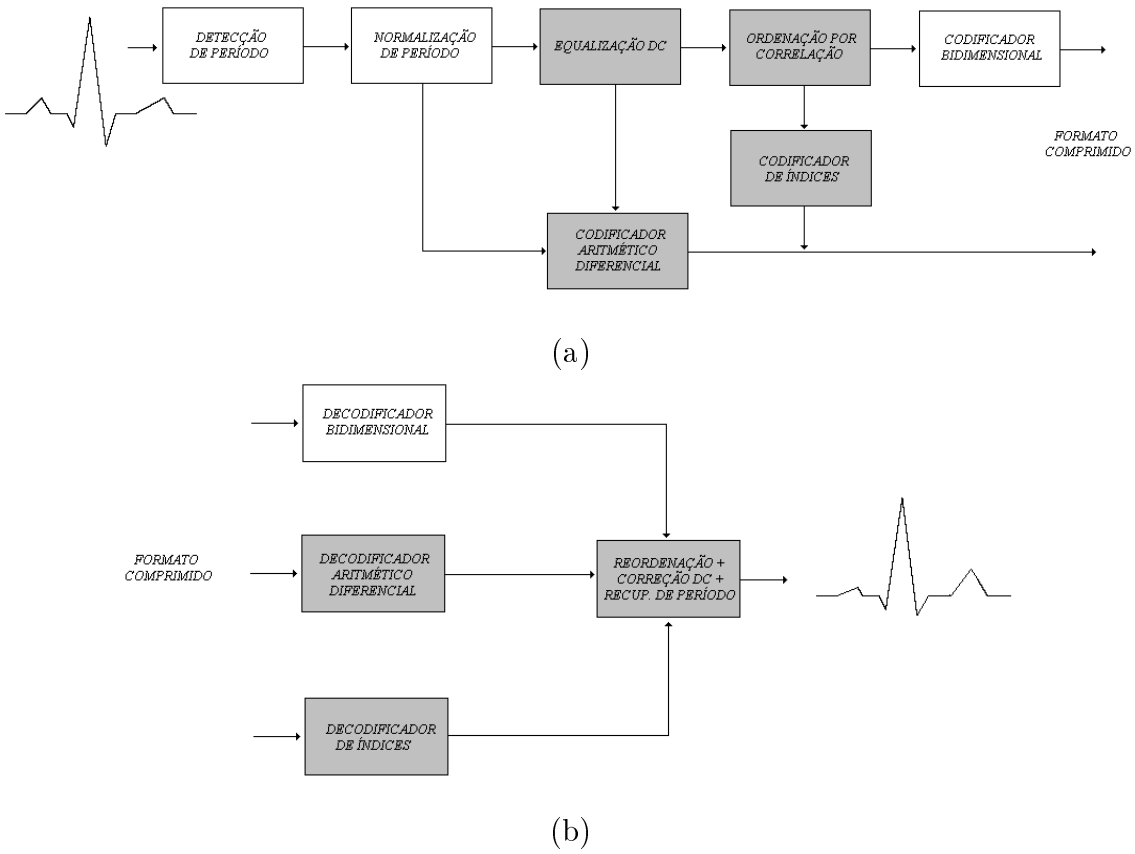


Figura 4.37: Diagrama em blocos do CODEC proposto. (a) Codificador. (b) Decodificador.

DC e da ordenação original; tais informações são codificadas utilizando-se um codificador aritmético [74]. O procedimento de recuperação do sinal é simplesmente o inverso do que foi apresentado e não exige estimações ou cálculos especiais.

Resultados de simulações

Para a avaliação de eficácia das técnicas propostas, realizaram-se testes com dois compressores de imagens, o JPEG2000 [70] e H.264/AVC intra-quadros [69], e três registros da base de dados MIT/BIH: 100, 117 e 119. Estes registros foram escolhidos por serem os mais significativos e devido ao fato dos decodificadores que representam o estado da arte e outros [3–7] apresentarem resultados para os mesmos. Os sinais foram amostrados a 360 Hz com 11 bits de resolução (padrão MIT/BIH). As dimensões das imagens de ECG resultantes dependem do número de períodos detectados e dos seus comprimentos. Os registros 100, 117 e 119 foram convertidos em imagens de dimensões 284×761, 427×506 e 328×660, respectivamente. O

comprimento original de cada período, os níveis DC originais, a ordenação original e o nível de equalização foram codificados com um codificador aritmético e enviados como informação auxiliar. Para se avaliar a qualidade dos sinais reconstruídos, novamente utilizou-se o PRD e o CR. Os resultados estão mostrados nas Tabelas 4.7 e 4.8.

O esquema de compressão com normalização de período, equalização DC e ordenação por complexidade é identificado como Tipo 3. É importante notar que figuras de desempenho também são apresentadas para os outros casos: quando se utiliza apenas normalização de período e equalização DC (Tipo 1) ou quando se utiliza a normalização juntamente com a ordenação por complexidade (Tipo 2), para os registros 100 e 119.

Quando o Tipo 3 é utilizado, melhores desempenhos são atingidos com relação aos resultados proporcionados pelo Tipos 1 ou 2. Além disso, o esquema proposto superou todos os outros métodos para todos os registros. A única exceção foi o registro 119 a uma taxa de compressão de 20,9, reconstruído pelo método apresentado em Chou et. al [3] (para uma taxa de compressão de $10:1$, o método proposto superou o apresentado em [3]).

Dois compressores de imagens foram utilizados nos testes: o JPEG2000 (versão Kakadu disponível em [114]) e o H.264/AVC intra-quadros version 12.1 do *software* de referência (disponível em [115]). Todos os parâmetros padrão foram utilizados para o JPEG2000, variando-se apenas o passo de quantização (fixado em 0.00005); os cabeçalhos foram removidos, pois os mesmos podem ser recuperados no decodificador. O H.264/AVC intra-quadros utilizou o perfil FREXt High 100, juntamente com a otimização taxa-distorção, o filtro de blocagem e o CABAC, além de blocos de transformação e predição de dimensões 8×8. Para um melhor desempenho da predição empregada pelo H.264, as dimensões da imagem de ECG foram fixadas em múltiplos de 16; a última linha foi repetida e o período foi normalizado para

$$N_n = 16 \left\lceil \frac{N_n}{16} \right\rceil . \tag{4.50}$$

Uma avaliação geral dos registros revela que, se a taxa de compressão for alta, o H.264/AVC intra-quadros apresenta um melhor desempenho (em altas taxas, resíduos são mais fáceis de comprimir); caso contrário, sugere-se utilizar o JPEG2000.

Observou-se também que, quando o H.264/AVC intra-quadros é utilizado, a

ordenação por complexidade não é tão vantajosa e pode ser retirada. Sendo assim, a ordenação por complexidade é utilizada somente quando o desvio padrão dos comprimentos dos períodos detectados for maior que 10%.

Tabela 4.7: Comparação de desempenho ($PRD \times CR$) entre diferentes esquemas decompressão de ECG.

Algoritmo	**Registro**	**CR**	**PRD**
Lee et. al [5]	100	24 : 1	8.10
Chou et. al app. 2 [3]	100	24 : 1	4.06
Norm - JPEG2000	100	24 : 1	6.01
Tipo 1 - JPEG2000	100	24 : 1	4.10
Tipo 2 - JPEG2000	100	24 : 1	5.16
Tipo 3 - JPEG2000	100	24 : 1	3.95
Tipo 1 - H.264	100	24 : 1	3.47
Tipo 3 - JPEG2000	100	10 : 1	2.12
Tipo 1 - H.264	100	10 : 1	2.08
Tipo 3 - JPEG2000	117	24 : 1	1.72
Tipo 1 - H.264	117	24 : 1	1.64
Chou et. al app. 2 [3]	117	13 : 1	1.18
Tipo 3 - JPEG2000	117	13 : 1	1.07
Tipo 1 - H.264	117	13 : 1	1.14
Wei et. al [6]	117	10 : 1	1.18
Bilgin et. al [4]	117	10 : 1	1.03
Chou et. al app. 1 [3]	117	10 : 1	0.98
Tipo 3 - JPEG2000	117	10 : 1	0.86

4.2.6 Técnicas de pré-procesamento aliadas ao MMP

Como foi mostrado nas seções 4.2.3 e 4.2.4, o MMP pode ser aplicado à compressão de sinais de eletrocardiograma, apresentando um bom desempenho. Os padrões recorrentes multiescalas são bastante adequados aos sinais de ECG, principalmente devido ao fato destes serem compostos por um conjunto de sub-ondas que

Tabela 4.8: Comparação de desempenho ($PRD \times CR$) entre diferentes esquemas decompressão de ECG (cont.).

Algoritmo	**Registro**	**CR**	**PRD**
Tipo 1 - H.264	117	10 : 1	0.95
Bilgin et. al [4]	117	8 : 1	0.86
Tipo 3 - JPEG2000	117	8 : 1	0.75
Tipo 1 - H.264	117	8 : 1	0.81
Bilgin et. al [4]	119	21.6 : 1	3.76
Tai et. al [7]	119	20 : 1	2.17
Chou et. al app. 2 [3]	119	20.9 : 1	1.81
Norm - JPEG2000	119	20.9 : 1	3.90
Tipo 1 - JPEG2000	119	20.9 : 1	3.77
Tipo 2 - JPEG2000	119	20.9 : 1	2.90
Tipo 3 - JPEG2000	119	20.9 : 1	1.92
Tipo 3 - H.264	119	20.9 : 1	1.78
Chou et. al app. 2 [3]	119	10 : 1	1.03
Norm - JPEG2000 [3]	119	10 : 1	1.37
Tipo 3 - JPEG2000	119	10 : 1	0.93
Tipo 3 - H.264	119	10 : 1	1.00
Tipo 3 - JPEG2000	119	8 : 1	0.74
Tipo 3 - H.264	119	8 : 1	0.83

se repetem durante todo o exame.

Apesar disso, o traçado de ECG pode sofrer alterações ao longo do exame, seja devido a mudanças nas condições do paciente ou devido a alguma desordem, que pode ser então diagnosticada. Como o ECG depende da repetição das sub-ondas para uma adaptabilidade ótima do seu dicionário, tais variações podem comprometer o desempenho do algoritmo.

Na seções 4.2.4 e 4.2.5, mostrou-se que um simples pré-processamento antes da codificação pode promover um aumento do desempenho, pois apresenta-se ao compressor uma estrutura mais simples e homogênea para a codificação. Sendo assim, pode-se incorporar uma etapa completa de pré-processamento ao MMP, composta por várias técnicas de equalização/normalização, gerando um sinal capaz de ser comprimido com grande eficácia.

Nesta seção, incorporam-se as técnicas deselvolvidas na seção 4.2.5 ao que já tinha sido apresentado nas seções 4.2.3 e 4.2.4, gerando-se um esquema completo e abrangente para a compressão de sinais de ECG. A etapa de pré-processamento precede o MMP, reformatando o sinal original e revelando sua estrutura inerente.

Técnicas incorporadas ao algoritmo MMP

A etapa de pré-processamento agregada ao algoritmo MMP é composta por cinco técnicas de pré-processamento, cada uma com o objetivo de corrigir uma dada característica do sinal: detecção de período, normalização de período, equalização DC (EqDC), ordenação por similaridade (*Similarity Sorting* - SS) e redução de faixa dinâmica, aplicadas nessa mesma ordem. A primeira, a segunda e a quinta técnicas já foram abordadas na seção 4.2.4. A terceira e a quarta técnicas também já foram abordadas, só que na seção 4.2.5. Entretanto, a ordenação por similaridade aqui utilizada para o MMP difere um pouco da ordenação por complexidade apresentada na seção 4.2.5.

A ordenação por complexidade, por ser baseada na semelhança com o primeiro segmento, acaba criando regiões razoavelmente uniformes na imagem de ECG, sendo bastante adequada a algoritmos geralmente empregados em compressores de imagens. Entretanto, dado que o MMP é um esquema unidimensional e o dicionário final de cada bloco de entrada $\mathbf{X}^j$ é o inicial do próximo, necessita-se de uma técnica

que realmente faça com que períodos muito semelhantes sejam adjacentes. Para que isso aconteça, criou-se a ordenação por similaridade, que será apresentada no próximo parágrafo.

Para que a ordenação por similaridade seja executada, primeiramente as variâncias de todos os períodos são computadas e o segmento com o menor valor é colocado como o primeiro batimento, ou seja, o primeiro período de ECG. Os períodos seguintes são então posicionados em ordem decrescente de similaridade com o anterior (e não com o primeiro, como na ordenação por complexidade); verificou-se que um bom critério de similaridade com relação ao m-ésimo período é novamente o erro médio quadrático, dado por

$$MSE_k = \frac{1}{L_N} \sum_{i=1}^{L_N} (x_k(i) - x_{m-1}(i))^2, \tag{4.51}$$

onde $x_{m-1}(i)$ são as amostras do $(m-1)$-ésimo periodo reordenado e $x_k(i)$ são as amostras do k-ésimo perído de ECG.

O efeito desta técnica de pré-processamento num sinal que sofreu equalização DC da Figura 4.38(c) é mostrado na Figura 4.38(d). É possível perceber que agora períodos adjacentes são muito similares e os padrões aprendidos pelo MMP a cada período podem ser facilmente utilizados para a codificação dos subseqüentes. Além disso, o envelope de sinal dos períodos de ECG varia suavemente ao longo do exame (sua variância aumenta com o decorrer da codificação), permitindo ao MMP aprender rapidamente o comportamento atual. A Figura 4.38(e) mostra o sinal com faixa dinâmica reduzida (ver as equações (4.45) e (4.46)).

Pode-se questionar que o *período normalizado médio* apresentado na seção 4.2.4 é capaz de reduzir a faixa dinâmica do sinal apenas se o mesmo for aproximadamente periódico. Realmente, se o sinal varia muito, a redução de faixa dinâmica é prejudicada. Entretanto, a normalização de período em conjunto com a equalização DC tendem a melhorar muito a correlação entre períodos, proporcionando bons resultados mesmo para sinais irregulares.

O processo de decodificação

A decodificação do sinal comprimido é realizada através do processamento do feixe de bits gerado pelo MMP, que compreende a recuperação da árvore de seg-

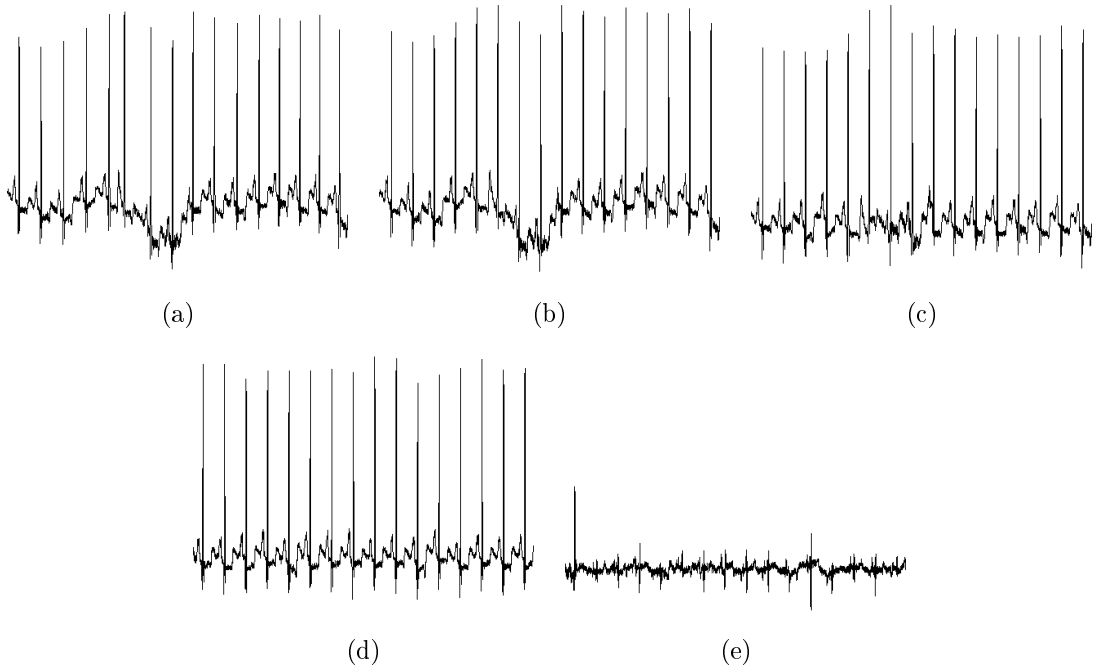


Figura 4.38: Efeitos das técnicas de pré-processamento no registro 100 da base de dados de ECGs MIT/BIH. (a) Períodos originais. (b) Períodos normalizados. (c) Equalização DC aplicada a (b). (d) Ordenação por similaridade aplicada a (c). (e) Sinal com faixa dinâmica reduzida.

mentação $\mathcal{S}$, com o *flags* binários, e a designação de um elemento de dicionário para cada índice i_j. Enquanto a decodificação é realizada, concatenações dos blocos escolhidos são adicionadas ao dicionário, assim como ocorreu durante a codificação. O sinal é descompactado, regenera-se a faixa dinâmica original, a ordem e o nível DC originais são então recuperados, nesta mesma ordem, e cada segmento é escalonado para o seu comprimento incial, utilizando-se a informação adicionada ao cabeçalho do arquivo, o que resulta no sinal reconstruído.

Resultados de simulações

A eficácia do esquema proposto foi verificada através de testes com as primeiras 216000 amostras (10 minutos) dos registros 100, 102, 107, 115, 117, e 119 da base de dados de ECGs MIT/BIH. Novamente, esses registros foram escolhidos por existirem muitos resultados na literatura informados para os mesmos [3–7]. Os sinais foram amostrados a 360 Hz com 11 bits de resolução. O comprimento original de cada período, os níveis DC originais e o nível de equalização foram codificados (com um codificador aritmético) e enviados como informação auxiliar ao decodificador.

A ordenação original foi também codificada, mas com um codificador aritmético de contexto adaptativo, que registra as ocorrências de índices, colocando-se a sua probabilidade em zero. Cada sinal de entrada é particionado em segmentos de 64 amostras (ver seções 2.1 e 4.2.3), que são seqüencialmente processados pelo algoritmo. A qualidade do sinal reconstruído é avaliada através da métrica de PRD e da taxa de compressão CR, já apresentadas na seção 4.2.2

Os resultados estão sumarizados nas Tabelas 4.9, 4.10 e 4.11. O algoritmo proposto superou todos os outros métodos para os registros 100, 102, 107, 115, e 117, e apresentou resultados comparáveis para o registro 119. As comparações com o algoritmo proposto em [7] levaram em consideração o tamanho de bloco sugerido pelo próprio autor (de dimensões 32×256) [7]. Embora os métodos testados apresentem resultados próximos em baixas taxas de compressão, o método proposto é bastante superior em altas taxas, mantendo um valor de PRD razoável (que não compromete o processo de diagnóstico), o que é o principal objetivo de um esquema de compressão de ECG. Uma análise preliminar dos registros reconstruídos, similar à realizada na seção 4.2.3, mostrou que a informação para diagnóstico também é mantida neste caso (eqDC + SS), gerando distorções ainda menores nas estruturas. É possível perceber, das Tabelas 4.9, 4.10 e 4.11, que a ordenação por similaridade (SS) não é sempre utilizada. Testes mostraram que esta técnica só é vantajosa quando o desvio padrão dos comprimentos dos períodos detectados for maior que 8%, podendo ser retirada se isto não for verdade. Devido ao seu *overhead* (transmissão da ordenação original), a ordenação por similaridade pode causar uma redução no PRD de aproximadamente 0,05 a 0,1%. Vale ressaltar que o algoritmo em [7] utiliza um método para a detecção do complexo QRS muito mais avançado e que o desempenho do mesmo varia bastante com o tamanho do bloco.

É importante notar que a complexidade computacional associada às técnicas propostas é baixa. A equalização DC é composta por uma divisão por período e somas (ver as equações (4.47) e (4.48)). A complexidade do procedimento de ordenação é maior e também muito influenciada pelas operações descritas pela equação (4.49), que são executadas para os m períodos restantes, a cada passo. Entretanto, esses cálculos podem ser bastante acelerados através do uso de tabelas de *Lookup*, mantendo-se a complexidade global baixa; a decodificação é mais simples e consiste

Tabela 4.9: Comparação de desempenho ($PRD \times CR$) entre os algoritmos em [3–7] e o MMP com técnicas de pré-processamento para a compressão de ECG.

Algoritmo	**Registro**	**CR**	**PRD**
Chou et. al app. 2 [3]	100	24 : 1	4.06
Tipo 3 - JPEG2000 seção 4.2.5	100	24 : 1	3.95
Tipo 1 - H.264 seção 4.2.5	100	24 : 1	3.47
MMP seção 4.2.4	100	24 : 1	3.41
MMP EqDC	100	24 : 1	3.30
Tipo 3 - JPEG2000 seção 4.2.5	100	10 : 1	2.12
Tipo 1 - H.264 seção 4.2.5	100	10 : 1	2.08
MMP seção 4.2.4	100	10 : 1	2.10
MMP EqDC	100	10 : 1	2.03
MMP seção 4.2.4	102	22.49 : 1	3.02
MMP EqDC/SS	102	22.49 : 1	2.49
Tai et. al [7]	107	10.7 : 1	1.90
MMP seção 4.2.4	107	10.7 : 1	1.70
MMP EqDC	107	10.7 : 1	1.61
Tai et. al [7]	115	30.6 : 1	4.10
MMP seção 4.2.4	115	30.6 : 1	3.20
MMP EqDC/SS	115	30.6 : 1	2.92
Tipo 3 - JPEG2000 seção 4.2.5	117	24 : 1	1.72
Tipo 1 - H.264 seção 4.2.5	117	24 : 1	1.64

Tabela 4.10: Comparação de desempenho ($PRD \times CR$) entre os algoritmos em [3–7] e o MMP com técnicas de pré-processamento para a compressão de ECG (cont.).

Algoritmo	**Registro**	**CR**	**PRD**
MMP seção 4.2.4	117	24 : 1	1.42
MMP EqDC	117	24 : 1	1.26
Chou et. al app. 2 [3]	117	13 : 1	1.18
Tipo 3 - JPEG2000 seção 4.2.5	117	13 : 1	1.07
Tipo 1 - H.264 seção 4.2.5	117	13 : 1	1.14
MMP seção 4.2.4	117	13 : 1	0.98
MMP EqDC	117	13 : 1	0.91
Wei et. al [6]	117	10 : 1	1.18
Bilgin et. al [4]	117	10 : 1	1.03
Chou et. al app. 1 [3]	117	10 : 1	0.98
Tipo 3 - JPEG2000 seção 4.2.5	117	10 : 1	0.86
Tipo 1 - H.264 seção 4.2.5	117	10 : 1	0.95
MMP seção 4.2.4	117	10 : 1	0.85
MMP EqDC	117	10 : 1	0.79
Bilgin et. al [4]	117	8 : 1	0.86
Tipo 3 - JPEG2000 seção 4.2.5	117	8 : 1	0.75
Tipo 1 - H.264 seção 4.2.5	117	8 : 1	0.81
MMP seção 4.2.4	117	8 : 1	0.75
MMP EqDC	117	8 : 1	0.72

Tabela 4.11: Comparação de desempenho ($PRD \times CR$) entre os algoritmos em [3–7] e o MMP com técnicas de pré-processamento para a compressão de ECG (cont.).

Algoritmo	**Registro**	**CR**	**PRD**
Lee et. al [5]	119	24 : 1	10.5
Bilgin et. al [4]	119	21.6 : 1	3.76
Tai et. al [7]	119	20 : 1	2.17
Chou et. al app. 2 [3]	119	20.9 : 1	1.81
Tipo 3 - JPEG2000 seção 4.2.5	119	20.9 : 1	1.92
Tipo 3 - H.264 seção 4.2.5	119	20.9 : 1	1.78
MMP seção 4.2.4	119	20.9 : 1	2.00
MMP EqDC/SS	119	20.9 : 1	1.83
Chou et. al app. 2 [3]	119	10 : 1	1.03
Norm. - JPEG2000	119	10 : 1	1.37
Tipo 3 - JPEG2000 seção 4.2.5	119	10 : 1	0.93
Tipo 3 - H.264 seção 4.2.5	119	10 : 1	1.00
MMP seção 4.2.4	119	10 : 1	1.10
MMP EqDC/SS	119	10 : 1	1.07
Tipo 3 - JPEG2000 seção 4.2.5	119	8 : 1	0.74
Tipo 3 - H.264 seção 4.2.5	119	8 : 1	0.83
MMP seção 4.2.4	119	8 : 1	0.96
MMP EqDC/SS	119	8 : 1	0.91

em apenas atribuir a posição e o nível originais a cada elemento.

4.2.7 O MMP aplicado à compressão de sinais de EMG

O eletromiograma (EMG) consiste em um sinal elétrico representando a contração de músculos do corpo humano [116]. Tal sinal pode ser adquirido de basicamente duas maneiras: com eletrodos superficiais ou eletrodos de agulha [117]. Os sinais adquiridos com eletrodos superficiais, o principal objetivo da presente aplicação, são similares a ruído, enquanto sinais de eletrodos de agulha apresentam alguma peridiocidade, como os sinais de ECG abordados nas seções anteriores. Um sinal de EMG típico é mostrado na Figura 4.39.

O sinal de EMG superficial [118] apresenta uma aparência similar a um ruído, devido ao fato de ser composto pela superposição de sinais oriundos de muitos grupos de fibras musculares próximas ao sensor. Sendo assim, pode ser interessante aplicar uma decomposição adequada ao mesmo [119–121], capaz de identificar os potenciais de ação produzidos por cada grupo. Esse tipo de processamento, por exemplo, pode ser executado no contexto de estudos sobre fenômenos musculares ou do desenvolvimento de sistemas de auxílio ao movimento [122].

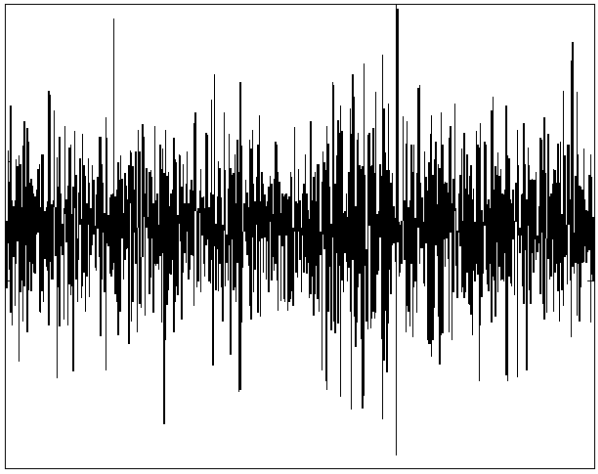


Figura 4.39: Um sinal de EMG adquirido com eletrodos superficiais.

Com o avanço da telemedicina, a necessidade da transmissão de sinais de EMG aumentou. Além disso, o armazenamento de tais sinais também é importante, possibilitando uma análise comparativa de comportamento, análise do desenvolvimento de doenças e diagnóstico (distrofia, danos em nervos periféricos, fraqueza e etc.).

Esse tipo de sinal apresenta larguras de banda de até 600 Hz e pode ser necessário analisar vários canais, durante longos períodos de tempo [123]. Essas situações necessitam de bons métodos de compressão, capazes de representar tais sinais de forma compacta, mas ainda preservando toda a informação relevante para análise e diagnóstico.

Nesta seção, o algoritmo MMP será aplicado à compressão de sinais de EMG superficiais, comprovando o seu comportamento universal e colocando o MMP como um bom candidato para a codificação de sinais biológicos. Em [10], a utilização do MMP na codificação de dados multidimensionais foi justificada pelo seu desempenho com vetores gaussianos. Sabe também que sinais de EMG apresentam um comportamento aproximadamente gaussiano [124], dependendo do nível de compressão voluntária máxima (*Maximum Voluntary Compression* - MVC). Sendo assim, é razoável utilizar o MMP na compressão de sinais de EMG, com a possibilidade de bons desempenhos.

O sinal de EMG

Os músculos são compostos por fibras delgadas organizadas em grupos, que são chamados de unidades motoras [125]. Essas fibras são enervadas por um único neurônio motor, fazendo com que as mesmas reajam conjuntamente durante as contrações musculares. A unidade motora é ativada por impulsos elétricos ao longo do neurônio motor, enviados pelo sistema nervoso. Quando esses impulsos chegam a uma taxa rápida o suficiente, uma força constante é produzida. Cada impulso nervoso induz uma descarga elétrica em cada uma das fibras musculares, que se espalha ao longo das mesmas e produz um potencial elétrico. O sinal de EMG é a composição das descargas de todas as unidades motoras que o eletrodo pode detectar e representa a atividade elétrica dos músculos durante as contrações [125].

O sinal de EMG pode ser usado para a detecção de atividade elétrica muscular anormal devido a muitas desordens, como distrofia muscular, inflamação ou fraqueza, miastenia grave, danos no nervo periférico e hérnias de disco, entre outras. Há dois tipos de sinais de EMG: intramuscular e superficial. O primeiro é obtido com eletrodos de agulha inseridos nos músculos e o segundo envolve a colocação de eletrodos na pele logo acima dos músculos, consistindo numa abordagem não inva-

siva. Embora o EMG intramuscular tenha sido muito utilizado no passado, devido à boa qualidade do sinal obtido, o interesse em EMGs superficiais aumentou muito nos últimos anos, principalmente devido à ausência de danos à pele, o que o torna importante no monitoramento da progressão de desordens em nervos e músculos.

A forma de onda nos eletrodos superficiais apresenta uma aparência semelhante a um ruído. De fato, sabe-se que o sinal de EMG gravado durante um esforço constante, com ângulo constante e contrações não fatigantes, pode ser modelado como um processo estocástico de distribuição gaussiana e com média zero, dependendo do nível de MVC ($> 30\%$) [124]. Essa é uma característica muito importante, pois pode ajudar a melhorar o algoritmo de codificação e permite suposições iniciais sobre o sinal de entrada.

Resultados de simulações

O MMP é um codificador universal e foi mostrado em [10] que o mesmo pode apresentar um bom desempenho na compressão de sinais gaussianos, principalmente devido à abordagem baseada em casamento de padrões multiescalas. Dadas essas constatações, o MMP é um bom candidato para a compressão de sinais de EMG. Nesta seção, serão apresentados resultados para tal aplicação.

O MMP utilizado na compressão de sinais de EMG será o apresentado na seção 4.2.3, sem as técnicas de pré-processamento, devido à falta de estruturas e periodicidade. A eficácia do esquema será verificada através de testes com sinais de EMG coletados do *biceps brachii* de 13 pacientes, com eletrodos pré-amplificados e durante contrações isométricas, enquanto os mesmos estão sentados com o antebraço paralelo ao torso e sustentando 60% de MVC. O sinal de EMG resultante foi então amostrado a 2000 Hz e quantizado com 12 bits. A duração dos sinais varia de 1,3 a 3,0 minutos. Cada sinal de entrada é então particionado em segmentos de 64 amostras, que são seqüencialmente processadas pelo algoritmo. É importante notar que, dado o MVC empreendido, o sinal de saída apresenta um comportamento aproximadamente gaussiano. A qualidade dos sinais reconstruídos foi avaliada através do uso da métrica de PRD, apresentada na seção 4.2.3, e do fator de compressão (*Compression Factor* - CF), avaliado como

$$CF = \frac{B_o - B_c}{B_o} \times 100\%, \tag{4.52}$$

onde B_o é o número total de bits do sinal original e B_c é o número total de bits do sinal comprimido, incluindo o cabeçalho. Os resultados estão sumarizados na Figura 4.40. O algoritmo MMP foi aplicado a cada um dos 13 sinais de EMG isométricos, levando a fatores de compressão variando de 50% a 95%. A Figura 4.40(a) mostra curvas de CF $\times$ PRD para todos os sinais, juntamente com uma curva média, calculada da mesma maneira que em 4.2.3. A Figura 4.40(b) ilustra uma comparação com os algoritmos em [123,126]. O primeiro é baseado em *wavelets* e num esquema de alocação dinâmica de bits, que utiliza a *camada de Kohonen* (um tipo de rede neural artificial). O segundo é baseado no EZW e provou ser superior a outros esquemas baseados em *wavelets*. Vale ressaltar que os resultados em [123] e [126] representam o estado da arte – Guerrero et. al [117] mostrou que métodos de compressão baseados em transformadas tendem a apresentar bons resultados, sendo que os melhores são proporcionados pelos baseados em *wavelets*. Os sinais utilizados nesta seção são muito similares aos utilizados em [123], no qual uma comparação com os resultados em [126] é feita. Na verdade, os autores de [123] gentilmente cederam estes sinais de teste. O algoritmo MMP superou ambos os métodos testados para sinais de contração isométrica. Embora os outros métodos testados tendam a se aproximar em altas taxas de compressão, o MMP é significativamente superior para a mesma faixa. Em CFs abaixo de 84%, o método proposto mantém um PRD razoável, sendo esse limite, em geral, suficiente para não haver comprometimento do diagnóstico ou ainda permitir o seu uso, o que é necessário em compressão de sinais biológicos. Na Figura 4.41, pode-se ver uma porção de um sinal de EMG original e sua versão reconstruída com o MMP a um CF de 83,68%, assim como a diferença entre os mesmos.

O desempenho do algoritmo MMP também pode ser comparado com o apresentado em [127], no qual sinais de EMG foram quantizados com 12 bits e amostrados a 2.048 Hz. Entretanto, os sinais gravados foram filtrados com um filtro passa-banda sintonizado na faixa de 10-400 Hz e sub-amostrados para 1.024 Hz antes de serem comprimidos. Embora os sinais utilizados durante os testes tenham sido amostrados a uma taxa de 2.000 Hz e apresentem maiores larguras de banda, uma comparação grosseira é possível. Em [127], os autores reportaram um PRD de 5,95% para um CF de 87,3% e uma MVC de 50% e um PRD de 5,26% para um CF de 87,3% e

uma MVC de 70%, resultando em uma média de 5,60%. O algoritmo MMP proporciona um PRD de 5,30% para as mesmas condições. Isto é uma indicação de que o MMP é comparável ao apresentado em [127]. Testes também foram realizados para verificar se o método proposto preserva as principais características espectrais dos sinais de EMG, representadas pelas freqüência média, freqüência mediana, variância e obliqüidade, calculadas respectivamente como

$$f_{média} = \frac{\sum_{i=0}^{N-1} f_i.P[f_i].(f_i - f_{i-1})}{\sum_{i=0}^{N-1} P[f_i].(f_i - f_{i-1})}, \tag{4.53}$$

$$\sum_{i=0}^{f_{mediana}} P[f_i].(f_i - f_{i-1}) = \sum_{i=f_{mediana}}^{N-1} P[f_i].(f_i - f_{i-1}) = \frac{1}{2}\sum_{i=0}^{N-1} P[f_i].(f_i - f_{i-1}), \tag{4.54}$$

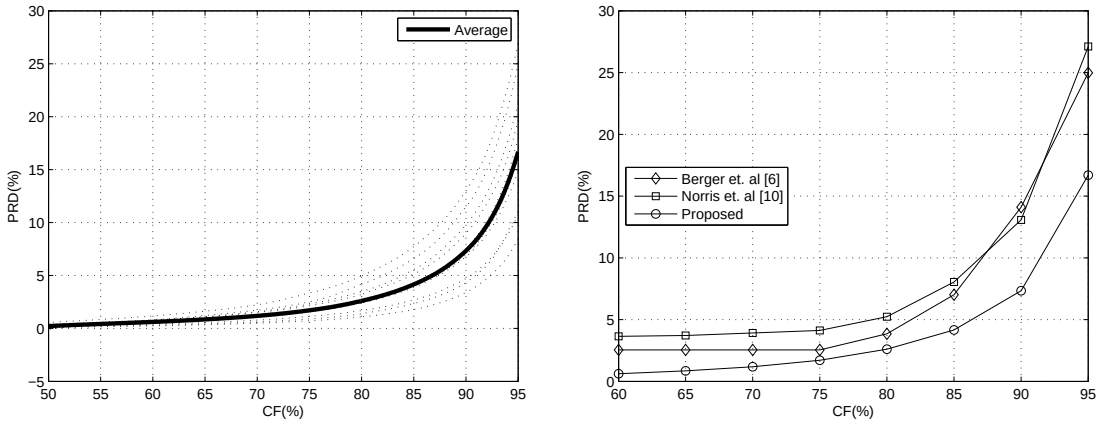
$$VAR = \sum_{i=0}^{N-1} (f_i - f_{média})^2.P[f_i].(f_i - f_{i-1}) \tag{4.55}$$

e

$$OBLI = \frac{\sum_{i=0}^{N-1} (f_i - f_{média})^3.P[f_i].(f_i - f_{i-1})}{\left(\sum_{i=0}^{N-1} (f_i - f_{média})^2.P[f_i].(f_i - f_{i-1})\right)^{3/2}}, \tag{4.56}$$

como feito em [127, 128]. Nestas equações, $P[f]$ é a densidade espectral de potência (*Power Spectral Density* - PSD) para a freqüência f e N é o número de freqüências para as quais há valores de espectro de potência. A PSD foi estimada através do método de periodograma [129], dividindo-se o sinal em *epochs* (janelas de tempo) de 1 segundo. Algumas características espectrais podem ser utilizadas diretamente na detecção de anomalias: por exemplo, variações em $f_{média}$ e $f_{mediana}$ podem indicar mudanças na velocidade de condução da fibra muscular [130] ou mesmo fadiga [131]. Ao se realizarem os cálculos de tais características, notou-se que alguns registros apresentavam valores estranhos, completamente diferentes dos originais. Uma análise mais detida destes sinais revelou que há apenas ruído no início e no final dos mesmos, com amplitudes baixas. Isto resulta em amostras reconstruídas com valor zero e conseqüentemente saídas de periodograma nulas. Este tipo de comportamento compromete o cálculo de tais características espectrais, como pode ser verificado. Dado isso, as partes inicial e final de alguns registros não foram consideradas durante os cálculos. A variação relativa (percentual) nessas variáveis, para um CF de aproximadamente 87,3%, foi de 1,6201 $\pm$ 1,1184 para freqüência

média, $0{,}9849 \pm 1{,}2462$ para a freqüência mediana, $9{,}1189 \pm 5{,}9106$ para a variância, e $56{,}2935 \pm 21{,}3781$ para a obliqüidade (apresentadas na forma *percentagem média* $\pm$ *desvio*). Os resultados para as freqüências média e mediana estão abaixo de 10% e são comparáveis aos resultados em [127]; entretanto, os momentos não são bem preservados e apresentam muita variação. A razão para este comportamento deve-se ao fato do MMP trabalhar bem no domínio do tempo, utilizando o erro quadrático médio como métrica de distorção; isso favorece a preservação do envelope do sinal (o que pode ser inferido dos bons resultados em termos de PRD), mas não das características espectrais.



(a) (b)

Figura 4.40: Resultados de simulações: $PRD \times CF$. (a) Desempenho para os 13 sinais de teste de contração isométrica. (b) Comparação com algoritmos baseados em *wavelets*.

4.2.8 Análise dos resultados dos algoritmos de compressão baseados no MMP

As seções anteriores apresentaram vários algoritmos de compressão baseados no MMP, utilizados para a compressão de imagens e sinais de ECG e EMG. A grande vantagem do MMP, em comparação com outros algoritmos de codificação, reside no fato do dicionário ser adaptativo, o que permite ao algoritmo ser aplicado a um dado tipo de informação sem a necessidade de treinamento dos padrões para codificação. Basicamente, para uma compressão eficiente, basta incorporar o conhecimento sobre a estrutura do sinal ao algoritmo base, permitindo ao MMP explorar de forma eficaz

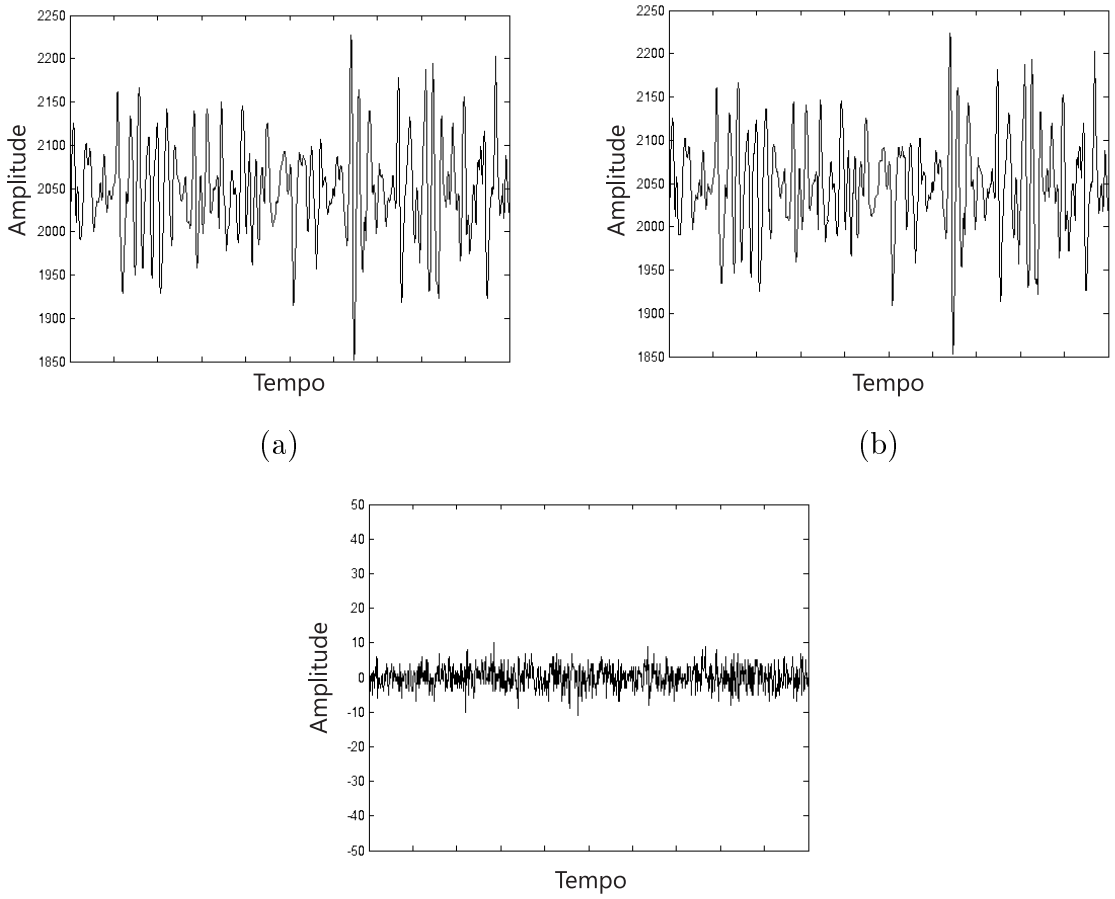


Figura 4.41: Efeito do algoritmo MMP nosinal de EMG. (a) Original. (b) Reconstruído a um CF de 83,68%, com 3,51% de PRD. (c) Sinal diferença.

a correlação existente, e/ou estender o algoritmo base com técnicas específicas aos dados a serem codificados.

Apesar dos sinais em questão serem bastante diferentes, é fácil perceber que as ferramentas de codificação são bastante semelhantes, podendo-se ressaltar que:

i) O MMP consegue explorar com grande eficácia a informação existente na vizinhança causal, orientando a escolha dos elementos para a codificação e a formação do dicionário. Esta característica pode ser observada com a introdução do critério de continuidade para a compressão tanto de imagens quanto de sinais de ECG e pode ser explorada de diversas formas. Nos exemplos apresentados, foi possível definir um subconjunto dos dados com uma característica específica, atendendo a um modelo previamente estabelecido ou variável, como no caso dos contextos na seção 4.1.1;

ii) O dicionário do MMP é construído com o próprio dado de entrada, o que o torna bastante correlacionado com o mesmo, ou seja, produz elementos bastante semelhantes. Como os blocos no dicionário resultam de codificações anteriores, os mesmos podem ser utilizados no futuro para codificar conjuntos de dados com as mesmas características dos primeiros;

iii) O dicionário $\mathcal{D}$ pode ser particionado, ou seja, subdividido em conjuntos disjuntos através de um dado critério válido para os dados em compressão. Tal processamento constitui uma ferramenta bastante eficiente na codificação de sinais em geral, pois permite que sejam formados dicionários com características comuns, facilitando a escolha feita pelo algoritmo. Uma partição conveniente, por exemplo, pode gerar sub-dicionários com blocos que tenham uma dada rugosidade, como nas seções 4.1.1 e 4.1.2, ou ainda as partições especificadas na equação (4.16). Obviamente, também é possível gerar conjuntos com uma dada distância euclidiana entre os seus membros, restringindo o erro resultante de uma aproximação, por exemplo;

iv) Os dados podem ser transformados e proporcionar uma estrutura que seja facilmente aprendida pelo MMP, aumentando a sua adaptabilidade. Isso pode ser observado nas transformações utilizadas pelos esquemas para a compressão de sinais de ECG (seções 4.2.4 e 4.2.6), que se utilizam da equalização dos períodos do batimento cardíaco para que o MMP aprenda mais rapidamente os padrões de compressão;

v) Extensões específicas ao sinal de entrada podem ser facilmente incorporadas ao algoritmo base, aumentado a adaptabilidade do MMP e resultando em melhores desempenhos durante a codificação. Com isso, um *toolbox* de técnicas poderá ser criado, sendo que, durante a codificação de um dado sinal, um subconjunto das ferramentas mais adequado pode ser escolhido;

vi) O MMP é eficaz tanto na codificação de dados brutos, sem qualquer processamento, quanto dados transformados, resíduos ou mesmo distribuições bem definidas coma as gaussianas (e até mesmo laplacianas), o que foi comprovado com os experimentos envolvendo sinais de ECG. Isto reforça a sua característica de codificador universal, abrindo espaço para o desenvolvimento de codificadores

de uso genérico, que podem ser aplicados, por exemplo, tanto a imagens e áudio quanto a sinais biológicos;

vii) O MMP possui características de transformação, quantização e codificação de entropia, que permitem a codificação dos dados em um único passo. Se devidamente ajustado, cada uma dessas características pode ser utilizada para a abordagem de um dado problema durante a codificação. Por exemplo, para reforçar a utilização de seqüências típicas, pode-se recorrer ao dicionário de vizinhança da seção 4.1.2;

viii) Ao se proporcionar um maior número de possibilidades de segmentação, o desempenho do MMP acaba aumentando, pois os elementos de codificação tendem a se adaptar aos contornos do sinal, como na nova estratégia de otimização sugerida na seção 4.2.3;

ix) A árvore de segmentação pode ser considerada como a "assinatura" de um bloco de entrada e reflete a existência de elementos de dicionário capazes de codificá-lo. Se o dicionário for controlado, um outro bloco similar codificado num momento posterior apresentará uma árvore bastante correlacionada. Esta constatação pode ser utilizada, no segundo bloco, para se enviar apenas a parte diferente da nova árvore.

Com os resultados apresentados para imagens, percebe-se que o MMP é bastante eficaz na codificação de cenas muito complexas (com muitas bordas e transições), não sofrendo da falta de capacidade de localização de componentes de alta freqüência como as *wavelets*, ou muito simples, sendo esta última característica devido à rápida adaptação do dicionário. Entretanto, para cenas com conteúdo moderado (transições suaves), como as existentes na imagem *Lena*, o dicionário não encontra um ponto de aglomeração dos elementos, sendo obrigado a aprender muitos padrões diferentes. Uma possível solução para esse problema pode residir na criação de um gradiente de atualização, indicando a direção de atualização preferencial, o que controlaria os novos elementos a serem inseridos no dicionário.

No que diz respeito aos sinais de ECG, o MMP ainda parece necessitar de uma maior capacidade de exploração das redundâncias intra-batimento (dentro de um mesmo período). Assim como existe agora um critério baseado na vizinhança do

bloco, poderia haver também um critério que atingisse as amostras dentro de um mesmo bloco. Tal desenvolvimento daria maior flexibilidade ao algoritmo e permitiria uma maior adaptabilidade ao sinal em codificação.

A aplicação para a compressão de sinais de EMG apenas começou e ainda há a necessidade de desenvolvimento de técnicas de pré-processamento específicas para este tipo de sinal. Apesar de não haver periodicidade clara, como nos sinais de ECG, o sinal de EMG pode ser decomposto, criando-se âncoras para o desenvolvimento de uma dada técnica, através de pontos de referência no sinal (*e.g.* complexo QRS). Além disso, procedimentos de normalização do sinal de entrada, supondo o seu comportamento aproximadamente gaussiano, seriam úteis. Por exemplo, poder-se-ia realizar um simples processamento para a obtenção de um sinal com média zero e desvio padrão unitário.

Há ainda a possibilidade de se pesquisar outras características do MMP. Por exemplo, as partições de dicionário exploram a entropia relacionada à estrutura do dicionário, reduzindo a taxa de codificação de um dado elemento. Sendo assim, a própria quantização do MMP poderia ser também explorada, incluindo no dicionário elementos com uma certa distribuição e passo dos intervalos de quantização.

O procedimento de particionamento do MMP, responsável pela criação da árvore de segmentação, ainda é bastante restrito. Apesar da incorporação do procedimento de união, ainda não há uma maneira sistemática de se obter partições completamente arbitrárias, como elementos de dimensão 3 para sinais unidimensionais e dimensões 16×1 para imagens, por exemplo. Essa característica permitiria ao MMP criar elementos completamente casados com as estruturas presentes no sinal, reduzindo a visibilidade das bordas (blocagem) e ajudando na própria adaptação do dicionário.

Como comentário final, deve-se ressaltar como característica mais importante do MMP o seu comportamento universal. Os resultados apresentados posicionam o MMP como um esquema de codificação versátil, capaz de ser aplicado a vários conjuntos de dados distintos. Por exemplo, poder-se-ia desenvolver um esquema, baseado no mesmo, para a compressão dos três sinais biológicos mais pesquisados: eletrocardiograma (ECG), eletromiograma (EMG) e eletroencefalograma (EEG), tendo como aplicação direta um sistema de monitoramento de pacientes.

Capítulo 5

Códigos para aplicação em compressão distribuída

Como já foi dito, códigos de canal de alto desempenho, como códigos Turbo e LDPC, são bons candidatos a códigos de Slepian-Wolf, sendo que a maioria dos codificadores mais recentes é baseada nos mesmos. Com isso, é necessária uma introdução ao seu conceito e funcionamento. As próximas seções explorarão a codificação/decodificação de tais códigos e suas principais características.

5.1 Os códigos turbo

Os códigos turbo [76, 132] foram apresentados à comunidade científica em [76] e representam um importante marco na codificação de canal, juntamente com a redescoberta dos códigos LDPC. Seu nome vem da semelhança com o princípio de funcionamento da turbina, o que pode ser evidenciado apenas no decodificador. A idéia-chave do algoritmo de decodificação turbo consiste na exploração da troca sucessiva de informação extrínseca entre diferentes blocos de processamento em um receptor de dados, de maneira similar à troca entre turbina e compressor num motor a jato [133]. Em resumo, os códigos turbo são baseados em conceitos já conhecidos no mundo científico, combinados com algumas abordagens inovadoras.

Um codificador turbo genérico está representado na Figura 5.1 e consiste em dois codificadores convolucionais de taxa 1/2 conectados por um entrelaçador e um perfurador; a taxa global é de 1/3 (só os bits de paridade são utilizados nos

codificadores componentes) e os codificadores estão organizados de forma paralela.

Os bits sistemáticos x^s são enviados diretamente para o primeiro codificador e o segundo recebe uma versão embaralhada $x^{s'}$ (entrelaçada) dos mesmos. Os codificadores realizam convoluções módulo-2 e disponibilizam, na saída, um subconjunto (devido ao perfurador) dos bits de paridade x^{1p} e x^{2p}.

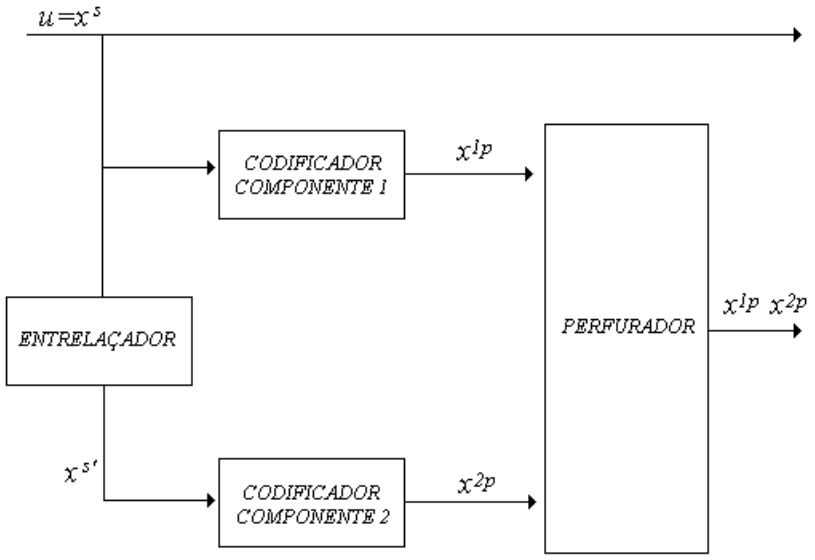


Figura 5.1: Codificador turbo genérico.

Os codificadores componentes são recursivos e sistemáticos e configuram como os grandes responsáveis pelo bom desempenho do codificador turbo. A matriz geradora de um código convolucional sistemático é dada por

$$G(D) = \left[\begin{array}{cc} 1 & \frac{g_2(D)}{g_1(D)} \end{array} \right]. \tag{5.1}$$

Se a seqüência de entrada for divisível por $g_1(D)$, o peso de *Hamming* da seqüência de saída será finito; caso contrário, o peso de será infinito [132]. Esta constatação, que é a justificativa para utilização de codificadores recursivos, tem duas conseqüências:

- Uma entrada de peso 1 resultará em uma seqüência de peso infinito, ou seja, a saída do codificador divergirá do caminho da seqüência nula;
- Existe uma família de entradas de peso 2 na forma $D^j(1+D^{q-1})$ que produzirão saídas de pesos finitos, ou seja, divergem do caminho da seqüência nula e mais tarde reencontram o mesmo.

O entrelaçador, por sua vez, troca as posições dos N bits de entrada, de modo que o código pareça aleatório para o canal, o que melhora o seu desempenho. Esta constatação já havia sido feita por *Shannon* [134], que sugeriu ser um bom código aquele que possuísse uma característica aleatória. O valor de N deve ser grande, geralmente maior que 1000. O perfurador tem o objetivo de excluir bits de paridade e adaptar a taxa do código, mantendo a máxima distância entre as palavras-código. Como os componentes do codificador turbo são lineares, o código final é linear.

Sabe-se que o erro na recepção de um bit, dado que o sinal foi transmitido com modulação BPSK e corrompido por ruído gaussiano branco aditivo (*Additive White Gaussian Noise* - AWGN), é calculado por

$$P_e = Q\left(\sqrt{SNR}\right) = Q\left(\sqrt{\frac{E_c}{E_n}}\right) \tag{5.2}$$

onde Q é a *função de distribuição cumulativa complementar* [135], ou seja, a área sob a região de erro na função de *distribuição de probabilidade* gaussiana, E_c é a energia do símbolo de canal (*chip*) e E_n é a energia do ruído. O gráfico da função $Q(x)$ está ilustrado na Figura 5.2. Como cada bit de informação é transmitido com uma taxa de codificação r, a razão sinal ruído é calculada como

$$SNR = \frac{E_c}{E_N} = \frac{rE_b}{N_0/2} = \frac{2rE_b}{N_0}, \tag{5.3}$$

onde E_b é a energia de bit e $N_0/2$ é a densidade espectral do ruído.

Se a palavra código $00\ldots0$ (zero) for transmitida, a probabilidade de a mesma ser decodificada como uma outra palavra código X_k, com $k \in \{1, 2, \ldots 2^N - 1\}$, sendo N o tamanho da palavra de dados e utilizando-se um decodificador de máxima verossimilhança, é

$$P_e = Q\left(\sqrt{\frac{2rd_kE_b}{N_0}}\right), \tag{5.4}$$

onde d_k é o peso de *Hamming* da k-ésima palavra código. Então, a taxa de erro de bits pode ser dada por

$$BER = \bigcup_{k=1}^{2^N-1} P(X_k|0) \leq \sum_{k=1}^{2^N-1} \frac{w_k}{N} Q\left(\sqrt{\frac{2rd_kE_b}{N_0}}\right), \tag{5.5}$$

o que resulta em

$$BER \leq \sum_{w=1}^{N} \sum_{v=1}^{\binom{N}{w}} \frac{w}{N} Q\left(\sqrt{\frac{2rd_{wv}E_b}{N_0}}\right), \tag{5.6}$$

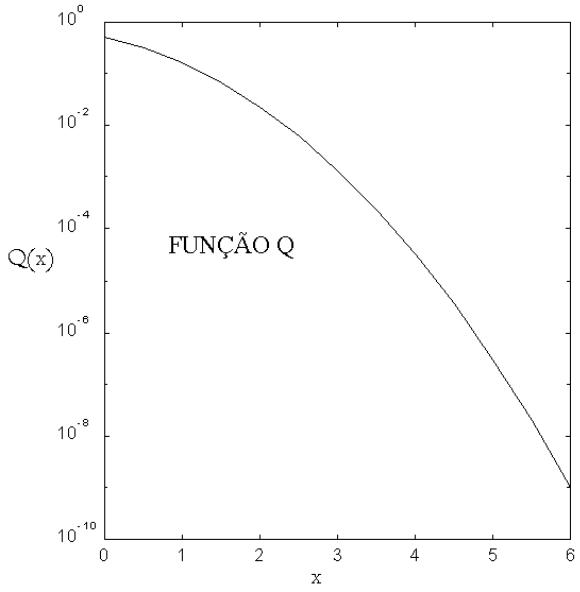


Figura 5.2: Gráfico da função Q.

onde N é o tamanho da palavra de dados e d_{wv} é o peso da v-ésima palavra código produzida por uma entrada de peso w. Avaliando-se os termos do somatório:

- Para $w = 1$, o peso da saída será muito grande e o termo pode ser desprezado;
- Para $w = 2$, apenas uma fração das entradas será divisível por $g_1(D)$ e produzirá palavras-código de peso mínimo nos dois codificadores componentes ($d^{TC}_{2,min}$);
- Para $w > 2$, o mesmo argumento de $w = 2$ é válido, com um número de palavras de peso mínimo geralmente menor.

Sendo assim, a probabilidade de erro pode ser aproximada por

$$BER \approx \max_{w \geq 2} \left\{ \frac{w.n_w}{N} Q\left(\sqrt{\frac{2rd^{TC}_{w,min}E_b}{N_0}} \right) \right\}, \tag{5.7}$$

onde n_w é o número das palavras código de distância mínima e, juntamente com $d^{TC}_{w,min}$, é função do entrelaçador utilizado.

Com base na equação acima, é fácil notar que a eficácia do código pode ser elevada simplesmente aumentando-se o tamanho do entrelaçador (N na equação (5.7)), fato conhecido como "ganho do entrelaçador". Além disso, se os códigos constituintes não fossem recursivos, a divisão por $g_1(D)$ não existiria e muitos outros termos

seriam levados em consideração na avaliação da probabilidade de erro, diminuindo a capacidade dos códigos turbo.

Na decodificação de códigos turbo, o algoritmo de Viterbi não é uma opção viável, pois o uso do entrelaçador resulta em uma estrutura de treliça muito complexa. Entretanto, para este tipo de abordagem, é muito mais eficiente utilizar estratégias sub-ótimas, onde dois decodificadores operam paralelamente e cooperativamente, como o algoritmo em [136], conhecido como BCJR. No algoritmo BCJR, a decisão sobre o dado transmitido é baseada na probabilidade a posteriori logarítmica, definida como

$$L(u_k) = \log_e \left(\frac{P(u_k = +1|y)}{P(u_k = -1|y)} \right), \tag{5.8}$$

onde u_k (ou x_k) é a entrada original do codificador (modulada em BPSK) e y a versão corrompida de x. Se o resultado é maior que zero, $+1$ foi transmitido; caso contrário, -1. Como os codificadores são convolucionais e podem ser descritos por treliças,

$$L(u_k) = \log_e \left(\frac{\sum_{S^+} p(s_{k-1} = s', s_k = s, y)/p(y)}{\sum_{S^-} p(s_{k-1} = s', s_k = s, y)/p(y)} \right), \tag{5.9}$$

onde $s_k \in S$ é o estado do codificador no momento k, S^+ é o conjunto de transições (s', s) causado por uma entrada $+1$ e S^- para uma entrada -1.

O único fator variável na equação acima é $p(s_{k-1} = s', s_k = s, y)$, que será chamado de $\mathcal{P}$, e pode ser calculado como

$$\begin{aligned}
\mathcal{P} &= p(s_{k-1} = s', s_k = s, y_1^k, y_{k+1}^N) \\
&= p(y_{k+1}^N | s_{k-1} = s', s_k = s, y_1^k) p(s_{k-1} = s', s_k = s, y_1^k) \\
&= p(y_{k+1}^N | s_{k-1} = s', s_k = s, y_1^k) p(s_k = s, y_k | s_{k-1} = s', y_1^{k-1}) p(s_{k-1} = s', y_1^{k-1}) \\
&= p(y_{k+1}^N | s_k = s) p(s_k = s, y_k | s_{k-1} = s') p(s_{k-1} = s', y_1^{k-1}) \\
&= \beta_k(s).\gamma_k(s', s).\alpha_{k-1}(s'),
\end{aligned} \tag{5.10}$$

com

$$\gamma_k(s', s) = p(s_k = s, y_k | s_{k-1} = s'), \tag{5.11}$$

$$\alpha_k(s) = p(s_k = s, y_1^k) = \sum_{s' \in S} \alpha_{k-1}(s') \gamma_k(s', s) \tag{5.12}$$

e

$$\beta_{k-1}(s') = p(y_k^N | s_{k-1} = s') = \sum_{s \in S} \beta_k(s) \gamma_k(s', s). \tag{5.13}$$

É importante salientar que $\alpha_k(s)$ é obtido com uma recursão para frente e $\beta_{k-1}(s')$ com uma recursão para trás.

Apesar do fator $p(y)$ ser constante, a sua remoção causaria uma instabilidade numérica no algoritmo para a computação da probabilidade *a posteriori* logarítmica. Este termo pode ser mantido definindo-se

$$\tilde{\alpha}_k(s) = \alpha_k(s)/p(y_1^k) \tag{5.14}$$

e

$$\tilde{\beta}_k(s) = \beta_k(s)/p(y_{k+1}^N|y_1^k), \tag{5.15}$$

e dividindo-se $p(s_{k-1} = s', s_k = s, y)$ por $p(y_1^{k-1})p(y_{k+1}^N|y_1^k)$, o que resulta em

$$p(s_{k-1} = s', s_k = s|y)p(y_k) = \tilde{\alpha}_{k-1}(s').\gamma_k(s', s).\tilde{\beta}_k(s), \tag{5.16}$$

sendo que $\tilde{\alpha}_k(s)$ e $\tilde{\beta}_k(s)$ podem ser calculados recursivamente como

$$\tilde{\alpha}_k(s) = \frac{\sum_{s'} \tilde{\alpha}_{k-1}(s')\gamma_k(s', s)}{\sum_s \sum_{s'} \tilde{\alpha}_{k-1}(s')\gamma_k(s', s)} \tag{5.17}$$

e

$$\tilde{\beta}_{k-1}(s') = \frac{\sum_s \tilde{\beta}_k(s)\gamma_k(s', s)}{\sum_s \sum_{s'} \tilde{\alpha}_{k-1}(s')\gamma_k(s', s)} \tag{5.18}$$

Ao final, a probabilidade a posteriori torna-se

$$L(u_k) = \log_e \left(\frac{\sum_{S^+} \tilde{\alpha}_{k-1}(s').\gamma_k(s', s).\tilde{\beta}_k(s)}{\sum_{S^-} \tilde{\alpha}_{k-1}(s').\gamma_k(s', s).\tilde{\beta}_k(s)} \right). \tag{5.19}$$

Utilizando-se a regra de *Bayes*, a probabilidade *a posteriori* pode ser escrita como

$$\begin{aligned}
L(u_k) &= \log_e \left(\frac{P(u_k = +1|y)}{P(u_k = -1|y)} \right) \\
&= \log_e \left(\frac{\frac{P(u_k=+1,y)}{P(y)}}{\frac{P(u_k=-1,y)}{P(y)}} \right) \\
&= \log_e \left(\frac{P(u_k = +1, y)}{P(u_k = -1, y)} \right) \\
&= \log_e \left(\frac{P(y|u_k = +1)P(u_k = +1)}{P(y|u_k = -1)P(u_k = -1)} \right) \\
&= \log_e \left(\frac{P(y|u_k = +1)}{P(y|u_k = -1)} \right) + \log_e \left(\frac{P(u_k = +1)}{P(u_k = -1)} \right),
\end{aligned} \tag{5.20}$$

onde o segundo termo é conhecido como informação *a priori*, ou seja, um conhecimento prévio (ou estimativa) das probabilidades dos símbolos enviados. Para

decodificadores convencionais, este valor pode ser considerado como zero ($P(u_k = +1) = P(u_k = -1) = 1/2$), porém, dada a estratégia de decodificação considerada neste trabalho e representada na Figura 5.3, o segundo termo é a informação extrínseca recebida do outro decodificador do conjunto. Assim, cada decodificador recebe uma estimativa do outro, seqüencialmente, refinando-a e retornando a sua própria estimativa.

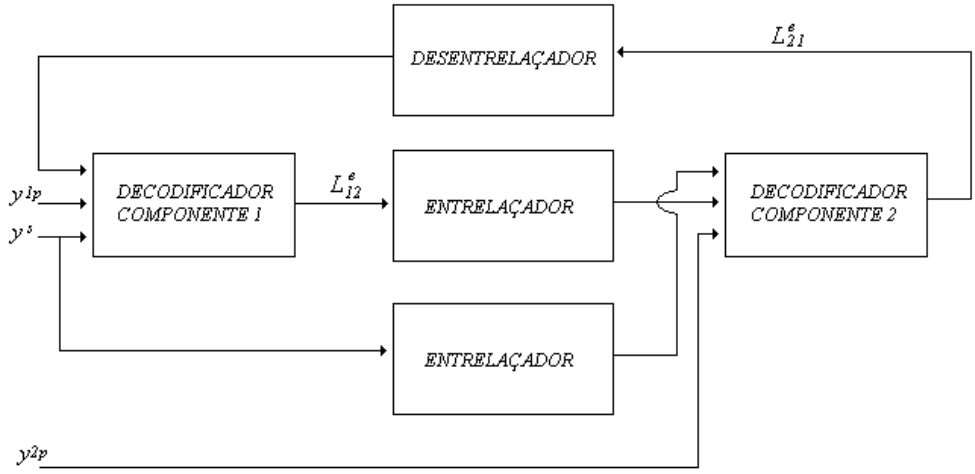


Figura 5.3: Decodificador turbo iterativo.

A informação extrínseca de cada decodificador pode ser obtida observando-se que

$$\begin{aligned}
\gamma_k(s', s) &= p(s_k = s, y_k | s_{k-1} = s') \\
&= \frac{p(s_k = s, s_{k-1} = s', y_k)}{p(s_{k-1} = s')} \\
&= \frac{p(y_k | s_k = s, s_{k-1} = s') p(s_k = s, s_{k-1} = s')}{p(s_{k-1} = s')} \\
&= p(y_k | s_k = s, s_{k-1} = s') p(s_k = s | s_{k-1} = s') \\
&= p(y_k | u_k) P(u_k), \qquad (5.21)
\end{aligned}$$

o que coloca $\gamma_k(s', s)$ numa forma similar à da probabilidade *a posteriori*. Pode-se definir a probabilidade $P(u_k)$ do bit de entrada u_k de modo que $P(u_k) = P(u_k = +1)$ para $u_k = +1$ e $P(u_k) = P(u_k = -1)$ para $u_k = -1$, ou seja, a probabilidade individual de cada bit ocorrer, levando-se em consideração a informação extrínseca, o que resulta em

$$P(u_k) = \left(\frac{e^{\left(-\frac{L^e(u_k)}{2}\right)}}{1 + e^{(-L^e(u_k))}} \right) e^{\left(u_k \frac{L^e(u_k)}{2}\right)} = K_1 e^{\left(u_k \frac{L^e(u_k)}{2}\right)}, \qquad (5.22)$$

onde

$$L^e(u_k) = \log_e \left(\frac{P(u_k = +1)}{P(u_k = -1)} \right). \tag{5.23}$$

O valor de $\gamma_k(s', s)$ pode ser completamente definido observando-se que as informações enviada e recebida consistem em bits sistemáticos (bits de informação) e bits de paridade, e que o canal em questão sofre a adição de ruído gaussiano, resultando em

$$\begin{aligned} p(y_k|u_k) &= p(y^s, y^p|u_k, x^p) \\ &= p(y^s|u_k).p(y^p|x^p) \\ &\propto e^{-\frac{(y_k^s - u_k^s)^2}{2\sigma^2}}.e^{-\frac{(y_k^p - x_k^p)^2}{2\sigma^2}} \\ &= e^{-\frac{(y_k^s - u_k^s)^2}{2\sigma^2} - \frac{(y_k^p - x_k^p)^2}{2\sigma^2}} \\ &= e^{-\frac{(y_k^s)^2 + u_k^2 + (y_k^p)^2 + (x_k^p)^2}{2\sigma^2}}.e^{\frac{u_k y_k^s + x_k^p y_k^p}{\sigma^2}} \\ &= K_2.e^{\frac{u_k y_k^s + x_k^p y_k^p}{\sigma^2}}. \end{aligned} \tag{5.24}$$

Logo, o valor final de $\gamma_k(s', s)$ será

$$\gamma_k(s', s) \propto e^{\left(u_k \frac{L^e(u_k)}{2}\right)}.e^{\frac{u_k y_k^s + x_k^p y_k^p}{\sigma^2}}. \tag{5.25}$$

Supondo-se que a modulação utilizada seja BPSK e

$$\frac{E_c}{N_0/2} = \frac{1}{\sigma^2}, \tag{5.26}$$

onde $E_c = rE_b$ é a energia de chip (símbolo de canal), $\gamma_k(s', s)$ pode ser refinado em

$$\begin{aligned} \gamma_k(s', s) &\propto e^{\left(\frac{u_k\left(L^e(u_k) + \frac{4E_c}{N_0} y_k^s\right)}{2}\right)} e^{\left(\frac{1}{2}\frac{4E_c}{N_0} x_k^p y_k^p\right)} \\ &= e^{\left(\frac{u_k\left(L^e(u_k) + L_c y_k^s\right)}{2}\right)} e^{\left(\frac{1}{2} L_c x_k^p y_k^p\right)} \\ &= e^{\left(\frac{u_k\left(L^e(u_k) + L_c y_k^s\right)}{2}\right)} \gamma_k^e(s', s). \end{aligned} \tag{5.27}$$

$\gamma_k^e(s', s)$ é parte da informação extrínseca do decodificador atual e será repassada mais tarde para o outro decodificador.

Como resultado final do desenvolvimento, tem-se

$$\begin{aligned} L(u_k) &= \log_e \left(\frac{\sum_{S^+} \tilde{\alpha}_{k-1}(s').\gamma_k(s', s).\tilde{\beta}_k(s)}{\sum_{S^-} \tilde{\alpha}_{k-1}(s').\gamma_k(s', s).\tilde{\beta}_k(s)} \right) \\ &= L_c y_k^s + L^e_{comp\_dec}(u_k) + \log_e \left(\frac{\sum_{S^+} \tilde{\alpha}_{k-1}(s').\gamma_k^e(s', s).\tilde{\beta}_k(s)}{\sum_{S^-} \tilde{\alpha}_{k-1}(s').\gamma_k^e(s', s).\tilde{\beta}_k(s)} \right) \\ &= L_c y_k^s + L^e_{comp\_dec}(u_k) + L^e_{current\_dec}(u_k), \end{aligned} \tag{5.28}$$

onde $L_c y_k^s$ é o valor de canal (composto por quantidades relativas ao canal de comunicação), $L^e_{comp\_dec}(u_k)$ é a informação extrínseca enviada pelo decodificador complementar e $L^e_{current\_dec}(u_k)$ é a informação extrínseca que está sendo calculada e será enviada para o outro decodificador. A cada sub-iteração, esta equação é avaliada por um dos decodificadores, que são ativados em série, sendo então enviada a informação extrínseca para o outro decodificador.

Os *codecs* turbo geralmente apresentam um ótimo desempenho, principalmente quando se trabalha com blocos de tamanho elevado (aumentando o ganho do entrelaçador). Uma das características associadas com a sua utilização é a decodificação complexa e iterativa, além de bastante susceptível a parâmetros e erros numéricos, o que torna mais difícil a sua implementação em *hardware*. Os códigos turbo irregulares geralmente apresentam um desempenho melhor que o dos regulares [135] (vários padrões possíveis de entrelaçamento).

5.2 Os códigos LDPC

Os códigos LDPC (*Low-Density Parity-check Code*) [77] constituem uma outra classe de códigos corretores de erros que proporcionam desempenho muito próximo da capacidade do canal, com decodificadores implementáveis em *software* ou *hardware*. Estes códigos foram propostos por Gallager em [137] e redescobertos por Tanner em [138] e MacKay em [139, 140], que perceberam os benefícios de códigos com matrizes esparsas no problema de correção de erros em canais de comunicação.

Os códigos LDPC são, na verdade, códigos em bloco lineares que possuem matrizes de checagem de paridade H com baixa densidade de 1's. Um código do tipo regular apresenta matrizes H com exatamente w_c 1's em cada coluna e $w_r = w_c(M/C)$ (M é o tamanho da palavra-código e C o número de equações de checagem de paridade) 1's em cada linha, sendo a taxa de código $R = 1 - w_c/w_r$. Para códigos irregulares, o número de 1's não é constante e é dado pelos polinômios de distribuição de grau (grau significa o número de 1's)

$$\lambda(x) = \sum_{d=1}^{d_m} \lambda_d x^{d-1} \tag{5.29}$$

e

$$\rho(x) = \sum_{d=1}^{d_s} \rho_d x^{d-1}, \tag{5.30}$$

onde o primeiro descreve o número de 1's por linha, com λ_d sendo a razão de todas as linhas com d 1's e o número total e d_m o máximo valor de 1's, e o segundo o número de 1's por coluna, com ρ_d sendo a razão de todas as colunas com d 1's e o número total e d_s o máximo valor de 1's. O significado do polinômio fica assim: ax^b significa que $a \times 100\%$ das linhas/colunas apresenta um número $b+1$ de 1's. tem Em geral, assim como nos códigos turbo, o desempenho dos códigos irregulares é superior ao dos códigos regulares [135].

Os códigos LDPC podem ser eficientemente representados por grafos bipartidos, conhecidos como grafos de Tanner [138], tendo com o código LDPC a mesma relação que a treliça com o código convolucional. Tais grafos são compostos por nós variáveis (representando os bits da palavra-código), nós de checagem (representando as equações de checagem da matriz H) e ramos, que os interconectam de acordo com os 1's na matriz H.

Por exemplo, supondo-se um código LDPC com matriz de cheque de paridade

$$H = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}, \tag{5.31}$$

o grafo associado a mesma seria o mostrado na Figura 5.4.

A propriedade de checagem da matriz H é baseada na seguinte observação: todos os bits conectados a um nó de checagem devem resultar em uma soma módulo-2 nula ($H^T G = 0$). O grafo de Tanner é uma ferramenta extremamente importante tanto para se entender o método de decodificação dos códigos LDPC como para se identificar deficiências nos projetos. Por exemplo, ciclos no grafo de Tanner,

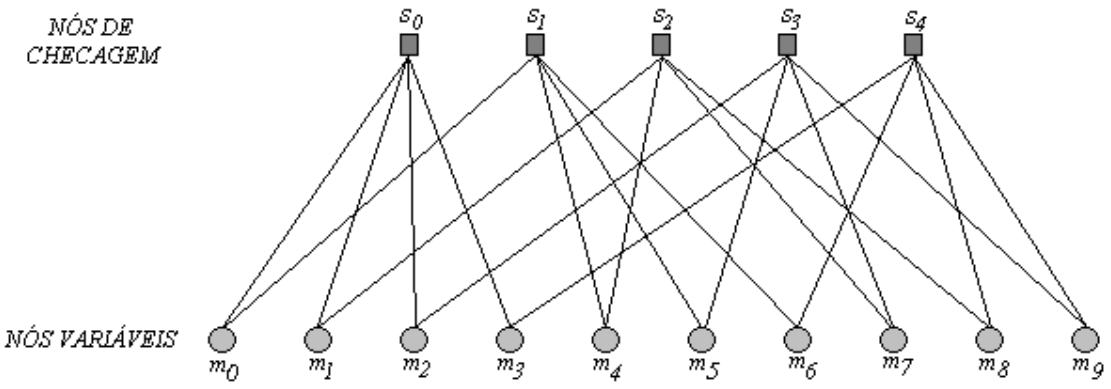


Figura 5.4: Representação gráfica de um código LDPC.

que são caracterizados por caminhos fechados, reduzem bastante o desempenho dos algoritmos de decodificação geralmente utilizados (troca de mensagens).

O algoritmo de decodificação utilizado nos códigos LDPC é geralmente conhecido como "algoritmo de troca de mensagens" e tem desempenho quase ótimo. Assim como acontece nos códigos turbo, o algoritmo avalia a probabilidade *a posteriori* de um bit da palavra-código transmitida ser 1, dada a palavra-código recebida. Com $Pr(m_i = 1|y)$, onde m_i é o bit i da palavra código enviada e y é a palavra código recebida, cada nó, de forma iterativa, avalia as mensagens recebidas dos nós aos quais está conectado, refina a estimativa, e retorna uma outra mensagem, como representado na Figura 5.5. Neste caso, o nó m_0 recebe a informação do nó s_0 e, juntamente com o sinal recebido y, determina uma estimativa $Pr(m_0 = b|y, s_0)$, enviando-a para s_1. Isto significa que m_0 só passa informações novas, ou extrínsecas, computadas para cada par $\{m_i, s_j\}$.

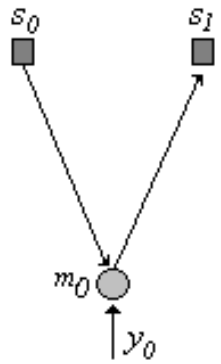


Figura 5.5: Diagrama de nós vizinhos de m_0 na Figura 5.4.

Na sub-iteração dos nós de checagem, ocorre um processamento idêntico, representado na Figura 5.6, onde informações extrínsecas são passadas dos nós de checagem para os nós variáveis, para cada par $\{s_i, m_j\}$. Sendo assim, o processo de decodificação dos códigos LDPC é realizado em duas etapas: na primeira, mensagens são enviadas dos nós variáveis para os nós de checagem, levando-se em consideração a informação de canal e as mensagens recebidas de todos os outros nós de checagem, exceto do nó em questão; numa segunda etapa, um processo idêntico ocorre dos nós de checagem para os variáveis. Após um dado número de iterações, o decodificador computa as estimativas e decide sobre a palavra $m = \{m_1, m_2, ...m_n\}$ recebida. O algoritmo supõe que as mensagens são estatisticamente independentes, o que é verdade desde que o grafo não possua ciclos.

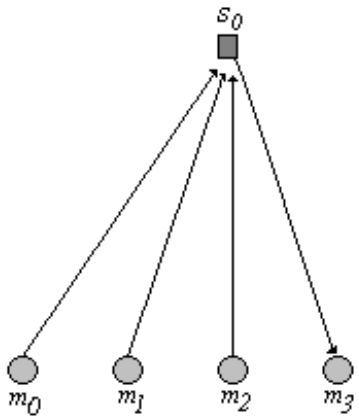


Figura 5.6: Diagrama de nós vizinhos de s_0 na Figura 5.4.

Em resumo, a mensagem enviada de um nó variável m_i consiste na probabilidade de que $m_i = b$, com $b = \{0, 1\}$, dado o valor recebido do canal (y_i), as mensagens enviadas pelos nós de checagem na iteração anterior $(M_{s,m})$ e que as equações de

checagem sejam satisfeitas (S), ou seja

$$
\begin{aligned}
M_{m_i,s_j}(0) = & \\
= & P(m_i = 0 | y_i, S_i, \bigcup_{k \neq j} M_{s_k,m_i}(0)) \\
= & \frac{P(m_i = 0, y_i, S_i, \bigcup_{k \neq j} M_{s_k,m_i}(0))}{P(y_i, S_i, \bigcup_{k \neq j} M_{s_k,m_i}(0))} \\
= & \frac{P(S_i | m_i = 0, y_i, \bigcup_{k \neq j} M_{s_k,m_i}(0)) P(m_i = 0, y_i, \bigcup_{k \neq j} M_{s_k,m_i}(0))}{P(y_i, S_i, \bigcup_{k \neq j} M_{s_k,m_i}(0))} \\
= & \frac{P(S_i | m_i = 0, y_i, \bigcup_{k \neq j} M_{s_k,m_i}(0)) P(\bigcup_{k \neq j} M_{s_k,m_i}(0) | m_i = 0, y_i) P(m_i = 0, y_i)}{P(y_i, S_i, \bigcup_{k \neq j} M_{s_k,m_i}(0))} \\
= & \frac{P(S_i | m_i = 0, y_i, \bigcup_{k \neq j} M_{s_k,m_i}(0)) P(\bigcup_{k \neq j} M_{s_k,m_i}(0) | m_i = 0, y_i) P(m_i = 0 | y_i) P(y_i)}{P(y_i, S_i, \bigcup_{k \neq j} M_{s_k,m_i}(0))} \\
= & \frac{P(S_i | m_i = 0, y_i, \bigcup_{k \neq j} M_{s_k,m_i}(0)) P(\bigcup_{k \neq j} M_{s_k,m_i}(0) | m_i = 0, y_i) P(m_i = 0 | y_i)}{P(S_i, \bigcup_{k \neq j} M_{s_k,m_i}(0) | y_i)} \\
= & \frac{P(S_i | m_i = 0, y_i, \bigcup_{k \neq j} M_{s_k,m_i}(0)) P(\bigcup_{k \neq j} M_{s_k,m_i}(0) | m_i = 0, y_i) P(m_i = 0 | y_i)}{P(S_i | y_i) P(\bigcup_{k \neq j} M_{s_k,m_i}(0) | y_i)} \\
= & \frac{P(S_i | m_i = 0, y_i, \bigcup_{k \neq j} M_{s_k,m_i}(0)) P(m_i = 0 | y_i)}{P(S_i)} \\
= & (1 - P_i) P(S_i | m_i = 0, y_i, \bigcup_{k \neq j} M_{s_k,m_i}(0)) / P(S_i) \\
= & (1 - P_i) P(S_i | m_i = 0, y_i, M_{s_1,m_i}(0), ..., M_{s_C,m_i}(0)) / P(S_i) \\
= & (1 - P_i) P(S_{i,1} | m_i = 0, y_i, M_{s_1,m_i}(0)) ... P(S_{i,C} | m_i = 0, y_i, M_{s_C,m_i}(0)) / P(S_i) \\
= & (1 - P_i) P(S_{i,1} | m_i = 0, M_{s_1,m_i}(0)) ... P(S_{i,C} | m_i = 0, M_{s_C,m_i}(0)) / P(S_i) \\
= & (1 - P_i) P(S_{i,1} | m_i = 0, \bigcup_{k \neq i} M_{m_k,s_1}(0)) ... P(S_{i,C} | m_i = 0, \bigcup_{k \neq i} M_{m_k,s_C}(0)) / P(S_i) \\
= & K_{i,j} (1 - P_i) \prod_{l \neq j} M_{s_l,m_i}(0) \qquad (5.32)
\end{aligned}
$$

e

$$
\begin{aligned}
M_{m_i,s_j}(1) &= P(m_i = 1 | y_i, S_i, \bigcup_{k \neq j} M_{s_k,m_i}(1)) \\
&= K_{i,j} P_i \prod_{l \neq j} M_{s_l,m_i}(1), \qquad (5.33)
\end{aligned}
$$

onde $K_{i,j}$ é escolhido para fazer com que $M_{m_i,s_j} + M_{m_i,s_j} = 1$, $P_i = P(m_i = 1|y_i)$ e $S_{i,j}$ é a equação de checagem do nó de checagem j que envolve o nó variável i. S_i e M_{s_k,m_i} são considerados independentes, pois as equações de checagem envolvendo o nó variável i são resultantes de outras mensagens (as mensagens enviadas para um nó são independentes da informação anteriormente enviada pelo mesmo). No caso dos nós de checagem, as mensagens enviadas consistem na probabilidade de que as equações de checagem de paridade sejam satisfeitas, dado $m_i = b$, com $b = \{0, 1\}$, e as mensagens enviadas pelos nós variáveis na iteração anterior ($M_{m,s}$). A probabilidade de que as equações sejam satisfeitas com $m_i = 0$ depende, na verdade, de um número par de 1's enviados pelos outros nós variáveis. Dada uma seqüência k_n independente e com M elementos e sabendo-se que $P(k_n = 1) = p_n$ e $P^L_{\#1=par}$ é a probabilidade de uma seqüência com L elementos ter um número par de 1's:

$$
\begin{aligned}
Para\ M = 1,\ P^1_{\#1=par} &= (1 - p_0) \\
Para\ M > 1,\ P^M_{\#1=par} &= \left(P^{M-1}_{\#1=par} \bigcap P(k_M = 0)\right) \bigcup \left(P^{M-1}_{\#1=\text{ímpar}} \bigcap P(k_M = 1)\right) \\
&= P^{M-1}_{\#1=par}(1 - p_M) + (P^{M-1}_{\#1=par} - 1)p_M \\
&= p_M + P^{M-1}_{\#1=par}(1 - 2p_M) \\
&= p_M + \sum_{k=2}^{M} \left[p_k \prod_{l=k+1}^{M} (1 - 2p_l) \right] + (1 - p_0) \prod_{l=2}^{M} (1 - 2p_l) \\
&= \frac{1}{2} - \frac{1}{2} \prod_{l=2}^{M} (1 - 2p_l) + (1 - p_0) \prod_{l=2}^{M} (1 - 2p_l) \\
&= \frac{1}{2} + \frac{1}{2} \prod_{l=2}^{M} (1 - 2p_l). \qquad (5.34)
\end{aligned}
$$

Logo, substituindo-se a seqüência independente pelas mensagens $M_{m_i,s_j}(b)$, com $b = \{0, 1\}$, tem-se

$$
\begin{aligned}
M_{s_j,m_i}(0) &= \frac{1}{2} + \frac{1}{2} \prod_{l \neq i} (1 - 2M_{m_i,s_j}(1)) \\
M_{s_j,m_i}(1) &= 1 - M_{s_j,m_i}(0) \qquad (5.35)
\end{aligned}
$$

O algoritmo é iniciado com $M_{m_i,s_j}(b) = P(m_0 = b|y_i)$, sendo que as estatísticas são refinadas a cada iteração. $P(m_0 = b|y_i)$ é dependente do canal, assim como na seção 5.1 e, para um canal que sofre a adição de ruído gaussiano, torna-se

$$
P(x_0 = c|y_i) = \left[1 + e^{\frac{-2yx}{\sigma^2}}\right]^{-1}, \qquad (5.36)
$$

onde $x_i = 1-2m_i$. A decodificação termina, por exemplo, quando o número máximo de iterações é alcançado ou quando $\tilde{m}H = 0$. Para evitar instabilidades numéricas, utiliza-se geralmente o decodificador no domínio do logaritmo, com

$$L(m_i) = \log_e \left(\frac{P(m_i = 1|y)}{P(m_i = 0|y)} \right). \tag{5.37}$$

O ótimo desempenho dos códigos LDPC advém do fato da matriz H ser esparsa, o que resulta numa grande capacidade de correção de erros. Há simulações realizadas onde códigos LDPC irregulares ficaram a apenas 0,04dB do limite de *Shannon*, com uma taxa de erro de bit de 10^{-6} em um canal AWGN [141]. Entretanto, essa característica só é explorada em sua plenitude se os blocos de codificação forem extremamente grandes. Por exemplo, a simulação mencionada em [141] utilizou blocos de comprimento 10^7.

Capítulo 6

A compressão distribuída de eletrocardiograma

Como já foi mencionado, atualmente há grande interesse no diagnóstico remoto de doenças, o que incorre na transmissão de grandes quantidades de dados para o consultório médico. Dado que as redes utilizadas para este fim geralmente apresentam larguras de banda reduzidas, além de um certo custo por *kilobyte* transmitido, a compressão desses dados acaba sendo um ponto de grande importância, necessitando de algoritmos com altas taxas de compressão e baixa degradação de sinal, preservando a informação médica a ser utilizada no diagnóstico. Além disso, é interessante que tais métodos sejam simples e baratos, o que aumentaria sua popularização entre médicos e pacientes e possibilitaria a sua incorporação a dispositivos de comunicação pessoais, como celulares, permitindo monitoramento diário e fácil transmissão dos dados adquiridos.

O ECG [100, 101], que é utilizado para o diagnóstico de doenças cardíacas, é um exemplo desse tipo de dado. Como também já foi mencionado, o monitoramento de pacientes para diagnóstico pode envolver vários canais durante 24 horas, demandando dispositivos com grandes capacidades de processamento e armazenamento.

Até onde se sabe, dada a literatura atual, todos os métodos desenvolvidos para a compressão de sinais de eletrocardiograma são baseados na abordagem tradicional, que trabalha com *codecs* assimétricos, onde o maior esforço de compressão é executado pelo codificador. Tal característica é aceitável quando o exame é realizado no consultório médico (onde dispositivos profissionais apropriados estão disponíveis) ou

quando um *checkup* completo é necessário e o eletrocardiograma é apenas uma pequena parte do resultado final, o que exige que o paciente se desloque até o hospital. Por outro lado, se o principal ou único objetivo é diagnosticar doenças cardíacas e o paciente deve ser monitorado por um longo período de tempo durante a sua rotina diária, ou o mesmo está distante do consultório, seria interessante que um dispositivo simples fosse capaz de codificar o sinal e transmitir o mesmo através de uma dada rede de comunicação. Além disso, se o paciente já possuir um dispositivo específico, como um marca-passo, seria desejável que o mesmo pudesse gravar o ECG, durante uma situação anormal, a enviá-lo para um posterior diagnóstico. Dadas as restrições inerentes a esses casos de uso, um esquema simples de compressão de ECG reduziria o custo do dispositivo e difundiria o seu uso, beneficiando tanto médicos quanto pacientes.

Este capítulo propõe uma nova classe de codificadores de ECG, baseados num paradigma de reversão de complexidade: a carga computacional relacionada à exploração das redundâncias é deslocada para o decodificador. Utilizando-se o teorema de Wyner-Ziv e um código de canal de alta capacidade de correção de erros (código turbo), é possivel deslocar a complexidade do esquema para o decodificador, gerando um codificador bastante simples. Duas abordagens diferentes são propostas: um *codec* no domínio da amostra e um outro *codec* no domínio da transformada; o primeiro utiliza diretamente as amostras de sinal e o segundo faz uso de uma transformação de descorrelação, que proporciona compactação de energia e facilita a compressão do ECG.

6.1 A estrutura para a compressão distribuída de ECG

Os codificadores distribuídos de ECG apresentados neste trabalho são baseados nas soluções desenvolvidas em [1, 142] para sinais de vídeo, que utilizam bits de paridade e um canal de *feedback* para controle de taxa. Os codificadores de Wyner-Ziv foram implementados com quantização uniforme e códigos turbo, com uma etapa de pré-processamento para adaptar o sinal de entrada e possibilitar a exploração da informação auxiliar no decodificador, além de proporcionar um maior desempenho

da máquina de codificação. O diagrama em blocos genérico dos *codecs* apresentados está ilustrado na Figura 6.1.

Na estrutura proposta, o sinal de ECG de entrada passa primeiramente por um bloco de pré-processamento, que recebe os batimentos originais (períodos de ECG) e normaliza o seu comportamento, tentando criar um sinal aproximadamente periódico (pelo menos todos os períodos terão exatamente o mesmo comprimento). Há quatro passos distintos: a detecção de período, a normalização de período, a equalização DC e a ordenação por similaridade, aplicadas nesta mesma ordem. Tais processamentos são realizados conforme descrito nas seções 4.2.4, 4.2.5 e 4.2.6. A detecção de período identifica todos os complexos QRS, o que pode ser executado com os algoritmos em [108, 111–113], deixando-se meio pico de QRS em cada extremidade do batimento; os períodos resultantes são então escalonados para o mesmo comprimento, conforme descrito na seção 4.2.4. Para se aumentar ainda mais a correlação entre períodos, os níveis DC dos batimentos são equalizados, grampeando-se cada batimento no menor valor maior ou igual a zero, como descrito na seção 4.2.5. O último passo é a ordenação por similaridade, que rearranja os períodos com base em sua semelhança, conforme descrito na seção 4.2.6. Primeiramente, o período com a menor variância é colocado na primeira posição; então, o período mais semelhante ao n-ésimo é colocado na posição $(n+1)$. Algumas dessas técnicas são utilizadas em compressores que representam o estado da arte na codificação de ECG [3, 4, 7] e também foram empregadas nos esquemas apresentados nas seções anteriores, além de serem facilmente incorporadas a codificadores padrão, como o H.264 intra-quadros e o JPEG2000 (ver a seção 4.2.5).

Suponha que $\{X_0, X_1, \ldots, X_{N-1}\}$ sejam os períodos pré-processados, chamados de segmentos. Após este primeiro passo, os períodos de ECG são clssificados em dois grupos: segmentos-chave (*Key Segments* - KS) e segmentos de Wyner-Ziv (*Wyner-Ziv Segments* - WZS). Cada grupo de Z segmentos é chamado de grupo de segmentos (*Group of Segments* - GoS) e compõe a unidade de processamento da estrutura proposta. A estrutura do GoS é mostrada na Figura 6.2. Para um GoS n, os segmentos-chave são dados por $KS_0 = X_{nZ}$ e $KS_{Z-1} = X_{nZ+(Z-1)}$, com $n = \{0, 1, \ldots, N_{GoS} - 1\}$, e os segmentos de Wyner-Ziv por $WZS_k = X_{nZ+1}$, com $k = \{1, \ldots, Z-2\}$ e $n = \{0, 1, \ldots, N_{GoS} - 1\}$; N_{GoS} é o número máximo do GoSs

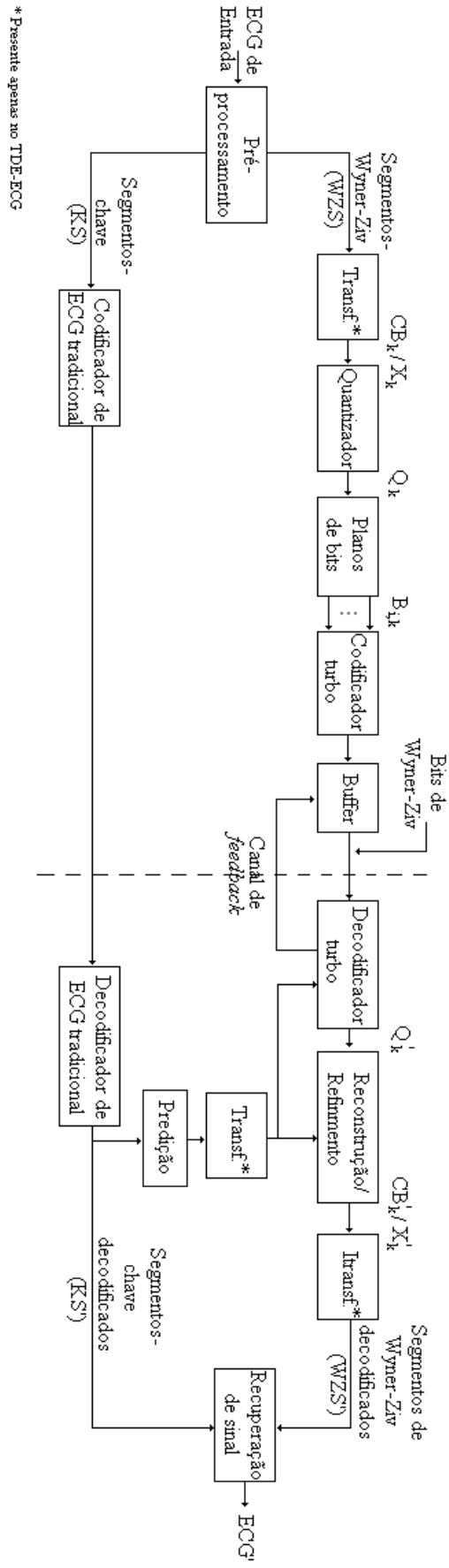


Figura 6.1: Codec genérico para a codificação distribuída de ECG.

existentes no sinal. No GoS, o primeiro e o último segmentos são chamados de segmentos-chave, que estão disponíveis como informação auxiliar no decodificador. O último segmento-chave do $(n-1)$-ésimo GoS é o primeiro segmento-chave do n-ésimo. De fato, estes segmentos podem ser comprimidos com um bom compressor de ECG e estar disponíveis no decodificador, antes da chegada dos segmentos de Wyner-Ziv. Entretanto, com o objetivo de uma análise mais simples, com foco nos segmentos que são processados utilizando-se o conceito de Wyner e Ziv, além de uma comparação, mesmo que grosseira, com os *codes* distribuídos já desenvolvidos para imagens [1, 142], assume-se que os KSs são reconstruídos perfeitamente no decodificador. Os outros $(Z-2)$ segmentos são chamados de segmentos Wyner-Ziv e são codificados utilizando-se conceitos de DSC.

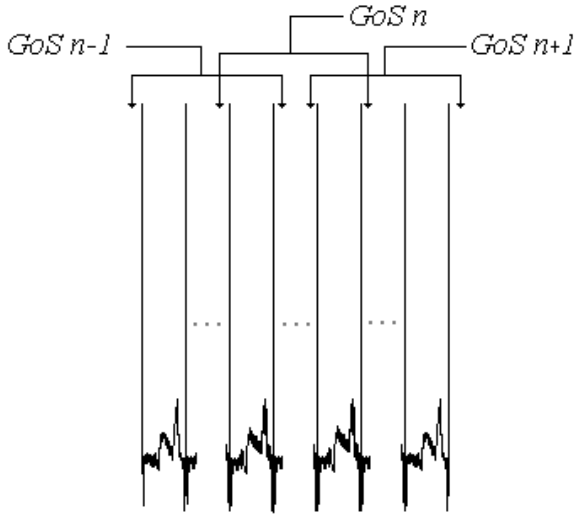


Figura 6.2: Estrutura do grupo de segmentos (GoS).

Dois *codecs* diferentes foram desenvolvidos: um *codec* no domínio da amostra, que processa diretamente as amostras de ECG e depois as envia para o codificador turbo, e um *codec* no domínio da transformada, que utilizada coeficientes de DCT ao invés das próprias amostras de sinal. Ambos fazem uso de conceitos de DSC para codificar o sinal de ECG e apresentam máquinas de codificação similares, porém, eles diferem no domínio no qual os dados são processados. Uma descrição detalhada de cada um será apresentada nas seções seguintes.

6.1.1 A etapa de pré-processamento

Como já foi dito, a etapa de pré-processamento é realizada como explicado nas seções 4.2.4, 4.2.5 e 4.2.6. Tal processamento é muito importante e tem o potencial de aumentar muito a correlação entre os períodos de ECG. Vale ressaltar que essa etapa também facilita a aplicação de conceitos de compressão distribuída de sinais, pois agora os segmentos de ECG podem ser diretamente comparados, ou seja, como os segmentos passam a possuir os mesmos comprimentos e níveis DC, a informação auxiliar pode ser facilmente gerada (*e.g.* através de interpolação). Se, por exemplo, os níveis DC não fossem os mesmos, uma interpolação (antecipada por algum processo de normalização) resultaria em um segmento num nível diferente do atual, dificultando o uso da informação obtida. A informação auxiliar resultante do pré-processamento é transmitida ao decodificador do mesmo modo apresentado nos capítulos anteriores, ou seja, o comprimento original de cada período, os níveis DC originais, a ordenação original e o nível de equalização são codificados com um codificador aritmético e enviados.

6.1.2 O *codec* no domínio da amostra

O primeiro tipo de codificador de ECG distribuído é chamado de codificador no domínio da amostra (*Sample-Domain ECG Encoder* – SDE-ECG) e não apresenta os blocos *Transf.* e *Itransf.* na Figura 6.1, processando diretamente as amostras de sinal. Os segmentos de Wyner-Ziv do GoS n são agrupados e formam um único vetor, que é então quantizado com um quantizador uniforme de 2^{kb} níveis, onde kb é o número de bits; uma palavra binária única é então atribuída a cada intervalo de quantização. Tal quantizador está ilustrado na Figura 6.3, onde $M = 2^{kb} - 1$, $N = 2^{kb}$ e SP é dado por

$$SP = \frac{2^{T_{min}}}{2^{kb}} \tag{6.1}$$

onde T_{min} é o número mínimo de bits necessário para a representação da faixa dinâmica do sinal. Os L vetores resultantes, com $L = 1, 2, 3, \ldots$, são agrupados e repassados para o bloco de extração de planos de bits. Os planos de bits i das amostras quantizadas k (amostras quantizadas de ECG), ou seja, $B_{i,k}$, são enviados, em seqüência, para o codificador turbo de Slepian-Wolf, que é implementado com

códigos turbo perfurados de taxa adaptável [78]. O codificador turbo é composto por dois codificadores convolucionais recursivos [143] idênticos, cada um com taxa de 1/2. A taxa total do conjunto fica então em 1/3, dado que o bit de paridade de cada codificador componente e o bit sistemático são considerados. Entretanto, apenas os bits de paridade são enviados; a informação sistemática é estimada no decodificador através dos segmentos-chave KS.

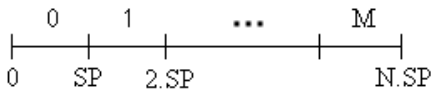


Figura 6.3: Estratégia de quantização para p SDE-ECG.

Vale ressaltar que o codificador turbo desenvolvido para a estrutura de compressão distribuída apresenta dois entrelaçadores, um antes de cada codificador componente, conforme ilustrado na Figura 6.4. Tal abordagem tenta atacar a propriedade de memória do "canal de correlação", pois a qualidade da informação auxiliar acaba sendo maior em algumas áreas que em outras. Isto implica que, em alguns planos de bits, haverá vários bits consecutivos afetados pelo "ruído de correlação", fazendo com que o primeiro codificador convolucional componente tenha que corrigir longas seqüências de bits incorretos, como num erro em rajada. Obviamente, este comportamento acaba exigindo um número maior de bits de paridade, porém, como os bits são enviados para um bloco inteiro e não apenas para uma determinada área, vários bits desnecessários serão requisitados. Logo, um outro entrelaçador antes do primeiro codificador componente tende a espalhar bastante áreas com muitos bits errados, realocando os mesmos em partes distintas do bloco de codificação, o que proporciona um melhor resultado em termos de taxa-distorção. O entrelaçador utilizado é do tipo *S-Random*, como apresentado em [144].

Os bits de paridade gerados pelo codificador turbo, chamados de bits de Wyner-Ziv, são armazenados num buffer e enviados para o decodificador quando requisitados (via canal de *feedback*). O número de bits enviados a cada pedido é regido por um simples esquema de perfuração, mostrado na Figura 6.5. Dado o período de perfuração N, também mostrado na Figura 6.5, e supondo que há um modo "nenhum bit", o primeiro padrão de perfuração é composto por zero bits; o segundo padrão é composto pelo grupo $G1$ de $P1$ (primeiro codificador), o terceiro pelos grupos $G1$

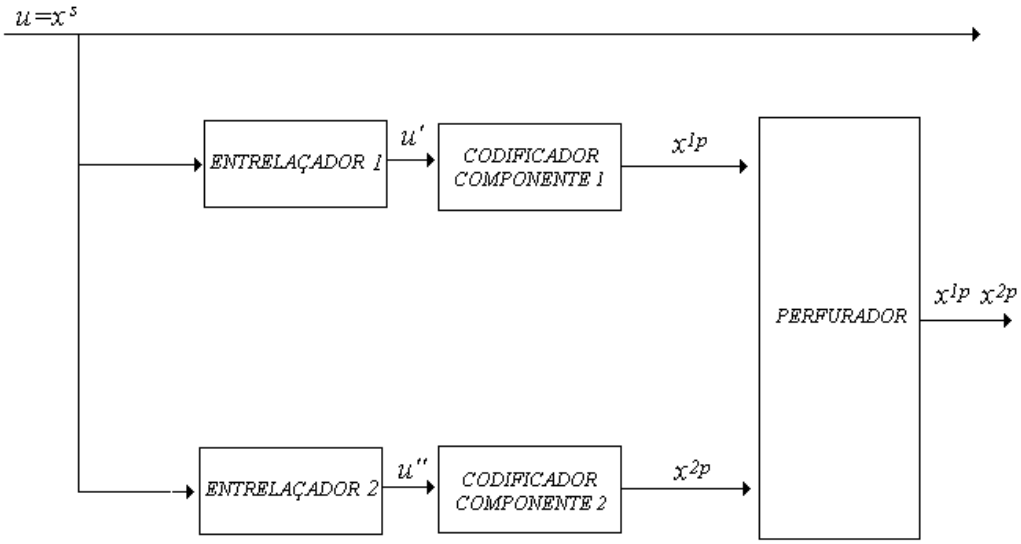


Figura 6.4: Estrutura adotada para o codificador turbo de Wyner-Ziv.

de $P1$ e $G1$ de $P2$, o quarto pelos grupos $G1$ e $G2$ de $P1$ e $G1$ de $P2$, o quinto pelos grupos $G1$ e $G2$ de $P1$ e $G1$ e $G2$ de $P2$ e o sexto pelos grupos $G1$, $G2$ e $G3$ de $P1$ e $G1$ e $G2$ de $P2$, sendo os demais montados da mesma maneira. Então, há no máximo $(2N + 1)$ padrões de perfuração e, a cada vez que o codificador pede mais bits, os elementos necessários para compor o próximo padrão na hierarquia são enviados. Os padrões de perfuração podem ser matematicamente descritos por

$$P_{Punct} = \{G(1)_{P_1}, \ldots, G(R)_{P_1}\} \cup \{G(1)_{P_2}, \ldots, G(S)_{P_2}\} \tag{6.2}$$

e

$$R = \left\lfloor \frac{(k+1)}{2} \right\rfloor, S = \left\lfloor \frac{k}{2} \right\rfloor, k = 0, 1, \ldots, 2N. \tag{6.3}$$

Vale a pena ressaltar que algumas implementações práticas de DSC empregam esquemas mais diretamente relacionados ao trabalho de Wyner e Ziv, enviando a síndrome de um dado *coset* do código. Ao invés disso, no presente trabalho enviam-se os bits de paridade e deixa-se a decisão do *coset* para o codificador turbo, ou seja, este fica responsável por agrupar as palavras código em conjuntos [1].

O decodificador gera a informação auxiliar a partir dos segmentos-chave de cada GoS. Para um dado GoS, a informação auxiliar para cada segmento do mesmo é calculada através de uma simples interpolação dos segmentos-chave do GoS, sendo

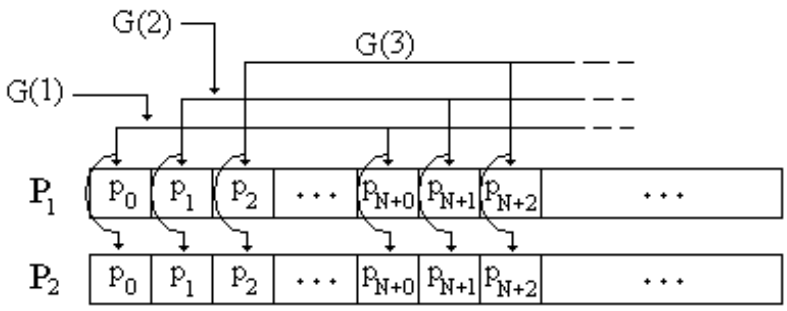


Figura 6.5: Estratégia de perfuração adotada.

dada por

$$SI_n(k) = \left\lfloor \frac{(Z-1-n).KS_0(k) + n.KS_{Z-1}(k)}{Z-1} \right\rfloor, n = 1, \ldots, Z-2, \tag{6.4}$$

onde $SI_n(k)$ é a informação auxiliar para a amostra k do segmento de Wyner-Ziv n, $KS_x(k)$ é a amostra k do segmento-chave x e Z é o tamanho do GoS. Isto é possível graças à estrutura do sinal de ECG (ver a seção 4.2.1) e à etapa de pré-processamento, que aumenta a correlação entre segmentos adjacentes e revela a real estrutura do sinal de ECG.

A informação auxiliar SI_n gerada e os bits de paridade P_{Punct} recebidos do codificador são utilizados pelo decodificador para recuperar os dados originais. Entretanto, a informação auxiliar é apenas útil ao decodificador se um modelo que descreva a sua relação com os dados originais é conhecido. Supondo que a informação auxiliar SI_n seja uma versão corrompida dos segmentos originais, a relação $SI_n = WZS_n + \mathcal{N}$ é válida, onde WZS_n é o segmento de Wyner-Ziv n e $\mathcal{N}$ é conhecido como o "ruído de correlação". Durante os testes, muitos registros foram analisados e descobriu-se que $\mathcal{N}$ comporta-se aproximadamente como um processo laplaciano, como reportado pela maioria dos trabalhos em codificação distribuída de vídeo (*Distributed Video Coding* - DVC) presentes na literatura [1,142,145,146]. Tal resultado foi obtido através da aproximação da diferença entre segmentos preditos e originais por gaussianas generalizadas [147], ou seja, várias curvas foram utilizadas e a que melhor representou a distribuição dos dados foi a laplaciana. Dado isso, a distribuição da diferença entre a informação auxiliar e os dados originais é modelada como

$$f(SI_n(k) - WZS_n(k)) = K.e^{-\alpha|SI_n(k)-WZS_n(k)|}, \tag{6.5}$$

onde K é um fator de escalonamento que faz com que a integral da distribuição seja 1 (de fato, K não é igual a $\frac{\alpha}{2}$, pois as amostras de ECG apresentam uma faixa de valores limitada, ou seja, constituem uma laplaciana truncada). O α deve então ser estimado e transmitido ao decodificador.

Apesar do alfa ser transmitido, o modelo ainda é incompleto, pois o "canal de correlação" varia e não pode ser completamente descrito pela laplaciana. Sendo assim, ainda há necessidade de se enviar os bits de paridade, resultantes do processo de codificação turbo.

O decodificador, com o modelo estatístico mencionado acima, decodifica sucessivamente todos os planos de bits $B_{i,k}$, começando com o mais significativo. Cada plano de bits que já foi decodificado é utilizando na decodificação dos planos seguintes, restringindo a faixa de integração da função de densidade de probabilidade. Por exemplo, se há três planos de bits para decodificar e o primeiro foi recuperado como 1, a decodificação considerará apenas a faixa entre x_{min} e x_{max} delimitada por este mesmo plano. Logo, cada plano de bits restringe ainda mais a área da função de densidade de probabilidade a ser considerada para o próximo.

As probabilidades relacionadas ao plano de bits atual são geradas com base nos planos de bits passados e a relação entre os dados originais e a informação auxiliar. Como se utilizam decodificadores do tipo *soft-input soft-output* [136], é necessário que as probabilidades dos bits sejam repassadas para os mesmos de forma adequada. Entretanto, a equação (6.5) apresenta probabilidades para símbolos e não para bits. Supondo-se que $WZS_n^i(k)$ seja o i-ésimo plano de bits da amostra k do segmento de Wyner-Ziv n de um dado GoS, a probabilidade de que o mesmo seja igual a zero é computada como

$$p(WZS_n^i(k) = 0) = K \sum_{y \in H} e^{-\alpha |SI_n(k) - y|}, \tag{6.6}$$

onde H é composto por todos os valores de k bits, na faixa restrita, que apresentam o plano de bits i igual a zero, como feito em [146].

A cada tentativa de decodificação, o decodificador requisita bits do codificador, relacionados a algum padrão de perfuração, e tenta recuperar o plano de bits corrente. Se o plano de bits não for corretamente reconstruído, o decodificador requisita mais bits de paridade, que completarão o próximo padrão de perfuração. Dado que

os segmentos são quantizados com kb bits, o decodificador pode requisitar, no máximo, kb bits para cada amostra.

Após decodificar todos os planos de bits, o decodificador forma o *stream* de amostras quantizadas Q'_k. A reconstrução é realizada de modo similar ao que é feito em [148]. Há três situações possíveis: a informação auxiliar fica num intervalo abaixo do intervalo quantizado recuperado $q_k(n)$, a informação auxiliar fica num intervalo acima do intervalo quantizado recuperado e informação auxiliar fica no mesmo intervalo recuperado, como mostrado na Figura 6.6. Como o método para a geração da informação auxiliar é um preditor polarizado, os melhores valores de reconstrução podem não ser a informação auxiliar ou os valores extremos do intervalo de quantização, como em [1], ou nem mesmo o centro do intervalo de quantização. Com base nisso, define-se o valor de reconstrução como

$$WZS_n(k) = \beta.SI_n(k) + (1-\beta)RCI_n(k) \tag{6.7}$$

onde $WZS_n(k)$ é o valor reconstruído da amostra k do segmento n, $SI_n(k)$ é a informação auxiliar gerada para a amostra k do segmento n, $RCI_n(k)$ é o centro do intervalo de reconstrução para a amostra k do segmento n e β é a constante de polarização, que é escolhida através treinamento. Nas duas primeiras situações apresentadas, ou seja, quando o intervalo da informação auxiliar fica abaixo ou acima do intervalo recuperado, utiliza-se um mesmo β, para todos os GoSs, que por sua vez é diferente do utilizado quando o intervalo da informação auxiliar é igual ao recuperado.

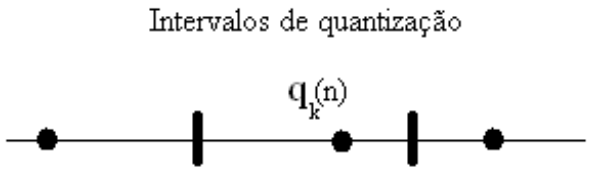


Figura 6.6: Casos possíveis na reconstrução da amostra.

A reconstrução realizada com descrito apresenta resultados melhores que os obtidos com a estratégia em [1]. Após este passo, o ECG sofre então o processamento inverso com relação às técnicas de pré-processamento e fica completamente recuperado, podendo então, por exemplo, ser utilizado para a obtenção de diagnósticos ou figuras de mérito.

6.1.3 O codec no domínio da transformada

O *codec* no domínio da amostra é capaz de apresentar bons resultados, levando-se em consideração que a codificação distribuída geralmente ocasiona uma perda de taxa (ver o capítulo 3) em relação à codificação com exploração de redundâncias apenas no codificador, e ilustra um possível modo de se processar um sinal de ECG utilizando-se conceitos de DSC. Entretanto, é normalmente mais difícil se explorar as redundâncias do sinal no domínio da amostra, dado que amostras adjacentes não estão descorrelacionadas. Um modo de se suplantar este problema reside na utilização de uma transformação de descorrelação, como a transformada discreta do cosseno. Esta é a base do codificador no domínio da transformada (*Transform Domain ECG Encoder* – TDE-ECG) apresentado nesta seção.

Na presente estrutura do TDE-ECG, há outra técnica incluída no bloco de pré-processamento: uma reorganização bidimensional dos segmentos de ECG, como feito na seção 4.2.5. Foi observado que o desempenho do algoritmo de compressão pode ser melhorado através da codificação do sinal de ECG como uma matriz bidimensional, utilizando-se ferramentas eficazes desenvolvidas para a compressão de imagens (ver a seção 4.2.5). Cada período de ECG é então posicionado em uma linha da imagem, que por sua vez é enviada para um bloco de transformação. A transformação de imagens utilizada é a DCT inteira para blocos de dimensões 4×4 definida no padrão de compressão de vídeo H.264 [69], a qual pode ser realizada com aritmética de 16 bits e deslocamentos de bits, sem multiplicações [149]. As matrizes descritoras da transformada estão ilustradas na Figura 6.7. Vale ressaltar que a matriz $I\tilde{DCT}$ não é exatamente a inversa da matriz DCT, pois a sua multiplicação só resulta na matriz identidade I se $I\tilde{DCT} * H * DCT = I$. Logo, a matriz H deve ser utilizada na transformação dos dados ou pode ainda ser incorporada à etapa de quantização.

$$DCT = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \quad H = \begin{bmatrix} \frac{1}{4} & 0 & 0 & 0 \\ 0 & \frac{1}{5} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{5} \end{bmatrix} \quad I\tilde{DCT} = \begin{bmatrix} 1 & 1 & 1 & \frac{1}{2} \\ 1 & \frac{1}{2} & -1 & -1 \\ 1 & -\frac{1}{2} & -1 & 1 \\ 1 & -1 & 1 & -\frac{1}{2} \end{bmatrix}$$

Figura 6.7: Matrizes descritoras da DCT adotada.

Os coeficientes de transformada são agrupados em bandas de coeficientes CB_k,

onde k é o número do coeficiente, conforme ilustrado na Figura 6.8. Cada banda de coeficientes forma um vetor independente, contendo os coeficientes que se localizam na mesma posição (na matriz de dimensões 4×4) de todos os blocos, que é codificado com base nas características de cada banda. Os coeficientes são quantizados com um quantizador uniforme de 2^{M_k} níveis, com $k = 0, 1, 2, \ldots, 15$, e cujo número de intervalos depende do valor absoluto máximo da banda de coeficientes atual. A banda DC (CB_0) é quantizada do mesmo modo que as amostras de ECG no codificador no domínio da amostra. Logo, o passo de quantização é dado por

$$SP = \frac{VDC_{max}}{2^{M_k}}, \tag{6.8}$$

onde SP é o passo de quantização e VDC_{max} é o valor máximo da banda DC. Entretanto, os coeficientes das bandas AC podem apresentar valores sinalizados e são quantizadas com intervalos de quantização simétricos em torno de zero, como mostrado na Figura 6.9, onde $M = 2^{M_k} - 2$, $N = 2^{M_k-1} - 1$ e SP é o passo de quantização. Cada vetor de banda quantizado Q_k é enviado para o bloco de extração de planos de bits. O resto da codificação é idêntico à versão no domínio da amostra. SP, na Figura 6.9, é calculado como

$$SP = \frac{2 * \lceil |VAC_{max}(k)| \rceil}{2^{M_k}}. \tag{6.9}$$

onde $VAC_{max}(k)$ é o valor máximo da banda AC de ordem k. Sendo assim, os valores máximos de cada banda devem ser enviados juntamente com a informação auxiliar, para que os intervalos de quantização possam ser recuperados; as matrizes de quantização, por sua vez, são conhecidas tanto pelo codificador como pelo decodificador. No presente caso, os valores máximos são enviados em formato não codificado, com precisão de 16 bits.

Assim como foi feito para o decodificador no domínio da amostra, o decodificador no domínio da transformada também gera a informação auxiliar a partir dos segmentos-chave de cada GoS, utilizando a mesma interpolação descrita na equação (6.4). Entretanto, os segmentos preditos são rearranjados numa matriz bidimensional e a mesma DCT utilizada no codificador é aplicada. Os coeficientes de transformada preditos e os bits de paridade, juntamente com um modelo de probabilidades laplaciano, são enviados para a máquina de decodificação turbo. Os coeficientes de transformada preditos e originais, como normalmente assumido na

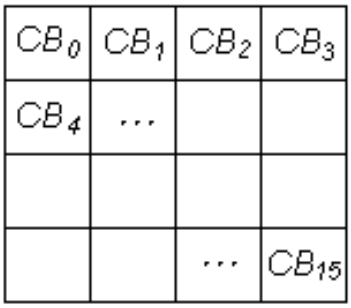


Figura 6.8: Bandas de coeficientes.

literatura [1, 142, 145, 146], apresentam uma relação dada pela distribuição laplaciana (ver a equação (6.5)). O resto da decodificação é realizado como apresentado para o SDE-ECG, inclusive a reconstrução, que é aplicada aos coeficientes de DCT. Os coeficientes de transformada reconstruídos sofrem então a transformação inversa, utilizando-se a matriz $I\tilde{DCT}$ definida em [69] (ver a figura 6.7). Como último passo, a inversão das técnicas de pré-processamento é realizada.

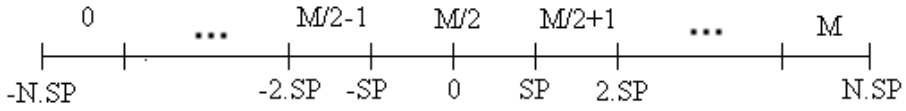


Figura 6.9: Quantização dos coeficientes AC.

O TDE-ECG apresenta uma máquina de codificação muito similar à utilizada pelo SDE-ECG, porém, os dados são fundamentalmente diferentes e as adaptações apropriadas foram descritas acima. É importante notar que a complexidade do TDE-ECG é maior que a apresentada pelo SDE-ECG, porém, é ainda mais baixa que a de codificadores tradicionais representando o estado da arte, que são baseados em esquemas de codificação mais elaborados, executados depois de uma etapa de transformação [4, 105], ou através de uma busca exaustiva no dicionário [10, 96, 97]. Por exemplo, pode-se efetuar uma simples comparação entre o TDE-ECG e o algoritmo proposto em [105]. Como a DCT utilizada no TDE-ECG é do tipo inteira, o que já foi mencionado, realizam-se as operações de transformação sem a necessidade de multiplicações em ponto flutuante, o que acaba ocorrendo com a transformada *wavelet*, que, na melhor das hipóteses, necessita de uma multiplicação por coeficiente [150]. No algoritmo proposto em [105], que utiliza a *wavelet* biortogonal 9/7,

são necessárias, em média, 8 multiplicações por coeficiente. No passo de quantização executado no TDE-ECG, cada coeficiente é dividido por um valor, de acordo com a sua matriz de quantização, o que se traduz em uma divisão/multiplicação por amostra. Após esse passo, a codificação turbo é realizada, a qual envolve permutações e somas módulo-2, o que é bastante rápido. No caso do algoritmo em [105], após a etapa de transformação, realiza-se uma quantização vetorial, com busca em dicionário. Supondo-se o erro médio quadrático como critério de fidelidade, que é um dos mais utilizados, seria necessário aproximadamente uma multiplicação por amostra, para cada elemento do dicionário. Logo, a diferença de complexidade é alta, podendo ser estimada valores maiores que $10:1$. Com relação ao método presente em [3], que utiliza o JPEG2000, a diferença é menor. Supondo-se a transformada *wavelet* irreversível, a complexidade relacionada à *wavelet* é a mesma para o algoritmo em [105], porém, o tratamento dos coeficientes é menos complexo. A quantização é similar à utilizada pelo TDE-ECG, porém, a codificação aritmética aplicada aumenta a complexidade. Dependendo de como é implementado o algoritmo, é possível se estimar a diferença de complexidade em valores maiores que $5:1$. No que diz respeito aos dois compressores de ECG distribuídos, o SDE-ECG e o TDE-ECG, as simulações realizadas mostraram uma diferença de complexidade de aproximadamente $1:2$ a $1:3$ (SDE:TDE). Logo, se o TDE-ECG for utilizado, o *hardware* alvo deve estar preparado para suportar tal aumento.

6.1.4 Resultados de simulações

Para se verificar a eficácia do algoritmo proposto, foram realizados testes com as primeiras 216000 amostras (10 minutos) dos registros 100, 102, 107, 115, 117 e 119 da base de dados de ECGs MIT/BIH. Novamente, esses registros foram escolhidos devido à existência de resultados de compressão para os mesmos na literatura [3, 4, 6, 7] e nas seções 4.2.3, 4.2.4, 4.2.5 e 4.2.6. Os sinais foram amostrados a 360 Hz, com 11 bits de resolução. Dois tamanhos foram assumidos para o GoS: 3 e 5.

Com relação ao SDE-ECG, os níveis de quantização adotados foram 64, 32, 16 e 8, que geram quatro pontos taxa-distorção distintos. Os grupos de segmentos quantizados são então aglomerados, com dez GoSs ($L = 10$) por conjunto, antes de serem enviados ao codificador turbo. O período de perfuração Ni [143] de cada

plano de bits i é dado por

$$N_i = \begin{cases} 32, & i > \lfloor B/2 \rfloor \\ 24, & i \leq \lfloor B/2 \rfloor \end{cases}, \tag{6.10}$$

onde B é o número de planos de bits. O parâmetro α do modelo laplaciano é calculado no codificador, para cada registro, e enviado ao decodificador; o mesmo α é utilizado para toda a seqüência. O parâmetro β, ou seja, a constante de polarização da função de reconstrução, foi analisado para muitos registros, incluindo os de teste, e foi fixado em: 1/2, quando a informação auxiliar reside na mesma faixa de quantização reconstruída, e 1/4 para os outros casos. Os codificadores convolucionais componentes do codificador turbo são idênticos e de taxa 1/2, sendo descritos pela matriz geradora

$$G(D) = \left[\begin{array}{cc} 1 & \frac{g_2(D)}{g_1(D)} \end{array} \right] = \left[\begin{array}{cc} 1 & \frac{1+D+D^3+D^4}{1+D+D^4} \end{array} \right]. \tag{6.11}$$

Após cada tentativa de decodificação do plano de bits atual, o decodificador calcula a probabilidade de erro de bits P_e e verifica se a mesma é menor que 0,001; caso isso não seja verdade, o mesmo pede mais bits do codificador, deslocando-se para o próximo padrão de perfuração. A codificação turbo é executada por 20 iterações.

Para o TDE-ECG, quatro matrizes de quantização diferentes foram definidas, as quais estão ilustradas na Figura 6.10. Logo, há quatro pontos taxa-distorção distintos. Cada banda de coeficientes quantizada CB_k é então enviada para o codificador turbo. O período de perfuração Ni é o mesmo usado pelo SDE-ECG. O parâmetro α do modelo laplaciano é calculado no codificador, para cada banda de coeficientes CB_k, e enviado para o decodificador, assim como os valores máximos de cada banda (devido à quantização). O parâmetro β assumiu os mesmos valores apresentados para o SDE-ECG, também obtidos através de treinamento com vários registros. O resto das características do codificador turbo é idêntico ao adotado para o SDE-ECG.

As dimensões das imagens de ECG resultantes, enviadas para o TDE-ECG, variam bastante e dependem do número de períodos detectados e seu comprimento médio. Além disso, o número de segmentos WZS deve, obrigatoriamente, ser múltiplo de 4. Os registros 100, 102, 107, 115, 117 e 119, para um GoS de tamanho 3, resultaram em imagens de dimensões 761×284, 729×296, 705×308, 633×340,

$$\begin{bmatrix} 32 & 16 & 8 & 8 \\ 16 & 8 & 8 & 4 \\ 8 & 8 & 4 & 0 \\ 8 & 4 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 64 & 32 & 16 & 8 \\ 32 & 16 & 8 & 4 \\ 16 & 8 & 4 & 4 \\ 8 & 4 & 4 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 128 & 64 & 32 & 16 \\ 64 & 32 & 16 & 8 \\ 32 & 16 & 8 & 4 \\ 16 & 8 & 4 & 0 \end{bmatrix} \quad \begin{bmatrix} 256 & 64 & 32 & 32 \\ 64 & 32 & 32 & 16 \\ 32 & 32 & 16 & 8 \\ 32 & 16 & 8 & 4 \end{bmatrix}$$

Figura 6.10: Matrizes de quantização para o TDE-ECG.

505×428 e 657×328, e, para um GoS de tamanho 5, em imagens de dimensões 753×284, 737×296, 705×308, 641×340, 513×428 e 657×328, respectivamente. O comprimento original de cada período, os níveis DC originais, a ordenação original e o nível de equalização são codificados com um codificador aritmético e enviados como informação auxiliar ao decodificador, juntamente com os dados específicos de cada codificador. No caso do TDE-ECG, os segmentos WZS são aglomerados e divididos em blocos de dimensões 4×4, aplicando-se então a DCT. A qualidade dos sinais reconstruídos foi avaliada com a métrica de PRD e a taxa de compressão CR, já apresentadas na seção 4.2.1. As figuras de PRD $\times$ CR levam em consideração apenas os segmentos de Wyner-Ziv. Os segmentos-chave foram codificados sem distorção, visando simplicidade na avaliação de desempenho dos algoritmos, que agem apenas nos segmentos de Wyner-Ziv, como na maioria dos trabalhos para sinais de vídeo presentes na literatura [1, 142]. Além disso, alguns experimentos mostraram que os resultados utilizando segmentos-chave reconstruídos com um bom algoritmo são muito próximos dos com reconstrução ideal, variando levemente para cada registro.

É importante ressaltar que, até o momento da elaboração do presente trabalho, não havia outros métodos de compressão distribuída para sinais de ECG presentes na literatura. Em princípio, os algoritmos elaborados (TDE-ECG e SDE-ECG) são os primeiros representantes desta nova classe de compressores distribuídos para ECG. Isto também leva a outra conclusão: não há resultados disponíveis apenas para os segmentos de Wyner-Ziv. Logo, os resultados para os outros algoritmos levam em consideração todos os batimentos nos primeiros 10 minutos de exame. Entretanto,

experimentos preliminares com os codificadores baseados em MMP mostraram que a diferença nas figuras de PRD ao se avaliar apenas os segmentos de Wyner-Ziv ou todos, para os registros testados, ficou sempre abaixo de $0,2\%$. Sendo assim, ainda é válido comparar o SDE-ECG e o TDE-ECG com outros compressores tradicionais presentes na literatura.

Os resultados estão sumarizados nas Tabelas 6.1, 6.2, 6.3 e 6.4 e nas Figuras 6.11 até 6.13. Percebe-se, dos resultados apresentados para o SDE-ECG, que os mesmos não são muito expressivos, mas ainda podem ser considerados aceitáveis. O motivo para este comportamento reside na falta de ferramentas para a exploração das dependências intra-batimentos. O SDE-ECG processa diretamente as amostras do sinal de ECG de entrada, sem qualquer procedimento de descorrelação ou alguma técnica com o objetivo de tratar a relação entre períodos adjacentes. Entretanto, este *codec* apresenta baixa complexidade e, dependendo das restrições apresentadas pelo dispositivo alvo, pode ser considerada uma solução viável. Vale ressaltar que as figuras de PRD proporcionadas pelo método (as menores que 5%) não chegam a prejudicar o diagnóstico do médico.

O melhor desempenho, levando-se em consideração os métodos distribuídos propostos, foi obtido com o TDE-ECG, que mesmo assim não ficou próximo de todos os métodos em teste. Isso já era esperado, dada a perda de taxa inerente à DSC, que foi abordada na seção 3.2, equação (3.2). Mesmo assim, os resultados são expressivos, pois figuras de PRD próximas às de duas versões do MMP foram obtidas (registro 119), além de também haver um desempenho semelhante ao dos codecs utilizando JPEG2000 e H.264 intra-quadros, para alguns registros (olhar tabelas dos capítulos anteriores). Levando-se em consideração que as figuras de PRD presentes nas Tabelas 6.1 a 6.4 são apenas para os segmentos de Wyner-Ziv, diferentemente dos outros métodos, tal fato indica que a aplicação do paradigma de DSC à compressão de sinais de ECG é promissora e mostra que é possível uma boa aproximação com relação aos resultados dos codificadores tradicionais. Os bons resultados apresentados pelo TDE-ECG são, na maior parte, devidos ao uso da DCT, que proporciona uma melhor exploração das redundâncias intra-batimentos. Apesar de não apresentar um desempenho muito alto, o TDE-ECG pode ser uma alternativa quando a distorção é a característica de maior importância, com a ressalva de que o dispositivo alvo seja

Tabela 6.1: Comparação de desempenho ($PRD \times CR$) entre os algoritmos em [3–7], SDE-ECG e TDE-ECG.

Algoritmo	**Registro**	**CR**	**PRD**
Chou et. al app. 2 [3]	100	24 : 1	4.06
Tipo 3 - JPEG2000 seção 4.2.5	100	24 : 1	3.95
Tipo 1 - H.264 seção 4.2.5	100	24 : 1	3.47
MMP seção 4.2.4	100	24 : 1	3.41
MMP EqDC	100	24 : 1	3.30
SDE-ECG Gos 3	100	26.33 : 1	11.27
SDE-ECG Gos 5	100	25.01 : 1	14.21
TDE-ECG Gos 3	100	29.26 : 1	3.81
TDE-ECG Gos 5	100	23.23 : 1	4.06
Tipo 3 - JPEG2000 seção 4.2.5	100	10 : 1	2.12
Tipo 1 - H.264 seção 4.2.5	100	10 : 1	2.08
MMP seção 4.2.4	100	10 : 1	2.10
MMP EqDC	100	10 : 1	2.03
SDE-ECG Gos 3	100	11.29 : 1	3.10
SDE-ECG Gos 5	100	10.32 : 1	3.98
TDE-ECG Gos 3	100	12.78 : 1	3.01
TDE-ECG Gos 5	100	11.42 : 1	3.25
MMP seção 4.2.4	102	22.49 : 1	3.02
MMP EqDC/SS	102	22.49 : 1	2.49
SDE-ECG Gos 3	102	26.76 : 1	10.18
SDE-ECG Gos 5	102	24.82 : 1	13.16
TDE-ECG Gos 3	102	28.45 : 1	5.28
TDE-ECG Gos 5	102	27.35 : 1	6.47

Tabela 6.2: Comparação de desempenho ($PRD \times CR$) entre os algoritmos em [3–7], SDE-ECG e TDE-ECG (cont.).

Algoritmo	**Registro**	**CR**	**PRD**
Tai et. al [7]	107	10.7 : 1	1.90
MMP seção 4.2.4	107	10.7 : 1	1.70
MMP EqDC	107	10.7 : 1	1.61
SDE-ECG Gos 3	107	13.64 : 1	6.66
SDE-ECG Gos 5	107	13.53 : 1	12.68
TDE-ECG Gos 3	107	9.57 : 1	2.77
TDE-ECG Gos 5	107	9.66 : 1	4.44
Tai et. al [7]	115	30.6 : 1	4.10
MMP seção 4.2.4	115	30.6 : 1	3.20
MMP EqDC/SS	115	30.6 : 1	2.92
SDE-ECG Gos 3	115	26.87 : 1	14.48
SDE-ECG Gos 5	115	24.98 : 1	18.14
TDE-ECG Gos 3	115	36.41 : 1	4.19
TDE-ECG Gos 5	115	39.27 : 1	4.64
Tipo 3 - JPEG2000 seção 4.2.5	117	24 : 1	1.72
Tipo 1 - H.264 seção 4.2.5	117	24 : 1	1.64
MMP seção 4.2.4	117	24 : 1	1.42
MMP EqDC	117	24 : 1	1.26
SDE-ECG Gos 3	117	21.57 : 1	2.17
SDE-ECG Gos 5	117	16.82 : 1	2.65
TDE-ECG Gos 3	117	29.58 : 1	1.62
TDE-ECG Gos 5	117	27.22 : 1	2.06
Chou et. al app. 2 [3]	117	13 : 1	1.18

Tabela 6.3: Comparação de desempenho ($PRD \times CR$) entre os algoritmos em [3–7], SDE-ECG e TDE-ECG (cont.).

Algoritmo	**Registro**	**CR**	**PRD**
Tipo 3 - JPEG2000 seção 4.2.5	117	13 : 1	1.07
Tipo 1 - H.264 seção 4.2.5	117	13 : 1	1.14
MMP seção 4.2.4	117	13 : 1	0.98
MMP EqDC	117	13 : 1	0.91
Wei et. al [6]	117	10 : 1	1.18
Bilgin et. al [4]	117	10 : 1	1.03
Chou et. al app. 1 [3]	117	10 : 1	0.98
Tipo 3 - JPEG2000 seção 4.2.5	117	10 : 1	0.86
Tipo 1 - H.264 seção 4.2.5	117	10 : 1	0.95
MMP seção 4.2.4	117	10 : 1	0.85
MMP EqDC	117	10 : 1	0.79
SDE-ECG Gos 3	117	9.99 : 1	1.42
SDE-ECG Gos 5	117	7.97 : 1	1.73
TDE-ECG Gos 3	117	7.45 : 1	0.98
TDE-ECG Gos 5	117	11.45 : 1	1.61
Bilgin et. al [4]	117	8 : 1	0.86
Tipo 3 - JPEG2000 seção 4.2.5	117	8 : 1	0.75
Tipo 1 - H.264 seção 4.2.5	117	8 : 1	0.81
MMP seção 4.2.4	117	8 : 1	0.75
MMP EqDC	117	8 : 1	0.72
Lee et. al [5]	119	24 : 1	10.5
Bilgin et. al [4]	119	21.6 : 1	3.76
Tai et. al [7]	119	20 : 1	2.17

Tabela 6.4: Comparação de desempenho ($PRD \times CR$) entre os algoritmos em [3–7], SDE-ECG e TDE-ECG (cont.).

Algoritmo	**Registro**	**CR**	**PRD**
Chou et. al app. 2 [3]	119	20.9 : 1	1.81
Tipo 3 - JPEG2000 seção 4.2.5	119	20.9 : 1	1.92
Tipo 3 - H.264 seção 4.2.5	119	20.9 : 1	1.78
MMP seção 4.2.4	119	20.9 : 1	2.00
MMP EqDC/SS	119	20.9 : 1	1.83
SDE-ECG Gos 3	119	21.23 : 1	3.66
SDE-ECG Gos 5	119	20.32 : 1	4.55
TDE-ECG Gos 3	119	20.35 : 1	1.81
TDE-ECG Gos 5	119	29.84 : 1	2.66
Chou et. al app. 2 [3]	119	10 : 1	1.03
Norm. - JPEG2000	119	10 : 1	1.37
Tipo 3 - JPEG2000 seção 4.2.5	119	10 : 1	0.93
Tipo 3 - H.264 seção 4.2.5	119	10 : 1	1.00
MMP seção 4.2.4	119	10 : 1	1.10
MMP EqDC/SS	119	10 : 1	1.07
SDE-ECG Gos 3	119	11.62 : 1	2.38
SDE-ECG Gos 5	119	10.44 : 1	3.23
TDE-ECG Gos 3	119	12.93 : 1	1.60
TDE-ECG Gos 5	119	9.45 : 1	1.69
Tipo 3 - JPEG2000 seção 4.2.5	119	8 : 1	0.74
Tipo 3 - H.264 seção 4.2.5	119	8 : 1	0.83
MMP seção 4.2.4	119	8 : 1	0.96
MMP EqDC/SS	119	8 : 1	0.91

capaz de suportar um aumento na complexidade do codificador.

Comparando-se os resultados obtidos com os da literatura sobre compressão distribuída, verifica-se que a diferença de desempenho entre compressão tradicional e compressão distribuída, para sinais de ECG, parece menor que a apresentada para sinais de vídeo. Tal fato reforça a adequação do paradigma de compressão distribuída a sinais de ECG, que pode gerar soluções adaptadas e sem grandes perdas de desempenho.

6.1.5 Discussão e análise dos resultados

O desenvolvimento dos compressores distribuídos de ECG seguiu a abordagem normalmente adotada na compressão distribuída de sinais: um codificador no domínio da amostra e outro com uma etapa de transformação, implementada com a DCT. Além disso, os *codecs* fizeram uso das técnicas de pré-processamento desenvolvidas nas seções 4.2.4, 4.2.5 e 4.2.6.

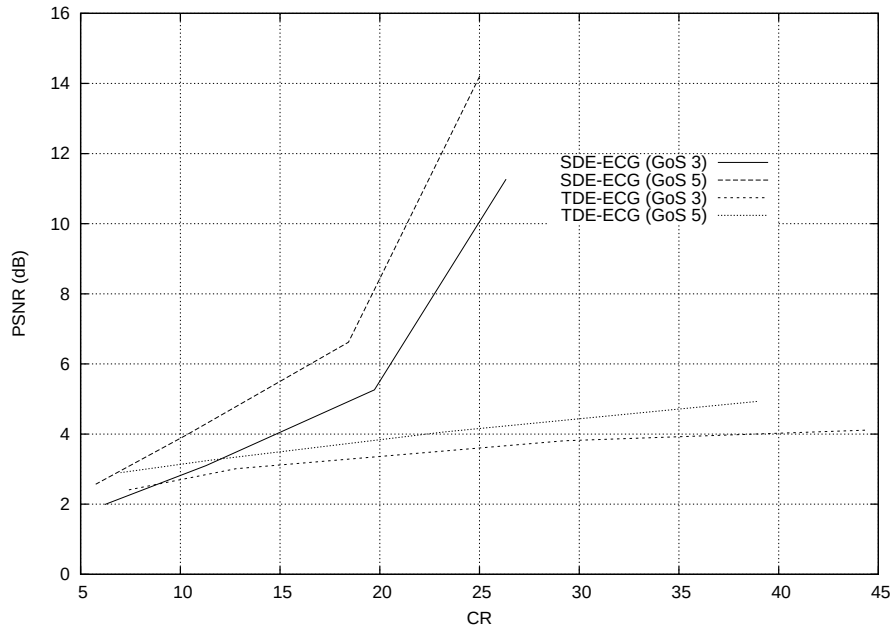
Apesar dos algoritmos funcionarem de maneira satisfatória, ainda há uma característica bastante restritiva, comum à maioria dos algoritmos baseados na abordagem descrita em [1,142]: o canal de *feedback*. Apesar de proporcionar um controle ótimo da taxa de codificação, fazendo com que o codificador se ajuste perfeitamente ao "canal de correlação", necessita-se da presença de tal canal no sistema, o que pode não ser uma realidade. Tentativas de remoção do canal de *feedback* estão presentes na literatura [151,152], porém, não há uma solução comprovada ou modelo genérico. Sendo assim, aplicações como compressão no dispositivo de marca-passo ficam comprometidas. É claro que há a possibilidade de se incorporar um algoritmo similar ao descrito em [152], para o controle de taxa, mas ainda haveria perdas e a taxa de compressão seria menor que a com canal de *feedback*. Há ainda a possibilidade de se complementar a estimativa inicial de controle de taxa com o canal de *feedback*, reduzindo-se o número de requisições [153].

Mesmo com a limitação do canal de *feedback*, ainda seria possível a instalação destes algoritmos de compressão em dispositivos móveis, que monitorariam o paciente durante a sua rotina diária e utilizariam a comunicação bidirecional já existente nesses aparelhos. Com um canal de *feedback* natural, por exemplo via GPRS, o sinal poderia ser enviado e processado *online* pelo decodificador distribuído, que

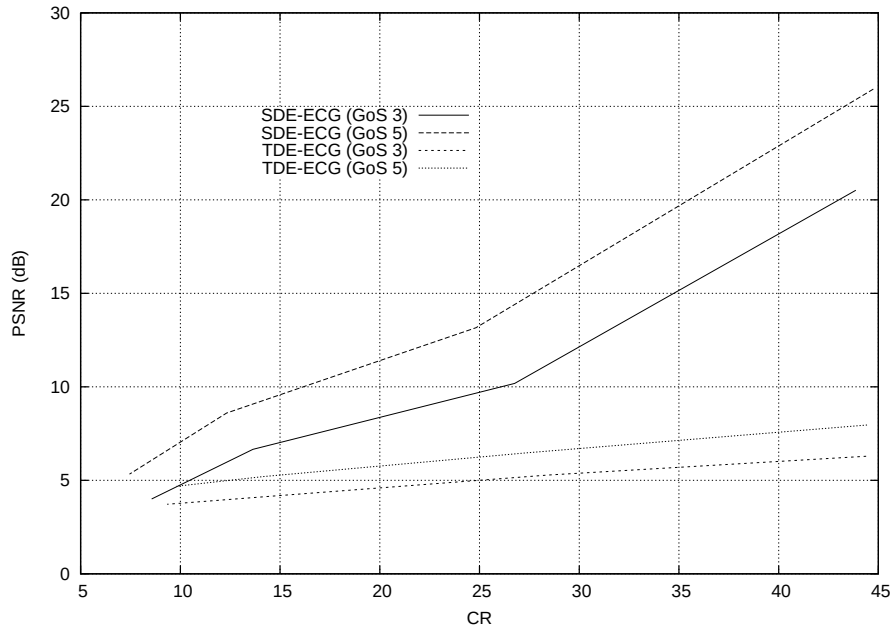
controlaria a cadência de compressão.

Ao se analisar o desempenho dos codificadores distribuídos de ECG, principalmente com relação ao registro 102, percebe-se que os mesmos não apresentam desempenhos comparáveis aos dos demais algoritmos testados. Isso se deve à falta de periodicidade nesse registro (batimentos mais irregulares), que compromete um pouco a eficácia das ferramentas utilizadas. Por outro lado, o bom desempenho apresentado pelo TDE-ECG na compressão do registro 119 é, na maior parte, devido à ordenação por similaridade, que criou três regiões distintas na imagem de ECG (ver Figura 4.36).

Outra característica interessante dos algoritmos desenvolvidos é que há uma certa flexibilidade no processamento efetuado pelo codificador, dada pelo tamanho do GoS. Com um GoS grande, mais segmentos são codificados de acordo com o conceito de Wyner e Ziv, reduzindo a computação necessária para a codificação. Além disso, variando-se o tamanho do GoS, um maior número de pontos taxa-distorção podem ser obtidos. Esta flexibilidade permite maior adaptabilidade ao dispositivo alvo, através de um compromisso entre taxa, distorção e complexidade. Tal esquema é similar ao grupo de quadros (*Group of Pictures* - GoP) empregado em compressores convencionais de vídeo [9, 69], que permite uma variação na razão entre os quadros **I**, **P** e **B**.

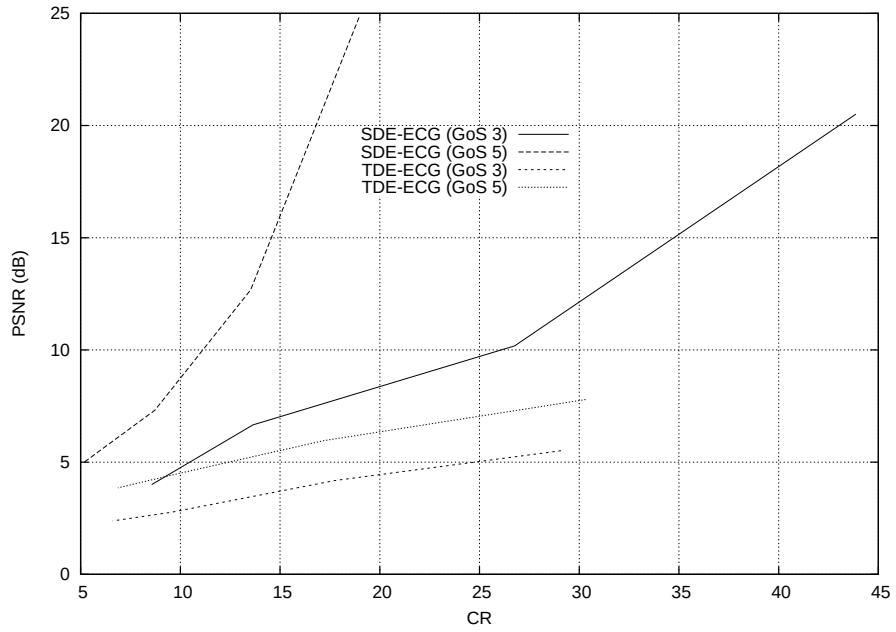


(a)

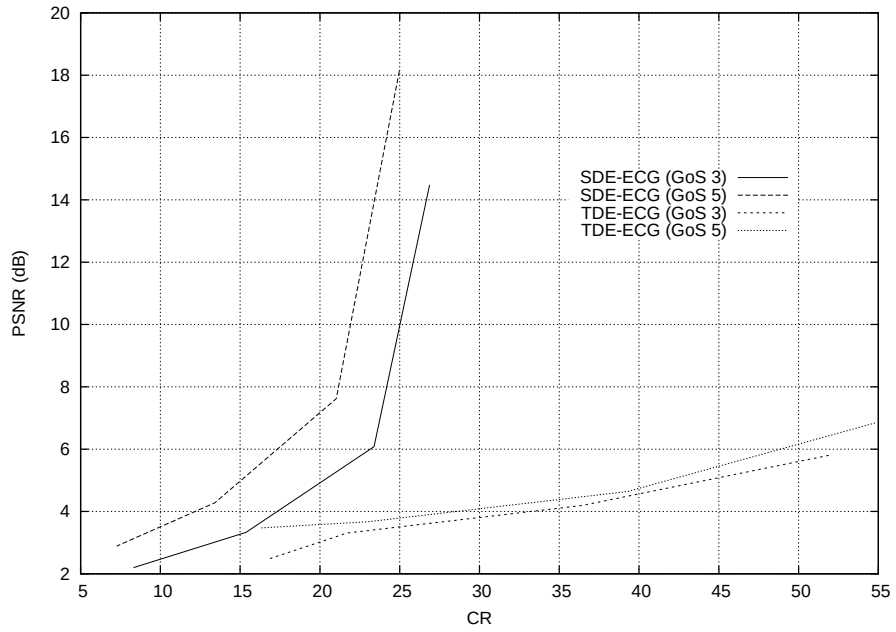


(b)

Figura 6.11: Desempenho na compressão de Registros da base de dados de ECGs MIT/BIH para os *codecs* SDE-ECG e TDE-ECG. (a) Registro 100. (b) Registro 102.

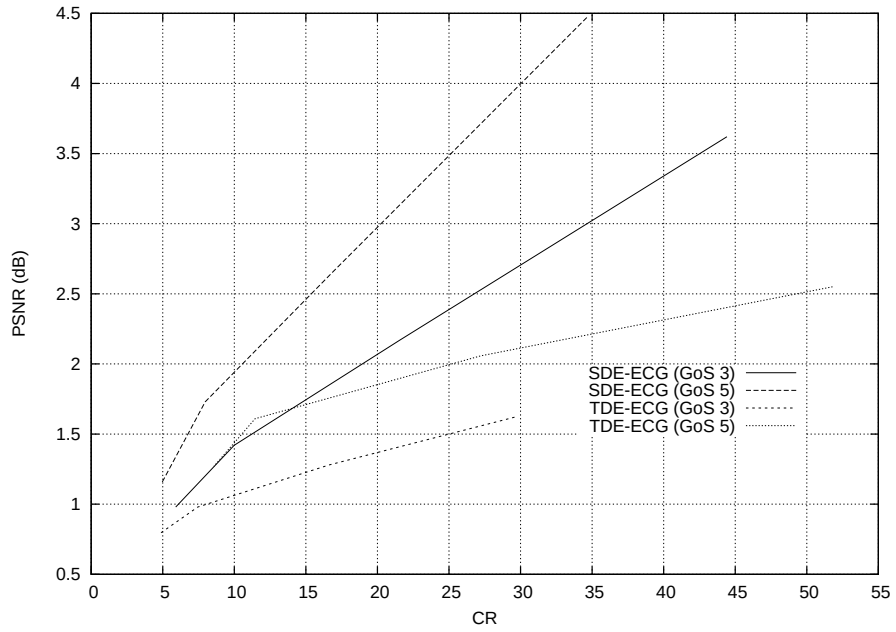


(a)

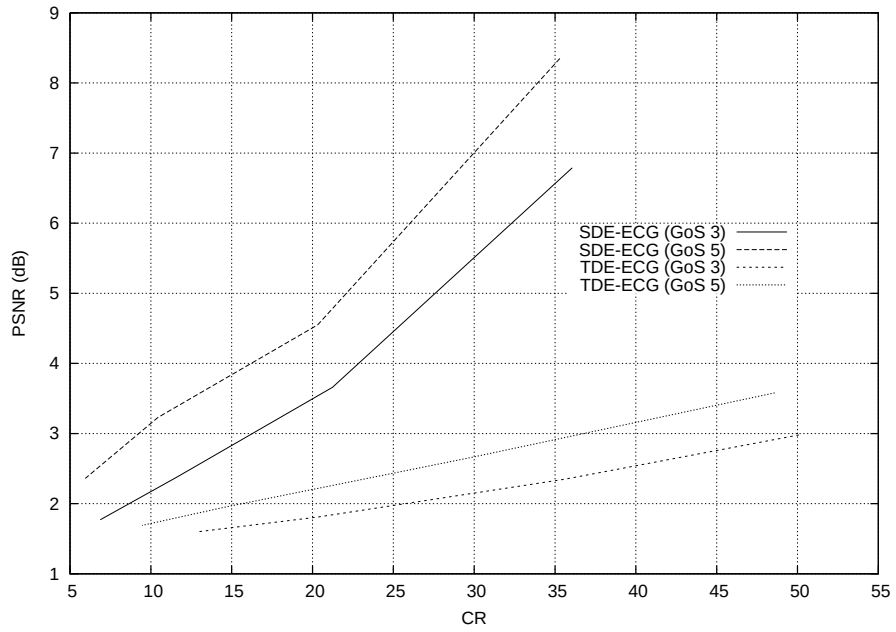


(b)

Figura 6.12: Desempenho na compressão de Registros da base de dados de ECGs MIT/BIH para os *codecs* SDE-ECG e TDE-ECG. (a) Registro 107. (b) Registro 115.



(a)



(b)

Figura 6.13: Desempenho na compressão de Registros da base de dados de ECGs MIT/BIH para os *codecs* SDE-ECG e TDE-ECG. (a) Registro 117. (b) Registro 119.

Capítulo 7

Considerações finais

Foram desenvolvidos novos esquemas de compressão baseados no algoritmo MMP, que são capazes de processar sinais variados, como imagens, ECGs e EMGs, juntamente com novas ferramentas para pré-processamento e modelagem de fonte.

Foi proposto um novo modelo estatístico para fontes, adequado a imagens suaves e imagens contendo texto e gráficos, que foi incorporado ao MMP, permitindo reduzir a taxa necessária para a codificação. Esta modificação, juntamente com a nova forma de codificação dos elementos do dicionário proposta em [154], as novas regras de atualização do dicionário aqui propostas e as modificações na regra de formação do dicionário inicial e na otimização da árvore de segmentação, contribuem para um aumento significativo no desempenho do algoritmo.

O algoritmo final desenvolvido para a compressão de imagens, chamado de APM-MMP, mostrou-se, com relação às imagens suaves, bastante superior aos seus antecessores, mantendo um desempenho igual ou melhor para o resto do espectro testado. A codificação das imagens mais suaves apresentou aumentos na $PSNR$ que chegaram a 1,6 dB (imagem *Lena*), além de sensível aumento da qualidade subjetiva em todas as imagens testadas.

Outro benefício obtido com a adoção do novo modelo de probabilidades foi uma grande redução no efeito de blocagem. Isto se deve ao fato dos blocos serem escolhidos através da similaridade entre seus *pixels* de borda e os de outros blocos já codificados, mantendo-se as estruturas através dos mesmos, o que torna o processamento aplicado ao bloco atual dependente daqueles que já foram codificados. O efeito de blocagem é um problema bastante incômodo no MMP padrão e possui

soluções através de pós-filtragem adaptativa, que não são, entretanto, inerentes ao algoritmo utilizado.

No que diz respeito aos sinais de ECG, novas técnicas de pré-processamento foram desenvolvidas, que formatam o sinal de entrada e revelam a sua estrutura básica que estava oculta, permitindo uma maior exploração das redundâncias intra- e inter-batimentos. Uma grande vantagem dessas técnicas reside na possibilidade de fácil incorporação a praticamente qualquer codificador, seja este um padrão normatizado [69, 70] ou um novo esquema desenvolvido [10].

Os bons resultados obtidos na compressão de sinais de EMG, utilizando-se o algoritmo MMP unidimensional básico, reafirmam o comportamento universal do mesmo, que consiste em um algoritmo capaz de processar até mesmo sinais com característica de ruído. Além disso, coloca-se o MMP como uma possibilidade para o processamento de sinais biológicos, podendo o mesmo ser utilizado também para sinais de EEG.

Neste trabalho, levando-se em consideração a literatura atual, apresentou-se o primeiro esquema de compressão distribuída adequado a sinais de ECG, criando-se uma nova classe de *codecs*, capazes de apresentar uma reversão de complexidade entre codificador e decodificador. Diferentemente da compressão tradicional, a maior carga de computação pode ser movida para o decodificador, possibilitando a incorporação desses algoritmos a dispositivos com pouca capacidade de processamento. Há ainda a possibilidade de compressão de sinais de ECG correlacionados, como os provenientes de derivações diferentes, que pode ser implementada com base nas estruturas já desenvolvidas.

Os objetivos intencionados com a realização deste trabalho foram essencialmente alcançados, pois foram obtidos algoritmos com bom desempenho e comportamento universal, sem a necessidade de heurísticas individualizadas na escolha dos parâmetros. Entretanto, outras melhorias ainda podem ser realizadas, as quais têm o potencial de proporcionar resultados ainda mais significativos que os alcançados, tais como:

- Extender o modelo de probabilidades, procurando-se por novas funções de ponderação capazes de modelar mais adequadamente outros tipos de sinais de entrada, como sinais de EMG;

- Criar um esquema de partição de blocos mais flexível, como, por exemplo, em [68], permitindo ao MMP criar uma árvore de segmentação mais adequada aos dados, o que aumentaria o desempenho taxa-distorção do algoritmo;

- Projetar critérios de continuidade que utilizem mais blocos para o modelo de probabilidades, possibilitando uma predição mais exata dos elementos adequados à codificação do bloco de entrada;

- Pesquisar novos métodos de pré-processamento, proporcionando um desempenho ainda maior do compressor adotado. Por exemplo, poderi-se-iam equalizar os picos do complexo QRS ou mesmo realizar transformações para a redução da faixa dinâmica;

- Na redução da faixa dinâmica, ao invés de se utilizar o segmento médio, poder-se-ia utilizar o último período de ECG reconstruído, permitindo uma referência mais próxima do sinal atual, conforme mostra a Figura 7.1. Esta extensão seria semelhante ao DPCM, mas orientada a períodos de ECG;

- Ainda está aberta a possibilidade de técnicas de pré-processamento para sinais de EMG. Por exemplo, o sinal de entrada aproximadamente gaussiano poderia ser normalizado ou uma decomposição poderia ser realizada, criando-se âncoras para o processamento de segmentos específicos do mesmo;

- Utilizando-se as técnicas e as estruturas empregadas para sinais de ECG e EMG, poder-se-ia criar um esquema adequado a compressão de sinais de EEG. Num esquema universal para sinais biológicos, manter-se-ia a mesma estrutura básica do MMP e selecionar-se-iam as técnicas empregadas num dado momento, de acordo com as especificidades de cada sinal;

- Com relação aos esquemas de compressão distribuída para ECG, o primeiro passo é a realização de um estudo para avaliar a remoção do canal de *feedback*, permitindo a codificação *offline* do sinal;

- A geração de informação lateral ainda é bastante simples, consistindo apenas em uma interpolação. Poder-se-ia, ao invés disso, criar um modelo AR mais complexo ou uma compensação de deslocamento das ondas de ECG.

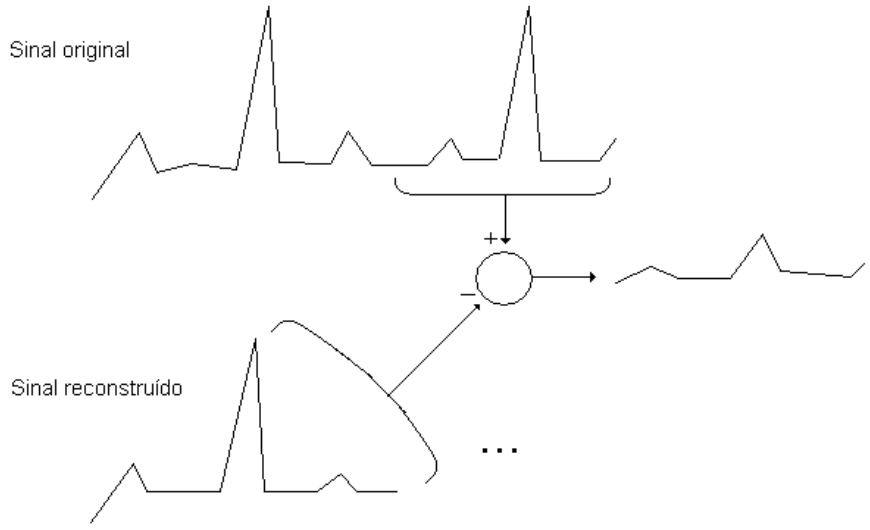


Figura 7.1: Possível geração de resíduos no MMP para a compressão de sinais de ECG, substituindo-se a técnica na seção 4.2.4.

Os algoritmos desenvolvidos com base no MMP provaram ser competitivos, apresentando resultados comparáveis aos dos codificadores baseados em *Wavelets* ou outros representando o estado da arte. Esta superioridade foi obtida por meio da introdução de um modelo de probabilidades para a fonte, que atribui valores diferentes de probabilidade dependendo da vizinhança causal.

As técnicas de pré-processamento, apesar de introduzirem alguma distorção no sinal, compensam com a equalização das estruturas, que aumenta bastante a correlação entre batimentos similares.

Por último, vale a pena ressaltar a importância do desenvolvimento de uma nova classe de codificadores de ECG, baseada em conceitos de DSC, principalmente devido à reversão no paradigma de complexidade e à aplicação a casos de uso ainda não adequadamente abordados, onde a baixa complexidade do codificador é o ponto de maior importância.

Apêndice A

Lista de publicações realizadas durante o desenvolvimento da tese

- Periódicos:

  i) FILHO, E. B. L., DA SILVA, E. A. B., DE CARVALHO, M. B., JÚNIOR, W. S. S., KOILLER, J., "Electrocardiographic Signal Compression using Multiscale Recurrent Patterns", *IEEE Transactions on Circuits and Systems-I: Regular Papers*, v. 52, n. 12, pp. 2739–2753, December 2005.

  ii) FILHO, E. B. L., DA SILVA, E. A. B., DE CARVALHO, M. B., PINAGÉ, F. S., "Universal Image Compression using Multiscale Recurrent Patterns with Adaptive Probability Model", *IEEE Transactions on Image Processing*, v. 17, n. 4, pp. 512–527, April 2008.

  iii) FILHO, E. B. L., RODRIGUES, N. M. M., DA SILVA, E. A. B., DE FARIA, S. M. M., DA SILVA, V. M. M., DE CARVALHO, M. B., "ECG Signal Compression Based on DC Equalization and Correlation Sorting", *IEEE Transactions on Biomedical Engineering*, v. 55, n. 7, pp. 1923–1926, July 2008.

  iv) FILHO, E. B. L., DA SILVA, E. A. B., DE CARVALHO, M. B., "On EMG Signal Compression With Recurrent Patterns", *IEEE Transactions on Biomedical Engineering*, v. 55, n. 7, pp. 1920–1923, July 2008.

  v) FILHO, E. B. L., RODRIGUES, N. M. M., DA SILVA, E. A. B., DE CARVALHO, M. B., DE FARIA, S. M. M., DA SILVA, V. M. M.,

"On ECG Signal Compression with One-dimensional Multiscale Recurrent Patterns Allied to Pre-Processing Techniques", *IEEE Transactions on Biomedical Engineering*, Accepted for future publication.

- Anais de congressos:

  i) FILHO, E. B. L., JÚNIOR, W. S. S., DE CARVALHO, M. B., DA SILVA, E. A. B., "Compressão de Sinais de Eletrocardiograma utilizando Recorrência de Padrões Multiescalas com Critério de Continuidade Interblocos e Segmentação Flexível", In: *Anais do XXII Simpósio Brasileiro de Telecomunicações*, Campinas, SP, Brasil, pp. 968–973, Setembro de 2005.

  ii) JÚNIOR, W. S. S., FILHO, E. B. L., DA SILVA, E. A. B., DE CARVALHO, M. B., MENDONÇA, G. V., "Compressão de Sinais Multidimensionais utilizando Recorrência de Padrões Multiescalas com Segmentação Flexível", In: *Anais do XXII Simpósio Brasileiro de Telecomunicações*, Campinas, SP, Brasil, pp. 114–119, Setembro de 2005.

  iii) FILHO, E. B. L., DA SILVA, E. A. B., JÚNIOR, W. S. S., DE CARVALHO, M. B., "ECG Compression using Multiscale Recurrent Patterns with Period Normalization", In: *Proceedings of the IEEE International Symposium on Circuits and Systems*, Kos, Greece, pp. 1607–1610, May 2006.

  iv) DESSET, C., FILHO, E. B. L., LENOIR, G., "WiMAX Downlink OFDMA Burst Placement for Optimized Receiver Duty-Cycling", In: *Proceedings of the IEEE International Conference on Communications*, Glasgow, Scotland, pp. 5149–5154, June 2007.

  v) FILHO, E. B. L., ABECASSIS, U., JÚNIOR, W. S. S., DA SILVA, E. A. B., DE CARVALHO, M. B., "Casamento Lateral Generalizado Combinado à Recorrência de Padrões Multiescalas: Um Novo Esquema para a Compressão de Imagens", In: *Anais do XXV Simpósio Brasileiro de Telecomunicações*, Recife, PE, Brasil, Setembro de 2007.

  vi) FILHO, E. B. L., ABECASSIS, U., JÚNIOR, W. S. S., DA SILVA, E. A. B., DE CARVALHO, M. B., "Multiscale Recurrent Patterns and Gener-

alised Side-Match Applied to Image Compression", In: *Proceedings of the Picture Coding Symposium*, Lisboa, Portugal, September 2007.

Apêndice B

Imagens de teste

Neste apêndice, são mostradas as imagens de teste originais *Lena*, *Peppers* e *F-16*, de dimensões 512×512, *Cameraman*, *Einstein* e *House*, de dimensões 256×256, e *PP1205* e *PP1209*, também de dimensões 512×512. As imagens *Lena*, *Peppers* e *F-16* foram obtidas no site http://sipi.usc.edu/database. As imagens *Cameraman* e *House* foram obtidas no site http://iie.fing.edu.uy/ense/asign/codif/material.htm e a *Einstein* em http://scien.stanford.edu/labsite/scien_test_images_videos.html. As imagens *PP1205* and *PP1209* são versões digitalizadas, respectivamente, das páginas 1205 and 1209 da revista *IEEE Transactions on Image Processing*, volume 9, número 7, de Julho de 2000. A *PP1205* contém apenas texto e fórmulas, enquanto a *PP1209* é uma composição de imagens em níveis de cinza (duas versões comprimidas da Lena), texto, fórmulas e gráficos. Ambas foram adquiridas a 75 dpi, com o scanner Epson Perfection 1240U, e seus tamanhos eram aproximadamente de 210mm $\times$ 280mm. Estas imagens podem ser obtidas em ftp://ftp.lps.ufrj.br/pub/profs/eduardo/MMP/.

Figura B.1: *Lena* original, 512×512, 8bpp.

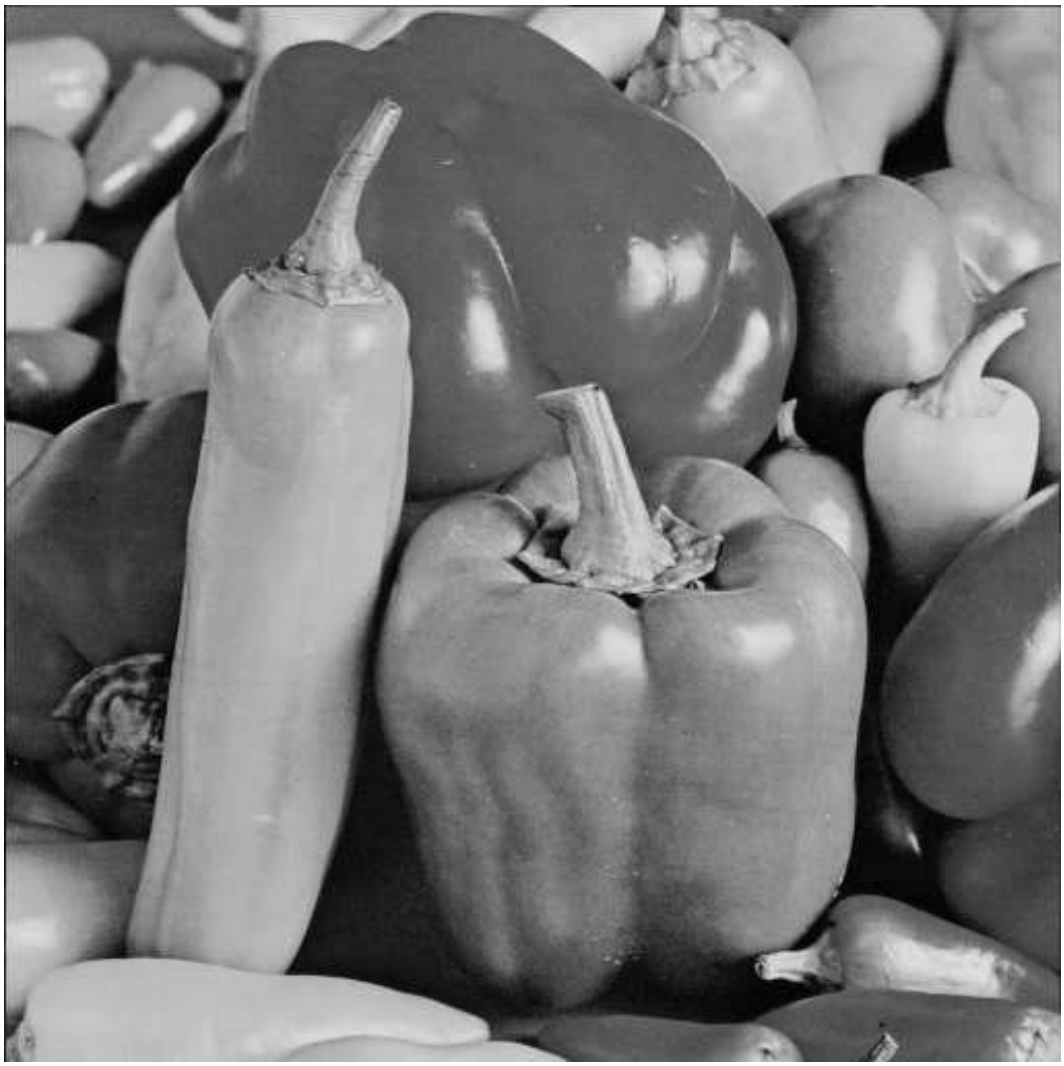


Figura B.2: *Peppers* original, 512×512, 8bpp.

Figura B.3: *F-16* original, 512×512, 8bpp.

Figura B.4: *Cameraman* original, 256×256, 8bpp.

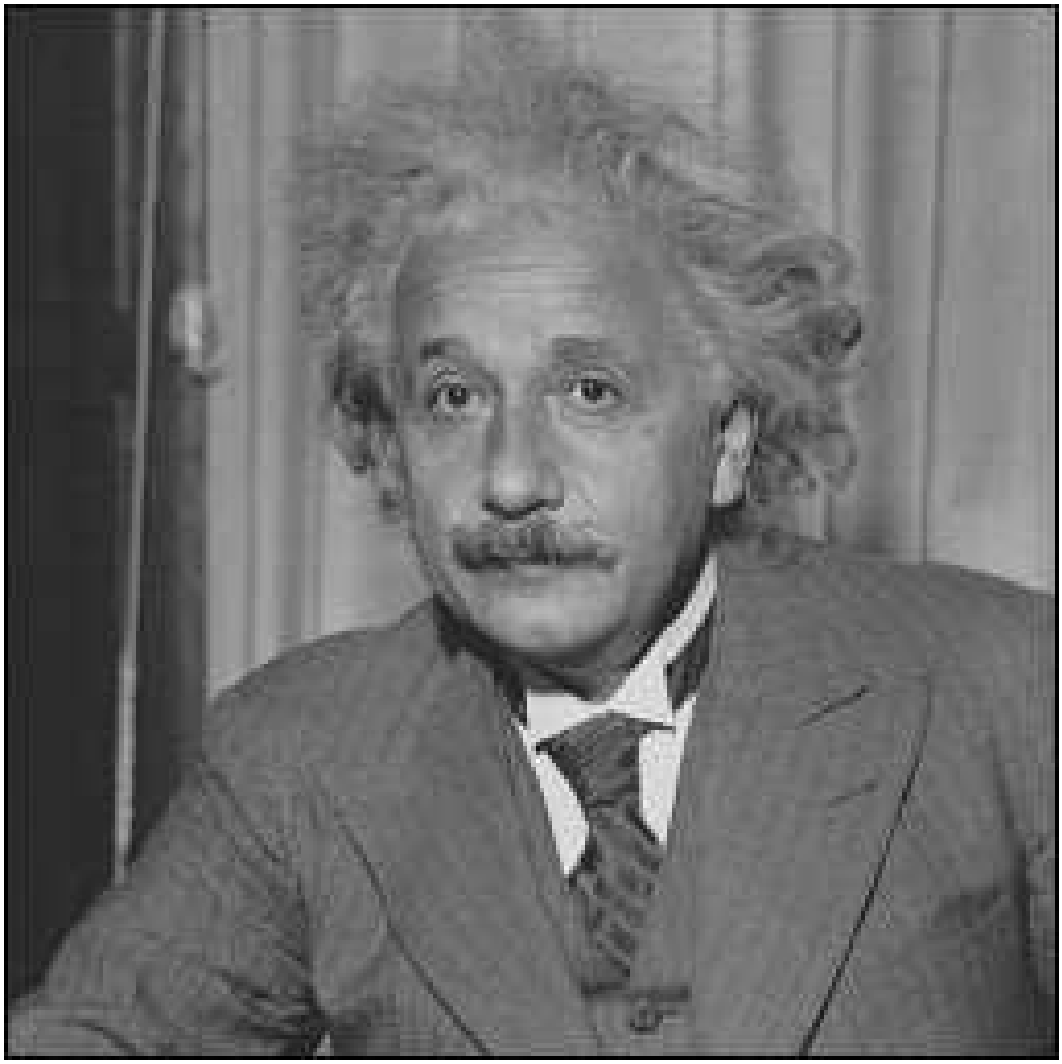
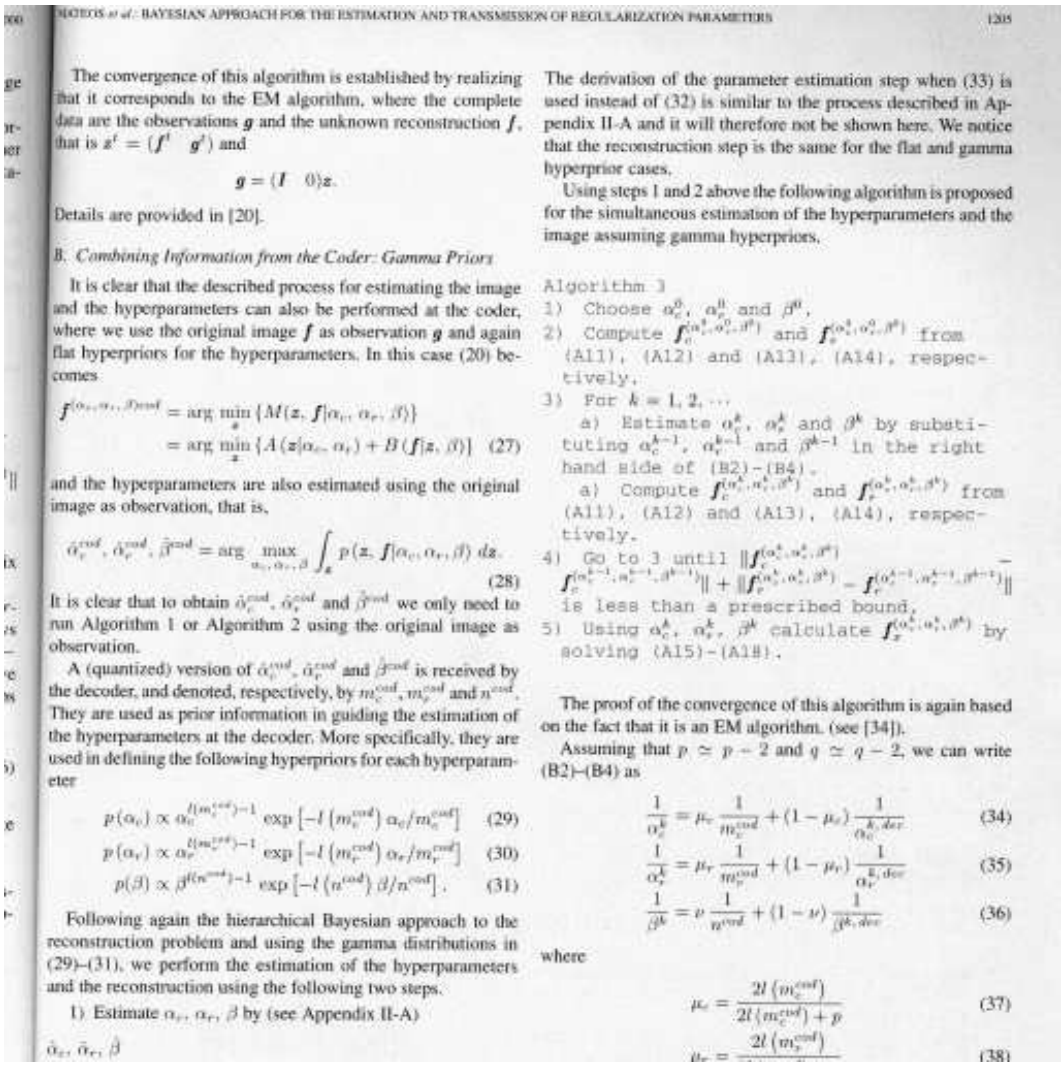


Figura B.5: *Einstein* original, 256×256, 8bpp.

Figura B.6: *House* original, 256×256, 8bpp.



NICOLIS *et al.*: BAYESIAN APPROACH FOR THE ESTIMATION AND TRANSMISSION OF REGULARIZATION PARAMETERS 1205

The convergence of this algorithm is established by realizing that it corresponds to the EM algorithm, where the complete data are the observations $\boldsymbol{g}$ and the unknown reconstruction $\boldsymbol{f}$, that is $\boldsymbol{z}^t = (\boldsymbol{f}^t \quad \boldsymbol{g}^t)$ and

$$\boldsymbol{g} = (\boldsymbol{I} \quad 0)\boldsymbol{z}.$$

Details are provided in [20].

B. Combining Information from the Coder: Gamma Priors

It is clear that the described process for estimating the image and the hyperparameters can also be performed at the coder, where we use the original image $\boldsymbol{f}$ as observation $\boldsymbol{g}$ and again flat hyperpriors for the hyperparameters. In this case (20) becomes

$$\begin{aligned}\boldsymbol{f}^{(\alpha_c,\alpha_r,\beta)cod} &= \arg\min_{\boldsymbol{z}} \{M(\boldsymbol{z}, \boldsymbol{f}|\alpha_c, \alpha_r, \beta)\} \\ &= \arg\min_{\boldsymbol{z}} \{A(\boldsymbol{z}|\alpha_c, \alpha_r) + B(\boldsymbol{f}|\boldsymbol{z}, \beta)\} \quad (27)\end{aligned}$$

and the hyperparameters are also estimated using the original image as observation, that is,

$$\hat{\alpha}_c^{cod}, \hat{\alpha}_r^{cod}, \hat{\beta}^{cod} = \arg\max_{\alpha_c,\alpha_r,\beta} \int_{\boldsymbol{z}} p(\boldsymbol{z}, \boldsymbol{f}|\alpha_c, \alpha_r, \beta)\, d\boldsymbol{z}. \quad (28)$$

It is clear that to obtain $\hat{\alpha}_c^{cod}$, $\hat{\alpha}_r^{cod}$ and $\hat{\beta}^{cod}$ we only need to run Algorithm 1 or Algorithm 2 using the original image as observation.

A (quantized) version of $\hat{\alpha}_c^{cod}$, $\hat{\alpha}_r^{cod}$ and $\hat{\beta}^{cod}$ is received by the decoder, and denoted, respectively, by m_c^{cod}, m_r^{cod} and n^{cod}. They are used as prior information in guiding the estimation of the hyperparameters at the decoder. More specifically, they are used in defining the following hyperpriors for each hyperparameter

$$p(\alpha_c) \propto \alpha_c^{l(m_c^{cod})-1} \exp\left[-l\left(m_c^{cod}\right)\alpha_c/m_c^{cod}\right] \quad (29)$$

$$p(\alpha_r) \propto \alpha_r^{l(m_r^{cod})-1} \exp\left[-l\left(m_r^{cod}\right)\alpha_r/m_r^{cod}\right] \quad (30)$$

$$p(\beta) \propto \beta^{l(n^{cod})-1} \exp\left[-l\left(n^{cod}\right)\beta/n^{cod}\right]. \quad (31)$$

Following again the hierarchical Bayesian approach to the reconstruction problem and using the gamma distributions in (29)–(31), we perform the estimation of the hyperparameters and the reconstruction using the following two steps.

1) Estimate α_c, α_r, β by (see Appendix II-A)

$\hat{\alpha}_c$, $\hat{\alpha}_r$, $\hat{\beta}$

The derivation of the parameter estimation step when (33) is used instead of (32) is similar to the process described in Appendix II-A and it will therefore not be shown here. We notice that the reconstruction step is the same for the flat and gamma hyperprior cases.

Using steps 1 and 2 above the following algorithm is proposed for the simultaneous estimation of the hyperparameters and the image assuming gamma hyperpriors.

Algorithm 3

1) Choose α_c^0, α_r^0 and β^0,
2) Compute $\boldsymbol{f}_c^{(\alpha_c^0,\alpha_r^0,\beta^0)}$ and $\boldsymbol{f}_r^{(\alpha_c^0,\alpha_r^0,\beta^0)}$ from (A11), (A12) and (A13), (A14), respectively.
3) For $k = 1, 2, \cdots$
   a) Estimate α_c^k, α_r^k and β^k by substituting α_c^{k-1}, α_r^{k-1} and β^{k-1} in the right hand side of (B2)–(B4).
   a) Compute $\boldsymbol{f}_c^{(\alpha_c^k,\alpha_r^k,\beta^k)}$ and $\boldsymbol{f}_r^{(\alpha_c^k,\alpha_r^k,\beta^k)}$ from (A11), (A12) and (A13), (A14), respectively.
4) Go to 3 until $\|\boldsymbol{f}_c^{(\alpha_c^k,\alpha_r^k,\beta^k)} - \boldsymbol{f}_c^{(\alpha_c^{k-1},\alpha_r^{k-1},\beta^{k-1})}\| + \|\boldsymbol{f}_r^{(\alpha_c^k,\alpha_r^k,\beta^k)} - \boldsymbol{f}_r^{(\alpha_c^{k-1},\alpha_r^{k-1},\beta^{k-1})}\|$ is less than a prescribed bound.
5) Using α_c^k, α_r^k, β^k calculate $\boldsymbol{f}_x^{(\alpha_c^k,\alpha_r^k,\beta^k)}$ by solving (A15)–(A18).

The proof of the convergence of this algorithm is again based on the fact that it is an EM algorithm. (see [34]).

Assuming that $p \simeq p-2$ and $q \simeq q-2$, we can write (B2)–(B4) as

$$\frac{1}{\alpha_c^k} = \mu_c \frac{1}{m_c^{cod}} + (1-\mu_c)\frac{1}{\alpha_c^{k,dev}} \quad (34)$$

$$\frac{1}{\alpha_r^k} = \mu_r \frac{1}{m_r^{cod}} + (1-\mu_r)\frac{1}{\alpha_r^{k,dev}} \quad (35)$$

$$\frac{1}{\beta^k} = \nu \frac{1}{n^{cod}} + (1-\nu)\frac{1}{\beta^{k,dev}} \quad (36)$$

where

$$\mu_c = \frac{2l\left(m_c^{cod}\right)}{2l\left(m_c^{cod}\right)+p} \quad (37)$$

$$\mu_r = \frac{2l\left(m_r^{cod}\right)}{\quad} \quad (38)$$

Figura B.7: *PP1205* original, 512×512, 8bpp.

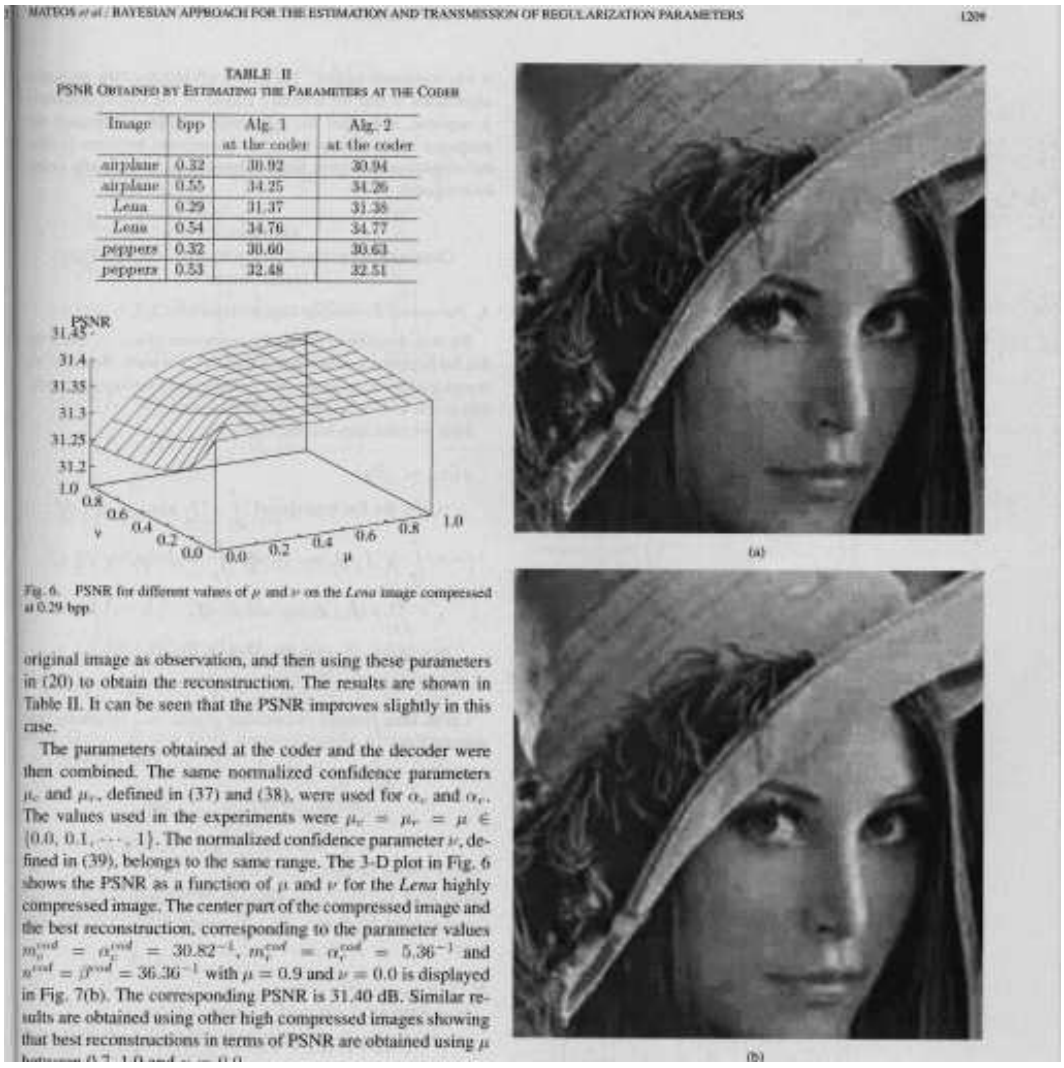


Figura B.8: *PP1209* original, 512×512, 8bpp.

Referências Bibliográficas

[1] GIROD, B., AARON, A. M., RANE, S., et al., "Distributed Video Coding", *Proceedings of the the IEEE*, v. 93, n. 1, pp. 71–83, January 2005.

[2] AARON, A. M., RANE, S., GIROD, B., "Wyner-Ziv Video Coding with Hash-Based Motion Compensation at the Receiver". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 3097–3100, Singapore, October 2004.

[3] CHOU, H.-H., CHEN, Y.-J., SHIAU, Y.-C., et al., "An Effective and Efficient Compression Algorithm for ECG Signals with Irregular Periods", *IEEE Transactions on Biomedical Engineering*, v. 53, n. 6, pp. 1198–1205, June 2006.

[4] BILGIN, A., MARCELLIN, M. W., ALTBACH, M. I., "Compression of Electrocardiogram Signals using JPEG2000", *IEEE Transactions on Consumer Electronics*, v. 49, n. 4, pp. 833–840, November 2003.

[5] LEE, H., BUCKLEY, K. M., "ECG Data Compression using Cut and Align Beats Approach and 2-D Transforms", *IEEE Transactions on Biomedical Engineering*, v. 46, n. 5, pp. 556–565, May 1999.

[6] WEI, J.-J., CHANG, C.-J., CHOU, N.-K., et al., "ECG Data Compression using Truncated Singular Value Decomposition", *IEEE Transactions on Information Technology in Biomedicine*, v. 5, n. 4, pp. 290–299, December 2001.

[7] TAI, S.-C., SUN, C.-C., TAN, W.-C., "2-D ECG Compression Method Based on Wavelet Transform and Modified SPIHT", *IEEE Transactions on Biomedical Engineering*, v. 52, n. 6, pp. 999–1008, June 2005.

[8] SAYOOD, K., *Introduction to Data Compression*. Morgan Kaufmann: San Francisco, CA, USA, 2005.

[9] MITCHELL, J. L., PENNEBAKER, W. B., FOGG, C. E., et al., *MPEG Video Compression Standard*. Kluwer Academic Publishers, 2001.

[10] DE CARVALHO, M. B., DA SILVA, E. A. B., FINAMORE, W. A., "Multidimensional Signal Compression using Multiscale Recurrent Patterns", *Signal Processing: Image and Video Coding beyond Standards*, v. 82, n. 11, pp. 1559–1580, November 2002.

[11] FILHO, E. B. L., DE CARVALHO, M. B., DA SILVA, E. A. B., "Compressão de Sinais Multidimensionais utilizando Recorrência de Padrões Multiescalas com Critério de Continuidade Interblocos". In: *Anais do XXI Simpósio Brasileiro de Telecomunicações*, Belém, PA, Brasil, Setembro de 2004.

[12] FILHO, E. B. L., DE CARVALHO, M. B., DA SILVA, E. A. B., "Multidimensional Signal Compression using Multi-Scale Recurrent Patterns with smooth Side-Match Criterion". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 3201–3204, Singapore, October 2004.

[13] DUARTE, M. H. V., DE CARVALHO, M. B., DA SILVA, E. A. B., et al., "Multiscale Recurrent Patterns Applied to Stereo Image Coding", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 15, n. 11, pp. 1434–1447, November 2005.

[14] RODRIGUES, N. M. M., DA SILVA, E. A. B., DE CARVALHO, M. B., et al., "Improving H.264/AVC Inter Compression with Multiscale Recurrent Patterns". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 1353–1356, Atlanta, GA, USA, October 2006.

[15] RODRIGUES, N. M. M., DA SILVA, E. A. B., DE CARVALHO, M. B., et al., "Basis Optimisation for Multiscale Recurrent Pattern Coding". In: *Proceedings of the 3rd International Workshop on Mathematical Techniques and Problems in Telecommunications*, pp. 1353–1356, Leiria, Portugal, September 2006.

[16] COVER, T. M., THOMAS, J. A., *Elements of Information Theory*. Wiley-Interscience, July 2006.

[17] XIONG, Z., LIVERIS, A. D., CHENG, S., "Distributed Source Coding for Sensor Networks", *IEEE Signal Processing Magazine*, v. 21, n. 5, pp. 80–94, September 2004.

[18] PRADHAN, S., KUSUMA, J., RAMCHANDRAN, K., "Distributed Compression in a Dense Microsensor Network", *IEEE Signal Processing Magazine*, v. 19, n. 2, pp. 51–60, March 2002.

[19] SLEPIAN, D., WOLF, J. K., "Noiseless Coding of Correlated Information Sources", *IEEE Transactions on Information Theory*, v. 19, n. 4, pp. 471–480, July 1973.

[20] WYNER, A., ZIV, J., "The Rate-Distortion Function for Source Coding with Side Information at the Decoder", *IEEE Transactions on Information Theory*, v. 22, n. 1, pp. 1–10, January 1976.

[21] WYNER, A., "On Source Coding with Side Information at the Decoder", *IEEE Transactions on Information Theory*, v. 21, n. 3, pp. 294–300, May 1975.

[22] WYNER, A., "The Rate-Distortion Function for Source Coding with Side Information at the Decoder-II: General Sources", *Information and Control*, v. 38, n. 1, pp. 60–80, July 1978.

[23] WYNER, A., "Recent Results in the Shannon Theory", *IEEE Transactions on Information Theory*, v. 20, n. 1, pp. 2–10, January 1974.

[24] ZHAO, Y., GARCIA-FRIAS, J., "Data Compression of Correlated Non-Binary Sources using Punctured Turbo Codes". In: *Proceedings of the IEEE Data Compression Conference*, pp. 242–251, Snowbird, UT, USA, March 2002.

[25] LIVERIS, A. D., XIONG, Z., GEORGHIADES, C., "Compression of Binary Sources with Side Information at the Decoder using LDPC Codes", *IEEE Communications Letters*, v. 6, n. 10, pp. 440–442, October 2002.

[26] PRADHAN, S. S., RAMCHANDRAN, K., "Distributed Source Coding using Syndromes (DISCUS): Design and Construction". In: *Proceedings of the IEEE Data Compression Conference*, pp. 158–167, Snowbird, UT, USA, March 1999.

[27] WANG, X., ORCHARD, M., "Design of Trellis Codes for Source Coding with Side Information at the Decoder". In: *Proceedings of the IEEE Data Compression Conference*, pp. 361–370, Snowbird, UT, USA, March 2001.

[28] AARON, A. M., GIROD, B., "Compression with Side Information using Turbo Codes". In: *Proceedings of the IEEE Data Compression Conference*, pp. 252–261, Snowbird, UT, USA, March 2002.

[29] SCHONBERG, D., PRADHAN, S. S., RAMCHANDRAN, K., "LDPC Codes can Approach the Slepian-Wolf Bound for General Binary Sources". In: *Proceedings of the 40th Allerton Conference on Communication, Control, and Computing*, pp. 576–585, Monticello, IL, USA, October 2002.

[30] LAN, C. F., LIVERIS, A. D., NARAYANAN, K., et al., "Slepian-Wolf Coding of Multiple M-ary Sources using LDPC Codes". In: *Proceedings of the IEEE Data Compression Conference*, p. 549, Snowbird, UT, USA, March 2004.

[31] PRADHAN, S. S., RAMCHANDRAN, K., "Distributed Source Coding: Symmetric Rates and Applications to Sensor Networks". In: *Proceedings of the IEEE Data Compression Conference*, pp. 363–372, Snowbird, UT, USA, March 2000.

[32] PRADHAN, S. S., CHOU, J., RAMCHANDRAN, K., "Duality Between Source Coding and Channel Coding and its Extension to the Side Information Case", *IEEE Transactions on Information Theory*, v. 49, n. 5, pp. 1181–1203, May 2003.

[33] SERVETTO, S. D., "Lattice Quantization with Side Information". In: *Proceedings of the IEEE Data Compression Conference*, pp. 510–519, Snowbird, UT, USA, March 2000.

[34] XIONG, Z., LIVERIS, A. D., CHENG, S., et al., "Nested Quantization and Slepian-Wolf Coding: a Wyner-Ziv Coding Paradigm for I.I.D. Sources". In: *Proceedings of the IEEE Workshop on Statistical Signal Processing*, pp. 399–402, St. Louis, MO, USA, September 2003.

[35] LIU, Z., CHENG, S., LIVERIS, A. D., et al., "Slepian-Wolf Coded Nested Quantization (SWC-NQ) for Wyner-Ziv Coding: Performance Analysis and Code Design". In: *Proceedings of the IEEE Data Compression Conference*, pp. 322–331, Snowbird, UT, USA, March 2004.

[36] REBOLLO-MONEDERO, D., ZHANG, R., GIROD, B., "Design of Optimal Quantizers for Distributed Source Coding". In: *Proceedings of the IEEE Data Compression Conference*, pp. 13–22, Snowbird, UT, USA, March 2003.

[37] REBOLLO-MONEDERO, D., AARON, A. M., GIROD, B., "Transforms for High-Rate Distributed Source Coding". In: *Proceedings of the Asilomar Conference on Signals, Systems, and Computers*, pp. 850–854, Pacific Grove, CA, USA, November 2003.

[38] AARON, A. M., ZHANG, R., GIROD, B., "Wyner-Ziv Coding of Motion Video". In: *Proceedings of the Asilomar Conference on Signals, Systems, and Computers*, pp. 240–244, Pacific Grove, CA, USA, November 2002.

[39] AARON, A. M., RANE, S., ZHANG, R., et al., "Wyner-Ziv Coding for Video: Applications to Compression and Error Resilience". In: *Proceedings of the IEEE Data Compression Conference*, pp. 93–102, Snowbird, UT, USA, March 2003.

[40] AARON, A. M., SETTON, E., GIROD, B., "Toward Practical Wyner-Ziv Coding of Video". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 869–872, Barcelona, Spain, October 2003.

[41] AARON, A. M., RANE, S., SETTON, E., et al., "Transform-Domain Wyner-Ziv Codec for Video". In: *Proceedings of the SPIE Visual Communications and Image Processing Conference*, v. 5308, pp. 520–528, San Jose, CA, USA, January 2004.

[42] PURI, R., RAMCHANDRAN, K., "PRISM: A New Robust Video Coding Architecture Based on Distributed Compression Principles". In: *Proceedings of the 40th Allerton Conference on Communication, Control, and Computing*, Alleton, IL, USA, October 2002.

[43] XIAO, J.-J., RIBEIRO, A., LUO, Z.-Q., et al., "Distributed Compression-Estimation using Wireless Sensor Networks", *IEEE Signal Processing Magazine*, v. 23, n. 4, pp. 27–41, July 2006.

[44] PURI, R., MAJUMBAR, A., ISHWAR, P., et al., "Distributed Video Coding in Wireless Sensor Networks", *IEEE Signal Processing Magazine*, v. 23, n. 4, pp. 94–106, July 2006.

[45] XIONG, Z., CABARCAS, F., "Approaching the Slepian-Wolf Boundary using Practical Channel Codes", *Signal Processing*, v. 86, n. 11, pp. 3096–3101, November 2006.

[46] TAN, P., TIFFANY, J. L., "A general and Optimal Framework to Achieve the Entire Rate Region for Slepian-Wolf Coding", *Signal Processing*, v. 86, n. 11, pp. 3102–3114, November 2006.

[47] ZHAO, Y., GARCIA-FRIAS, J., "Turbo Compression/Joint Source-Channel Coding of Correlated Binary Sources with Hidden Markov Correlation", *Signal Processing*, v. 86, n. 11, pp. 3115–3122, November 2006.

[48] VARODAYAN, D., AARON, A. M., GIROD, B., "Rate-Adaptive Codes for Distributed Source Coding", *Signal Processing*, v. 86, n. 11, pp. 3123–3130, November 2006.

[49] LAJNEF, K., GUILLEMOT, C., SIOHAN, P., "Distributed Coding of Three Binary and Gaussian Correlated Sources using Punctured Turbo Codes", *Signal Processing*, v. 86, n. 11, pp. 3131–3149, November 2006.

[50] ZHAO, Y., ZHONG, W., GARCIA-FRIAS, J., "Transmission of Correlated Senders Over a Rayleigh Fading Multiple Access Channel", *Signal Processing*, v. 86, n. 11, pp. 3150–3159, November 2006.

[51] REBOLLO-MONEDERO, D., RANE, S., AARON, A. M., et al., "High-Rate Quantization and Transform Coding with Side Information at the Decoder", *Signal Processing*, v. 86, n. 11, pp. 3160–3179, November 2006.

[52] CHEUNG, N.-M., TANG, C., ORTEGA, A., et al., "Efficient Wavelet-Based Predictive Slepian-Wolf Coding for Hyperspectral Imagery", *Signal Processing*, v. 86, n. 11, pp. 3180–3195, November 2006.

[53] TAGLIASACCHI, M., MAJUMBAR, A., RAMCHANDRAN, K., et al., "Robust Wireless Video Multicast Based on a Distributed Source Coding Approach", *Signal Processing*, v. 86, n. 11, pp. 3196–3211, November 2006.

[54] XU, Q., STANKOVIC, V., XIONG, Z., "Layered Wyner-Ziv Video Coding for Transmission over Unreliable Channels", *Signal Processing*, v. 86, n. 11, pp. 3212–3225, November 2006.

[55] ZAMIR, R., SHAMAI, S., EREZ, U., "Nested Linear/Lattice Codes for Structured Multiterminal Binning", *IEEE Transactions on Information Theory*, v. 48, n. 6, pp. 1250–1276, June 2002.

[56] RANE, S., AARON, A. M., GIROD, B., "Error-resilient Video Transmission using Multiple Embedded Wyner-Ziv Descriptions". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 666–669, Genova, Italy, September 2005.

[57] MAJUMBAR, A., PURI, R., ISHWAR, P., et al., "Complexity/Performance Trade-offs for Robust Distributed Video Coding". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 678–681, Genova, Italy, September 2005.

[58] XU, Q., STANKOVIC, V., LIVERIS, A. D., et al., "Distributed Joint Source-channel Coding of Video". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 674–677, Genova, Italy, September 2005.

[59] CHEUNG, N.-M., WANG, H., ORTEGA, A., "Correlation Estimation for Distributed Source Coding under Information Exchange Constraints". In:

Proceedings of the IEEE International Conference on Image Processing, pp. 682–685, Genova, Italy, September 2005.

[60] SUN, J., LI, H., "A Wyner-Ziv Coding Approach to Transmission of Interactive Video over Wireless Channels". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 686–689, Genova, Italy, September 2005.

[61] GEHRIG, N., DRAGOTTI, P. L., "Different - Distributed and Fully Flexible Image Encoders for Camera Sensor Networks". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 690–693, Genova, Italy, September 2005.

[62] TAGLIASACCHI, M., TUBARO, S., SARTI, A., "Combining MCTF with Distributed Source Coding". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 797–800, Genova, Italy, September 2005.

[63] LI, Z., DELP, E. J., "Wyner-Ziv Video Side Estimator: Conventional Motion Search Methods Revisited". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 825–828, Genova, Italy, September 2005.

[64] ARTIGAS, X., TORRES, L., "Iterative Generation of Motion-Compensated Side Information for Distributed Video Coding". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 833–836, Genova, Italy, September 2005.

[65] NONNIS, A., GRANGETTO, M., MAGLI, E., et al., "Improved Low-complexity Intraband Lossless Compression of Hyperspectral Images by Means of Slepian-Wolf Coding". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 829–832, Genova, Italy, September 2005.

[66] FORNEY, G. D., "Coset Codes - Part I: Introduction and Geometrical Classification", *IEEE Transactions on Information Theory*, v. 34, n. 5, pp. 1123–1151, September 1988.

[67] FORNEY, G. D., "Coset Codes - Part II: Binary Lattices and Related Codes", *IEEE Transactions on Information Theory*, v. 34, n. 5, pp. 1152–1187, September 1988.

[68] FRANCISCO, N. C., RODRIGUES, N. M. M., DA SILVA, E. A. B., et al., "Multiscale Recurrent Pattern Image Coding With a Flexible Partition Scheme". In: *Proceedings of the IEEE International Conference on Image Processing*, San Diego, CA, USA, October 2008.

[69] WIEGAND, T., SULLIVAN, G. J., BJØNTEGAARD, G., et al., "Overview of the H.264/AVC Video Coding Standard", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 13, n. 7, pp. 560–576, July 2003.

[70] TAUBMAN, D. S., MARCELLIN, M. W., *JPEG2000: Image Compression Fundamentals, Standards, and Practice*. Kluwer Academic Publishers, 2001.

[71] SULLIVAN, G. J., BAKER, R. L., "Efficient Quadtree Coding of Images and Video", *IEEE Transactions on Image Processing*, v. 3, n. 3, pp. 327–331, May 1994.

[72] DENN, M. M., *Optimization by Variational Methods*. McGraw-Hill Book Company, 1969.

[73] RAMCHANDRAN, K., VETTERLI, M., "Best Wavelet Packet Bases in a Rate Distortion Sense", *IEEE Transactions on Image Processing*, v. 2, n. 2, pp. 160–175, February 1993.

[74] WITTEN, I. H., NEAL, R. M., CLEARY, J. G., "Arithmetic Coding for Data Compression", *Communications of the ACM*, v. 30, n. 6, pp. 520–540, June 1987.

[75] ZIV, J., LEMPEL, A., "Compression of Individual Sequences via Variable-Rate Coding", *IEEE Transactions on Information Theory*, v. 24, n. 5, pp. 530–536, September 1978.

[76] BERROU, C., GLAVIEUX, A., THITIMAJSHIMA, P., "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes". In: *Proceedings of*

the IEEE International Conference on Communications, pp. 1064–1070, Geneva, Zwiterland, May 1993.

[77] RYAN, W. E., "An Introduction to LDPC Codes", chap. CRC Handbook for Coding and Signal Processing for Recording Systems, CRC Press, 2004.

[78] ROWITCH, D., MILSTEIN, L., "On the Performance of Hybrid FEC/ARQ Systems using Rate Compatible Punctured Turbo Codes", *IEEE Transactions on Communications*, v. 48, n. 6, pp. 948–959, June 2000.

[79] COSTA, M., "Writing on Dirty Paper", *IEEE Transactions on Information Theory*, v. 29, n. 3, pp. 439–441, May 1983.

[80] KASNER, J. H., MARCELLIN, M. W., HUNT, B. R., "Universal Trellis Coded Quantization", *IEEE Transactions on Image Processing*, v. 8, n. 12, pp. 1677–1687, December 1999.

[81] MURESAN, D., EFFROS, M., "Quantization as Histogram Segmentation: Globally Optimal Scalar Quantizer Design in Network Systems". In: *Proceedings of the IEEE Data Compression Conference*, pp. 302–311, Snowbird, UT, USA, March 2002.

[82] REBOLLO-MONEDERO, D., ZHANG, R., GIROD, B., "Design of Optimal Quantizers for Distributed Source Coding". In: *Proceedings of the IEEE Data Compression Conference*, pp. 13–22, Snowbird, UT, USA, March 2003.

[83] EYUBOGLU, M. V., FORNEY, G. D., "Lattice and Trellis Quantization with Lattice- and Trellis-Bounded Codebooks – High-Rate Theory for Memoryless Sources", *IEEE Transactions on Information Theory*, v. 39, n. 1, pp. 46–59, January 1993.

[84] VAN DER VLEUTEN, R. J., WEBER, J. H., "Construction and Evaluation of Trellis-Coded Quantizers for Memoryless Sources", *IEEE Transactions on Information Theory*, v. 41, n. 3, pp. 853–859, May 1995.

[85] FRAGOULI, C., WESEL, R. D., SOMMER, D., et al., "Turbo Codes with Non-Uniform Constellations". In: *Proceedings of the IEEE International Conference on Communications*, pp. 11–15, Helsinki, Finland, June 2001.

[86] RAPHAELI, D., GUREVITZ, A., "Constellation Shaping for Pragmatic Turbo Coded Modulation", *Electronics Letters*, v. 38, n. 14, pp. 717–719, July 2002.

[87] CONWAY, J. H., SLOANE, N. J. A., *Sphere Packings, Lattices and Groups*. Springer Verlag: New York, NY, USA, 1988.

[88] SUN, Y., LIVERIS, A. D., STANKOVIC, V., et al., "Near-Capacity Dirty-Paper Code Designs Based on TCQ and IRA Codes". In: *Proceedings of the International Symposium on Information Theory*, pp. 184–188, Adelaide, Australia, September 2005.

[89] HONG, D., VAN DER SCHAAR, M., PESQUET-POPESCU, B., "Arithmetic Coding with Adaptive Context-Tree Weighting for the H.264 Video Coders". In: *Proceedings of the SPIE Visual Communications and Image Processing Conference*, v. 5308, pp. 1226–1235, San Jose, CA, USA, January 2004.

[90] WU, X., MEMON, N., "Context-Based Lossless Interband Compression - Extending CALIC", *IEEE Transactions on Image Processing*, v. 9, n. 6, pp. 994–1001, June 2000.

[91] MARPE, D., SCHWARZ, H., WIEGAND, T., "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 13, n. 7, pp. 620–636, July 2003.

[92] BLAHUT, R. A., *Principles and Practice of Information theory*. Addison-Wesley publishing Company: Cambridge, MA, USA, 1988.

[93] FILHO, E. B. L., *Compressão de Imagens utilizando Recorrência de Padrões Multiescalas com Critério de Continuidade Inter-Blocos*, Tese de mestrado, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, Abril de 2004.

[94] WEI, H. C., TSAI, P. C., WANG, J. S., "Three-Sided Side Match Finite-State Vector Quantization", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 10, n. 1, pp. 51–58, February 2000.

[95] KIM, T., "Side Match and Overlap Match Vector Quantizers for Images", *IEEE Transactions on Image Processing*, v. 1, n. 2, pp. 170–185, February 1992.

[96] YANG, S. B., TSENG, L. Y., "Smooth Side-Match Classified Vector Quantizer with Variable Block Size", *IEEE Transactions on Image Processing*, v. 10, n. 5, pp. 677–685, May 2001.

[97] YANG, S. B., "General-Tree-Structured Vector Quantizer for Image Progressive Coding Using the Smooth Side-Match Method", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 13, n. 2, pp. 193–202, February 2003.

[98] ZAGHETTO, A., DE QUEIROZ, R. L., "Segmentation-Driven Compound Document Coding Based on H.264/AVC-INTRA", *IEEE Transactions on Image Processing*, v. 16, n. 7, pp. 1755–1760, July 2007.

[99] ZAGHETTO, A., DE QUEIROZ, R. L., "MRC Compression of Compound Documents using H.264/AVC-I". In: *Proceedings of the XXV Simpósio Brasileiro de Telecomunicações*, Recife, PE, Brazil, September 2007.

[100] KLABUNDE, R. E., *Cardiovascular Physiology Concepts*. Lippincott Williams & Wilkins: Philadelphia, PA, USA, 2004.

[101] MALMIVUO, J., PLONSEY, R., *Bioelectromagnetism - Principles and Applications of Bioelectric and Biomagnetic Fields*. Oxford University Press: New York, NY, USA, 1995.

[102] JALALEDDINE, S., HUTCHENS, C., STRATTAN, R., et al., "ECG Data Compression Techniques - A Unified Approach", *IEEE Transactions on Biomedical Engineering*, v. 37, n. 4, pp. 329–343, April 1990.

[103] NAVE, G., COHEN, A., "ECG Compression using Long Term Prediction", *IEEE Transactions on Biomedical Engineering*, v. 40, n. 9, pp. 877–885, September 1993.

[104] LU, Z., KIM, D. Y., PEARLMAN, W. A., "Wavelet Compression of ECG Signals by the Set Partitioning in Hierarchical Trees Algorithm", *IEEE Transactions on Biomedical Engineering*, v. 47, n. 7, pp. 849–856, July 2000.

[105] MIAOU, S. G., YEN, H. L., LIN, C. L., "Wavelet-Based ECG Compression using Dynamic Vector Quantization with Tree Codevectors in Single Codebook", *IEEE Transactions on Biomedical Engineering*, v. 49, n. 7, pp. 671–680, July 2002.

[106] MIAOU, S. G., CHAO, S. N., "Wavelet-Based Lossy-to-Lossless ECG Compression in a Unified Vector Quantization Framework", *IEEE Transactions on Biomedical Engineering*, v. 52, n. 3, pp. 539–543, March 2005.

[107] SHUKLA, R., DRAGOTTI, P. L., DO, M. N., et al., "Rate-Distortion Optimized Tree-Structured Compression Algorithms for Piecewise Polynomial Images", *IEEE Transactions on Image Processing*, v. 14, n. 3, pp. 343–359, March 2005.

[108] ABOY, M., CRESPO, C., MCNAMES, J., et al., "A Biomedical Signal Processing Toolbox". In: *Proceedings of the 16th International EURASIP Conference BIOSIGNAL 2002*, v. 16, pp. 49–52, Brno, Czech Republic, June 2002.

[109] VAIDYANATHAN, P. P., *Multirate Systems and Filter Banks*. Prentice-Hall Inc.: Englewood Cliffs, NJ, USA, 1993.

[110] LEHMAN, T. M., GONNER, C., SPITZER, K., "Survey: Interpolation Methods in Medical Image Processing", *IEEE Transactions on Medical Imaging*, v. 18, n. 11, pp. 1049–1075, November 1999.

[111] LEE, J., JEONG, K., YOON, J., et al., "A Simple Real-Time QRS Detection Algorithm". In: *Proceedings of the 18th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, v. 4, pp. 1396–1398, Amsterdam, Netherlands, October 1996.

[112] CHEN, S., CHEN, H., CHAN, H., "A Real-Time QRS Detection Method Based on Moving-Averaging Incorporating with Wavelet Denoising", *Computer Methods and Programs in Biomedicine*, v. 82, n. 3, pp. 187–195, June 2006.

[113] KADAMBE, S., MURRAY, R., BOUDREAUX-BARTELS, G. F., "Wavelet Transform-Based QRS Complex Detector", *IEEE Transactions on Biomedical Engineering*, v. 46, n. 7, pp. 838–848, July 1999.

[114] *http://www.kakadusoftware.com*, acessado em Novembro de 2007.

[115] *http://iphome.hhi.de/suehring/tml/index.htm*, acessado em Novembro de 2007.

[116] DE LUCA, C. J., "Physiology and Mathematics of Myoelectric Signals", *IEEE Transactions on Biomedical Engineering*, v. 26, n. 6, pp. 313–325, June 1979.

[117] GUERRERO, A. P., MAILHES, C., "On the Choice of an Electromyogram Data Compression Method". In: *Proceedings of the 19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, v. 4, pp. 1558–1561, Chicago, Illinois, USA, October 1997.

[118] POZZO, M., BOTTIN, A., FERRABONE, R., et al., "Sixty-Four Channel Wearable Acquisition System for Long-Term Surface Electromyogram Recording with Electrode Arrays", *Medical and Biological Engineering and Computing*, v. 42, n. 4, pp. 455–466, July 2004.

[119] PLEVIN, E., ZAZULA, D., "Decomposition of Surface EMG Signals using Non-Linear LMS Optimisation of Higher-Order Cumulants". In: *Proceedings of the 15th IEEE Symposium on Computer-Based Medical Systems*, pp. 149–154, Maribor, Slovenia, June 2002.

[120] CAREY, R. M., CLANCY, E. A., "EMG Decomposition Annotation Comparison Method". In: *Proceedings of the 15th IEEE 31st Annual Northeast Bioengineering Conference*, pp. 100–101, Hoboken, NJ, USA, April 2005.

[121] FARINA, D., MERLETTI, R., ENOKA, R. M., "The Extraction of Neural Strategies from the Surface EMG", *Journal of Applied Physiology*, v. 96, n. 4, pp. 1486–1495, April 2004.

[122] HOSHINO, T., TOMONO, M., FURUSAWA, R., et al., "Development of a Motion Support System by using an Electromyogram". In: *Proceedings of the IEEE International Conference on Systems, Man and Cyberneticss*, v. 5, pp. 4432–4437, The Hague, The Netherlands, October 2004.

[123] BERGER, P. A., NASCIMENTO, F. A. O., CARMO, J. C., et al., "Compression of EMG Signals with Wavelet Transform and Artificial Neural Networks", *Institute of Physics Publishing: Physiological Measurement*, v. 27, n. 6, pp. 457–465, June 2006.

[124] NAZARPOUR, K., SHARAFAT, A. R., FIROOZABADI, S. M., "Negentropy Analysis of Surface Electromyogram Signal". In: *Proceedings of the 13th IEEE Workshop on Statistical Signal Processing*, pp. 974–977, Bordeaux, France, July 2005.

[125] NIKOLIC, M., *Detailed Analysis of Clinical Electromyography Signals EMG Decomposition: Findings and Firing Pattern Analysis in Controls and Patients with Myopathy and Amytrophic Lateral Sclerosis*, Ph.D. Thesis, Faculty of Health Science, University of Copenhagen, 2001.

[126] NORRIS, J. A., ENGLEHART, K., LOVELY, D., "Steady-State and Dynamic Myoelectric Signal Compression using Embedded Zero-Tree Wavelets". In: *Proceedings of the 23th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 1879–1882, October 2001.

[127] CAROTTI, E. S. G., DE MARTIN, J. C., MERLETTI, R., et al., "Compression of Surface EMG Signals with Algebraic Code Excited Linear Prediction", *Medical Engineering & Physics*, v. 29, n. 2, pp. 253–258, May 2006.

[128] CAROTTI, E. S. G., DE MARTIN, J. C., FARINA, D., et al., "Linear Predictive Coding of Myoelectric Signals". In: *Proceedings of the IEEE In-*

ternational Conference on Acoustics, Speech, and Signal Processing, v. 5, pp. 629–632, Philadelphia, PA, USA, March 2005.

[129] MERLETTI, R., BALESTRA, G., KNAFLITZ, M., "Effect of FFT Based Algorithms on Estimation of Myoelectric Signal Spectral Parameters". In: *Proceedings of the 11th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, v. 3, pp. 1022–1023, Seattle, WA, USA, November 1989.

[130] MERLETTI, R., GULISASHVILI, A., LO CONTE, L. R., "Estimation of Shape Characteristics of Surface Muscle Signal Spectra from Time Domain Data", *IEEE Transactions on Biomedical Engineering*, v. 42, n. 8, pp. 769–776, August 1995.

[131] STANDRIDGE, R. K., KONDRASKE, G. V., MOONEY, V., et al., "Temporal Characterization of Myoelectric Spectral Moment Changes: Analysis of Common Parameters", *IEEE Transactions on Biomedical Engineering*, v. 35, n. 10, pp. 789–797, October 1988.

[132] RYAN, W. E., "A Turbo Code Tutorial", Unpublished paper available at http://citeseer.ist.psu.edu/ryan97turbo.html.

[133] HAGENAUER, J., "The Turbo Principle: Tutorial Introduction and State of the Art". In: *Proceedings of the International Symposium on Turbo Codes and Related Topics*, pp. 1–11, Brest, France, September 1997.

[134] SHANNON, C. E., "A Mathematical Theory of Communication", *Bell System Technical Journal*, v. 27, n. 4, pp. 379–423, July 1948.

[135] HAYKIN, S., *Communications Systems*. John Wiley & Sons Inc.: New York, NY, USA, May 2000.

[136] BAHL, L., COCKE, J., JELINEK, F., et al., "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate", *IEEE Transactions on Information Theory*, v. 20, n. 2, pp. 284–287, March 1974.

[137] GALLAGER, R., "Low-Density Parity-Check Codes", *IRE Transactions on Information Theory*, v. 8, pp. 21–28, January 1962.

[138] TANNER, R. M., "A Recursive Approach to Low Complexity Codes", *IEEE Transactions on Information Theory*, v. 27, n. 5, pp. 533–547, September 1981.

[139] MACKAY, D., NEAL, R., "Good Codes Based on Very Sparse Matrices". In: *Proceedings of the 5th IMA Conference on Cryptography and coding*, pp. 100–111, Springer-Verlag: Berlin, Germany, 1995.

[140] MACKAY, D., "Good Error-Correcting Codes Based on Very Sparse Matrices", *IEEE Transactions on Information Theory*, v. 45, n. 2, pp. 399–431, March 1999.

[141] CHUNG, S.-Y., FORNEY, G. D., RICHARDSON, T. J., et al., "On the Design of Low-Density Parity-Check Codes within 0.0045 dB of the Shannon Limit", *IEEE Communications Letters*, v. 5, n. 2, pp. 58–60, February 2001.

[142] BRITES, C., ASCENSO, J., PEREIRA, F., "Improving Transform Domain Wyner-Ziv Video Coding Performance". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Toulouse, France, May 2006.

[143] WICKER, S. B., *Error Control Systems for Digital Communication and Storage*. Prentice Hall, 1994.

[144] DIVSALAR, D., POLLARA, F., "Turbo Codes for PCS Applications". In: *Proceedings of the IEEE International Conference on Communications*, pp. 54–59, Seattle, WA, USA, June 1995.

[145] ASCENSO, J., BRITES, C., PEREIRA, F., "Improving Frame Interpolation with Spatial Motion Smoothing for Pixel Domain Distributed Video Coding". In: *Proceedings of the 5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Services*, Smolenice, Slovakia, June 2005.

[146] DALAI, M., LEONARDI, R., PEREIRA, F., "Improving Turbo Codec Integration in Pixel-Domain Distributed Video Coding". In: *Proceedings of*

the IEEE International Conference on Acoustics, Speech and Signal Processing, Toulouse, France, may 2006.

[147] SHARIFI, K., LEON-GARCIA, A., "Estimation of Shape Parameter for Generalized Gaussian Distributions in Subband Decompositions of Video", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 5, n. 1, pp. 52–56, February 1995.

[148] VATIS, Y., KLOMP, S., OSTERMAN, J., "Enhanced Reconstruction of the Quantised Transform Coefficients for Wyner-Ziv Coding". In: *Proceedings of the IEEE International Conference on Multimedia and Expo*, pp. 172–175, Beijing, China, july 2007.

[149] FAN, C. P., "Fast 2-Dimensional 4×4 Forward Integer Transform Implementation for H.264/AVC", *IEEE Transactions on Circuits and Systems – II: Express Briefs*, v. 53, n. 3, pp. 174–177, March 2006.

[150] XIONG, Z., RAMCHANDRAN, K., ORCHARD, M. T., et al., "A Comparative Study of DCT- and Wavelet-Based Image Coding", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 9, n. 5, pp. 692–695, August 1999.

[151] MORBÉE, M., PRADES-NEBOT, J., PIZURICA, A., et al., "Rate Allocation Algorithm for Pixel-Domain Distributed Video Coding Without Feedback Channel". In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, v. 1, pp. 521–524, Hawaii, USA, April 2007.

[152] BELKOURA, Z. M., SIKORA, T., "Towards Rate-Decoder Complexity Optimisation in Turbo-Coder Based Distributed Video Coding". In: *Proceedings of the Picture Coding Symposium*, Beijing, China, April 2006.

[153] GUILLEMOT, C., PEREIRA, F., TORRES, L., et al., "Distributed Monoview and Multiview Video Coding", *IEEE Signal Processing Magazine*, v. 24, n. 5, pp. 67–76, September 2007.

[154] PINAGÉ, F. S., *Avaliação do Desempenho de Algoritmos de Compressão de Imagens Usando Recorrência de Padrões Multiescalas*, Tese de mestrado, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, Julho de 2005.

[155] WANG, Y., OSTERMANN, J., ZHANG, Y., *Video Processing and Communications*. Prentice-Hall: Englewood Cliffs, NJ, USA, 2002.