QUANTIZAÇÃO VETORIAL ADAPTATIVA MULTIESCALAS COM OTIMIZAÇÃO TAXA-DISTORÇÃO

Murilo B. de Carvalho, Eduardo A. B. da Silva e Weiler Alves Finamore

Resumo - Neste artigo nós propomos um novo método de compressão com perdas para sinais multidimensionais baseado em recorrência de padrões multiescalas chamado MMP (Multidimensional Multiscale Parser). Neste método, um sinal multidimensional é recursivamente segmentado em vetores de comprimento variável, sendo cada segmento codificado usando expansões e contrações de vetores em um dicionário. O dicionário é continuamente atualizado durante a codificação usando concatenações de versões expandidas e contraídas de vetores previamente codificados. Este processo requer apenas a transmissão da estrutura da árvore de segmentação e dos índices dos vetores do dicionário utilizados, dispensando qualquer informação lateral para atualizar o dicionário no decodificador. A segmentação é feita de um modo otimizado segundo um critério de taxa-distorção. Uma versão bidimensional do algoritmo proposto foi implementada e testada com diversos tipos de imagens digitalizadas. Os resultados mostram que o nosso método de atualização de dicionário é eficaz na adaptação do MMP à diversos tipos diferentes de imagens, concedendo ao algoritmo um carater universal. Para imagens compostas de texto e fórmulas, o MMP supera o algoritmo SPIHT por mais de 3dB a uma taxa de 0.5 bpp, enquanto que para imagens mistas, combinando texto, gráficos e imagens em tons de cinza, o desempenho do MMP superou o do SPIHT por cerca de 1.5 dB na mesma taxa. Nós concluímos o artigo com uma análise teórica do processo de casamento aproximado de padrões multiescalas para o caso de fontes Gaussianas, que nos fornece uma indicação de que este processo de compressão pode ser uma boa escolha, especialmente a baixas taxas de bits.

Palavras-chave: Casamento de padrões recorrentes, decomposição multiescalas, compressão de sinais multidimensionais, quantização vetorial.

Abstract - In this paper we propose a new multidimensional lossy signal compression method based on multiscale recurrent patterns, referred to as MMP (Multidimensional Multiscale Parser). In it, a multidimensional signal is recursively segmented into variable-length vectors, and each segment is encoded using expansions and contractions of vectors in a dictionary. The dictionary is updated while the data is being encoded, using concatenations of expanded and contracted versions of previously encoded vectors. The only data encoded are the segmentation tree and the indexes of the vectors in the dictionary, and therefore no side information is necessary for the dictionary updating. The signal segmentation is carried out through a rate-distortion optimization procedure. A two-dimensional version of the MMP algorithm was implemented and tested with several kinds of image data. We have observed that the proposed dictionary updating procedure is effective in adapting the algorithm to a large variety of image content, lending to it a universal flavor. For text and graphics images, it outperforms the state-of-the-art SPIHT algorithm by more that 3dB at 0.5bpp, while for mixed document images, containing text, graphics and grayscale images, by more than 1.5dB at the same rate. We conclude the paper with a theoretical analysis of the approximate matching of gaussian vectors using scales, which gives a justification of why approximate multiscale matching is a good option, specially at low rates.

Keywords: Recurrent pattern matching, multiscale decomposition, multidimensional signal compression, vector quantization.

1. INTRODUÇÃO

A maioria dos métodos que representam o estado-da-arte em compressão de imagens e vídeo utilizam uma estrutura de codificação dividida em três blocos: transformação, quantização e codificação de entropia. Por exemplo, nos padrões JPEG [1] e MPEG [2], a imagem ou o quadro diferença com compensação de movimento é inicialmente transformada em um conjunto de coeficientes usando uma DCT por blocos. Estes coeficientes são em seguida quantizados por quantizadores escalares e depois codificados em entropia usando codificação de corridas de zeros e códigos de Huffman. No método EBCOT para compressão de imagens, recentemente adotado no padrão JPEG2000 [3], a transformada wavelet é utilizada, juntamente com quantização escalar por aproximações sucessivas seguida de um codificador aritmético de entropia. O sucesso destes esquemas depende da validade da hipótese das imagens típicas serem essencialmente de natureza passa-baixas, de modo que a maior parte da energia fica concentrada nos coeficientes de baixa freqüência. Neste caso, existem métodos eficientes de codificação dos coeficientes quantizados. Contudo, esta hipótese não é verdadeira para uma grande classe de imagens, como por exemplo, texto e gráficos. Tais imagens devem ser codificadas por algoritmos dedicados. Surge então um problema quando a imagem é mista, ou seja, contém texto, gráficos e imagens em tons de cinza misturados. A abordagem usual neste caso é comutar adaptativamente o método de codificação utilizado, dependendo do conteúdo local da ima-

Murilo B. de Carvalho está na Universidade Federal Fluminense. Eduardo A. B. da Silva está na Universidade Federal do Rio de Janeiro. Weiler Alves Finamore está na Pontifícia Universitária Católica do Rio de Janeiro. Emails: murilo@telecom.uff.br, eduardo@lps.ufrj.br, weiler@cetuc.puc-rio.br.

Editores responsáveis: Antonio Sérgio Bezerra Sombra, Ricardo Menezes Campello de Souza e Max Gerken. Submetido em 31/Dez/2001; revisado em 21/Mar/2002; aceito em 02/Abr/2002.

gem [4]. Outro exemplo no qual os métodos baseados em transformada não são totalmente satisfatórios é a codificação de quadros diferença com compensação de movimento. Para tais sinais a propriedade de compactação de energia nos coeficientes de baixa freqüência em geral não é válida.

Considerando-se o exposto acima, vale a pena estudar métodos de compressão alternativos que não sejam baseados no paradigma transformação-quantização-codificação de entropia, se quisermos superar as limitações intrínsecas dos métodos que utilizam transformadas. Neste artigo nós propomos um método de compressão de dados multidimensionais que nós chamamos de MMP ("Multidimensional Multiscale Parser"). Neste método, o sinal a ser comprimido é inicialmente segmentado em blocos de comprimento variável. Cada bloco é codificado usando contrações e expansões de vetores pertencentes a um dicionário, sendo esta a razão do uso do termo "multiescalas". O dicionário é atualizado enquanto o sinal é codificado através da inclusão de concatenações de expansões e contrações de vetores previamente codificados. Podemos ver o MMP como um codificador que usa casamento de padrões recorrentes multiescalas. Normalmente o MMP é inicializado com um dicionário muito simples e aprende seus vetores a partir do próprio sinal. Por isso, o MMP tende a funcionar bem como uma variedade de fontes diferentes e adapta-se bem a fontes não estacionárias. Ele funciona bem em taxas altas ou baixas e possui inclusive a capacidade de fazer compressão sem perdas. Uma vez que seu procedimento de segmentação pode ser facilmente descrito em espaços de qualquer dimensão, sua extensãao para o uso com fontes multidimensionais é trivial.

A organização deste artigo é a seguinte: na Seção 2 o problema do casamento de padrões multiescalas é definido e uma versão unidimensional do MMP controlada por um parâmetro de distorção alvo é apresentada. Na Seção 3 nós apresentamos uma versão do MMP otimizada no sentido taxa-distorção. A Seção 4 contém uma generalização do MMP para uso com fontes multidimensionais enquanto a Seção 5 mostra uma versão mais rápida do MMP utilizando múltiplos dicionários. Resultados experimentais obtidos com imagens são apresentados na Seção 6 e a Seção 7 contém as conclusões. No Apêndice A mostramos uma análise matemática do casamento aproximado de vetores Gaussianos usando dicionários com múltiplas escalas que serve de justificativa para o uso de escalas no MMP.

2. O ALGORITMO MMP

Neste trabalho nós estudamos métodos para comprimir dados com perdas usando um novo conceito que nós chamamos *casamento aproximado de padrões multiescalas*. Trata-se de uma extensão do casamento aproximado de padrões tradicional onde nós permitimos que vetores de comprimentos diferentes possam ser casados. Para que isto seja possível nós empregamos uma transformação de escala $T_N^M : \mathbb{R}^M \mapsto \mathbb{R}^N$ para ajustar os tamanhos dos vetores antes da tentativa de casamento [5]. Por exemplo, se desejarmos usar um vetor S de comprimento N' para representar um outro vetor X de comprimento diferente N, nós primeiro calculamos $\mathrm{S}^s = T_N^{N'}[\mathrm{S}]$, uma versão escalada de S que tem o mesmo comprimento N de X (podemos também apenas escrever $\mathrm{S}^s = T_N[\mathrm{S}]$ ficando subentendido que $N' = \ell(\mathrm{S})$, ou seja, o comprimento de S). Esta operação está ilustrada na Figura 1.

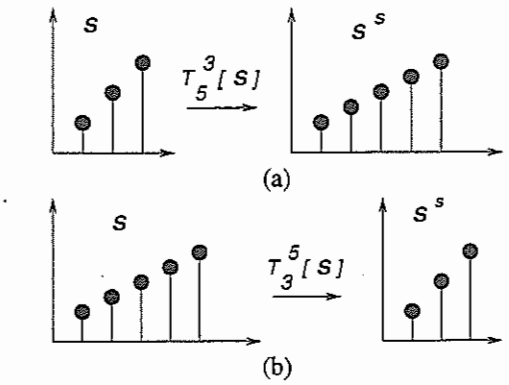


Figura 1. Transformação de escala $\mathrm{S}^s = T_N^M[\mathrm{S}]$: (a) $N > M$; (b) $N < M$

Nós podemos então usar S^s para representar X desde que eles sejam suficientemente próximos. Isto está ilustrado na Figura 2.

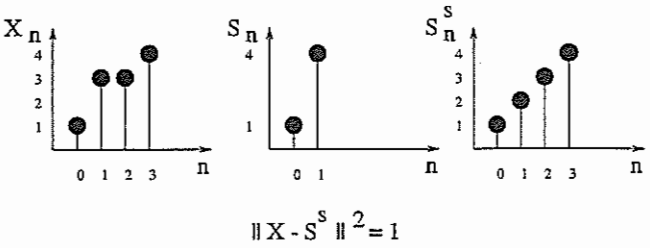


Figura 2. Casamento aproximado de padrões multiescalas.

No Apêndice A nós fazemos uma análise matemática das propriedades de casamento de vetores Gaussianos. Lá, nós comparamos o desempenho de um QV (Quantizador Vetorial) cujo dicionário é construído a partir de blocos de amostras do sinal de entrada, com o desempenho de outro QV, cujo dicionário é constituído de versões escaladas de blocos de amostras da entrada. Nós concluímos que, mesmo para fontes Gaussianas sem memória, pode ser vantajoso usar dicionários com escalas, especialmente em baixas taxas. Isso nos dá uma indicação teórica de que o uso do casamento aproximado de padrões multiescalas pode melhorar o desempenho. A seguir nós propomos o algoritmo de compressão MMP, que é baseado no casamento aproximado de padrões multiescalas.

No MMP, um vetor de entrada X é segmentado em L blocos sem superposição e cada bloco é representado por uma versão *dilatada/contraída* de um vetor pertencente a um dicionário $\mathcal{D}$. O dicionário é construído adaptativamente enquanto o sinal é codificado e é regularmente atualizado com concatenações de padrões previamente ocorridos. Esta regra de atualização foi inspirada pelo funcionamento do algoritmo de compressão de dados sem perdas Lempel-Ziv [6]. Existem trabalhos anteriores de extensão do algoritmo de Lempel-Ziv para o caso com perdas onde o casamento exato de padrões foi substituido por casamento aproximado de padrões. Podemos chamar estas extensões de algoritmos LLZ ("Lossy-Lempel-Ziv"). Em [7], [8], [9], algumas variações do LLZ

são exploradas. Quando aplicados à compressão de imagens com perdas, estes métodos provaram ser competitivos nas taxas acima de um bit por pixel, se comparados ao codificador JPEG. Em [10], uma versão bidimensional do LLZ é desenvolvida e aplicada à compressão de imagens. Embora seu desempenho seja superior ao das versões unidimensionais, continua sendo inferior ao do JPEG para taxas inferiores a um bit por pixel. Também existem trabalhos relacionados em quantização vetorial adaptativa [11],[12]. Nosso algoritmo propõe uma abordagem nova, diferindo destes trabalhos anteriores tanto no método de segmentação quanto na técnica de atualização do dicionário, além de introduzir o uso do casamento de padrões multiescalas. De fato, uós acreditamos que o bom desempenho em taxas inferiores a um bit por pixel, em contraste com o dos outros algoritmos citados, foi em grande parte devido ao uso do casamento aproximado de padrões multiescalas.

A versão mais simples do MMP é controlada por um parâmetro de distorção alvo d^*. Nesta versão do MMP, nós tentamos aproximar o vetor de entrada $\mathbf{X}$ usando o melhor elemento no dicionário como $\hat{\mathbf{X}} = T_N[\mathrm{S}_{i_0}]$. Se a tentativa falhar, ou seja, a distorção $d(\mathbf{X}, \hat{\mathbf{X}})$ está acima de Nd^*, nós dividimos $\mathbf{X}^0 = \mathbf{X}$ em dois segmentos $\mathbf{X}^1$ e $\mathbf{X}^2$, cada um de comprimento $N/2$, e então nós procuramos aproximar cada um usando versões escaladas dos vetores no dicionário. Se a tentativa de representar qualquer um dos segmentos usando o dicionário corrente falha, aquele segmento será subdividido em dois e o procedimento repetido. Este processo está ilustrado na Figura 3.

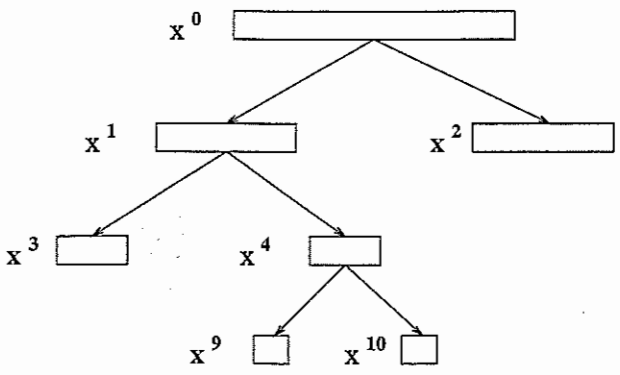


Figura 3. Segmentação no MMP.

Tendo descrito como, dado um dicionário, o MMP codifica um segmento, devemos descrever como o dicionário é atualizado. Devemos primeiramente observar que existe uma árvore de segmentação $\mathcal{S}$ associada à um dado vetor de entrada $\mathbf{X}$ e uma dada distorção alvo d^*. A Figura 4 mostra a árvore de segmentação $\mathcal{S}$ associada à segmentação na Figura 3. Cada nodo n_j de $\mathcal{S}$ está associado a um segmento $\mathbf{X}^j$ do vetor de entrada. O comprimento dos vetores associados ao nodos na profundidade p é $2^{-p}N$. Um nodo n_j da árvore de segmentaçã pode ter dois filhos, nodos n_{2j+1} e n_{2j+2}, ou então nenhum filho. Um nodo sem filhos é um nodo folha. No MMP, apenas os nodos folhas da árvore de segmentação são associados a vetores do dicionário que são usados para aproximar o vetor de entrada.

Por exemplo, a árvore de segmentação na Figura 4 corresponde a uma segmentação do vetor de entrada $\mathbf{X}$ em quatro segmentos, $(\ \mathbf{X}^3\ \ \mathbf{X}^9\ \ \mathbf{X}^{10}\ \ \mathbf{X}^2\ )$. Neste exemplo, cada segmento será aproximado por versões escaladas de vetores no dicionário, levando a uma representação $\hat{\mathbf{X}} = (\ \hat{\mathbf{X}}^3\ \ \hat{\mathbf{X}}^9\ \ \hat{\mathbf{X}}^{10}\ \ \hat{\mathbf{X}}^2\ )$.

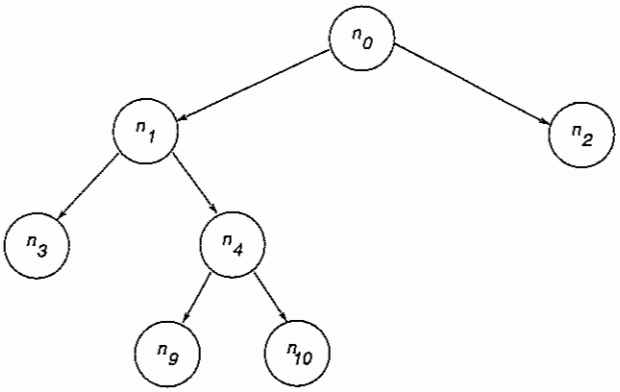


Figura 4. Árvore de segmentação.

O dicionário do MMP é atualizado do seguinte modo: sempre que as aproximações $\hat{\mathbf{X}}^{2j+1}$ e $\hat{\mathbf{X}}^{2j+2}$ associadas aos nodos filhos n_{2j+1} e n_{2j+2} estiverem disponíveis, o MMP formará uma estimativa do segmento $\hat{\mathbf{X}}^j$, associado ao nodo pai n_j, composta pela concatenação das aproximações dos segmentos associados aos dois nodos filhos. No exemplo da Figura 3, quando $\hat{\mathbf{X}}^9$ e $\hat{\mathbf{X}}^{10}$ estão disponíveis podemos concatena-las para obter uma nova aproximação $\hat{\mathbf{X}}^4$. Esta nova aproximação pode então ser incluída no dicionário

Portanto, no MMP, é necessário fornecer ao decodificador apenas a informação relativa aos indices do dicionário correspondendo às aproximações nos nodos folha e a informação necessária para especificação da árvore de segmentação. Isto é, o algoritmo produz uma saída composta de uma seqüência de inteiros i_m consistindo dos índices do dicionário e uma seqüência de flags binários b_n que especificam a árvore de segmentação $\mathcal{S}$. A seqüência de flags determina $\mathcal{S}$ por meio de uma série de decisões binárias, da raiz para as folhas. Nós utilizamos o flag binário 0 para indicar uma subdivisão em duas partes, e o flag 1 para indicar um nodo folha. Por exemplo, a árvore na Figura 4 é representada pela seqüência de flags 0, 0, 1, 0, 1, 1, 1.

Neste ponto, dois aspectos importantes devem ser ressaltados:

1. Embora o MMP use dicionários multiescalas, a escala de um elemento do dicionário usado para codificar um segmento não precisa ser transmitida, pois pode ser inferida da árvore de segmentação.

2. O dicionário é atualizado pelo decodificador usando informação que pode ser derivada apenas dos flags de segmentação e dos índices do dicionário recebidos. Portanto, não há necessidade de enviar informação lateral para atualizar o dicionário do receptor.

3. OTIMIZAÇÃO DA ÁRVORE DE SEGMENTAÇÃO

A árvore de segmentação gerada pela versão do MMP parametrizada por uma distorção alvo é criada usando-se decisões locais baseadas em cálculos de distorção apenas, e não é globalmente ótima num sentido taxa-distorção. O desempenho do MMP pode ser melhorado se otimizarmos sua árvore de segmentação [13].

Observando-se as Figuras 3 e 4, cada nodo n_j está associado a um segmento do vetor de entrada $\mathbf{X}^j$ cuja melhor aproximação é obtida por uma versão escalada do elemento $\mathbf{S}_{i_j}$ do dicionário, denotada como $\mathbf{S}_{i_j}^s$. Desse modo, podemos associar a cada nodo a distorção:

$$D(n_j) = d\left(\mathbf{X}^j, \mathbf{S}_{i_j}^s\right) \tag{1}$$

Nós denotamos $R(n_j)$ a taxa necessária para especificar o índice i_j, ou seja:

$$R(n_j) = -\log_2(\Pr(i_j)) \tag{2}$$

onde $\Pr(i_j)$ é a probabilidade de ocorrência do índice i_j do dicionário. A distorção total será:

$$D(\mathcal{S}) = \sum_{n_j \in \mathcal{S}_\mathcal{L}} D(n_j) \tag{3}$$

onde $\mathcal{S}_\mathcal{L}$ é o conjunto de nodos folhas de $\mathcal{S}$. A quantidade de bits necessária para codificar esta aproximação é a taxa $R(\mathcal{S})$, que é dada por:

$$R(\mathcal{S}) = R_t(\mathcal{S}) + \sum_{n_j \in \mathcal{S}_\mathcal{L}} R(n_j) \tag{4}$$

onde $R_t(\mathcal{S})$ é a taxa necessária para especificar a árvore de segmentação.

A melhor segmentação $\mathcal{S}^*$, num sentido taxa-distorção, levará à menor taxa $R(\mathcal{S})$ dado que a distorção $D(\mathcal{S})$ não é superior a uma distorção alvo D^* ou, alternativamente, á menor distorção $D(\mathcal{S})$ a uma taxa R^*. Este é um problema de minimização com condição de contorno definido como:

$$\begin{aligned} \mathcal{S}^* &= \arg\min_{\mathcal{S} \in \mathcal{S}_{R^*}} D(\mathcal{S}), \\ \mathcal{S}_{R^*} &= \{\mathcal{S} : R(\mathcal{S}) = R^*\} \end{aligned} \tag{5}$$

Para encontrar a solução $\mathcal{S}^*$ introduzimos um multiplicador de Lagrange λ. Sabe-se que ao minimizar o custo Lagrangeano $J(\mathcal{S}) = D(\mathcal{S}) + \lambda R(\mathcal{S})$ [14], encontra-se simultaneamente a solução do problema original se escolhemos $R(\lambda) = R^*$. Assim temos que:

$$\begin{aligned} \mathcal{S}^* &= \arg\min_{\mathcal{S}} J(\mathcal{S}) \\ &= \arg\min_{\mathcal{S}} \left(\sum_{n_j \in \mathcal{S}_\mathcal{L}} D(n_j) \right) + \lambda \left(R_t(\mathcal{S}) + \sum_{n_j \in \mathcal{S}_\mathcal{L}} R(n_j) \right) \\ &= \arg\min_{\mathcal{S}} \lambda R_t(\mathcal{S}) + \sum_{n_j \in \mathcal{S}_\mathcal{L}} (D(n_j) + \lambda R(n_j)) \\ &= \arg\min_{\mathcal{S}} \lambda R_t(\mathcal{S}) + \sum_{n_j \in \mathcal{S}_\mathcal{L}} J(n_j) \end{aligned} \tag{6}$$

onde $J(n_j) = D(n_j) + \lambda R(n_j)$.

Uma sub-árvore $\mathcal{S}(n_j)$ de $\mathcal{S}$ no nodo n_j é a árvore binária composta de todos os nodos de $\mathcal{S}$ que tem n_j como nó raiz. A Figura 5 ilustra a sub-árvore $\mathcal{S}(n_4)$ da árvore binária da Figura 4. Denotaremos $\mathcal{S} - \mathcal{S}(n_j)$ a árvore obtida de $\mathcal{S}$ pela podagem da sub-árvore $\mathcal{S}(n_j)$.

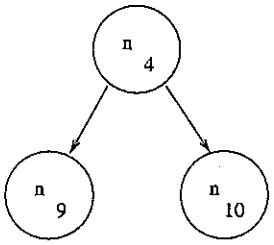


Figura 5. A sub-árvore $\mathcal{S}(n_4)$ da árvore binária na Figura 4.

Se os custos Lagrangeanos $J(n_l)$ associados às aproximações de cada segmento $\mathbf{X}^l$ são independentes, então o custo Lagrangeano das sub-árvores $J(\mathcal{S}(n_l))$ e $J(\mathcal{S}(n_m))$ são também independentes, bastando que todos os nodos das duas sub-árvores sejam diferentes. Neste caso, um algoritmo de busca rápido, similar ao descrito em [15], pode ser implementado considerando-se que se $J(n_l) \leq J(\mathcal{S}(n_{2l+1})) + J(\mathcal{S}(n_{2l+2}))$ então as sub-árvores $\mathcal{S}(n_{2l+1})$ e $\mathcal{S}(n_{2l+2})$ devem ser podadas para que o custo diminua. Infelizmente, este não é o caso do MMP, uma vez que os custos $J(n_l)$ estão acoplados pelo procedimento de atualização do dicionário. Isto é, o cálculo da variação global do custo devido à podagem da sub-árvoree $\mathcal{S}(n_l)$ deve levar em consideração a variação no custo dos outros nodos $J(n_k)$ devida à mudança do dicionário (que foi causada pela podagem da sub-árvore $\mathcal{S}(n_l)$). Contudo, se os dicionários iniciais são suficientemente grandes, pode-se argumentar que o efeito da atualização do dicionário na minimização de $J(n_l)$ pode ser desprezado. Nas implementações práticas do MMP, usualmente impõe-se um limite superior ao tamanho M do vetor de entrada $\mathbf{X}$ por causa da quantidade limitada de memória disponível. Por isso o sinal de entrada deve ser inicialmente dividido em blocos de tamanho M que são processados sequencialmente pelo algorítmo (o dicionário inicial para cada bloco será o dicionário final obtido após o processamento do bloco anterior). Ao processar um bloco de tamanho M, os dicionários do MMP serão aumentados de, no máximo, $2M - 1$ elementos. Portanto se o tamanho do dicionário inicial é muito maior do que $2M - 1$, o efeito da atualização sobre os custos será pequeno. Apesar de não ocorrer nos primeiros blocos, eventualmente o dicionário torna-se grande o bastante para que os custos $J(n_l)$ tornem-se praticamente desacoplados. Neste sentido, o algoritmo em [15] pode ser usado para encontrar uma solução aproximada para (5).

Contudo, se desejarmos usar blocos relativamente grandes ou se o dicionário é muito pequeno (como ocorre em taxas muito baixas), nós devemos modificar o algoritmo para levar em consideração o impacto do procedimento de atualização do dicionário sobre os custos. Sabemos que o dicionário é atualizado pela inclusão da concatenação de segmentos previamente codificados. Portanto, se decidimos podar uma sub-

árvore, esta ação não afeta apenas o custo na sub-árvore, mas pode afetar todos os nodos à direita da raiz da sub-árvore. Ou seja, se podamos uma sub-árvore, pode acontecer de estarmos também removendo um elemento do dicionário que seria usado depois para representar um segmento de entrada, e portanto aumentando o custo total. A idéia é podar uma sub-árvore apenas quando o aumento potencial no custo dos nodos subsequentes, devido à remoção de alguns elementos do dicionário, for menor ou igual à redução no custo provida pela remoção daquela sub-árvore. O algoritmo para encontrar a árvore de segmentação para um bloco de comprimento M está descrito a seguir. Os descendentes do nodo n_j são os nodos n_{2j+1} e n_{2j+2}.

passo 1 Inicialize $\mathcal{S}$ como a árvore completa de profundidade $\log_2(M)+1$.

passo 2 Faça $J_i = \infty$ para os M nodos folhas, ou seja, para $i = M-1, M, \ldots, 2M-2$.

passo 3 Faça $p = \log_2(M)$ e $\mathcal{S}_0 = \mathcal{S}$.

passo 4 Para cada nodo $n_l \in \mathcal{S}$ na profundidade p, ou seja, para $l \in \{2^{p-1}-1, 2^{p-1}, \ldots, 2^p-2\}$, calcule:

(i) $J_l = J(n_l) + \lambda R_{1_l}$, onde $J(n_l)$ é o custo para representar o segmento de entrada associado ao nodo n_l e R_{1_l} é a taxa necessária para indicar que o nodo n_l é uma folha.

(ii) $\Delta J_l = \sum_{n_r \in \mathcal{S}-\mathcal{S}(n_l)} (J(n_r) - J'(n_r))$, onde $J'(n_l)$ é calculado usando-se o dicionário sem $\hat{\mathbf{X}}^l = \begin{pmatrix} \hat{\mathbf{X}}^{2l+1} & \hat{\mathbf{X}}^{2l+2} \end{pmatrix}$, ou seja, o dicionário que seria obtido sem a subárvore $\mathcal{S}(n_l)$.

passo 5 Se $J_l - J_{2l+1} - J_{2l+2} - \lambda R_{0_l} \leq \Delta J_l$ então pode os nodos n_{2l+1} e n_{2l+2} de $\mathcal{S}$. (R_{0_l} é a taxa necessária para indicar a subdivião do segmento em dois, e J_{2l+1}, J_{2l+2} foram avaliados na iteração anterior com $p+1$). Caso contrário, o custo do nodo n_l é atualizado comi $J_{2l+1} + J_{2l+2} + \lambda R_{0_l}$.

passo 6 Faça $p = p-1$.

passo 7 Repita os passos 4 ao 6 até que $p = 0$.

passo 8 Se $\mathcal{S} = \mathcal{S}_0$ então a otimização terminou. Caso contrário volte ao passo 3.

É interessante considerar porque ΔJ_l é avaliado usando todos os nodos pertencentes a $\mathcal{S}(n_l)$, uma vez que apenas os nodos folhas contribuem para o custo total. A idéia do algoritmo é podar uma sub-árvore apenas quando temos certeza que o custo não irá aumentar. O cálculo de ΔJ_l deve ser conservador porque não se sabe, no momento que uma decisão é tomada no nodo n_l, quais nodos irão ser as folhas (as folhas da árvore atual podem vir a ser podadas depois). Ou seja, quando avaliamos $\Delta J_l < J_l - J_{2l+1} - J_{2l+2} - \lambda R_{0_l}$, podamos a sub-árvore $\mathcal{S}(n_l)$ apenas quando temos certeza, independentemente de qual venha a ser a árvore final $\mathcal{S} - \mathcal{S}(n_l)$, que o custo irá decrescer. Se o custo local aumenta com a podagem e decidimos não podar, esta decião pode ser a correta ou não, dependendo da árvore no futuro $\mathcal{S} - \mathcal{S}(n_l)$. É por esta razão que o algoritmo deve ser iterado até a convergência, sendo esta garantida por que o custo nunca aumenta a cada iteração.

4. MMP COM FONTES MULTIDIMENSIONAIS

Pode-se interpretar o processo de segmentação do MMP como subdividindo o vetor de entrada $\mathbf{X}^j$ em dois segmentos em torno do ponto $p_s = \ell(\mathbf{X}^j)/2$, onde $\ell(\mathbf{X}^j)$ é o comprimento de $\mathbf{X}^j$. Neste caso, p_s é um ponto num espaço unidimensional. Este processo pode ser facilmente estendido para operar com fontes multidimensionais, se consideramos o ponto de segmentação $\mathbf{p}_s$ definido num espaço multidimernsional. Por exemplo, no caso bidimensional, o vetor de entrada é substituído por uma *matriz* de entrada $\mathbf{X}$ de tamanho $N \times N$. Neste caso, existem várias possibilidades de escolha para o ponto de segmentação $\mathbf{p}_s = \begin{pmatrix} p_l & p_c \end{pmatrix}$. Por exemplo, pode-se escolher $\mathbf{p}_s = \begin{pmatrix} \ell_{row}(\mathbf{X}^j)/2 & \ell_{col}(\mathbf{X}^j)/2 \end{pmatrix}$, onde $\ell_{row}(\mathbf{X}^j)$ é o número de linhas da matriz $\mathbf{X}^j$ e $\ell_{col}(\mathbf{X}^j)$ é seu número de colunas. Neste caso, a matriz $\mathbf{X}^j$ seria subdividida em quatro sub-matrizes como $\mathbf{X}^j = \begin{pmatrix} \mathbf{X}^{4j+1} & \mathbf{X}^{4j+2} \\ \mathbf{X}^{4j+3} & \mathbf{X}^{4j+4} \end{pmatrix}$. Esta escolha de $\mathbf{p}_s$ levaria a uma segmentação do tipo "*quad-tree*". Esta segmentação pode ser representada por uma árvore quaternária onde um nodo n_j pode ter quatro filhos, $n_{4j+1}, n_{4j+2}, n_{4j+3}, n_{4j+4}$, ou nenhum filho. Um nodo n_j na profundidade p seria associado a uma matriz $\mathbf{X}^j$ de tamanho $2^{-p}N \times 2^{-p}N$.

Pode-se manter a representação em árvore binária caso sejam usados pontos de segmentação diferentes. Por exemplo, uma regra possível é: Se $\ell_{col}(\mathbf{X}^j) > \ell_{row}(\mathbf{X}^j)$ então $\mathbf{p}_s = \begin{pmatrix} 0 & \ell_{col}(\mathbf{X}^j)/2 \end{pmatrix}$, caso contrário $\mathbf{p}_s = \begin{pmatrix} \ell_{row}(\mathbf{X}^j)/2 & 0 \end{pmatrix}$. Esta escolha para os pontos de segmentação leva à segmentação ilustrada na Figura 6.

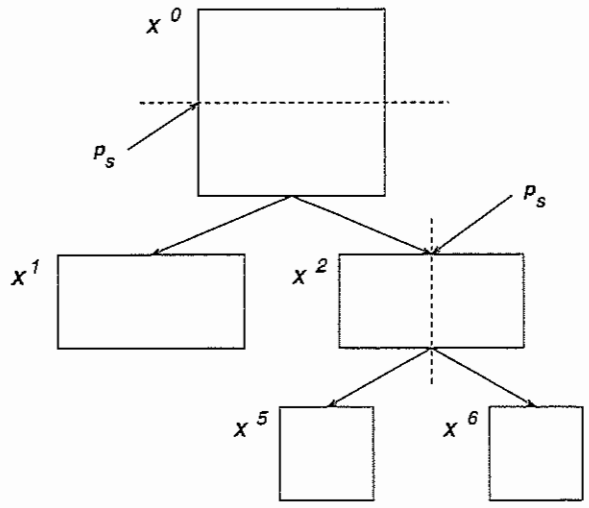


Figura 6. Um exemplo de segmentação bidimensional.

Em outras palavras, se $\ell_{col}(\mathbf{X}^j) > \ell_{row}(\mathbf{X}^j)$ então a matriz $\mathbf{X}^j$ é subdividida como $\mathbf{X}^j = \begin{pmatrix} \mathbf{X}^{2j+1} & \mathbf{X}^{2j+2} \end{pmatrix}$, caso contrário $\mathbf{X}^j = \begin{pmatrix} \mathbf{X}^{2j+1} \\ \mathbf{X}^{2j+2} \end{pmatrix}$. Esta segmentação pode

ser representada por uma árvore binária, e um nodo n_j na profundidade p estará associado a uma matriz $\mathbf{X}^j$ de tamanho $\left(2^{-\lfloor\frac{p+1}{2}\rfloor}N \times 2^{-\lfloor\frac{p}{2}\rfloor}N\right)$, onde $\lfloor x\rfloor$ é o maior inteiro menor ou igual a x. Em nossos experimentos, esta segmentação bimária proporcionou um melhor desempenho do que a segmentação quadtree.

O MMP bidimensional deve usar uma transformação de escala bidimensional $T_{N,M}^{N',M'} : \mathbb{R}^{N'M'} \mapsto \mathbb{R}^{NM}$ para poder fazer casamento aproximado multiescalas de padrões bidimensionais. Esta transformação mapeia uma matriz de dimensão $N' \times M'$ em uma matriz de dimensão $N \times M$. Analogamente ao caso unidimensional pode-se usar a notação simplificada $T_{N,M}[\mathbf{X}^j]$ implicando que $\mathbf{X}^j$ é de tamanho $N' \times M'$. Este procedimento pode ser trivialmente estendido para mais de duas dimensões.

5. IMPLEMENTAÇÃO DO MMP USANDO MÚLTIPLOS DICIONÁRIOS

Quando o MMP tenta representar um segmento de entrada $\mathbf{X}^j$ usando um dos vetores em seu dicionário $\mathcal{D}$, ele primeiramente deve usar uma transformação de escala para ajustar o tamanho de cada vetor em $\mathcal{D}$ para igualar o comprimento de $\mathbf{X}^j$. Se o comprimento do vetor de entrada é N, o procedimento de segmentação pode criar vetores de comprimentos $N/2, N/4, \ldots, 1$. Sendo assim, há no máximo $1 + \log_2(N)$ comprimentos, ou escalas, diferentes. Portanto, para poupar tempo, pode-se manter $1+\log_2(N)$ cópias do dicionário, uma para cada escala, para evitar a computação da transformação de escala todas as vezes que um casamento for feito. Desse modo, basta usar a transformação de escala quando um vetor novo está sendo incluído no dicionário. Denotaremos a cópia do dicionário na escala $2^{-p}N$ como $\mathcal{D}^p$. Se estivermos usando este esquema de múltiplos dicionários, para incluir um novo vetor $\hat{\mathbf{X}}^j$ no dicionário devemos de fato incluir $T_{2^{-p}N}[\hat{\mathbf{X}}^j]$ em $\mathcal{D}^p$ para $p = 0, 1, \ldots, \log_2(N)$.

6. RESULTADOS EXPERIMENTAIS

Nós implementamos o MMP na sua versão controlada por distorção-alvo e na sua versão com otimização taxa-distorção, ambas com a segmentação bidimensional (árvore binária) descrita na Seção 4. Nós denominamos estas implementações 2D-MMP e 2D-MMP-RD respectivamente. Nós usamos os programas para comprimir imagens preto-e-branco digitalizadas. As imagens foram inicialmente divididas em blocos $N \times N$ que foram processados sequencialmente pelo algoritmo. Nós usamos a abordagem de múltiplos dicionários com $1 + 2\log_2(N)$ codebooks. O dicionário inicial na escala 1×1 foi $\mathcal{D}_0^0 = \{-128, -124, \ldots, -4, 0, +4, \ldots, +124\}$. Os dicionários iniciais $\mathcal{D}_0^p$ nas outras escalas $\left(2^{-\lfloor\frac{p+1}{2}\rfloor}N \times 2^{-\lfloor\frac{p}{2}\rfloor}N\right), p = 0, 1, \ldots, 2\log_2(N)$ foram obtidos de $\mathcal{D}_0^0$ pela aplicação de uma transformação de escalas bidimensional. Nós escolhemos usar uma transformação de escalas *separável* baseada em uma transformação unidimensional aplicada independentemente às linhas e às colunas de $\mathbf{X}^j$. A transformação de escalas unidimensional foi implementada usando operações clásicas de mudança de taxa de amostragem. Quando queremos mudar do comprimento N_0 para o comprimento N, com $N > N_0$, nós primeiro mudamos para o comprimento N_0N usando um interpolador linear como filtro. Em seguida nós subamostramos por um fator N_0. Este procedimento está definido na equação 7.

$$\begin{aligned} m_n^0 &= \left\lfloor \frac{n\,(N_0-1)}{N} \right\rfloor \\ m_n^1 &= \begin{cases} m_n^0+1, & m_n^0 < N_0-1 \\ m_n^0, & m_n^0 = N_0-1 \end{cases} \\ \alpha_n &= n\,(N_0-1) - Nm_n^0 \\ S_n^s &= \left\lfloor \frac{\alpha_n\left(S_{m_n^1} - S_{m_n^0}\right)}{N} \right\rfloor + S_{m_n^0}, \\ & \quad n = 0, 1, \ldots, N-1 \end{aligned} \tag{7}$$

Quando $N_0 > N$, nós inicialmente mudamos para o comprimento N_0N usando um interpolador linear como filtro e então nós aplicamos um filtro de média móvel de comprimento $N_0 + 1$. Em seguida fazemos uma subamostragem por um fator N_0. Este procedimento está definido na equação 8.

$$\begin{aligned} m_{n,k}^0 &= \left\lfloor \frac{n\,(N_0-1)+k}{N} \right\rfloor \\ m_{n,k}^1 &= \begin{cases} m_{n,k}^0+1, & m_{n,k}^0 < N_0-1 \\ m_{n,k}^0, & m_{n,k}^0 = N_0-1 \end{cases} \\ \alpha_n &= n\,(N_0-1) + k - Nm_n^0 \\ S_n^s &= S_{m_{n,k}^0} + \frac{1}{N_0+1}\sum_{k=0}^{N_0} \left\lfloor \frac{\alpha_{n,k}\left(S_{m_{n,k}^1} - S_{m_{n,k}^0}\right)}{N} \right\rfloor, \\ & \quad n = 0, \ldots, N-1 \end{aligned} \tag{8}$$

A transformação de escala bidimensional foi implementada como na equação 9:

$$\begin{aligned} \mathbf{Y}_i &= T_M[\mathbf{S}_i], i = 0, 1, \ldots, \ell\,(\mathbf{S}_i) - 1 \\ \mathbf{S}_j^s &= \left(T_N\left[\left(\mathbf{Y}^T\right)_j\right]\right)^T, j = 0, 1, \ldots, M-1 \end{aligned} \tag{9}$$

onde $\mathbf{X}_i$ denota as linhas de $\mathbf{X}$. A seqüência de índices do dicionário i_m, bem como a seqüência de flags binários b_n foram codificadas usando um codificador aritmético adaptativo com contextos distintos para cada escala.

As Figuras 7(a), 7(b), 7(c) e 7(d) mostram o desempenho taxa × distorção dos algoritmos com as imagens Lena, F-16, PP1205 e PP1209, todas de dimensões 512×512. Em todos os casos, as imagens foram divididas em blocos 8×8. Os resultados obtidos com os algoritmos SPIHT [16] e JPEG [1] Também são mostrados para comparação. As imagens Lena e F-16 foram obtidas do endereço http://www.sipi.usc.edu. A imagem PP1205 foi obtida por "scanner" da página 1205 da revista *IEEE Transactions on Image Processing*, volume 9, nnmber 7, July 2000. Ela contém texto e fórmulas matemáticas. A imagem PP1209 foi obtida da página 1209 da mesma revista e é uma imagem composta de imagens preto-e-branco (duas versões comprimidas da Lena), texto, fórmulas e gráficos.

As figuras mostram que:

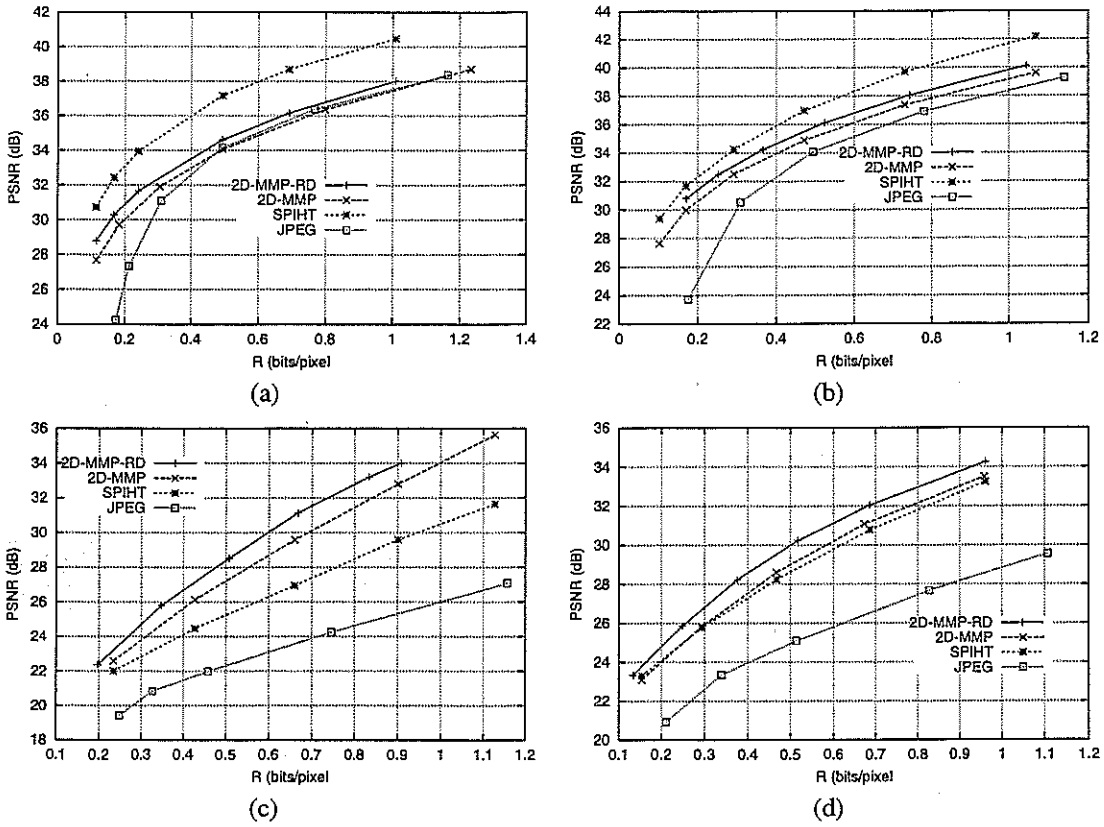


Figura 7. Curva taxa × distorção para:(a) LENA; (b) F-16; (c) PP1205 e (d) PP1209 512 × 512.

i) A versão 2D-MMP-RD superou a versão 2D-MMP com todas as imagens de teste.

ii) O 2D-MMP-RD superou o SPIHT para a imagem de texto e para a imagem composta (por $\approx$ 4.4 dB para PP1205 e $\approx$ 1.0 dB para PP1209).

iii) O algoritmo SPIHT superou o 2D-MMP-RD para as imagens em níveis de cinza (por $\approx$ 2.5 dB para Lena e $\approx$ 1.9 dB para F-16).

iv) O 2D-MMP-RD superou o JPEG com todas as imagens de teste.

As Figuras 8, 9 e 10 mostram algumas das imagens de teste comprimidas pelo 2D-MMP-RD e pelo SPIHT a 0.50 bits/pixel. Pode-se observar algum efeito de blocagem na imagem Lena comprimida pelo MMP que está ausente na representação obtida com o SPIHT, que se utiliza da transformada wavelet. De fato, a DWT específica utilizada, baseada no conjunto de filtros FIR biortogonais de fase linear de [17], possui vetores base longos que ajudam a eliminar os efeitos de blocagem. Contudo, esta mesma propriedade se torna inconveniente quando tentamos aproximar sinais com muitas bordas, como em PP1205 e PP1209. Observa-se que o desempenho do MMP nestes casos é muito bom. É também importante ressaltar que, embora inferior ao do SPIHT com imagens de níveis de cinza, o desempenho do MMP foi superior ao do codificador JPEG. Além disso, o fato de o MMP ter desempenho bom para texto, gráficos e imagens compostas sugere que o procedimento de atualização do dicionário do MMP lhe confere um certo caráter universal. Isto ganha relevância especialmente se considerarmos que o dicionário inicial usado foi o mesmo em todos os experimentos, sendo $\mathcal{D}_0^0 = \{-128, -124, \ldots, -4, 0, +4, \ldots, +124\}$.

7. CONCLUSÕES

Neste artigo nós descrevemos um novo método para compressão de sinais multidimensionais com perdas, baseado em casamento de padrões recorrentes multiescalas, chamado MMP.

Um dos pilares do MMP é o uso de escalas. Isto é, podemos usar, além dos vetores em um dicionário, todas as expansões e contrações deles. O sinal de entrada é segmentado em vetores de comprimento variável e cada segmento é codificado usando um elemento do dicionário escalado. A segmentação pode ser otimizada num sentido taxa × distorção.

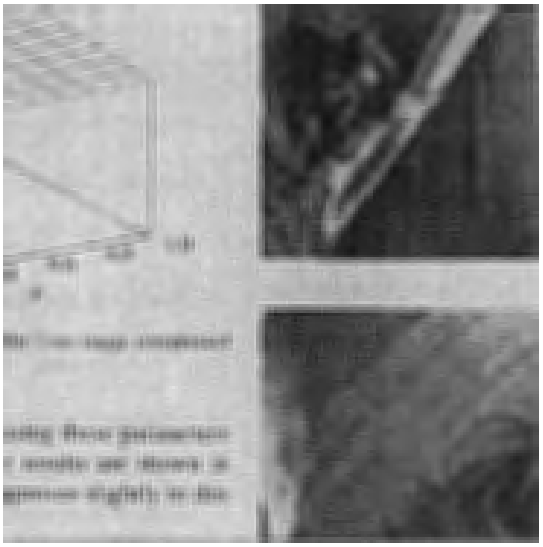
Outro pilar do MMP é o procedimento de atualização do dicionário. O dicionário é inicialmente muito simples, e é atualizado a medida que o dado é codificado, usando expansões e contrações de vetores previamente codificados. A atualização do dicionário é feita de tal modo que o decodificador pode atualizar seu dicionário sem a necessidade de receber nenhuma informação lateral. Desse modo o MMP pode adaptar-se a tipos diversos de fontes, como imagens em tons de cinza, quadros-diferença com compensação de movimento, texto e gráficos, entre outros. Num sentido limitado, pode-se dizer que o MMP possui um caráter universal.

Uma importante característica do MMP é que o seu procedimento de segmentação pode ser trivialmente executado em

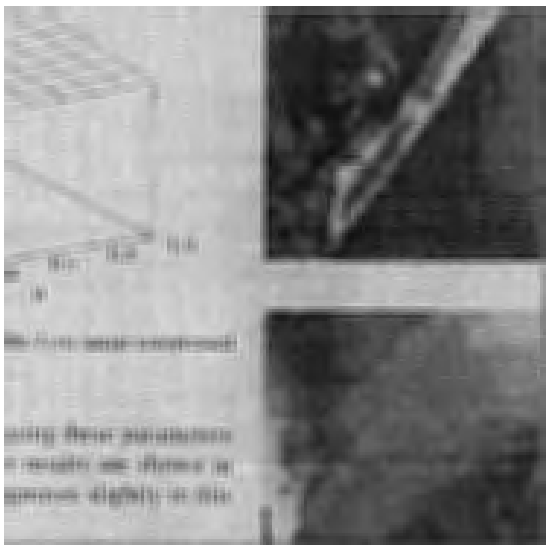
Figura 8. Imagem Lena: (a) 2D-MMP-RD a 0.5 bpp. PSNR = 34.7 dB.; (b) Detalhe; (c) SPIHT a 0.50 bpp. PSNR = 37.2 dB.; (d) Detalhe.

Figura 9. Imagem PP1205: (a) Original; (b) Detalhe; (c) 2D-MMP-RD a 0.5 bpp. PSNR = 28.5; (d) SPIHT a 0.50 bpp. PSNR = 25.2 dB.

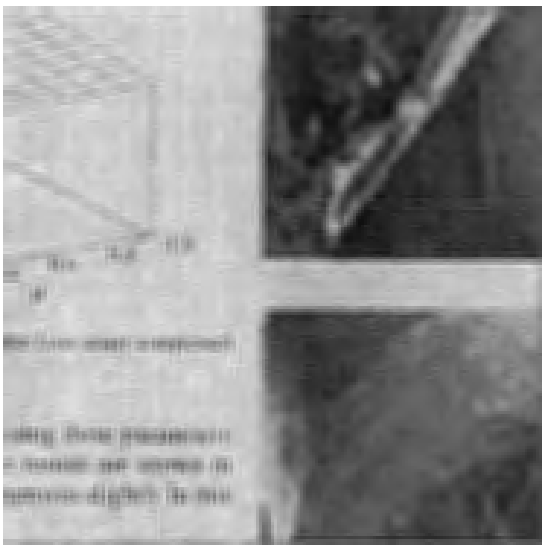
(a)



(b)



(c)



(d)

Figura 10. Imagem PP1209: (a) Original; (b) Detalhe; (c) 2D-MMP-RD a 0.5 bpp. PSNR = 29.9; (d) SPIHT a 0.50 bpp. PSNR = 28.7 dB.

múltiplas dimensões; portanto o MMP pode ser facilmente adaptado para operar diretamente com dados multidimensionais.

Vale a pena observar que o MMP pode operar sem perdas, ou seja com distorção nula, desde que o dicionário inicial expanda a faixa dinâmica do sinal de entrada. Isto é possível porque, no pior caso, a árvore de segmentação irá crescer até a profundidade máxima, com suas folhas correspondendo a escalares. Estes escalares podem então ser codificados com distorção nula pelos escalares do dicionário inicial. Portanto o MMP tem o potencial de executar compressão com e sem perdas de um modo simples e unificado.

É importante mencionar que o algoritmo MMP desenvolvido aqui representa apenas os primeiros passos na perquisa de uma classe de algoritmos de compressão baseados no casamento de padrões recorrentes multiescalas. A anáise matemática feita no Apêndice mostra que o casamento aproximado de vetores Gaussianos usando diferentes escalas pode ser vantajoso. Isto, em princípio, nos fornece uma boa indicação das potencialidades dos métodos desenvolvidos aqui. Isto é reforçado pelo fato do desempenho do método já ser razoavelmente bom para uma grande classe de imagens. Portanto, em nossa opinião, vale a pena aprofundar a pesquisa no uso de casamento aproximado de padrões recorrentes multiescalas.

8. APÊNDICE A: CASAMENTO APROXIMADO DE PADRÕES COM FONTES GAUSSIANAS SEM MEMÓRIA

Neste Apêndice nós investigamos as propriedades de casamento de vetores Gaussianos. Nosso propósito é obter uma justificativa teórica para o uso do casamento de padrões multiescalas em esquemas de compressão de dados.

8.1 CASAMENTO COM DICIONÁRIO DE VETORES GAUSSIANOS

Seja $\mathbf{x} = (x_1 x_2 \dots x_N)^T$ um vetor Gaussiano de média zero com matriz covariância $\mathbf{K}_x$ e seja $\mathcal{D} = \{\mathbf{y}^1, \mathbf{y}^2, \dots, \mathbf{y}^M\}$ um dicionário onde $\mathbf{y}^i = \begin{pmatrix} y_1^i & y_2^i & \dots & y_N^i \end{pmatrix}^T$ são vetores Gaussianos de média zero com matriz covariância $\mathbf{K}_{y^i} = \mathbf{K}_x$. Sejam $\mathbf{x}$ e $\mathbf{y}^i, i \in 1, M$ mutuamente independentes [20].

Nós diremos que ocorre um casamento quando pelo menos um dos vetores do dicionário pode ser usado para representar $\mathbf{x}$ com distorção menor ou igual a Nd^*. Se a medida de distorção é o erro quadrático médio, então teremos um casamento se $||\mathbf{x} - \mathbf{y}^i||^2 \leq Nd^*$ para algum $i \in [1, M]$. A probabilidade de casamento é então:

$$P_m = 1 - Pr\{||\mathbf{x} - \mathbf{y}^i||^2 > Nd^* \text{ para todo } i \in [1, M]\} \tag{10}$$

Usando todos os M vetores do dicionário, nós formamos um vetor $\mathbf{u} = ((\mathbf{y}^1)^T(\mathbf{y}^2)^T \dots (\mathbf{y}^M)^T)^T$. Os vetores do dicionário são Gaussianos, independentes, de média zero e com matrizes covariância $\mathbf{K}_x$, portanto o vetor $\mathbf{u}$ é Gaussiano de média zero com matriz covariância $\mathbf{K}_u = \mathbf{I}_M \otimes \mathbf{K}_x$, onde $\mathbf{I}_M$ é a matriz identidade $M \times M$ e $\otimes$ denota um produto de Kronecker. Para uma realização particular $\mathbf{X}$ do vetor aleatório $\mathbf{x}$ definimos o vetor diferença $\mathbf{r}^i = \mathbf{y}^i - \mathbf{X}, \quad i \in [1, M]$ e o vetor de vetores diferença $\mathbf{v}$ como:

$$\begin{aligned} \mathbf{v} &= ((\mathbf{r}^1)^T(\mathbf{r}^2)^T \dots (\mathbf{r}^M)^T)^T \\ &= \mathbf{u} - 1_M \otimes \mathbf{X} \end{aligned} \tag{11}$$

onde 1_M é o vetor $M \times 1$ de componentes unitárias $(11 \dots 1)^T$. O vetor $\mathbf{v}$ tem vetor média dado por $\mu_v = E\{\mathbf{v}\} = E\{\mathbf{u} - 1_M \otimes \mathbf{X}\} = -1_M \otimes \mathbf{X}$ e sua matriz covariância é $\mathbf{K}_v = \mathbf{K}_u = \mathbf{I}_M \otimes \mathbf{K}_x$.

Portanto, $\mathbf{v}$ possui uma função densidade de probabilidades (fdp) condicionada ao valor de $\mathbf{X}$ da forma:

$$p_{v|X}(\mathbf{V}) = \frac{1}{(2\pi)^{\frac{MN}{2}} |\mathbf{K}_v|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{V}-\mu_v)^T \mathbf{K}_v^{-1} (\mathbf{V}-\mu_v)} \tag{12}$$

O vetor diferença $\mathbf{r}^i$ possui norma dada por $||\mathbf{r}^i||^2 = (\mathbf{r}^i)^T \mathbf{r}^i = \mathbf{v}^T(\phi_M^i \otimes \mathbf{I}_N)\mathbf{v}$, onde ϕ_M^i é a *matriz indicadora* $M \times M$ na qual todos os elementos são zero exceto o elemento na linha i e na coluna i que é um. Podemos então definir o vetor de normas $\mathbf{z} = (||\mathbf{r}^1||^2 ||\mathbf{r}^2||^2 \dots ||\mathbf{r}^M||^2)^T$ cuja função característica [20] é dada por:

$$M_z(\mathbf{s}) = E\left\{e^{-\mathbf{s}^T \mathbf{z}}\right\} = E\left\{ e^{-\sum_{i=1}^{M} s_i \mathbf{v}^T (\phi_M^i \otimes \mathbf{I}_N)\mathbf{v}} \right\}$$

$$= \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} e^{-\sum_{i=1}^{M} s_i \mathbf{V}^T (\phi_M^i \otimes \mathbf{I}_N)\mathbf{V}} p_{v|X}(\mathbf{V}) d\mathbf{V}$$

$$= \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} e^{-\sum_{i=1}^{M} s_i \mathbf{V}^T (\phi_M^i \otimes \mathbf{I}_N)\mathbf{V}} \times$$

$$\times \frac{1}{(2\pi)^{\frac{MN}{2}} |\mathbf{K}_v|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{V}-\mu_v)^T \mathbf{K}_v^{-1} (\mathbf{V}-\mu_v)} d\mathbf{V}$$

$$= \left(\frac{|\mathbf{K'}_v|}{|\mathbf{K}_v|}\right)^{\frac{1}{2}} e^{-\frac{1}{2}\mu_v^T \mathbf{D} \mu_v} \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} \frac{1}{(2\pi)^{\frac{MN}{2}} |\mathbf{K'}_v|^{\frac{1}{2}}} \times$$

$$\times e^{-\frac{1}{2}(\mathbf{V}-\mu'_v)^T \mathbf{K'}_v^{-1} (\mathbf{V}-\mu'_v)} d\mathbf{V}$$

$$= \left(\frac{|\mathbf{K'}_v|}{|\mathbf{K}_v|}\right)^{\frac{1}{2}} e^{-\frac{1}{2}\mu_v^T \mathbf{D} \mu_v}$$

onde

$$\mathbf{K'}_v = \left(\mathbf{K}_v^{-1} + 2\sum_{i=1}^{M} s_i (\phi_M^i \otimes \mathbf{I}_N)\right)^{-1}$$

$$\mu'_v = \left(\mathbf{I}_{MN} + 2\sum_{i=1}^{M} s_i (\phi_M^i \otimes \mathbf{I}_N)\mathbf{K}_v\right)^{-1} \mu_v$$

$$\mathbf{D} = \left(\mathbf{K}_v + \sum_{i=1}^{M} \frac{1}{2s_i} (\phi_M^i \otimes \mathbf{I}_N)\right)^{-1}$$

e finalmente:

$$M_z(\mathrm{s}) = \left(\left|\mathbf{I}_{MN} + 2\sum_{i=1}^{M} s_i \mathbf{K}_v(\phi_M^i \otimes \mathbf{I}_N)\right|\right)^{-\frac{1}{2}} \times \times e^{-\frac{1}{2}\mu_v^T\left(\mathbf{K}_v + \sum_{i=1}^{M}\frac{1}{2s_i}(\phi_M^i \otimes \mathbf{I}_N)\right)^{-1}\mu_v} \quad (13)$$

Substituindo-se as expressões para média e covariância do vetor $\mathbf{v}$ e $\mathbf{I}_M = \sum_{i=1}^{M}\phi_M^i$ em 13 nós obtemos:

$$\begin{aligned} M_z(\mathrm{s}) &= \left|\sum_{i=1}^{M}\phi_M^i \otimes \mathbf{I}_N + 2\sum_{i=1}^{M} s_i(\mathbf{I}_M \otimes \mathbf{K}_x) \times \right. \\ &\quad \left. \times(\phi_M^i \otimes \mathbf{I}_N)\right|^{-\frac{1}{2}} \times \\ &\times e^{-\frac{1}{2}(1_M^T \otimes \mathbf{X}^T)\left(\mathbf{I}_M \otimes \mathbf{K}_x + \sum_{i=1}^{M}\frac{1}{2s_i}(\phi_M^i \otimes \mathbf{I}_N)\right)^{-1}(1_M \otimes \mathbf{X})} \\ &= \left|\sum_{i=1}^{M}(\phi_M^i \otimes \mathbf{I}_N + 2s_i\phi_M^i \otimes \mathbf{K}_x)\right|^{-\frac{1}{2}} \times \\ &\times e^{-\frac{1}{2}(1_M^T \otimes \mathbf{X}^T)\left(\sum_{i=1}^{M}(\frac{1}{2s_i}\phi_M^i \otimes \mathbf{I}_N + \phi_M^i \otimes \mathbf{K}_x)\right)^{-1}(1_M \otimes \mathbf{X})} \\ &= \left|\sum_{i=1}^{M}\phi_M^i \otimes (\mathbf{I}_N + 2s_i\mathbf{K}_x)\right|^{-\frac{1}{2}} \times \\ &\times e^{-\frac{1}{2}(1_M^T \otimes \mathbf{X}^T)\left(\sum_{i=1}^{M}\phi_M^i \otimes (\frac{1}{2s_i}\mathbf{I}_N + \mathbf{K}_x)\right)^{-1}(1_M \otimes \mathbf{X})} \\ &= \prod_{i=1}^{M}|\mathbf{I}_N + 2s_i\mathbf{K}_x|^{-\frac{1}{2}} e^{-s_i\mathbf{X}^T(\mathbf{I}_N + 2s_i\mathbf{K}_x)^{-1}\mathbf{X}} \\ &= \prod_{i=1}^{M} M_{z_i}(s_i) \end{aligned} \quad (14)$$

A equação 14 mostra que, para $\mathbf{X}$ fixo, as componentes $z_i = ||\mathbf{r}^i||^2$ do vetor $\mathbf{z}$ são variáveis aleatórias iid (independentes e idênticamente distribuídas). A função distribuição de probabilidades acumulada (FDP) de z_i é dada pela transformada inversa de lapalace $F_{z_i|\mathbf{X}}(Z_i, \mathbf{X}) = \int_0^{Z_i} p_{z_i|\mathbf{X}}(\lambda, \mathbf{X})d\lambda = \mathcal{L}^{-1}\left\{\frac{1}{s}M_{z_i}(s)\right\}$, e podemos escrever:

$$\begin{aligned} Pr\{||\mathbf{x} - \mathbf{y}^i||^2 > Nd^*|\mathbf{X}\} &= Pr\{z_i > Nd^*|\mathbf{X}\} \\ &= 1 - F_{z_i|\mathbf{X}}(Nd^*, \mathbf{X}) \end{aligned} \quad (15)$$

A probabilidade de casamento condicionada será:

$$P_{m|\mathbf{X}}(N, M, d^*) = 1 - \left(1 - F_{z_i|\mathbf{X}}(Nd^*, \mathbf{X})\right)^M \quad (16)$$

Para M finito, existe uma probabilidade não nula de que um dicionário aleatório não inclua a solução ótima para o problema de compressão na taxa zero, neste caso o vetor nulo. Este dicionário aleatório é semelhante ao que o MMP constrói quando o vetor de entrada é Gaussiano sem memória. Contudo, no MMP o dicionário inicial contém o vetor nulo (de fato, contém todos os vetores cujas componentes são idênticas). Iremos então. antes de prosseguir a nossa análise, substituir um dos elementos do dicionário aleatório pelo vetor nulo 0_N, e avaliar a probabilidade de casamento para o novo dicionário com o zero incluído. Neste caso, a equação 16 torna-se:

$$\begin{aligned} &P_{m|\mathbf{X}}(N, M, d^*, \mathbf{X}) = \\ &\begin{cases} 1 - \left(1 - F_{z_1|\mathbf{X}}(Nd^*, \mathbf{X})\right)^{M-1} & , \ ||X||^2 > Nd^* \\ 1 & , \ ||X||^2 \leq Nd^* \end{cases} \\ &= 1 - \left(1 - F_{z_1|X}(Nd^*, \mathbf{X})\right)^{M-1} S(\mathbf{X}^T\mathbf{X} - Nd^*) \end{aligned} \quad (17)$$

onde $S(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$ é a *função degrau unitário*. Se as componentes do vetor Gaussiano x são iid, sua matriz covariância é $\mathbf{K}_x = \sigma_x^2\mathbf{I}_N$. A equação 14 reduz-se neste caso a:

$$M_{z_i}(s) = (1 + 2s\sigma_x^2)^{-\frac{N}{2}} e^{-\frac{s\mathbf{X}^T\mathbf{X}}{1+2s\sigma_x^2}} \quad (18)$$

Nós então escrevemos a probabilidade de casamento condicionada usando 17 e 18 como:

$$\begin{aligned} P_{m|\mathbf{X}}(N, M, d^*, \mathbf{X}) &= 1 - e^{-\frac{(M-1)(Nd^* + ||\mathbf{X}^T\mathbf{X})}{2\sigma_x^2}} \times \\ &\times \left(\sum_{k=0}^{\infty}\frac{1}{k!}\left(\frac{\mathbf{X}^T\mathbf{X}}{2\sigma_x^2}\right)^k \sum_{l=0}^{k+\frac{N}{2}-1}\frac{1}{l!}\left(\frac{Nd^*}{2\sigma_x^2}\right)^l\right)^{M-1} \times \\ &\times S(\mathbf{X}^T\mathbf{X} - Nd^*) \end{aligned} \quad (19)$$

A equação 19 mostra que no caso sem memória $P_{m|\mathbf{X}}(N, M, d^*)$ é uma função da norma de $\mathbf{X}$ apenas, e não do formato. Por isso podemos calcular $P_m(N, M, d^*)$ tomando o valor esperado com relação à fdp da norma $||\mathbf{x}||$ e não de x. Como as componentes do vetor x são Gaussianas iid de média zero e variância σ_x^2, a sua norma possui distribuição *qui-quadrado* [20]. Assim sendo a probabilidade de casamento se torna:

$$Pm(N, M, d^*) = \int_0^{\infty} P_{m|\lambda}(N, M, d^*, \lambda)\frac{\lambda^{\frac{N}{2}-1}e^{-\frac{\lambda}{2\sigma_x^2}}}{(2\sigma_x^2)^{\frac{N}{2}}\Gamma(\frac{N}{2})}d\lambda \quad (20)$$

onde $P_{m|\lambda}(N, M, d^*)$ é dado pela equação 19 com λ substituindo $\mathbf{X}^T\mathbf{X} = ||\mathbf{X}||^2$.

A função distribuição de probabilidades $F_d(D) = Pr\{d \leq D\}$ da distorção por amostra d é dada pela equação 20 com $d^* = D$. Sendo assim, a função densidade de probabilidades condicionada $f_{d|d\leq d^*}(D)$ de $d|d \leq d^*$, e o valor esperado condicionado $E\{d|d \leq d^*\}$ são:

$$\begin{aligned} f_{d|d\leq d^*}(D) &= \frac{1}{F_d(d^*)}\frac{d}{dD}F_d(D) \\ E\{d|d \leq d^*\} &= \frac{1}{F_d(d^*)}\int_0^{d^*}\xi\frac{d}{d\xi}F_d(\xi)d\xi \\ &= d^* - \frac{1}{F_d(d^*)}\int_0^{d^*}F_d(\xi)d\xi \end{aligned} \quad (21)$$

Se ocorre um casamento, o valor esperado condicionado da distorção total $E\{d|\text{match}\}$ para um vetor de comprimento N é dado pela equação 21:

$$\begin{aligned} Dm(N, M, d^*) &= E\{d|\text{match}\} \\ &= d^* - \frac{1}{F_d(d^*)}\int_0^{d^*}Pm(N, M, \xi)d\xi \end{aligned} \quad (22)$$

8.2 CASAMENTO DE VETORES GAUSSIANOS DE ESCALAS DIFERENTES

Uma característica importante deste trabalho é o casamento aproximado de padrões multiescalas. Portanto é interessante analisar as propriedades de casamento de um vetor Gaussiano de tamanho N aos vetores de um dicionário de vetores Gaussianos de tamanho $N/2, N/4, \ldots, 2$ interpolados para o tamanho N.

Usaremos a notaçãoW $\mathbf{y}^s = T_{N'}^N[\mathbf{y}]$ para denotar a operação de mudança de escala que converte um vetor de tamanho $N' \times 1$ em outro vetor de tamanho $N \times 1$. Uma forma de implementar a transformação de escala é usando uma DCT [21]. Para transformar um vetor $\mathbf{y}$ de tamanho $N' \times 1$ em um vetor $\mathbf{y}^s$ de tamanho $N \times 1$ ($N' \leq N$), primeiramente calcula-se a DCT de N' pontos de $\mathbf{y}$, dada por $\mathcal{Y} = \mathbf{C}_{N'}\mathbf{y}$. Em seguida, adicionam-se $N - N'$ zeros a $\mathcal{Y}$ obtendo-se $\mathcal{Y}^s = \begin{pmatrix} \mathcal{Y} \\ \mathbf{0}_{N-N'} \end{pmatrix}$. Finalmente pode-se obter o vetor interpolado calculando-se a DCT inversa de N pontos de $\mathcal{Y}^s$, $\mathbf{y}^s = \mathbf{C}_N^T\mathcal{Y}^s$. Uma vez que a DCT é uma transformação unitária, a probabilidade de casamento pode ser avaliada no domínio transformado. De fato:

$$\begin{aligned} &Pr\{\|\mathcal{X} - \mathcal{Y}^s\|^2 \leq Nd^*\} = Pr\{\|\mathbf{C}_N\mathbf{x} - \mathbf{C}_N\mathbf{y}^s\|^2 \leq Nd^*\} \\ &= Pr\{(\mathbf{C}_N\mathbf{x} - \mathbf{C}_N\mathbf{y}^s)^T(\mathbf{C}_N\mathbf{x} - \mathbf{C}_N\mathbf{y}^s) \leq Nd^*\} \\ &= Pr\{(\mathbf{x} - \mathbf{y}^s)^T\mathbf{C}_N^T\mathbf{C}_N(\mathbf{x} - \mathbf{y}^s) \leq Nd^*\} \\ &= Pr\{\|\mathbf{x} - \mathbf{y}^s\|^2 \leq Nd^*\} \end{aligned} \quad (23)$$

A DCT é uma transformação linear e por isso a versão transformada do vetor Gaussiano $\mathbf{x}$ é também Gaussiana [20] com matriz covariância dada por:

$$\mathbf{K}_{\mathcal{X}} = E\{\mathcal{X}\mathcal{X}^T\} = E\{\mathbf{C}_N^T\mathbf{x}\mathbf{x}^T\mathbf{C}_N\} = \mathbf{C}_N^T\mathbf{K}_{\mathbf{x}}\mathbf{C}_N \quad (24)$$

Se $\mathbf{x}$ sem memória, então $\mathcal{X}$ também será sem memória. De fato, $\mathbf{K}_{\mathbf{x}} = \sigma^2\mathbf{I}_N$ e $\mathbf{K}_{\mathcal{X}}$ é dado por:

$$\mathbf{K}_{\mathcal{X}} = \mathbf{C}_N^T\sigma^2\mathbf{I}_N\mathbf{C}_N = \sigma^2\mathbf{C}_N^T\mathbf{C}_N = \sigma^2\mathbf{I}_N = \mathbf{K}_{\mathbf{x}} \quad (25)$$

Portanto a probabilidade de casamento de um vetor Gaussiano sem memória $\mathbf{x}$, de tamanho $N \times 1$, considerando-se um dicionário de vetores Gaussianos escalados de tamanho original $N' \times 1$ é igual a probabilidade de casamento de um vetor Gaussiano sem memória de tamanho $N \times 1$ usando um dicionário de vetores Gaussianos de tamanho $N' \times 1$ com $N - N'$ zeros adicionados a cada um dos vetores do dicionário.

$$\begin{aligned} &Pr\{\|\mathbf{x} - T_{N'}^N[\mathbf{y}^i]\|^2 \leq Nd^*\} = \\ &= Pr\{\|\mathbf{x} - \begin{pmatrix} \mathbf{y}^i \\ \mathbf{0}_{N-N'} \end{pmatrix}\|^2 \leq Nd^*\} \\ &= Pr\{\|\mathbf{x}' - \mathbf{y}^i\|^2 + \|\mathbf{x}''\|^2 \leq Nd^*\} \\ &= E_{\mathbf{x}''}\{Pr\{\|\mathbf{x}' - \mathbf{y}^i\|^2 \leq Nd^* - \|\mathbf{X}''\|^2\}\} \end{aligned} \quad (26)$$

onde $\mathbf{x}'$ é um vetor Gaussiano sem memória de tamanho $N' \times 1$ e $\mathbf{X}''$ é uma realização particular de um vetor Gaussiano sem memória $\mathbf{x}''$ de tamanho $N - N' \times 1$.

A probabilidade de casamento multiescalas $Pm^s(N, N', M, d^*)$ é portanto:

$$Pm^s(N, N', M, d^*) = \int_0^{Nd^*} Pm\left(N', M, \frac{Nd^* - \lambda}{N'}\right) \times \times \frac{\lambda^{\frac{N-N'}{2}-1}e^{-\frac{\lambda}{2\sigma_x^2}}}{(2\sigma_x^2)^{\frac{N-N'}{2}}\Gamma(\frac{N-N'}{2})}d\lambda \quad (27)$$

A Figura 11 mostra a probabilidade de casamento multiescalas para um vetor Gaussiano de tamanho $N = 64 \times 1$ usando-se um dicionário de 8192 vetores de comprimentos $N' = 64, 32, 16, \ldots, 2$. É interessante observar que, neste exemplo, a probabilidade de casamento para um dicionário de vetores usando escalas é superior á probabilidade de casamento para um dicionário de vetores de tamanho $N' = 64$. Isto pode ser entendido considerando-se:

- Para $N = 64$ e $M = 8192$ a aproximação para $\mathbf{X}$ usando o dicionário sem escalas terá taxa média $R = 0.2031$ e distorção média $D = 0.9747$. Neste nível de distorção, uma versão passa-baixas de $\mathbf{X}$ será normalmente suficiente para aproximar a fonte. Portanto, há pouco a ser ganho, em termos de redução da distorção, se for usado um dicionário de vetores de tamanho $N' = 64$ ao invés de outro com vetores de tamanho $N' = 32$, por exemplo.
- O casamento multiescalas pode ser visto como o casamento simultâneo de vetores Gaussianos de tamanho N' e um casamento de um vetor Gaussiano de tamanho $N - N'$ com o vetor nulo. A medida que N' diminui, torna-se mais fácil o casamento dos vetores Gaussianos de tamanho N' porque, como o número de vetores do dicionário M é mantido fixo, o número de bits por componente aumenta. Por outro lado, torna-se mais difícil fazer o casamento do vetor Gaussiano de comprimento $(N - N')$ com o vetor nulo, porque a energia média do vetor aumenta linearmente com o comprimento. No exemplo ilustrado na Figura 11, a probabilidade de casamento é máxima quando $N' = 32$ ou $N' = 16$.

A Figura 12 mostra a distorção média dado um casamento multiescalas para um vetor Gaussiano sem memória de tamanhos 64×1 usando um dicionário contendo 8192 vetores de comprimentos $64, 32, 16, \ldots, 2$. Como a probabilidade de casamento é maior neste caso, a distorção média é menor para o casamento multiescalas.

Podemos então concluir que *o casamento multiescalas pode melhorar o desempenho do casamento aproximado de padrões em taxas baixas.*

REFERÊNCIAS

[1] W. B. Pennebaker and J. L. Mitchell, "JPEG Still Image Compression Standard", Van Nostrand Reinhold, New York, 1993.
[2] J. L. Mitchell, W. B. Pennebaker, Chad E. Fogg and D. J. LeGall, "MPEG Video Compression Standard", Chapman & Hall, New York, 1997.
[3] ISO/IEC JTC1/SC29/WG1 (ITU/T SG28), "JPEG2000 Verification Model 5.3", 1999.

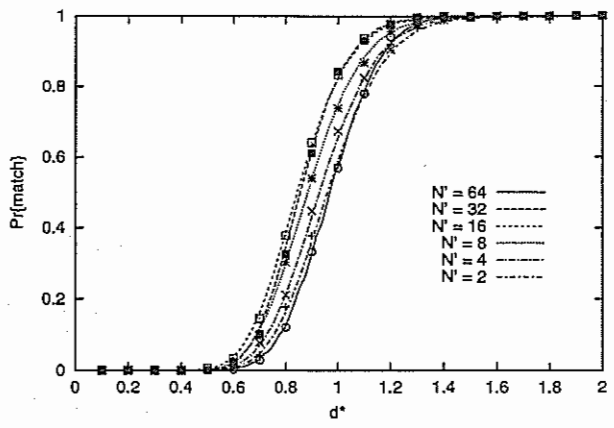


Figura 11. Casamento com escalas para um dicionário com 8192 vetores.

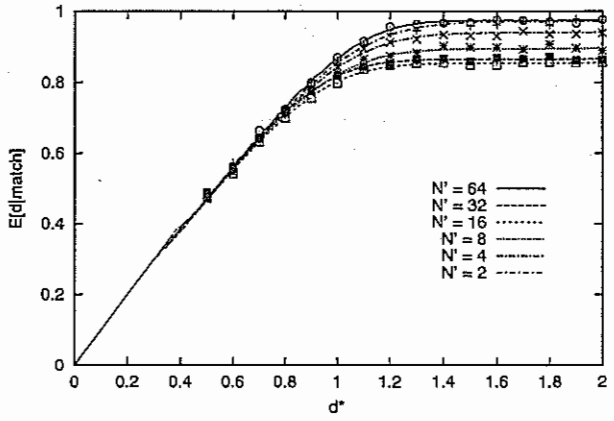


Figura 12. Distorção em um casamento com escalas para um dicionário com 8192 vetores.

[4] L. Bottou, P. Haffner, P. G. Howard, P. Simard, Y. Bengio and Y. LeCun, "High quality document image compression with Djvu," Journal of Electronic Imaging, VOL. 7, NO. 3, pp. 410-428, 1998.

[5] M. B. de Carvalho and E. A. B. da Silva, "A universal multidimensional lossy compression algorithm," 1999 IEEE International Conference on Image Processing, October 1999.

[6] J.Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," IEEE Transactions on Information Theory, VOL. it-24, NO. 5, pp. 530-536, September 1978.

[7] M. B. de Carvalho and W. A. Finamore, "Lossy Lempel-Ziv on subband coding of images," IEEE International Symposium of Information Theory, June 27 - July 1, 1994, Trondheim, Norway.

[8] C. Chan and M. Vetterli, " Lossy compression of individual signals based on string matching and one pass codebook design," Proceedings of ICASSP'95, pp. 2491-2494, Detroit, 1995.

[9] 4. M. Atallah, Y. Génin, and W. Szpankowski, " Pattern Matching Image Compression: Algorithmic and Empirical Results, "IEEE Transactions on Pattern Analysis and Machine Intelligence, 21, 618-627, 1999.

[10] M. Alzina, W. Szpankowski, and A. Grama, " 2D-Pattern Matching Image and Video Compression, " IEEE Transactions on Image Processing, 2001.

[11] M. Effros, P. A. Chou and R.M. Gray, "One-pass adaptive universal vector quantization," Proceedings of ICASSP'94, VOL. 5, pp. 625-628, Adelaide, 1994.

[12] C. Constantinescu C. and J. A. Storer, "Improved techniques for single-pass adaptive vector quantization," Proceedings of the IEEE, VOL. 82, NO. 6, pp. 933-939 June 1994.

[13] M. B. de Carvalho, E. A. B. da Silva and W. A. Finamore, "Rate Distortion Optimized Adaptive Multiscale Vector Quantization," 2001 IEEE International Conference on Image Processing, Thessaloniki, October 2001.

[14] R. A. Blahut, " Principles and Practice of Information theory," Addison-Wesley publishing Company, 1988.

[15] G. J. Sullivan and R. L. Baker, "Efficient quadtree coding of images and video," *IEEE Transactions on Image Processing*, vol.3, No. 3, pp. 327-331, May 1994.

[16] A.Said and W.A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.6, pp.243–250, June 1996.

[17] M. Antonini, M. Barlaud, P. Mathieu, and J. Daubechies, "Image coding using wavelet transform," *IEEE Transactions on Image Processing*, Vol. 1, pp.205-221, April 1992.

[18] P. P. Vaidyanathan, "Multirate Systems and Filter Banks," Prentice-Hall Inc., 1993.

[19] V. K. Goyal and M. Vetterli, "Quantized Overcomplete Expansions in $\mathbb{R}^N$: Analysis, Synthesis, and Algorithms," *IEEE Transactions on Information Theory*, Vol. 44, No 1, pp. 16–31 January 1998.

[20] A. Papoulis, "Probability, Random Variables and Stochastic Processes,", McGRAW-HILL BOOK COMPANY, 1984.

[21] K. H. Tan, M. Ghanbari, "Layered image coding using the DCT pyramid," *IEEE Transactions on Image Processing*, Vol. 4, No 4, pp. 512-516, April 1995.

Murilo B. de Carvalho nasceu em Petrópolis - RJ, Brasil em 1964. Formou-se Engenheiro de Telecomunicações pela Universidade Federal Fluminense (UFF) em 1986. Obteve os títulos de Mestre em em Ciências em Engenharia Elétrica (Sistemas de Comunicações) pela Pontifícia Universidade Católica do Rio de Janeiro (PUC-RJ) em 1994 e Doutor em Ciências em Engenharia Elétrica (Processamento de Sinais) pela Universidade Federal do Rio de Janeiro (COPPE/UFRJ) em 2001. Atualmente é professor adjunto do departamento de Engenharia de Telecomunicações da UFF. Seus pricipais interesses incluem processamento digital de imagens e vídeo, codificação de canal e fonte e processamento de sinais.

Eduardo A. B. da Silva nasceu no Rio de Janeiro, Brasil em 1963. Ele se formou Engenheiro Eletrônico pelo Instituto Militar de Engenharia em 1984, Mestre em Ciências em Engenharia Elétrica pela COPPE/UFRJ em 1990 e Ph.D. em Eletrônica pela Universidade de Essex, Inglaterra, em 1995. Em 1987 e 1988 ele trabalhou no Departamento de Engenharia Elétrica do Instituto Militar de Engenharia. Desde 1989 ele trabalha no Departamento de Eletrônica da Escola de Engenharia da UFRJ. Além disso, ele trabalha, desde 1996, no Programa de Engenharia Elétrica da COPPE/UFRJ. Ele é Editor Associado da IEEE Transactions on Circuits and Systems, Parte I. Seus interesses em pesquisa estão nas áreas de processamento digital de sinais e imagens, especialmente em codificação de sinais, transformadas wavelet, morfologia matemática e aplicações em telecomunicações.

Weiler Alves Finamore formou-se Engenheiro Elétrico pela Pontifícia Universidade Católica do Rio de Janeiro (PUC-RJ) em 1969. Obteve os títulos de M.Sc. e Ph.D., ambos em Engenharia Elétrica, pela universidade de Winsconsin-Madison em 1974 e 1978. De 1970 a 1979 ele foi professor do Departamento de Engenharia Elétrica da Universidade Federal de Belém do Pará. Passou 3 anos como pesquisador visitante no Centro Científico Rio da IBM Brasil (1990, 1991 e 1994). Desde 1979 leciona no CETUC-PUC-RJ, estando envolvido em ensino e pesquisa em comunicações, teoria da informação e processamento de imagens.