Quantização Vetorial adaptativa com multiresolução

Murilo B. de Carvalho

Depto. de Eng. de Telecomunicações
Universidade Federal Fluminense
R. Passos da Pátria, 156
Niteroi - RJ
24210-240, BRASIL
murilo@lps.ufrj.br

Eduardo A. B. da Silva

PEE/COPPE/DEL/EE
Universidade Federal do Rio de Janeiro
Cx. P. 68504,
Rio de Janeiro, RJ
21945-970, BRASIL
eduardo@lps.ufrj.br

Weiler Alves Finamore
CETUC
Pontifícia Universidade Católica do Rio de Janeiro
Rio de Janeiro, RJ
BRASIL
weiler@cetuc.puc-rio.br

Sumário—

Neste artigo é apresentado um novo algorítmo de quantização vetorial adaptativa. O método utiliza uma abordagem de múltiplos dicionários, cada um correspondendo a uma resolução diferente. Um procedimento recursivo de segmentação é utilizado para atualizar os dicionários com concatenações, contrações e dilatações de vetores anteriormente codificados. O algoritmo apresenta bom desempenho com uma ampla classe de dados de entrada e, quando aplicado diretamente a imagens digitais supera alguns algoritmos de compressão especializados como o JPEG.

1 Introdução

A quantização vetorial (QV) é uma técnica largamente utilizada na solução do problema de compressão de dados. O quantizador vetorial é constituido de um conjunto de vetores $\mathcal{C} = \{\mathbf{s}_0, \ldots, \mathbf{s}_{I-1}\}$ chamado dicionário e uma medida de distância entre dois vetores $d(\mathbf{X}, \mathbf{Y})$. A QV é usada para aproximar um vetor de entrada $\mathbf{X}$ escolhendo-se o vetor $\mathbf{s}_i \in \mathcal{C}$ que minimiza a distância $d(\mathbf{X}, \mathbf{s}_i)$. A escolha do dicionário $\mathcal{C}$ afeta diretamente o desempenho do QV e portanto é usualmente feita usando-se alguma técnica de otimização. Quando o vetor $\mathbf{X}$ de entrada é estatisticamente bem conhecido, existem métodos de projeto como o LBG [1] e o ECVQ (Entropy-Constrained Vector Quantizer) [2] que levam a bons dicionários. Contudo, se a fonte é desconhecida ou a estatística varia, o desempenho destes esquemas piora bastante.

Neste artigo, nós propomos o MMP (Multi-dimensional Multiscale Parser), um algoritmo para quantização vetorial adaptativa de uma fonte multi-dimensional. No MMP, não há nescessidade de otimizar previamente os dicionários porque estes são construidos enquanto o MMP codifica os dados. A versão unidimensional do MMP opera em um vetor $\mathbf{X} = (X_0 \ldots X_{N-1})^T$ emitido por uma fonte vetorial $\mathbf{x} = (x_0 \ldots x_{N-1})^T$. Nós assumimos que $N = 2^K$, onde K é um inteiro. O MMP usa $K+1$ dicionários $\{\mathcal{C}^{(0)}, \ldots, \mathcal{C}^{(K)}\}$ de vetores, inicializados com $\mathcal{C}_0^{(k)} = \{\mathbf{s}_0^{(k)}, \ldots, \mathbf{s}_{I_k-1}^{(k)}\}, k = 0, 1, \ldots, K$, e uma distorção alvo d^*. Os vetores em $\mathcal{C}^{(k)}$ são de comprimento 2^k, ou seja, eles estão na *escala* 2^k. O MMP começa procurando no dicionário de maior escala, $\mathcal{C}^{(K)}$, pelo vetor $\mathbf{s}_i^{(K)}$ que minimiza o erro quadrático $\xi = \|\mathbf{X} - \mathbf{s}_i^{(K)}\|^2$. Se o erro quadrático ξ é menor ou igual a distorção alvo d^* vezes o comprimento do vetor N, então a busca termina. Caso contrário, $\mathbf{X}$ é dividido em dois segmentos de igual comprimento $\mathbf{X}' = (X_0 \ldots X_{N/2-1})^T$ e $\mathbf{X}'' = (X_{N/2} \ldots X_{N-1})^T$. Então o MMP busca no dicionário $\mathcal{C}^{(K-1)}$ de escala $N/2$ pelas melhores aproximações para os dois segmentos, ou seja, o

mesmo procedimento que foi aplicado ao vetor $\mathbf{X}$ é aplicado recursivamente a $\mathbf{X}'$ e $\mathbf{X}''$. Esta técnica de segmentação recursiva é semelhante à utilizada no "Universal Multiscale Matching Pursuits" [3], que é um algoritmo de compressão de dados com perdas baseado em expansões de sinais em frames. Após retornar das chamadas recursivas, nós teremos duas aproximações $\hat{\mathbf{X}}'$ e $\hat{\mathbf{X}}''$. Forma-se então uma estimativa do vetor original $\hat{\mathbf{X}}$, na escala N, por concatenação, ou seja, $\hat{\mathbf{X}} = (\hat{\mathbf{X}}'^T \hat{\mathbf{X}}''^T)^T$. O dicionário $\mathcal{C}^{(K)}$ é atualizado pela inclusão de $\hat{\mathbf{X}}$. Uma vez que o desempenho de um dicionário, num sentido taxa versus distorção, tende a melhorar quando seu tamanho aumenta, o MMP também atualiza todos os dicionários das outras resoluções. Para isso o MMP muda a escala do vetor $\hat{\mathbf{X}}$ para as escalas $2^k, k = 0, 1, \ldots, K-1$ usando uma transformação de escala $\hat{\mathbf{X}}^k = T_{2^k}^{(2^K)}[\hat{\mathbf{X}}^K]$. A transformação de escala é uma função $T_N^M : \mathbb{R}^M \to \mathbb{R}^N$ que mapeia um vetor de M componentes em um vetor de N componentes. No restante deste artigo, por simplicidade, eliminaremos o índice superescrito M e escreveremos apenas $T_N[\mathbf{X}]$ para representar a transformação de escala, sempre que o numero de componentes de $\mathbf{X}$ estiver claro no contexto. Este método de atualização leva a construção de bons dicionários desde que a transformação T_N^M seja adequadamente escolhida. Por exemplo, considere-se o caso de o vetor de entrada $\mathbf{X}$ ser extraído de uma fonte Gaussiana sem memória. Quando $N > M$ uma operação de filtragem anti-aliazing segida de decimação [6] levará a um vetor de dimensão reduzida porém também Gaussiano sem memória. Portanto, atualizar o dicionário da escala k' com versões decimadas de vetores da escala maior k'' $(k'' > k')$ é equivalente a atualizar com vetores da própria escala k'. Quando $N < M$, uma interpolação gera um vetor Gaussiano, porém com memória. Ainda assim, podemos mostrar que um dicionário composto de vetores Gaussianos na escala k', interpolados da escalak'' $(k'' > k')$, possui propriedades de codificação similares às de um dicionário de vetores Gaussianos sem memória na escala k', podendo inclusive superar o dicionário sem memória em taxas baixas.

Este algoritmo é facilmente estendido para operar em matrizes e conjuntos de dados multi-dimensionais diretamente ao invés de vetores. Sua versão bidimensional, adequada para o processamento de imagens digitais, é descrita a seguir.

2 Descrição do algoritmo MMP em duas dimensões, o 2D-MMP

O algoritmo MMP descrito na seção anterior pode, em principio, ser aplicado a matrizes e conjuntos de dados multi-dimensionais sem nenhuma alteração. Contudo, pode ser vantajoso modificar a regra de segmentação para explorar melhor as correlações multidimensionais. No caso de uma imagem digital, o MMP irá operar sobre uma matriz $\mathbf{X}$ de dimensão $N \times M$, onde $N = 2^K, M = 2^L$. Os dicionários possuirão duas escalas, ou seja $C^{(k,l)}$ onde $k \in [0, K]$ e $l \in [0, L]$. Varias alternativas de segmentação são possíveis. A regra adotada neste trabalho consiste em segmentar as linhas de $\mathbf{X}$ (ou seja, dividir $\mathbf{X}$ em duas matrizes com a metade do numero de linhas de $\mathbf{X}$ e mesmo número de colunas) quando $N > M$, e segmentar as colunas quando $N \leq M$. Isto é, quando $k > l, k \leftarrow k-1$ e quando $k \leq l, l \leftarrow l-1$. O algoritmo detalhado é descrito a seguir:

Sejam:

- $\mathbf{X}$ uma matiz $N = 2^K \times M = 2^L$.
- $\mathcal{C}_0^{(k,l)} = \{\mathbf{s}_0^{(k,l)}, \ldots, \mathbf{s}_{I-1}^{(k,l)}\}$, $(k,l) = (0,0), (1,0), (1,1) \ldots, (K,L)$ um conjunto de $K+L+1$ dicionários iniciais contendo cada um $I_{k,l}$ matrizes de tamanho $2^k \times 2^l$.
- d^* uma distorção alvo.
- $T_{N,M}[\mathbf{X}]$ uma transformação de escala que mapeia a matriz $\mathbf{X}$ em uma matriz de tamanho $N \times M$.

Procedure $\hat{\mathbf{X}} = \text{encode}(\mathbf{X}, \mathcal{D}, d^*)$:

step 1 Encontre o índice i no codebbok de mesma escala (k,l) de $\mathbf{X}$ de modo que $||\mathbf{X} - \mathbf{s}_i^{(k,l)}||$ seja mínimo e faça $\hat{\mathbf{X}} = \mathbf{s}_i^{(k,l)}$.

step 2 `if` $N = M = 1$, transmita o índice i e retorne $\hat{\mathbf{X}}$.
`else` vá ao passo 4.

step 3 `if` $||\mathbf{X} - \hat{\mathbf{X}}||^2 \leq NMd^*$ `then` transmita flag '1', transmita o índice i e retorne $\hat{\mathbf{X}}$.
`else` vá para o passo 5.

step 4 transmita flag '0'.

step 5 `if` $N > M$ segmente $\mathbf{X} = \begin{pmatrix} \mathbf{X}' \\ \mathbf{X}'' \end{pmatrix}$,
onde $\mathbf{X}'$ e $\mathbf{X}''$ são $N/2 \times M$
`else` segmente $\mathbf{X} = \begin{pmatrix} \mathbf{X}' & \mathbf{X}'' \end{pmatrix}$
onde $\mathbf{X}'$ e $\mathbf{X}''$ são $N \times M/2$

step 6 calcule $\hat{\mathbf{X}}' = \text{encode}(\mathbf{X}', \mathcal{D}, d^*)$

step 7 calcule $\hat{\mathbf{X}}'' = \text{encode}(\mathbf{X}'', \mathcal{D}, d^*)$

step 8 `for` $n = 0, 1, \ldots, K+L+1$,
$i = \lfloor (\frac{n+1}{2}) \rfloor$,

$j = \lfloor (\frac{n}{2}) \rfloor$,
$\mathcal{C}^{(i,j)} = \mathcal{C}^{(i,j)} \cup \left\{ T_{2^i,2^j}[\hat{\mathbf{X}}] \right\}$
onde $\lfloor x \rfloor$ denota o maior inteiro menor ou igual a x .

step 9 retorne $\hat{\mathbf{X}}$

3 Resultados experimentais

O algoritmo descrito na seção 2 foi implementado em um programa de computador para simulação. A transformação $T_{mn}(g)$ usada foi uma operação clássica de mudança de taxa de amostragem usando um interpolador linear como filtro [6]. Por exemplo, no caso unidimensional, para mudar do tamanho m para o tamanho M nós primeiramente mudamos para um tamanho intermediário mM através de uma interpolação linear. Então se $m < M$ este vetor mM é subamostrado para o tamanho M retendo-se apenas M amostras igualmente espaçadas. Se $m > M$, o vetor mM é dividido em M blocos de tamanho m e a média de cada bloco é calculada. Os M valores de média dos M blocos irão compor o vetor de tamanho M. No caso bidimensional, nós primeiro mudamos do tamanho $m \times n$ para o tamanho $M \times n$ aplicando o procedimento unidimensional descrito a cada coluna da matriz $m \times n$. Em seguida nós mudamos para o tamanho $M \times N$ operando analogamente nas colunas.

A distorção alvo pode ser escolhida em qualquer ponto do intervalo $[0, \infty)$, e a distorção efetivamente obtida após a decodificação será menor ou igual a d^* desde que o dicionário inicial na escala 1×1, $\mathcal{C}_0^{(0)}$, possua resolução suficiente. Por exemplo, para que o MMP possua capacidade *lossless*, ou seja, possua capacidade de reproduzir o vetor de entrada com distorção nula, deve-se ter $\mathcal{A} \subset \mathcal{C}_0^{(0)}$, onde $\mathcal{A}$ é o alfabeto (finito) aonde as componentes do vetor de entrada estão definidas (neste caso a entrada poderá ser aproximada com distorção zero, nem que para isto todas as componentes do vetor de entrada sejam representadas uma a uma, com distortorção nula, usando o dicionário da escala 1×1). Para lidar com a disponibilidade finita de memória, um esquema de podagem foi implementado: Cada novo elemento incluido nos dicionários é associado a um contador que é inicializado com zero no momento da inclusão. Quando um elemento de um dicionário é selecionado para representar o vetor de entrada, seu contador associado é zerado enquanto todos os demais contadores são incrementados de um. Quando o dicionário atinge um determinado tamanho pré especificado, o elemento cujo contador associado possui o maior valor é retirado do dicionário sempre que for necessário incluir um novo elemento. Os índices dos dicionários foram codificados por um codificador aritmético adaptativo com um modelo estatístico independente para cada dicionário. Os flags indicadores de segmentação também foram codificados pelo codificador aritmético adaptativo com modelos independentes por escala.

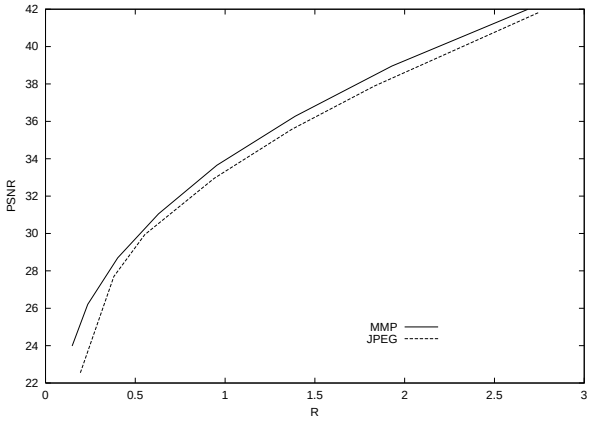

Figura 1: Desempenho para LENA 256×256.

A figura 1 mostra resultados de simulação para a imagem LENA 256×256. Os parâmetros usados foram: $|\mathcal{C}^{(k,l)}| \leq 8192$, (k,l) = (0,0), (1,0), (1,1), (2,1), ... , (3,3) e dicionário inicial na escala 1×1 igual a
$\mathcal{C}_0^{(0,0)} = \{-124, -120, \ldots, -4, 0, 4, \ldots, 120\}$. Os dicionários iniciais nas demais escalas foram obtidos a partir deste dicionário usando a transformação de escala. A imagem foi inicialmente dividida em blocos de tamanho 8×8 sem superposição, que foram processados sequencialmente pelo algoritmo. A medida de distorção indicada na figura é a relação sinal-ruído de pico PSNR (Peak Signal-to-Noise Ratio).
$PSNR = 10 \log \left(\frac{255^2}{\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (s_{ij} - \hat{s_{ij}})^2} \right)$. O resultado obtido com o algoritmo JPEG [4] também é mostrado. Observa-se que o MMP, sem nenhuma otimização a priori de dicionários, supera o desempenho do JPEG que é um algoritmo específico otimizado para imagens. Deve-se ressaltar que quantizadores vetoriais obtidos com o uso dos métodos tradicionais como o LBG e o ECVQ não apresentam bom desempenho quando aplicados diretamente a este tipo de fonte.

A figura 2 mostra os resultados para a fonte Gaussiana. Esta fonte foi construída a partir de 65536 amostras de um processo estocástico de ruído branco Gaussiano de variância $\sigma^2 = 957$, compondo uma matriz 256×256. Resultados para o JPEG, ECVQ e a distorção mínima teórica dada por $D(R) = \sigma^2 2^{-2R}$ [5] também são mostrados. Aqui a superioridade do MMP é ainda maior porque o

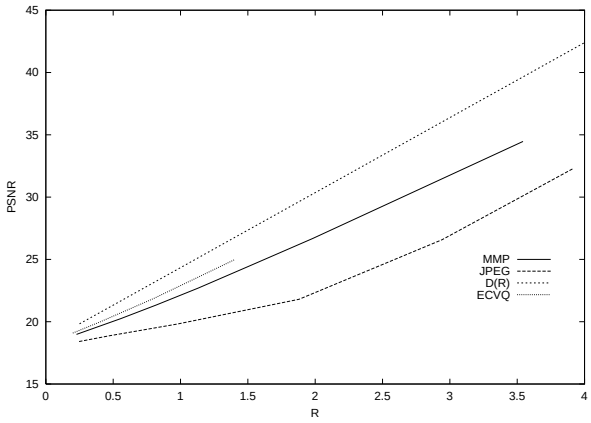

Figura 2: PSNR × taxa para fonte Gaussiana 256× 256.

JPEG não foi otimizado para fonte Gaussiana. O desempenho mostra-se bom mesmo quando comparado com QVs tradicionais otimizados para esta fonte, como o ECVQ.

4 Efeito de blocagem no MMP

Inicialmente, o MMP busca o elemento $\mathbf{s}_i^{(K)}$ em seu dicionário de maior resolução $\mathcal{C}^{(K)}$ que representa o vetor de entrada $\mathbf{X}$ com a menor distorção $\xi = \|\mathbf{X}-\mathbf{s}_i^{(K)}\|^2$. Se a distorção obtida ξ for maior que a distorção alvo d^*, então o MMP irá segmentar o vetor de entrada e tentará aproximar cada metade usando seu dicionário de escala imediatamente inferior $\mathcal{C}^{(K-1)}$. Quando as aproximações para ambos os segmentos $\hat{\mathbf{X}}'$ e $\hat{\mathbf{X}}''$ estiverem disponíveis, o MMP irá criar um novo vetor a ser incluido no dicionário $\mathcal{C}^{(K)}$ *concatenando* $\hat{\mathbf{X}} = (\hat{\mathbf{X}}'\hat{\mathbf{X}}'')$. Este procedimento garante que a distorção ξ será menor ou igual a d^*, uma vez que as distorções em cada segmento $\xi' = \|\mathbf{X}' - \hat{\mathbf{X}}'\|^2$ e $\xi'' = \|\mathbf{X}'' - \hat{\mathbf{X}}''\|^2$ também sejam menores ou iguais a d^*. No entanto, o MMP não tem nenhum controle sobre a continuidade da aproximação na fronteira dos dois segmentos concatenados, a menos que a distorção alvo seja zero e o dicionário de escala 1×1 contenha todo o alfabeto da fonte. De fato, a aproximação usando concatenação pode ser vista como uma soma de dois vetores $\hat{\mathbf{X}}'_s = \begin{pmatrix} X'_0 & \dots & X'_{N/2-1} & 0 & \dots & 0 \end{pmatrix}$ e $\hat{\mathbf{X}}''_s = \begin{pmatrix} 0 & \dots & 0 & X''_{N/2} & \dots & X''_{N-1} \end{pmatrix}$ cujas componentes *não apresentam superposição*. Se nós quisermos que a aproximação seja mais suave no ponto de concatenação para $d^* > 0$, podemos usar vetores com superposição. Contudo ao usar superposição, não podemos mais garantir que os dois segmentos $\hat{\mathbf{X}}'$ e $\hat{\mathbf{X}}''$, cada um com distorção inferior ou igual à distorção alvo, irão gerar um bloco $\hat{\mathbf{X}}$ com distorção inferior a d^*. Portanto, quando há superposição, a escolha de um elemento no dicionário para representar $\mathbf{X}'$ afeta a escolha para $\mathbf{X}''$ e vice-versa.

Para evitar o dificil problema de otimização conjunta, podemos usar vetores sem superposição para analisar o vetor $\mathbf{X}$ no codificador e outra com superposição para sintetisar o vetor $\hat{\mathbf{X}}$ no decodificador. Por exemplo, suponhamos que o MMP segmente o vetor de entrada $\mathbf{X}$ em quatro segmentos de mesmo comprimento e aproxime cada um usando um vetor cujas componentes sejam todas iguais, ou seja, vetores representando patamares constantes. O decodificador deve reconstruir o vetor de entrada a partir dos quatro índices de dicionário representando os patamares constantes. Isto é análogo a recuperar uma aproximação $\hat{\mathbf{X}}$ para $\mathbf{X}$ a partir de sua versão decimada por $N/4$. É sabido da teoria de bancos de filtros [6] que é possível recuperar uma versão passa-baixas de $\mathbf{X}$ a partir de sua versão decimada usando um filtro de síntese diferente do filtro de análise. Assim, nós podemos usar vetores para reconstrução com superposição que atendam a algum requisito de suavidade e vetores diferentes, sem superposição, para análise.

Neste trabalho nós usamos uma técnica de pós-filtragem adaptativa FIR para modificar o formato dos vetores de reconstrução no decodificador do seguinte modo:

- Primeiramente, nós utilizamos o decodificador padrão para encontrar uma primeira aproximação para $\mathbf{X}$. Ou seja, $\hat{\mathbf{X}} = \begin{pmatrix} \mathbf{s}_{i_0}^{(N_0)} & \mathbf{s}_{i_1}^{(N_1)} & \dots & \mathbf{s}_{i_{p-1}}^{(N_{p-1})} \end{pmatrix}$ será composto de uma concatenação de p segmentos, cada um uma versão de um vetor $\mathbf{s}_{i_k}$ no dicionário escalada para o tamanho N_k.

- Em seguida, nós substituímos cada $\mathbf{s}_{i_k}^{(N_k)}$ por uma concatenação de vetores do dicionário inicial. Ou seja, $\hat{\mathbf{X}} = \begin{pmatrix} \mathbf{s}_{i_0}^{(L_0)} & \mathbf{s}_{i_1}^{(L_1)} & \dots & \mathbf{s}_{i_{q-1}}^{(L_{q-1})} \end{pmatrix}$, onde os vetores $\mathbf{s}_{i_k}$ pertencem ao dicionário inicial $\mathcal{D}_0$.

- Finalmente, nós aplicamos um filtro de média móvel e retardo zero a $\hat{\mathbf{X}}$. O comprimento do filtro em cada componente $\hat{X}_n$ de $\hat{\mathbf{X}} = \begin{pmatrix} \hat{X}_0 & \dots & \hat{X}_{N-1} \end{pmatrix}$ é proporcional ao comprimento do segmento $\mathbf{s}_{i_k}^{(L_k)}$ que contém a amostra $\hat{\mathbf{X}}_n$, ou seja L_k. Assim sendo, o comprimento do filtro é ajustado amostra a amostra. Considerando novamente, por exemplo, que o vetor reconstruído $\hat{\mathbf{X}}$ possui quatro segmentos de vetores constantes, e o filtro de média móvel possui comprimento $L_k +$

Figura 3: MMP sem controle de blocagem.

1, este procedimento substitui os quatro segmentos planos sem superposição de tamanho $N/4$ por quatro segmentos triangulares com superposição e de tamanho $N/2$.

A figura 4 ilustra o processo. Nesta figura, o vetor de saída é composto de três segmentos. A componente sendo filtrada está indicada pela seta. Como pode ser visto, o comprimento do filtro FIR é proporcional ao tamanho do vetor de reconstrução original.

A figura 3 ilustra a imagem LENA obtida com MMP sem controle de blocagem e a figura 5 ilustra a imagem lena obtida com controle de blocagem. Ambas as imagens foram codificadas a uma taxa de 0.40 bits por pixel. As relações sinal-ruído de pico são respectivamente 28.7 e 28.5 dB.

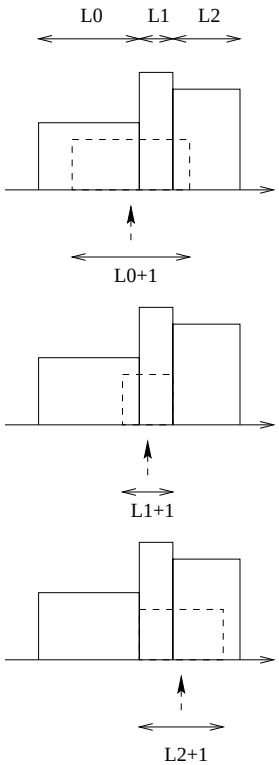

Figura 4: FIR Post-filtering process.

5 Conclusão

Neste artigo apresentamos um novo algoritmo para quantização vetorial adaptativa de fontes multi-dimensionais. Resultados de simulações com fontes Gaussianas mostram que uma versão bidimensional do MMP apresenta desempenho comparável a de quantizadores vetoriais tradicionais otimizados para a fonte Gaussiana. É sabido que quantizadores vetoriais obtidos por métodos tradicionais como o LBG e o ECVQ não apresentam bons resultados quando aplicados diretamente a

Figura 5: MMP com controle de blocagem.

imagens. Entretanto, o MMP obteve um desempenho superior ao do algoritmo JPEG que é otimizado para imagens. O MMP possui também generalizações diretas para dimensões superiores. Ele apresenta bom desempenho com uma larga classe de dados, mas espera-se melhorias se os dicionários iniciais forem otimizadas para o uso com uma classe mais restrita de fontes, como imagens. O problema de blocagem foi abordado e uma solução, baseada na superposição das componentes dos vetores de reconstrução, foi proposta. Contudo esperamos obter melhorias no desempenho se a forma dos vetores de reconstrução for otimizada com respeito aos vetores usados para análise.

Referências

[1] Y. Linde, A. Buzo and R. M. Gray "An algorithm for vector quantizer design," IEEE Transactions on Communications, VOL. COM-28, No. 1, January, 1980.

[2] P. A. Chou, T. Lookabaugh and R. M. Gray, "Entropy-constrained vector quantization," IEEE Transactions on Acoustics Speech and Signal Processing, VOL. 37, No. 1, January 1989.

[3] M. B. de Carvalho e E. A. B. da Silva, "Compressão de sinais multi-dimensionais via decomposição em base e casamento de padrões," 17^o Simpósio Brasileiro de Telecomunicações, 1999.

[4] W. Pennebaker and J. Mitchel, "JPEG Still Image Data Compression Standard," Van Nostrand Reinhold, 1994.

[5] R. E. Blahut, "Principles and Practice of Information Theory" Addison-Wesley publishing Company, 1988.

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