SPIHT Pós Quantização: Uma Alternativa para Codificação de Vídeo

Marcos Ricardo A. Morais, Francisco Madeiro e Elmar Uwe K. Melcher

***Resumo*— Este artigo apresenta uma técnica de codificação sem perda de coeficientes quantizados de vídeo, oriundos da aplicação de transformadas de bloco. O algoritmo proposto, denominado SPQ (*SPIHT pós quantização*), baseia-se no reagrupamento em subbandas sugerido inicialmente por Xiong *et al.*, do qual difere pelo fato de que não utiliza a quantização intrínseca do algoritmo SPIHT, aplicando a codificação aos elementos já quantizados. São apresentados resultados de simulações envolvendo codificação de vídeo em um arcabouço de um codificador MPEG-4, modificado para acomodar o algoritmo SPIHT. Os resultados mostram o mérito do SPQ em termos de redução do número de bits utilizados na codifcação.**

***Palavras-Chave*— Codificação de vídeo, SPIHT.**

***Abstract*— This paper presents a technique for lossless coding of video quantized coefficients, obtained by block transforms. The proposed algorithm, referred to as SPQ (*post quantizing SPIHT*), is based on the subband rearrangement proposed by Xiong *et al.*, from which it differs by the fact the intrinsic SPIHT quantization is not used – the coding is performed in the quantized coefficients. Simulation results concerning video coding are presented, by using an existing MPEG-4 coder, modified to include the SPIHT algorithm. Results show the merit of SPQ in terms of reducing the number of bits used in the coding process.**

***Keywords*— Video coding, SPIHT.**

I. Introdução

As transformadas de bloco, como a DCT (*Discrete Cosine Transform*) e a LT (*Lapped Transform*), que realizam o processamento atuando em pequenas regiões da imagem, têm encontrado aplicação freqüente na compressão de imagem e vídeo, haja visto que apresentam reduzida complexidade computacional e eficiência na compressão [1]–[3]. A transformada discreta de cosseno (DCT) figura no padrão JPEG [4] e nos padrões H.261, H.263 e MPEG [2], [5], [6].

O encadeamento de técnicas de transformação-quantização-codificação forma o paradigma TQC, amplamente utilizado na compressão com perda de variados tipos de sinais, como áudio [7], imagem [1], [4], [8] e vídeo [2], [9].

Em um sistema de compressão de imagem ou vídeo baseado em transformada de bloco, os coeficientes obtidos pela transformação da imagem de entrada são quantizados, e os valores obtidos passam por uma codificação sem perda, também chamada codificação entrópica. Para que os melhores resultados sejam obtidos, deve-se buscar a otimização de todos os passos do sistema. A transformada deve ser adequada ao sinal a ser codificado, concentrando as informações essenciais para o sistema visual humano em poucos coeficientes significativos, de forma que, ao serem desprezados ou quantizados de forma mais grosseira os demais coeficientes, a representação obtida ainda seja satisfatória. A quantização deve ser realizada de forma a representar da melhor maneira possível (com o menor erro de representação) os coeficientes transformados.

Este artigo apresenta uma técnica de codificação sem perda de coeficientes quantizados de vídeo, oriundos da aplicação de transformadas de bloco. O algoritmo proposto, denominado SPQ (*SPIHT pós quantização*), baseia-se no reagrupamento em subbandas sugerido inicialmente por Xiong *et al.* [10], do qual difere pelo fato de que não utiliza a quantização intrínseca do algoritmo SPIHT [11], aplicando a codificação aos elementos já quantizados. A separação entre quantização e codificação no SPQ permite a utilização de quantizadores otimizados, podendo levar a ganhos na codificação.

São apresentados resultados de simulações envolvendo codificação de vídeo em um arcabouço de um codificador MPEG-4, modificado para acomodar o algoritmo SPIHT. A codificação de vídeo natural no padrão MPEG-4 utiliza a seqüência de zeros, RL (*Run Length*), seguida por um código de comprimento variável, VLC (*Variable Length Coding*), com tabelas pré-estabelecidas, ou seja, sem adaptação. Denotaremos essa técnica como RLVLC. O trabalho contempla uma avaliação comparativa de desempenho dos algoritmos SPQ e RLVLC no que diz respeito à taxa de bits obtida.

II. Codificação por Zerotree

A codificação progressiva (*embedded*) por árvore de zeros (*zerotree*) é freqüentemente associada à transformada *wavelet* diádica. Através das características multiresolucionais da transformada *wavelet* é possível desenvolver métodos simples, porém muito eficientes, de codificação de coeficientes transformados.

Os codificadores EZW original (*embedded* por *zerotree* com *wavelet*) [12] e suas variações efetivamente ordenam os coeficientes por planos de bits e transmitem primeiramente os bits mais significativos ao explorar a relação entre o nó pai e seus descendentes em uma árvore *wavelet*. A importância do trabalho está na aplicação conjunta de dois conceitos muito importantes, codificação progressiva e árvore de zeros, resultando uma codificação eficiente com relação à taxa-distorção.

A base do EZW é a estrutura *zerotree*, a qual baseia-se no princípio de que numa decomposição *wavelet* os elementos correspondentes à mesma posição da imagem, nos níveis de decomposição, têm o valor absoluto das suas amplitudes relacionadas. Os coeficientes *wavelet* das subbandas de detalhes possuem, em geral, menor amplitude que os coeficientes das subbandas de aproximação. Se um elemento é maior do que um determinado limiar, dizemos que o elemento é significativo. Numa decomposição *wavelet* de uma imagem, se um elemento não é significativo, há uma alta probabilidade de que os elementos correspondentes à mesma posição espacial não sejam significativos. Estes elementos podem ser visualizados como uma estrutura de árvore. Se toda a árvore é não significante, dizemos tratar-se de uma *zerotree*. Este esquema de codificação resulta uma seqüência de bits progressiva, ou seja, de refinamento sucessivo, aliado a outras vantagens, como o controle exato de taxa de bits e a possibilidade de reconstrução

Marcos Ricardo A. Morais, Universidade Federal de Campina Grande, Campina Grande, PB, Brasil, e-mail: morais@dee.ufcg.edu.br. Francisco Madeiro, Escola Politécnica de Pernambuco – Universidade de Pernambuco, Recife, PE, Brasil, e-mail: franciscomadeiro@yahoo.com.br. Elmar Uwe K. Melcher, Universidade Federal de Campina Grande, Campina Grande, PB, Brasil, e-mail: elmar@dsc.ufcg.edu.br. Francisco Madeiro e Elmar Melcher também estão com o Instituto de Estudos Avançados em Comunicações (Iecom), Campina Grande, PB, Brasil.

perfeita (a reconstrução perfeita só pode ser obtida quando a transformada mapeia inteiros em inteiros).

A abordagem de codificação progressiva baseia-se na idéia de que a informação mais importante (definida como aquela que mais diminui uma certa medida de distorção) deve ser transmitida primeiro. Assumindo que a medida de distorção é o erro médio quadrático (MSE, *mean square error*), que a transformada é ortogonal e que os coeficientes $C_{i,j}$ são transmitidos um a um, é conhecido [12], [13] que o MSE decai de $\frac{1}{N}|C_{i,j}|$, em que N é o número total de pixels. Se um bit é transmitido por vez, esta abordagem pode ser generalizada ao se ordenar os coeficientes por planos de bits e os bits mais significantes serem transmitidos primeiro. O esquema de transmissão progressiva resulta um fluxo de bits que pode ser truncado a qualquer ponto e o decodificador produz uma imagem reconstruída correspondente com qualidade próxima da que seria obtida com a otimização para aquele ponto. O algoritmo EZW pode ser considerado como uma combinação de um quantizador escalar com zona morta e um codificador entrópico para codificar coeficientes *wavelet*.

III. SPIHT

O codificador SPIHT (*Set Partitioning in Hierarchical Trees*) [11] pode ser visto como um aperfeiçoamento do codificador EZW, atingindo um melhor desempenho de taxa-distorção.

O algoritmo SPIHT utiliza três conceitos básicos: (1) codifica/transmite as informações mais importantes inicialmente através de uma representação por plano de bit dos pixels, (2) a transmite ordenadamente dos planos de bit de refinamento e (3) a codificação é realizada ao longo de caminhos/árvores chamadas árvores de orientação espacial, que exploram de forma eficiente as propriedades da imagem transformada pela *wavelet* 2D.

A maioria da energia de uma imagem está concentrada nas componentes de baixa freqüência [11]. Conseqüentemente, a energia (variância) diminui ao passo que nos movemos do nível mais alto da pirâmide (subbanda de freqüência mais baixa) para os níveis mais baixos da pirâmide de subbandas (freqüência mais alta). Além disso, foi observado que há uma similaridade entre subbandas, e espera-se que os coeficientes fiquem ordenados em magnitude (decrescente) se nos movermos para baixo na pirâmide seguindo a mesma orientação espacial. Por exemplo, espera-se que se forem identificadas grandes áreas de pouca atividade no nível mais alto da pirâmide (baixa freqüência, aproximação), estes sejam replicados nos níveis mais baixos nas mesmas posições espaciais. Estes aspectos levaram à introdução do conceito de árvore de orientação espacial, visto a seguir.

O SPIHT consiste de dois estágios principais, ordenação e refinamento. No estágio de ordenação, SPIHT ordena os pixels por magnitude em respeito a um limiar, que geralmente é uma potência de dois, chamado nível de significância. Porém, esta ordenação é parcial, porque não há uma ordem pré-estabelecida entre os coeficientes com o mesmo nível de significância ou maior bit significante. Esta ordenação é baseada no teste de significância dos pixels ao longo das árvores de orientação espacial com raiz no nível mais alto da pirâmide na imagem transformada *wavelet*.

As árvores de orientação espacial foram introduzidas para testar a significância de um grupo de pixels para a compressão eficiente ao explorar as propriedades de similaridade e localidade da magnitude de uma imagem transformada *wavelet* 2D. Em outras palavras, o SPIHT explora a propriedade primeiro observada por Lewis and Knowles [14] de que se um pixel no nível mais alto da pirâmide é insignificante, é muito provável que seus descendentes sejam insignificantes. A Fig. 1 descreve a relação pai-descendentes nas árvores de orientação espacial. Nas árvores de orientação espacial, cada nó consiste de 2×2 pixels adjacentes, e cada pixel no nó tem 4 descendentes, exceto no mais alto nível da pirâmide, em que um pixel em um nó indicado por '*' nessa figura não tem nenhum descendente.

Na implementação prática, o SPIHT mantém três listas, a lista dos pixels insignificantes (LIP, *List of Insignificant Pixels*), a lista dos pixels significantes (LSP, *List of Significant Pixels*) e a lista dos conjuntos insignificantes (LIS, *List of Insignificant Sets*). No estágio de inicialização, o SPIHT inicializa a LIP com todos os pixels do mais alto nível da pirâmide, a LIS com todos os pixels do mais alto nível da pirâmide exceto os pixels que não têm descendentes e a LSP como uma lista vazia. A função básica do algoritmo de ordenação é particionar recursivamente os conjuntos que estão representados na LIS para localizar individualmente os pixels significantes, pixels insignificantes e conjuntos insignificantes menores e mover suas coordenadas para as listas apropriadas, LSP, LIP e LIS, respectivamente. Após cada estágio de ordenamento, o SPIHT apresenta os bits de refinamento do nível atual de significância dos bits dos pixels que foram movidos para a LSP nos limiares mais altos. Desta forma, as magnitudes dos pixels significantes são refinadas com os bits que mais decrescem o erro. Este processo continua, sendo o limiar diminuído sucessivamente por um fator de dois até que a taxa de bits ou a qualidade de imagem desejada seja atingida.

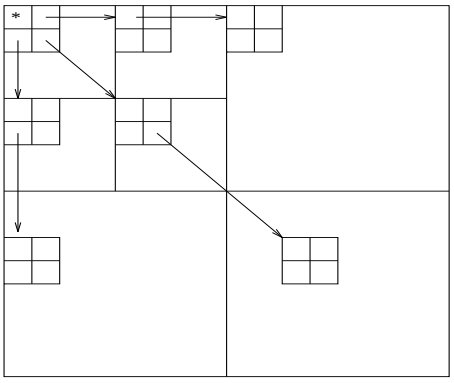

Fig. 1. Árvore de orientação espacial.

Durante a codificação, os pixels na LIP são testados e aqueles que são significantes no nível de quantização atual são movidos para a LSP. De forma similar, os conjuntos são avaliados seqüencialmente seguindo a ordem da LIS, e quando um conjunto é determinado como significante ele é removido da LIS e particionado em novos subconjuntos. Os novos subconjuntos com mais de um elemento são acrescentados de volta ao final da LIS, enquanto as coordenadas dos conjuntos que possuem um único elemento são acrescentados ao final da LIS ou da LSP, dependendo se eles são insignificantes ou significantes, respectivamente. Os conjuntos inseridos na LIS têm ainda uma outra classificação, como conjuntos do tipo A ou do tipo B. Os conjuntos do tipo A englobam toda uma árvore de descendentes, sem incluir o nó pai, que indica apenas a posição inicial da árvore. Os conjuntos do tipo B possuem todos os descendentes excluindo-se os 4 descendentes imediatos. Esta classificação extra simplifica a implementação.

A decodificação é realizada por um algoritmo igual ao da codificação, substituindo-se a saída de bits por entrada. Isto ocorre porque o fluxo de controle do algoritmo é dominado pelas significâncias dos pixels e dos conjuntos, e são essas

significâncias que são enviadas.

IV. Codificação de Vídeo Natural no Padrão MPEG-4

Esta seção aborda a codificação de vídeo natural segundo o padrão MPEG-4, cuja descrição pode ser vista em [2], [15], [16].

O padrão MPEG-4 normalmente trabalha com seqüências de imagens no formato YUV 4:2:0, ou seja, 1 valor de luminância para cada pixel da imagem e 1 par de valores de crominância (U e V) para um bloco de 2×2 pixels da imagem. Para a codificação, cada imagem é dividida em macroblocos. Um macrobloco tem dimensões de 16×16 pixels e é formado por 6 blocos 8×8, 4 blocos de luminância Y e 2 blocos de crominância (U e V).

Nas imagens codificadas como *intra*, ou seja, sem referência a outra imagem da seqüência, utiliza-se um processo muito similar à codificação de imagem do padrão JPEG, em que cada um dos blocos de 8×8 pixels pertencentes aos macroblocos é transformado pela DCT. Os coeficientes DCT são então quantizados (efetivamente divididos por um número inteiro e arredondados). Em seguida, há uma etapa de predição dos coeficientes DC e AC. Esses coeficientes são varridos e alinhados em um vetor segundo uma ordem pré-estabelecida, escolhida entre três opções (zigzag, varredura horizontal e varredura vertical), da banda de freqüência mais baixa para a banda de freqüência mais alta. Com a varredura, aproveita-se a característica de agrupamento de energia da DCT. Faz-se uma codificação de seqüência de zeros (*run-length*) e os valores obtidos são codificados usando uma tabela de código de comprimento variável, indexada por comprimento da seqüência de zeros, valor do coeficiente e se este é o último não nulo. Este processo é muito eficiente mas não explora, além do passo de predição, a dependência estatística entre os blocos.

Nas imagens codificadas como *inter*, realiza-se um processo de estimação de movimento entre a imagem atual e uma imagem de referência, em que se associa a cada macrobloco (ou bloco, dependendo do modo escolhido) um vetor de deslocamento. Esse vetor representa a predição do valor do macrobloco atual partindo da imagem de referência. Os vetores obtidos são codificados e enviados. À formação da imagem de predição dá-se o nome de compensação de movimento. A diferença entre a imagem atual e a imagem compensada denomina-se imagem resíduo. A imagem resíduo é codificada de forma semelhante à codificação de uma imagem *intra*. Realiza-se a transformação DCT seguida por um passo de predição (que só ocorre para os macroblocos *intra* inseridos na imagem *inter*), quantização e codificação entrópica com seqüência de zeros e código de comprimento variável. A alta correlação existente entre imagens subseqüentes leva a uma boa predição realizada pela estimação e compensação de movimento. Assim, a energia da imagem resíduo é pequena, o que leva a pequenos valores de coeficientes DCT. Quando os coeficientes são quantizados, dependendo do coeficiente de quantização utilizado, o número de zeros resultantes pode ser muito alto e os coeficientes não zero geralmente possuem valores bem pequenos. Isto motiva a codificação conjunta dos coeficientes na tentativa de representar o maior número de zeros com o menor número de bits.

V. SPIHT Pós Quantização

Nesta seção veremos a o algoritmo de codificação de transformada de bloco baseado no reagrupamento dos coeficientes em subbandas e no codificador SPIHT. O algoritmo SPIHT engloba uma quantização escalar com zona morta e uma codificação entrópica dos coeficientes através de *zerotrees* e particionamento de conjuntos. A quantização intrínseca do SPIHT não é utilizada aqui. Em vez disso, utiliza-se, no presente trabalho, um quantizador escalar já disponível no MPEG-4. Esta separação permite que o quantizador seja otimizado para a estatística do sinal em questão, usando técnicas como o quantizador de Loyd-Max ou o quantizador ótimo, no sentido taxa-distorção de Ratnakar [17]–[19]. Como o codificador é aplicado após a quantização, denotamos este método por SPQ, ou SPIHT pós quantização.

As transformadas de bloco e as transformadas com sobreposição (*lapped*) produzem um particionamento uniforme do espectro, enquanto a transformada *wavelet* tem uma decomposição do sinal em oitavas. Todas as subbandas de uma transformada de bloco têm o mesmo tamanho. Um nó pai não teria quatro nós de descendentes como no caso de uma representação *wavelet*. Investigando a analogia entre a transformada *wavelet* e a transformada de bloco, como mostrado na Fig. 2, observa-se que o nó pai, os filhos (descendentes imediatos) e os demais descendentes em uma árvore *wavelet* cobrem a mesma localidade espacial, e isto também ocorre para os coeficientes de um bloco de transformada de bloco. De fato, uma árvore *wavelet* em uma decomposição em N níveis é análoga a um bloco de coeficientes de uma transformada de blocos com 2^N bandas.

Essa analogia nos dá indicativos da utilização de métodos que tenham obtido sucesso na codificação baseada em *wavelet* sobre coeficientes de transformada de bloco. Encontramos essa abordagem nos trabalhos de Xiong [10] e de Queiroz e Tran [3]. Nesses trabalhos, para que os algoritmos de codificação *wavelet* baseados em *zerotree* possam ser utilizados na codificação de coeficientes de transformadas de bloco, a estrutura de dados (árvore espacial) da *zerotree* e a definição dos conjuntos de descendentes foram modificados ligeiramente. Denotaremos esta estrutura de árvore como árvore modificada. Denotaremos a árvore espacial definida no SPIHT como árvore original. A estrutura da árvore modificada pode ser vista na Fig. 3. A árvore original foi mostrada na Seção III. Adiante veremos testes realizados com os dois tipos de árvore.

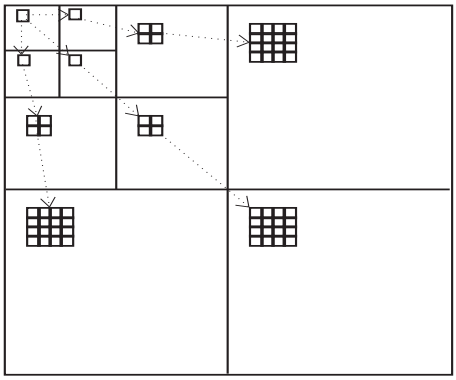

Fig. 3. Árvore espacial sugerida por Xiong.

A técnica de SPQ foi avaliada em uma arcabouço de codificação de vídeo compatível com o padrão MPEG-4 (perfil de codificação Main Profile). Os quadros do tipo *intra* são codificados como imagens, sem referência a outros quadros. Utilizando o mesmo arcabouço, podemos obter resultados para imagem (*intra*) e vídeo (*inter*)[1]. O SPQ opera sobre

[1]Desta forma, com poucas modificações, a técnica pode ser aplicada à codificação de imagens, como, por exemplo, no padrão JPEG.

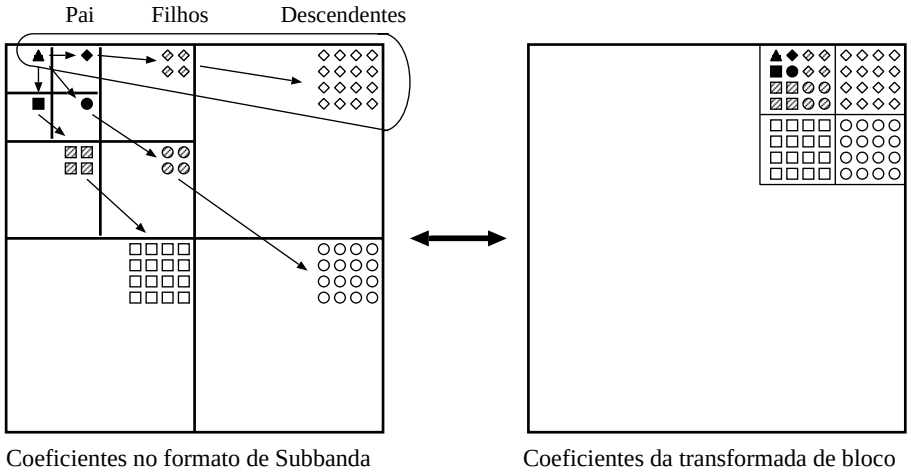

Fig. 2. Analogia entre transformadas de bloco e decomposição *wavelet*.

toda a imagem, no plano de luminância Y e nos planos de crominância U e V, tanto nos quadros intra quanto inter.

Adiante veremos os detalhes da substituição do codificador entrópico para os coeficientes DCT do MPEG-4 pelo SPQ.

Na implementação prática, o SPQ se distingue do SPIHT basicamente na condição de parada do algoritmo. O SPIHT normalmente opera até que um determinado número de bits seja atingido ou uma certa distorção seja atingida. No caso do SPQ, o algoritmo atravessa todos os planos de bits significativos que são encontrados.

O SPQ utiliza como etapa final um codificador aritmético otimizado para dois símbolos, com possibilidade de seleção adaptativa de contextos [20], [21]. Esta possibilidade de adaptação de contextos, se bem explorada, pode proporcionar uma diminuição no número de bits utilizados. No SPQ, a seleção dos contextos foi realizada de forma a aproveitar a correlação existente entre o número de pixels significativos em um conjunto de descendentes imediatos e a significância das árvores que são descendentes destes pixels.

Foi utilizado um conjunto de 14 contextos, sejam eles: 1 contexto para codificação de sinal; 1 contexto para codificação dos bits de refinamento; 4 contextos para codificação adaptativa das significâncias dos conjuntos tipo B; 4 contextos para codificação das significâncias dos conjuntos tipo A dependente da codificação dos conjuntos tipo B; e finalmente 4 contextos para a codificação dos pixels descendentes imediatos. As definições destes conjuntos e das listas foram vistas na Seção III.

O contexto para a codificação da significância dos pixels é selecionado de acordo com o número de pixels que já foram significantes no mesmo bloco 2×2. Esta escolha é armazenada na LIS junto com os conjuntos do tipo B. Durante a codificação da significância dos conjuntos do tipo A e do tipo B, este valor é recuperado e utilizado na seleção de novos contextos. Desta forma, há uma propagação ao longo do particionamento dos conjuntos do número de pixels significativos no começo da árvore. Só há um contexto para o sinal porque o sinal é considerado descorrelacionado. Pelo mesmo motivo, temos apenas um contexto para os bits de refinamento.

A codificação MPEG-4 prevê a predição de coeficientes DCT a partir de seus vizinhos causais. Para blocos do tipo *intra*, há predição dos coeficientes DC e AC. Para blocos do tipo inter, há apenas predição DC. Na predição DC, subtrai-se o coeficiente DC do bloco atual do coeficiente DC do bloco imediatamente anterior ou imediatamente acima (na linha de blocos anterior). A seleção de que bloco é usado na predição é baseada na direção de maior variação destes coeficientes. Os coeficientes AC da primeira linha e da primeira coluna são preditos a partir do mesmo bloco usado para o coeficiente DC. Esta predição melhora a codificação ao reduzir a amplitude dos coeficientes. No entanto, faz-se necessário avaliar melhor se esta predição produz bons resultados para a codificação com plano de bits e *zerotree*. Foram realizados testes com e sem predição.

Como proposto por Queiroz [3], a banda formada pelos coeficientes do nível DC ainda pode se tornar mais descorrelacionada, aplicando alguns níveis de decomposição *wavelet*, a depender das dimensões do quadro. Como o objetivo desta codificação é a reconstrução perfeita, haja visto que os coeficientes já foram quantizados, faz-se necessário o uso de *wavelets* que mapeiem inteiros em inteiros. Por simplicidade, optou-se por utilizar uma variação da *wavelet* de Haar, definida da seguinte forma. Seja $x = (x_0, x_1, \ldots, x_{N-1})$ o vetor de N elementos, N par, a ser transformado e $y = (y_0, y_1, \ldots, x_{N-1})$ o vetor de saída. O resultado da filtragem passa-baixa a_i e passa alta d_i é dado por

$$\begin{aligned} a_i &= \lfloor \tfrac{x_i + x_{i+1}}{2} \rfloor \\ d_i &= x_i - x_{i+1}, \end{aligned} \tag{1}$$

para i par, e o vetor de saída é organizado como

$$\begin{aligned} y_{i/2} &= a_i \\ y_{\frac{i+N}{2}} &= d_i. \end{aligned} \tag{2}$$

A transformada inversa é calculada como

$$\begin{aligned} x_{2i} &= a_i + \lfloor \tfrac{d_i + 1}{2} \rfloor \\ x_{2i+1} &= x_i - d_i, \end{aligned} \tag{3}$$

sendo este procedimento realizado para as linhas e colunas da imagem a ser transformada. Este processo pode ser repetido para formar mais níveis de decomposição *wavelet*.

VI. Resultados

Nesta seção apresentamos os resultados da codificação de vídeo utilizando o codificador SPQ. Foram realizados testes com os dois tipos de árvores (modificada e original), com e sem predição MPEG-4 e com a aplicação opcional de transformada *wavelet* na banda DC. Convém informar que os resultados são apresentados como bytes utilizados para

a codificação, sem nenhuma referência à qualidade obtida, porque essa não é modificada em comparação com o codificador original do MPEG-4. Deve ser observado também que é contabilizada, na quantidade de bytes mostrada para o codificador RLVLC do MPEG-4, a representação dos blocos não codificados, representados de forma eficiente com apenas um bit. No caso do SPQ, a área correspondente ao bloco 8×8 classificada como "não codificada" é zerada e codificada conjuntamente com os demais blocos, sem nenhuma referência explícita.

Na Tabela I, observamos os resultados para a seqüência "Miss America" (Fig. 4), para o coeficiente de quantização (parâmetro de quantização, fator de qualidade, parâmetro de qualidade) $Q = 2$, que corresponde à alta qualidade (alta taxa de codificação). Na segunda coluna da tabela, temos o número de bytes utilizados pela codificação MPEG-4 convencional. Estes valores representam apenas os bytes utilizados na codificação da textura, excluindo-se os bits de controle e de representação dos vetores de movimento. O quadro de número 1 foi codificado como *intra*. Os demais foram codificados como *inter*. Observa-se que a maior redução no número de bytes neste codificador ocorre para os quadros intra, com a aplicação da decomposição normal (de SPIHT), o uso de predição e de nova decomposição *wavelet* dos coeficientes DC. O ganho não é tão significativo para os quadros *inter*. Um dos motivos para isto é o grande número de macroblocos em que a textura não é codificada ou utiliza-se apenas 1 ou 2 coeficientes. Isto ocorre quando a compensação atinge uma boa predição. Nestes quadros gastam-se menos bytes quando não se utiliza a transformada *wavelet*. Também neste caso a decomposição modificada não fornece os melhores resultados. Na Tabela II vemos a codificação da mesma seqüência para um coeficiente de quantização $Q = 15$. Isto corresponde a uma qualidade baixa e a uma pequena taxa de codificação. Para os quadros *inter*, observa-se que as duas decomposições produzem valores similares. Observa-se também resultados ligeiramente melhores sem o uso de *wavelet*.

Fig. 4. Primeiro quadro da seqüência *Miss America*. Cena típica de vídeo-conferência, mostrando cabeça e ombros, com muito pouca movimentação. Formato CIF.

Na Fig. 5 observamos o número de bytes usados na codificação do primeiro quadro da seqüência "Miss America", com codificação *intra*, em função do coeficiente Q. A maior economia em termos do número de bytes (em relação ao MPEG-4 convencional, RLVLC), cerca de 20%, ocorre para $Q = 2$. A economia diminui com o aumento de Q (diminuição da qualidade). Foi usada neste teste a combinação que deu o melhor resultado para baixos valores de Q, ou seja, decomposição original, predição MPEG-4 e decomposição *wavelet* dos coeficientes DC.

Para a seqüência "Flower Garden" (Fig. 6), os codificadores

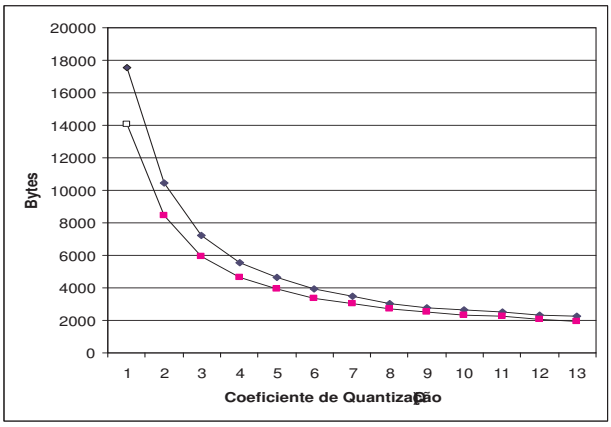

Fig. 5. Resultado da codificação intra para a seqüência "Miss America", variando-se o fator de qualidade Q. O codificador SPQ foi configurado com a decomposição original, predição dos coeficientes e decomposição *wavelet* dos coeficientes DC. A curva superior corresponde ao MPEG-4 convencional e a inferior ao SPQ

de vídeo, em geral, não conseguem produzir uma baixa taxa de bits, em virtude de ela possuir um movimento global e texturas complexos. A codificação desta seqüência é beneficiada pelo uso do SPQ. Obtém-se uma economia, em relação ao MPEG-4 convencional, de cerca de 37% do número de bytes, para o quadro *intra* com $Q = 2$. A economia diminui com o aumento do Q. Neste caso, a maior economia ocorre quando se utiliza a decomposição modificada, sem predição MPEG-4 nem decomposição *wavelet* dos coeficientes DC.

Fig. 6. Primeiro quadro da seqüência *Flower Garden*. Cena com movimentação rápida de toda a imagem devido à movimentação da câmera com relação à imagem. Cena com muitas texturas. Formato SIF.

Para seqüências com muita textura, onde particularmente a codificação padrão do MPEG-4 não realiza um bom trabalho, obtém-se com o SPQ uma grande redução no número de bytes utilizados. Os melhores resultados sempre ocorrem para altas taxas.

Para baixas taxas de codificação, os coeficientes DCT possuem valores pequenos e são dispostos de forma esparsa. Este efeito se agrava nos quadros *inter*, em que a amplitude dos coeficientes é menor e a distribuição de freqüência é diferente, não havendo um agrupamento na vizinhança do nível DC. A codificação destes coeficientes torna-se basicamente uma codificação de posição dos coeficientes não zero. O melhor aproveitamento desta esparsidade pode levar a técnicas mais eficientes de codificação dos coeficientes DCT em vídeo.

VII. Conclusão

Este trabalho apresentou o SPQ (SPIHT pós quantização), um método para codificação sem perdas de coeficientes quantizados, oriundos de transformadas de bloco. Os resultados

TABELA I
CODIFICAÇÃO DA SEQÜÊNCIA "MISS AMERICA" COM $Q = 2$. OS NÚMEROS SÃO EM BYTES.

		Decomposição Normal			Decomposição Modificada	
Quadro	MPEG-4		sem *wavelet*	com *wavelet*	sem *wavelet*	com *wavelet*
1	17580	sem pred	14789	14689	15143	15033
		com pred	14108	14052	14826	14741
2	9512		8811	8970	9025	9199
3	9491		8772	8866	9023	9133
4	9963		9133	9235	9311	9427
5	10205		9376	9446	9550	9646
6	10016		9107	9233	9492	9628
7	9720		8918	8982	9298	9385
8	9763		8932	9095	9290	9449
9	9415		8583	8666	8907	9006
10	9762		8855	8908	9236	9337

TABELA II
CODIFICAÇÃO DA SEQÜÊNCIA "MISS AMERICA" COM $Q = 15$. OS NÚMEROS SÃO EM BYTES.

		Decomposição Normal			Decomposição Modificada	
Quadro	MPEG-4		sem *wavelet*	com *wavelet*	sem *wavelet*	com *wavelet*
1	2184	sem pred	2363	2288	2065	2051
		com pred	1818	1887	1830	1888
2	129		110	116	110	119
3	160		139	159	144	163
4	196		196	219	182	206
5	221		212	227	213	228
6	210		215	239	221	245
7	134		115	134	122	143
8	141		124	141	131	152
9	185		194	222	199	225
10	96		71	82	75	84

de simulação mostraram que o SPQ constitui uma alternativa adequada para codificação de vídeo. Para a seqüência "Miss America", por exemplo, para codificação intra com parâmetro de quantização $Q = 2$, o SPQ levou a uma economia de cerca de 20% do número de bytes em relação ao MPEG-4 convencional. Para a seqüência "Flower Garden", obteve-se uma economia correspondente de cerca de 37% para codificação intra. Como trabalho futuro, o método SPQ será avaliado no padrão H.264.

REFERÊNCIAS

[1] G. K. Wallace. "The JPEG Still Picture Compression Standard". *Communications of the ACM*, vol. 34, pp. 30–44, April 1991.

[2] R. Koenen. "Overview of the MPEG-4 Standard". Technical Report JTC1/SC29/WG11 N4030, ISO/IEC, March 2001.

[3] R. L. de Queiroz and T. D. Tran. *The Handbook on Transforms and Data Compression*, chapter Lapped transforms for image compression, pp. 197–265. CRC Press, October 2000.

[4] W. B. Pennebaker and J. L. Mitchell. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, NY, USA, 1993.

[5] ITU-T. "Video Codec for Audiovisual Services at p×64 kbit/s, (International Telecommunication Union - Telecommunication Standardisation Sector) Recommendation H.261", 1993.

[6] I. I. standard 13818-2. "Generic Coding of Moving Pictures and Associated audio information: Part2 - Video", 1995.

[7] D. Y. Pan. "Digital Audio Compression". *Digital Technical Journal of Digital Equipment Corporation*, vol. 5, no. 2, pp. 28–33, 1993.

[8] C. Christopoulos, A. N. Skodras and T. Ebrahimi. "The JPEG2000 Still Image Coding System: an Overwiew". *IEEE Transactions on Consumer Electronics*, vol. 46, no. 4, pp. 1103–1127, November 2000.

[9] ITU. "Video coding for low bit rate communication". ITU-T Recommendation H.263, March 1996.

[10] Z. Xiong, O. Guleryuz and M. T. Orchard. "A DCT-based Embedded Image Coder". *IEEE Signal Processing Letters*, vol. 3, no. 11, pp. 289–290, 1996.

[11] 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.

[12] J. M. Shapiro. "Embedded Image Coding Using Zerotrees of Wavelet Coefficients". *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3445–3462, December 1993.

[13] G. M. Davis and A. Nosratinia. "Wavelet-based Image Coding: An Overview". *Applied and Computational Control, Signals, and Circuits*, vol. 1, no. 1, Spring 1998.

[14] A. S. Lewis and G. Knowles. "Image Compression Using the 2-D Wavelet Transform". *IEEE Transactions on Image Processing*, vol. 1, no. 2, pp. 244–250, April 1996.

[15] R. Koenen. "Overview of the MPEG-4 Standard". ISO/IEC JTC1/SC29/WG11, July 1996.

[16] A. Puri, R. L. Schmidt and B. G. Haskell. "Performance Evaluation of the MPEG-4 Visual Coding Standard". In *Proceedings of the SPIE*, volume 3309, pp. 417–433, San Jose, CA, USA, January 1998.

[17] V. Ratnakar and M. Livny. "RD-OPT: An Efficient Algorithm For Optimizing DCT Quantization Tables". In *Proceedings of IEEE Data Compression Conference (DCC)*, pp. 332–341, 1995.

[18] V. Ratnakar. "Quality-Controlled Lossy Image Compression". Ph.D. thesis, University of Wisconsin – Madison, 1997.

[19] V. Ratnakar and M. Livny. "Extending RD-OPT with Global Thresholding for JPEG Optimization". In *Proceedings of IEEE Data Compression Conference (DCC)*, pp. 379–386, 1996.

[20] T. C. Bell, J. G. Cleary and I. H. Witten. *Text Compression*. Prentice Hall, Englewood Cliffs, NJ, USA, 1990.

[21] I. Witten, R. Neal and J. Cleary. "Arithmetic Coding for Data Compression". *Communications of the ACM*, vol. 30, pp. 520–540, 1987.