Compressão de mapas de profundidade usando predição e funções polinomiais

Renam C. da Silva, Luís F. R. Lucas, Eduardo A. B. da Silva, Carla L. Pagliari,
Nuno M. M. Rodrigues, Sérgio M. M. Faria

***Resumo*— Em técnicas que se baseiam nas informações de textura e de profundidade para gerar vistas virtuais, os mapas de profundidade precisam ser representados com um número reduzido de *bits* sem comprometer a qualidade visual das vistas virtuais. A codificação de mapas usando segmentação flexível e aproximação dos resíduos de predição por funções lineares (LFP) tem alcançado resultados promissores. As vistas virtuais, geradas a partir de mapas codificados com o LFP, têm melhor qualidade visual quando comparadas às vistas geradas com mapas codificados com diversos algoritmos testados. Neste artigo é proposto aproximar o resíduo de predição usando, além de funções lineares, funções constantes e quadráticas. E ainda, utilizar quantizadores treinados para cada taxa pretendida. Os resultados obtidos, empregando o esquema proposto, tem desempenho superior em termos de taxa de compressão dos mapas e de qualidade visual das vistas virtuais.**

***Palavras-Chave*— Mapa de profundidade, codificação, predição, funções polinomiais.**

***Abstract*— In techniques that are based on the texture and depth information to generate virtual view, the depth maps have to be represented with a reduced number of bits without compromising the visual quality of virtual view. The coding of maps using flexible segmentation and approximation of the prediction residuals by linear functions (LFP) has achieved promising results. The virtual views generated from maps coded with LFP have better visual quality when compared with others tested algorithms. In this paper we propose to approximate the residue using, in addition to linear functions, quadratic and constant functions. And using trained quantizers for each desired rate. The results obtained by employing the proposed scheme show a superior performance in terms of compression ratio and visual quality of virtual views.**

***Keywords*— Depth map, coding, prediction, polynomial functions.**

I. INTRODUÇÃO

Técnicas para a exibição de imagem e vídeo em três dimensões têm sido uma ativa área de pesquisa devido ao potencial de aplicação em produtos e serviços. Os conteúdos exibidos em 3D permitem uma percepção mais realista e imersiva da cena. Para a exibição em 3D, é necessário que a cena seja capturada em pontos distintos, gerando uma grande quantidade de dados, o que potencialmente demanda bastante recursos de armazenamento e/ou transmissão. Recentemente, o interesse em sistemas com múltiplas vistas tem crescido, pois esses sistemas permitem uma experiência mais rica e interativa. As técnicas que usam informações de textura e de profundidade para gerar vistas virtuais (DIBR) [1] têm sido consideradas para aplicações em 3D, pois permitem a cobertura da cena com um arranjo esparso de câmeras. Esses métodos sintetizam vistas virtuais a partir de imagens de textura e de profundidade, dessa forma requerem menos recursos de armazenamento e transmissão.

Independente do método utilizado para gerar vistas virtuais, é necessário que a informação de profundidade assim como as imagens de textura da cena sejam codificadas de forma eficiente. Para a codificação de imagens de textura existem padrões bem estabelecidos - como JPEG, H264/AVC e mais recentemente o HEVC - que exibem resultados estado da arte. A aplicação direta desses algoritmos para compressão de mapas não é adequada, devido a priorização da qualidade visual adotada nesses padrões, enquanto que em mapas é mais importante preservar a informação de profundidade da cena. Mapas de profundidade frequentemente aprensentam grandes regiões suaves com abruptas transições nas fronteiras dessas regiões. Algoritmos tradicionais para compressão de textura, quando aplicados na codificação de mapas, exibem resultados com artefatos nas transições das regiões suaves penalizando severamente as vistas virtuais sintetizadas com algoritmos DIBR.

Algoritmos, que consideram as características particulares de mapas, têm sido propostos. Em [2] foi proposto adaptar o padrão JPEG2000 para incluir uma priorizaração de regiões de interesse presentes em mapas. Nesse esquema, a codificação apresenta problemas quando o mapa contém muitos objetos. A técnica de amostragem não uniforme através de uma segmentação em malha triangular foi investigada na codificação de mapas em [3]. Nas descontinuidades dos mapas essa abordagem necessita de muitos segmentos para a codificação. O algoritmo de recorrência de padrões multi-escalas (MMP) também foi usado na codificação de mapas em [4]. Apesar dos resultados promissores, a complexidade computacional do MMP é elevada. A codificação de mapas usando funções suaves foi investigada em [5] e [6] . O padrão de compressão HEVC, em fase final de definição, está sendo estendido para codificação de mapas [7]. Em [6], a aproximação dos resíduos de predição com funções lineares aliada a uma segmentação flexível (LFP) foi proposta e comparada com diversos algoritmos. Dentre os quais, o padrão H264/AVC (*stereo* e *intra*), MMP, *platelet* [5] e o HEVC (*intra*). Foram geradas vistas virtuais usando as imagens de

Renam C. da Silva[1], Luís F. R. Lucas[1,3], Eduardo A. B. da Silva[1], Carla L. Pagliari[2], Nuno M. M. Rodrigues[3,4], Sérgio M. M Faria[3,4]
[1]PEE/COPPE/DEL/POLI, Universidade Federal do Rio de Janeiro, Brasil. [2]DEE, Instituto Militar de Engenharia, Brasil. [3]Instituto de Telecomunicações, Portugal. [4]ESTG, Instituto Politécnico de Leiria, Portugal. E-mails: renam.silva@smt.ufrj.br, luis.lucas@smt.ufrj.br, eduardo@smt.ufrj.br, carla@ime.eb.br, nuno.rodrigues@co.it.pt, sergio.faria@co.it.pt. Este trabalho foi parcialmente financiado pelo CNPq e pela CAPES.

textura originais e os mapas de profundidade codificados com os algoritmos citados. Em seguida, foi computada a fidelidade das vistas virtuais em relação as vistas reais capturadas e também com vistas geradas usando os mapas originais. O resultado alcançado com o LFP foi superior aos algoritmos citados.

Neste trabalho, propomos aproximar o bloco de resíduo usando, além de funções lineares, funções constantes e quadráticas. Investigar o efeito da quantização dos coeficientes das funções de aproximação empregadas no LFP na qualidade visual das vistas virtuais. E ainda, utilizar quantizadores treinados para cada taxa pretendida.

O artigo está organizado da seguinte maneira. Na seção II descrevemos o algoritmo LFP. Na seção III são descritos os esquemas propostos para melhorar o desempenho do algoritmo LFP, tanto na compressão dos mapas quanto na melhoria da qualidade visual das vistas virtuais geradas. Em IV apresentamos os resultados alcançados e em V as conclusões.

II. O ALGORITMO LFP

O LFP [6] foi proposto e aplicado na codificação de mapas de profundidade partindo da premissa que a imagem de um mapa de profundidades é um conjunto de regiões suaves com bruscas transições nas fronteiras dessas regiões. Partindo dessa observação foi proposto usar técnicas de predição, que é muito favorável em regiões suaves. Após a predição, o resíduo é segmentado de uma forma flexível e aproximado utilizando funções lineares. A estratégia de segmentação flexível permite que regiões suaves sejam aproximadas por blocos maiores, contribuindo para uma codificação eficiente. As transições que, em geral, indicam mudanças na profundidade (entre objetos da cena), são codificadas usando técnicas de predição e aproximações com blocos menores preservando a informação de mudança da profundidade.

As decisões de como segmentar um bloco bem como o modo de predição empregado são sinalizados ao decodificador através de *flags*. De cada bloco B^l, da escala l, é subtraída a predição $\tilde{B}^l$ gerando um resíduo S^l. O resíduo é aproximado por uma função linear ou por uma palavra do dicionário D^l. Os *flags* indicando como a segmentação e a predição foram realizadas são codificados com o codificador aritmético, assim como a escolha se o bloco é melhor aproximado pela função ou por um vetor do dicionário. Em seguida, os coeficientes da função de aproximação, caso seja escolhida a aproximação por uma função, ou o índice do vetor do dicionário da escala correspondente é codificado. O decodificador reconstrói a imagem a partir dos *flags* de segmentação e predição bem como o método utilizado para aproximação.

A. Segmentação e predição

Inicialmente, a imagem do mapa de profundidades é segmentada em blocos de 32×32 *pixels*. Os blocos de 32×32 podem ainda ser segmentados em blocos de 32×16, 16×32 e 16×16 *pixels*. A partir de 16×16 a segmentação pode ser realizada livremente, permitindo 25 dimensões diferentes. Consideramos que os blocos de 32×32 pertencem à escala 27 enquanto que os blocos de 1×1 pertencem à escala 0.

Os modos de predição utilizados no LFP são baseados nos modos do H264/AVC. São empregados 9 modos de predição, iguais aos modos utilizados em blocos de 4×4 *pixels* das amostras *luma* do H264/AVC. O modo de predição utilizado será aquele que fornecer o resíduo com menor norma-$L1$. No LFP a predição pode ser realizada em qualquer bloco de tamanho igual ou maior que 4×4 *pixels*.

O codificador informa ao decodificador a escolha de segmentação e predição de cada bloco através de *flags*. São empregados cinco *flags*: dois para informar se o bloco deve ser segmentado na vertical ou horizontal, dois para indicar que a predição não muda, mas há uma segmentação vertical ou horizontal, e um para indicar que a predição e o bloco não devem ser segmentados.

O decodificador reconstrói a imagem, bloco a bloco, a partir das decisões tomadas pelo codificador de como realizar a segmentação do bloco e da predição bem como o modo de predição e a aproximação empregada.

B. Aproximação do resíduo

Cada bloco de resíduo S^l é aproximado por uma função ou por um vetor do dicionário da escala l. O dicionário de cada escala é composto de blocos aproximados anteriormente pela função, cada escala tendo um próprio dicionário. Os dicionários de todas as escalas são inicializados com um vetor nulo. A função linear utilizada na aproximação é dada por:

$$f(\tilde{x}, \tilde{y}) = \alpha_0 \tilde{x} + \alpha_1 \tilde{y} + \alpha_2 \tag{1}$$

Onde, $\tilde{x} = (x - 2^{m+1})$, $\tilde{y} = (y - 2^{n-1} - 1)$, m é o comprimento horizontal do bloco e n é o comprimento vertical do bloco. Na equação 1, $\tilde{x}$ e $\tilde{y}$ são as versões deslocadas das coordenadas dos *pixels* no bloco. Dessa forma o termo α_2 da equação 1 é igual a média dos valores do bloco. Numa boa predição, o resíduo tem distribuição concentrada e centrada em zero. Portanto, o coeficiente α_2 pode ser eficientemente codificado para blocos de média zero. Os coeficientes são determinados minimizando a norma-$L2$ do erro de aproximação, dado por:

$$E = \| e \|^2 = \sum_{i=1}^{mn} {e_i}^2$$
$$e_i = (s_i - (\alpha_0 \tilde{x_i} + \alpha_1 \tilde{y_i} + \alpha_2)) \tag{2}$$

Onde s_i é uma amostra do resíduo. Os coeficientes são quantizados, não uniformemente, usando os passos de quantização (q) a seguir.

Para α_2

$$q = \begin{cases} 1, & \text{se} \quad -10 < \alpha < 10, \\ 4, & \text{se} \quad -22 < \alpha \leq -10 \quad \text{ou} \quad 10 \leq \alpha < 22, \\ 8, & \text{se} \quad -86 < \alpha \leq -22 \quad \text{ou} \quad 22 \leq \alpha < 86, \\ 13, & \text{se} \quad \alpha \leq -86 \quad \text{ou} \quad \alpha \geq 86, \end{cases}$$

Para α_0 e α_1

$$q = \begin{cases} 1, & \text{se} \quad -10 < \alpha < 10, \\ 4, & \text{se} \quad -22 < \alpha \leq -10 \quad \text{ou} \quad 10 \leq \alpha < 22, \\ 8, & \text{se} \quad -62 < \alpha \leq -22 \quad \text{ou} \quad 22 \leq \alpha < 62, \\ 13, & \text{se} \quad \alpha \leq -62 \quad \text{ou} \quad \alpha \geq 62, \end{cases}$$

Os coeficientes, após quantizados, são codificados com o codificador aritmético, a escala l é usada como contexto. O bloco reconstruído a partir dos coeficientes da função é adicionado no dicionário da escala l, podendo ser reutilizado.

C. Otimização taxa-distorção

A árvore de segmentação $\mathcal{T}$ de cada bloco de 32×32 *pixels* é otimizada de maneira que minimize a função custo J na equação 3:

$$J(\mathcal{T}) = D(\mathcal{T}) + \lambda R(\mathcal{T}) \quad (3)$$

Onde $D(\mathcal{T})$ é a distorção, $R(\mathcal{T})$ é a taxa. O parâmetro λ pondera o compromisso entre a taxa desejada e a distorção incorrida. O erro médio absoluto é a métrica de distorção utilizada no LFP, pois foi verificado experimentalmente que é superior à métrica do erro médio quadrático [6]. Podemos notar que o custo de codificar cada bloco de 32×32 depende das escolhas de segmentação, da predição utilizada e da aproximação escolhida em cada escala.

A árvore de segmentação $\mathcal{T}$ ótima, para cada bloco de 32×32, é escolhida podando os ramos a partir da árvore inteiramente expandida. É importante notar que este processo recursivo testa todas as combinações de segmentação, predição e aproximação do resíduo.

Em cada escala l o bloco é segmentado tanto na vertical quanto na horizontal. O custo de codificar o bloco inteiro (sem segmentação) é comparado com os custos de codificar os blocos resultantes da segmentação vertical e horizontal. A estratégia que resultar no menor custo é escolhida. Essa técnica é aplicada recursivamente de baixo para cima, ou seja, a partir de blocos da escala 0 até os blocos da escala 27. Em cada etapa, para cada bloco, é utilizado o modo de predição que fornecer o resíduo de menor norma-$L1$ e aproximação com menor custo, em termos de *bits* e distorção. As equações 4 e 5 medem, respectivamente, os custos de aproximação da função e do vetor do dicionário.

$$J_{fun}(S^l) = D_{fun}(S^l) + \lambda[R(flg_{fun}) + \sum_{j=0}^{2} R(\alpha_j)] \quad (4)$$

$$J_{dic}(S^l) = D_{dic}(S^l) + \lambda[R(flg_{dic}) + R(ind)] \quad (5)$$

Onde flg_{fun} é o *flag* que indica o uso da função como aproximação, α_j o coeficiente da função, flg_{dic} é o *flag* que indica o uso de um vetor do dicionário para a aproximação e ind é o índice do vetor do dicionário. O resultado da otimização fornece a melhor escolha de segmentação do bloco e da predição, do modo de predição, e aproximação.

III. Esquema proposto

A. Aproximação do bloco de resíduo

Além de aproximar os blocos com funções lineares (equação 1), sugerimos utilizar também funções constantes e quadráticas, definidas respectivamente como:

$$g(\tilde{x}, \tilde{y}) = \alpha_0; \quad (6)$$

$$h(\tilde{x}, \tilde{y}) = \alpha_0\tilde{x}^2 + \alpha_1\tilde{y}^2 + \alpha_2\tilde{x} + \alpha_3\tilde{y} + \alpha_4\tilde{x}\tilde{y} + \alpha_5; \quad (7)$$

Onde $\tilde{x}$ e $\tilde{y}$ foram definidos anteiormente (seção II-B). A escolha da melhor função de aproximação é indicada ao decodificador através de um *flag*. No LFP são utilizados dois *flags*, um para informar a aproximação utilizando um vetor do dicionário e outro para a aproximação usando uma função (linear, pois originalmente apenas a função linear é utilizada). Além desses dois *flags*, utilizamos outro *flag* para informar a função de aproximação: constante, linear ou quadrática. Em seguida, os coeficientes da função são codificados. Para a função constante (definida em 6) o coeficiente α_0 é a média do bloco de resíduo. Para a função quadrática (definida em 7) minimizamos a norma-$L2$ do erro de aproximação.

O algoritmo calcula o custo de aproximação usando um vetor do dicionário e o custo de aproximação usando cada uma das três funções, a aproximação com menor custo é utilizada. O custo de aproximação utilizando um vetor do dicionário é calculado com a equação 5. Para o cálculo do custo de aproximação utilizando as funções, usamos a equação 4, contudo, além do *flag* que indica a aproximação com a função, é levado em conta o *flag* que indica qual a função. Os coeficientes da função de aproximação são incluídos no cálculo do custo respectivo.

Para isolar os artefatos imputados apenas pelo processo de codificação dos mapas, a PNSR foi computada a partir da vista virtual obtida com o algoritmo DIBR usando as imagens de textura e os mapas codificados com o LFP em relação a vista virtual sintetizada usando as imagens de textura e os mapas originais. As figuras 1, 2 e 3 exibem os resultados da vista virtual do quadro 0 da câmera 1 da sequência *Ballet*.

Na figura 1, diversos esquemas de aproximação do resíduo foram usados, permitindo avaliar o efeito da função de aproximação na qualidade visual da vista virtual gerada. O esquema de quantização original do LFP foi utilizado nesse exemplo. Neste trabalho, o algoritmo de síntese usado foi o VSRS-3.5 [10]. Nos gráficos desta seção, *c*, *l* e *q* correspondem às funções constante, linear e quadrática, respectivamente. A curva *lfp* foi obtida com o algoritmo LFP original.

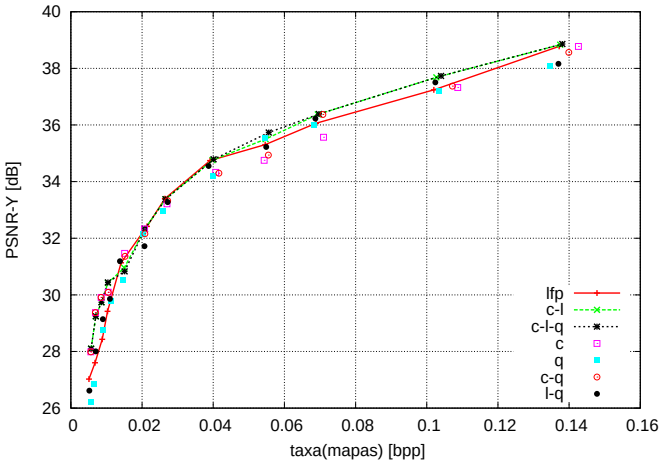

Fig. 1: Funções de aproximação (*Ballet* câmera 1)

B. *Quantização*

O esquema de quantização empregado no LFP foi definido de maneira razoável, mas intuitiva. Como pode ser verificado na seção II-B, a quantização utilizada no LFP prioriza as amostras do resíduo com valor próximo de zero com uma quantização mais fina. À medida que o valor das amostras do resíduo se afasta de zero, a quantização é feita de forma mais grosseira.

Para avaliar o efeito da quantização, foram testados quantizadores com quantidades diferentes de níveis de reconstrução. A partir das amostras dos valores dos coeficientes da função (equação 1), estimamos os parâmetros da gaussiana generalizada que modela a distribuição de cada coeficiente [8]. As amostras dos coeficientes foram obtidas utilizando o algoritmo LFP original para um conjunto de imagens de textura (imagens de treinamento). Em seguida, foram gerados dados segundo a gaussiana generalizada que modela a distribuição de cada coeficiente. E então, foram obtidos os quantizadores ótimos, dado a quantidade de níveis de reconstrução, usando o algoritmo *Lloyd-Max* [9]. O mesmo procedimento foi conduzido usando o LFP com função de aproximação apenas constante e apenas quadrática.

Na figura 2, podemos verificar o efeito da quantização dos coeficientes da função de aproximação (apenas função linear) do LFP na qualidade visual das vistas virtuais sintetizadas. Na curva *lfp*, os mapas foram codificados com o LFP original. Nas curvas 5-81 foram utilizados os mapas codificados com o LFP, empregando o esquema de quantização proposto. Os números das curvas correspondem a quantidade de níveis de reconstrução dos quantizadores. A curva *ch* (*convex hull*) é a casca convexa das curvas 5-81.

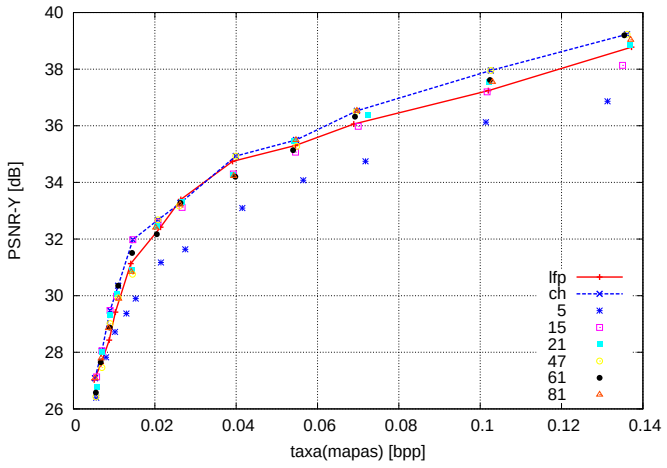

Fig. 2: Efeito da quantização (*Ballet* câmera 1)

A figura 3 exibe os resultados obtidos quando combinamos o esquema de aproximação do resíduo de predição e o esquema de quantização. Apenas os resultados dos melhores esquemas foram exibidos. Baseado nos resultados obtidos, definimos o esquema de quantização, cujos resultados se aproxima da casca convexa dos resultados obtidos com os quantizadores testados, como:

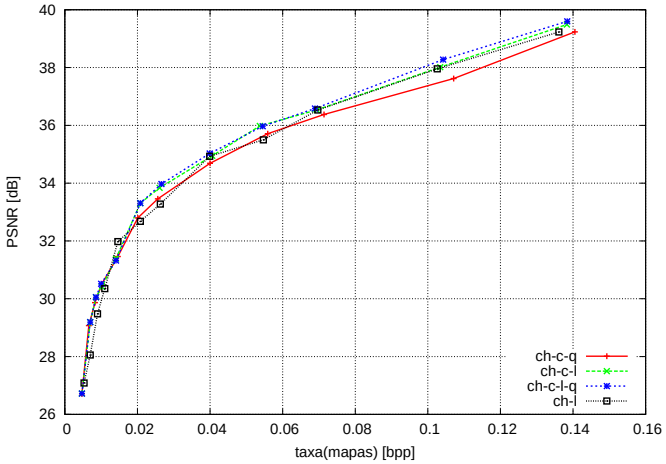

Fig. 3: Combinando os esquemas propostos (*Ballet* câmera 1)

$$Q = \begin{cases} Q5, & \text{se} \quad \lambda > 1000, \\ Q15, & \text{se} \quad 1000 \geq \lambda > 299, \\ Q21, & \text{se} \quad 299 \geq \lambda > 75 \quad , \\ Q47, & \text{se} \quad 75 \geq \lambda > 49 \quad , \\ Q61, & \text{se} \quad 49 \geq \lambda > 14 \quad , \\ Q81, & \text{se} \quad \lambda \leq 14 \quad , \end{cases}$$

Onde, $Q5$ corresponde ao quantizador com 5 níveis de reconstrução, e assim sucessivamente. O esquema de aproximação do resíduo que emprega as funções constante, linear e quadrática foi adotado, pois verificamos que exibe resultado igual ou superior aos demais testados.

IV. Resultados Experimentais

As vistas virtuais obtidas com o VSRS-3.5[10] - usando os mapas codificados com o esquema proposto - foram comparadas com as vistas virtuais geradas utilizando os mapas codificados com LFP original. Em [6], foi mostrado que os resultados alcançados com o LFP são superiores aos resultados obtidos com diversos algoritmos testados, por esse motivo comparamos os resultados apenas com o LFP. Reiteramos que, nos resultados exibidos, a PSNR foi calculada em relação a vista virtual sintetizada usando os mapas originais. Dessa forma, avaliamos apenas eventuais artefatos inseridos pela codificação dos mapas, já que as imagens de textura originais (sem codificação) foram utilizadas no algoritmo de síntese.

As sequências[1] testadas foram *Ballet* (síntese da câmera 1), *Breakdancers* (síntese da câmera 1), *Book-arrival* (síntese da câmera 9) e *Champagne* (síntese da câmera 40). Apenas o quadro 0 de cada sequência foi sintetizado.

O ganho em PSNR da vista virtual sintetizada usando, além das imagens de textura, os mapas codificados com LFP e empregando os esquemas propostos depende da sequência. Para a *Ballet*, por exemplo, o ganho máximo é próximo de 1 dB. Nas figuras 4, 5, 6, 7, as curvas *ch* são as cascas convexas utilizando os quantizadores testados.

[1] *Ballet* e *Breakdancers* estão disponíveis em http://research.microsoft.com/en-us/um/people/sbkang/3dvideodownload/. *Champagne* está disponível em http://www.tanimoto.nuee.nagoya-u.ac.jp/ fukushima/mpegftv/. *Book-arrival* está disponível em http://sp.cs.tut.fi/mobile3dtv/stereo-video/

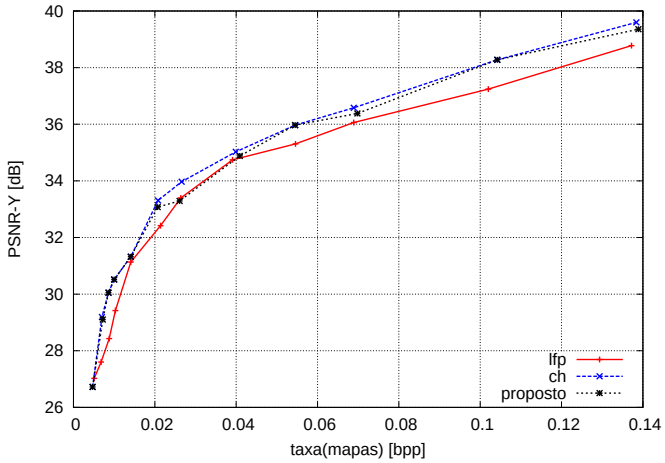

Fig. 4: *Ballet* câmera 1

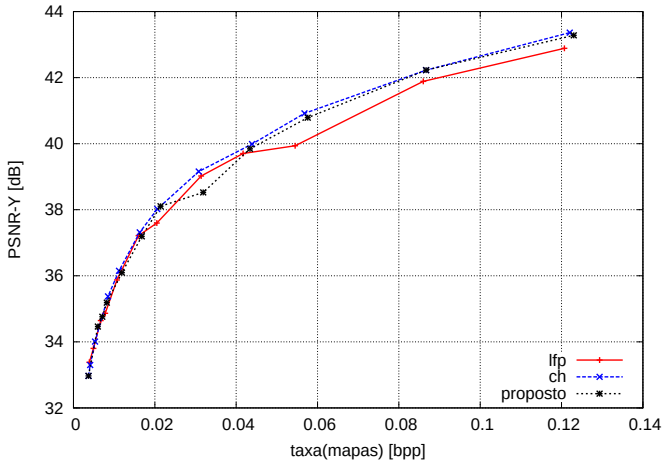

Fig. 5: *Breakdancers* câmera 1

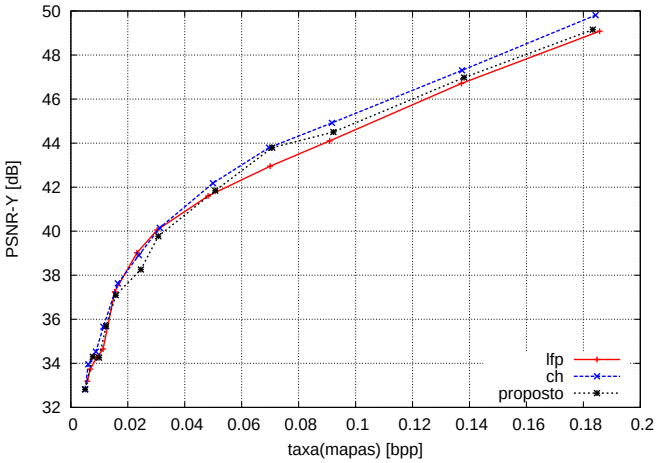

Fig. 6: *Book-arrival* câmera 9

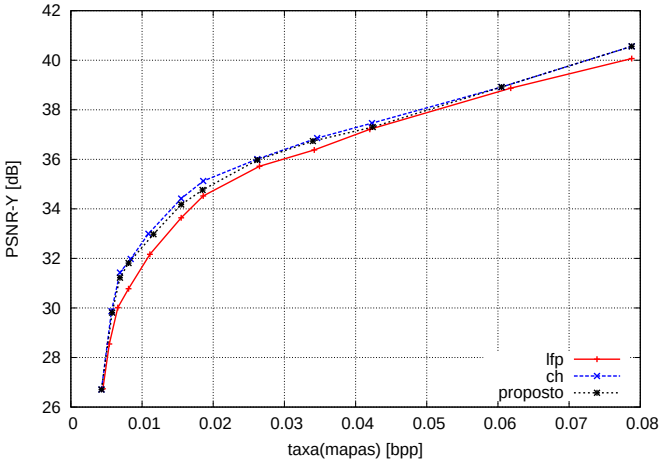

Fig. 7: *Champagne* câmera 40

A proposta de usar, além da função linear, as funções constante e quadrática na aproximação do resíduo de predição permitiu uma melhor otimização. Sobretudo, devido à flexibilidade do algoritmo de escolher a aproximação com menor custo em termos de taxa-distorção. Além disso, o uso de quantizadores treinados propiciou uma melhor alocação de *bits*.

V. Conclusões

Neste trabalho foi proposto usar no algoritmo LFP um esquema de aproximação do resíduo de predição usando, além de funções lineares, funções constantes e quadráticas, e ainda, usar um esquema de quantização em função da taxa pretendida.

Os mapas codificados com o LFP, empregando os esquemas propostos, quando utilizados em algoritmos DIBR para sintetizar vistas virtuais, fornecem resultados com melhor qualidade visual em relação aos resultados alcançados com o LFP original. Este por sua vez, exibe resultados superiores aos resultados dos algoritmos propostos na literatura.

Como trabalhos futuros, sugerimos incluir no algoritmo, técnicas de predição temporal e estender para a codificação de vídeos de mapa de profundidades.

Referências

[1] Philips Applied Technologies. *MPEG-C PART 3: Enabling the Introduction of Video Plus Depth Contents*. 2008, Suresnes, France.

[2] Ravi Krishnamurthy; Bing-Bing Chai; Hai Tao; Sriram Sethuraman. *Compression and Transmission of Depth Maps for Image-Based Rendering*. ICIP, 2001. Vol. 3. Pág. 828-831.

[3] Michel Sarkis; Wagar Zia; Klaus Diepold. *Fast Depth Map Compression and Meshing with Compressed Tritree*. 9th Asian Conference on Computer Vision, 2010 . Vol. 5995. Pág. 44-55.

[4] Rodrigues, N.M.M.; Pagliari, C.L.; da Silva, E.A.B.; de Faria, S.M.M.; Perez, M.M.; de Carvalho, M.B. . *Mutiscale Recurrent Pattern Matching Approach for Depth Map Coding*. Picture Coding Symposium. Pág. 294-297.

[5] P. Merkle; Y. Morvan; A. Smolic; D. Farin; K. Muller; P.H.N. de With; T. Wiegand. *The effects of multiview depth video compression on multiview rendering*. Elsevier, Image Communication, 2009. Vol. 24. Pág. 73-88.

[6] Lucas, L.F.R.; Rodrigues, N.M.M.; Pagliari, C.L.; da Silva, E.A.B.; de Faria, S.M.M. . *Efficient Depth Map using Linear Residue Approximation*. International Conference on Image Processing, 2012. Pág. 1305-1308 .

[7] H. Schwarz; K. Wegner. *Test Model under Consideration for HEVC Based 3D Video Coding*. ISO/IEC JTC1/SC29/WG11 MPEG, Doc. M12350, Nov. 2011, Geneva, Switzerland.

[8] J. Armando Domínguez-Molina; Graciela González-Farías; Ramón M. Rodríguez-Dagnino. *A practical procedure to estimate the shape parameter in the generalized Gaussian Distribution*.

[9] Joel Max. *Quantizing for Minimum Distortion*. IRE Transations on Information Theory, 1960. Vol. 1. Pág. 7-12.

[10] M. Tanimoto, T. Fujji, and K. Suzuki. *View synthesis algorithm in view synthesis reference software 3.5*. Documeno M16090, ISO/IEC JTC1/SC29/WG11 (MPEG), Maio, 2009.