GRAPH BASED TRANSFORMS FOR DEPTH VIDEO CODING

Woo-Shik Kim *

Video & Image Processing
Systems and Applications R&D Center
Texas Instruments Inc.
Dallas, TX 75243

Sunil K. Narang, Antonio Ortega

Signal and Image Processing Institute
Ming Hsieh Dept. of Electrical Engineering
University of Southern California
Los Angeles, CA 90089

ABSTRACT

In this paper a graph-based transform is proposed as an alternative to the discrete cosine transform. An image or video signal is represented as a graph signal, where the graph is generated so as not to cross an image edge in a local region, i.e., square block. Then, spectral representation of graph signal is used to form transform kernels by finding eigenvectors of Laplacian matrix of the graph. This method requires to include additional information, i.e., edge map or adjacency matrix, into a bitstream so that a decoder can regenerate the exactly same graph used at an encoder. The novelty of this paper includes finding the optimal adjacency matrix and compressing it using context-based adaptive binary arithmetic coding. Coding efficiency improvement can be achieved when an image block contains arbitrarily shaped edges by applying the transform not across the edges. The proposed transform is applied to coding depth maps used for view synthesis in a multi-view video coding system, and provides 14 % bit rate savings on average.

Index Terms— transform coding, image coding, video compression

1. INTRODUCTION

DCT has been widely used for block based image and video compression. It provides an efficient way to represent the signal both in terms of coding efficiency and computational complexity as an orthogonal 2-D separable transform. However, it is known to be inefficient for coding blocks containing arbitrary shaped edges. For example, if DCT is applied to a block containing an object boundary which is neither a horizontal nor a vertical line, e.g., diagonal or round shape, or mixture of these, the resulting transform coefficients tend not to be sparse and high frequency components can have significant energy. This leads to higher bitrate and potentially highly visible coding artifacts if operating at low rate.

To solve this problem variations of DCT have been proposed, such as shape adaptive DCT [1], directional DCT [2, 3, 4], spatially varying transform [5, 6], variable block-size transform [7], direction-adaptive partitioned block transform [8], etc., in which the transform block size is changed according to edge location or the signal samples are rearranged to be aligned to the main direction of dominant edges in a block. Karhunen-Loève transform (KLT) is also used for shape adaptive transform [9] or intra prediction direction adaptive transform [10]. These approaches can be applied efficiently to certain patterns of edge shapes such as straight line with preset orientation angles; however, they are not efficient with edges having arbitrary shapes. Platelets [11] are applied for depth map coding [12], and approximate depth map images as piece-wise planar signals. Since depth maps are not exactly piece-wise planar, this representation will have a fixed approximation error.

Wavelet based approaches have also been studied, including curvelets [13], bandelets [14], contourlets [15, 16], directionlets [17], etc. Edge-adaptive wavelets have been applied for depth map coding by using shape-adaptive lifting [18] and switching between long filters in homogeneous areas and short filters over the edges [19]. Graph-based wavelets [20, 21] are proposed to preserve edge information in a depth map [22]. All these approaches try not to apply transform across the edge; however, these are not amenable to a block based coding architecture, which has been widely adopted in international standards for image and video coding such as JPEG, MPEG-2, MPEG-4, H.264/AVC, etc.

To solve these problems, we have proposed the graph based transform (GBT) as an edge adaptive block transform that represents signals using graphs, where no connection between nodes (or pixels) is set across an image edge. Note that while "edge" can refer to a link or connection between nodes in graph theory, we only use the term "edge" to refer an image edge to avoid confusion. In [23] we showed that this method can work well for depth map coding, which consists of smooth regions with sharp edges between objects in different depths.

However, our previously proposed approach [23] requires a significant bitrate to code edge map information, which restricts the performance of the transform. Also, it does not work well with images having weak edges. Therefore, optimizing the transform by considering edge strength and bitrate to code the edge map is key to achieve performance improvement. In this paper, we propose a method to find the optimal transform to improve coding efficiency by finding optimal adjacency matrix which can be directly used to derive the transform instead of using edge detection method. We also propose an efficient method for coding the adjacency matrix using context-based adaptive binary arithmetic coding (CABAC) [24]. Experimental results shows that the proposed method can work well with various kind of depth maps both containing strong and weak edges. Consequently, the number of blocks coded by GBT has greatly increased, and bitrate reduction of 14 % on average is observed.

The rest of the paper is organized as follows. In Section 2, the proposed GBT is described, i.e., how to construct a transform. In Section 3 it is described how to apply GBT for image coding, followed by experimental results and conclusions in Sections 4 and 5, respectively.

*Send correspondence to wskim@ti.com.

2. CONSTRUCTION OF GRAPH BASED TRANSFORM

The transform construction procedure consists of three steps: (i) edge detection on a residual block, (ii) generation of a graph from pixels in the block using the edge map (iii) construction of transform matrix from the graph.

In the first step, after the intra/inter prediction, edges are detected in a residual block based on the difference between the neighboring residual pixel values. A simple thresholding technique can be used to generate the binary edge map. Then, the edge map is compressed and included into a bitstream, so that the same transform matrix can be constructed at the decoder side.

In the second step, each pixel position is regarded as a node in a graph, G, and neighboring nodes are connected either by 4-connectivity or 8-connectivity, unless there is edge between them. From the graph, the adjacency matrix $\mathbf{A}$ is formed, where $\mathbf{A}(i,j) = \mathbf{A}(j,i) = 1$ if pixel positions i and j are immediate neighbors not separated by an edge. Otherwise $\mathbf{A}(i,j) = \mathbf{A}(j,i) = 0$. The adjacency matrix is then used to compute the degree matrix $\mathbf{D}$, where $\mathbf{D}(i,i)$ equals the number of non-zero entries in the i-th row of $\mathbf{A}$, and $\mathbf{D}(i,j) = 0$ for all $i \neq j$.

In the third step, from the adjacency and the degree matrices, the Laplacian matrix is computed as $\mathbf{L} = \mathbf{D} - \mathbf{A}$ [25]. Then, projecting a signal G onto the eigenvectors of the Laplacian $\mathbf{L}$ yields a spectral decomposition of the signal, i.e., it provides a "frequency domain" interpretation of the signal on the graph. Thus, a transform matrix can be constructed from the eigenvectors of the Laplacian of the graph. Since the Laplacian $\mathbf{L}$ is symmetric, the eigenvector matrix $\mathbf{E}$ can be efficiently computed using the well-known cyclic Jacobi method [26], and its transpose, $\mathbf{E}^{\mathrm{T}}$, is taken as GBT matrix. Note that the eigenvalues are sorted in descending order, and corresponding eigenvectors are put in the matrix in order. This leads to transform coefficients ordered in ascending order in frequency domain.

A problem in this three-step approach is how to find an optimal edge map in the first step in order to get the best transform in the third step. Considering that an edge map is used to generate an adjacency matrix in the second step, it will be easier to optimize the adjacency matrix directly to solve this problem. Therefore, we propose to combine the first and second steps. Instead of generating the edge map explicitly, we can find the best transform kernel for the given block signal by searching the optimal adjacency matrix. When 4-neighbor connectivity is considered in a 4×4 block, there are 12 horizontal edges and 12 vertical edges. Accordingly there are 2^{24} possible adjacency matrices.

Instead of searching the whole space to find the most optimal adjacency matrix, a greedy algorithm can be applied. By defining a cost function, the cost for including each edge can be calculated. Then, the number of edges are increased from zero to 24, leading to stages 0 to 24. At stage 0 the cost is calculated when there is no edge. At stage 1 the cost is calculated for each edge location by setting one of them as an edge at a time. The one with the minimal cost is selected as the optimal edge at stage 1. At the next stage, the edge found in the previous stage is included, and an additional edge is found as in the stage 1 excluding the edge found in stage 1. In this manner, at stage k, one additional edge is included in addition to all the edges found until stage $k-1$. We can calculate the cost for each stage by including an additional edge, and choose the best stage which results in the minimal cost.

Then, this will give the optimal adjacency matrix. Note that this would be less optimal than the adjacency matrix found by full search in terms of coding efficiency. However, this will be more optimal than full search approach or edge map based approach considering tradeoff between computational complexity to calculate the adjacency matrix and coding efficiency.

The equations below show the cost function to search the optimal adjacency matrix using the greedy algorithm. First, we define a coefficient cost function as

$$\mathrm{Cost}_{\mathrm{coeff}} = \mathbf{f}^{\mathrm{T}}\mathbf{L}\mathbf{f} = \frac{1}{2}\sum_{i,j} a_{ij}(f_i - f_j)^2, \tag{1}$$

where $\mathbf{f}$ is a vector of the input depth map block, f_i is the i-th element in this vector, a_{ij} is the corresponding element in the adjacency matrix. Then, we divide this with the square of quantization step size, Q, and take log base two to get the coefficient bitrate as

$$\mathrm{Cost}_{\mathrm{coeff\_rate}} = \log_2\left(\frac{\sum_{i,j} a_{ij}(f_i - f_j)^2}{2Q^2}\right). \tag{2}$$

The final cost is represented as the sum of the coefficient bitrate and edge bitrate as

$$\begin{aligned} \mathrm{Cost} &= \mathrm{Cost}_{\mathrm{coeff\_rate}} + k\mathrm{Cost}_{\mathrm{edge\_rate}} \\ &= \log_2\left(\frac{\sum_{i} a_{ij}(f_i - f_j)^2}{2Q^2}\right) + k \cdot m, \end{aligned} \tag{3}$$

where m is the bitrate to code the adjacency matrix, which can be represented using 24 bits and further compressed using entropy coding. The scaling factor k can be applied to balance the coefficient rate and edge rate.

Transform coefficients are computed as follows. For an $N \times N$ block of residual pixels, form a one-dimensional input vector $\mathbf{x}$ by concatenating the columns of the block together into a single $N^2 \times 1$ dimensional vector, i.e., $\mathbf{x}(Nj+i) = X(i,j)$ for all $i,j = 0,1,...,N-1$. The GBT transform coefficients are then given by $\mathbf{y} = \mathbf{E}^{\mathrm{T}} \cdot \mathbf{x}$, where $\mathbf{y}$ is also an $N^2 \times 1$ dimensional vector.

In terms of computational complexity, it can be noticed that the proposed approach has $O(N^2)$ complexity, while the full search requires $O(2^N)$. Therefore, significant complexity reduction is achieved. Compared to the edge detection method, which can be regarded as having $O(N)$ complexity, the proposed method has higher complexity. However, it would be difficult for the edge detection method to get a good result with a fixed threshold value. Therefore, the proposed method using the greedy algorithm can provide the optimal solution at a reasonable complexity.

3. IMAGE AND VIDEO COMPRESSION USING GBT

In this section, we describe how GBT is applied for image and video compression. Usually transform coding is performed on the residual signal after prediction coding such as spatial prediction of temporal prediction, followed by quantization. GBT can be also applied to the residual signal, and the transform coefficients can be quantized as other transform coding methods. We use a uniform scalar quantizer for transform coefficients followed by entropy coding. Unlike DCT which uses zigzag scan of transform coefficients for entropy coding, GBT does not need any such arrangement since its coefficients are already arranged in ascending order in frequency domain.

To achieve the best performance one can choose between DCT and GBT for each block. For example for each block the rate-distortion (RD) cost can be calculated for both DCT and GBT, and the best one can be selected. Overhead indicating the transform that was chosen can be encoded into the bitstream for each block, and

the adjacency matrix information is provided only for blocks coded using GBT.

We propose an efficient way to compress the adjacency matrix using CABAC. For a given 4×4 block the adjacency matrix can be formed considering 4 neighbor connectivity. Though the matrix size is 16×16, only 24 elements can be changed according to the connectivity, and other elements will have fixed values. These elements have binary value already, i.e., connected or not. However, better performance can be achieved if correlation between links are exploited. Fig. 1 shows nodes and links which can affect the adjacency matrix, where 1 to 12 links are related to horizontal edges as shown in the left image, and 13 to 24 links are related to vertical edges as shown in the right image.

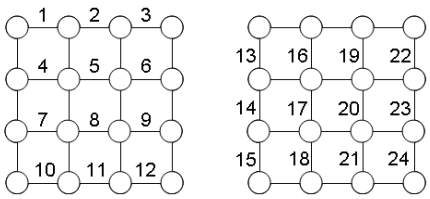


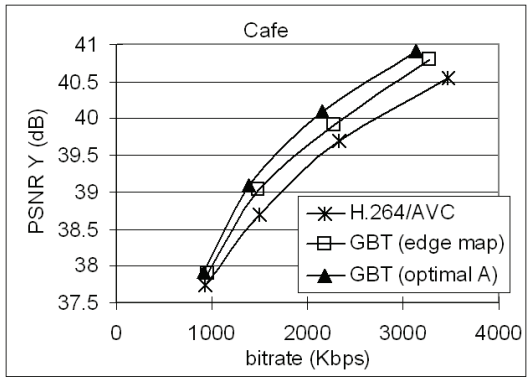
Fig. 1. Adjacency matrix coding for 4×4 block with 4 neighbor connectivity. Circle denotes node or pixel and solid line represents link between nodes. Left image shows links related to horizontal edges, and right image shows links related to vertical edges.

From the typical node structure of a square image block, it can be inferred that there will be high correlation among links in same horizontal or vertical line, i.e., $\{1, 2, 3\}$, $\{4, 5, 6\}$, $\{7, 8, 9\}$, $\{10, 11, 12\}$ for horizontal lines, and $\{13, 14, 15\}$, $\{16, 17, 18\}$, $\{19, 20, 21\}$, $\{22, 23, 24\}$ for vertical lines. However, the first link in each group cannot use the correlation. Therefore, we set up two contexts according to link locations. For links 1, 4, 7, 10, 13, 16, 19, and 22, which are the first nodes in each line, context is set to zero. In this case, no prediction is performed to make binary decisions, so-called *bins*, and bin is set to one if connected, and set to zero if not. For the other links, context is set to one, and bin is set to zero if its left (in case of horizontal line links) or upper (in case of vertical line links) link has same connection status (i.e., connected or not connected) as the current link, otherwise bin is set to one. Then, bins are compressed using binary arithmetic coding.

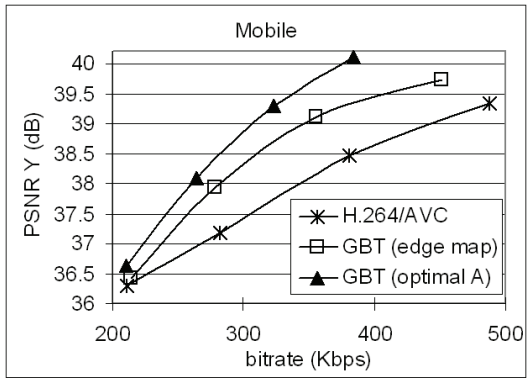
4. EXPERIMENTAL RESULTS

In this section, the performance of GBT is evaluated using various 3-D video test sequences, which consist of left and right reference view video along with their depth video. The intermediate view is synthesized using the MPEG view synthesis reference software (VSRS) 3.0 [27] with the compressed depth video either by using conventional DCT based transform or the proposed GBT. The implementation is based on H.264/AVC reference software JM17.1. 15 frames are coded for each sequence, where the first frame is coded as intra frame and the others as inter frame. No adaptive rounding control is applied for either DCT or GBT, and a fixed rounding offset of 1/3 is used. The transform mode is signaled for each 4×4 block to indicate the best transform between the H.264/AVC integer transform which is a modification of DCT and the proposed GBT. GBT kernel is generated in two ways. First, normal edge detection scheme is applied to detect edges in a block. In this case, for the blocks coded using GBT, the edge map is losslessly encoded and sent to the decoder. Secondly, instead of finding edge map, we find optimal adjacency matrix using the method described in Section 2, where the adjacency matrix is losslessly encoded and sent to the decoder. We compare these two methods to the case where the depth maps are encoded using H.264/AVC.

Fig. 2 shows the RD curve comparisons between the proposed methods and H.264/AVC, where the bitrates for GBT cases include transform selection bits and edge map or adjacency matrix information bits in addition to two depth map coding bits. QP values of 24, 28, 32, and 36 are used to encode depth maps. PSNR is calculated by comparing the rendered view without any compression of depth maps as the reference and the synthesized video using the decoded depth maps [28]. Note that in both cases, decoded video is used for view synthesis to measure the distortion due to depth map compression.



(a) Book Arrival



(b) Mobile

Fig. 2. RD curve comparison between GBT and H.264/AVC, where GBT is formed using edge detection or by finding optimal adjacency matrix (x-axis: bitrate to code left and right depth maps and edge map or adjacency matrix information in case of GBT, y-axis: Y PSNR of the synthesized intermediate view): (a) Cafe (b) Mobile.

From the RD curves in Fig. 2, it can be noticed the GBT generated using optimal adjacency matrix performs better than H.264/AVC, while the performance of the GBT generated using edge detection scheme lies in the middle. It is observed that significant amount of bitrate saving can be achieved by GBT along with PSNR improvement. The bitrate savings increase as overall bitrate increases. This is because more blocks are chosen to be coded using GBT at high bitrates, since the portion of additional bits for adjacency matrix in total bitrate reduces. It can be also noticed that different performance gain is achieved in each test sequence. The performance of GBT depends on the number of edges in a frame, and the edge strength. If there is large amount of noise around an object boundary, GBT may not provide much gain over DCT. Therefore, large gain can be achieved for sequences containing many strong edges, such as Mobile and Cafe. In addition, the performance also can be affected by sensitivity of view synthesis procedure to depth map quality as analyzed in [29]. All the results for 10 test sequences are given in Table 1, where the coding efficiency is represented using BD-bitrate (BDBR) [30]. Note that minus sign means bitrate reduction.

Table 1. BD-bitrate results of GBT compared to H.264/AVC.

Sequence	Bitrate reduction (BDBR) (%)	
	GBT (edge map)	GBT (optimal A)
Cafe	-12.6	-20.7
Newspaper	-1.9	-6.4
Book Arrival	-17.1	-15.7
Balloons	-1.8	-7.1
Champagne Tower	-2.1	-9.8
Kendo	-1.3	-3.9
Mobile	-16.4	-24.0
Car Park	-15.4	-19.1
Hall	-4.4	-6.9
Street	-13.9	-24.6
Average	-8.7	-13.8

5. CONCLUSIONS AND FUTURE WORK

We have proposed a new transform coding method for image and video compression based on a spectral representation of a graph signal. GBT is applied in a hybrid manner for each block, so that best transform between GBT and DCT can be selected to achieve best RD performance. We also have proposed a method to find an optimal way to construct transform kernels using a greedy algorithm, and an efficient method to compress adjacency matrix using CABAC. Experimental results shows coding efficiency improvement of 14 % on average when it is used to compress depth map images. Future research includes analysis of GBT performance and extension to general image and video compression.

6. REFERENCES

[1] W. Philips, "Comparison of techniques for intra-frame coding of arbitrarily shaped video object boundary blocks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 7, pp. 1009–1012, Oct. 1999.

[2] B. Zeng and J. Fu, "Directional discrete cosine transforms for image coding," in *Proc. of IEEE Int. Conf. Multimedia and Expo, ICME 2006*, Toronto, Canada, Jul. 2006, pp. 721–724.

[3] J. Fu and B. Zeng, "Directional discrete cosine transforms: A theoretical analysis," in *Proc. of IEEE Int. Conf. Acoustics, Speech and Signal Proc., ICASSP 2007*, Honolulu, HI, USA, Apr. 2007, vol. I, pp. 1105–1108.

[4] B. Zeng and J. Fu, "Directional discrete cosine transforms - a new framework for image coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 3, pp. 305–313, Mar. 2008.

[5] C. Zhang, K. Ugur, J. Lainema, and M. Gabbouj, "Video coding using spatially varying transform," in *Proc. of 3rd Pacific Rim Symposium on Advances in Image and Video Technology, PSIVT 2007*, Tokyo, Japan, Jan. 2009, pp. 796–806.

[6] C. Zhang, K. Ugur, J. Lainema, and M. Gabbouj, "Video coding using variable block-size spatially varying transforms," in *Proc. of IEEE Int. Conf. Acoustics, Speech and Signal Proc., ICASSP 2009*, Taipei, Taiwan, Apr. 2009, pp. 905–908.

[7] M. Wien, "Variable block-size transforms for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 604–613, Jul. 2003.

[8] C.-L. Chang, M. Makar, S. S. Tsai, and B. Girod, "Direction-adaptive partitioned block transform for color image coding," *IEEE Trans. Image Proc.*, vol. 19, no. 7, pp. 1740–1755, Jul. 2010.

[9] T. Sikora, S. Bauer, and B. Makai, "Efficiency of shape-adaptive 2-D transforms for coding of arbitrarily shaped image segments," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, no. 3, pp. 254–258, Jun. 1995.

[10] Y. Ye and M. Karczewicz, "Improved H.264 intra coding based on bi-directional intra prediction, directional transform, and adaptive coefficient scanning," in *Proc. of IEEE Int. Conf. Image Proc., ICIP 2008*, San Diego, CA, USA, Oct. 2008, pp. 2116–2119.

[11] R. Willett and R. Nowak, "Platelets: A multiscale approach for recovering edges and surfaces in photon-limited medical imaging," *IEEE Trans. Medical Imaging*, vol. 22, no. 3, pp. 332–350, Mar. 2003.

[12] Y. Morvan, D. Farin, and P. H. N. de With, "Platelet-based coding of depth maps for the transmission of multiview images," in *Proc. of SPIE: Stereoscopic Displays and Applications (2006)*, San Jose, CA, USA, Jan. 2006, vol. 6055.

[13] E. Candès and D. Donoho, *Curvelets - a surprisingly effective nonadaptive representation for objects with edges*, Vanderbilt University Press, 1999.

[14] E. Le Pennec and S. Mallat, "Sparse geometric image representations with bandelets," *IEEE Trans. Image Proc.*, vol. 14, no. 4, pp. 423–438, Apr. 2005.

[15] M. N. Do and M. Vetterli, "The contourlet transform: An efficient directional multiresolution image representation," *IEEE Trans. Image Proc.*, vol. 14, no. 12, pp. 2091–2106, Dec. 2005.

[16] D. D.-Y. Po and M. N. Do, "Directional multiscale modeling of images using the contourlet transform," *IEEE Trans. Image Proc.*, vol. 15, no. 6, pp. 1610–1620, Jun. 2006.

[17] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli, and P.L. Dragotti, "Directionlets: Anisotropic multidirectional representation with separable filtering," *IEEE Trans. Image Proc.*, vol. 15, no. 7, pp. 1916–1933, Jul. 2006.

[18] M. Maitre and M. N. Do, "Joint encoding of the depth image based representation using shape-adaptive wavelets," in *Proc. of IEEE Int. Conf. Image Proc., ICIP 2008*, San Diego, CA, USA, Oct. 2008, pp. 1768–1771.

[19] I. Daribo, C. Tillier, and B. Pesquet-Popescu, "Adaptive wavelet coding of the depth map for stereoscopic view synthesis," in *Proc. of 2008 IEEE 10th Workshop on Multimedia Signal Processing*, Queensland, Australia, Oct. 2008, pp. 413–417.

[20] S. K. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter-banks for graph structured data," *submitted to IEEE Trans. Signal Proc.*

[21] E. Martínez-Enríquez, F. Díaz de María, and A. Ortega, "Video encoder based on lifting transforms on graphs," in *Proc. of IEEE Int. Conf. Image Proc., ICIP 2011*, Brussels, Belgium, Sep. 2011.

[22] A. Sanchez, G. Shen, and A. Ortega, "Edge-preserving depth-map coding using tree-based wavelets," in *Prof. of Asilomar Conference on Signals and Systems*, Pacific Grove, CA, USA, Nov. 2009.

[23] G. Shen, W.-S. Kim, S. K. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth-map coding," in *Proc. of 28th Picture Coding Symposium, PCS 2010*, Nagoya, Japan, Dec. 2010.

[24] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 620–636, Jul. 2003.

[25] D. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," in *Elsevier: Appplied and Computational Harmonic Analysis*, Apr. 2010, vol. 30, pp. 129–150.

[26] H. Rutishauser, "The jacobi method for real symmetric matrices," *Numerische Mathematik*, vol. 9, no. 1, Nov. 1966.

[27] M. Tanimoto, T. Fujii, and K. Suzuki, "View synthesis algorithm in view synthesis reference software 2.0 (VSRS2.0)," Document M16090, ISO/IEC JTC1/SC29/WG11, Feb. 2009.

[28] Video, "Description of exploration experiments in 3D video coding," Document N11274, ISO/IEC JTC1/SC29/WG11, Apr. 2010.

[29] W.-S. Kim, A. Ortega, J. Lee, and H. Wey, "3-D video quality improvement using depth transition data," in *Proc. of IEEE Int. Workshop on Hot Topics in 3D, Hot 3D 2011*, Barcelona, Spain, Jul. 2011.

[30] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," Document VCEG-M33, ITU-T SG16 Q.6, Apr. 2001.