GRAPH-BASED REPRESENTATION AND CODING OF 3D IMAGES FOR INTERACTIVE MULTIVIEW NAVIGATION

Benedicte Motz [*], *Gene Cheung* [#], *Pascal Frossard* [$]

[*] The Graduate University for Advanced Studies, [#] National Institute of Informatics, [$] EPFL

ABSTRACT

Instead of lossily coding depth images resulting in undesirable geometric distortion, graph-based representation (GBR) describes disparity information as a graph with a controllable accuracy. In this paper, we propose a more compact graphical representation called GBR-plus to code both disparity and color information of a target view given a reference view. Specifically, first we differentiate between disocclusion holes (occluded spatial regions in the reference view) and rounding holes (insufficiently sampled regions in the reference view) in the synthesized target view, so that the decoder can optionally complete rounding holes via signal interpolation without coding overhead. Second, we use a compact graphical representation to delimit disparity-shifted boundaries of objects in the target view, which is coded losslessly. Finally, color pixels in disocclusion holes are predicted using adjacent background pixels as predictors, and prediction residuals in a local neighborhood are coded using Graph Fourier Transform (GFT). Experimental results show that GBR-plus outperforms previous GBR, and has comparable performance as HEVC at mid to high bitrates with lower encoder complexity.

Index Terms— 3D imaging, graphical representation, graph Fourier transform

1. INTRODUCTION

In a typical multiview imaging system [1], a 1D array of closely spaced cameras capture both color and depth images of a 3D scene from different viewpoints, which are then coded and transmitted to the decoder for rendering of additional intermediate virtual views via depth-image-based rendering (DIBR) [2]. Coding depth images—which convey important disparity information—using conventional transform coding plus lossy quantization [3–6] leads to uncontrolled geometric distortion, resulting in undesirable bleeding artifacts in DIBR-synthesized images [7]. One alternative is to employ edge-adaptive transforms [8–10] and wavelets [11], but this requires coding of object contours [12] as an additional overhead.

In response, the authors in [13] argued that, since disparity information provided by a depth image is akin to motion information in motion prediction during single-view video coding, like motion vectors disparity information can be coarsely represented but should be *losslessly* coded. Hence they proposed a compact *graph-based representation* (GBR) that losslessly encodes disparity information to displace entire pixel patches from one or more reference view(s) to a target view. Because GBR suffers no geometric errors stemming from lossy coding, GBR can outperform conventional depth image coding schemes at high rates [13].

In this paper, we propose a more compact graphical representation called GBR-plus to code both disparity and color information of a neighboring target view given a reference view. We focus on an interactive multiview streaming (IMVS) scenario where a user can interactively request new view $v \pm 1$, having reconstructed view v as reference [14, 15]. Specifically, first we differentiate between disocclusion holes (occluded spatial regions in the reference view) and rounding holes (insufficiently sampled regions in the reference view) in the synthesized target view, so that the decoder can optionally complete rounding holes via signal interpolation without coding overhead. Second, we use a compact graphical representation to delimit disparity-shifted boundaries of objects in the target view, which is coded losslessly and efficiently as two binary images using JBIG [16]. Finally, color pixels in disocclusion holes are predicted using adjacent background pixels as predictors, and prediction residuals in a local block are coded using Graph Fourier Transform (GFT) [17], where the underlying graph can be constructed synchronously at both encoder and decoder given the common disparity information described in GBR-plus. Experimental results show that GBR-plus outperforms previous GBR, and has comparable performance as HEVC at mid to high bitrates with lower encoder complexity.

2. GBR OVERVIEW

Instead of lossily coding deth maps, [13] proposed GBR to code graphically described disparity information *losslessly* for subsequent view(s) given transmitted first view as reference. It was demonstrated that at high rates, GBR outperforms competing disparity representations due to its compactness and lossless coding. We overview the operations in GBR next.

The aim of GBR is to first represent disparity information graphically, and then losslessly code the chosen representation for accurate reconstruction at the decoder. For simplic-

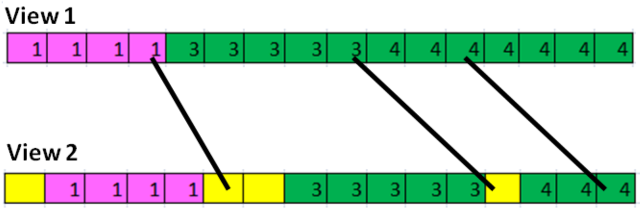


Fig. 1: Example of GBR representing disparity for view 2 given view 1 as reference. Same color pixels correspond to the same object. Numbers are disparities in pixel displacement.

ity, we assume here that viewpoint images are rectified, so that the disparities of objects across two neighboring frames are purely horizontal. Hence we can focus on describing disparities for a given row i across two different views.

Specifically, a pixel row in an original view is divided into *patches*, each containing pixels with the same disparity value. Edges are then drawn from patch boundaries in the original view to pixel locations in the target view that delimit the new patch locations. As an example, in Fig. 1 there are three edges that designate end locations of three patches in the target view. There are new *appearing pixels* in the target view (shown in yellow) that either: i) enter into the field of view due to view change, or ii) newly appear in the target view between displaced patches. Appearing pixels are coded separately. The coding overhead of GBR is hence the coding of the graph that indicates patch displacements, plus the coding of new pixels.

3. GBR-PLUS

We design a new graphical representation called *GBR-plus*—an improvement over GBR in [13]—for IMVS [14, 15], where a user interactively requests from a server data for a neighboring camera view $v \pm 1$ (called *target view*) having already reconstructed camera view v (called *reference view*).

3.1. Overview of GBR-plus

First, GBR-plus describes disparities of *objects* (as opposed to patches) across two frames, where by object we mean a physical surface area that is visible from both views. This means that the object can change (slightly) in size from one view to another, due to rounding of per-pixel disparity values. If an object has a width of N pixels in reference view and M pixels in the target view where $N < M$, then there is a N-to-M pixel mapping from reference to target view. We propose an interpolation method to perform this mapping (to be discussed in Section 3.3). In contrast, GBR maps N pixels to N grid locations in the target view, leaving $M - N$ empty locations, whose pixels are explicitly coded. Unlike disocclusion, because these $M - N$ missing pixels (called *rounding holes*) are from the same object surface, they often can be interpolated with sufficiently high quality using neighboring pixels. Thus our decoder-side N-to-M pixel mapping approach can potentially achieve bit saving with little distortion penalty.

We illustrate the graphical representation of disparity in GBR-plus in the example shown in Fig. 2. We see that the left and right boundaries of object two, $v_L(2)$ and $v_R(2)$ in the reference view are mapped to boundaries $u_L(2)$ and $u_R(2)$ in the target view, and the object's width has changed from 8 to 9 pixels. The decoder can perform this 8-to-9 pixel mapping using our proposed interpolation method. In contrast, as shown in Fig. 1 GBR would treat the same surface area as two patches (each of the same disparity value), and the one missing pixel is coded explicitly.

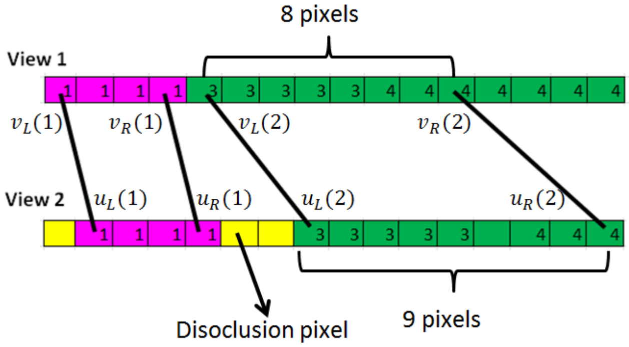


Fig. 2: Example of GBR-plus connecting object boundaries between the reference view (view 1) and the target view (view 2).

Second, disparities are represented as edges connecting object boundaries in the reference view to boundaries in the target view. Because the edges never cross from reference to target view, *i.e.*, the order of left / right boundaries of objects are the same in both views, we can represent the start and end points of edges as two binary images coded using JBIG [16].

Finally, to efficiently code pixels in *disocclusion holes* (spatial areas in target view that are occluded by foreground objects in the reference view) in the target view, we first perform intra-prediction using neighboring background pixels as predictor to lower the signal energy. We then construct a graph to connect disocclusion hole pixels and compute a graph Fourier transform (GFT) [17] for coding of the prediction residuals. We describe the details in these steps next.

3.2. Intra-Prediction and Coding of Disocclusion Pixels

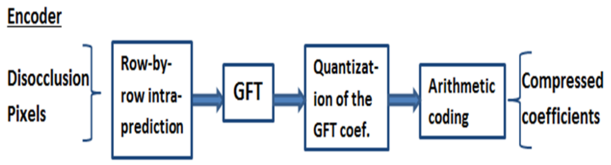


Fig. 3: Block diagram of the encoder. At the decoder the inverse operations are executed in the reverse order.

In contrast to disparity, color pixels in disocclusion holes can and should be lossily coded to achieve a good RD tradeoff. To reduce their signal energy, we first perform row-by-row intra-prediction as follows. For a given disocclusion pixel

y_i, denote the closest left and right available mapped color pixels in the same row in the target image as x_L and x_R. Between x_L and x_R, the pixel x_i with the smaller disparity value is denoted as the *background pixel predictor*. The prediction residual is then computed as $z_i = x_i - y_i$.

To code prediction residuals of the disocclusion pixels, we use *Graph Fourier Transform* (GFT) [17] so that correlations among neighboring disocclusion pixels are maximally exploited during transform coding for compression gain. We first construct a graph to connect disocclusion pixels as follows. For each $K \times K$ pixel block in the target view image, we connect each disocclusion pixel with its four closest disocclusion pixel neighbors within a neighborhood circle of radius r. The radius r is chosen large enough so that the likelihood of disconnected pixel patches or individual isolated pixels is small. On the other hand, too large an r will result in connecting pixels that are not correlated in the first place. An example is shown in Fig. 4, where 13 disocclusion pixels are connected to form a graph.

Next, we compute weight $w_{i,j}$ for each edge connecting pixel i and j, where $w_{i,j}$ is an exponential term of the Euclidean square of the inter-pixel distance:

$$w_{i,j} = \exp\left(-\frac{\|p_i - p_j\|^2}{\sigma_I^2}\right) \tag{1}$$

where p_i is the 2D coordinate of pixel x_i, and σ_I is a scaling parameter.

Given edge weights $w_{i,j}$, we define the *adjacency matrix* $\mathbf{A}$, where $A_{i,j} = w_{i,j}$, and the diagonal *degree matrix* $\mathbf{D}$, where $D_{i,i} = \sum_j A_{i,j}$. Finally, the *graph Laplacian matrix* $\mathbf{L}$ is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$ [17]. Rows of GFT Φ are the eigen-vectors of $\mathbf{L}$. We thus project prediction residual $\mathbf{z}$ onto the GFT basis, *i.e.* $\mathbf{Z} = \Phi\mathbf{z}$, then the GFT coefficients $\mathbf{Z}$ are quantized and entropy-coded for transmission to the decoder.

Because the decoder, having received the graphical representation of the disparity information, can deduce the exact same locations of disocclusion pixels as the encoder, for each $K \times K$ block it can also construct the same graph connecting the disocclusion pixels, compute the edge weights and derive the same GFT. Thus there is no need to explicitly transmit extra side information to inform the decoder the GFT used for each block, as done in [10].

3.3. N-to-M Pixel Mapping at the Decoder

To perform the N-to-M pixel mapping for the same object, we propose the following interpolation method called `graphInter`[1]. The idea is to interpolate a new set of evenly spaced M pixels in the target view given available evenly spaced N pixels in the reference view. Specifically, we first construct a graph of N nodes, where each node pair is separated by a distance $1/(N-1)$, representing the N pixels in the reference view. Then we construct in addition $M-2$ nodes, where together with the first and last reference nodes represent the M new pixels in the target view. A node pair in these M nodes are at distance $1/(M-1)$ apart. Fig. 5 shows an example of 4-to-5 pixel mapping.

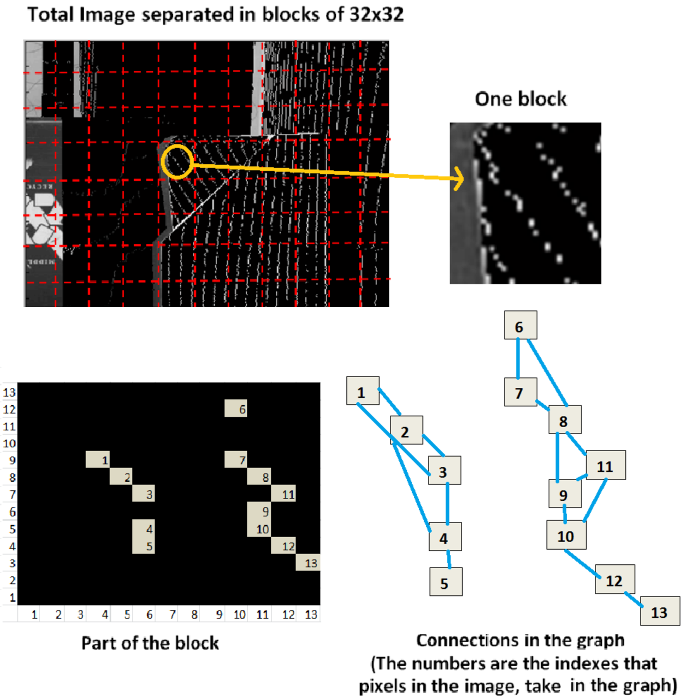

Total Image separated in blocks of 32x32
One block
Part of the block
Connections in the graph
(The numbers are the indexes that
pixels in the image, take in the graph)
1
2
3
4
5
6
7
8
9
10
11
12
13

Fig. 4: Graph construction of the color pixels in image `Plastic`, where only the prediction residuals have to be coded.

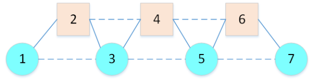


Fig. 5: Example of interpolation from 4 pixels to 5 pixels.

We next connect the each reference node to its neighboring reference nodes, and also to the two closest target nodes as well. Similar connectivity is performed for the target nodes. We compute the edge weights according to (1) based on Euclidean distance. As done previously, from the computed weights we can derive the adjacency matrix $\mathbf{A}$, the degree matrix $\mathbf{D}$, and the graph Laplacian $\mathbf{L}$. If the reconstructed signal $\mathbf{x}$ is smooth with respect to the graph on which it lives, then a *graph Laplacian regularizer* $\mathbf{x}^T\mathbf{L}\mathbf{x}$ tends to be small [19]. Denote by $\mathbf{y}$ the N reference pixels, by $\mathbf{x}$ the $M+N-2$ pixels from reference and target views, and by $\mathbf{H}$ the $N \times (M+N-2)$ matrix that selects N reference pixels from vector $\mathbf{x}$. We can thus perform the N-to-M pixel mapping via the following optimization:

$$\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \lambda\mathbf{x}^T\mathbf{L}\mathbf{x} \tag{2}$$

where the first term is a fidelity term that ensures the recon-

[1] A simpler version of the proposed method for N-to-M pixel mapping is presented in [18], which focused on interpolation but not coding.

structed signal is close to the observed N reference pixels, and the second term is a graph-signal smoothness prior [19]. The optimization (2) is solved iteratively, where in subsequent iterations the edge weight is updated using product of exponentials of Euclidean distance and of difference in pixel intensity computed in the previous iteration. See [19] for details.

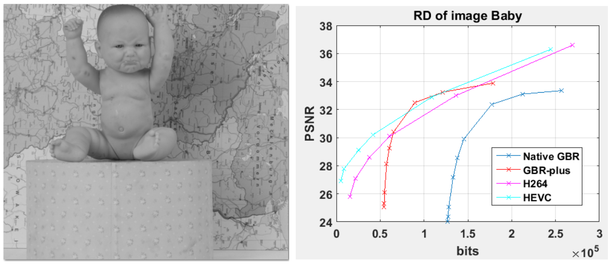
3.4. Mode Selection in GBR-plus

Given the coding and interpolation tools discussed above in GBR-plus, *coding modes* can actually be selected in an RD optimal way so that the optimal combination of tools are used for each block. Specifically, inside a given $K \times K$ block, the disparity of the objects can be coded using GBR-plus graphical representation and N-to-M pixel mapping performed at the decoder—this is mode 1. If our proposed method described in Section 3.3 results in poor interpolation quality, then the objects can be sub-divided and redefined as smaller objects, each with pixels of the same disparity value (similar to patches in GBR). Then there is a pixel-to-pixel mapping from reference to target view, and the prediction residuals of the appearing pixels can be explicitly coded, resulting in higher quality—this is mode 2. Finally, instead of explicit coding of the appearing pixels, each appearing pixel between patches can be computed as the local average of the nearest mapped reference pixels to the left and right; this is the conventional pixel interpolation method used in DIBR for rounding holes [2]. This is mode 3. At the encoder, we select the optimal mode for each block based on an RD criteria, similar to mode selection in video coding standards [20].

4. EXPERIMENTS

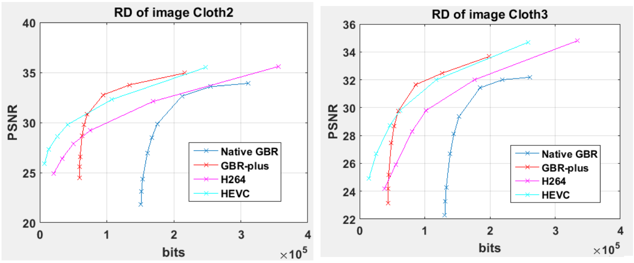
To demonstrate the coding performance of our proposed GBR-plus, we conducted the following experiments. As input we used three multiview sequences `Baby`, `Cloth2`, `Cloth3` from the Middlebury database[2]. We coded the first reference view using JPEG with a sufficiently high Quality factor to ensure a PSNR of 45dB. We measured the RD performance of the color and depth maps for the subsequently requested second target view. To induce different tradeoff points, we varied the QP for the coding of disocclusion pixels using GFT. PSNR is computed from the difference between the reconstructed second view and the camera-captured view.

We compare GBR-plus to three competing schemes. GBR is an implementation of [13] for coding of color map and disparity information, where all hole pixels are explicitly coded using our proposed GFT with no interpolation performed at the decoder. We also employed H.264 and HEVC to code color and depth maps of the second target view at the same QP using the first coded view as reference.



(a) (b)

Fig. 6: Captured first view (a) and comparison of coding performance (PSNR vs. coding rate) (b) for `Baby`.



(a) (b)

Fig. 7: Comparison of coding performance (PSNR vs. coding rate) for `Cloth2` (a) and `Cloth3` (b).

The resulting plots of image quality in PSNR versus coding rates are shown in Fig. 6(b), 7(a) and 7(b) for the three test sequences. We first observe that GBR-plus outpeforms GBR noticeably for all three sequences (up to 2.6dB gain), thanks to a combination of a more compact graphical representation, intra-prediction and GFT coding of prediction residuals, and interpolation of rounding holes at the decoder. We also see that for `Cloth2` and `Cloth3`, GBR-plus outperforms H.264 for the whole bitrate range and outperforms HEVC at mid to high bitrates, while for `Baby` GBR-plus is competitive at mid bitrates but in general worse than HEVC. The reason is because the background in `Baby` contains complex texture (as shown in Fig.6(a)) that cannot be easily interpolated, limiting the benefit of our proposed decoder-side interpolation. On the other hand, unlike HEVC, for each code block GBR-plus has much fewer number of coding modes to choose from, resulting in a lower encoder complexity.

5. CONCLUSION

Given a reconstructed reference view at the decoder, we propose a new representation GBR-plus to first compactly represent disparity of objects in a target view as a graph. The graph can be efficiently coded as two binary images coded using JBIG. Disocclusion color pixels in the target view are first predicted using neighboring background pixels, and prediction residuals are coded using graph Fourier transform (GFT). Color pixels of objects that change in size in the target view can be interpolated with the help of a graph-signal smoothness prior. Experimental results show significant coding gain over a previous representation GBR, and comparable coding performance with HEVC for three multiview sequences.

[2] http://vision.middlebury.edu/stereo/data/scenes2006/

6. REFERENCES

[1] M. Tanimoto, M. P. Tehrani, T. Fujii, and T. Yendo, "Free-viewpoint TV," in *IEEE Signal Processing Magazine*, January 2011, vol. 28, no.1.

[2] D. Tian, P.-L. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3D video," in *Applications of Digital Image Processing XXXII, Proceedings of the SPIE*, 2009, vol. 7443 (2009), pp. 74430T–74430T–11.

[3] W.-S. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila, "Depth map coding with distortion estimation of rendered view," in *SPIE Visual Information Processing and Communication*, San Jose, CA, January 2010.

[4] G. Cheung, A. Kubota, and A. Ortega, "Sparse representation of depth maps for efficient transform coding," in *IEEE Picture Coding Symposium*, Nagoya, Japan, December 2010.

[5] G. Cheung, J. Ishida, A. Kubota, and A. Ortega, "Transform domain sparsification of depth maps using iterative quadratic programming," in *IEEE International Conference on Image Processing*, Brussels, Belgium, September 2011.

[6] G. Cheung, V. Velisavljevic, and A. Ortega, "On dependent bit allocation for multiview image coding with depth-image-based rendering," in *IEEE Transactions on Image Processing*, November 2011, vol. 20, no.11, pp. 3179–3194.

[7] Patrick Ndjiki-Nya, Martin Köppel, Dimitar Doshkov, Haricharan Lakshman, Philipp Merkle, Karsten Müller, and Thomas Wiegand, "Depth image-based rendering with advanced texture synthesis for 3-d video," *Multimedia, IEEE Transactions on*, vol. 13, no. 3, pp. 453–465, 2011.

[8] G. Shen, W.-S. Kim, S.K. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *IEEE Picture Coding Symposium*, Nagoya, Japan, December 2010.

[9] G. Cheung, W. s. Kim, A. Ortega, J. Ishida, and A. Kubota, "Depth map coding using graph based transform and transform domain sparsification," in *IEEE International Workshop on Multimedia Signal Processing*, Hangzhou, China, October 2011.

[10] W. Hu, G. Cheung, A. Ortega, and O. Au, "Multiresolution graph Fourier transform for compression of piecewise smooth images," in *IEEE Transactions on Image Processing*, January 2015, vol. 24, no.1, pp. 419–433.

[11] M. Maitre, Y. Shinagawa, and M.N. Do, "Wavelet-based joint estimation and encoding of depth-image-based representations for free-viewpoint rendering," in *IEEE Transactions on Image Processing*, June 2008, vol. 17, no.6, pp. 946–957.

[12] I. Daribo, D. Florencio, and G. Cheung, "Arbitrarily shaped motion prediction for depth video compression using arithmetic edge coding," in *IEEE Transactions on Image Processing*, November 2014, vol. 23, no. 11, pp. 4696–4708.

[13] T. Maugey, A. Ortega, and P. Frossard, "Graph-based representation for multiview image geometry," in *IEEE Transactions on Image Processing*, May 2015, vol. 24, no.5, pp. 1573–1586.

[14] G. Cheung, A. Ortega, and N.-M. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," in *IEEE Transactions on Image Processing*, March 2011, vol. 20, no.3, pp. 744–761.

[15] X. Xiu, G. Cheung, and J. Liang, "Delay-cognizant interactive multiview video with free viewpoint synthesis," in *IEEE Transactions on Multimedia*, August 2012, vol. 14, no.4, pp. 1109–1126.

[16] Fumitaka Ono, William Rucklidge, Ronald Arps, and Cornel Constantinescu, "Jbig2-the ultimate bi-level image coding standard," in *Image Processing, 2000. Proceedings. 2000 International Conference on*. IEEE, 2000, vol. 1, pp. 140–143.

[17] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," in *IEEE Signal Processing Magazine*, May 2013, vol. 30, no.3, pp. 83–98.

[18] B. Motz, G. Cheung, A. Ortega, and P. Frossard, "Resampling and interpolation of DIBR-synthesized images using graph-signal smoothness prior," in *(accepted to) APSIPA ASC*, Hong Hong, December 2015.

[19] J. Pang, G. Cheung, A. Ortega, and O. C. Au, "Optimal graph Laplacian regularization for natural image denoising," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Brisbane, Australia, April 2015.

[20] Thomas Wiegand, Heiko Schwarz, Anthony Joch, Faouzi Kossentini, and Gary J Sullivan, "Rate-constrained coder control and comparison of video coding standards," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 688–703, 2003.