Re-sampling and Interpolation of DIBR-synthesized Images using Graph-signal Smoothness Prior

Benedicte Motz*, Gene Cheung†, Antonio Ortega‡ and Pascal Frossard*
* EPFL, Lausanne, Switzerland
E-mail: {benedicte.motz, pascal.frossard}@epfl.ch Tel/Fax: +41-0-21-693-56-55
† National Institute of Informatics, Tokyo, Japan
E-mail: cheung@nii.ac.jp Tel/Fax: +81-3-4212-2567
‡ University of Southern California, Los Angeles, USA
E-mail: antonio.ortega@sipi.usc.edu Tel/Fax: +1-213-740-2320

***Abstract*—In depth-image-based rendering (DIBR), a new virtual viewpoint image is synthesized by mapping color pixels from one or more reference views to the new image grid using corresponding disparity values. However, due to necessary roundings to the 2D grid, "rounding holes" appear in synthesized objects, which are typically filled using local interpolation. In this paper, leveraging on a recent 3D image compression scheme called graph-based representation (GBR) that losslessly codes disparity information, we propose instead to re-sample and interpolate missing on-grid pixels of an object at decoder using mapped off-grid color pixels as reference. Specifically, we first describe the underlying data kernel for the desired signal using a weighted graph, where the edge weights reflect non-integer distances between reference and missing pixels. A graph-signal smoothness prior is then assumed to complete missing pixel values via iterative unconstrained quadratic optimization. Experimental results show that the synthesized objects have better quality than conventional local interpolation methods.**

I. Introduction

In a typical multiview imaging system [1], a 1D array of closely spaced cameras capture both texture (color) and depth images of a 3D scene from different viewpoints. If only a conventional 2D display is available, then a client only observes one view of the scene at a time, but can interactively request a view-switch from the server to transmit data of a neighboring view—a scenario called *interactive multiview streaming* (IMVS) [2]. In such IMVS scenario, the server should exploit inherent inter-view correlation to minimize transmission rate of the requested neighboring view. Previous research has proposed efficient compression algorithms for 3D images while facilitating interactive view-switching in this IMVS scenario [2–6].

Neighboring viewpoint images capturing the same 3D scene are clearly correlated. Using geometric information provided by captured depth images, the majority of pixels in one viewpoint image can be synthesized from a neighboring one via *depth-image-based rendering* (DIBR)[1] [7]. In one notable work, [8] 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 should be losslessly coded. The alternative of lossily coding a depth image to convey disparity information [9–11] can lead to geometric errors and result in synthesized view distortions that can be estimated [12] but are difficult to contain. [8] thus proposed a compact *graph-based representation* (GBR) that losslessly encodes disparity information to displace entire pixel patches from the original view to the synthesized view. Pixels that are directly observable in the synthesized view but not the original view (due to disocclusion, out-of-view, etc) are encoded in addition and transmitted separately. Compared to lossy depth map coding schemes [9–11], GBR does not lead to geometric errors stemming from lossy compression, resulting in much higher synthesized view quality at the same bitrate [8].

When a pixel patch is displaced in the synthesized view according to disparity information encoded in GBR, the displacement vector is rounded to an integer position on the 2D image grid. That means an object with a width of N pixels in the original view that becomes width M pixels in the synthesized view, where $M > N$ (*e.g.*, if the object is slightly closer to the camera in the synthesized view), will be represented as multiple pixel patches in GBR, with missing pixels transmitted separately. See Fig. 1 as an illustration.

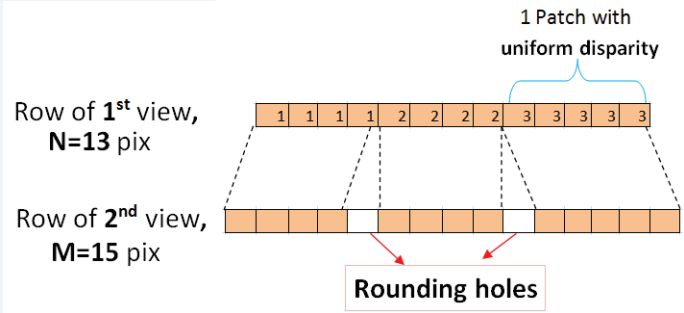


Fig. 1: Illustration of GBR, where patches of uniform disparities are mapped from the first to the second view, and new pixels are coded and transmitted separately.

However, this representation can be sub-optimal when the surface of the object is entirely visible in the original view with sufficient number of pixel samples, in which case one should

[1]For simplicity, we assume here that the captured 3D scene is *Lambertian*, meaning that a 3D voxel as observed from different viewpoints shows the same color intensity.

not transmit extra information beyond the N-pixel samples in the original view.

In this paper, we propose a new re-sampling & interpolation strategy at the decoder, so that the M pixels of the same object in the synthesized view can be constructed using only the available N pixels in the original view. In particular, we propose two possible interpolation procedures. The first method is local linear interpolation, where N original pixels are first rounded to N on-grid locations in the synthesized view, and the remaining $M-N$ on-grid pixels are linearly interpolated using neighboring mapped pixels. In the second method, N original pixels are mapped to N off-grid locations (no rounding) based on available disparity information, then the new M on-grid pixels are interpolated using a graph-signal smoothness prior [13, 14] and the N off-grid pixels as reference. The recommended interpolation method per patch is encoded as side information and transmitted with GBR disparity information. Experimental results show that our graph-based interpolation method can outperform the local linear interpolation method by up to 1.9dB in smooth patches, and enabling interpolation at the decoder can improve GBR compression rate by up to 29% at reasonable PSNR range.

The outline of the paper is as follows. We first discuss related works in Section II. We then overview our coding and view synthesis sytem in Section III. Our proposed pixel mapping strategy and interpolation method are discussed in Section IV and V respectively. Finally, we present results and conclusion in Section VI and VII respectively.

II. Related Work

Optimizing DCT-based compression of depth images for decoder-side DIBR view synthesis has been studied [10, 12, 15]. However, using fixed transforms like DCT can lead to blurring of edges in decoded depth images, resulting in undesirable bleeding artifacts during view synthesis that degrade visual quality. One alternative is to employ edge-preserving transforms [9, 11, 16, 17] and wavelets [18], but this requires coding of object contours [19, 20] as an additional overhead. In contrast, [8] proposed GBR to code minimally described disparity information losslessly for subsequent view(s) given transmitted first view as reference. It has been shown that at high bitrate, GBR outperforms competing disparity representations due to its compactness and lossless coding. We overview the operations in GBR next.

A. Overview of Graph-based Representation (GBR)

The aim of GBR is to first compactly represent disparity information and then losslessly code the chosen representation for accurate reconstruction of a given set of views. Specifically, a pixel row in an original view is divided into patches, each containing pixels with the same disparity value. Links are then drawn from patch boundaries in the original view to pixel locations in the synthesized view that delimit the new patch locations. As an example, in Fig. 2 there are three links that designate end locations of three patches in the synthesized

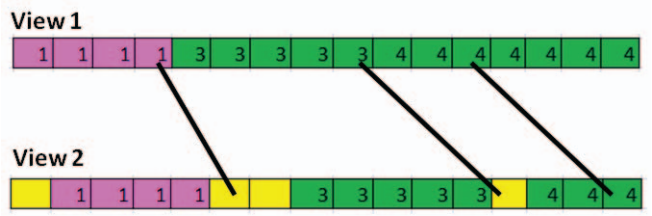


Fig. 2: Example of the connections of the GBR between two views (The same colored pixels belong to a same object, the numbers on the pixels correspond to the disparity

view. New pixels in the synthesized view (shown in yellow)—*appearing pixels* that come into view due to view change and *disocclusion pixels* that were not visible in the original view due to foreground object occlusion—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. For the decoder, only pixel displacement and decoding of new pixels need to be performed to reconstruct a synthesized view. In contrast, our proposal further enables the decoder to interpolate chosen new pixels, which will render the coding of some pixels unnecessary and lower the coding overhead.

B. Graph-signal Priors for Image Interpolation

Image interpolation is an example of inverse imaging problem and can be solved using a variety of techniques, such as bilateral filter [21], kernel regression [22], etc. With the recent advances in graph signal processing (GSP) [23], graph-signal priors have been developed for inverse imaging problems like denoising [13, 14, 24] and interpolation [25, 26]. Leveraging on these prior works, our proposed image interpolation scheme builds a unique graph connecting off-grid and on-grid pixels for signal reconstruction. See Section V for details.

III. System overview

Like [2] we assume an interactive multiview streaming scenario, where a client requests a chosen neighboring view $v \pm 1$ after observing view v, and in response the server transmits data needed for correct decoding of the requested view. The actual transmitted data is a version of GBR overviewed in Section II-A: a graph describing displacement of patches, plus coding of new pixels in the synthesized view. The key difference is that we distinguish between *disocclusion pixels* and *rounding pixels*: in our scheme only disocclusion pixels are coded and transmitted. Disocclusion pixels are pixels representing spatial areas that are not visible in the original view due to occlusion of foreground objects. In contrast, rounding pixels are surfaces of objects that are visible in the original view, but because the objects take up slightly more pixels in the synthesized view due to rounding, there are insufficient number of pixels for one-to-one mapping from the original view. Rounding pixels are thus interpolated using available pixels in the original view using one of two possible methods; the appropriate method per rounding hole based on

interpolation performance is signaled to the decoder using a single encoded flag.

We propose two different interpolation methods, both of which perform interpolation row-by-row. The first method called `linearInter` first maps available pixels in the original view to *on-grid* pixel locations in the synthesized view and interpolates missing on-grid pixels using neighboring mapped pixels. The second method called `graphInter` maps available pixels in the original view to *off-grid* pixel locations in the synthesized view via a linear disparity function, then interpolates all missing on-grid pixels via a graph-based formulation. We discuss them in order next.

IV. Formulation of the Problem

A. Mapping of reference pixels

To construct M on-grid pixels of the same object in the synthesized view using N pixel samples of the original view as reference, `linearInter` first maps N original pixels to on-grid integer positions of the synthesized view given disparity information provided by GBR. If $M > N$, then there exist $M - N$ extra pixels in the synthesized view that have no corresponding pixels in the original view and require interpolation. This is illustrated in Fig. 3, where the seven blue pixels in the original view 1 are mapped to the seven red on-grid pixels in view 2. There is an extra pixel between red pixels 3 and 5. `linearInter` computes an arithmetic mean of the surrounding mapped pixels (red pixels 3 and 5 in the example) as the reconstructed value.

In contrast, `graphInter` interpolates all M new on-grid pixels in the synthesized view using N pixel samples in the original view. There are two steps. In the first step, using the disparity information provided by GBR we first compute a *linear disparity function* to map N pixel samples in the original view to N off-grid locations in the synthesized view. Fig. 3 shows the seven off-grid pixel samples in the synthesized view 2 in light red. In the second step, we estimate a signal of length $N+M$ with M on-grid pixels (in dark red) and N off-grid (in light red) samples mapped from the original view using a graph-signal smoothness prior. We discuss these two steps in order.

B. Computing the linear disparity function

To map N pixel samples in the original view to off-grid locations in the synthesized view, we compute a linear disparity function $f(x)$ via linear regression: given per-pixel integer disparity information provided by GBR, we minimize the sum of squared difference between mapped integer locations y_i and a fitted line $f(x) = \alpha x + \beta$:

$$(\alpha^*, \beta^*) = \arg\min_{\alpha,\beta} \sum_{i}^{N} (y_i - (\alpha i + \beta))^2 \qquad (1)$$

Using computed (α^*, β^*), we can then map the N pixel samples in the original view to off-grid positions in the synthesized view via $f(x)$.

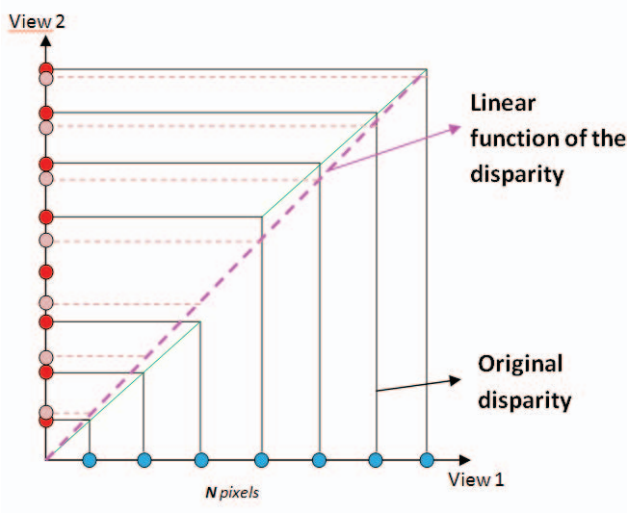


Fig. 3: Position of known and unknown pixels in both views according to the depth map

V. Interpolation Algorithm

We now describe how the $N + M$ sample signal is reconstructed in the synthesized view. Note that to have a better interpolation quality, we split the texture signals in block segments of 16 pixels, which are optimized independently and overlapped by two pixels on each side as shown in Fig. 4 to avoid blocking artifacts. We first describe the construction of an appropriate graph for the length $N + M$ signal. We then formulate an optimization problem to compute the desired signal efficiently.

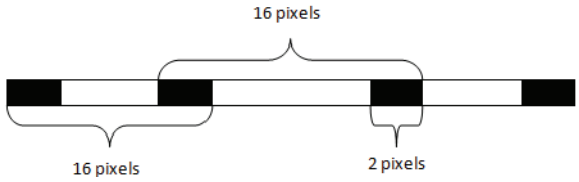


Fig. 4: overlapping blocks of rows. The dark areas correspond to the overlapped pixels

A. Graph construction

We construct a line graph with $N + M$ nodes, each node represents a pixel sample in the length $N + M$ signal. In the first iteration, we first connect neighboring pixels in the graph with edges, and then connect neighboring off-grid pixels in the graph with edges. See Fig. 5 for an illustration. The reason is the following: while the N off-grid samples are neighboring on-grid pixels in the original view with known pixel intensities, the on-grid pixels in the synthesized view do not have estimated intensities yet. We thus compute edge weights differently. For weight $w_{i,j}$ of an edge connecting two off-grid pixels, we use:

$$w_{i,j} = \exp\left(-\frac{\|I_i - I_j\|^2}{\sigma_I^2}\right) \qquad (2)$$

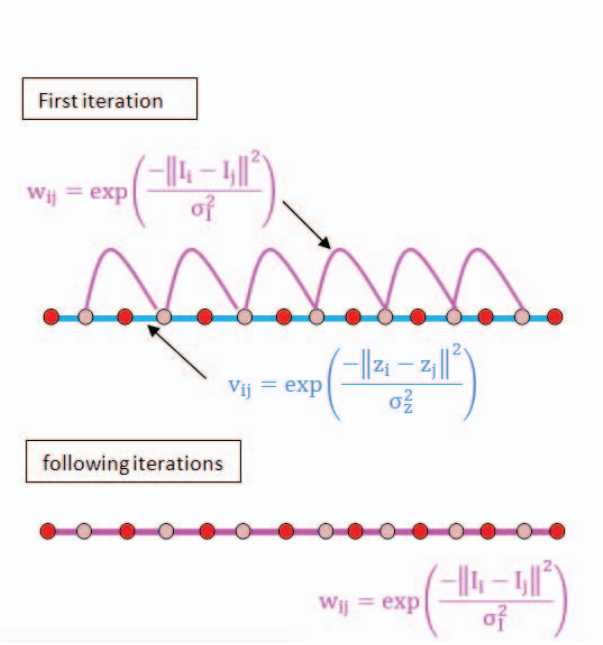


Fig. 5: Graph construction example of one row, where the red pixels are the M unknown pixels and the light red pixels are the N known pixels

where I_i is the intensity of pixel i and σ_I is a scaling parameter, which is tuned such that the values in the weighted matrix are of the same order.

For weight $w_{i,j}$ of an edge connecting off-grid and on-grid pixels, we use:

$$v_{ij} = \exp\left(-\frac{\|z_i - z_j\|^2}{\sigma_z^2}\right) \quad (3)$$

where z_i is the position of pixel i and σ_z is another scaling parameter, tuned like σ_I.

For subsequent iterations, because the on-grid pixels now have intensity estimates, we can simply use (2) to compute weights of edges only connecting neighboring pixels in the graph.

B. Defining the problem objective

Given the defined edge weights, we can write an adjacency matrix $\mathbf{A}$ where $a_{i,j} = w_{i,j}$. The degree matrix $\mathbf{D}$ is a diagonal matrix with diagonal entries $d_{i,j} = \sum_j A_{i,j}$. The *graph Laplacian* $\mathbf{L}$ is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$ [23]. If we now define the desired $N + M$ sample signal as a vector $\mathbf{x}$, then the graph-signal smoothness prior [13, 14] can be written as $\mathbf{x}^T\mathbf{L}\mathbf{x}$. Unlike signal-independent smoothness priors like total variation (TV) [27], it has been demonstrated [28, 29] that graph-signal smoothness prior does not over-smooth and can preserve edge structures in images well.

Second, we need a fidelity term so that the computed length $N + M$ signal is consistent with the N-pixel observation $\mathbf{y}$ in the original view. Let $\mathbf{H}$ be a matrix that picks out the N off-grid samples in the signal $\mathbf{x}$. The fidelity term will thus be: $\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2$.

Third, we require that our solution $\mathbf{x}$ to be consistent in overlapped pixels with the previously computed solution $\mathbf{x}_L$ to the left. Enforcement of consistency in overlapped regions helps eliminate undesirable blocking artifacts stemming from separately optimized patch solutions, as demonstrated in [30]. Let $\mathbf{B}_L$ and $\mathbf{B}_R$ be matrices that pick out overlapping pixels at the right and left boundaries, respectively. To ensure that the overlapped segment is consistent, we can add a term $\|\mathbf{B}_L\mathbf{x} - \mathbf{B}_R\mathbf{x}_L\|_2^2$.

We finally arrive at the problem objective, which can now be written as:

$$\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \mu\mathbf{x}^T\mathbf{L}\mathbf{x} + \delta\|\mathbf{B}_L\mathbf{x} - \mathbf{B}_R\mathbf{x}_L\|_2^2 \quad (4)$$

where μ and δ are parameters to trade off among different quantities. The problem in (4) is an unconstrained quadratic optimization problem with a closed form solution. (4) is then solved iteratively, each time the Laplacian $\mathbf{L}$ is updated with new weights using the most recently computed solution $\mathbf{x}$. The algorithm terminates after 3 or 4 iterations, depending on the resulted quality.

VI. Experiments

We now demonstrate the performance of our proposed scheme through a series of experiments. We first show that our proposed graph-based interpolation scheme `graphInter` can in some cases out-perform the default scheme `linearInter` noticeably. We then show that when employing our improved GBR coding scheme where transmission of rounding pixels is avoided at encoder (which are subsequently interpolated at the decoder), the bitrate reduction over original GBR is significant.

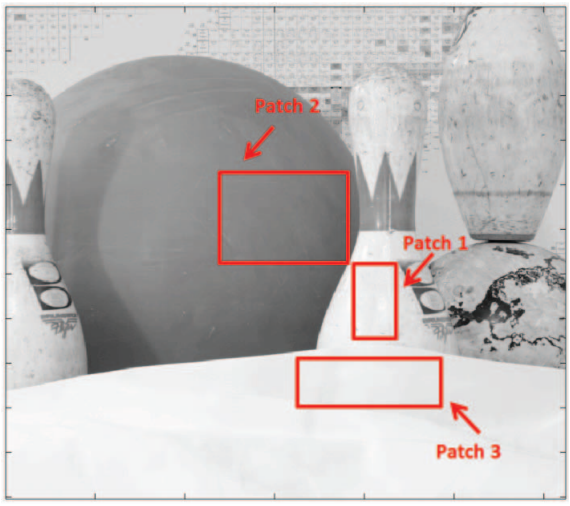
To evaluate the quality of interpolation using our two methods, we perform experiments on four image sequences: `Bowling`, `Laundry`, `Plastic`, `Baby` from the Middlebury database[2]. Given an original image and its corresponding disparity information provided by GBR, we synthesize a neighboring view row-by-row. The rounding holes (the difference in disparity values of surrounding pixel patches is less than a threshold value of 4) are interpolated at the decoder using one of our two proposed methods.

We observe that depending on the particular patch of a given image, `linearInter` or `graphInter` may have more superior interpolation performance. For illustration, for each image we select particular patch examples, shown in Fig. 6, 7, 8 and 9. First, we show the original view indicating the locations of the patches. Then, for each patch, the ground truth is shown, followed by the patch interpolated using `linearInter` and the one interpolated using `graphInter`. For objective measure we include a table showing the PSNR for each of the patches, where the better PSNR value of the two methods is colored.

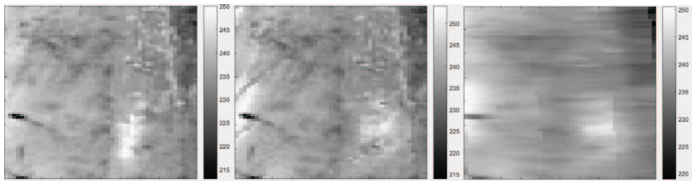
From Tables I, II, III, IV and the interpolated patches, we can see that `graphInter` is better in case of smooth surfaces with little texture content, which is the signal type

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

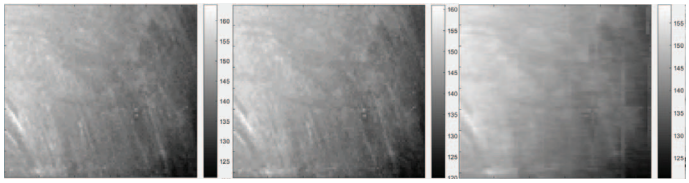
that graphInter can perform well. On the other hand, linearInter is better when the object surface has fast-varying textural content (*e.g.* Fig. 9 and Table IV).



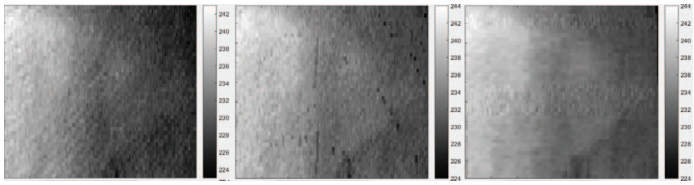
(a) Original 1st view of Bowling



(b) orig. patch (c) linear inter. (d) graph inter.



(e) orig. patch (f) linear inter. (g) graph inter.



(h) orig. patch (i) linear inter. (j) graph inter.

Fig. 6: Resulting patches after linear or graph-based interpolation

Finally, to compare the RD performance of original GBR and our proposed scheme where rounding holes are not coded at the encoder but interpolated at the decoder, we implemented a compression scheme for hole pixel coding using graph Fourier transform (GFT), similarly done in [16]. We vary the quantization parameter (QP) for coding of the hole pixels to induce different tradeoff points. In Fig. 10 we see the PSNR versus rate curves using our proposed scheme compared to original GBR for Laundry and Plastic. We observe a large reduction in bitrate due to saving from not transmitting rounding holes explicitly—up to a 60% reduction in bitrate for the same PSNR. Fig. 11 shows similar curves for Bowling and Baby. In this case, we observe similar large reduction in bitrate, but because the interpolation quality is relatively poorer, at high-bitrate our proposal is not competitive. This means that at high-bitrate, the encoder should encode selected rounding hole pixels to improve quality at the expense of larger bitrate. This is left for future work.

TABLE I: Comparison of PSNR between linearInter and graphInter for the patches in the image Bowling

test sequence	linearInter	graphInter
patch 1	37.84dB	38.41dB
patch 2	42.6dB	41.1dB
patch 3	39.7dB	39.6dB

TABLE II: Comparison of PSNR between linearInter and graphInter for the patches in the image Laundry

test sequence	linearInter	graphInter
patch 1	33.2dB	33dB
patch 2	32.2dB	32.9dB
patch 3	39.7dB	39.4dB

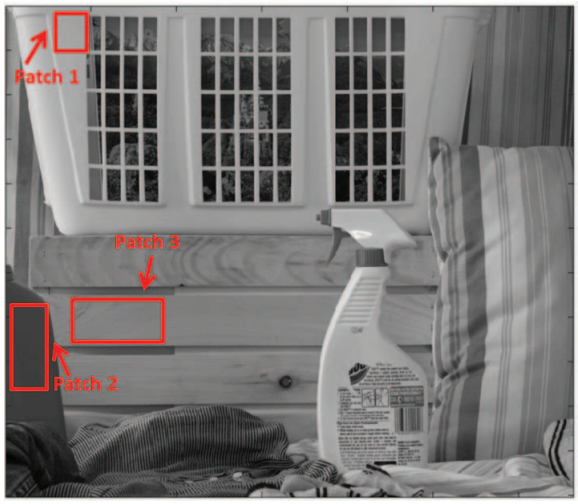
VII. Conclusion

To enable efficient transmission of neighboring viewpoint images during an interactive multiview streaming session, previous graph-based representation (GBR) losslessly codes disparity information of pixel patches, plus lossy coding of new pixels in the virtual view in addition. In this paper, we propose to improve GBR by enabling the decoder to re-sample and interpolate M pixels of an object given N available pixel samples of the same object in the original view. This means that the $M - N$ rounding pixels due the slight change of object size from original to virtual view do not need to be encoded, lowering the transmission cost. We propose two interpolation schemes: i) map N available pixels from the original view to on-grid pixel locations in the virtual view and linear interpolate $M - N$ new pixels locally; and ii) construct an $N + M$ sample signal in the virtual view using a graph-signal smoothness prior. The chosen method per rounding hole is signalled to the decoder as coded side information. Experimental results first show that either linear or graph-based interpolation can in some cases improve image quality, and second show that bitrate reduction by not transmitting coded pixels in the rounding holes can be significant.

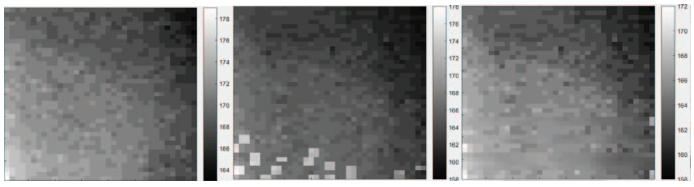
References

[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.

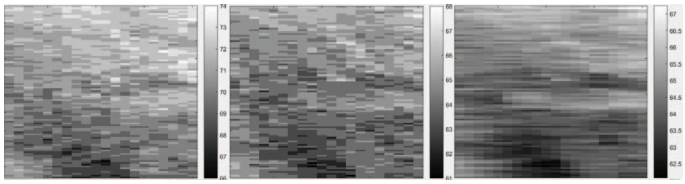
[2] G. Cheung, A. Ortega, and N.-M. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," in *IEEE Transactions on Image Processing*, March 2011, vol. 20, no.3, pp. 744–761.



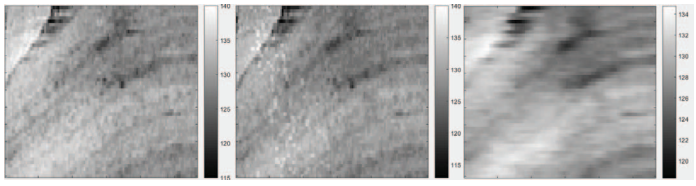
(a) Original 1st view of `Laundry`



(b) orig. patch (c) linear inter. (d) graph inter.

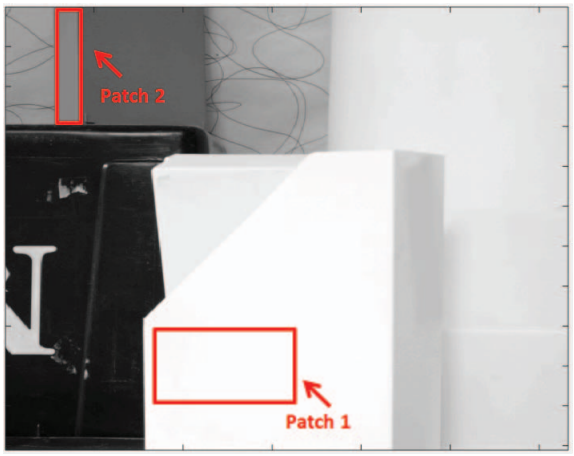


(e) orig. patch (f) linear inter. (g) graph inter.

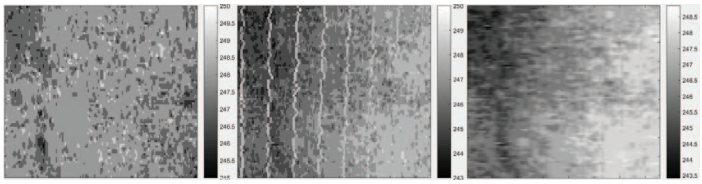


(h) orig. patch (i) linear inter. (j) graph inter.

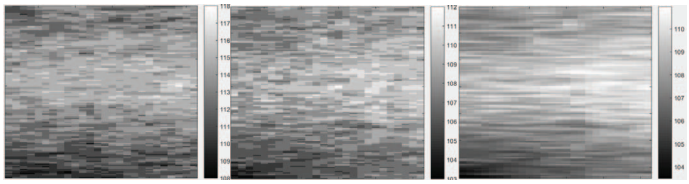
Fig. 7: Resulting patches after linear or graph-based interpolation



(a) Original 1st view of `Plastic`



(b) orig. patch (c) linear inter. (d) graph inter.



(e) orig. patch (f) linear inter. (g) graph inter.

Fig. 8: Resulting patches after linear or graph-based interpolation

TABLE III: Comparison of PSNR between `linearInter` and `graphInter` for the patches in the image `Plastic`

test sequence	`linearInter`	`graphInter`
patch 1	44dB	44.1dB
patch 2	33.8dB	34dB

[3] 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.

[4] T. Maugey, I. Daribo, G. Cheung, and P. Frossard, "Navigation domain partitioning for interactive multiview imaging," in *Special Issue on 3D Video Representation, Compression, Rendering, IEEE Transactions on Image Procesing*, September 2013, vol. 22, no.9, pp. 3459–3472.

[5] A. De Abreu, P. Frossard, and F. Pereira, "Optimizing multiview video plus depth prediction structures for interactive multiview video streaming," in *IEEE Journal of Selected Topics in Signal Processing*, April 2015, vol. 9, no.3, pp. 487–500.

[6] L. Toni, G. Cheung, and P. Frossard, "In-network view re-sampling for interactive free viewpoint video streaming," in *IEEE International Conference on Image Processing*, Quebec City, Canada, September 2015.

[7] 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.

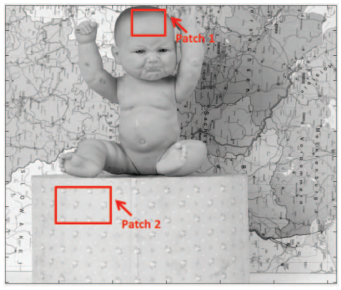
[8] T. Maugey, A. Ortega, and P. Frossard, "Graph-based representation for multiview image geometry," in *IEEE Transactions on Image Processing*, May 2015, vol. 24, no.5, pp. 1573–1586.

[9] 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.

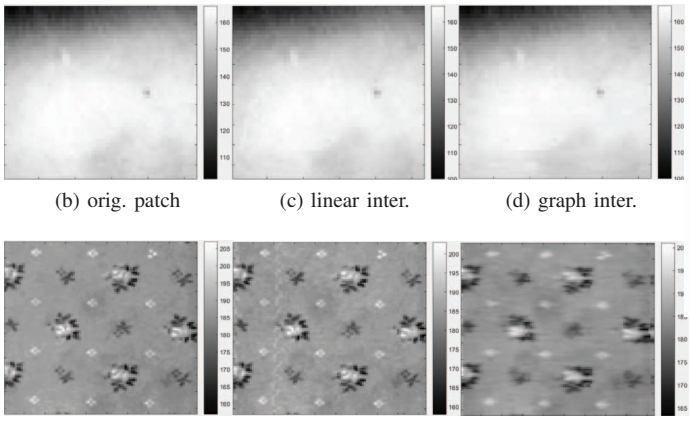
[10] G. Cheung, J. Ishida, A. Kubota, and A. Ortega, "Transform domain sparsification of depth maps using iterative quadratic programming," in *IEEE International Conference on Image Processing*, Brussels, Belgium, September 2011.

[11] 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.

[12] W.-S. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila, "Depth map



(a) Original 1st view of Baby



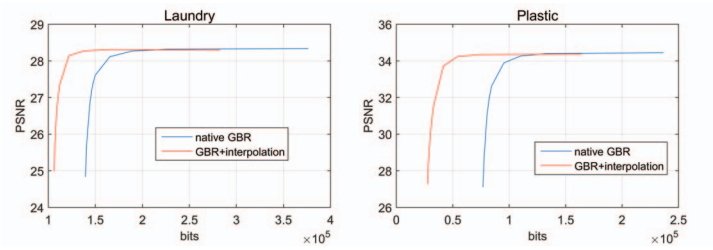
(b) orig. patch (c) linear inter. (d) graph inter.

(e) orig. patch (f) linear inter. (g) graph inter.

Fig. 9: Resulting patches after linear or graph-based interpolation

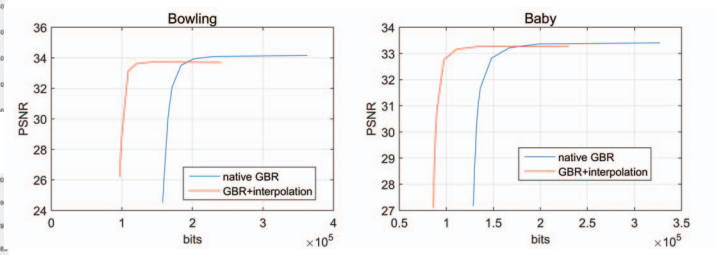
TABLE IV: Comparison of PSNR between linearInter and graphInter for the patches in the image Baby

test sequence	linearInter	graphInter
patch 1	47.3dB	47.7dB
patch 2	41.7dB	37.6dB



(a) Laundry (b) Plastic

Fig. 10: PSNR versus rate curves for native GBR and proposed scheme for Laundry and Plastic.



(a) Bowling (b) Baby

Fig. 11: PSNR versus rate curves for native GBR and proposed scheme for Bowling and Baby.

coding with distortion estimation of rendered view," in *SPIE Visual Information Processing and Communication*, San Jose, CA, January 2010.

[13] J. Pang, G. Cheung, W. Hu, and O. C. Au, "Redefining self-similarity in natural images for denoising using graph signal gradient," in *APSIPA ASC*, Siem Reap, Cambodia, December 2014.

[14] 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.

[15] 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.

[16] W. Hu, G. Cheung, X. Li, and O. Au, "Depth map compression using multi-resolution graph-based transform for depth-image-based rendering," in *IEEE International Conference on Image Processing*, Orlando, FL, September 2012.

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

[18] 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.

[19] I. Daribo, G. Cheung, and D. Florencio, "Arithmetic edge coding for arbitrarily shaped sub-block motion prediction in depth video coding," in *IEEE International Conference on Image Processing*, Orlando, FL, September 2012.

[20] 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.

[21] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proceedings of the IEEE International Conference on Computer Vision*, Bombay, India, 1998.

[22] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," in *IEEE Transactions on Image Processing*, February 2007, vol. 16, no.3, pp. 349–366.

[23] 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.

[24] W. Hu, X. Li, G. Cheung, and O. Au, "Depth map denoising using graph-based transform and group sparsity," in *IEEE International Workshop on Multimedia Signal Processing*, Pula, Italy, October 2013.

[25] Y. Mao, G. Cheung, A. Ortega, and Y. Ji, "Expansion hole filling in depth-image-based rendering using graph-based interpolation," in *IEEE International Conference on Acousitics, Speech and Signal Processing*, Vancouver, Canada, May 2013.

[26] Y. Mao, G. Cheung, and Y. Ji, "Image interpolation during DIBR view synthesis using graph Fourier transform," in *3DTV-Conference*, Budapest, Hungary, July 2014.

[27] Leonid I Rudin, Stanley Osher, and Emad Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992.

[28] W. Hu, G. Cheung, X. Li, and O. Au, "Graph-based joint denoising and super-resolution of generalized piecewise smooth images," in *IEEE International Conference on Image Processing*, Paris, France, October 2014.

[29] P. Wan, G. Cheung, D. Florencio, C. Zhang, and O. Au, "Image bit-depth enhancement via maximum-a-posteriori estimation of graph AC

component," in *IEEE International Workshop on Image Processing*, Paris, France, October 2014.

[30] X. Liu, G. Cheung, X. Wu, and D. Zhao, "Inter-block soft decoding of JPEG images with sparsity and graph-signal smoothness priors," in *IEEE International Conference on Image Processing*, Quebec City, Canada, September 2015.