LUMINANCE CODING IN GRAPH-BASED REPRESENTATION OF MULTIVIEW IMAGES

Thomas Maugey[1], *Yung-Hsuan Chao*[2], *Akshay Gadde*[2], *Antonio Ortega*[2], *Pascal Frossard*[1]

[1]Ecole Polytechnique Fédérale de Lausanne (EPFL) [2]University of Southern California

ABSTRACT

Multi-view video transmission poses great challenges because of its data size and dimension. Therefore, how to design efficient 3D scene representations and coding (of luminance and geometry) has become a critical research topic. Recently, the graph-based representation (GBR) is introduced, which provides a lossless compression of multi-view geometry by connecting informative pixels among views. This representation has been shown as a promising alternative to the classical depth-based representation, where the view synthesis accuracy is hard to control. In this work, we study the luminance compression under GBR, which is not well considered in existing literature. With a proper structural reformulation, we show that the graph-based transform can be applied on the GBR paradigm, hence better extracting the correlation among pixels along graph connections. Moreover, we extend the popular SPIHT coding scheme to further improve coding efficiency. The experimental results show that our method leads to better RD coding performance as compared the classical luminance coding algorithms.

Index Terms— Multiview image coding, 3D representation, graph-based representation, graph-based transform

1. INTRODUCTION

Multiview video transmission has become a very popular research field in recent years. It is at the core of many novel applications in cinema, live event transmission, augmented reality, etc. At the same time, it poses interesting research challenges (compression, transmission, etc.) due to the increase in data dimension and size. In particular, classical image-based representation models have been recently challenged by new representation methods with lower redundancy and better compression performance. More formally, after the acquisition process, the 3D scene is described with multiview color images and, in some scenarios, depth maps. The traditional depth-based representation method [1] consists of directly representing the multiview set using these color and depth images for each view. At the coding step, the luminance signal is compressed using depth maps to remove redundancies with a depth image-based rendering technique [2]. However, the depth-based approach becomes limited when the geometry signal has to be compressed, since the effect of the lossy coding of depth is not well controlled [3, 4]. Moreover, this representation has a lot of redundancies which lead to some cost in coding performance. Coding depth directly is thus not very effective, and one could rather re-arrange data, and design different representations.

In [5], we have proposed an alternative representation based on a graph structure, called *graph-based representation (GBR)*. Instead of using depth maps, our approach describes the geometry information with connections between pixels. These connections are drawn between pixels of a view and their visible neighbors in the next view. In Fig. 1, we show the GBR format for an illustrative scene made of one foreground (*i.e.*, pixels a to d) and one background (pixels 1 to 12). The reference view (view 1) is transmitted entirely. However, only the novel pixels of view 2, *i.e.*, pixels 7 and 11, are described in the GBR. The geometry signal is only composed of two links between pixels 7 and 8, and between pixels 11 and 12. This information is sufficient to reconstruct the two views exactly at the decoder. We have shown in [6] that this method leads to a better control of distortion in the case of lossy geometry compression, compared to depth-based image representation. In other words, we have shown the potential of the GBR approach for describing the geometry of a multiview image set. However, the coding of the texture information has not been optimized in our previous work, which does not consider the specific properties of graph-based representations. In particular, the links between pixels of different views have not been well exploited to compress the luminance information. Coding of luminance thus has to be adapted to the new data structure in order to eventually build a complete multiview compression scheme. This is our objective in this paper.

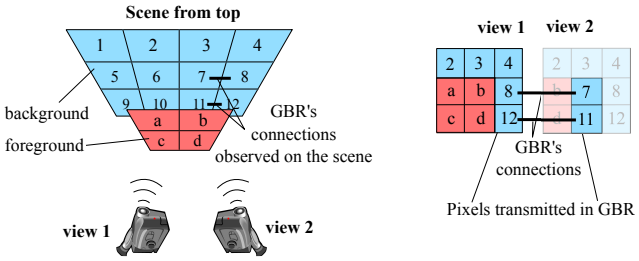


Fig. 1. Illustration of GBR concepts for a simple scene. View 1 is the reference image, transmitted entirely. In view 2, only the disoccluded pixels 7 and 11 are novel with respect to view 1. They are thus represented in the GBR and linked to pixels 8 and 12 of view 1.

We propose a coding method that relies on the properties of the links between pixels. In Fig. 1, we represent the GBR connections in the 3D scene. We see that these connections not only give a depth or disparity information, as we discussed earlier, they also provide a neighborhood information. Indeed, although pixels 7 and 8 appear in two different views, they represent neighboring points in the 3D scene. That means that the corresponding values are likely to be correlated, which should be exploited in the coding scheme. In order to do that, we extend the $GraphBior$ filters proposed in [7, 8, 9] to our GBR data format, so that a transform is applied along the graph connections. We thus code together pixels in different views that are neighbors in the 3D scene. One of the strengths of the graph-based transform is the possibility of choosing different weights depending on the importance of the pixel correlation. In the example of Fig. 1, we observe that pixels b and 8 are nearby in the image domain (in view 1), but are not neighbors in the 3D scene. We thus decrease the weight between them to a low value (*e.g.*, 0.1), in the graph in order

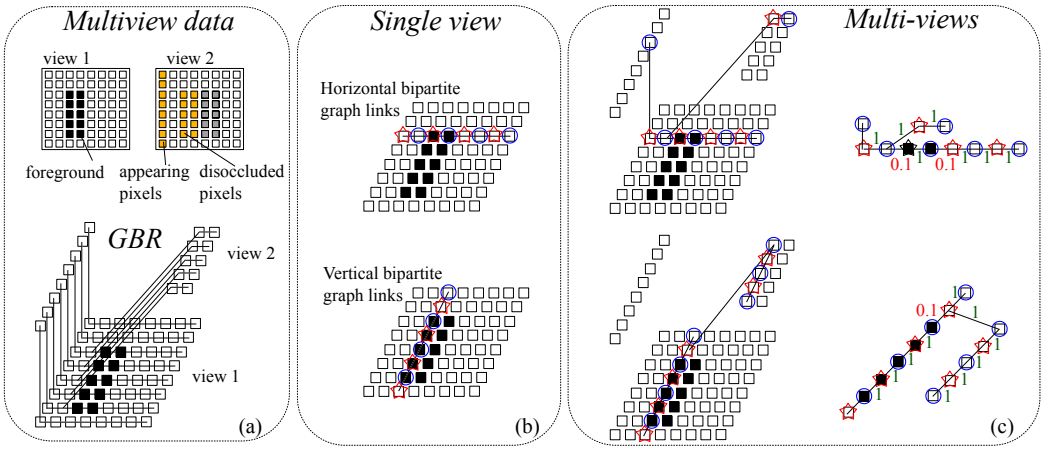


Fig. 2. Bipartite graph construction in GBR used in $GraphBior$ filterbanks. (a) Example of multi-view data and the links between the 1_{th} reference view and the adjacent "new" pixels in the 2_{nd} view. (b) The bipartite graph construction in a single image and (c) the bipartite graph construction in GBR for multi-view system. The blue and red pixels represent different node sets in the bipartite graph (best viewed in color)

to signal that these two pixels do not belong to the same object in the scene, hence they are likely to have different color values. This relies on the idea that the GBT exploits correlation given by graph weights, and tends to provide effective compaction for signals that are smooth on the graph.

In outline, the proposed complete coding schemes has the following steps. We first build a GBR description of a multiview set. Then, we apply the GBT as described above. The transformed coefficient are then coded using an extension of the SPIHT algorithm [10]. In the framework adopted for this paper, we are interested in the joint coding of $N > 1$ views. Since we already study the geometry rate in [5], we only consider here the luminance rate. We run experiments on different test sequences, and we show that our method obtain better rate-distortion performance than baseline techniques that do not exploit the connections as neighborhood information.

2. GRAPH-BASED REPRESENTATION

We recall here the main ideas of the GBR construction process, as well as the view reconstruction at the decoder. Readers are referred to [5] for details. Let us consider a scene captured by N cameras with the same resolution and focal length f. The image captured by the n^{th} camera is denoted by I_n, with $1 \leq n \leq N$, where $I_n(r, c)$ is the pixel at row r and column c in I_n. We only consider translation between cameras, and we assume that the views are rectified. In other words, the geometrical correlation between the views $\{I_n\}$ is only horizontal. We further assume that an accurate depth image, Z_n, is available at the encoder for every viewpoint, I_n. We then compute $N - 1$ dense disparity maps from these depth images. In what follows we assume that the set of images contains one *reference* view (typically the first, left-most, image) and $N-1$ *predicted* views.

We categorize the different types of pixels depending on how they change from one view to another. Due to camera translation, a new part of the scene appears on the right or left of the image (*appearing* pixels) and another part disappears (*disappearing* pixels). As we move from one camera to a next, foreground objects move faster than the background. As a result, some background pixels may appear behind objects (*disoccluded* pixels). Conversely, some background pixels may become hidden by a foreground object (*occluded* pixels). If we consider a pair of views (reference and target), a row of the target view can be reconstructed by copying pixels from the corresponding row of the reference view, except when the above mentioned types of pixels occur (in which case "new" pixels have to be inserted). Our graph approach directly conveys this information by transmitting either i) a link to *the location in the reference row* where pixels should be copied from, or ii) *the values of new pixels* to be inserted.

A graph with N layers describes 1 reference view and $N - 1$ predicted views. Its construction uses the information provided in the depth maps Z_n, $1 \leq n \leq N - 1$. The constructed graph is made of two components, which are described by two matrices of size $N \times W$, where N is the number of layers (*i.e.,* the number of views encoded by the graph) and W is the image width. These two matrices are the color values Γ_r and the connections Λ_r and represent color and geometry information for all pixels of all views, where r is the row index (a pair of matrices per row). The matrices Γ_r and Λ_r are generated based on the following principles. Pixel intensity values are stored in the layer (view) where they appear first. This means that a given layer only contains pixels that were not present in a lower layer. Then, the connections simply link these "new" pixels to the position of their neighbor in the previous layer. At the decoder side, the reconstruction involves traversing the graph (left to right) and copying pixel values from either layer 1 or layer 2 to the output, following the links in the graph.

3. LUMINANCE CODING USING GRAPH

3.1. Background

Although GBR provides a more compact and accurate representation for 3D geometry, our previously proposed encoding scheme for luminance values [5] does not fully take advantage of such a structure. That is, the cross-view links in GBR reflect the spatial adjacency in 3D geometry; the connected pixels are thus likely to have similar luminance, which is extremely suitable for wavelet coding. However, in [5] the wavelet coding is only performed within the reference image; the cross-view similarity is coded by differential coding. This scheme ignores the potential redundancy (as shown in Sec. 4). Besides, the compression quality for predicted images relies on the reconstruction of the reference image, which might results in error-propagation.

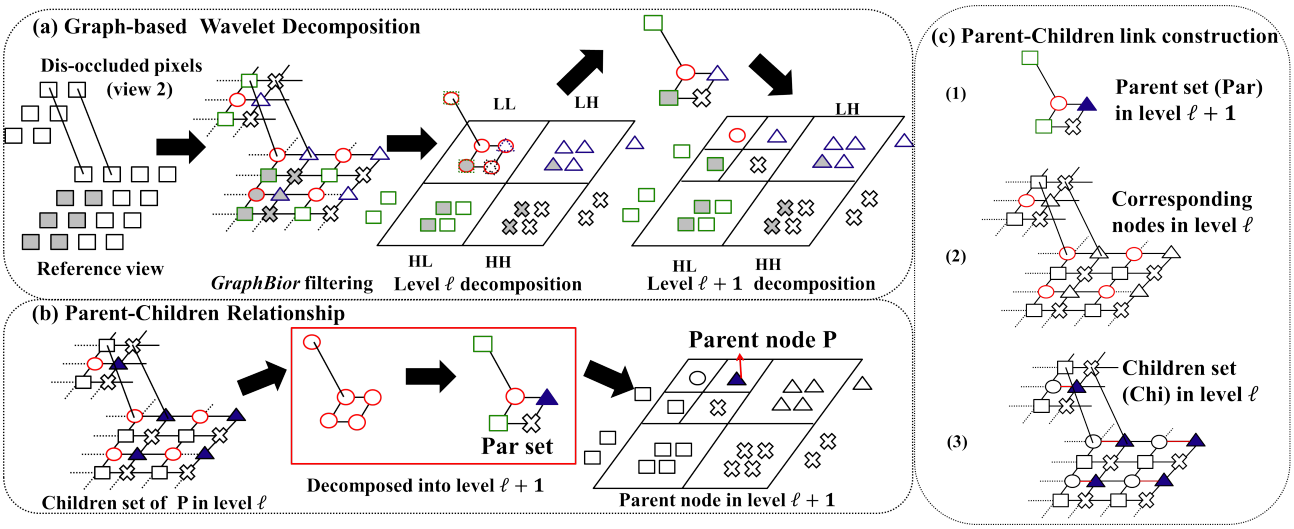


Fig. 3. (a) Wavelet decomposition on graphs in $GraphBior$, where shape {circle, triangle, square, and cross} denote coefficients in LL, LH, HL, HH subbands. (b) Parent-children relationship: P node in LH band of level $\ell + 1$ has five children from two views in level ℓ marked with blue. (c)The procedure of finding the children node in level ℓ for the parent node in level $\ell + 1$ (best viewed in color)

In order to better exploit the GBR structure in luminance compression, we adapt the graph-based wavelet filterbanks (denoted as *GraphBior*) [8, 9], which can be applied to any signal that can be represented with bipartite graphs. As shown next, the GBR structure can be decomposed into a series of bipartite graphs without any additional overhead; Therefore, *GraphBior* can be applied directly. Note that *GraphBior* has the flexibility to adapt to the local variation within an image by adjusting the link weights. Such property can provide potential R-D improvement, which will be shown later in the experiment part.

3.2. Graph-based transform with GBR

Before describing the bipartite graph reformulation in the multi-view system, let's first review the simpler single-view case. As shown in Fig.2(b), an image graph with 4-connected pixels can be represented separately by the horizontal and vertical line graphs, which are bipartite. This reformulation is used in [9] for image compression with separable *GraphBior* filterbanks. We extend the approach to the multi-view system encoded by GBR, as demonstrated in Fig. 2(c). Similar to the single view case, pixels are again connected horizontally and vertically into two subgraphs. However, now the rows and columns are of tree graphs (which by definition, are also bipartite) with branches connecting disoccluded pixels between views.

As suggested in [9], the link weights on the graph can be adjusted to better describe the local variation of signal. In this paper, we test two different weight selection methods: One is the non-weighted graph ($nGBT$), where all the links have unit weight ; the other is the weighted graph ($wGBT$), where smaller weight (0.1 in our test) are assigned to links connecting foreground to background (Fig.2(c) right hand side), while all other edges have unit weight.

Once the subgraphs are constructed, *GraphBior* can be applied on the two bipartite graphs sequentially: Each node set in one bipartite graph stores output from either high pass (H) or low pass filters (L). When jointly considering the two graphs, the output coefficients are divided into {LL, LH, HL, HH} sets with different frequency and spatial orientation, similar to the standard wavelet transform.

3.3. Extension of SPIHT algorithm

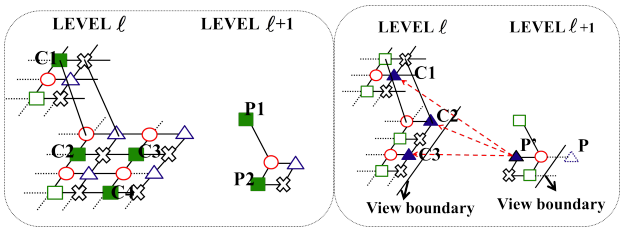
After deriving the wavelet coefficients with *GraphBior*, the next step is encoding. In this paper, we adopt the concept of Set Partitioning In Hierarchical Trees (SPIHT) algorithm. In SPIHT, each coefficient (Parent) is linked to the ones (Children) in the finer level corresponding to the same spatial region via a tree structure, called *spatial orientation tree*. SPIHT then exploits the spatial self-similarity across sub-bands in different levels to prune the tree. That is, if a coefficient in the coarse level has small magnitude, its descendants are likely to have low energy, and thus often may not need to be coded.

However, due to the branches connecting multi-views in our graph structure, the parent-children link cannot be easily constructed as in the image-based case using SPIHT. To be specific, we show an example of wavelet decomposition on graphs of multi-view system in Fig.3(a) and an example of parent-children relationship in Fig.3(b). As illustrated, the parent node P (blue triangle in Fig.3(b) has children in level ℓ from multiple views, making it difficult to construct the *spatial orientation tree* simply by SPIHT. We therefore propose an algorithm, called *GraphSPIHT*, which is able to build the tree across views, through the graph links in different wavelet levels.

Similar to the conventional SPIHT, the tree structure is built within coefficients with the same orientation. As the example in Fig.3(b), the children (in level ℓ) of parent P (in level $\ell + 1$) are exactly the neighbors of a set of LL nodes (red circles). More importantly, these LL nodes, after further decomposed into level $\ell+1$ (denoted as Par set), will contain the parent node P. Therefore, we can use this relationship to determine the children. The parent-children construction in $GraphSPIHT$ is summarized as follows (also illustrated in Fig. 3(c): (1) Find the subgraph (Par) for the considered parent P in level $\ell + 1$. This can be done by searching P's nearest neighbors with different orientations. (2) Find the LL nodes in level ℓ corresponding to Par set. (3) The children of P is the neighbors of the LL set with the same orientation as P. To be noticed, although we only show the case of parent in LH band, this procedure can also be used for parent nodes in HL and HH subband as well.

Based on the parent-children construction defined as above, *GraphSPIHT* then scans and encodes the coefficients similar to SPIHT. That is, if a coefficient or its descendants take large magnitude, the children will be put into the encoding queue for later scanning. However, due to the branches in the graph structure, one coefficient might be shared by multiple parent nodes (*MultiParents*). As shown in Fig. 4(a). If the process in Fig3(c) is applied on $P1$ and $P2$, then $C1$, $C2$, $C3$, and $C4$ will be identified as children for both of them. The situation will lead to repeated coding for those children nodes since they will be put into the coding queue twice. To solve this problem, a coefficient will be assigned to the parent node on the same view once *MultiParents* occurs.

Besides, in some situation, a node may not have any parent pointing to it (*nParent*); thus the coefficient is left out during encoding. This situation mostly occurs around view boundary, as shown in Fig. 4(b), where $C1$, $C2$, and $C3$ have no parent node found in level $\ell + 1$ (P is out of the boundary). For this case, we will assign parent nodes to be the nearest neighbor (P') with the same orientation.



(a) (b)

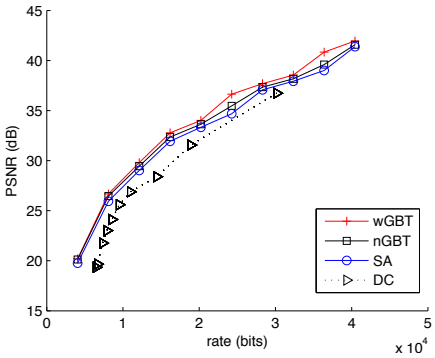
Fig. 4. (a) *MultiParents* and (b) *nParent* cases in *GraphSPIHT*. The red dashed lines in (b) denote the new parent assignment (best viewed in color)

4. EXPERIMENTS

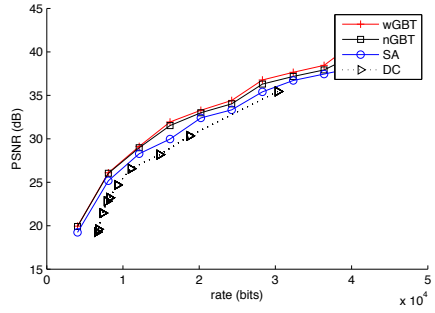
We evaluate the performance of our proposed methods: $wGBT$ and $nGBT$ along with *GraphSPIHT* encoding on the two artificial datasets each with two camera views. As shown in Fig. 5, the datasets consists of four square, and four diagonal shaped foreground objects in the 3D scene. The rate-distortion results are compared with two baseline methods: The first baseline is the differential coding (DC) approach [6] as mentioned in Sec.3.1. The second method is the shape-adaptive (SA) image coding proposed in [11]. The paper provides different choices of filterbanks, boundary extension, and coding scheme. For fair comparison, we use $CDF9/7$ filterbanks and asymmetric boundary extension in both SA-DWT and $GraphBior$ methods, and SPIHT coding in SA-DWT. We apply SA-DWT on the reference view and each disoccluded region independently. All the coefficients generated are coded together with SPIHT, where *spatial orientation trees* are built with coefficients in the same region.

From the obtained rate-distortion results in Fig. 6, we can see that our proposed method outperforms the baseline methods in both datasets. The results can be interpreted as follows: On the one hand, considering cross-view redundancies, our methods lead to better compression compared to shape adaptive method, where each region is coded separately. For non-square disoccluded regions, the improvement is increased due to the *GraphSPIHT*, which extends the *spatial orientation tree* to the cross-view neighbors, instead of relating coefficient to redundant node out of object. On the other hand, with a unified multi-view encoding with the graph-based method, we achieve better compression than DC, which exploits

Fig. 5. Reference view of "four squares", "four diagonals" images.



(a)



(b)

Fig. 6. Rate-distortion evaluation for (a) "four squares" and (b) "four diagonals" test images (190×190, 2 views). Only the luminance rate is considered.

cross-view similarity in two stages. Finally, we see that weighted graph $wGBT$ slightly outperforms $nGBT$ since it reduces the filtering effect across different objects, resulting in less high frequency energy shown in the wavelet domain.

5. CONCLUSION

In this paper, we propose a new coding strategy to compress color signal in the graph-based representation. We use a graph-based transform that is able to extract the correlation between pixels across views. Experiments show that our method outperforms the previous color coding tools involved in the initial graph-based representation. Second, we show that it leads to efficient RD performance, with significant improvement over classical luminance coding algorithms.

6. REFERENCES

[1] K. Müller, P. Merkle, and T. Wiegand, "3D video representation using depth maps," *Proc. IEEE*, vol. 99, no. 4, pp. 643–656, Apr. 2011.

[2] D. Tian, P. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3D videos," *Proc. of SPIE, the Int. Soc. for Optical Engineering*, vol. 7443, 2009.

[3] W. Kim, A. Ortega, P. Lai, T. D, and C. Gomila, "Depth map distortion analysis for view rendering and depth coding," in *Proc. IEEE Int. Conf. on Image Processing*, Cairo, Egypt, Nov 2009.

[4] G. Cheung, V. Velisavlevic, and A. Ortega, "On dependent bit allocation for multiview image coding with depth-image-based rendering," *IEEE Trans. on Image Proc.*, vol. 20, pp. 3179–3194, 2011.

[5] T. Maugey, A. Ortega, and P. Frossard, "Graph-based representation and coding of multiview geometry," in *Proc. Int. Conf. on Acoust., Speech and Sig. Proc.*, Vancouver, Canada, 2013.

[6] ——, "Multiview image coding using graph-based approach," in *IEEE Workshop on 3D Image/Video Technologies and Applications (IVMSP)*, Seoul, Korea, Jun. 2013.

[7] S. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter-banks for graph structured data," *IEEE Trans. on Signal Proc.*, vol. 60, pp. 2786 – 2799, 2012.

[8] ——, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *IEEE Trans. on Signal Proc.*, vol. 61, pp. 4673 – 4685, 2013.

[9] S. Narang, Y. Chao, and A. Ortega, "Critically sampled graph-based wavelet transforms for image coding," in *APSIPA ASC*, Kaohsiung, Taiwan, Oct 2013.

[10] A. Said and W. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 6, pp. 243 – 250, 1996.

[11] S. Li and W. Lis, "Shape-adaptive discrete wavelet transforms for arbitrarily shaped visual object coding," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 10, pp. 725 – 743, 2000.