A FAST LAYER-BASED MULTIVIEW IMAGE CODING ALGORITHM

Andriy Gelman, Jon Oñativia, Pier Luigi Dragotti

Communications and Signal Processing Group, Imperial College, London, UK

ABSTRACT

In [1, 2] we presented a multiview image coding scheme. The approach is based on extracting depth layers from multiview images. Each layer is related to an object in the scene and is highly redundant. We exploit this redundancy by using a properly disparity compensated Wavelet Transform, followed by quantisation and entropy coding of the transform coefficients. In addition, to reconstruct the data we encode a 2D contour associated with each layer.

In this paper we describe extensions to this coding scheme aimed at reducing the complexity of the algorithm, while retaining its rate-distortion (RD) performance. In particular, we propose a novel low-complexity scheme to encode the layer contours. Simulation results show that the approach remains competitive with the original method. In addition we show that the reduced complexity algorithm still outperforms H.264/AVC.

Index Terms— Multiview image coding, wavelet transform, image-based rendering, contour encoding

1. INTRODUCTION

Multiview images are obtained by recording a scene from different viewpoints using an array of cameras. These datasets have become an important component in a wide range of signal processing applications. In the computer graphics community, multiview images are used to create photorealistic novel viewpoints where no camera exists. The process of creating virtual views from images is known as image-based rendering (IBR) [3].

In our previous work [1, 2] we analysed the structure of multiview images and developed a coding scheme which achieves a high compression and outperforms H264/AVC. The method is based on extracting 'layers' from multiview images (see Figure 1); the redundancy of each layer is then reduced using a multi-dimensional Discrete Wavelet Transform (DWT). This is followed by efficient coding of transform coefficients and the layer contours.

The approach in [1, 2] however has two high complexity stages. These are the layer extraction method and the encoding of the layer contours. The layer extraction is implemented using a variational scheme, where the boundary of each layer is iteratively evolved in the direction which minimises an appropriate cost function. In addition, the layer contours are modeled using the level-set method which in itself is a complex procedure. The layer contours are encoded using the quadtree prune-join scheme [4], and although this approach achieves the same rate-distortion (RD) performance as an optimal coding scheme, the method is highly complex.

In this paper, we present alternatives to the outlined high complexity coding stages. Namely, we present a low complexity method to extract the layers and a novel approach to encode the layer contours. Simulation results show that the alternative stages incur a small cost in RD performance to the complete coding scheme. Moreover, the method still outperforms H.264/AVC.

This paper is organised as follows. Next we outline the problem setup, review the data structure of multiview images and introduce the layer-based representation. In Section 3 we overview the coding scheme and present the reduced complexity algorithm extensions. Finally, we evaluate the complete coding method in Section 4 and conclude in Section 5.

2. MULTIVIEW IMAGE STRUCTURE AND LAYER-BASED REPRESENTATION

For clarity we assume that the input dataset is an array of images. The cameras which capture the scene are uniformly spaced on a line, and their direction is perpendicular to the location line. In addition we assume a calibrated setup, such that the location of each camera is explicitly known. This analysis allows us to parameterise the dataset using:

$$I = P_3\left(x, y, V_x\right), \tag{1}$$

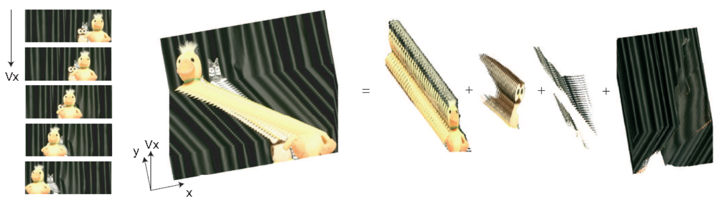

Fig. 1. Multiview image layer-based representation. A multiview image dataset can be segmented into a set of layers, where each one is related to a constant depth in the scene.

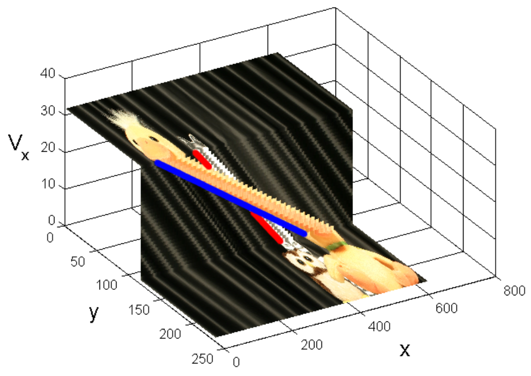

Fig. 2. EPI volume cross section. This figure is obtained by stacking the images into a volume (in the V_x direction) and taking a vertical slice through the dataset. Two EPI lines which are modelled by two points in space are illustrated. The EPI line corresponding to a smaller depth (blue) occludes the larger depth EPI line (red).

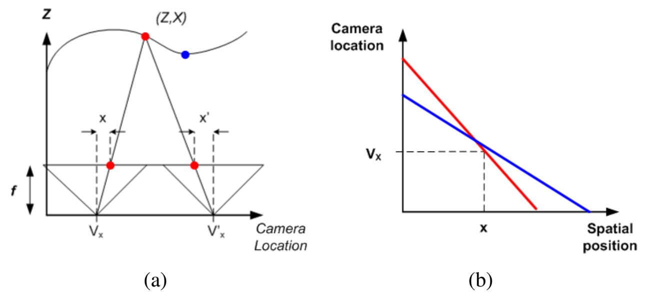

Fig. 3. (a) Camera setup. The sampling camera moves along a straight line; the direction of the camera is perpendicular to the camera location line. (b) Each point in space maps to a line in the EPI volume. Observe that the blue object is closer to the focal plane and therefore occludes the red object. It can be shown using (2) and (3) that a data sample (x, y, V_x) can be mapped onto a different viewpoint V'_x with spatial coordinates $x' = x - \frac{f(V'_x - V_x)}{Z}$ and $y' = y$.

where V_x corresponds to the camera location, (x, y) are the spatial coordinates of each image and I is the pixel intensity (we consider a monochromatic setting). This type of dataset is also known as an EPI volume [5]. Note that our work is not restricted to this setup, and that the method can be extended to encode a 2D array of images.

The EPI volume defined by (1) is a highly structured and redundant dataset. By structure we mean that the fundamental component of multiview images are lines along which the pixel intensity is approximately constant; this concept is shown in Figure 2. This illustration is obtained by stacking an array of images into a volume and taking a cross section through the dataset. It can be observed that pixels are redundant along lines of varying gradients. The set of pixels along which the intensity of the volume is constant is known as an *EPI line*.

In order to demonstrate why the fundamental component of multiview images are EPI lines, consider the setup in Figure 3(a). Here we show a simplified version of the scene: the horizontal axis corresponds to the camera location line; the line parallel to it defines the focal plane of each camera[1]; and the vertical axis defines the depth of the scene. The curved line corresponds to the surface of the object.

Given this setup consider a point in space with coordinates (X, Y, Z). Assuming a Lambertian scene[2] this point will appear in each of the images (V_x) with coordinates

$$x = \frac{fX}{Z} - \frac{fV_x}{Z}, \tag{2}$$

$$y = \frac{fY}{Z}, \tag{3}$$

[1]Each camera in the setup is modelled by the pinhole model.

[2]Light ray intensity is constant when an object is observed from a different angle.

where f is the focal length. As illustrated in Figure 3(b), the spatial coordinate x is linearly related to the camera location V_x. The rate of change in the pixel location, also known as the *disparity* $\Delta p = \frac{f}{Z}$, is inversely related to the depth of the object. This analysis tell us that a point in space with coordinates (X, Y, Z), maps to a constant intensity line in the EPI volume. Moreover, objects closer to the focal plane (smaller Z), correspond to lines with a larger disparity.

The EPI lines in the volume have varying gradients (due to objects located at different depth in the scene) and may intersect at a point. Clearly, when two lines intersect, the EPI line corresponding to a smaller depth (larger disparity) will occlude all the EPI lines which are related to larger depths (smaller disparity) in the scene. This principle is illustrated in Figure 2.

2.1. Layer-based representation

In the previous section we showed that multiview images consist of EPI lines along which the intensity of the pixels is approximately constant. This concept can be further extended by grouping EPI lines with the same disparity into a single volume (we call this a layer). In our work we segment a multiview image dataset into a set of layers, where each one corresponds to a constant depth in the scene. We call this the *layer-based representation*. An example of the representation is illustrated in Figure 1.

In addition to segmenting the data into redundant regions, this representation allows us to efficiently encode the contour of each layer. Consider a particular layer in the representation. If the layer does not intersect with other layers in the dataset, its segmentation can be efficiently defined by a 2D contour γ on one of the image viewpoints shifted by the disparity Δp to the remaining images in the dataset. This concept is illustrated in Figure 4(a). Then, to infer the occluded regions, we can use the property that each layer consists of

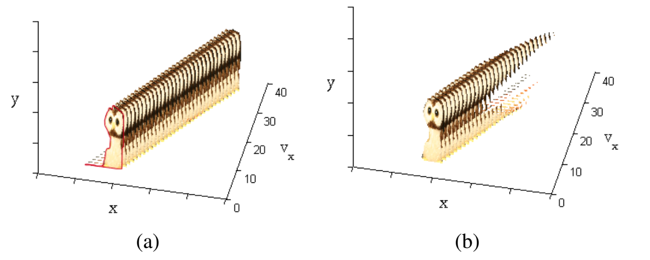

Fig. 4. Layer from the Animal Farm dataset. (a) The unoccluded layer can be defined using the contour $\gamma(s)$ on one viewpoint projected to the remaining frames. The 2D contour is denoted by the red curve on the first image. (b) Occluded layer can be inferred by removing the regions which intersect with other layers related to a smaller depth.

EPI lines related to a constant depth in the scene. Therefore, to infer the layer segmentation in Figure 4(b) we simply remove the regions which intersect with other layers that are modelled by a larger disparity (smaller depth). In summary the boundary of each layer in Figure 1 can be defined by a set of 2D contours $\{\gamma_1, \ldots, \gamma_N\}$ and a set of disparities $\{\Delta p_1, \ldots, \Delta p_N\}$, where N is the total number of layers.

3. MULTIVIEW IMAGE CODING ALGORITHM

In this section we present our reduced complexity layer-based multiview image coding algorithm. Next we outline a high-level overview of the coding scheme.

3.1. High-level overview

A high-level overview of the coding method is shown in Figure 5. The input is an array of multiview images and the output is a bitstream that meets the bit budget $\mathbf{R_t}$. Initially, the images are segmented into a set of layers using the low complexity algorithm outlined in Section 3.2.

Then, given a target bit budget for the contour encoding, $\mathbf{R_s}$, we encode the layer contours $\{\gamma_1, \ldots, \gamma_N\}$ in a lossy or lossless modality. In the lossless case, the layer contours are coded using a modified version of the Freeman method [6]. In the lossy setup, we compress the contours by transmitting a subset of the original vertices (as discussed in Section 3.3). Note that the set of disparities $\{\Delta p_1, \ldots, \Delta p_N\}$ is losslessly encoded and stored.

This stage is followed by the coding of the texture in the layer-based representation. First, we apply a 1D DWT in the inter-view domain. The transform is applied to each layer independently, and is modified to filter the pixels in the direction of the EPI lines. In addition, we explicitly use the knowledge of the layer contours to modify the transform when filtering across an artificial boundary [2]. Subsequently we apply a 2D DWT to the spatial coordinates to further remove any redundancy. Given a target bit budget $\mathbf{R_x}$, the transform

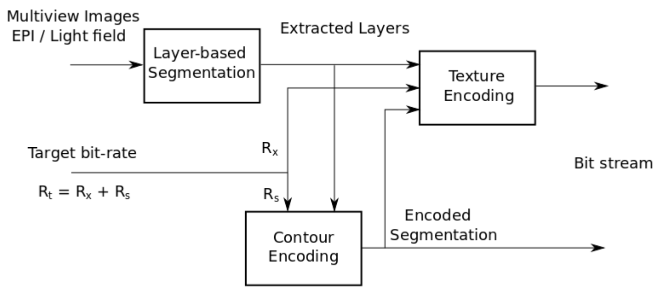

Fig. 5. High-level block diagram of the multiview image coding algorithm. We refer the reader to [1] for an in depth description of the method.

coefficients are quantised and entropy coded using a method similar to EBCOT [7].

The rate allocation between the layer contours $\mathbf{R_s}$ and the texture $\mathbf{R_x}$ is performed according to the strategy presented in [1] such that $\mathbf{R_t} = \mathbf{R_x} + \mathbf{R_s}$.

In the following, we describe in more detail the reduced complexity stages of the algorithm.

3.2. Reduced complexity layer-extraction method

Here we overview our approach to segment a multiview image array into the layer-based representation as in Figure 1. Recall from Section 2.1 that the problem can be analysed as obtaining a set of 2D contours $\{\gamma_1, \ldots, \gamma_N\}$ and a set of the disparities $\{\Delta p_1, \ldots, \Delta p_N\}$ according to some metric.

We choose the metric by using the analysis that the intensity in the direction of each EPI line is approximately constant. This implies that the parameters can be chosen by minimising the pixel variance in the direction of the EPI lines. We refer the reader to [8] for an in depth overview of our approach to construct the cost function. In [8], this problem was solved in an iterative approach by evolving the contour of each layer γ_k, followed by the estimation of the disparities Δp_k. However, the evolution of the layer contours is a complex procedure, where an appropriate velocity vector must be evaluated to minimise the cost function.

To significantly reduce the complexity of this method, we make an assumption that a good approximation of a set of contours $\{\gamma_1, \ldots, \gamma_N\}$ can be obtained using a stereo matching algorithm (we use [9] in our implementation). Recall that a stereo matching algorithm outputs a piecewise constant disparity map, and this can be used to define the set of 2D contours.

The optimisation problem can hence be simplified to evaluating the set of disparities $\{\Delta p_1, \ldots, \Delta p_N\}$, assuming the layer contours are fixed. This is implemented using the MATLAB non-linear optimisation toolbox.

3.3. Reduced complexity lossy contour encoding method

The input to this stage are a set of 2D contours $\{\gamma_1, \ldots, \gamma_N\}$ and a bit budget $\mathbf{R_s}$. The lossy layer contour encoding method is based on a piecewise linear description of the contours.

Each contour γ_k is defined by a set of vertices $\gamma_k = \{(x_1, y_1), \ldots, (x_{L_k}, y_{L_k})\}$, where L_k is the total number of vertices in γ_k. We compress the contour by transmitting only a subset of the vertices, and this leads to a piecewise linear approximation as shown in Figure 6.

These vertices are selected to satisfy two constraints: a maximum perpendicular error e_{max} between the real contour and the approximated linear contour, and a maximum distance between two consecutive vertices Δ_{max} along the x and y directions. Figure 6 illustrates these constraints. We obtain the approximation in a top-down approach by iteratively removing vertices when the two constraints are satisfied. This method is inspired by Schuster and Katsaggelos' [10]. Their approach also approximates a boundary of a shape by a polygon, but the vertex selection is based on finding a shortest path in a graph or by convex optimisation. Our approach presents a lower complexity. We also note that the method in [11] is in spirit similar to ours. However, they encode the boundaries of a segmented image and such boundaries are adjacent. Whereas we encode contours of potentially overlapping layers.

The retained vertex locations are then differentially encoded according to the maximal distance Δ_{max}. The resulting bitstream is compressed with a lossless arithmetic encoder to further increase the compression efficiency.

Observe that the maximum perpendicular error e_{max} is directly related to the length of the bitstream. Bigger errors result in selecting fewer vertices of the original contour, leading to a shorter bitstream. For a given bit budget $\mathbf{R_s}$, and a given set of binary shape images, the encoding algorithm adapts the maximum perpendicular error to satisfy the bit budget. We have experimentally determined that starting with an error of 5 pixels, and reducing it by 0.5 in each iteration gives a good RD behaviour, and the resulting bit rates satisfy the requirements of the overall multiview compression algorithm.

We note that the proposed contour encoding scheme can be extended to support a progressive (scalable) description. The idea behind this feature is that a subset of the total bitstream permits reconstructing a coarse version of the contours that can be refined as more bits are available. This is implemented by encoding different stages of refinement in the bitstream. The first stage contains a coarse approximation of the contour corresponding to vertices constrained by a large e_{max} and Δ_{max}. This coarse approximation is then refined with subsequent stages that add new vertices. The new vertices correspond to smaller constraints e'_{max} and Δ'_{max}. Various stages of refinement can be added, each stage further reducing the constraints.

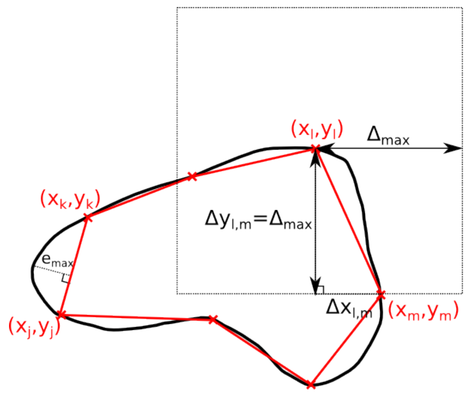

Fig. 6. Vertex finding constraints in contour approximation. Vertices (x_j, y_j) and (x_k, y_k) are constrained by the maximal perpendicular error e_{max} and vertices (x_l, y_l) and (x_m, y_m) are constrained by Δ_{max} in the vertical direction.

4. ALGORITHM EVALUATION

To evaluate the performance of the proposed method we compare it to H.264/AVC (High Profile) and to the original algorithm presented in [1, 2].

In Figures 7 and 8 we show an example of where the reduced complexity method outperforms H.264/AVC. The corresponding multiview datasets are Teddy EPI $[368 \times 352 \times 4]$ and Doll EPI $[544 \times 608 \times 4]$ (both from [12]). When encoding Teddy EPI, observe that our approach achieves a PSNR gain of over 2dB at 0.05bpp. In comparison to the original algorithm, the low complexity stages incur a small loss in RD performance. At high rate, the original method has more accurate disparity compensation at the layer edges, and this leads to more efficient compression. In addition, at low rates, the low complexity contour encoding method incurs a loss in comparison to the original quadtree prune-join contour encoding scheme.

In Figure 9 we show an example of when the simplified layer extraction method significantly affects the RD performance at high rates. In this case, the layer contours are not accurate around the object boundaries and this affects the compression performance. To confirm this analysis, we also include a RD curve (denoted by the green curve) where the accurate layer contours are obtained using the original level-set method and are encoded using the proposed contour encoding scheme of Section 3.3. As illustrated the contour encoding method does not affect the performance at high rates. The contour encoding stage can therefore be safely replaced by the new approach.

Regarding the reduction in complexity, we note that the novel contour encoding approach reduces the execution time of this stage by more than two orders of magnitude compared to the quadtree approach in a MATLAB implementation as shown in Table 1.

Table 1. Performance comparison of the contour encoding stage. The time is measured on MATLAB implementations running the algorithm 100 times and averaging the elapsed time.

Dataset	Time (seconds) - Quadtree	Time (seconds) - New approach
Teddy	2792.34	18.41
Doll	1004.37	15.53
Tsukuba	3801.54	6.02

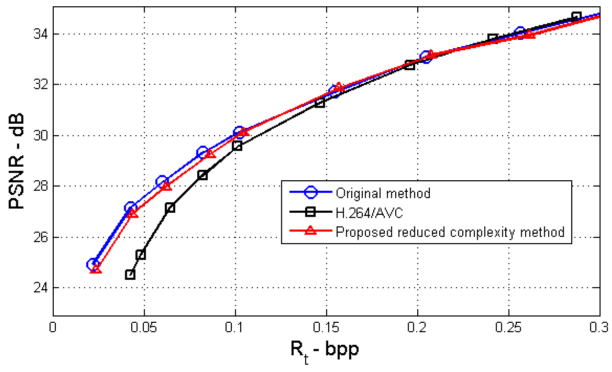

Fig. 7. Proposed method evaluation on Teddy EPI.

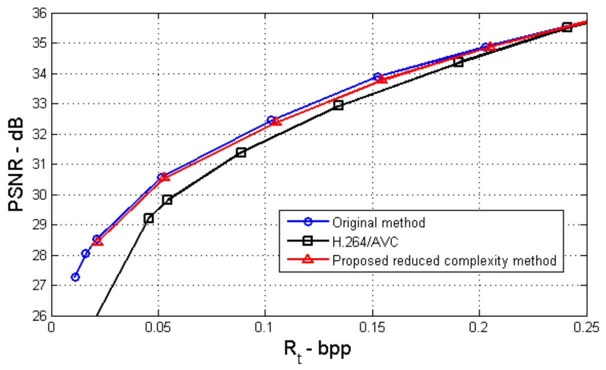

Fig. 8. Proposed method evaluation on Doll EPI.

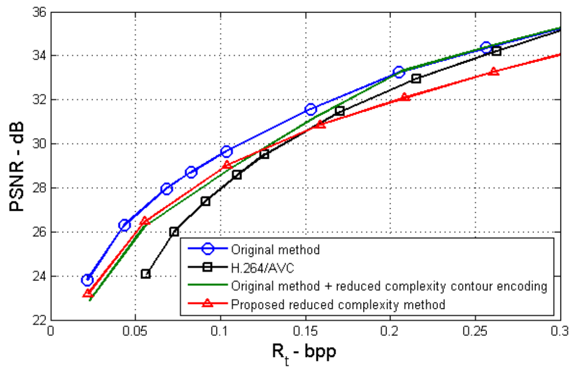

Fig. 9. Proposed method evaluation on Tsukuba EPI.

5. CONCLUSION

In this paper we proposed low complexity extensions to a layer-based multiview image compression algorithm [1, 2]. Firstly, we presented an efficient approach to extract layers from multiview images. The method assumes that the outline of each layer can be accurately obtained by using a stereo-matching algorithm. Secondly, we presented a low complexity alternative to encoding the outline of each layer. Our approach is based on transmitting a subset of vertices, and this results in a piecewise linear approximation of the contours. The subset of retained vertices is selected by satisfying two constraints: a maximal perpendicular error to the original contour, and a maximal distance to a neighbouring vertex. The method is also suitable for progressive coding of the contours. Simulation results show the alternative low complexity stages result in a significant reduction of running time and a small RD performance loss in comparison to the original algorithm. Moreover, the method still outperforms H.264/AVC on a number of real datasets.

6. REFERENCES

[1] A. Gelman, P. L. Dragotti, and V. Velisavljevic, “Multiview image coding using depth layers and an optimized bit allocation,” *IEEE Transactions on Image Processing, to appear*, 2012.

[2] A. Gelman, P. L. Dragotti, and V. Velisavljević, “Multiview image compression using a layer-based representation,” in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Sep. 2010.

[3] H. Y. Shum, S. C. Chan, and S. B. Kang, *Image-Based Rendering*, Springer-Verlag, 2007.

[4] R. Shukla, P. L. Dragotti, M. N. Do, and M. Vetterli, “Rate-distortion optimized tree-structured compression algorithms for piecewise polynomial images,” *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 343–359, Mar. 2005.

[5] R. Bolles, H.H. Baker, and D. Marimont, “Epipolar-plane image analysis: An approach to determining structure from motion,” *International Journal of Computer Vision*, vol. 1, pp. 7–55, March 1987.

[6] Y.K. Liu and B. Zalik, “An efficient chain code with huffman coding,” in *Pattern Recognition*, Apr. 2005, vol. 38, pp. iv/553–iv/557.

[7] D. Taubman, “High performance scalable image compression with EBCOT,” *IEEE Transactions on Image Processing*, vol. 9, no. 7, pp. 1158–1170, Jul. 2000.

[8] A. Gelman, J. Berent, and P. L. Dragotti, “Layer-based sparse representation of multiview images,” *EURASIP Journal on Advances in Signal Processing*, Mar. 2012.

[9] V. Kolmogorov and R. Zabih, “Multi-camera scene reconstruction via graph cuts,” in *Proceedings of the 7th European Conference on Computer Vision-Part III (ECCV)*. 2002, pp. 82–96, Springer-Verlag.

[10] G. M. Schuster and A. K. Katsaggelos, “An optimal polygonal boundary encoding scheme in the rate distortion sense,” *IEEE Transactions on Image Processing*, vol. 7, no. 1, pp. 13–2, Jan. 1998.

[11] M. Servais, T. Vlachos, and T. Davies, “Progressive polygon encoding of segmentation maps,” in *International Conference on Image Processing, 2004 (ICIP’04)*, 2004, vol. 2, pp. 1121–1124.

[12] D. Scharstein and R. Szeliski, “Middlebury data sets,” *http://vision.middlebury.edu/stereo*.