Multiview Image Coding using Depth Layers and an Optimized Bit Allocation

Andriy Gelman, *Student Member, IEEE,* Pier Luigi Dragotti, *Senior Member, IEEE,* and Vladan Velisavljević, *Senior Member, IEEE*

Abstract

In this paper, we present a novel wavelet-based compression algorithm for multiview images. The method uses a layer-based representation, where the three-dimensional (3D) scene is approximated by a set of depth planes with associated constant disparities. The layers are extracted from a collection of images captured at multiple viewpoints and transformed using the 3D discrete wavelet transform (DWT). The DWT consists of the one-dimensional (1D) disparity compensated DWT across the viewpoints and two-dimensional (2D) shape-adaptive DWT across the spatial dimensions. Finally, the wavelet coefficients are quantized and entropy coded along with the layer contours. To improve the rate-distortion (RD) performance of the entire coding method, we develop a bit allocation strategy for distribution of the available bit budget between encoding the layer contours and the wavelet coefficients. The achieved performance of our proposed scheme outperforms the state-of-the-art codecs for several data sets of varying complexity.

Index Terms

Multiview image coding, compression, wavelet transforms, bit rate allocation.

I. Introduction

Rendering visual data is an important component of most entertainment and information services, ranging from e-commerce to computer gaming. A recent trend in the state-of-the-art has been to combine

A. Gelman and P. L. Dragotti are with the Department of Electrical and Electronic Engineering, Imperial College London, London, SW7 2AZ United Kingdom (e-mail: ag504@imperial.ac.uk; p.dragotti@imperial.ac.uk).

V. Velisavljević is with Deutsche Telekom Laboratories, Ernst-Reuter-Platz 7, 10587 Berlin, Germany (email: vladan.velisavljevic@telekom.de).

image primitives with the traditional geometry-driven algorithms to achieve improved photo-realistic results. These are also known as image-based rendering (IBR) algorithms. Examples of intensively researched applications are 3DTV and free viewpoint TV (FTV) [1]. In FTV, the audience can choose the viewing angle, whereas 3DTV adds a perception of depth to enhance the viewer's experience.

One of the main challenges in IBR is data compression and transmission [2]. In a typical setup, such as a Light Field [3], hundreds of cameras may be used to capture the scene. Therefore, to store or transmit this data, an efficient compression algorithm is essential.

Many compression algorithms with variations in complexity, efficiency, scalability and random access have been proposed. For instance, when encoding a 2D array of images, a popular approach [4], [5], [6] has been to use the predictive block-based transform coding extended to the additional viewing dimensions. Wavelet based codecs [7], [8] achieve high compression rates and naturally support scalable bit streams at the expense of higher complexity. In addition, the lifting implementation [9] has been used to ensure invertibility and reduce the complexity. A number of proposed approaches [10], [11], [12], [13] exploit the geometrical structure in the scene by 3D modeling prior to encoding the data so that the compression efficiency is improved. The bit rate allocation problem between texture and scene geometry has, for example, been studied in [14] and [15], where the geometry is defined using a per pixel disparity map. In our case, we model the scene using a set of constant depth planes and this requires a different rate-distortion (RD) model to the one proposed in the literature.

We propose a novel compression algorithm for multiview images. For clarity, the problem is simplified to the compression of a 1D array of images taken with a camera moving along a straight line. The viewing direction of the camera is perpendicular to this line. Initially, the data is partitioned into layers, where each layer is modeled by a constant depth plane. The redundancy of the data is reduced by using a 3D DWT applied in a separable approach across the viewpoint and spatial dimensions, followed by quantization and entropy coding. To achieve a high compression rate, the viewpoint transform is modified to include disparity compensation, and is implemented using the lifting scheme. The contours of the layers are also transmitted and can be encoded in a lossy or lossless modality.

To ensure an efficient RD performance, we also address the problem of rate allocation between the layer contours and the transform coefficients. Given a target bit budget, the problem lies in finding the correct ratio which will minimize the output distortion. It is shown that a closed-form solution can be obtained when the contours are piecewise linear. In the more general case of non-piecewise linear contours, the theoretical analysis is combined with an empirical approach to obtain the correct allocation. We also consider an alternative strategy where a constant percentage of the total bit budget is allocated to the layer contours. We show that both methods lead to a compression performance which is significantly better than the performance achieved by the H.264/AVC [16] and MVC [17] codecs.

The outline of this paper is as follows. In Section II we describe the structure and redundancy of multiview images and introduce the layer-based representation. We give a high-level overview of the novel compression algorithm in Section III. In Section IV we review a segmentation method used to obtain the layer-based representation and present the compression method in Section V. The RD modeling is outlined in Section VI. Finally, we show experimental results in Section VII and conclude the paper in Section VIII.

II. Multiview image data structure

Capturing the light rays of the entire 3D scene is a complex task due to a high amount of information required to be recorded. The general concept of *plenoptic functions* [18] results in a 7D function, where three coordinates determine the location of the viewpoint (camera in the scene), two are used to describe the inclining angle of the light rays and one coordinate each for the time and spectral component (color). Due to the complexity of the problem, many simplifications have been proposed to reduce the dimensionality of this function [19].

In this work, we drop the temporal and color dimensions, thus, processing only multiple view *still monochromatic* images. Further, we constrain the camera locations to a 1D array or a 2D plane[1]. These concepts are known as *epipolar-plane image* (EPI) array and *light field* [3], respectively. We demonstrate in Section VII the analysis and results for both concepts.

In the remaining part of this section, we review the EPI and light field data structures, highlight the data redundancy and introduce the layer-based representation.

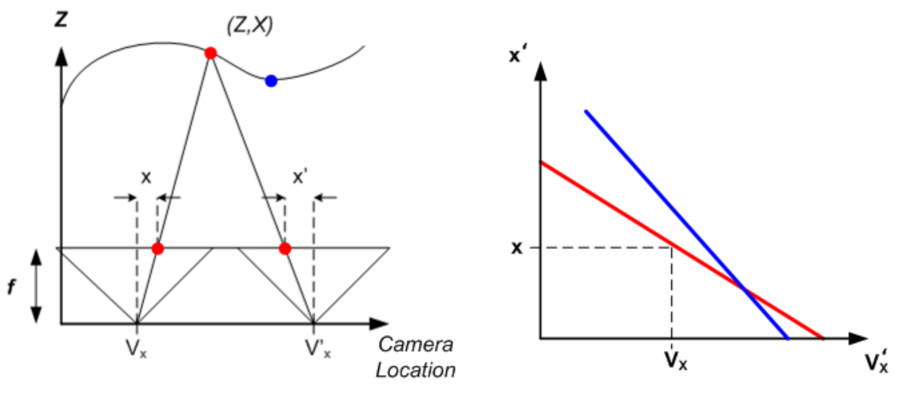
A. Epipolar-Plane Image Array and Light Field

Consider the setup in Fig. 1(a), where the scene is sampled using a pinhole camera along the viewpoint V_x. In this case, the multiview data can be parameterized using a 3D function, known as the EPI volume:

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

where (x, y) are the spatial coordinates of the camera located at position V_x and I is the monochromatic intensity of the light ray passing through the focal plane and the camera center.

[1]The directions of the camera are perpendicular to the 2D camera plane.



(a) (b)

Fig. 1. (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 $y' = y$.

Understanding the structure of the EPI volume is important to achieve high compression. Consider a point in space with coordinates (X, Y, Z). This point is projected to each image along the V_x dimension such that the intersection of the light ray and the focal plane is given by:

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

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

where f is the focal length. Observe that the spatial coordinate x is linear with respect to the camera position and the gradient, which is also called the disparity $\Delta p = \frac{f}{Z}$, is inversely proportional to the depth. We define the set of coordinates in (2) and (3) for each point in space (X, Y, Z) as an EPI line. Furthermore, assuming that the scene is Lambertian[2], there are no occlusions and the intensity of the light ray does not change along its path, the intensity of the EPI line is constant. This property is illustrated in Fig. 1(b), where a point in space maps onto a line in the EPI domain.

In addition to the constant intensity along the EPI lines, observe that the occlusion ordering can also be predicted. Recall that the disparity Δp of an EPI line is inversely proportional to the depth. Therefore, as shown in Fig. 1(b), when two lines intersect, the line corresponding to the larger disparity will occlude the other.

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

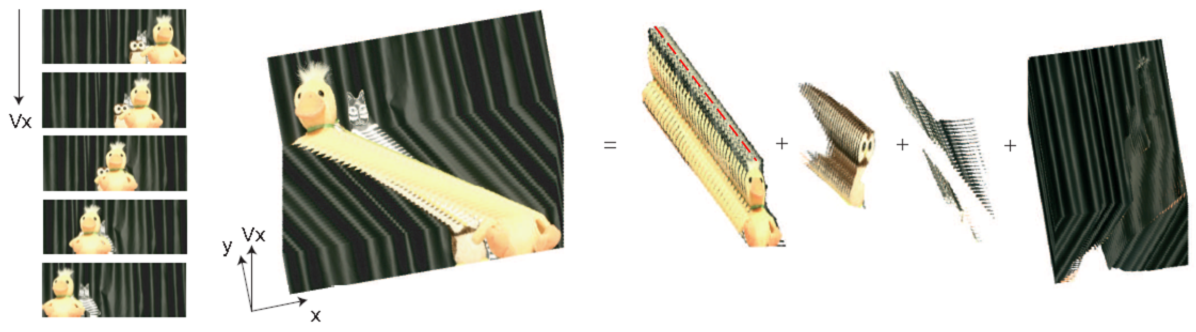


Fig. 2. Animal Farm [20] layer-based representation. The original dataset can be divided into a set of layers, where each layer is related to a constant depth in scene. The dashed line corresponds to one particular EPI line. Observe that the contours of each layer remain constant, unless there is an intersection with another layer which is modeled by a smaller depth.

The same analysis can also be applied to a light field where the scene is sampled along both the V_x and V_y dimensions. In this case, the point of intersection of the light ray and the focal plane is given by:

$$x = \frac{fX}{Z} - \frac{fV_x}{Z}, \tag{4}$$

$$y = \frac{fY}{Z} - \frac{fV_y}{Z}. \tag{5}$$

Observe that in comparison to an EPI line, (4) and (5) form a 2D set of points, along which, the intensity of the light field is constant. For clarity, in the rest of this paper we only consider the EPI volume to present our compression method.

B. Layer-based representation

The layer-based representation is an extension of the EPI line concept. The representation partitions the multiview data into homogenous layers, where each layer is a collection of EPI lines modeled by a constant depth plane. Recall that the constant depth assumption implies that the EPI lines have the same gradient and are redundant in the direction of the disparity. An example of a representation with four layers is shown in Fig. 2.

In the proposed algorithm, the layer segmentation must also be transmitted in order to reconstruct the data correctly. Note that given no prior knowledge of the scene geometry, the layer contours would have to be encoded at each image viewpoint. This information would require a large number of bits and be costly in terms of compression. However, using the fixed depth model, an unoccluded layer can be defined by a 2D boundary on one image projected to the remaining viewpoints. To infer occlusions, just as in the case of EPI lines, a layer will be occluded when it intersects with a layer which is modeled by

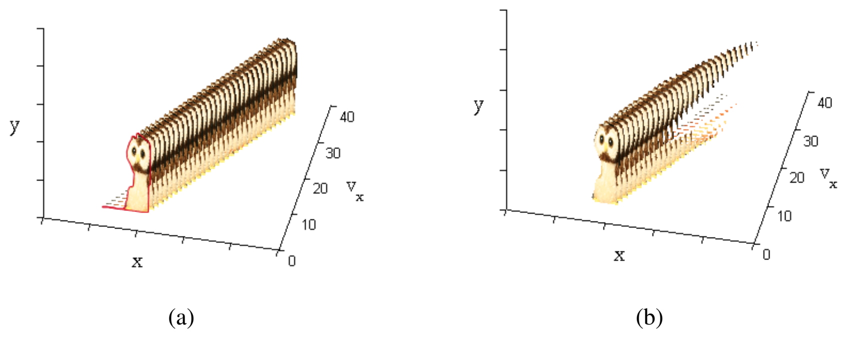


Fig. 3. Layer from the Animal Farm dataset. (a) The unoccluded layer can be defined using the contour on one viewpoint projected to the remaining frames. The 2D contour is denoted by the red curve on the first image. (b) Occlusions can be inferred by removing the regions where there are intersections with layers which are modeled by a smaller depth.

a smaller depth. Therefore, to fully define the segmentation as illustrated in Fig. 3(b), we simply require the layer contour on one viewpoint, its disparity and the preceding layers corresponding to a smaller depth.

Analogous to standard video coding, the layer-based representation defines the motion vectors used in predictive coding. In a conventional coding scheme, motion vectors are evaluated using a block-based approach and are encoded for each predictive frame, which has a higher encoding rate than the layer-based representation. In addition, a block-based approach does not effectively model real data. In practice a block often overlaps with two objects experiencing different motion. This will in turn create a large residual error and therefore cause edges to appear blurred. The layer-based representation on the other hand models the boundary as a smooth curve and is therefore more realistic. Finally, occluded regions often create a large residual error and have to be intra-coded. In our case, the location of occluded regions is explicitly known and we modify the inter-view transform to deal with these regions.

The layer-based representation is an important concept of the proposed compression algorithm and we present an overview of the method next.

III. High-level compression algorithm overview

The high-level layout of the novel compression algorithm is shown Fig. 4. The input data is initially segmented to obtain the layer-based representation. Each layer is assigned a global disparity, which is losslessly encoded and transmitted. The layers are then passed to the coding stage where a disparity compensated 3D DWT is applied to remove the redundancy. This is followed by quantization and entropy coding of the transform coefficients. In order to reconstruct the texture at the decoder, the layer contours are also encoded in a lossy or lossless modality and transmitted.

Observe that for a given total bit budget, there exists a trade-off between the number of bits allocated to encode the layer contours and the texture. This problem is solved in the RD control stage, which prior to encoding, correctly distributes the bits between the texture and the layer contours to maximize the output fidelity of the images. The encoding method is outlined in Algorithm 1.

Algorithm 1 Compression algorithm overview

```
STEP 1: Segment the multiview images to obtain the layer-based representation.
STEP 2: For a given bit budget $\mathbf{R_t}$, find the correct rate allocation to encode the layer contours $\mathbf{R_s}$ and texture $\mathbf{R_x}$ such that $\mathbf{R_x} + \mathbf{R_s} = \mathbf{R_t}$.
STEP 3: Given the rate constraint $\mathbf{R_s}$, encode the layer-contours in a lossy or lossless modality.
STEP 4: If the layer contours are encoded in a lossy modality, update the layer-based representation such that they correspond to the encoded version.
STEP 5: Given the target bit rate $\mathbf{R_x}$ encode the texture in each layer.
```

In the following sections, we describe each of the points in more detail. First we review the segmentation process used to obtain the layer-based representation. Then we describe the texture and contour encoding stages. Finally, we present the RD method which correctly distributes the bit budget between the layer contours and the texture.

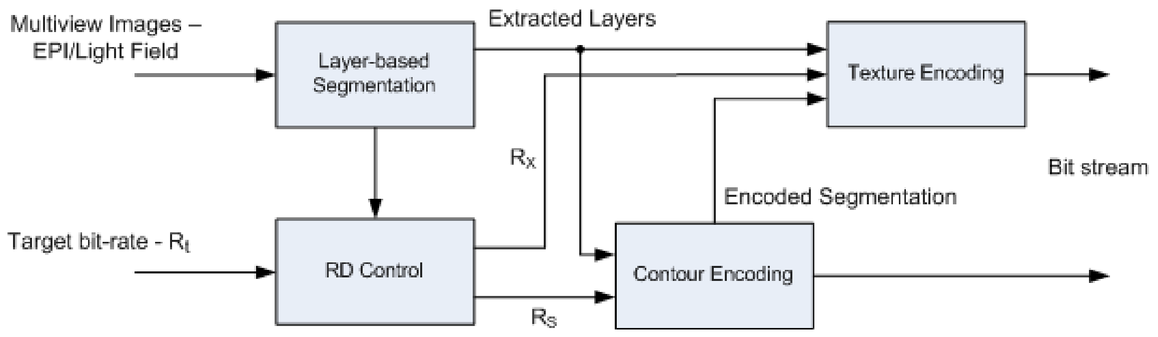


Fig. 4. High-level algorithm layout. Layers are extracted in the layer-based segmentation block, and each layer is transformed using a 3D DWT in the texture coding stage. The layer contours are encoded in a lossy or lossless modality and transmitted to the decoder. The method includes a RD control stage, which correctly distributes the bit budget between the layer contours and the texture to maximize the output fidelity.

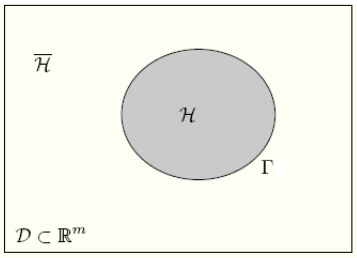


Fig. 5. An m-dimensional dataset $\mathcal{D}$ is partitioned into $\mathcal{H}$ and $\overline{\mathcal{H}}$. The boundary is a closed curve defined by Γ [23].

IV. Layer-based segmentation

Data segmentation is the first stage of the proposed compression algorithm and is used to obtain the layer-based representation. Here we review the segmentation algorithm proposed in [20], [21], which achieves accurate results by taking into account the structure of multiview data. We review the method by first describing a general segmentation problem and then show how the solution can be modified to extract layers from an EPI volume. For more details about the method and region-based segmentation we refer to [20], [21], [22], [23].

A. General region-based data segmentation

Consider a general segmentation problem shown in Fig. 5. The aim is to partition an m-dimensional dataset $\mathcal{D} \subset \mathbb{R}^m$ into subsets $\mathcal{H}$ and $\overline{\mathcal{H}}$, where the boundary which delimits the two regions is defined by $\Gamma(\boldsymbol{\sigma})$, with $\boldsymbol{\sigma} \in \mathbb{R}^{m-1}$. This type of problem can be solved using an optimization framework, where the boundary is obtained by minimizing an objective function J:

$$\Gamma = \arg\min\{J(\Gamma)\}. \tag{6}$$

The cost function in (6) can be defined using either a boundary or region-based approach. The boundary methods evaluate the cost only on Γ and, hence, they are influenced by local data properties and easily affected by noise. In turn, the region-based methods evaluate the cost function over a complete region and they are, therefore, more robust. A typical region-based cost function [22] can be defined as:

$$J(\Gamma) = \int_{\mathcal{H}} d(\mathbf{x}, \mathcal{H})\, d\mathbf{x} + \int_{\overline{\mathcal{H}}} d\left(\mathbf{x}, \overline{\mathcal{H}}\right) d\mathbf{x} + \int_{\Gamma} \eta\, d\boldsymbol{\sigma}, \tag{7}$$

where the descriptor $d(\cdot)$ measures the homogeneity of each region and $\mathbf{x} \in \mathbb{R}^m$. Note that (7) has an additional regularization term η, which acts to minimize the length of the boundary.

The optimization problem defined in (6) cannot be solved directly for Γ. An iterative solution can, however, be obtained by making the boundary a function of an evolution parameter τ. Consider modeling the boundary using a partial differential equation (PDE), also known as an active contour [24]:

$$\frac{\partial \Gamma\left(\boldsymbol{\sigma}, \tau\right)}{\partial \tau} = v\left(\boldsymbol{\sigma}, \tau\right) = F\left(\boldsymbol{\sigma}, \tau\right) n\left(\boldsymbol{\sigma}, \tau\right), \tag{8}$$

where v is a velocity vector, which can be expressed in terms of a scalar force F acting in the outward normal direction to the boundary n. The velocity vector v can be evaluated in terms of the descriptor $d\left(\cdot\right)$ by differentiating (7) with respect to τ. Applying the Eulerian framework [22], the derivative can be expressed in terms of boundary integrals:

$$\frac{\partial J\left(\Gamma\left(\tau\right)\right)}{\partial \tau} = \int_{\Gamma(\tau)} \left[d\left(\mathbf{x}, \mathcal{H}\right) - d\left(\mathbf{x}, \overline{\mathcal{H}}\right) + \eta \kappa\left(\mathbf{x}\right)\right] \left(v \cdot n\right) d\boldsymbol{\sigma}, \tag{9}$$

where κ is the curvature of the boundary Γ and $\cdot$ denotes the dot product. The velocity vector, which evolves Γ in the steepest descent direction can be deduced using the Cauchy-Schwarz inequality as:

$$v = \left[d\left(\mathbf{x}, \overline{\mathcal{H}}\right) - d\left(\mathbf{x}, \mathcal{H}\right) - \eta \kappa\left(\mathbf{x}\right)\right] n. \tag{10}$$

The above framework is also known as 'competition-based' segmentation. This is clear from (10), where a point on the boundary will experience a positive force if it belongs to the region and vice versa, hence evolving the contour in the correct direction. In conclusion, the general segmentation problem can be solved by modeling the boundary Γ as a PDE and evolving the contour in the direction of the velocity vector v.

B. Multiview Image Segmentation

In the segmentation problem for our multiview image compression method, the goal is to extract N layers, where each layer is modeled by a constant depth Z_i or the associated disparity Δp_i. Therefore, the objective function[3] can be defined as:

$$J = \sum_{i=1}^{N} \int_{\mathcal{H}_i} d\left(\mathbf{x}, \Delta p_i\right) d\mathbf{x}, \tag{11}$$

where $\mathbf{x} = [x, y, V_x]$.

Recall that in the ideal case the intensity along the EPI lines is constant. We can therefore define the descriptor $d\left(\mathbf{x}, \Delta p_i\right)$ to minimize the squared error along the EPI lines:

$$d\left(\mathbf{x}, \Delta p_i\right) = \left[I\left(\mathbf{x}\right) - \mu\left(\mathbf{x}, \Delta p_i\right)\right]^2, \tag{12}$$

[3]We have not included the regularization terms for the sake of clarity.

where $\mu(\mathbf{x}, \Delta p_i)$ is the mean of the EPI line which passes through a point $\mathbf{x}$ and has a disparity Δp_i.

The aim is then to obtain the layer boundaries Γ_i and the disparity values Δp_i for $i = 1, \ldots, N$ by minimizing (11). Observe that this problem contains a large number of variables and is challenging to solve. We use the iterative solution proposed in [21] and outlined in Algorithm 2. First, the boundary of each layer Γ_i and the number of layers N is initialized using a stereo matching algorithm [25]. Then the disparity of each layer is evaluated by minimizing the squared error along the EPI lines. The problem is subsequently solved in an iterative approach by evolving each boundary separately assuming all the remaining layers are constant. This allows the cost function (11) to be expressed in the same format as the general segmentation problem in (7). For example, consider optimizing the l-th layer. In this case, the objective function can be expressed as:

$$J = \int_{\mathcal{H}_l} d(\mathbf{x}, \Delta p_l)\, d\mathbf{x} + \int_{\overline{\mathcal{H}_l}} d^{out}(\mathbf{x})\, d\mathbf{x}, \tag{13}$$

where $d^{out}(\mathbf{x}) = d(\mathbf{x}, \Delta p_i)$ when $\mathbf{x} \in \mathcal{H}_i$ for all $i \neq l$. Here, the second integral is the union of the remaining competition regions.

Using the same principles as in the general segmentation problem, the velocity vector corresponding to the l-th layer can be expressed in terms of the descriptor functions:

$$v_l = \left[d^{out}(\mathbf{x}) - d(\mathbf{x}, \Delta p_l)\right] n_l. \tag{14}$$

Note that in the above equation, the boundary is a 3D vector. Using the fact that each layer is modeled by a constant depth plane, the velocity vector can be simplified to a 2D vector as proposed in [20]:

$$\overline{v}_l = \left[D^{out}(s) - D(s, \Delta p_l)\right] \overline{n}_l, \tag{15}$$

where $\overline{n}_l$ is a 2D outward normal vector and s parameterizes the new 2D layer contour $\overline{\Gamma}_l$. The descriptors $D(\cdot)$ and $D^{out}(\cdot)$ are obtained as the sum of the old descriptors $d(\cdot)$ and $d^{out}(\cdot)$ over the viewpoint dimension V_x, respectively.

There are two main advantages in simplifying the evolution from a 3D to a 2D contour. First, the approach ensures that the layer boundary remains consistent across the views. Secondly, the complexity is reduced from evolving a 3D object to a 2D contour.

In conclusion, (15) defines a 2D velocity vector, which evolves the layer boundary towards the correct segmentation for each layer. An example of the extracted layers using the method outlined in Algorithm 2 is shown in Fig. 2. In addition, in Fig. 6 we show a comparison between an initialized layer-boundary using the stereo matching algorithm and the final layer contour. The obtained layers are used in the texture coding block to efficiently reduce the redundancy of the data and achieve high compression.

Algorithm 2 Layer extraction algorithm

```
STEP 1: Initialize the 2D boundary of each layer $\overline{\Gamma}_1, \overline{\Gamma}_2, \ldots, \overline{\Gamma}_N$ using a stereo matching algorithm (Algorithm [25] in our implementation).
STEP 2: Estimate the disparity of each layer $\Delta p_1, \Delta p_2, \ldots, \Delta p_N$ by minimizing the squared error along the EPI lines.
STEP 3: Iteratively evolve each layer boundary assuming the remaining layers are constant:
for $l = 1$ to N do
    Evaluate the velocity vector $\overline{v}_l$ of the $l$-th layer.
    Evolve the boundary $\overline{\Gamma}_l$ according to the velocity vector.
end for
STEP 4: Return to STEP 2 or exit algorithm if the change in the cost (11) is below a predefined threshold.
```

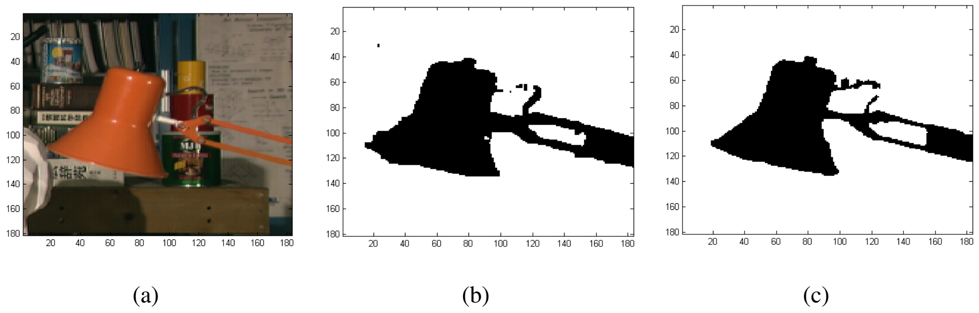


Fig. 6. The layer extraction algorithm improves the accuracy of the layer-based representation and therefore the compression efficiency. (a) Tsukuba dataset [26]. (b) Initialized layer contour using a stereo matching algorithm [25]. (c) Layer contour after running Algorithm 2.

V. Texture and Contour Encoding

In this section, we describe the contour and texture encoding blocks from Fig. 4. We denote with $\mathbf{R_s}$ and $\mathbf{R_x}$ the bits allocated for encoding the contours and texture, respectively. First, the extracted layer contours are encoded using a lossy or lossless modality to meet the given target bit budget $\mathbf{R_s}$. Then, the resulting layer segmentation is used to partition the texture from the multiview images into the layers. The texture layers are transformed to remove redundancy in the signal using a 3D DWT composed of two consecutive steps: the 1D disparity compensated DWT across the views and the 2D shape-adaptive (SA) DWT across the space. To improve the efficiency of the second spatial DWT, the inter-view wavelet subbands from each layer are recombined so that they are processed jointly. Finally,

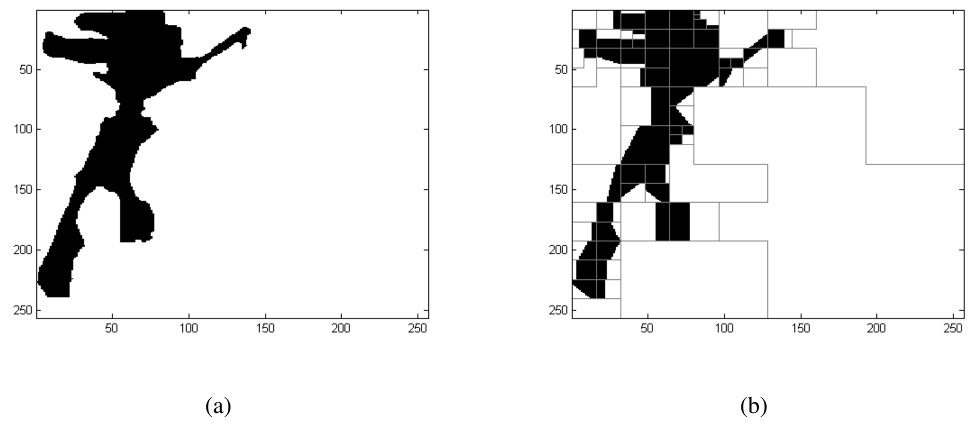


Fig. 7. (a) Original layer contour from Tsukuba dataset [26]. (b) Quadtree prune-join piecewise linear approximation obtained using the algorithm proposed in [27].

the wavelet coefficients are quantized and entropy coded in order to meet the allocated target bit budget $\mathbf{R_x}$. Each of these steps is explained next.

A. Encoding of the contours

The contour encoding block transmits the segmentation needed to correctly decode each layer. Recall, from the properties of multiview data outlined in Section II, that the segmentation of a 3D layer can be defined by a contour on one of the image viewpoints and the layer's disparity. The input to the layer contour encoding algorithm is therefore an array of binary images (one for each layer) and a target bit budget $\mathbf{R_s}$. A typical binary layer is illustrated in Fig. 7(a).

The 2D contours can be encoded in a lossless or lossy modality. The lossless method is used for $\mathbf{R_s}$ larger than the required lossless bit budget. The problem of lossless contour encoding has been extensively studied with the majority of the work based on Freeman chain codes [28]. Here, we use the algorithm proposed in [29], where the boundary is differentially encoded using Huffman entropy coding.

In the case when the given $\mathbf{R_s}$ is below the lossless encoding rate, a piecewise linear approximation of the layer contours is used. We use a 2D quadtree model to capture and quantize these linear segments, where each block of the quadtree is approximated by a $\{0, 1\}$ or intermediate tile. The intermediate tile is a binary image with an edge modeled by a linear function. These quadtree coefficients are also quantized and entropy coded. The type of tile for an arbitrary block is chosen by minimizing the cost $D(R_p) + \lambda_s R_p$, where $D(R_p)$ and R_p are the distortion associated to the p-th tile and the corresponding encoding rate, respectively.

To encode the segmentation, we use a bottom-up approach where the quadtree is pruned whenever the sum of the cost of the leaves is higher than the cost of their parent. The merging process iteratively continues as long as the Lagrangian cost decreases or the complete image is represented by a single tile. In addition, a joining scheme is used as proposed in [27], where the neighboring blocks that cannot be pruned within the quadtree representation are joined to improve the compression performance. Fig. 7 shows a comparison between an original layer contour and its quadtree prune-join representation.

The output rate of the coding algorithm is controlled by the parameter λ_s, which defines the trade-off between rate and distortion. To achieve the specified total target bit budget $\mathbf{R_s}$, a bisection search [30] for the correct λ_s is applied.

B. Inter-view 1D DWT

In this stage, the redundancy of the texture in each layer shown in Fig. 2 is reduced using a 1D disparity compensated DWT along the viewpoint dimension. While any 1D DWT can be used, for simplicity we implement the disparity compensated Haar transform. It is applied by modifying the standard lifting equations [9] and including a warping operator $\mathcal{W}$ as follows:

$$\begin{aligned} \mathscr{L}_o\left[n\right] &= \frac{P_o\left[n\right] - \mathcal{W}\{P_e\left[n\right]\}}{2}, \\ \mathscr{L}_e\left[n\right] &= P_e\left[n\right] + \mathcal{W}\{\mathscr{L}_o\left[n\right]\}, \end{aligned} \tag{16}$$

where, $P_o\left[n\right]$ and $P_e\left[n\right]$ represent 2D images with spatial coordinates (x, y) located at odd $(2n+1)$ and even $(2n)$ camera locations, respectively. Following (16), $\mathscr{L}_e\left[n\right]$ contains the 2D low-pass subband and $\mathscr{L}_o\left[n\right]$ the high-pass subband. We obtain a multi-resolution decomposition by iteration of the transform applied to the low-pass component $\mathscr{L}_e\left[n\right]$. Assuming that $\mathcal{W}$ is invertible and the images are spatially continuous, the above transform can be shown to be equivalent to the standard DWT applied along the motion trajectories [8].

In both the prediction and update steps in (16), the warping operator $\mathcal{W}$ is chosen to maximize the inter-image correlation. This is achieved by using a projective operation that maps one image onto the same viewpoint as its odd/even complement in the lifting step. Using (2) and the fact that the layers are modeled by a constant disparity, we define the warping operation from viewpoint n_1 to n_2 as:

$$\mathcal{W}_{n_1 \to n_2}\{P\left[n_1\right]\}\left(x, y\right) = P\left[n_1\right]\left(x + \Delta p\left(n_2 - n_1\right), y\right), \tag{17}$$

where Δp is the layer disparity.

Note that in the case of an occlusion, the DWT leads to filtering across an artificial boundary and, thus, results in a reduced compression efficiency. To prevent this, we use the concept proposed in [31]

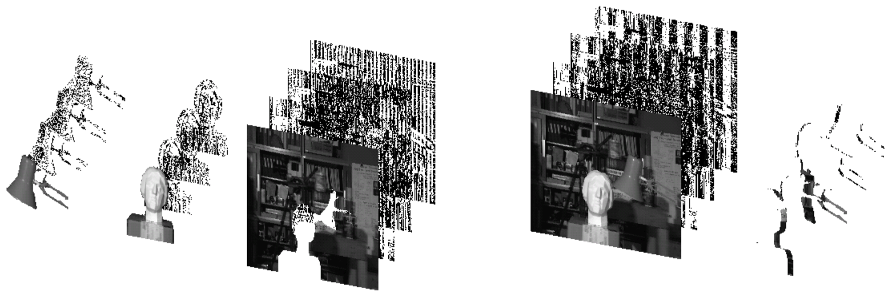


Fig. 8. (a) Tsukuba transform coefficients following the inter-view transform. Each transformed layer is composed of one low-pass subband and three high frequency images. (b) Recombined layers. The view subbands from each layer are grouped into a single image to increase the number of decompositions that can be applied by the spatial transform. Note that additional pixels must be transmitted to correctly reconstruct the layers at the decoder. A shape-adaptive implementation is used to efficiently deal with the irregular boundary of the occluded data.

to create a shape-adaptive transform in the view domain. The transform in (16) is modified whenever a pixel at an even or odd location is occluded such that

$$\mathscr{L}_e[n] = \begin{cases} P_e[n], & \text{occlusion at } 2n+1 \\ \widehat{\mathcal{W}}\{P_o[n]\}, & \text{occlusion at } 2n \end{cases}, \tag{18}$$

and the high pass coefficient in $\mathcal{L}_o[n]$ is set to zero. In (18), the warping operator $\widehat{\mathcal{W}}$ is set to an integer pixel precision to ensure invertibility and is set to be the *ceiling* of the disparity in (17). Note also that the Haar transform in (16) is not orthonormal. To ensure correct bit allocation, we normalize the wavelet coefficients so that the l_2 norm is preserved.

C. Shape-adaptive 2D DWT

To improve the efficiency of the spatial transform, the subbands from each layer view are grouped together into a single image and are further jointly processed. A comparison between the original and recombined layers is illustrated in Fig. 8. Note that due to occlusions and the way in which the inter-view transform is implemented, two or more layers may overlap in each subband. In this case, the overlapped pixels must be separately transmitted to the decoder to achieve correct reconstruction. We emphasize, however, that, since the segmentation is known prior to encoding, no additional overhead bits are required for encoding the location and segmentation of these overlapping pixels.

The occluded pixels are commonly bounded by an irregular (non-rectangular) shape. For that reason, the standard 2D DWT applied to the entire spatial domain is inefficient, because of the boundary effect.

To improve the coding efficiency, we use the SA DWT [31] within arbitrarily shaped objects. The method reduces the magnitude of the high pass coefficients by symmetrically extending the texture whenever the wavelet filter is crossing the boundary. The 2D DWT is built as a separable transform with linear-phase symmetric wavelet filters (9/7 or 5/3), which, together with the symmetric signal extensions, leads to critically sampled transform subbands.

D. Quantization and Entropy Coding

The transform coefficients are then quantized and entropy coded. Our codec uses context adaptive arithmetic coding to attain bit rates close to the entropy of the source.

For a given bit budget constraint $\mathbf{R_x}$, the optimal bit allocation among the transform coefficients is achieved using a method similar to EBCOT [32]. Initially, the coefficients are partitioned into blocks and losslessly encoded by bit-plane to obtain the operational RD curves. Then, given a Lagrangian multiplier λ_x, a bit allocation R^* for each block is chosen such that the cost function J is minimized, where:

$$J = D\left(R_i\right) + \lambda_x R_i, \tag{19}$$

and $\left(R_i, D\left(R_i\right)\right)$ is the operational rate-distortion pair associated to each block. To meet the allowed bit budget $\sum\limits_l R_l^* = \mathbf{R_x}$ a bisection search [30] for the correct multiplier λ_x is applied. We note that since the operational RD curves are known, this process is not computationally expensive. Once the optimal values R_l^* are evaluated, the encoded bit streams are truncated and transmitted.

VI. Rate-distortion modeling and optimization

We now study the correct distribution of the bit budget between encoding the layer contours and the DWT coefficients so that the overall RD performance is improved. The problem is defined as follows: given a target bit budget $\mathbf{R_t}$, the goal is to find an allocation which will minimize the output distortion:

$$\begin{aligned}\left[\mathbf{R_s}, \mathbf{R_x}\right] &= \text{argmin } \left\{D\left(\mathbf{R_s}, \mathbf{R_x}\right)\right\} \\ &\quad \text{such that } \mathbf{R_s} + \mathbf{R_x} \leq \mathbf{R_t}, \end{aligned} \tag{20}$$

where $\mathbf{R_s}$ and $\mathbf{R_x}$ are the number of bits allocated to the layer contours and texture, respectively and the distortion is measured in terms of the sum of squared differences (SSD) between the input and the output.

Note that the distortion $D\left(\mathbf{R_s}, \mathbf{R_x}\right)$ is jointly dependent on the contour and texture encoding. A lossy contour encoding modifies the layer segmentation, which, in turn, changes the disparity parameters used in the texture coding. Using the modified disparity parameter leads to an inefficient transform across the viewpoints. To solve this optimization problem, firstly, the upper bound of the distortion due to the lossy encoding of the layer contours is evaluated. Then, the total distortion is approximated as the sum of the distortion due to encoding the texture and the upper bound of the distortion of the layer contours.

In the following, we derive the models of the RD relations for the layer contours and texture encoding. Then, the constrained optimization problem is solved using Lagrangian multipliers. It is shown that a closed-form solution can be obtained when the layer contours are piecewise linear. For that reason and in order to obtain the rate allocation for arbitrary (non-piecewise linear) contours, the theoretical analysis is combined with an empirical approach, as discussed in Section VI-D.

A. RD Texture Modeling

To model the texture, we assume the obtained transform coefficients are independent Gaussian variables with the RD relation [33]:

$$D\left(R\right) = \sigma^2 2^{-2R}, \tag{21}$$

where σ^2 is the variance. Therefore, the total distortion due to encoding the texture is modeled by:

$$D_x\left(\mathbf{R_x}\right) = \sum_{j=1}^{L}\sum_{i=1}^{K_j} C_j N_{ij} \sigma_{ij}^2 2^{-2R_x^{ij}}, \tag{22}$$

where R_x^{ij} is the rate allocated to each transform coefficient, σ_{ij}^2 is the variance of the coefficients and N_{ij} is the number of transform coefficients. The subscript ij corresponds to the parameter of the *i*-th subband in the *j*-th layer. L denotes the total number of layers and K_j is the number of subbands in the *j*-th layer. The scaling factor $0 \leq C_j \leq 1$ is introduced in order to model occlusions. Recall that,

$$\mathbf{R_x} = \sum_{j=1}^{L}\sum_{i=1}^{K_j} N_{ij} R_x^{ij}, \tag{23}$$

is the total bit budget allocated to the texture. Note also that the parameters C_j and σ_{ij}^2 can be estimated using the original data and transform coefficients, respectively.

B. RD Layer Contour Modeling

The operational RD function of the layer contours is modeled by computing the upper bound of the distortion for a given bit budget.

To obtain the RD bound, first, recall that the contour of a layer at each image viewpoint is constant unless occluded. Using this property, we upper bound the distortion on a 2D image and scale it to evaluate the distortion in a 3D layer.

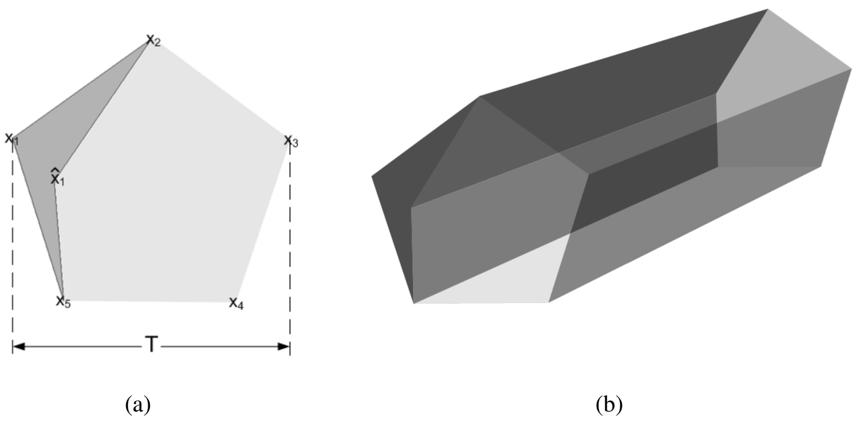


Fig. 9. (a) 2D Slice of a layer with a piecewise constant segmentation. Here, x represents the original vertices and $\hat{x}$ the location of a quantized vertex. The shaded region denotes the error due to quantizing the vertex x_1. (b) Shaded region outlines the error in a 3D layer, which is obtained by propagating the error from the first viewpoint.

Consider a 2D slice of a layer $\mathcal{L}_1$ as shown in Fig. 9(a). Assume that the contour is piecewise linear and has a fixed number of vertices (also known as a polygon model). The distortion can be upper bounded by quantizing the locations of the vertices:

$$e^2 \leq T^2 V \zeta^2 2^{-\frac{R_s}{2V}}, \tag{24}$$

where V is the number of vertices, T is the size of the bounding image, $\zeta = \max\{\mathcal{L}_1\} \leq 255$ is the maximal amplitude of the texture and R_s is the number of bits allocated to encoding the contour. The derivation of this bound is shown in Appendix A. As illustrated in Fig. 9(b), the error which is denoted by the shaded region can be scaled by the total number of images to compute the distortion in a 3D layer. The total distortion due to encoding L layers can therefore be upper bounded by:

$$D_s\left(\mathbf{R_s}\right) \leq \sum_{j=1}^{L} T_j^2 V_j M \zeta_j^2 C_j 2^{-\frac{R_s^j}{2V_j}}, \tag{25}$$

where M is the total number of images and the subscript j denotes the parameter corresponding to the j-th layer. Furthermore,

$$\mathbf{R_s} = \sum_{j=1}^{L} R_s^j, \tag{26}$$

is the total encoding rate of the layer contours.

Note that the model in (25) is evaluated by upper bounding the number of coefficients affected by the quantization of the layer contours. During coding, these pixels are classified to an incorrect layer and therefore create high-pass transform coefficients. In turn, this will add to the texture distortion when the coefficients are quantized. We model this distortion by scaling the number of pixels by ζ_j, which, in the worst case scenario, is the maximal value of the texture pixels equal to 255. At high bit rate, however, observe that the rate allocation between the texture and the layer contours is independent of the scaling ζ_j and only determined by the exponent of the RD functions [33].

Note also that the analyzed upper bound of the distortion in (25) does not rely on the quadtree algorithm used in Section V-A. However, as shown in [27], the RD performance is accurately modeled by our analysis, especially at high bit rates. In the continuation, the derived RD relations are used to find the optimal bit budget distribution between the layer contours and texture encoding.

C. RD Optimization - piecewise linear layer contours

We use Lagrangian multipliers [34] to find the bit rate distribution between the layer contours and the texture. Consider approximating the total distortion as the summation of the texture (22) and layer contours distortion (25):

$$\begin{aligned} D\left(\mathbf{R_x},\mathbf{R_s}\right) &\sim D_x\left(\mathbf{R_x}\right)+D_s\left(\mathbf{R_s}\right) \\ &= \sum_{j=1}^{L} C_j \left(\sum_{i=1}^{K_j} N_{ij}\sigma_{ij}^2 2^{-2R_x^{ij}} + T_j^2 V_j M \zeta_j^2 2^{-\frac{R_s^j}{2V_j}}\right), \end{aligned} \tag{27}$$

where $\mathbf{R_x}$ and $\mathbf{R_s}$ are defined in (23) and (26), respectively. For a given bit budget $\mathbf{R_t}$, the encoding rate must satisfy the following inequality:

$$\mathbf{R_x}+\mathbf{R_s}\leq \mathbf{R_t}. \tag{28}$$

The above problem can be solved using the reverse water-filling algorithm [33] based on Lagrangian multipliers. Assuming a high rate analysis, which implies that all the rates are positive, we obtain the following solutions:

$$\mathbf{R_x}=\sum_{j=1}^{L}\sum_{i=1}^{K_j} N_{ij}\left(\frac{1}{2}\log_2\left[2\ln\left(2\right)C_j\sigma_{ij}^2\right]+\frac{\alpha}{2}\right) \tag{29}$$

and

$$\mathbf{R_s}=\sum_{j=1}^{L}\left(2V_j\log_2\left[\frac{\ln\left(2\right)C_j M T_j^2\zeta_j^2}{2}\right]+2V_j\alpha\right), \tag{30}$$

where the constant α is defined by:

$$\alpha=\left(\sum_{j=1}^{L}\left[\sum_{i=1}^{K_j}\frac{N_{ij}}{2}+2V_j\right]\right)^{-1}\left(\mathbf{R_t}-\sum_{j=1}^{L}\left[\sum_{i=1}^{K_j}\frac{N_{ij}}{2}\log_2\left[2\ln\left(2\right)C_j\sigma_{ij}^2\right]+2V_j\log_2\frac{\ln\left(2\right)C_j M T_j^2\zeta_j^2}{2}\right]\right).$$

The derivation of these rate distribution equations is shown in Appendix B.

In the above analysis and RD optimization, we assume the layer contours are piecewise linear. The closed-form solutions in (29) and (30), therefore, depend on the number of vertices. Next, we generalize the rate allocation for encoding arbitrary layer contours.

D. RD optimization - arbitrary layer contours

The bit allocation strategy in (29) and (30) requires the knowledge of the number of vertices V_j in each contour. Given a set of arbitrary contours (non-piecewise linear) this parameter also depends on the target bit budget. For example, at low rates only a coarse approximation should be used and this corresponds to a small number of vertices, whereas at high rates an accurate reconstruction is required.

To find the correct allocation for non-piecewise linear contours we use the following approach: first we obtain a number of piecewise linear approximations of the layer contours for different error margins. We illustrate two versions of the resulting approximations in Fig. 10(a) and (b). These approximations are obtained using a fast algorithm proposed in [35]. The approach initializes a vertex at each point on the contour, which are then iteratively removed until a maximum allowed distortion is achieved.

For each approximation we obtain a different V_j and use it to evaluate the closed-form rate allocation curves in (29) and (30). The resulting rate allocation curves are illustrated in Fig. 10(c). Here, one rate allocation curve is due to the number of vertices in Fig. 10(a) and the other to the V_j obtained from Fig. 10(b).

We also show in Fig. 10(d) the PSNR achieved by the complete compression algorithm when the approximations in Fig. 10(a) and Fig. 10(b) are used to infer the bit allocation. The figure shows that the bit allocation obtained using the coarse approximation leads to better overall performance at low rates, and vice versa at high rates. Therefore, the problem is to choose the right approximation and thus the right number of vertices given the total bit budget. To solve this, a small number of bit rates are tested and for each of these rates various approximations are considered. The corresponding bit allocations, obtained using (29) and (30), are encoded using the complete compression algorithm. The operational points with the least distortion are then retained and the intermediate bit budgets interpolated. This process is illustrated in Fig. 11. In this example (see Fig. 11(a)) three bit budgets are chosen (i.e., 0.03bpp, 0.07bpp and 0.11bpp), and, since there are two possible rate allocation curves, we end up with six operational points. For each point the complete compression algorithm is executed leading to six PSNR values $(D_1, D_3, \ldots, D_6)$ shown on the plot. We pick D_1 because $D_1 > D_2$, similarly D_3 and D_6 are chosen since $D_3 > D_4$ and $D_6 > D_5$. The complete bit allocation strategy is obtained by linear interpolation of the three rate allocations related to D_1, D_3 and D_6. This is shown in Fig. 11(b).

In addition to the vertices V_j, observe that the rate allocation equations also require the knowledge of the number of layers L. In our approach we directly use the number of layers extracted using the method outlined in Section IV. In theory, this number should also depend on the total bit budget. However, we

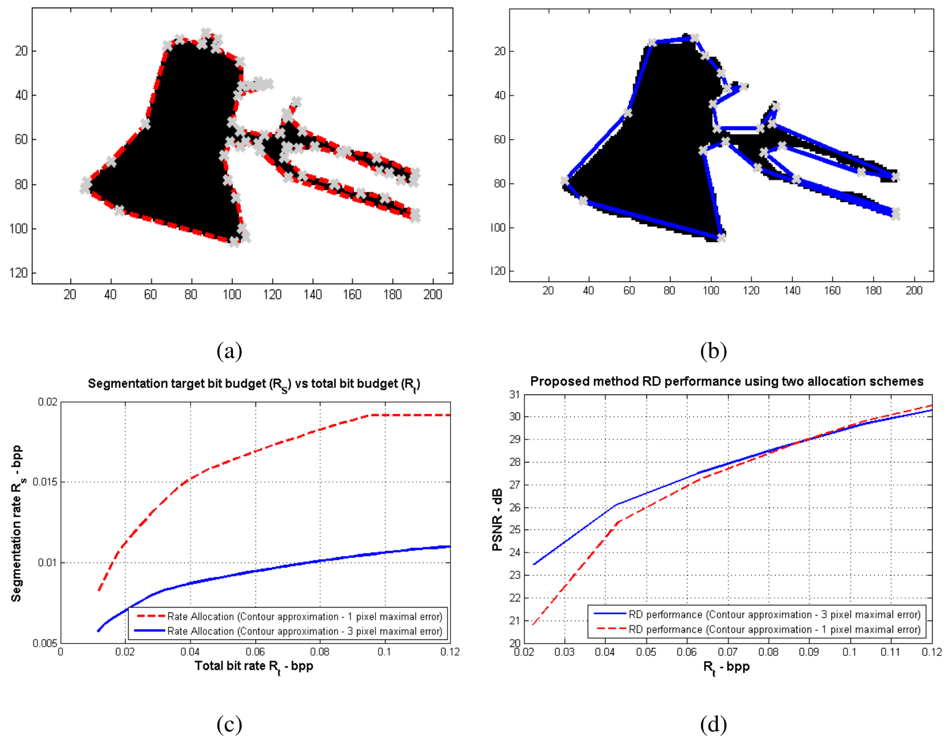
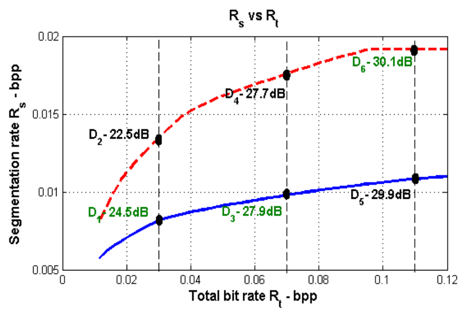


Fig. 10. Two piecewise linear approximations of a layer in the Tsukuba dataset [26] and the corresponding rate allocation curves for each model. (a) Approximation obtained using a maximal perpendicular error of 1 pixel. (b) Approximation obtained using a maximal perpendicular error of 3 pixels. (c) Two closed-form rate allocation curves corresponding to each approximation obtained using (29) and (30). Here, the x-axis corresponds to the total bit budget R_t and the y-axis to the rate allocated to the segmentation R_s. (d) The obtained PSNR curves using the rate allocation curves in (c).

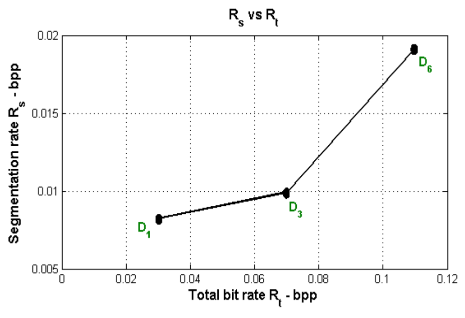
have evidence from [36] that at bit rates of interest the number of layers provided by the method is correct in the RD sense.

VII. Experimental Results

We now evaluate the performance of our compression codec. To comprehensively test the algorithm, four EPI datasets are used as shown in Fig. 12: Animal Farm $[232 \times 624 \times 16]$ from [20], Tsukuba $[284 \times 382 \times 4]$, Teddy $[368 \times 352 \times 4]$ and Doll $[544 \times 608 \times 4]$ (the last three from [26]). In addition, to show that the proposed method naturally scales to the additional viewpoint dimension, we also present the Tsukuba Light Field $[272 \times 368 \times 4 \times 4]$ results. Note that the datasets vary in geometric and texture complexity. Teddy has a wide range of disparities, whereas Tsukuba and Animal Farm can be well approximated using a small number of depth planes. In addition, the data vary in terms of the number of views and the spatial resolution. Without loss of generality, only the monochromatic components of



(a)



(b)

Fig. 11. Proposed approach to obtain the rate allocation curve for arbitrary layer contours. (a) A number of total bit budgets are chosen and the points which lie on the closed-form rate allocation curves are encoded. For each bit budget, the allocation which corresponds to the least distortion is retained. (b) The intermediate points are subsequently interpolated.

(a) Animal Farm $[232 \times 624 \times 16]$

(b) Teddy $[368 \times 352 \times 4]$

(c) Tsukuba EPI $[284 \times 382 \times 4]$

(d) Doll $[544 \times 608 \times 4]$

Fig. 12. Multiview image analysis datasets.

the images are encoded. The performance analysis is separated into two subsections. First, the accuracy of the proposed bit budget allocation strategy between the layer contours and the texture is presented. Then, the RD optimized codec is compared to the state-of-the-art H.264/AVC [16] and MVC [17] coding

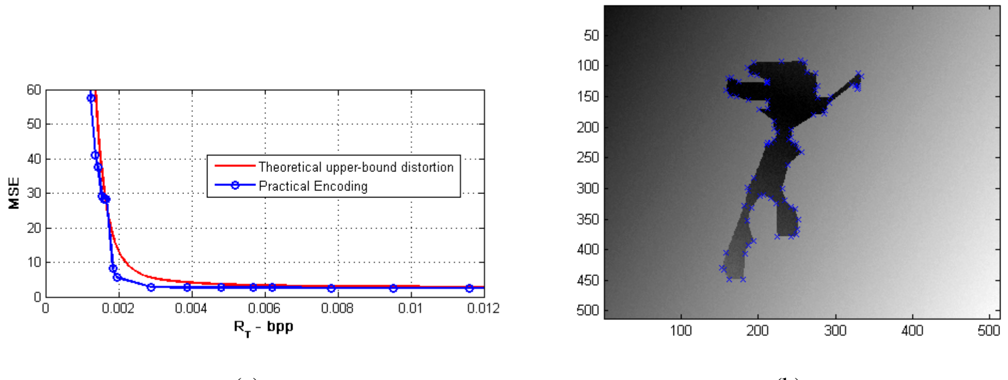


Fig. 13. (a) Experimental and theoretical RD performance when encoding a synthetic dataset $[512 \times 512 \times 4]$ with one layer and a background shown in (b). The layer contour is piecewise linear and contains 79 vertices. The vertices are labeled with blue crosses.

algorithms.

A. RD modeling evaluation

In this subsection, the performance of the RD optimization strategy in bit rate distribution between the layer contours and the texture is analyzed. First we verify distortion model in Section VI-C and then the proposed rate allocation scheme in Section VI-D for arbitrary layer contours.

Consider a synthetic dataset in Fig. 13(b) consisting of one layer and a background. The layer contour contains 79 vertices and the texture is linear with an additive Gaussian signal. We verify the RD bound in (27) by encoding the multiview images using the proposed algorithm, where the rate allocation to encode the layer contours and the texture is obtained using (30) and (29), respectively. The experimental and theoretical RD curves are shown in Fig. 13(a). These results show that the distortion model closely matches the experimental results.

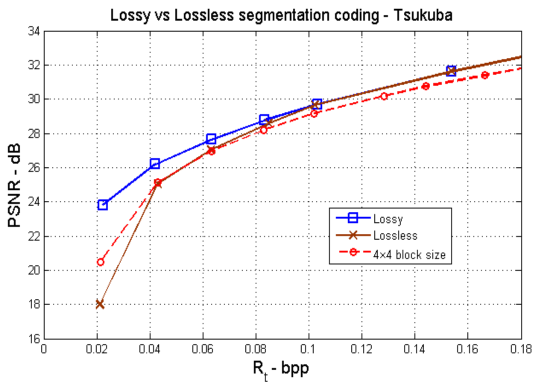
To verify the rate allocation strategy when encoding arbitrary (non-piecewise linear) layer contours we operate as follows. First we encode the dataset using the outlined approach. Referring to Section VI-D we use three piecewise approximations and five bit budgets to derive the rate allocation curves for each dataset. The overall RD performance is then compared to an exhaustive scheme where all possible combinations are considered. A RD comparison of the proposed approach and an exhaustive search is shown in Table I, where we show that except for low bit rates, our method matches the performance of the exhaustive search.

In addition we also show a comparison of proposed allocation strategy and the lossless scenario in

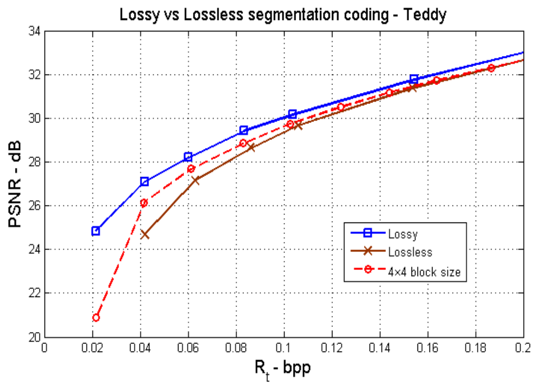
TABLE I

RD PERFORMANCE COMPARISON BETWEEN THE PROPOSED RATE ALLOCATION METHOD AND AN EXHAUSTIVE SEARCH.

Dataset	**Tsukuba**				**Teddy**			
Rate/bpp	**0.05**	**0.1**	**0.2**	**0.3**	**0.056**	**0.1**	**0.2**	**0.3**
Proposed method/dB	26.65	29.632	33.17	35.32	27.93	29.98	33.01	34.83
Exhaustive search/dB	26.73	29.634	33.17	35.32	28.05	30.05	33.01	34.85



(a) Tsukuba EPI



(b) Teddy

Fig. 14. A comparison of the proposed allocation strategy with the lossless and block-based coding of the layer contours. In the lossless case the layer contours are encoded using the Freeman algorithm. In the block-based approach the layer contours are encoded using the quadtree prune-join scheme, where the minimal block size is set to 4×4 pixels and no intermediate tiles are used. This representation is losslessly encoded over all the bit rates. In (a) the Tsukuba layer-based representation is simple and requires a small number of bits. Therefore, beyond a bit rate of 0.12bpp the proposed method encodes the layer contours in the lossless modality and the two curves converge.

Fig. 14. In the lossless case, the segmentation is encoded using the modified Freeman algorithm at all bit rates. Observe that at a PSNR of 29dB the proposed approach provides a bit rate saving of 20% when encoding Teddy. We note that due to the layer contours being more irregular and the representation having a larger number of layers, the improvement of the proposed approach is higher on Teddy than Tsukuba. We also compare the proposed approach to a block-based coding strategy. In this case the layer contours are encoded using the quadtree prune-join scheme where the smallest block size is 4×4 pixels and can only be represented using a 0 or 1 tile (with no intermediate tiles). The representation is then losslessly encoded over all of the bit rates. This type of approach is simpler but less effective than our solution since there is no optimization between the texture and the layer contours.

In conclusion, the presented results in Fig. 14 and Table I support the proposed rate allocation strategy.

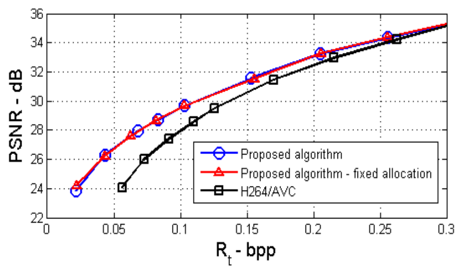
We note that although only Teddy and Tsukuba results have been shown, Doll and Animal Farm datasets show a similar pattern.

B. Compression Comparison

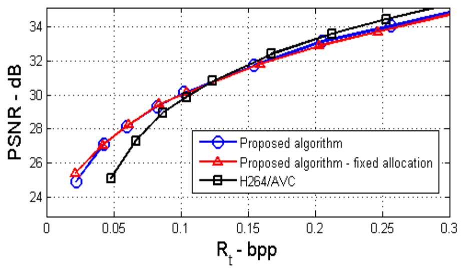
In this section, the performance of the proposed algorithm is compared to the state-of-the-art H.264/AVC. To encode the data, the multiview images are treated as a set of video frames along the temporal dimension and they are compressed using the High Profile. We use the 2 pass encoding mode with the hexagonal search and quarter-pixel precision motion estimation settings. All macro-block partitions are considered and the trellis RD optimization is applied during the final encoding of a macro block. In addition the early SKIP and coefficient thresholding on P-frames settings have been disabled. We specify the number of B-frames used in each dataset in Fig. 15. A quantitative comparison of the proposed method and H.264/AVC is shown in Fig. 15. In addition to the proposed rate allocation strategy, we include the results of a fixed allocation, where a constant 10% of the total bit rate is allocated to encode the layer contours. Note that that the fixed allocation strategy has a lower complexity and is an approximation to the proposed method. In this scheme, the number of bits used to encode the segmentation is set to $R_s = 0.1R_t$, and this is an approximation of a typical curve obtained in Fig.11(b).

The results show that at low bit rates the proposed codec achieves a better RD performance on all datasets. The gains are different and they are influenced by the accuracy of the layer segmentation algorithm outlined in Section IV. For example, in the Tsukuba EPI dataset, our proposed algorithm achieves an improvement of almost 3dB at a bit rate of $R_t = 0.05$bpp, whereas, in case of Teddy, this gain is around 2dB. Moreover, observe that the performance of the alternative allocation strategy is similar to the proposed approach, with a maximal PSNR reduction of 0.13dB and 0.2dB in case of Tsukuba EPI and Teddy, respectively. To show that the proposed algorithm naturally scales to an additional viewpoint dimension, we present the Tsukuba Light Field results in Fig. 15(e). In this case, to obtain a more fair evaluation, we compare our results to the H.264/MVC implementation specialized for multiview video. In comparison our method attains a bit rate saving of 40% at a PSNR of 30dB. The qualitative results are shown in Fig. 16.

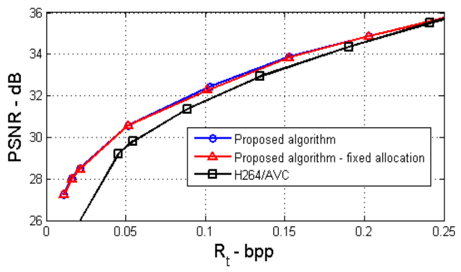
Regarding high rate encoding, it is shown that the proposed algorithm is competitive with H.264/AVC and MVC. Analyzing Animal Farm and Tsukuba, the proposed codec achieves better RD performance over the complete range of bit rates. In these cases, the layer based representation captures the 3D scene efficiently and the segmentation is accurate. Indeed, the data fits our designed representation and seems to truly consist of layers. Additionally, there are a large number of occluded regions, which our algorithm



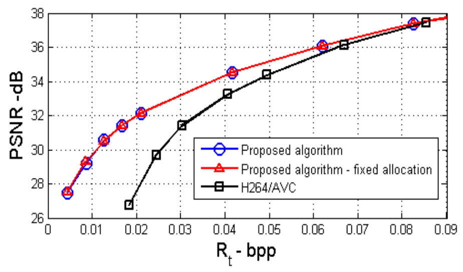
(a) Tsukuba EPI



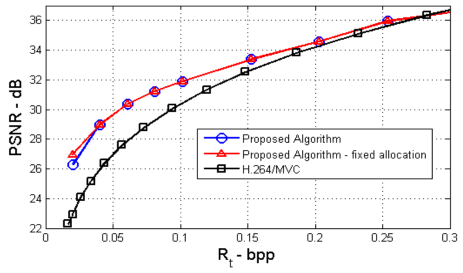
(b) Teddy



(c) Doll



(d) Animal Farm

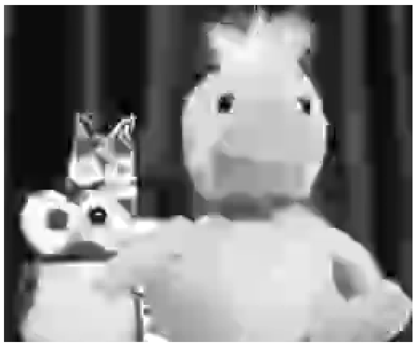


(e) Tsukuba Light Field

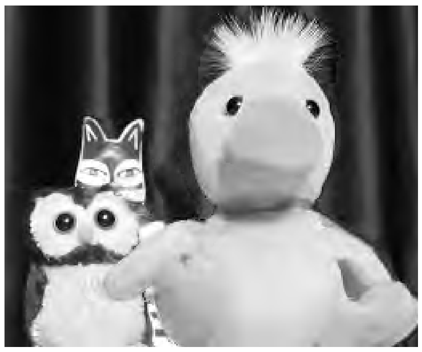
Fig. 15. Quantitative comparison of the proposed algorithm with H.264/AVC and MVC. In (a), (b) and (c) H.264/AVC encodes the dataset with one I-frame, two P-frames, and one B-frame. In (d) one I-frame, eight P-frames and seven B-frames are used. Regarding the proposed method, we apply a maximal number of decompositions in the inter-view DWT.

deals with efficiently. On the other hand, H.264/AVC attains better results when encoding Teddy at higher rates. In this case, the 3D structure of the scene is complicated with a large number of disparities which are not captured effectively by the layer-based representation. Hence, the segmentation errors create high-pass transform coefficients, which degrade the performance of our method.

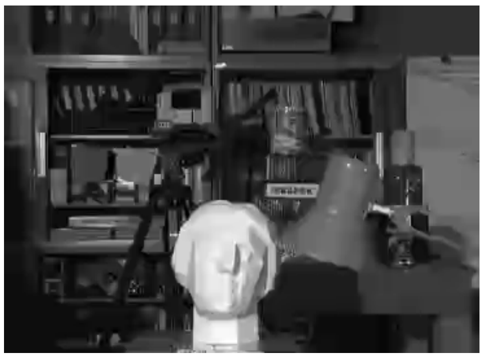
Regarding the complexity of the proposed method, we note that the most computationally intensive stage is the layer-based extraction which uses a level-set method to evolve the layer contours. This process, however, can be implemented offline with the obtained layer contours being used to compress the data for different bit budgets. The complexity of the encoding method itself is determined by the



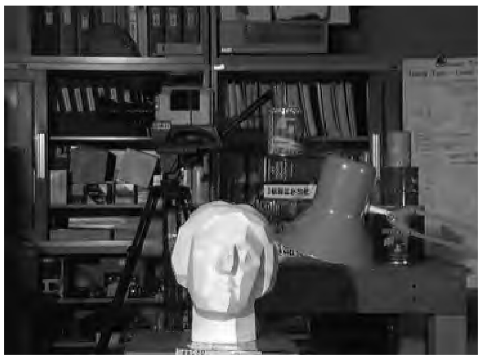
(a)



(b)



(c)



(d)

Fig. 16. Qualitative comparison of the proposed algorithm with H.264/AVC and MVC. (a) Animal Farm encoded using H.264/AVC at 0.024bpp (PSNR 28.93dB). (b) Animal Farm encoded using the proposed algorithm at 0.021bpp (PSNR 32.14dB). (c) Tsukuba Light Field encoded using H.264/MVC at 0.056bpp (PSNR 27.53dB). (d) Tsukuba Light Field encoded using proposed algorithm at 0.052bpp (PSNR 29.77dB).

separable wavelet transform which is *O(N)*, where *N* is the total number of pixels in the dataset.

In conclusion, the presented results have shown that the proposed algorithm achieves an improved RD performance at low rates and is competitive at high rates in comparison to H.264/AVC and MVC.

VIII. Conclusion

We presented a novel compression algorithm for multiview images. A fundamental component of the method is the layer-based representation, which partitions the data into layers modeled by a constant depth plane. Initially, the contours which outline each layer are encoded in a lossy or lossless modality. Then, each layer is efficiently transformed using a separable 3D DWT, first using a 1D disparity compensated DWT across the views, followed by a 2D SA DWT along the spatial dimensions. Finally, the transform coefficients are quantized and entropy coded using a context adaptive arithmetic coder. To improve the RD performance, we proposed two bit rate allocation schemes which correctly distribute the rate between the transform coefficients and the layer contours. Experimental results have shown that the proposed algorithm achieves improved RD performance in comparison to H.264/AVC and MVC on several datasets and for a wide range of bit rates. Future research includes making the algorithm more robust to segmentation errors and extending the codec to encode multiview videos.

ACKNOWLEDGMENT

We would like to thank Jesse Berent for many fruitful discussions and for providing the Animal Farm dataset. We would also like to thank the anonymous reviewers whose constructive comments led to an improved manuscript.

APPENDIX A
PROOF OF (24)

Consider a polygon, bounded by a box of size T having V vertices. The distortion bound is derived by quantizing the vertex locations for a given bit budget of R bits, where each vertex is, therefore, allocated $\frac{R}{V}$ bits. This corresponds to quantizing the x and y locations with a step-size $\Delta = T2^{-\frac{R}{2V}}$. The maximal quantization error ϵ along each dimension can be upper bounded by:

$$\begin{aligned} \epsilon &\leq \frac{\Delta}{2} \\ &= \frac{T}{2}2^{-\frac{R}{2V}}. \end{aligned} \tag{31}$$

Therefore, using Pythagoras, the maximal distance to the original vertex can be upper bounded by $\frac{T}{\sqrt{2}}2^{-\frac{R}{2V}}$. Since each side of the polygon is bounded by $T\sqrt{2}$, the number of pixels affected by quantizing the vertices is bounded by $T^2V2^{-\frac{R}{2V}}$.

Denote with ζ the maximal amplitude of the texture in the polygon. Assuming the texture is set to zero in the quantization error regions, the total distortion can be upper bounded by:

$$e^2 \leq \zeta^2T^2V2^{-\frac{R}{2V}}, \tag{32}$$

which coincides with (24).

APPENDIX B
PROOF OF (29) AND (30)

Here, we prove the optimum rate distribution between segmentation and texture in (29) and (30). The cost function to be minimized is defined by:

$$\begin{aligned} D\left(\mathbf{R_x}, \mathbf{R_s}\right) &\sim D_x\left(\mathbf{R_x}\right) + D_s\left(\mathbf{R_s}\right) \\ &= \sum_{j=1}^{L} C_j \left(\sum_{i=1}^{K_j} N_{ij}\sigma_{ij}^2 2^{-2R_x^{ij}} + MT_j^2 V_j \zeta_j^2 2^{-\frac{R_s^j}{2V_j}} \right), \end{aligned} \tag{33}$$

where $\mathbf{R_x}$ and $\mathbf{R_s}$ are defined as in (23) and (26), respectively. The minimization is subject to a bit rate constraint:

$$\mathbf{R_t} \geq \mathbf{R_x} + \mathbf{R_s} \tag{34}$$

$$= \sum_{j=1}^{L}\sum_{i=1}^{K_j} N_{ij} R_x^{ij} + \sum_{j=1}^{L} R_s^j. \tag{35}$$

The constrained optimization can be transformed into an unconstrained one using Lagrangian multipliers, where the new objective function to be minimized is given by:

$$J\left(\mathbf{R_x}, \mathbf{R_s}, \lambda\right) = \sum_{j=1}^{L} C_j \left(\sum_{i=1}^{K_j} N_{ij}\sigma_{ij}^2 2^{-2R_x^{ij}} + MT_j^2 V_j \zeta_j^2 2^{-\frac{R_s^j}{2V_j}} \right) + \lambda \left(\sum_{j=1}^{L}\sum_{i=1}^{K_j} N_{ij} R_x^{ij} + \sum_{j=1}^{L} R_s^j - \mathbf{R_t} \right). \tag{36}$$

The minimum is obtained by partially differentiating the cost function with respect to the free variables and setting the derivative to zero:

$$\frac{\partial J}{\partial R_x^{ij}} = -2\ln(2)\, C_j N_{ij} \sigma_{ij}^2 2^{-2R_x^{ij}} + \lambda N_{ij} = 0, \tag{37}$$

$$\frac{\partial J}{\partial R_s^{j}} = -\frac{\ln(2)}{2} C_j MT_j^2 \zeta_j^2 2^{\frac{-R_s^j}{2V_j}} + \lambda = 0. \tag{38}$$

Therefore,

$$R_x^{ij} = \frac{1}{2}\log_2\left[2\ln(2)\, C_j \sigma_{ij}^2\right] - \frac{1}{2}\log_2 \lambda, \tag{39}$$

$$R_s^{j} = 2V_j \log_2\left[\frac{\ln(2)\, C_j MT_j^2 \zeta_j^2}{2}\right] - 2V_j \log_2 \lambda. \tag{40}$$

Consider that the total Lagrangian cost is minimized when the bit rate constraint is active:

$$\mathbf{R_t} = \sum_{j=1}^{L}\sum_{i=1}^{K_j} N_{ij} R_x^{ij} + \sum_{j=1}^{L} R_s^j. \tag{41}$$

Substituting (39) and (40) into the above equation and solving for $\log_2 \lambda$ we obtain the following:

$$\log_2 \lambda = -\left(\sum_{j=1}^{L} \left[\sum_{i=1}^{K_j} \frac{N_{ij}}{2} + 2V_j \right] \right)^{-1} \left(\mathbf{R_t} - \sum_{j=1}^{L} \left[\sum_{i=1}^{K_j} \frac{N_{ij}}{2} \log_2\left[2\ln(2)\, C_j \sigma_{ij}^2\right] + 2V_j \log_2 \frac{\ln(2)\, C_j MT_j^2 \zeta_j^2}{2} \right] \right). \tag{42}$$

We denote with $\alpha = -\log_2 \lambda$. Substituting (39) and (40) into (23) and (26), we obtain the following solutions:

$$\mathbf{R_x} = \sum_{j=1}^{L} \sum_{i=1}^{K_j} N_{ij} \left(\frac{1}{2} \log_2 \left[2 \ln(2)\, C_j \sigma_{ij}^2 \right] + \frac{\alpha}{2} \right), \tag{43}$$

and

$$\mathbf{R_s} = \sum_{j=1}^{L} \left(2V_j \log_2 \left[\frac{\ln(2)\, C_j M T_j^2 \zeta_j^2}{2} \right] + 2V_j \alpha \right), \tag{44}$$

which coincides with the layer contour and texture rates in (29) and (30), respectively.

REFERENCES

[1] A. Kubota, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang, "Multiview imaging and 3DTV," *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 10–21, Nov. 2007.

[2] C. Zhang and T. Chen, "A survey on image-based rendering–representation, sampling and compression," *Signal Processing: Image Communication*, vol. 19, no. 1, pp. 1–28, 2004.

[3] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of Computer Graphics (SIGGRAPH)*, Aug. 1996, pp. 31–42.

[4] M. H. Kiu, X. S. Du, R. J. Moorhead, D. C. Banks, and R. Machiraju, "Two dimensional sequence compression using MPEG," *in Visual Communication and Image Processing (VCIP)*, pp. 914–921, Jan. 1998.

[5] C. Zhang and J. Li, "Compression of lumigraph with multiple reference frame (MRF) prediction and just-in-time rendering," in *Proceedings of the IEEE Data Compression Conference (DCC)*, Mar. 2000, pp. 253–262.

[6] M. Magnor and B. Girod, "Data compression for light-field rendering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 3, pp. 338–343, Apr. 2000.

[7] B. Girod, C.-L. Chang, P. Ramanathan, and X. Zhu, "Light field compression using disparity-compensated lifting," *IEEE Transactions on Image Processing*, vol. 15, no. 4, pp. 793–806, Apr. 2006.

[8] A. Secker and D. Taubman, "Lifting-based invertible motion adaptive transform (LIMAT) framework for highly scalable video compression," *IEEE Transactions on Image Processing*, vol. 12, no. 12, pp. 1530–1542, Dec. 2003.

[9] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *J. Fourier Anal. Appl*, vol. 4, pp. 247–269, 1998.

[10] M. Magnor and B. Girod, "Hierarchical coding of light fields with disparity maps," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, vol. 3, Oct. 1999, pp. 334–338.

[11] M. Magnor and P. Eisert, "Model-aided coding of multi-viewpoint image data," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, vol. 2, Sep. 2000, pp. 919–922.

[12] M. Magnor and B. Girod, "Model-based coding of multi-viewpoint imagery," in *Proceedings SPIE Visual Communications and Image Processing (VCIP)*, vol. 1, Jun. 2000, pp. 14–22.

[13] X. San, H. Cai, J.-G. Lou, and J. Li, "Multiview image coding based on geometric prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 11, pp. 1536 –1548, Nov. 2007.

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

[15] I. Daribo, C. Tillier, and B. Pesquet-Popescu, "Motion vector sharing and bitrate allocation for 3D video-plus-depth coding," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, pp. 1–13, 2009.

[16] "H.264/AVC Video Coding Algorithm." [Online]. Available: http://x264.nl/

[17] "H.264/MVC Multiview Video Coding Algorithm." [Online]. Available: CVS - garcon.ient.rwth-aachen.de

[18] E. H. Adelson and J. R. Bergen, "The plenoptic function and the elements of early vision," in *Computational Models of Visual Processing*. MIT Press, 1991, pp. 3–20.

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

[20] J. Berent and P. L. Dragotti, "Plenoptic manifolds: Exploiting structure and coherence in multiview images," *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 34–44, Nov. 2007.

[21] ——, "Segmentation of epipolar-plane image volumes with occlusion and disocclusion competition," in *Proceedings of IEEE 8th Workshop on Multimedia Signal Processing (MMSP)*, Oct. 2006, pp. 182–185.

[22] S. Jehan-Besson, M. Barlaud, and G. Aubert, "DREAM2S: Deformable regions driven by an eulerian accurate minimization method for image and video segmentation," *International Journal of Computer Vision*, vol. 53, no. 1, pp. 45–70, Jun. 2003.

[23] J. Berent, "Coherent multi-dimensional segmentation of multi-view images using a variational framework and applications to image based rendering," *PhD Thesis*, 2008. [Online]. Available: http://www.commsp.ee.ic.ac.uk/∼pld/group

[24] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, Jan. 1988.

[25] 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)*. Springer-Verlag, 2002, pp. 82–96.

[26] D. Scharstein and R. Szeliski. (vision.middlebury.edu/stereo/) Middlebury data sets.

[27] 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.

[28] H. Freeman, "On the encoding of arbitrary geometric configurations," *IEEE Transactions on Electronic Computers*, vol. EC-10, no. 2, pp. 260–268, Jun. 1961.

[29] Y. Liu and B. Zalik, "An efficient chain code with huffman coding," in *Pattern Recognition*, vol. 38, no. 4, Apr. 2005, pp. 553–557.

[30] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 36, no. 9, Sep. 1988, pp. 1445–1453.

[31] S. Li and W. Li, "Shape-adaptive discrete wavelet transforms for arbitrarily shaped visual object coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 5, pp. 725–743, Aug. 2000.

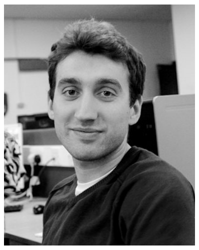
[32] D. Taubman, "High performance scalable image compression with EBCOT," *Image Processing, IEEE Transactions on*, vol. 9, no. 7, pp. 1158–1170, Jul. 2000.

[33] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

[34] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[35] M. Hötter, "Object-oriented analysis-synthesis coding based on moving two-dimensional objects," *Signal Processing: Image Communication*, vol. 2, no. 4, pp. 409–428, 1990.

[36] 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.



Andriy Gelman received his PhD and MEng degrees in Electrical and Electronic Engineering from Imperial College London, UK in 2012 and 2008, respectively. He is now a research scientist with Schlumberger, Stonehouse, UK specialising in coding and compression. His research interests include multiview image compression, interactive communication, image-based rendering and image de-noising.

Pier Luigi Dragotti (M02SM11) received the Laurea degree (summa cum laude) in electrical engineering from the University Federico II, Naples, Italy, in 1997; the M.S. degree in communications systems from the Swiss Federal Institute of Technology of Lausanne (EPFL), Switzerland, in 1998; and Ph.D. degree from EPFL, in April 2002. He is currently a Reader (Associate Professor) with the Electrical and Electronic Engineering Department, Imperial College, London, U.K. In 1996, he was a visiting student with Stanford University, Stanford, CA; from July to October 2000, he was a summer researcher with the Mathematics of Communications Department at Bell Labs, Lucent Technologies, NJ and, finally from January to May 2011 he was a visiting scientist at Massachusetts Institute of Technology (MIT) . His research interests include wavelet and sampling theory, image compression, image super-resolution, and image-based rendering. Dr. Dragotti has been an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING and is currently a member of the IEEE Image, Video and MultiDimensional Signal Processing (IVMSP) Technical Committee.

Vladan Velisavljević (M'06, SM'12) received the B.Sc. and M.Sc. (Magister) degree from the University of Belgrade, Serbia, in 1998 and 2000, respectively, and the Master and Ph.D. degree from EPFL, Lausanne, Switzerland, in 2001 and 2005.

From 1999 to 2000, he was a member of academic staff at the University of Belgrade. In 2000, he joined the Audiovisual Communications Laboratory (LCAV) at EPFL as teaching and research assistant, where he was working on his Ph.D. degree in the field of image processing. In 2003, he was a visiting student at Imperial College London. From 2006 to 2011, Dr. Velisavljevic was a Senior Research Scientist at Deutsche Telekom Laboratories, Berlin, Germany. Since October 2011, he is Senior Lecturer (Associate Professor) at Bedfordshire University, Luton, UK.

He has co-authored more than 40 research papers published in peer-reviewed journals and conference proceedings and he has been awarded or filed 4 patents in the area of image and video processing. He co-organized a special session at IEEE ICIP-2011 on compression of high-dimensional media data for interactive navigation and he is a co- chair of the Multimedia Computing and Communications Symposium (MCC) at IEEE ICNC-2013. His research interests include image, video and multiview video compression and processing, wavelet theory, multiresolution signal processing and distributed image/video processing.