A graph learning approach for light field image compression

Irene Viola*[a], Hermina Petric Maretic*[b], Pascal Frossard[b], and Touradj Ebrahimi[a]

[a]Multimedia Signal Processing Group (MMSPG), EPFL, Switzerland
[b]Signal Processing Laboratory (LTS4), EPFL, Switzerland

ABSTRACT

In recent years, light field imaging has attracted the attention of the academic and industrial communities thanks to its enhanced rendering capabilities that allow to visualise contents in a more immersive and interactive way. However, those enhanced capabilities come at the cost of a considerable increase in content size when compared to traditional image and video applications. Thus, advanced compression schemes are needed to efficiently reduce the volume of data for storage and delivery of light field content. In this paper, we introduce a novel method for compression of light field images. The proposed solution is based on a graph learning approach to estimate the disparity among the views composing the light field. The graph is then used to reconstruct the entire light field from an arbitrary subset of encoded views. Experimental results show that our method is a promising alternative to current compression algorithms for light field images, with notable gains across all bitrates with respect to the state of the art.

Keywords: Light field compression, graph learning, view reconstruction, image coding

1. INTRODUCTION

The first formal definition of light field as a function describing the appearance of a scene was given in 1939 by Andreī Gershun.[1] One of the most common ways of representing the light field is through the *plenoptic function*, which describes the radiance along the light rays in a three-dimensional space with constant illumination, as introduced by Adelson and Bergen.[2] In particular, the plenoptic function $\mathcal{L}$ describes the intensity of the light rays passing through every point in space (V_x, V_y, V_z) at every angle (θ, ϕ), with wavelength λ, in time t:

$$\mathcal{L} = \mathcal{L}(\theta, \phi, \lambda, t, V_x, V_y, V_z). \tag{1}$$

The function can be simplified if we restrict ourselves to a 3D region, free of occlusions, at a single time instance. Considering that radiance along rays remains constant in free space, the 7D plenoptic function can be simplified into a 4D light field function.[3] Notably, the 4D function can be parametrized as rays intersecting two planes: the uv plane, which describes the position of the rays in the aperture (object) plane, and the xy plane, which describes the position of the rays in the image plane.

$$\mathcal{L} = \mathcal{L}(u, v, x, y). \tag{2}$$

By extension, a digital 4D light field is obtained by sampling the 4D light field function defined in Equation 2. The digital 4D light field can be considered as a collection of perspective images (views) of the 3D scene projected on the xy plane, as observed from a position on the uv plane.

Light field photography is a promising technology to visualize and interact with three-dimensional scenes in a more realistic and immersive way. However, the increased volume of data generated in the acquisition poses new challenges in terms of efficient storage and transmission. In particular, new compression solutions that exploit the inherent redundancy of light field data need to be designed and implemented to effectively minimize the size of the data without compromising the perceptual visual quality. In 2014, the JPEG standardization committee

*Both authors contributed equally to this work.

Further author information: (Send correspondence to authors)
E-mail: firstname.lastname@epfl.ch

launched a new initiative called JPEG Pleno, whose goal is to create a standard framework for efficient storage and delivery of plenoptic contents, including light fields, point clouds, and holograms. In particular, JPEG Pleno aims at finding the minimum number of representation models for these types of content, which, when necessary, can also offer interoperability with existing standards, such as legacy JPEG and JPEG 2000 formats. Since then, JPEG committee has been actively pursuing the definition of a new standard representation and compression algorithm for light field images. In 2017, its efforts led to a Call for Proposals (CfP) for light field coding solutions, launched jointly with ICIP 2017 Grand Challenge on light field image coding.[4] The majority of the collected solutions focused on compressing light field images which were acquired using plenoptic (lenslet) cameras. Lenslet cameras rely on the additional optical element placed in front of the camera sensor to acquire both spatial and angular information. The resulting raw data, commonly referred to as lenslet image, presents a characteristic honeycomb structure, which is then rearranged to form the 4D light field structure.

In recent years, several solutions have been proposed to improve the coding efficiency of the lenslet image by exploiting the redundancy within its structure. Monteiro et al.[5] introduce a modified version of HEVC Intra profile which integrates Locally Linear Embedding (LLE) and Self Similarity (SS) to improve block estimation. More recently, Jin et al.[6] propose a macropixel-based intra prediction method which first applies image reshaping to align the macropixel structure to the HEVC coding unit grid, and then defines three prediction modes to improve the coding efficiency of the blocks.

Other solutions aim at improving the performance of existing video codecs by further exploiting the redundancies in the 4D LF structure of perspective views. Ahmad et al.[7] arrange the perspective views into a multiview structure that can be exploited by the corresponding extension of HEVC, namely MV-HEVC, and they propose a rate allocation scheme to optimize the performance by progressively assign the Quantization Parameters (QP). Jia et al.[8] propose a fully reversible transformation to 4D LF to create the perspective views, which are then optimally re-arranged and compressed using enhanced illumination compensation in JEM software*. They also implement adaptive filtering to optimally reconstruct the lenslet image. Zhao et al.[9] propose a novel compression scheme that encodes and transmits only part of the views using HEVC/H.265, while the non-encoded views are estimated as a linear combination of the already transmitted views. Similarly, the JPEG Pleno Verification Model (VM) software,[10,11] released in July 2018, uses disparity information to reconstruct the 4D light field from a predefined set of perspective views.

Meanwhile, graph learning methods have been attracting increasing interest in the field of graph signal processing. Among the first approaches, Dong et al.[12] consider a smooth signal model to infer the graph structure. Kalofolias[13] explores a similar model, adding an option to promote graph connectivity, and offering a computationally more efficient solution. Fracastoro et al.[14] propose a graph learning method for image coding, formulating a rate-distortion optimization problem that takes into account the cost of sending the graph.

Graph based methods for light field compression include the work of Maugey et al.[15] where graph based representations are used to describe multi view geometry. Similarly, Su et al.[16] adapt graph based representations to model colour and geometry. In their approach, the vertices of the graph correspond to each pixel in sub-aperture images, while the edges built from disparity information connect pairs of pixels across two images. Chao et al.[17] construct a graph between pixels through a Gaussian kernel, and use a graph lifting transform to compress light field images before demosaicking. In Yang et al.,[18] a fast graph filtering approach was proposed to improve block-based compressed light field image decoding.

In this work we combine the recent advances in graph learning to devise a new compression scheme for light field images that extensively exploits the redundancy in the light field structure to reconstruct the entire 4D light field from an arbitrarily chosen subset of perspective views. Graph learning techniques are used to estimate the similarities among neighboring views. As opposed to most other graph compression method, in this approach the graph is constructed such that each vertex represents one view. Edge weights relative to each pair of perspective images are learned from the data. The graph is then losslessly compressed and transmitted along with the selected perspective views. At the decoder side, an optimization problem is solved to optimally reconstruct the 4D light field. Results show the superiority of our method with respect to state-of-the-art solutions in light field compression.

*https://jvet.hhi.fraunhofer.de/

Our paper is organized as follows. An overview of graph signal processing fundamentals is given in Section 2. Our proposed approach is extensively described in Section 3, whereas the validating experiment is outlined in Section 4. Results are commented in Section 5, whilst Section 6 concludes the paper.

2. GRAPH SIGNAL PROCESSING PRELIMINARIES

In this section, we will introduce some basic notions in graph signal processing. For a more detailed description we refer interested readers to.[19] Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$ be an undirected, weighted graph with a set of m vertices $\mathcal{V}$, edges $\mathcal{E}$ and a weighted adjacency matrix W. Value W_{ij} equals 0 if there is no edge between i and j, while it designates the weight of that edge otherwise. We define a graph signal as a function $y : \mathcal{V} \to \mathbb{R}$, where $y(v)$ denotes the signal value on a vertex v. The graph Laplacian is defined as

$$L = D - W, \tag{3}$$

where D is a diagonal matrix containing node degrees. As a real symmetric matrix, the graph Laplacian has a complete set of orthonormal eigenvectors $\chi = \{\chi_0, \chi_1, ..., \chi_{m-1}\}$ with a corresponding set of non-negative eigenvalues. Furthermore, zero appears as an eigenvalue with multiplicity equal to the number of connected components of the graph, while the spectrum of the Laplacian matrix satisfies

$$\sigma(L) = \{0 = \lambda_0 \leq \lambda_1 \leq ... \leq \lambda_{m-1}\}. \tag{4}$$

This leads to the observation of graph signals in the graph spectral domain (as opposed to the vertex domain). Namely, we can define the graph Fourier transform $\hat{y}$ of a signal y at frequency λ_l as the expansion:

$$\hat{y}(\lambda_l) = \langle y, \chi_l \rangle = \sum_{i=1}^{m} y(i)\chi_l^*(i), \tag{5}$$

and the inverse graph Fourier transform as

$$y(i) = \sum_{l=0}^{m-1} \hat{y}(\lambda_l)\chi_l(i). \tag{6}$$

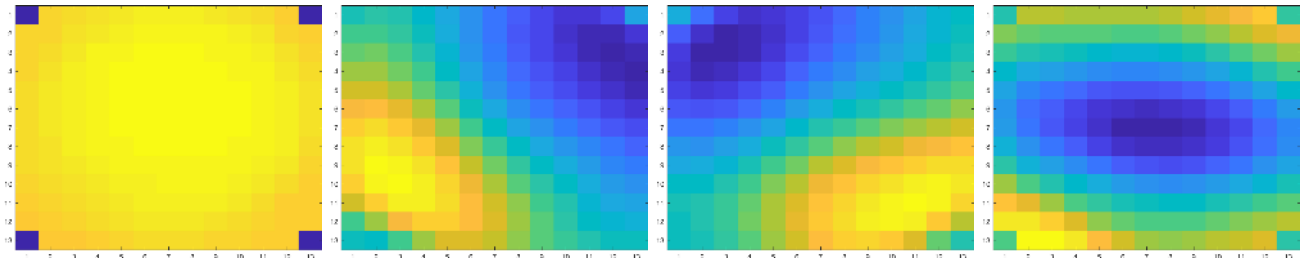
Here, the graph Laplacian eigenvectors form a Fourier basis and it straightforward to see that the corresponding eigenvalues carry a notion of frequency.

3. PROPOSED APPROACH

In this paper we propose a novel approach for light field image coding using graph learning. We explore extensive similarities between views by capturing them in a graph that models relationships between these views. We then select a subset of views to be compressed, whereas the remaining views will be recovered using the graph. This approach allows for better compression quality in sampled views, as more bits can be allocated to encode them. At the decoder side, the graph is used to recover the entire 4D light field from the subset of encoded views.

The intuition behind our approach stems from observing the presence of smoothness among neighboring images in a 4D light field structure. The idea is confirmed by simple PCA analysis of the signal, which shows a smooth, slowly transitioning behavior, further strengthening the suggestion of smoothness among neighboring views. Figure 1 shows this phenomena for *Bikes*, with each of the points in a PCA component representing one view. Graph signal processing is traditionally used to properly capture this smoothness on an irregular structure, defining notions equivalent to those seen in regular signal processing.[19]

In order to maximally exploit signal smoothness, we use a graph learning algorithm to obtain a structure our signal is most smooth on. We construct a graph with each vertex representing one view, and learn edges modelling relationships between corresponding views. This adaptive graph structure, as opposed to a simple fixed grid graph, ensures a much better signal representation at an acceptable cost. Graphs are convenient structures for compression as they encode a large amount of information through their Fourier domain, while retaining



(a) (b) (c) (d)

Figure 1: First 4 components of a PCA decomposition for the luminance component of *Bikes*. Each point represents one view.

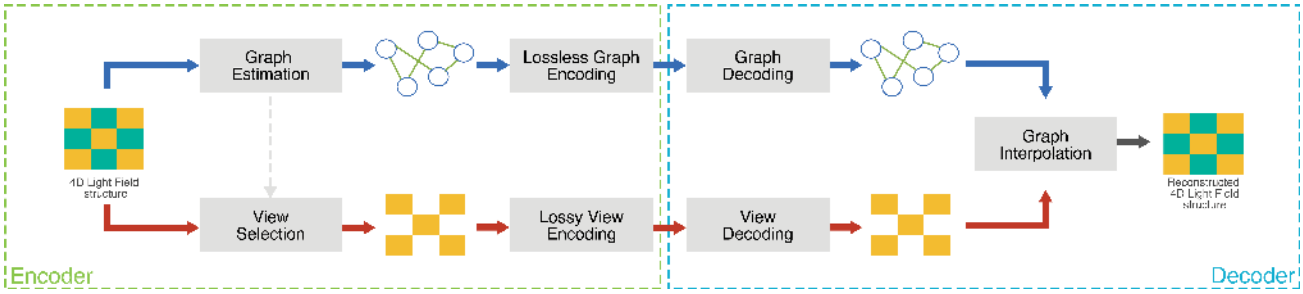


Figure 2: Overview of the compression scheme.

sparsity in the vertex domain. In fact, since a graph Fourier domain is obtained through eigendecomposition of the graph Laplacian matrix, the vectors representing the Fourier basis are different for every graph. Even more, if the graph has been constructed in a way that ensures signal smoothness, the Fourier basis vectors will be representative of this specific set of signals, and the signal will be smooth in this basis.

The general structure of our proposed compression scheme is depicted in Figure 2. The encoder first estimates the graph between all the views of our image. It then follows by losslessly encoding the weights of the graph. At the same time, the encoder will also select a subset of perspective images that are to be compressed directly, as opposed to the rest which will be estimated from them. It then performs a lossy compression of the selected subset. The decoder receives an encoded graph and a subset of views. After decoding both, it solves an optimization problem to estimate the remaining views, and in low bitrates, improves the existing ones. A MATLAB implementation of the proposed solution can be found at the following link: `https://github.com/mmspg/light-field-graph-codec`.

3.1 Encoder

Graph estimation represents the most crucial step in our encoding scheme, as it models the dependencies among perspective images and allows for a faithful reconstruction of the non-encoded views. To obtain a graph that will best describe relationships among the 4D light field, while emphasizing signal smoothness on its structure, we resort to one of the graph learning techniques. In order to reduce the overhead created by sending the graph weights, we consider each view as a vertex, to minimize the number of weights that need to be encoded. As described in,[13] the following optimization problem yields a graph representing smooth signals, while promoting connectedness and providing a mean to control graph sparsity. The problem reads as follows:

$$\textbf{argmin}_{W \in \mathcal{W}_m} \quad tr(Y^T L Y) - \alpha \mathbf{1}^T log(W\mathbf{1}) + \beta \|W\|_F^2 \tag{7}$$

$$L = D - W \tag{8}$$

$$\mathcal{W}_m = \{W \in \mathbb{R}_+^{m \times m} : W = W^T, diag(W) = 0\}, \tag{9}$$

in which W is a weight matrix uniquely describing a graph (and a graph Laplacian matrix L). The signal $Y \in \mathbb{R}^{m \times p}$ is a light field image, vectorized in such a way that each row represents one entire view, where $m = K \times N$ is the total number of views, and p the total number of pixels in one view. Increasing the parameter α enforces stronger connectivity in the graph, while decreasing β promotes sparsity.

In terms of our problem, ensuring a graph is connected is important, as it provides full flexibility in view selection. Specifically, if there were several separate connected components in the graph, view selection would need to provide samples from each of these components to ensure the reconstruction of the entire light field. While clearly a surpassable drawback, this would force the view selection to be dependent on the graph structure, complicating the procedure and making the problem no longer easily distributed. Additionally, graph sparsity also represents an important parameter, as it reduces the overhead of transmitting the graph weights. As shown by Kalofolias et al.,[13] the problem in 7 is convex and has an efficient solution. The code is publicly available in the GSP toolbox.[20]

Once the graph is encoded, an appropriate subset of views is selected. It is worth emphasizing that the graph learning step is carried out independently from the selection of the views and its compression. This brings several advantages. The first advantage is that different encoding solutions can be used to efficiently compress the subset of views. Moreover, as the graph is always encoded losslessly and thus represents a fixed overhead, it can easily be included in any rate allocation problem. The second advantage is that several strategies can be implemented for the selection of the views to be encoded, depending on the use case. For a fixed bitrate, spatial resolution can be favored over angular resolution by selecting a smaller subset of views, which will be compressed with a better quality. Conversely, sending a larger set of views will ensure a better angular resolution, while decreasing the overall quality of all the views. For instance, a progressive stream which would offer an increasingly superior angular resolution is straightforward to implement. Lastly, the learned graph structure can be used to select the subset of views in order to maximize the overall quality of the reconstructed light field, or to provide a trade-off between angular and spatial resolutions, depending on the desired application. As the dashed line in Figure 2 implies, one possibility is to exploit the knowledge of the estimated graph structure to select the views to be encoded, giving priority to more influential views to ensure a more faithful reconstruction.

The computational cost of learning the graph is $\mathcal{O}(m^2 p)$ for computing the distance between all views, and $\mathcal{O}(m^2)$ per iteration of the optimization problem. Taking into account the fact that the number of iterations i is limited[13] and considering that in the majority of cases $p \gg i$, the overall complexity of learning the graph can be written as $\mathcal{O}(m^2 p)$. The cost of encoding the subset of views depends on the compression method of choice. Thus, it might be dominant in the overall compression scheme. For instance, the cost of learning the graph would be negligible with respect to the cost introduced by state of the art video codecs commonly used to encode the perspective images.

3.2 Decoder

After recovering the lossless graph and a lossy subset of views, the decoder exploits the graph to estimate the full light field. In order to recover missing views, the decoder solves an optimization problem enforcing smoothness on the representative graph among the views. Namely, for a view selection matrix M, we want to solve:

$$\textbf{argmin}_X \quad tr(X^T L X) \tag{10}$$

$$s.t. \quad \hat{Y} = MX, \tag{11}$$

where $\hat{Y} \in \mathbb{R}^{m \times p}$ is a matrix containing decoded views in rows corresponding to one of the selected views, and zeros everywhere else. The view selection matrix M projects X to the space of selected view only, keeping only

(a) Bikes (b) Danger_de_Mort (c) Stone_Pillars_Outside (d) Fountain_&_Vincent_2

Figure 3: Central perspective view from each content used in the validating experiment.

the values in corresponding rows. Specifically, it is an identity matrix with zeros on the diagonal for all indices corresponding to not selected views.

This problem can equivalently be written as:

$$\mathbf{argmin}_X \quad tr(X^T L X) + \gamma \|\hat{Y} - MX\|_F^2 \tag{12}$$

with a tunable parameter γ that, if very small, allows changes also among the received views. It is worth noting here that received views went through a lossy compression. Therefore, when selected views are compressed with low bitrates, allowing small changes promoting smoothness on the graph can be beneficial.

Given the parameter γ, it is not difficult to see that the solution to problem 12 is given in closed form with:

$$\hat{X} = (M + \gamma L)^{-1} \hat{Y}, \tag{13}$$

which concludes the work of the decoder and gives the final estimation for the original light field image.

View interpolation has a closed form solution, with the computational cost of $\mathcal{O}(m^3)$ for matrix inversion and $\mathcal{O}(m^2 p)$ for multiplication. As $p \gg m$ in most cases, the overall complexity can be written as $\mathcal{O}(m^2 p)$. However, depending on the choice of compression method for the subset of views, the cost of decoding the subset of views might be dominant in the overall compression scheme.

4. VALIDATING EXPERIMENT

In this section we give an overview of the validating experiment to test the performance of our solution. Specifically, we present the coding conditions and outline the codec configuration. We then introduce a brief description of the anchors and, lastly, delineate how the objective metrics are computed.

4.1 Coding conditions

In order to facilitate the comparison between the proposed approach and the state of the art in light field coding, the same coding conditions as defined in the ICIP 2017 Grand Challenge were adopted for this experiment.[4] In particular, the following four light field contents were selected from the proposed lenslet dataset:[21] *Bikes*, *Danger_de_Mort*, *Stone_Pillars_Outside* and *Fountain_&_Vincent_2* (see Figure 3).

The Light Field toolbox v0.4 was employed to obtain the 4D light field structure of perspective views.[22,23] Prior to the transformation, each 10-bit lenslet image was devignetted and demosaicked. A total of 15×15 perspective views were obtained from the lenslet image, each with a resolution of 625×434 pixels; however, only the central 13×13 views were selected to be encoded and evaluated, following the JPEG Pleno Common Test Conditions.[24] Color and gamma correction was applied to each perspective view prior to the encoding.

The same compression ratios defined for the Grand Challenge were selected for the evaluation of the proposals, namely $R1 = 0.75$ bpp, $R2 = 0.1$ bpp, $R3 = 0.02$ bpp, $R4 = 0.005$ bpp. However, conforming to the JPEG Pleno Common Test Conditions,[24] the bpp were computed as the ratio between the total number of bits used to encode the content, and the total number of pixels in the entire light field, which in our case corresponds to $13 \times 13 \times 434 \times 625$ pixels.

4.2 Codec configuration

The graph was computed on the luminance values of the 4D light field structure. To reduce the overhead, only the luminance graph was transmitted, and it was used for the reconstruction of all YUV channels. Parameters $\alpha = 10^5$ and $\beta = 10$ were empirically chosen for the encoder, whereas for the decoder the parameter γ was set to 10^{-8} and $3 \cdot 10^{-4}$ for the luminance and for the chrominance channels, respectively. The weight matrix of the graph is symmetric and highly diagonally sparse. Therefore, the upper triangle of our weight matrix was rearranged using MATLAB function *spdiags* and losslessly compressed as a *mat* file. Information about the size of each graph can be found in Table 1.

For the experiment, the views composing the 4D light field structure were divided in two sets A and B, forming the views that would be compressed and transmitted alongside the graph, and the views that would be entirely reconstructed on the decoder side, respectively. A total of 85 out of 169 views were assigned to set A, whereas 84 views composed set B, as shown in Figure 4. The views in set A were subsequently converted to YUV color space, downsampled from 444 to 420, 10-bit depth, and compressed using the HEVC/H.265 reference software HM,[25] using profile Main10 and low delay configuration. The Quality Parameter (QP) was chosen to closely match the targeted compression ratio. A summary of the chosen QP and relative file size can be found in Table 1.

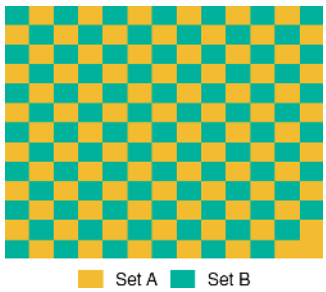


Figure 4: Composition of set A and set B.

4.3 Anchor selection

The results of our coding approach were compared to the results obtained from HEVC/H.265 anchor used in the ICIP 2017 Grand Challenge.[4] In the HEVC/H.265 anchor, the software implementation x265[†] is used to encode the perspective views, which were previously arranged in a serpentine order.

In addition, our results were compared to the best performing algorithm of the ICIP 2017 Grand Challenge[‡], which defines a linear dependency among different views in the angular domain, called Linear Approximation Prior (LAP).[9] As already introduced in section 1, in their work a subset of views is encoded using x265 and transmitted to the encoder along with the quantized linear coefficients. The rest of the views is then estimated using the LAP assumption.

Finally, the JPEG Pleno VM was used as third anchor.[10,11] The provided configuration for the four contents is used for the comparison. However, it should be noted that the configuration files are optimized for random access, which could negatively affect the performance of the codec in terms of objective quality.

4.4 Objective quality evaluation

To evaluate the performance of the proposed coding algorithm with respect to the anchors, PSNR and SSIM were selected from the literature as objective metrics, following the JPEG Pleno Common Test Conditions.[24] In particular, every perspective view at indices (k, l) was converted to YUV color space, 10-bit depth, using the

[†]https://www.videolan.org/developers/x265.html
[‡]http://2017.ieeeicip.org/ChallengeAward.asp

Table 1: Size of the compressed bitstreams, and relative Quality Parameters (QP) for every content and compression ratio.

Content	Graph size	QP	Set A size	Total size	Compression ratio	Target size
Bikes	2489 B	12	3956.07 kB	3958.50 kB	0.707 bpp	4196.89 kB
		23	500.05kB	502.48 kB	0.090 bpp	559.59 kB
		31	110.28 kB	112.71 kB	0.020 bpp	111.92 kB
		41	23.72 kB	26.15 kB	0.005 bpp	27.98 kB
Danger_de_Mort	2646 B	14	4024.55 kB	4027.14 kB	0.720 bpp	4196.89 kB
		25	518.73 kB	521.32 kB	0.093 bpp	559.59 kB
		33	114.30 kB	116.89 kB	0.021 bpp	111.92 kB
		42	24.78 kB	27.36 kB	0.005 bpp	27.98 kB
Stone_Pillars_Outside	3306 B	12	3965.9 kB	3969.2 kB	0.709 bpp	4196.89 kB
		22	504.52 kB	507.74 kB	0.091 bpp	559.59 kB
		28	103.25 kB	106.47 kB	0.019 bpp	111.92 kB
		35	24.25 kB	27.48 kB	0.005 bpp	27.98 kB
Fountain_&_Vincent_2	1884 B	12	4225.2 kB	4227 kB	0.755 bpp	4196.89 kB
		24	493.56 kB	495.4 kB	0.089 bpp	559.59 kB
		31	114.03 kB	115.87 kB	0.021 bpp	111.92 kB
		40	26.25 kB	28.09 kB	0.005 bpp	27.98 kB

conversion matrix defined in Recommendation ITU-R BT.709.6.[26] The metrics were then applied separately to the luma channel Y and for each viewpoint image, as follows:

$$PSNR_Y(k,n) = 10\log_{10}\frac{(2^{10}-1)^2}{MSE(k,n)}, \tag{14}$$

$$SSIM_Y(k,n) = \frac{(2\mu_I\mu_R + c_1)(2\sigma_{IR} + c_2)}{(\mu_I^2 + \mu_R^2 + c_1)(\sigma_I^2 + \sigma_R^2 + c_2)}, \tag{15}$$

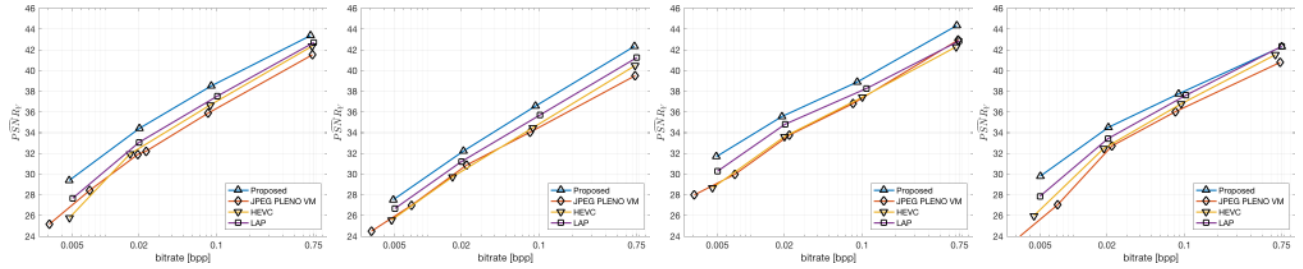
in which $MSE(k,n)$ is the mean square error between the reference and the reconstructed view at indices (k,n), μ_I and μ_R are the mean values, σ_I^2 and σ_R^2 are the variances, and σ_{IR} is the covariance of the two perspective views in channel Y. PSNR was computed for channels U, V according to Equation 14, and a weighted average[27] was obtained as follows:

$$PSNR_{YUV}(k,n) = \frac{6PSNR_Y(k,n) + PSNR_U(k,n) + PSNR_V(k,n)}{8}. \tag{16}$$

The average PSNR value for Y channel was then computed across the viewpoint images:

$$\widehat{PSNR}_Y = \frac{1}{(K-2)(N-2)}\sum_{k=2}^{K-1}\sum_{n=2}^{N-1} PSNR_Y(k,n). \tag{17}$$

Similarly, the average $\widehat{PSNR}_{YUV}$ and $\widehat{SSIM}_Y$ values were computed. Additionally, Bjontegaard rate savings percentages and PSNR gains[28] were computed with respect to all the anchors for $\widehat{PSNR}_Y$ and $\widehat{PSNR}_{YUV}$ values.

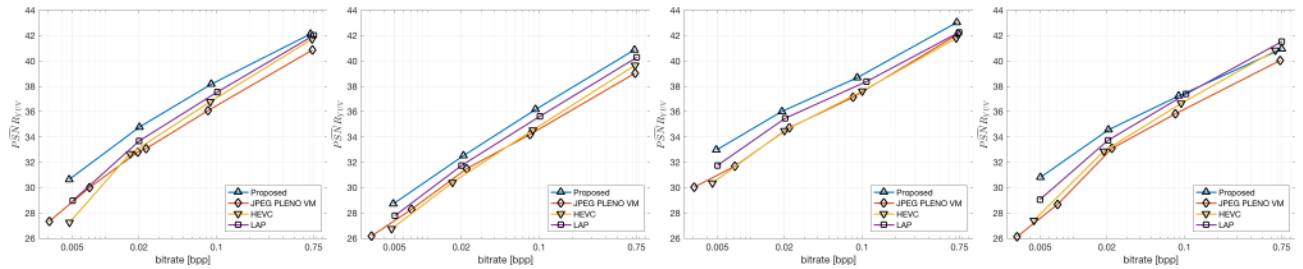


(a) Bikes (b) Danger_de_Mort (c) Stone_Pillars_Outside (d) Fountain_&_Vincent_2

Figure 5: $\widehat{PSNR}_Y$ vs bitrate for every content. The bitrate is shown in logarithmic scale to improve readability.

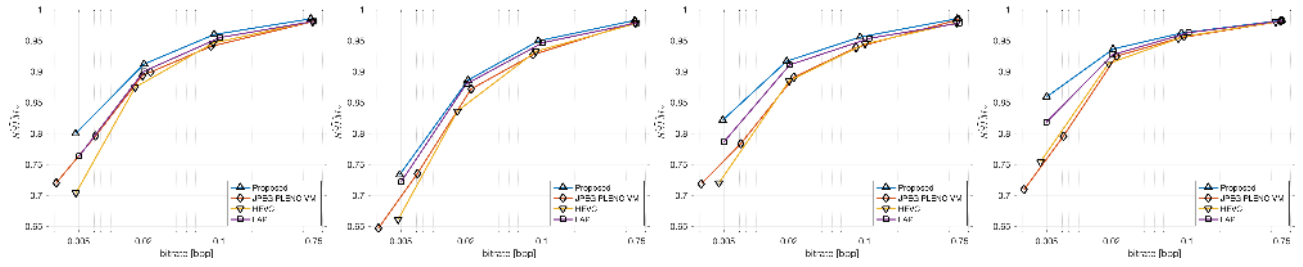
5. RESULTS AND DISCUSSION

Figure 5 shows the values of $\widehat{PSNR}_Y$ against the bitrate, separately for each content under examination. It can be seen how our proposal outperforms the anchors pretty consistently across different bitrates for contents *Bikes*, *Danger_de_Mort* and *Stone_Pillars_Outside*. For content *Fountain_&_Vincent_2* a notable gain can be observed for lower bitrates, whereas for high bitrates the performance is equivalent to codec LAP.



(a) Bikes (b) Danger_de_Mort (c) Stone_Pillars_Outside (d) Fountain_&_Vincent_2

Figure 6: $\widehat{PSNR}_{YUV}$ vs bitrate for different contents. The bitrate is shown in logarithmic scale to improve readability.



(a) Bikes (b) Danger_de_Mort (c) Stone_Pillars_Outside (d) Fountain_&_Vincent_2

Figure 7: $\widehat{SSIM}_Y$ vs bitrate for different contents. The bitrate is shown in logarithmic scale to improve readability.

Table 2: Bjontegaard rate savings with respect to the three anchors HEVC, JPEG Pleno VM and LAP, for all four contents, and on average.

	HEVC/H.265		JPEG PLENO VM		LAP	
	$\widehat{PSNR}_Y$	$\widehat{PSNR}_{YUV}$	$\widehat{PSNR}_Y$	$\widehat{PSNR}_{YUV}$	$\widehat{PSNR}_Y$	$\widehat{PSNR}_{YUV}$
Bikes	-46.22%	-43.10%	**-57.65%**	**-55.31%**	**-36.52%**	**-31.89%**
Danger_de_Mort	-47.53%	-45.37%	-50.97%	-47.64%	-30.14%	-26.14%
Stone_Pillars_Outside	**-53.46%**	**-50.02%**	-52.28%	-48.93%	-35.19%	-29.68%
Fountain_&_Vincent_2	-39.74%	-33.45%	-51.24%	-49.04%	-23.06%	-15.36%
Average	-46.74%	-42.99%	-53.04%	-50.23%	-31.23%	-25.77%

Table 3: Bjontegaard PSNR difference with respect to the three anchors HEVC, JPEG Pleno VM and LAP, for all four contents, and on average.

	HEVC/H.265		JPEG Pleno VM		LAP	
	$\widehat{PSNR}_Y$	$\widehat{PSNR}_{YUV}$	$\widehat{PSNR}_Y$	$\widehat{PSNR}_{YUV}$	$\widehat{PSNR}_Y$	$\widehat{PSNR}_{YUV}$
Bikes	1.93 dB	**1.53 dB**	**2.41 dB**	**1.86 dB**	**1.33 dB**	**0.98 dB**
Danger_de_Mort	1.89 dB	1.51 dB	2.08 dB	1.56 dB	1.06 dB	0.76 dB
Stone_Pillars_Outside	**1.96 dB**	1.47 dB	1.97 dB	1.44 dB	1.13 dB	0.75 dB
Fountain_&_Vincent_2	1.40 dB	1.01 dB	1.94 dB	1.51 dB	0.72 dB	0.39 dB
Average	1.80 dB	1.38 dB	2.10 dB	1.59 dB	1.06 dB	0.72 dB

A similar trend can be observed for values of $\widehat{PSNR}_{YUV}$ (Figure 6). In particular, it is worth noting that, although the performance of our proposal remains consistently better than or equivalent to the anchors, a smaller gain in dB can be observed. This may be due to the fact that anchors HEVC/H.265 and LAP apply a chroma subsampling factor of 422, which leads to improved color fidelity. Moreover, the choice of using the luminance graph to reconstruct the chroma values may lead to a loss in performance in the proposed codec.

Values of $\widehat{SSIM}_Y$ why show that our proposal has similar performance with respect to the anchors for high bitrates (Figure 7). However, a significant gain can be observed for low bitrates with respect to the other codecs.

Bjontegaard rate savings results (Tables 2 and 3) show that our proposal achieves on average a 46.74% rate reduction and a PSNR gain of 1.80 dB for $\widehat{PSNR}_Y$ (42.99% and 1.38 dB for $\widehat{PSNR}_{YUV}$, respectively) when compared to HEVC/H.265. The maximum rate reduction is achieved for content *Stone_Pillars_Outside* (53.46% and 50.02% for $\widehat{PSNR}_Y$ and $\widehat{PSNR}_{YUV}$, respectively), while the minimum gain is achieved for content *Fountain_&_Vincent_2* (39.74% and 33.45% for $\widehat{PSNR}_Y$ and $\widehat{PSNR}_{YUV}$, respectively). Slightly higher rate reductions can be achieved in comparison to the JPEG PLENO VM (53.04% and 50.23% for $\widehat{PSNR}_Y$ and $\widehat{PSNR}_{YUV}$, respectively), for which bigger PSNR gains can also be observed (2.1 and 1.59 dB for $\widehat{PSNR}_Y$ and $\widehat{PSNR}_{YUV}$, respectively). When analysing the difference in performance between the JPEG Pleno VM and our solution, it should be noted that configuring the codec for random access may result in a sub-par performance, objective metric-wise. Smaller, but still significant gains can be seen with respect to LAP codec, with a rate reduction of 31.23% and 25.77% on average for $\widehat{PSNR}_Y$ and $\widehat{PSNR}_{YUV}$, respectively, and a PSNR gain of 1.06 and 0.72 dB.

Results show that light field compression efficiency can benefit from sending only a subset of the perspective views and reconstructing the entire 4D light field at the receiver side, as shown by the superior performance of both the proposed solution and the LAP codec with respect to the HEVC/H.265 anchor. In particular, both LAP and our proposed solution rely on sparsely capturing the similarities among the perspective images to aid in the reconstruction process at the decoder side. However, in the LAP algorithm the reconstructed images are seen as a linear combination of only the views that have been encoded and sent, thus disregarding the

correlation among the views that need to be reconstructed. On the other hand, our approach encodes all the dependencies in the 4D light field, regardless of the set they belong to. Moreover, whereas the coefficients of the linear dependency among views are quantized in the LAP scheme, the graph weights are losslessly compressed in our solution to improve reconstruction quality. Results show that this approach achieves a superior performance in reconstructing the 4D light field.

6. CONCLUSIONS

In this paper we presented a new approach to compress light field images based on a graph learning technique. We demonstrate its theoretical soundness, as well as its application to image coding. Our validating experiment shows that sensible gains can be achieved by using our solution against state-of-the-art encoders. Future work will focus on improving coding efficiency by implementing a more efficient selection of encoded views based on graph structure, and by improving color performance by incorporating chroma information in the graph weights.

ACKNOWLEDGMENTS

This work has been conducted in the framework of the Swiss National Foundation for Scientific Research (FN 200021_159575) project Light field Image and Video coding and Evaluation (LIVE).

REFERENCES

[1] Gershun, A., "The light field," *Journal of Mathematics and Physics* **18**(1), 51–151 (1939).

[2] Adelson, E. H. and Bergen, J. R., "The plenoptic function and the elements of early vision," (1991).

[3] Levoy, M. and Hanrahan, P., "Light field rendering," in [*Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*], 31–42, ACM (1996).

[4] Viola, I. and Ebrahimi, T., "Quality assessment of compression solutions for ICIP 2017 Grand Challenge on light field image coding," *2018 International Conference on Multimedia and Expo Workshops* (2018).

[5] Monteiro, R., Lucas, L., Conti, C., Nunes, P., Rodrigues, N., Faria, S., Pagliari, C., Silva, E., and Soares, L. D., "Light field hevc-based image coding using locally linear embedding and self-similarity compensated prediction," in [*ICME 2016 Grand Challenge on Light Field Image Compression*], (2016).

[6] Jin, X., Han, H., and Dai, Q., "Plenoptic image coding using macropixel-based intra prediction," *IEEE Transactions on Image Processing* **27**(8), 3954–3968 (2018).

[7] Ahmad, W., Olsson, R., and Sjöström, M., "Interpreting plenoptic images as multi-view sequences for improved compression," in [*IEEE International Conference on Image Processing (ICIP)*], IEEE (2017).

[8] Jia, C., Yang, Y., Zhang, X., Zhang, X., Wang, S., Wang, S., and Ma, S., "Optimized inter-view prediction based light field image compression with adaptive reconstruction," in [*IEEE International Conference on Image Processing (ICIP)*], IEEE (2017).

[9] Zhao, S. and Chen, Z., "Light field image coding via linear approximation prior," in [*IEEE International Conference on Image Processing (ICIP)*], IEEE (2017).

[10] ISO/IEC JTC 1/SC29/WG1 JPEG, "JPEG PLENO Light Field Coding VM1." Doc. N80028, Berlin, Germany (July 2018).

[11] Astola, P. and Tabus, I., "Light Field Compression of HDCA Images Combining Linear Prediction and JPEG 2000," *EUSIPCO 2018* (2018).

[12] Dong, X., Thanou, D., Frossard, P., and Vandergheynst, P., "Learning laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing* **64**(23), 6160–6173 (2016).

[13] Kalofolias, V., "How to learn a graph from smooth signals," in [*Artificial Intelligence and Statistics*], 920–929 (2016).

[14] Fracastoro, G., Thanou, D., and Frossard, P., "Graph transform learning for image compression," in [*Picture Coding Symposium (PCS), 2016*], 1–5, IEEE (2016).

[15] Maugey, T., Ortega, A., and Frossard, P., "Graph-based representation for multiview image coding," *arXiv preprint arXiv:1312.6090* (2013).

[16] Su, X., Rizkallah, M., Maugey, T., and Guillemot, C., "Graph-based light fields representation and coding using geometry information," in [*Image Processing (ICIP), 2017 IEEE International Conference on*], 4023–4027, IEEE (2017).

[17] Chao, Y.-H., Cheung, G., and Ortega, A., "Pre-demosaic light field image compression using graph lifting transform," in [*Image Processing (ICIP), 2017 IEEE International Conference on*], 3240–3244, IEEE (2017).

[18] Yang, S., Cheung, G., Liu, J., and Guo, Z., "Soft decoding of light field images using pocs and fast graph spectral filters,"

[19] Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P., "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine* **30**(3), 83–98 (2013).

[20] Perraudin, N., Paratte, J., Shuman, D., Martin, L., Kalofolias, V., Vandergheynst, P., and Hammond, D. K., "GSPBOX: A toolbox for signal processing on graphs," *ArXiv e-prints* (Aug. 2014).

[21] Řeřábek, M. and Ebrahimi, T., "New light field image dataset," *8th International Conference on Quality of Multimedia Experience (QoMEX)* (2016).

[22] Dansereau, D. G., Pizarro, O., and Williams, S. B., "Decoding, calibration and rectification for lenselet-based plenoptic cameras," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* , IEEE (Jun 2013).

[23] Dansereau, D. G., Pizarro, O., and Williams, S. B., "Linear volumetric focus for light field cameras," *ACM Transactions on Graphics (TOG)* **34** (Feb. 2015).

[24] ISO/IEC JTC 1/SC29/WG1 JPEG, "JPEG PLENO - Light Field Coding Common Test Conditions." Doc. N80027, Berlin, Germany (July 2018).

[25] ITU-T Q.6/SG 16 and ISO/IEC JTC 1/SC 29/WG 11, "High Efficiency Video Coding (HEVC) reference software HM." [Online]. Available: `https://hevc.hhi.fraunhofer.de/trac/hevc/browser/trunk`.

[26] ITU-R BT.709.6, "Parameter values for the HDTV standards for production and international programme exchange." International Telecommunication Union (June 2015).

[27] Ohm, J.-R., Sullivan, G. J., Schwarz, H., Tan, T. K., and Wiegand, T., "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC)," *IEEE Transactions on Circuits and Systems for Video Technology* **22**(12), 1669–1684 (2012).

[28] Bjontegaard, Gisle, "Calculation of average PSNR differences between RD-curves." International Telecommunication Union (March 2001).