Graph Fourier Transform for Light Field Compression

Vitor Rosa Meireles Elias and Wallace Alves Martins

***Abstract*— This work proposes the use of graph Fourier transform (GFT) for light field data compression. GFT is a tool developed for the emerging field of digital signal processing on graphs, which combines graph theory and classical DSP in order to exploit signal-related information present on graph structures. The proposed method explores the high correlation of residual images from light field. Simulations with real light field data indicate significant reduction in the number of coefficients for data representation; for instance, an 8.97% reduction was achieved while keeping smaller mean squared error when compared to discrete cosine transform-based compression.**

***Keywords*— Signal Processing on Graphs, Graph Fourier Transform, Compression, Light Field, Discrete Cosine Transform.**

I. Introduction

Light field imaging is a promising technology that opens a variety of new possibilities to entertainment industries, such as photography and cinema, by capturing 4D data from a scene. Practical light field capturing techniques usually consist of a microlens array placed between the sensor plane and the main lens of a digital camera so that each microlens generates a micro-image associated with a different perspective of the scene. An alternative way of capturing a light field is through an array of cameras, or by moving a single camera and capturing the scene on a grid of determined positions. All these methods generate a large amount of data when compared to traditional imaging, since many images are used to compose a single scene. Dealing with the resulting huge amount of data is a challenging task [1], [2].

In order to provide solutions and a standard framework to deal with data generated by light field imaging and other techniques, the JPEG standardization committee created, in 2015, the "JPEG Pleno" initiative and the process is due to continue through the next years with a first international standard in 2018 [3].

Given the advances in video coding technology and the advent of high-efficiency encoders that are suitable to various types of contents, recent researches focus on improving the compression of light field data by using HEVC-based solutions [4]–[6]. Since HEVC encompasses a range of complex signal processing steps, researchers usually focus on individual components of the encoding scheme, such as intra or inter prediction, data-arrangement, or other key processing blocks.

This work focuses on the transform block of a simplified encoding scheme. Instead of employing *discrete-cosine transform* (DCT) along with *discrete-sine transform* (DST) as in the HEVC encoder, this paper proposes the use of *graph Fourier transform* (GFT) [7] in order to reduce the number of coefficients required to represent a block of prediction residual, which is the difference between an image and its prediction. GFT has recently arisen as one of the main signal processing tools within the emerging field of *Digital Signal Processing on Graphs* (DSP_G) [8]. In DSP_G, signals are represented as graphs, which may contain more information than usual signal representations, and the traditional DSP tools are translated and adapted to process data in the graph structure.

GFT applied to a graph signal is able to provide better compression quality in terms of mean squared error (MSE), while keeping the same number of transform coefficients as the two-dimensional DCT applied to a traditional image signal. However, unlike DCT, GFT and its companion *inverse graph Fourier transform* (IGFT) depend on the signal-related adjacency matrix and the graph structure requires extra data for signal representation [7], [9]. This work proposes a method that explores the redundancy among light field images in order to reduce the impact of the extra data on the present graph structure and compares the efficiency of this method to that of the DCT.

It is worth pointing out that the use of GFT within the image processing context is not new. Indeed, some previous works use GFT for image compression. The seminal paper [7] that introduces GFT as a tool for DSP_G presents image compression as an example of GFT application. In addition, the authors in [9] propose a hybrid image transform for color and depth images based on DCT and GFT.

This paper is organized as follows. Section II provides the background knowledge on light field and signal processing on graphs required for this work. Section III introduces the proposed compression methodology. Section IV describes the simulations and discusses the results obtained. Section V compiles some of the next steps for future works, and Section VI presents some conclusions of this paper.

II. Background on Light Field and DSP_G

Although both light field and graph theories are known for a long time, only recently technology allowed the construction of real and practical light field capturing devices and, even more recently, the fundamentals of digital signal processing on graphs were proposed. This section provides the basic knowledge about light field data and the DSP_G concepts and tools used in this work.

A. Light Field

Light field data usually consist of a set of multiple images of different perspectives from a scene that are captured either

Mr. Vitor R. M. Elias and Prof. Wallace A. Martins are with the Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil (emails: vitor.elias@smt.ufrj.br, wallace.martins@smt.ufrj.br).

The authors are grateful for the financial support provided by CNPq, CAPES, and FAPERJ, Brazilian research councils.

Fig. 1. Example of images that compose light field data.

by an array of microlenses, an array of cameras, or by moving and capturing with a single camera. Fig. 1 shows an example of 16 images that compose a light field obtained from the *Stanford light field archive* [10]. The entire light field data for this case was captured by a moving camera on a rectangular grid with 16×16 positions, yielding a total of 256 images; this is the data set used throughout the paper.

Some important applications that justify why light field imaging is a promising technology include the *light field rendering*, which allows the creation of novel views by manipulating multiple previously captured views, and the *synthetic aperture photography*, which allows photographs to be refocused after they are taken [11].

B. Digital Signal Processing on Graphs

A graph is the pair $G = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} = \{v_0, \ldots, v_{N-1}\}$ is the set of N *vertices*, and $\mathcal{A} = \{a_{00} \ldots a_{(N-1)(N-1)}\}$ is the set of N^2 *edges*. The relation between $\mathcal{V}$ and $\mathcal{A}$ is as follows: each element a_{ij} represents the edge connecting vertex v_j to vertex v_i.[1] In other words, an edge represents a relation between two vertices, and this relation depends on the underlying application of the graph. In a directed graph, edges have orientations and a_{ij} may differ from a_{ji}. If a graph is undirected, edges have no orientations. The set $\mathcal{A}$ of all edges can also be represented by an $N \times N$ *adjacency matrix* $\mathbf{A}$, which is symmetric if the graph is undirected.

A finite-duration complex-valued discrete-time signal $s[n]$ can be regarded as a function $s : \{0, 1, \ldots, N-1\} \to \mathbb{C}$ that maps points within a well-structured domain into the complex plane. Indeed, any two points n_1, n_2 within the domain $\{0, 1, \ldots, N-1\}$ can be compared, i.e. $n_1 > n_2$ or $n_1 = n_2$ or $n_1 < n_2$; for any $n \in \{0, 1, \ldots, N-1\}$, the inequalities $N - 1 \geq n \geq 0$ always hold. These domain properties induce several useful properties in the analysis of discrete-time signals $s[n]$. Nonetheless, there are many applications that call for the use of a more general domain and, in these cases, graphs may be the appropriate structure.

[1] In fact, if $a_{ij} = 0$ one can consider there is no edge connecting v_j to v_i.

The concept of signals on graphs uses the set of vertices $\mathcal{V}$ of a graph G as the domain of a dataset of N elements, and the set of edges $\mathcal{A}$ of the graph G to encode an underlying relationship between the elements of this dataset. For example, for a dataset of temperature measurements of N sensors, vertices can represent the sensors' spatial positions, whereas edges may represent the distances between pairs of sensors on an undirected graph. Note that, in this case, the domain is not ordered — one cannot state that a sensor 3D position is larger/smaller than the other, in principle.

In this paper, an entire signal $s : \mathcal{V} \to \mathbb{C}$ on the vertices of a graph is referred to as $\mathbf{s}$, in which the n-th entry of $\mathbf{s}$ is $s_n = s[v_n]$, with $v_n \in \mathcal{V}$. Tools in DSP_G are usually developed as equivalent forms of existing tools in classical DSP [8]. In DSP_G, given a graph G with adjacency matrix $\mathbf{A}$, a *graph shift* is defined as

$$\tilde{\mathbf{s}} = \mathbf{A}\mathbf{s}. \tag{1}$$

The operation defined in (1) is the graph equivalent of the *delay* or *shift*, which is the basic building block of filters in classical DSP. That is, in graph domain, shifting a signal $\mathbf{s}$ is equivalent to replacing each signal sample s_n with a linear combination of its neighborhood according to weights given by the adjacency matrix $\mathbf{A}$. Classical DSP shift can be viewed as a special case of graph shift when the adjacency matrix is the *cyclic shift matrix*

$$\mathbf{C} = \begin{bmatrix} & & & 1 \\ 1 & & & \\ & \ddots & & \\ & & 1 & \end{bmatrix}. \tag{2}$$

The concept of filters is also extended to the graph domain. Indeed, in classical DSP, a finite-duration impulse response (FIR) filter of length $L \leq N$ with coefficients h_l induces the following circular convolution:

$$\begin{bmatrix} \bar{s}[0] \\ \bar{s}[1] \\ \vdots \\ \bar{s}[N-1] \end{bmatrix} = \underbrace{\begin{bmatrix} h_0 & h_{N-1} & \cdots & h_1 \\ h_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & h_{N-1} \\ h_{N-1} & \cdots & h_1 & h_0 \end{bmatrix}}_{=\mathbf{H}(\mathbf{C})=\sum_{l=0}^{L-1} h_l \mathbf{C}^l} \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[N-1] \end{bmatrix}, \tag{3}$$

where $h_l = 0$ for $l \geq L$. Similarly, a linear, shift-invariant *graph filter* is defined as a polynomial over a general adjacency matrix $\mathbf{A}$, i.e.,

$$\mathbf{H}(\mathbf{A}) = \sum_{l=0}^{L-1} h_l \mathbf{A}^l. \tag{4}$$

C. Graph Fourier Transform

In classical DSP, the basis vectors that comprise a discrete Fourier transform are the orthonormal vectors that diagonalize the cyclic shift matrix $\mathbf{C}$ in (2), and the frequencies are their corresponding eigenvalues. Similarly, a graph Fourier transform can be defined by using the vectors that diagonalize the adjacency matrix $\mathbf{A}$. In other words, the *graph Fourier transform* is defined as

$$\hat{\mathbf{s}} = \mathbf{F}\mathbf{s} = \mathbf{V}^{-1}\mathbf{s}, \tag{5}$$

where $\mathbf{V}$ is a matrix whose columns are the eigenvectors of the adjacency matrix $\mathbf{A}$ if it is diagonalizable. If $\mathbf{A}$ is not diagonalizable, $\mathbf{V}$ is the set of generalized eigenvectors from the Jordan decomposition of $\mathbf{A}$ [7], [12]. The *inverse graph Fourier transform* is defined as

$$\mathbf{s} = \mathbf{F}^{-1}\hat{\mathbf{s}} = \mathbf{V}\hat{\mathbf{s}}. \quad (6)$$

If $\mathbf{V}$ is an orthogonal matrix, which is the case when $\mathbf{V}$ is the matrix whose columns are normalized eigenvectors of a symmetric adjacency matrix $\mathbf{A}$, then $\mathbf{V}^{-1} = \mathbf{V}^{\mathrm{T}}$.

III. Light Field Compression Methodology

This section describes the compression scheme, signal models, and figures of merit adopted throughout the paper.

A. Compression process

The main interest of this work is to evaluate how efficient the proposed GFT-based compression method can be in comparison to the traditional DCT when applied to blocks of prediction residual. In order to define how the residual is computed, the prediction needs to be defined for the light field case, since images are not associated with a time sequence as usual on a video coding process.

For the light field presented in Section II-A, considering the 16×16 grid over which the multiple images are taken, this work assumes that the first image from each of the 16 grid lines is an intra image, i.e., no prediction is assumed when encoding these images. The following 15 images from each line are inter images for which GFT and DCT are used to perform a lossy encoding of their corresponding residual blocks. Given the capturing process and the format of light field data, it is clear that images next to each other tend to be more similar than images far from each other. Hence, the prediction should be restricted to the spatial neighborhood of a given image in order to reduce the prediction residual. A simple prediction scheme adopted in this work is $\mathbf{I}_k^{\mathrm{p}} = \mathbf{I}_{k-1}$, where $k \in \{2, 3, \ldots, K\}$ is the index of the k-th image at each line of the light field array and $\mathbf{I}^{\mathrm{p}}$ is the prediction for an image. For the light field data treated in this paper, $K = 16$ images per light field line. Finally, the residual is given by $\mathbf{R}_k = \mathbf{I}_k - \mathbf{I}_k^{\mathrm{p}}$.

Fig. 2 illustrates two examples of other prediction schemes that may be utilized: zig-zag sequencing starting from the upper-left corner of the light field array, or blocks of images where all residuals are relative to a central image, for example.

B. Signal model

Once the residual image $\mathbf{R}_k$ is obtained, it is divided into T blocks of size 32×32 pixels. If much smaller blocks are used, such as 4×4 or 8×8 pixels, blocks at the same position for different residual images may have low correlation with each other. The proposed compression method requires that blocks at the same position for various residual images are similar.

In order for the GFT to be applied, the residual block needs to be represented as a graph. Consider the t-th block of $M_1 \times M_2$ pixels from residual image $\mathbf{R}_k$, denoted as $\mathbf{B}_{k,t}$, with $t \in \{1, 2, \ldots, T\}$. The signal $\mathbf{s}_{k,t}$ associated with the block is defined as the column vector formed by stacking the columns of $\mathbf{B}_{k,t}$. The corresponding adjacency matrix $\mathbf{A}_{k,t}$ is defined by the nearest-neighbor (NN) image model proposed in [13]. This model states that a pixel is only related to the nearest pixels as shown in Fig. 3, i.e., every edge from the graph representation that lies outside the neighborhood of a pixel is equal to zero. Considering a pixel from the center of the block, its neighborhood is composed by the pixels directly under, above, to the left, and to the right from the reference pixel. If a pixel is located at the upper-left corner of the block, only the pixel to the right and the pixel under it are considered as neighbors. This model leads to a sparse $\mathbf{A}_{k,t}$ matrix with a fixed structure. Sparsity comes from the fact that each vertex has at most four nonzero edges associated with its neighborhood. The structure is fixed because these edges are represented by the same entry positions in $\mathbf{A}_{k,t}$ for different blocks of the same size, although entry values (i.e. edge weights) may differ. Another key feature of the NN image model is that, for a given pair of columns of pixels in a block, all horizontal edges connecting the two columns have the same values. Likewise, for a pair of lines, all vertical edges have the same values. Considering this property, the $N \times N$ matrix $\mathbf{A}_{k,t}$ associated with an $M_1 \times M_2$ block, where $N = M_1 M_2$, has at most $(M_1 - 1) + (M_2 - 1)$ different nonzero values. The edge values are computed by minimizing $\|\mathbf{A}\mathbf{s} - \mathbf{s}\|_2$ subject to the matrix structure imposed by the previously described model properties. This is an overdetermined least-squares minimization problem.

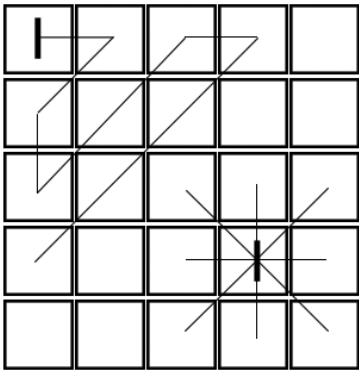

Fig. 2. Examples of alternative prediction schemes on the light field array. The bold mark in the center of a box indicates an intra image.

Note that $\mathbf{A}_{k,t}$ is symmetric as pixel relations are assumed to be undirected and, thus, is diagonalizable.

C. Proposed method

Once the adjacency matrix is computed for a block, graph Fourier transform matrix $\mathbf{F}_{k,t}$ can be computed as the transpose of the matrix composed by normalized eigenvectors of $\mathbf{A}_{k,t}$, as it is diagonalizable and symmetric. Transform coefficients $\hat{\mathbf{s}}_{k,t}$ are obtained by applying $\mathbf{F}_{k,t}$ to $\mathbf{s}_{k,t}$. The original block information can be recovered from transform coefficients using $\mathbf{F}_{k,t}^{\mathrm{T}}$ to perform the IGFT.

By setting the Q smallest transform coefficients to zero, the signal $\mathbf{s}_{k,t}^{Q}$ recovered by the IGFT is a compressed version of $\mathbf{s}_{k,t}$, and the compressed block $\mathbf{B}_{k,t}^{Q}$ is obtained from the compressed graph signal. In this paper, the MSE between

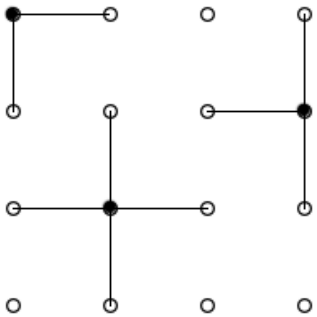

Fig. 3. Relation edges proposed in the nearest-neighbor image model for corner, edges, and central pixels.

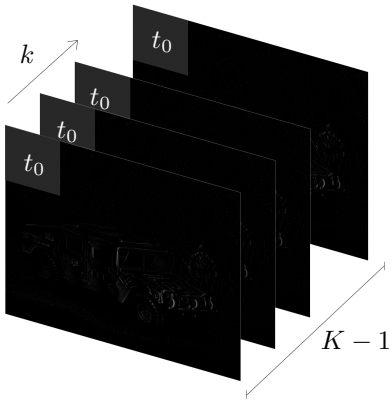

Fig. 4. Representation of a block position t_0 for $K-1$ residual images.

compressed and original residual blocks is used as figure of merit to determine how compression affects the images. It is expected that the MSE will behave as a non-decreasing function of Q, as there is a trade-off between compression rate and image quality. As both GFT and DCT concentrate most of signal energy on few transform coefficients, only a small part of the block information is lost during compression. When compared to DCT, GFT provides better MSE for the same Q, but the downside of GFT is that $\mathbf{F}_{k,t}$ depends on $\mathbf{A}_{k,t}$ associated with the block being compressed, unlike DCT, which is a fixed transform. That is, when performing inverse transform, transform coefficients and block-dependent $\mathbf{F}_{k,t}^{\mathrm{T}}$ are required in the IGFT case, and IDCT requires only the transform coefficients.

Considering light field data, images next to each other are similar and, consequently, the associated residual images are highly redundant. By exploring this redundancy, it is possible to avoid transmitting a different $\mathbf{F}_{k,t}$ or $\mathbf{A}_{k,t}$ with every single block. In the proposed scheme, for a given block position t_0, only one $\mathbf{A}_{k,t_0}$ is considered for the entire light field line, i.e., for $K-1$ residual images, as depicted in Fig. 4. In the proposed compression method, only the adjacency matrices associated with the central residual image $\mathbf{R}_8$, with blocks $\mathbf{B}_{8,t}$, are computed, for the case of a light field with 16 total images per line and 15 residual images. Note that the first image from each line is an intra image and no prediction nor residual image is associated with it.

By utilizing a single adjacency matrix for $K-1$ similar blocks, the impact of transmitting $\mathbf{A}_{k,t}$ along with the transform coefficients is reduced, but GFT efficiency may be degraded for the blocks that are not the reference blocks when the adjacency matrix is computed. Central residual image $\mathbf{R}_8$ is defined heuristically as reference, in order to minimize degradation of GFT. This definition assumes that blocks' similarity is reduced the further a image is from the reference image, and the central image is, on average, the best approximation for the entire line.

It is important to note that, by transmitting $\mathbf{A}$ instead of $\mathbf{F}$, the transmitted data is critically reduced, given the inherent sparsity and structure of the adjacency matrix discussed in Section III-B, but complexity on receiver side is increased since $\mathbf{F}$ needs to be computed from the adjacency matrix.

IV. Simulation and Results

During the simulations, only luminance component from the light field images is considered. Images are 512×640 pixels. Utilizing blocks with 32×32 pixels, DCT yields 1024 transform coefficients, from which the 100 largest coefficients are kept, i.e., $Q = 924$. MSE for reconstructed residual images from DCT compressed coefficients is computed. The algorithm searches for the largest Q for which GFT results on better MSE for a residual image when compared to DCT. That is, how many more GFT coefficents can be set to zero and still yield smaller mean squared error. In this simulation, Q is set for a whole image, thus, every block $\mathbf{B}_{k,t}$ in the k-th residual image is represented by the same number of transform coefficients.

The result for this simulation, considering only the first light field line, is shown in Fig. 5 as the difference between Q used for GFT and DCT. The resulting MSE for this case is shown in Fig. 6. As expected, the best case occurs for the reference residual image $\mathbf{R}_8$ from which $\mathbf{A}$ matrix is computed, where GFT is capable of providing better MSE while setting 28 more coefficients to zero. Results also show that, as images get further apart from the reference image, GFT efficiency decays when compared to DCT. For the last residual image $\mathbf{R}_{15}$, GFT needs two more coefficients than DCT in order to provide smaller MSE. Previous simulations are extended to the whole light field data and the result is presented in Fig. 7. This shows consistency of the method for all light field lines.

Considering the total number of coefficients, in the present simulation, DCT requires 100 transform coefficients per block. Each image is divided into $T = 320$ blocks, and there are 240 residual images. This yields 7,680,000 transform coefficients for the whole light field data in the DCT case. From the previous results, GFT requires 6,673,600 transform coefficients and 320 $\mathbf{A}$ matrices per line. As only 62 coefficients are needed to construct each $\mathbf{A}$, as shown in Section III-B, GFT requires a total of 6,991,040 coefficients. This means 8.97% reduction in number of coefficients while keeping smaller MSE than DCT.

When applied to two other light field sets from [10], namely *Lego Knights* and *Tarot cards and crystal ball*, the proposed compression methodology achieves a combined reduction in number of coefficients of 4.6% as compared to DCT.

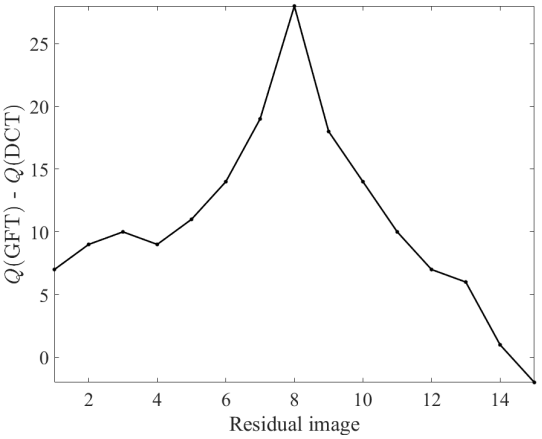

Fig. 5. Difference between Q values used for GFT and DCT on the first light field line.

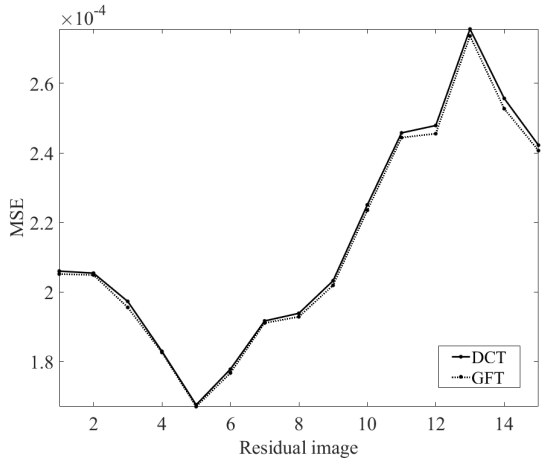

Fig. 6. MSE for each residual image for GFT and DCT.

V. Future Works

This paper contains some preliminary results of an ongoing research. Thus, some of the assumptions and heuristics adopted here admit deeper investigations, as both light field and DSP_G are developing fields, and putting them together is new to the best of our knowledge. Hence, the proposed method should be tested on a larger data base; the NN image model could be used considering data from all residual blocks instead of a single reference block for matrix $\mathbf{A}$ computation, which may improve GFT efficiency for various blocks; other image models and prediction schemes should be tested; new figures of merit for compression efficiency evaluation should be used.

VI. Conclusions

This work provided an efficient method of using GFT as an alternative to DCT when compressing residual image blocks from light field data. The proposed method allows significant reduction in the number of transmitted coefficients while providing smaller MSE when compared to DCT. The method was applied to real light field data and results indicate that GFT is a viable transform for compression purposes when there is high correlation between various images.

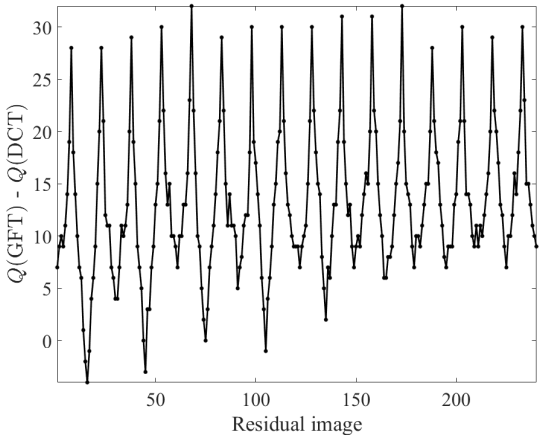

Fig. 7. Difference between Q values used for GFT and DCT on the whole light field.

References

[1] M. Magnor and B. Girod, "Data compression for light-field rendering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, pp. 338–343, Apr 2000.

[2] T. Sakamoto, K. Kodama, and T. Hamamoto, "A study on efficient compression of multi-focus images for dense Light-Field reconstruction," *2012 IEEE Visual Communications and Image Processing, VCIP 2012*, Nov 2012.

[3] T. Ebrahimi, S. Foessel, F. Pereira, and P. Schelkens, "JPEG Pleno: toward an efficient representation of visual reality," *IEEE Multimedia*, vol. 23, pp. 14–20, Oct.-Dec 2016.

[4] C. Perra and P. Assuncao, "High efficiency coding of light field images based on tiling and pseudo-temporal data arrangement," in *2016 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pp. 1–4, IEEE, Jul 2016.

[5] R. Monteiro, L. Lucas, C. Conti, P. Nunes, N. Rodrigues, S. Faria, C. Pagliari, E. da Silva, and L. Soares, "Light field HEVC-based image coding using locally linear embedding and self-similarity compensated prediction," in *2016 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pp. 1–4, IEEE, Jul 2016.

[6] Y. Li, R. Olsson, and M. Sjostrom, "Compression of unfocused plenoptic images using a displacement intra prediction," in *2016 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pp. 1–4, IEEE, Jul 2016.

[7] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: graph Fourier transform," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 62, pp. 6167–6170, IEEE, May 2013.

[8] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, pp. 1644–1656, Apr 2013.

[9] T. Kanamori, K. Mishiba, Y. Oyamada, and K. Kondo, "Sparse representation using weighted graph fourier transform based on color-depth correlation for depth map compression," *2016 IEEE 5th Global Conference on Consumer Electronics*, pp. 1–2, Oct 2016.

[10] "The (new) Stanford light field archive." `http://lightfield.stanford.edu/lfs.html`. Accessed: 2017-04-18.

[11] M. Levoy, "Light fields and computational imaging," *Computer*, vol. 39, pp. 46–55, Aug 2006.

[12] A. Sandryhaila and J. M. Moura, "Big data analysis with signal processing on graphs: representation and processing of massive data sets with irregular structure," *IEEE Signal Processing Magazine*, vol. 31, pp. 80–90, Sep 2014.

[13] A. Sandryhaila and J. M. F. Moura, "Nearest-neighbor image model," in *2012 19th IEEE International Conference on Image Processing*, no. 3, pp. 2521–2524, IEEE, Sep 2012.