Graph-based light fields representation and coding using geometry information
Xin Su, Mira Rizkallah, Thomas Maugey, Christine Guillemot

To cite this version:
Xin Su, Mira Rizkallah, Thomas Maugey, Christine Guillemot. Graph-based light fields representation and coding using geometry information. ICIP 2017 - IEEE International Conference on Image Processing, IEEE, Sep 2017, Beijing, China. pp.4023-4027, 10.1109/ICIP.2017.8297038. hal-01589626

HAL Id: hal-01589626
https://inria.hal.science/hal-01589626
Submitted on 18 Sep 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
**GRAPH-BASED LIGHT FIELDS REPRESENTATION AND CODING USING GEOMETRY INFORMATION**

**Xin Su, Mira Rizkallah, Thomas Maugey and Christine Guillemot**

INRIA, Campus de Beaulieu, Rennes 35042, France

**ABSTRACT**

This paper describes a graph-based coding scheme for light fields (LF). It first adapts graph-based representations (GBR) to describe color and geometry information of LF. Graph connections describing scene geometry capture inter-view dependencies. They are used as the support of a weighted Graph Fourier Transform (wGFT) to encode disoccluded pixels. The quality of the LF reconstructed from the graph is enhanced by adding extra color information to the representation for a sub-set of sub-aperture images. Experiments show that the proposed scheme yields rate-distortion gains compared with HEVC based compression (directly compressing the LF as a video sequence by HEVC).

**Index Terms**— Graph Based Representation (GBR), Graph Fourier Transform (GFT), Compression, Light fields (LF).

1. **INTRODUCTION**

Light fields (LF) have emerged as a representation of light rays emitted by a 3D scene and received by an observer at a particular point \((x, y, z)\) in space, along different orientations. A variety of capturing devices have been designed based on camera arrays [1], on single cameras mounted on moving ganttries, or on arrays of micro-lenses placed in front of the photosensor to obtain angular information about the captured scene [2, 3].

The problem of LF compression rapidly appeared as quite critical given their significant demand in terms of storage capacity. Classical block-based coding schemes such as JPEG applied for each image of the 2D array of images forming the lumigraph have been quite naturally considered yielding however limited compression performances (compression factors not exceeding 20 for an acceptable quality) [4]. A method based on video compression is presented in [5] where a few views are encoded in Intra while the other views are encoded as P-images in which each block can be predicted from one of the neighboring Intra views with or without disparity compensation, the choice of the prediction mode being made to optimize a rate-distortion measure. A second scheme is presented where several predictions of a view are computed from neighboring views using disparity maps, and averaged to give the final predicted view. The prediction residue is then encoded using classical coding tools (DCT, quantization). Multiview video compression and disparity compensation techniques are considered in [5, 6], and intra coding modes have also been proposed in [7] for LF compression using HEVC. The authors of [8] exploits inter-view correlation by using a homography-based low rank approximation of the LF, showing significant gains compared to HEVC. Inter-coding for real LF captured by micro-lenses based devices.

In this paper, we explore the use of GBR for LF. GBR has been proposed for describing the geometry of multi-view images, first for horizontally aligned cameras [9] and more recently for complex camera configurations [10]. Here, we consider GBR to represent LF using 3D geometry information. The graph connections are derived from the disparity and hold just enough information to synthesize other sub-aperture images from one reference image of the LF. Based on the concept of epipolar segment, the graph connections are sparsified (less important segments are removed) by a rate-distortion optimization. The graph vertices and connections are compressed using HEVC [11]. The graph connections capturing the inter-view dependencies are used as the support of a Graph Fourier Transform [12] used to encode disoccluded pixels.

However, the graph mostly represents scene geometry. Texture information is limited to a reference view and disoccluded pixels, which is not sufficient for reaching a high reconstructed LF quality. The bitrate distribution between texture and geometry (i.e. depth) is a key issue in view synthesis from multi-view data and depends on the camera configuration [13]. To enhance the quality of the reconstructed LF, the residuals of a subset of views are added to the graph representation. Experiments with synthetic LF from the dataset in [14] rendered with Blender [15] show that the proposed scheme achieves higher reconstruction quality at low rates compared with traditional video compression by HEVC.

2. **LIGHT FIELDS GEOMETRY**

We consider the simplified 4D representation of LF describing the radiance along rays by a function \(L(x, y, u, v)\) of 4 parameters at the intersection of the light rays with 2 parallel planes. This representation can be seen as an array of multi-view images \(\{I_{u,v}\}\). Each view \(I_{u,v} \in \mathbb{R}^{X \times Y \times 3}\) at position \((u, v)\) is an RGB image with \(X \times Y\) pixels. Given a pixel \((x, y)\) in \(I_{u,v}\), its corresponding pixel in \(I_{u',v'}\) (the pixel corresponding to the same 3D point in the real world), should have the same color values under the Lambertian as-
From principle, multiple views of a scene can be rendered from one unique view with the help of scene geometry. This is the core idea of depth image based rendering (DIBR). For instance, given a LF dataset with available depth images \( \{ Z_{u,v} \} \), pixel \((x',y')\) in \( I_{u',v'} \) corresponding to the same 3D point as the pixel \((x,y)\) in \( I_{u,v} \) can be located by

\[
(x', y') = (x + d_x, y + d_y),
\]

\[
d_x = B \frac{(u - u') \cdot f}{Z_{u,v}(x,y)}, \quad d_y = B \frac{(v - v') \cdot f}{Z_{u,v}(x,y)},
\]

where \( B \) is the distance between neighboring cameras, \( f \) is the focal length, \( Z_{u,v}(x,y) \) is the depth of pixel \((x,y)\) in \( I_{u,v} \). View \( I_{u',v'} \) thus can be rendered pixel by pixel by Eq.(1).

\( (d_x, d_y) \) is also known as disparity. In the tests, we consider synthetic LF [14] for which depth information is available. For real LF, depth has to be estimated using for example the methods in [16, 17].

Pixels in different views corresponding to the same 3D point have same or similar color values. In this paper, we represent inter-view dependencies in LF with a graph using geometry information, and use the graph as a support to encode the color information using graph-based transform coding.

### 3. GRAPH REPRESENTATION

#### 3.1. Graph construction

Let us denote the graph by \( G = \{ V, E \} \), where vertices \( V = \{ v_i \} \) correspond to each pixel in sub-aperture images \( \{ I_{u,v} \} \), and edges \( E = \{ e_{ij} \} \) connect pairs of pixels across two images.

**Graph connections for reference image.** As shown in Fig.1.a, image \( I_{1,1} \) (left bottom corner image marked in red) is selected as the reference view. Pixels on each row of \( I_{1,1} \) are grouped into a set of straight horizontal segments based on their depth. One segment has a constant depth. As shown in Fig.1.b, one row in \( I_{1,1} \) has been divided into 3 segments. Every segment in \( I_{1,1} \) is connected to one segment in every sub-aperture image by one graph edge, since the two segments correspond to the same straight segment in the real 3D world. For instance, in the toy example in Fig.1.a, the reference view \( I_{1,1} \) is connected with every sub-aperture image \( I_{u,v} \) by graph edges. However, for one straight segment in \( I_{1,1} \), all its connections to other sub-aperture images can be deduced from each other by Eq.(1). Therefore, for one segment in \( I_{1,1} \), only one of its connections is necessary in the final graph structure. In our GBR, we only keep the graph connections between \( I_{1,1} \) and \( I_{1,2} \) (the right sub-aperture image of \( I_{1,1} \)), as shown in Fig.1.a, the connections marked as red solid line are kept and the other connections marked as black dotted lines are redundant and removed. Fig.1.c gives an illustration of the kept graph connections between \( \{ I_{1,1} \} \) and \( \{ I_{1,2} \} \).

To simplify the graph representation, each graph connection is represented by a one-dimensional metric namely unidimensional disparity based on the epipolar segment concept [10]. The epipolar segment is a line segment consisting of all possible projections of a pixel with varying depth. The unidimensional disparity actually is the distance between the start point of the epipolar segment and the position of the true projection.

**Graph connections for disoccluded pixels.** Besides the reference image \( I_{1,1} \), the disoccluded pixels which are not visible in \( I_{1,1} \) are also considered in the graph construction. For the sake of simplicity, we only consider the disoccluded pixels in \( I_{u,v} \) (the top right corner image in Fig.1.a), since most of the disoccluded pixels in images \( \{ I_{u,v} \} \) \((1 < u < U, 1 < v < V)\) are visible in \( I_{u,v} \). To construct the graph connections for the disoccluded pixels, the same strategy has been applied here. In other words, these disoccluded pixels are treated as “reference pixels” for other sub-aperture images.

#### 3.2. Graph sparsification

As presented in [10, 18], the constructed graph in section 3.1 is sparsified based on a rate-distortion model,

\[
J(\mathcal{E}) = D(\mathcal{E}) + \alpha R(\mathcal{E}),
\]

where \( J \) is the Lagrangian cost (smaller \( J \) values mean better optimal status), \( D \) is the distortion of rendered sub-aperture images and \( R \) is the modeled bitrate cost for coding the graph connections. \( \alpha \) is the Lagrangian multiplier which represents the relation between bitrate and rendering quality (distortion).

To decrease the computational cost we compute the rendering distortion on only a subset of views. Edges are removed based on the shortest path optimization of

\[
\mathcal{E} = \arg \min_{\mathcal{E}} J(\mathcal{E}).
\]
I is classically compressed using HEVC, the arbitrarily shaped whereas red ones are edges with high weight (1). While \( I_{1,1} \) is classically compressed using HEVC, the arbitrarily shaped \( I_{0,U,V} \) requires dedicated tools. We propose to compress it using a graph-based compression scheme as follows.

Let \( S = [S_1 S_2] \) be the vector of all color values in \( V \) to be coded, where \( S_1 \) comprises the color values of the reference image \( l_{1,1} \) (separately coded with HEVC), and \( S_2 \) the color values of the disoccluded pixels \( I_{0,U,V} \). The color values in the reference view are initially propagated to the disoccluded pixels using an iterative diffusion method. More precisely, at the first iteration, the pixels \( I_{1,1} \) are predicted using a weighted graph. At the second iteration, the pixels \( I_{0,U,V} \) are predicted using a weighted graph.

\[
\hat{I}_{u,v} = \frac{1}{\sum w_i} \left( w_{1} \hat{I}_{u,v} + \sum_{i=1}^{M} w_{i} \hat{I}_{u,v} \right),
\]

where \( w_{1} = \exp\left(-\frac{(u-m)^2 + (v-n)^2}{2}\right), i = 1, \ldots, M \) and weight \( w_{i} \) depends on the distance between image \( I_{u,v} \) and \( I_{m,n} \). The rendered result of \( I_{u,v} \) is computed as

\[
\hat{I}_{u,v} = \frac{\sum w_i \hat{I}_{u,v} + \sum_{i=1}^{M} w_{i} \hat{I}_{u,v} \hat{I}_{m,n}}{\sum w_i + \sum_{i=1}^{M} w_{i}},
\]

where \( \hat{I}_{u,v} \) is the rendered result of \( I_{u,v} \) from selected "reference image" \( \hat{I}_{m,n} \).

4. CODING SCHEME

The proposed encoder is shown in Fig. 2. As explained in Section 3, from two sub-aperture images, namely the corner images \( I_{1,1} \) and \( I_{U,V} \), we construct the LF graph representation \( G' = (V, E) \). The graph edges are stored in a grey-level image which is coded using HEVC. The vertices are pixels in the images \( I_{1,1} \) and \( I_{U,V} \) (the parts of \( I_{1,1} \) that do not appear in \( I_{1,1} \) ) . A part of the graph is depicted in Fig. 3 where blue segments are edges with a small weight (0.5) whereas red ones are edges with higher weight (1). While \( I_{1,1} \) is classically compressed using HEVC, the arbitrarily shaped \( I_{0,U,V} \) requires dedicated tools.

To code the prediction residuals of the disoccluded pixels, we use the weighted Graph Fourier Transform (wGFT) [12]. The target disocclusion image is divided in 8 x 8 pixel blocks. In each block, we use the 4-neighbors graph which connects the disoccluded pixels to transform the residuals. More specifically, given the weight matrix \( W \), we define the diagonal degree matrix \( D \), where \( D_{ii} = \sum_{j} w_{ij} \). Lastly, the graph normalized weighted Laplacian matrix \( L_{\text{norm}} \) is computed as \( L_{\text{norm}} = I - D^{-1/2} W D^{-1/2} \). Let \( \Psi \) be the matrix whose columns contain the wGFT basis, i.e., the eigenvectors of the graph normalized Laplacian. The residuals are thus projected on the wGFT basis as \( R = \Psi R \). The coefficients are quantized for various quality factors following the method in [19], entropy coded then sent to the decoder side.

Because the decoder already received the disparity information in the graph-based representation, it can deduce the exact same locations of disoccluded pixels in the target image as the encoder. For each 8 x 8 block, it builds the same 4-neighbors graph connecting the disoccluded pixels, computes the edge weights using the disparity information and derives the same transform. Therefore, there is no need to send additional side information as done in edge-adaptive approaches [20, 21].

Finally, the remaining views \( I_{u,v} \) are coded as follows. They are first predicted using the graph-based representation. Then, a residual is computed with the true \( I_{u,v} \). This residual
5. EXPERIMENTS

We test our GBR on synthetic LF (with $U = 9$, $V = 9$, $X = 768$ and $Y = 768$) from the dataset in [14] rendered with Blender [15]. Three datasets, called Buddha, StillLife and monasRoom, have been tested here.

5.1. Evaluation of GFT

To show the interest of exploiting inter-view neighboring relations (i.e., graph edges) in coding the disocclusions, we first compare the performance of our graph-based compression scheme against HEVC inter-coding. We first code the disoccluded parts along with the reference view as a video sequence using HEVC. We vary the QP from 0 to 40. For each QP, a prediction of the disocclusions is computed (Sec. 4), then the residuals are coded while varying the quality factor from 10 to 90. The bitrate is the one needed to code the disocclusions. The PSNR is measured taking as reference the original disocclusions color values. From the results (Fig. 4), we notice that our approach outperforms HEVC with a higher PSNR for most QP values while preserving acceptable bitrates. Our diffusion method yields a good prediction with Buddha since the background mostly consists of smooth regions, and that explains the better coding performance. Whereas for monasRoom, the background is made of texture and wrong color values are propagated to the disoccluded areas resulting in residuals harder to code.

5.2. Light field representation and compression

We perform the GBR representation with fixed Lagrangian multiplier $\alpha = 0.5$ in Eq.(2). In this case, the graph sparsification highly depends on the distortion term $D(\mathcal{E})$. The number of sub-aperture images selected to add residuals is chosen as $\{1, 4, 9, 21\}$ with a regular sub-sampling pattern. The baseline method is the scheme which directly compresses the whole LF dataset as a video sequence using HEVC. We vary the QP from 0 to 40. For each QP, a prediction of the disocclusions is computed (Sec. 4), then the residuals are coded while varying the quality factor from 10 to 90. The bitrate is the one needed to code the disocclusions. The PSNR is measured taking as reference the original disocclusions color values. From the results (Fig. 4), we notice that our approach outperforms HEVC with a higher PSNR for most QP values while preserving acceptable bitrates. Our diffusion method yields a good prediction with Buddha since the background mostly consists of smooth regions, and that explains the better coding performance. Whereas for monasRoom, the background is made of texture and wrong color values are propagated to the disoccluded areas resulting in residuals harder to code.

Fig. 5. PSNR-rate performance of the proposed GBR on different datasets, (a) buddha, (b) stillLife and (c) monasRoom.
ACKNOWLEDGEMENT

This project has been supported in part by the EU H2020 Research and Innovation Programme under grant agreement No 694122 (ERC advanced grant CLIM).

REFERENCES


