Lossless archiving view arrays from plenoptic cameras when camera sensor images are available

Ioan Tabus and Emanuele Palma
*Computing Sciences Unit
Tampere University, Tampere, Finland
Email: ioan.tabus@tuni.fi, emanuele.palma@tuni.fi

***Abstract*— We propose two lossless archiving methods for the light field (LF) array of views created by plenoptic cameras, when the camera sensor images (also called lenslet images) are available. The two archiving methods are based on generative mechanisms, where we encode all information needed to run the processing pipeline that generates the LF array of views starting from source sensor images. The first archiving method starts by encoding the two sensor images needed as input in the processing pipeline: one scene sensor image, and one white image (available from the camera database of white images). The second archiver encodes a single lenslet-like image obtained by devignetting and debayering the scene sensor image and the white image. After encoding the input lenslet images both methods proceeds to encode all additional meta-information necessary for running the processing chain, starting from sensor image to the final LF array of views, and any needed corrections due to possible quantization along the processing chain, finally creating a lossless archive of the light field array of views. We exemplify the performance for a database of plenoptic camera images that was extensively used in the light field lossless compression literature, obtaining competitive archive file sizes.**

I. Introduction

The compression of light field array of views is a well studied topic, covered in hundreds of publications, see for example the recent survey [1]. Although the interest to *lossy* compression dominates the research publications, there is also considerable interest for *lossless* compression of plenoptic camera images, see e.g., [2] and the references therein.

A plenoptic camera is acquiring in a single shot a lenslet image, containing information about the light filed sampled from $(L_r \times L_c)$ directions of view. The camera sensor image has a resolution $(M_r \times M_c)$ pixels. In front of the sensor image there is a Bayer mask, and in front of the Bayer mask there is an array of $(N_r \times N_c)$ microlenses. The image recorded under each microlens represents the light coming from a set of different directions.

Due to its particular structure, the sensor image is very redundant, and we have introduced in [2] a method named sparse relevant regressors and contexts (SRRC), which makes use of the redundancy specific to color filter array (CFA) images in one hand, and on the other hand we exploited the redundancy existing between the pixels under two consecutive microlenses, from the array of microlenses. Our main encoding tool in this paper is based on SRRC, placed in two archiving architectures, introduced in Sections II and III. In these schemes we start from encoding the sensor images, and then we run the processing pipeline for obtaining the LF array of views.

The blue dotted block in the right of Figure 1 shows the processing pipeline for creating a LF array of views, having as input two sensor images, $\mathbf{X}$ and $\mathbf{W}$, and proceeding through a devignetting stage (where the gray levels in the scene lenslet are divided by the gray levels in the white image and then are normalized), a debayering of the devignetted image, then resampling and rectification for finally obtaining the $(L_r \times L_c)$ array of views.

In the last couple of years a few alternative were proposed for changing the existing processing flow from [3], see e.g. [4] and references therein.

In this paper we propose two archiving schemes, under the name *Compress Image and Metadata for reconstruction of Light Field array of views* (CIMLF). Our scheme CIMLF^0 offers a functionality which is very useful in the perspective of future debayering and plenoptic processing methods: the encoding/decoding will start with transmitting the important sensor image data, in the form of $\mathbf{X}$ and $\mathbf{W}$, then the decoder will be able to apply itself several plenoptic processing schemes, will obtain several versions of the $\mathbf{LF}$ array of views, and will select the result that suits better his goals.

A. Organization of the paper

We start in Section II by presenting the overall structure of the CIMLF^0 archiver for the LF array of views, which uses the SRRC plenoptic camera lenslet image coder as its main component. Then we present in Section III other variants of the CIMLF overall light field archiving algorithm, where instead of encoding the input sensor images, we start by encoding an intermediate image and then continue in performing the rest of the plenoptic processing pipeline. Section IV exemplifies experimental results for efficiently archiving the light field array of views using the sensor images and metadata from the camera.

II. Overall structure of the light field encoder CIMLF

We show in Fig. 1 the diagram of our archiver CIMLF^0, which encodes in an archive: a) the input sensor image $\mathbf{X}$ representing the scene; b) the sensor image $\mathbf{W}$, which is a sensor image of a white sheet, taken at specific camera settings ($\mathbf{W}$ belongs to a small databases of white images of

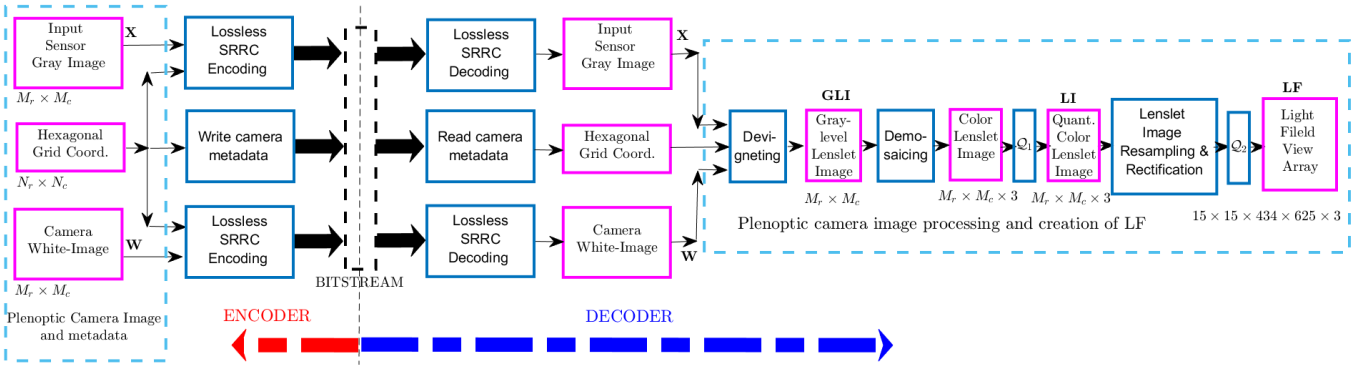

Fig. 1: Block scheme of CIMLF^0 (compress image and metadata for reconstruction of LF) archiver, which is archiving the input sensor images (using the SRRC coding block) and the metadata information, and at decoder it reconstructs the input sensor images and additionally it finally provides the efficient reconstruction of the LF array of views. The inputs are the sensor image of the scene, $\mathbf{X}$, and the sensor image of a white sheet, $\mathbf{W}$.

a given camera); and c) additional meta-information including the parameters needed to generate the hexagonal lattice of microlenses centers. After decoding this whole information, the decoder can run the plenoptic processing pipeline associated with the camera [3] (the right block delineated by blue lines). The matlab code specifying the pipeline operations and their additional optional setting parameters can also be packed within the initial archive (since this will increase the size of the archive by a negligible size, e.g., about 75 kB, representing only an additional 0.01 bpp for the bitrates reported in Table I).

In this way we can obtain self-extracting archives, from which we are able to reconstruct first the essential input information $\mathbf{X}$ and $\mathbf{W}$. Then the decoder can further run the plenoptic processing chain to reconstruct the final light field array of views, $\mathbf{LF}$, with the parameters provided by the encoder. Alternatively the decoder can run several other versions of the plenoptic processing chain, to reconstruct its own different versions of the light field array of views.

We are showing in the experimental section that in this way we can recreate the final LF array of views for each scene, using an archive of about 50 MB.

III. Variants of CIMLF coder for losslessly reconstructing a given light field view array

Instead of encoding two sensor images, $\mathbf{X}$ and $\mathbf{W}$, as we did in CIMLF^0, one can encode an intermediate image, namely the devignetted and demosaiced lenslet image $\mathbf{LI}$.

We exploit the fact that the plenoptic processing chain adopted by [5] creates the $\mathbf{LI}$ using single floating point operations, starting from the initial $\mathbf{X}$ and $\mathbf{W}$ on 10 bits, by first using devignetting to obtain an image which we denote $\mathbf{GLI}_0$, and then by demosaicing [6], to get a color lenslet image which we denote $\mathbf{LI}_0$, both images being stored in integer 16 bits format. After this, the color image $\mathbf{LI}_0$ is quantized with a quantizer $\mathcal{Q}_1$ to get the color image $\mathbf{LI}$, on 10 bits per color component [7].

If one tries to losslessly compress the lenslet image $\mathbf{LI}$, which has three color components, by using the efficient generic lossless color compression tool FLIF [8], one obtains a compressed size of 80 MB (or 13.96 bpp by normalizing with the number of pixels used in Table I), which is not competitive. However, we introduce a more efficient way to encode $\mathbf{LI}$, using in the first stage the SRRC codec, which is very efficient for sensor-like images (i.e., gray-level, recorded using a Bayer mask).

To start, we introduce an operation dubbed Bayer sampling, equivalent to applying an inverse demosaic operation to $\mathbf{LI}$, resulting in a gray-level image $\mathbf{GLI}$, which keeps on each of the four phases of the Bayer pattern the corresponding pixel from the color channel. As an example, with the Bayer pattern *grbg*, all pixels located at $GLI(2j_1+1, 2j_2+1)$ will extract the values from the green color component $LI(2j_1+1, 2j_2+1, 2)$, and in a similar way all different colors are sampled to a single gray level image, resulting in the formal definition of Bayer sampling:

$$\begin{aligned} GLI(2j_1+1, 2j_2+1) &= LI(2j_1+1, 2j_2+1, 2) \\ GLI(2j_1+1, 2j_2+2) &= LI(2j_1+1, 2j_2+2, 1) \\ GLI(2j_1+2, 2j_2+1) &= LI(2j_1+2, 2j_2+1, 3) \\ GLI(2j_1+2, 2j_2+2) &= LI(2j_1+2, 2j_2+2, 2). \end{aligned} \quad (1)$$

This is implemented by the block denoted "sub-sample to gray-level Bayer pattern" in Fig. 2. The image $\mathbf{GLI}$ is very well suited to be compressed by SRRC, and this is what we decide to losslessly encode, so the decoder will have a perfect copy of $\mathbf{GLI}$. However, when trying to reconstruct $\mathbf{LI}$ by demosaicing $\mathbf{GLI}$, we get only an approximate reconstruction.

Hence it appears that for reconstructing $\mathbf{LI}$ one can use the SRRC compression of $\mathbf{GLI}$ and use demosaicing to get $\widehat{\mathbf{LI}}$, but then one needs to additionally encode the color image of differences $\mathbf{LI} - \widehat{\mathbf{LI}}$. This two-step compression turns out to be more efficient than the direct compression of $\mathbf{LI}$.

A diagram of the encoding-decoding process involved is shown in Fig. 2.

Depending on the selection of the encoding blocks, we have experimented with three versions: a) the version denoted CIMLF^1, when the codec for the gray level lenslet image $\mathbf{GLI}$

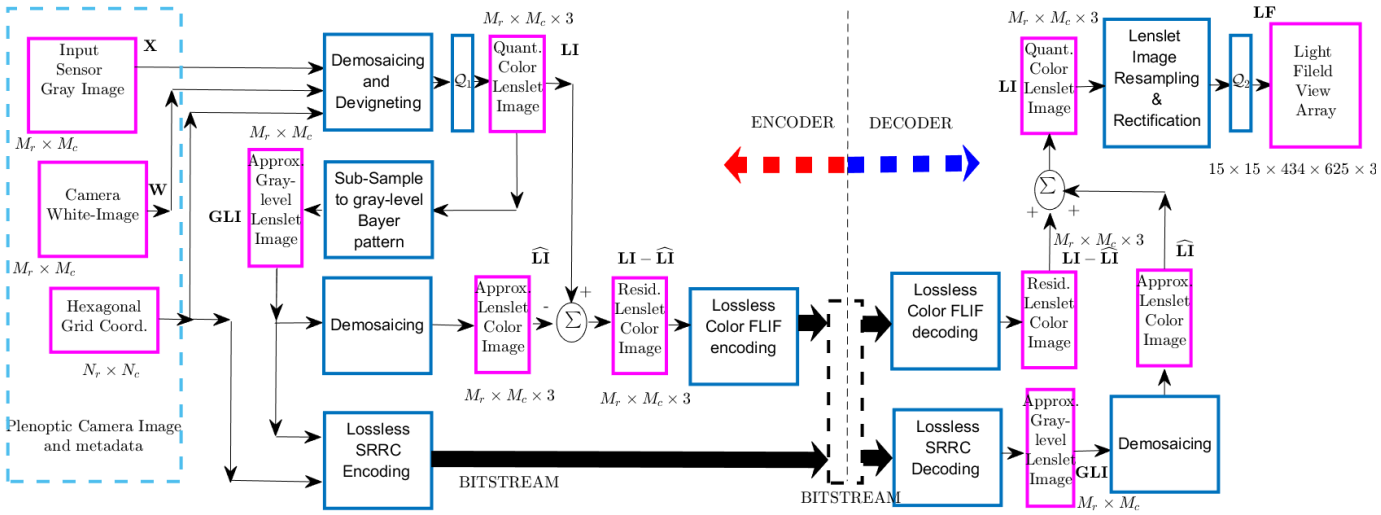

Fig. 2: Block scheme of CIMLF[1] for archiving image and metadata information for lossless reconstruction of the LF array of views. The SRRC codec encodes the **GLI** image, and a general use coder (e.g., FLIF, or JPEG 2000) encodes the difference $\mathbf{LI} - \widehat{\mathbf{LI}}$.

is the SRRC codec and the color difference $\mathbf{LI} - \widehat{\mathbf{LI}}$ is encoded by FLIF [8]; b) the version denoted CIMLF[2] when both **GLI** and $\mathbf{LI} - \widehat{\mathbf{LI}}$ are encoded by FLIF ; and c) the version denoted CIMLF[3], when both **GLI** and $\mathbf{LI} - \widehat{\mathbf{LI}}$ are encoded by JPEG 2000 codec.

IV. Experimental results

We report results on the light field datasets from [9]. We used the same 12 plenoptic scenes denoted $I_{01}, \ldots, I_{12}$, as in [2], where we have shown the results of encoding the sensor images **X** and **W** for each dataset.

A. Encoding the information needed for running the pipeline at the encoder and at decoder

When running the plenoptic pipeline for creating the LF array of views for one scene, one needs the following: the sensor image **X** of that scene and only the corresponding white image **W**, i.e. the white image for the particular mode used by the camera when taking the shot of the scene. Together **X** and **W** have in raw format 55.5+51.8= 107.3 MB (and for bikes we succeed to compress by SRRC and store them together in an archive of about 52.8 MB).

Apart of the white image **W**, one needs to transmit to the decoder only a very tiny amount of information contained in two configuration structures. First, the structure `LensletGridModel` contains the parameters for defining the centers of the lenslets. Second, the structure `DecodeOptions` contains parameters for the functions used by the matlab light field toolbox [3] to "decode" a LF array of views, from a lenslet image **LI**. Altogether the two structures in Lempel-Ziv compressed form require only 1091 bytes.

For decoding the LF array of views for a given scene, CIMLF[1] needs only the encoded version by SRRC of the gray level lenslet image **GLI**, the difference color lenslet image $\mathbf{LI} - \widehat{\mathbf{LI}}$ encoded by FLIF, and then also the two configuration structures. As an example for Bikes, $\mathbf{LI} - \widehat{\mathbf{LI}}$ is encoded by FLIF in 12.707 MB, **GLI** is encoded in 32.439 MB and the two configuration structures have 0.001 MB, resulting in total of 45.146 MB, packed by zip in an archive of 45.463 MB (including the info about file names in the archived folder). This archive is decoded losslessly into the 15×15 ppm files, identical to those stored at [9].

B. Performance of the CIMLF[0] archiver

The main task of the CIMLF[0] archiver is to encode the sensor images **X** and **W**, which is achieved by using SRRC [2]. The encoder also additionally transmits the meta-information needed for performing the operations of the plenoptic pipeline for obtaining the LF arrays. From this encoded bitstream, the decoder first reconstructs **X** and **W**, and then it can generate the final **LF** file of array of views.

We also included in the dearchiving program the processing pipeline for obtaining the LF array files that become part of the JPEG CTC datasets [5]. In this processing pipeline, the **LI** RGB image is quantized to 10 bits (this was the "input" data specified in the ICIP 2017 Grand Challenge). The output data is a 15×15 LF array of views identical to the 15×15 files stored at [9] (which include the 13×13 files specified in [5]). The results are shown in Table I. All results are shown in bits per pixel, with the number of pixels being $13 \times 13 \times 434 \times 625$ (to account only for data defined in [5]).

C. Performance of the CIMLF[1] archiver

The results of the CIMLF[1] archiver are shown in Table I, and they are better than the CIMLF[0] archiver in average by 1.31 bpp. In the same table we see results for CIMLF^2_{GLI}, which is not using at all the SRRC codec, but uses the FLIF codec for encoding both **GLI** and $\mathbf{LI} - \widehat{\mathbf{LI}}$, and its results are very good, but still worse by 0.18 bpp than CIMLF^1_{GLI}.

Finally, CIMLF$^3_{GLI}$ uses JPEG 2000 for encoding both $\mathbf{GLI}$ and $\mathbf{LI} - \widehat{\mathbf{LI}}$.

As an insight, we show in Table II what is the performance of the SRRC codec on the $\mathbf{GLI}$ images from Fig. 2, for the case when $\mathbf{GLI}$ is Bayer sampled (1) from the $\mathbf{LI}$ image quantized to 10 bits. We present the compression rates for SRRC with both patch labeling strategies, by Bayer phase and by depth, presented in [2]. All results are compressed file sizes per pixel, with the number of pixels being 5368×7728 for image $\mathbf{GLI}$ and $13 \times 13 \times 434 \times 625$ for image $\mathbf{LF}_{10}$. The encoders are: SRRC (sparse relevant regressors and contexts) [2]; FLIF method from [8]; and JP2 - JPEG 2000.

The last column shows the rate for the full reconstruction of the LF array for all images $I_{01}, \ldots, I_{12}$, obtained by encoding $\mathbf{GLI}$ with SRRC$_{GLI}$ (the phase classification variant), then encoding the difference color image and then applying the plenoptic processing, as shown in Fig. 2.

In order to provide reproducible results, we have saved the matlab code for the de-archiver from Fig. 1 and the necessary input bitstreams, in the form of archives for each scene, in the overall archive XW_Master.zip, and also have saved the matlab code for the de-archiver from Fig. 2 and the necessary input bitstreams, in the overall archive CTCDEC10B_Master.zip. The files and the corresponding installation instructions are uploaded at [10]. The decoder takes for each scene an input archive, and it generates the final LF array of 10 bit RGB images. All final files $\mathbf{LF}$ are lossless reconstructions of the files available at the JPEG Pleno database [11].

V. Discussion and conclusion

We can losslessly reconstruct the LF array of views using an average per pixel bitrate of 7.89 bpp for CILMF^1 and of 9.20 bpp for CILMF^0. Clearly, this could be achieved only by having available at the encoder (but not at the decoder) the input sensor images and some meta data information.

Only with title of illustration, we mention that for the same datsets, the lossless encoding bitrate is much higher (see, e.g., results comparing several methods in [12], indicating that all methods need more than 11 bpp for lossless reconstruction) if one uses generic lossless encoding programs, that do not make use of the sensor input image when encoding.

This shows that our proposed archiving method is a very efficient way to store the LF array of views obtained from plenoptic cameras (with the bonus of having inside the archive also the lossless version of the sensor images, which is useful if one later decides to use other plenoptic processing chains).

Acknowledgment

The work of Emanuele Palma was supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 764951.

TABLE I: Sizes of the files for reconstructing the LF array of 13×13 views quantized on 10 bits

Image	Images LF_{10}, color adjusted identical to CTC [5]			
Method	Reconstructed from Upstream Images:			
	CIMLF$^0_{X,W}$	CIMLF$^1_{GLI}$	CIMLF$^2_{GLI}$	CIMLF$^3_{GLI}$
I_{01}	9.23	**7.93**	8.07	9.18
I_{02}	9.13	**7.74**	7.87	8.99
I_{04}	9.13	**7.80**	8.03	9.03
I_{09}	9.30	**8.09**	8.37	9.52
Average	9.20	**7.89**	8.08	9.18

TABLE II: Coding results for the intermediate image $\mathbf{GLI}$ and for the final $\mathbf{LF}$ array of views.

Image	*GLI*, Quantized on 10 bits				LF_{10}
Method[a]	SRRC phase	SRRC Depth	FLIF	JP2	CIMLF^1
I_{01}	6.27	**6.26**	6.40	6.56	7.93
I_{02}	6.02	**6.00**	6.17	6.33	7.74
I_{03}	6.10	**6.08**	6.27	6.43	7.8
I_{04}	**6.09**	6.11	6.34	6.37	7.8
I_{05}	6.05	**6.00**	6.09	6.29	7.71
I_{06}	**6.28**	6.39	6.72	6.86	7.97
I_{07}	5.29	5.26	**4.46**	5.16	6.47
I_{08}	**6.30**	6.40	6.70	6.81	7.99
I_{09}	6.44	**6.43**	6.73	6.93	8.09
I_{10}	5.82	5.90	**5.47**	6.13	7.45
I_{11}	**6.28**	6.48	6.6	6.94	7.98
I_{12}	**6.34**	6.41	6.59	6.77	8.03
Average	**6.11**	6.14	6.21	6.47	7.75

References

[1] C. Conti, L. D. Soares, and P. Nunes, "Dense light field coding: A survey," *IEEE Access*, vol. 8, pp. 49 244–49 284, 2020.

[2] I. Tabus and E. Palma, "Lossless compression of plenoptic camera sensor images," *IEEE Access*, vol. 9, pp. 31 092–31 103, 2021.

[3] D. G. Dansereau, O. Pizarro, and S. B. Williams, "Decoding, calibration and rectification for lenselet-based plenoptic cameras," in *CVPR*, June 2013, pp. 1027–1034.

[4] P. Matysiak, M. Grogan, M. Le Pendu, M. Alain, E. Zerman, and A. Smolic, "High quality light field extraction and post-processing for raw plenoptic data," *IEEE Transactions on Image Processing*, vol. 29, pp. 4188–4203, 2020.

[5] F. Pereira, C. Pagliari, E. A. B. da Silva, I. Tabus, H. Amirpour, M. Bernardo, and A. Pinheiro, "JPEG Pleno Light Field Coding Common Test Conditions V3.3," Doc. ISO/IEC JTC 1/SC 29/WG1 N84049, 84th JPEG Meeting, Brussels, Belgium, 2019. [Online]. Available: https://jpeg. org/jpegpleno/documentation.html, 2019.

[6] R. Malvar, L.-W. He, and R. Cutler, "High-quality linear interpolation for demosaicing of Bayer-patterned color images," in *International Conference of Acoustic, Speech and Signal Processing*, May 2004.

[7] ISO/IEC JTC 1/SC29/WG1 JPEG, "JPEG Pleno Call for Proposals on Light Field Coding," in *ISO/IEC JTC 1/SC29/WG1 JPEG, Doc. N74014*, Jan 2017.

[8] J. Sneyers and P. Wuille, "FLIF: Free lossless image format based on maniac compression," in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 66–70.

[9] "JPEG Pleno Light Field Datasets according to common test conditions," http://plenodb.jpeg.org/lf/pleno_lf.

[10] I. Tabus and E. Palma, "Losslessly compressed light field datasets," http://ieee-dataport.org/2429.

[11] "JPEG Pleno Database: EPFL Light-field data set," http://plenodb.jpeg.org/lf/epfl.

[12] J. M. Santos, P. A. A. Assuncao, L. A. d. S. Cruz, L. M. N. Tavora, R. Fonseca-Pinto, and S. M. M. Faria, "Lossless compression of light fields using multi-reference minimum rate predictors," in *2019 Data Compression Conference (DCC)*, 2019, pp. 408–417.