An Untrained Neural Network Prior for Light Field Compression

Xiaoran Jiang, Jinglei Shi, Christine Guillemot

An Untrained Neural Network Prior for Light Field Compression

Xiaoran Jiang, Jinglei Shi, Christine Guillemot *Fellow, IEEE*

***Abstract*—Deep generative models have proven to be effective priors for solving a variety of image processing problems. However, the learning of realistic image priors, based on a large number of parameters, requires a large amount of training data. It has been shown recently, with the so-called deep image prior (DIP), that randomly initialized neural networks can act as good image priors without learning. In this paper, we propose a deep generative model for light fields, which is compact and which does not require any training data other than the light field itself. To show the potential of the proposed generative model, we develop a complete light field compression scheme with quantization-aware learning and entropy coding of the quantized weights. Experimental results show that the proposed method yields very competitive results compared with state-of-the-art light field compression methods, both in terms of PSNR and MS-SSIM metrics.**

***Index Terms*—light fields, compression, generative model, compact representation.**

I. Introduction

LIGHT field imaging has recently gained in popularity due to their potential for computer vision or computational photography applications. Compared to classical 2D imaging, light fields record the flow of rays in the form of large volumes of data, which retain both spatial and angular information of a scene. Several camera designs have been proposed for capturing light fields, from uniform arrays of pinholes placed in front of the sensor [1] to arrays of micro-lenses placed between the main lens and the sensor [2]–[4] and arrays of cameras [5], [6]. Some other designs use coded attenuation masks [7]–[9], with sparse reconstruction or deep learning methods [10], [11].

Light fields represent very large volumes of high-dimensional data. Finding effective but compact representations of light fields, that would capture both their spatial and angular redundancy, has therefore become a key challenge for practical use of this technology. This motivated the design of a variety of solutions, ranging from approaches extending HEVC Intra to directly compress the lenslet images [12], [13], to the compression of the set of views as pseudo-sequences using HEVC [14], [15], or using solutions based on 4D disparity-compensated transforms applied on spatio angular blocks [16], [17], [18], [19], [20]. A comparison of the performance of light field compression schemes using various video coding standards can also be found in [20]. Methods using view synthesis have also been proposed in [21] to synthesize all the views from a sparse set of input views, or in [22] where the authors use a linear approximation computed with Matching Pursuit for disparity based view prediction. Other view synthesis-based compression approaches have been proposed in [23] and in [24] using the Fourier Disparity Layer (FDL) representation introduced in [25], or exploiting the sparsity in the continuous Fourier domain as proposed in [26] respectively. Solutions using 3D representations have also been investigated as in [27] where the authors use an approximate 3D surface reconstruction to construct an eigen texture basis representation from the light field. Some approaches aim at providing scalability when coding light fields, such as in e.g., [28], [29], [30], [31] by designing layered compression schemes.

Neural Radiance Fields (NeRF) [32] have been recently introduced for light field view synthesis, hence, as other view synthesis methods, could be used as predictors in light field compression schemes, or directly as the light field representation. NeRF models, based on multi-layer perceptrons (MLP) are defined as models mapping continuous 5D vectors (3D coordinates plus 2D viewing directions) to volume density and view-dependent radiance. The model, trained to fit a set of input views, can be used to generate any view of the light field using volume rendering techniques. Many variants of NeRF have been proposed, to reduce the number of input views (e.g., [33]), or to generalize to new scenes (e.g. [34]). The authors in [35] first transform the 4D light field by leveraging Gegenbauer polynomials basis, and learn the mapping from these basis functions to color. The concept is further generalized to X-Fields in [36] defined as sets of 2D images taken across different view, time or illumination conditions. By limiting the novel viewpoints to be on the same side of the cameras, e.g., front views only, the NeuLF method in [37] aims at decreasing the inference time of NeRF, without sacrificing the rendering quality.

In this paper, we propose a neural network for compact light field representation. Like NeRF, it is untrained in the sense that it is learned only on the light field to be processed, without any additional training data. However, our motivation here was to design a lightweight network offering a good trade-off between the number of parameters, i.e. to decrease the bit rate needed to encode the light field representation, and the quality of the light field reconstruction. The proposed network is based on both a generative model that aims at modeling the spatial information that is static, i.e., found in all light field views,

This project has been in part supported by the EU H2020 Research and Innovation Programme under grant agreement No 694122 (ERC advanced grant CLIM), and in part by the French ANR research agency in the context of the artificial intelligence project DeepCIM.

X.Jiang, J. Shi and C. Guillemot are with the Inria Centre de Recherche Rennes - Bretagne Atlantique, Rennes 35042, France. (e-mail: xiaoran.jiang@inria.fr; jinglei.shi@inria.fr; christine.guillemot@inria.fr)

and on a convolutional Gated Recurrent Unit (ConvGRU) that is used to model variations between blocks of angular views.

The spatial view generative model is inspired from the deep decoder proposed in [38], itself built upon the deep image prior [39], proposed to solve inverse problems with 2D images. The model in [38] is a simpler under-parameterized model using filters of reduced support, proposed for compact image representation. Even if the compression performance of the deep decoder is not comparable to the performance that can be achieved with auto-encoders trained from large collections of images, as in [40]–[47], this untrained compact model can be applied to solve inverse problems, and not only compression.

We enhance the spatial generative model with spatial and channel attention modules, and with quantization-aware learning. The attention modules modulate the feature maps at the output of the different layers of the generator, following principles described in [48]. Our spatial information model also differs from the original deep decoder by the fact that it is formed by a set of layers common to all views within a block, hence modelling spatial information common to all views in a block, and by layers (or features) that are specific to each view.

The convolutional GRU models variations between angular views in light fields. The idea of disentangling static and transient information with deep neural networks has recently been explored in [49] for video frame prediction and [50] for video generation. However, while in [50] the latent space is trained using large training datasets, our input latent vectors do not require such optimization, since they are random vectors known from both the encoder and the decoder. In addition, we offer an option which expressively encodes the upscaling operations in learned weights in order to better fit the light field to process. The convGRU network takes randomly generated Gaussian noise as input, and produces a sequence of structured noise maps capturing variations across views, and this sequence is then fed to the adapted deep decoder based spatial generative model.

The weights of both the ConvGRU and the generator are learned end-to-end in order to minimize the reconstruction error of the target light field. The network weights can be considered as a representation of the input light field. The compactness of the representation obviously depends on the number of weights or network parameters, but not only. It also depends on the number of bits needed to accurately quantize each weight. Our network is thus learned using a strategy that takes into account weight quantization, in order to minimize the effect of weight quantization noise on the light field reconstruction quality.

We assess the rate-distortion performance of the quantization-aware learned representation for compression, in comparison with methods specifically designed for light fields, i.e., the prediction mode (4DPM) of JPEG-Pleno [51] and the method in [23], as well as with the encoding of the light field as a pseudo video sequence using video compression solutions, i.e., HEVC [52], [53]. We also considered recent deep learning video compression methods (the Hierarchical Learned Video Compression (HLVC) [54], the Recurrent Learned Video Compression (RLVC) [55] methods, and the OpenDVC [56] solution based on DVC [57]). While achieving very good distortion-rate performance, such motion estimation based deep compression networks, as well as similar methods such as in [58]–[61], often have complex structures, thus are not easy to train, and often need pre-trained optical flow estimators.

We also compared the compression performance of the model with two versions of quantized neural radiance field models (quantized NeRF), the NeRF model of [32] and a version with a reduced number of parameters, which we call NeRF-Slim in the paper. With NeRF-Slim, we show that one can indeed decrease the number of parameters, however at the expense of a loss in terms of reconstruction quality. We also assessed the interest of using ConvGRU to exploit angular view correlation, in comparison with the use of CoordConv [62] principles making the network aware of the coordinates of the data to process.

Our experimental results show that our method can achieve very good rate-distortion performance, outperforming very recent deep video compression methods requiring training on large datasets. It is also competitive against the standardized and highly optimized HEVC video compression tools applied to the sequence of views, as well as JPEG Pleno.

In summary, our contributions are as follows:

- We propose a novel deep generative model for light fields, based on a convolutional GRU modeling variations across the views, and on an adapted deep decoder modeling the spatial view information. The model is sufficiently compact to give convincing rate-distortion performance in compression applications.
- By introducing attention mechanisms and learned upsampling operations, our network is capable of generating real world light fields of good quality.
- We develop a complete light field compression method using this novel representation model learned in a quantization-aware manner in order to minimize the impact of weight quantization on light field reconstruction.
- We show that the resulting light field compression algorithm yields very good rate-distortion performances compared with state-of-the-art methods.

II. Network Formulation

A. Network overview

Let us consider an input light field, represented by a 4D function $L(x, y, s, t)$ describing the radiance along rays, with the two plane parameterization proposed in [63], [64]. The parameters (s, t) denote the angular (view) coordinates and (x, y) the spatial (pixel) coordinates. This 4D light field representation can be seen as an array of viewpoints (called sub-aperture images v) of the scene with varying angular coordinates s and t, or as a sequence of images $L = \left[L^1, L^2, ..., L^v\right]$. Our goal is to develop a deep generative model with few parameters for light fields, that is able to capture not only statistics within each sub-aperture image, but also correlation between the different viewpoints.

The proposed deep network architecture is shown in Fig. 1. The proposed network follows the principles of deep generative models which aim at transforming a randomly chosen

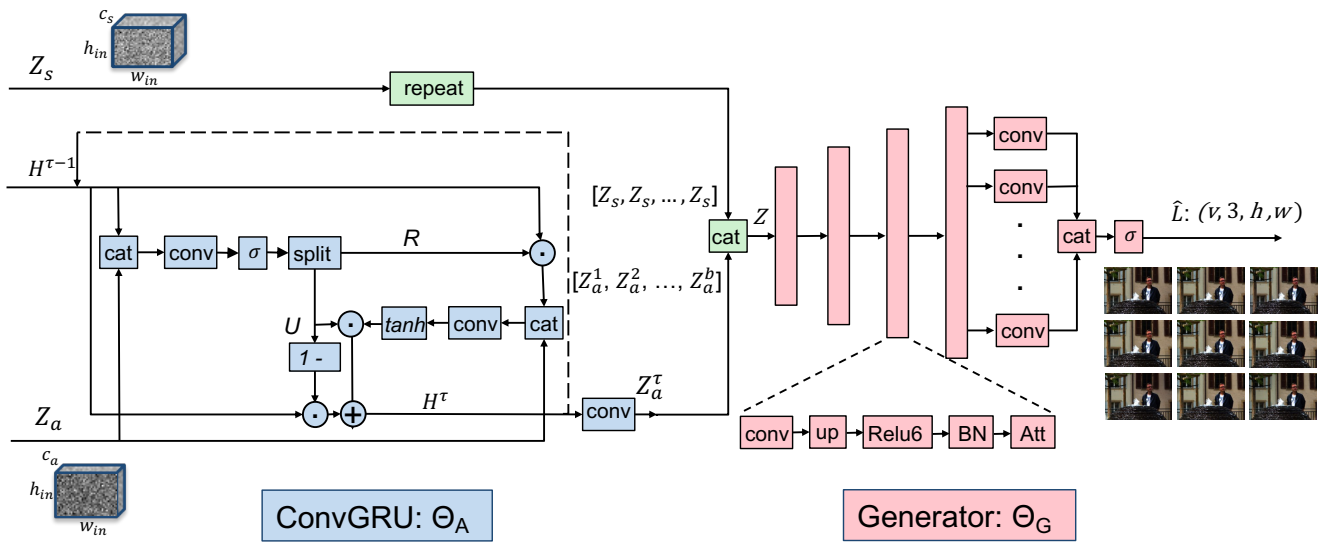

Fig. 1. Network overview. To process a light field with a large number of views, the light field is divided into b blocks of sub-aperture views. The convGRU network Θ_A takes randomly generated Gaussian noise volumes $Z_a \in \mathbb{R}^{c_a \times h_{in} \times w_{in}}$ as input, and produces a sequence of b structured noise volumes $[Z_a^1, Z_a^2, ..., Z_a^b]$, each volume being of dimension $(c_a \times h_{in} \times w_{in})$ capturing inter-block angular variations. This sequence is concatenated with b repeated noise volumes $Z_s \in \mathbb{R}^{c_s \times h_{in} \times w_{in}}$ corresponding to shared spatial information across the b blocks of views, and then fed in parallel to the spatial generative model Θ_G, which produces the light field views within each block. The model Θ_G consists of a decoder with several elementary blocks composed of convolution, upsampling, non-linear activation, batch normalization and attention modulation operators. The last layer uses distinct convolution filters to generate different views within a block based on the same input feature maps.

input noise map into image data. The proposed model is formed by a ConvGRU, denoted Θ_A in Fig. 1, that aims at modeling variations across the light field views followed by a deep generative model (Θ_G) mapping a sequence of code vectors in a latent space $Z = \left[Z^1, Z^2, ..., Z^v\right]$, to the views of the light field L.

Given the high correlation between light field views, especially for light fields with narrow baselines, one can consider that a light field static latent space, which contains shared spatial information of all the views, should be disentangled from the angular latent space, which contains the angular information differing from one view to another. Thus, the code vectors in the latent space can be expressed as:

$$Z = \left[\begin{bmatrix} Z_s \\ Z_a^1 \end{bmatrix}, \begin{bmatrix} Z_s \\ Z_a^2 \end{bmatrix}, ..., \begin{bmatrix} Z_s \\ Z_a^v \end{bmatrix} \right] \tag{1}$$

with $Z_s \in \mathbb{R}^{c_s \times h_{in} \times w_{in}}$ being the shared spatial latent code, and where Z_a^1, Z_a^2, ..., Z_a^v denote the angular latent codes corresponding to the different viewpoints. We will see in the next section that the sequence of latent codes Z_a^1, Z_a^2, ..., Z_a^v can be generated from one unique code $Z_a \in \mathbb{R}^{c_a \times h_{in} \times w_{in}}$, using a ConvGRU. The quantities c_a and c_s denote the numbers of channels of the input angular code vectors and that of the input spatial code vectors respectively. These $c_a + c_s = c_{in}$ channels are fed to the generator Θ_G to reconstruct the light field. h_{in} and w_{in} denote the spatial dimensions of the input noise maps.

The light field $\hat{L} \in \mathbb{R}^{v \times c \times h \times w}$, with v views, c color channels, and of spatial resolution $h \times w$, is then reconstructed from these disentangled latent codes via the network Θ_G inference $\hat{L} = \Theta_G(Z)$, where $Z \in \mathbb{R}^{c_{in} \times h_{in} \times w_{in}}$. Given that the angular component of Z is inferred from Z_a via the ConvGRU, the light field is actually reconstructed via an inference based on the whole network Θ, which can be written as

$$\hat{L} = \Theta(Z_s, Z_a) \tag{2}$$

The weights of both the convGRU Θ_A and the generator Θ_G are learned end-to-end in order to minimize the energy E as

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}}\, E(\Theta(Z_s, Z_a), L) \tag{3}$$

where the energy E can be defined by the mean square error (MSE) between the original light field and the reconstructed one, or using other metrics such as the Multi-Scale Structural Similarity (MS-SSIM).

As the MS-SSIM value increases as the image quality increases, the energy E is defined as the opposite of the MS-SSIM value: $E = 1 - MS\_SSIM\_value$. N_Θ, the number of network weight parameters is much smaller than the number of pixels in the light field, $N_\Theta \ll v \times c \times h \times w$, making the network a compact representation of the light field. Note that the only information we need to learn the network weights is the target light field L. In other words, the training of the entire network is self-supervised and does not need any external training data.

Note that the code vectors Z_a and Z_s in the latent space of the generative model are here noise map volumes generated from a standard normal distribution (see Fig. 1).

B. Angular prior utilizing a convolutional Gated Recurrent Unit

We use a convolutional Gated Recurrent Unit (ConvGRU) to model the variations between the light field angular views.

The ConvGRU unit generates, from a single latent code Z_a, the sequence of angular latent codes $\left[Z_a^1, Z_a^2, ..., Z_a^v\right]$. The ConvGRU is an efficient recurrent neural structure for sequential learning utilizing a gated mechanism as in a long short-term memory (LSTM), but with fewer parameters.

At a given step τ corresponding to a particular view, we compute

$$C = \sigma(\text{conv}(\text{cat}[Z_a, H^{\tau-1}])) \tag{4}$$

where σ denotes the sigmoid activation function to scale the output between 0 and 1, $H^{\tau-1}$ the hidden state of the previous step, cat[.,.] the concatenation operation, and conv() the convolution operation. The resulting tensor C is then split into two parts, the reset gate R and the update gate U:

$$[R, U] = \text{split}(C). \tag{5}$$

The current memory content can be computed as

$$M = \tanh(\text{conv}(\text{cat}[Z_a, R \odot H^{\tau-1}])), \tag{6}$$

with $\tanh$ being hyperbolic tangent activation and $\odot$ the Hadamard element-wise multiplication. The final memory at the current step τ is updated by element-wise multiplication using the update gate U, and the current latent map is computed after convolution as

$$\begin{aligned} H^{\tau} &= (1-U) \odot H^{\tau-1} + U \odot M, \\ Z_a^{\tau} &= \text{conv}(H^{\tau}). \end{aligned} \tag{7}$$

In our experiments, as a light field usually contains a relatively large number of sub-aperture views, learning a long sequence of angular latent codes, one angular latent code per view, can be expensive both in terms of memory and time consumption. In order to reduce the sequence length, we choose to process the light field views by block of views. We divide the light field into b blocks of n views each, with $b \times n = v$. If the light field is sufficiently dense, which is the case for real world light fields captured by plenoptic cameras, it is reasonable to suppose that within each block, the views can share the same angular latent code without loosing too much in terms of reconstruction quality. Therefore, the light field is reconstructed by blocks:

$$\hat{L} = \left[\Theta\left(\begin{bmatrix} Z_s \\ Z_a^1 \end{bmatrix}\right), \Theta\left(\begin{bmatrix} Z_s \\ Z_a^2 \end{bmatrix}\right), ..., \Theta\left(\begin{bmatrix} Z_s \\ Z_a^b \end{bmatrix}\right)\right]. \tag{8}$$

where $Z_a^{\beta}, \beta = 1 \dots b$ denotes the angular code vector for the block β.

Furthermore, the network parameters Θ can be split into two disjoint parts: Θ_A, the ConvGRU network that captures the inter-block level angular prior of the light field, and Θ_G, the generator network that represents a spatial prior for the light field views, as well as some angular variations within a light field block β. Eq.(4-7) can be re-written as

$$Z_a^{\beta} = \Theta_A(Z_a, \beta), \tag{9}$$

and the light field is thus reconstructed as

$$\begin{aligned} \hat{L} = \Bigg[\Theta_G\left(\begin{bmatrix} Z_s \\ \Theta_A(Z_a, 1) \end{bmatrix}\right), \Theta_G\left(\begin{bmatrix} Z_s \\ \Theta_A(Z_a, 2) \end{bmatrix}\right), \\ ..., \Theta_G\left(\begin{bmatrix} Z_s \\ \Theta_A(Z_a, b) \end{bmatrix}\right)\Bigg]. \end{aligned} \tag{10}$$

Inter-block angular latent codes Z_a^{β} inferred by Θ_A form a sequence of structured feature volumes which are concatenated with the shared spatial latent code Z_s. The resulting feature volume is then fed into Θ_G, which infers the light field views per block of views.

C. Generator network

In this section, we describe the generator network Θ_G that maps the structured noise generated by the ConvGRU network to the final light field. If the ConvGRU network Θ_A works at the inter-block level, the generator Θ_G works within each light field view block. For each block, the static latent code Z_s which is shared by all the views in the light field, is concatenated with the angular latent code of the block β, Z_a^{β}, i.e. shared by the views within the block β. For each block β, the generator maps the concatenated latent code $Z^{\beta} = \begin{bmatrix} Z_s \\ Z_a^{\beta} \end{bmatrix}$, to the n light field views of the block. This generator design should satisfy two conditions: 1) it should capture the spatial information of a single light field view; 2) it should be able to differentiate the different views within a block, even by taking the same input block-level code Z^{β}.

Similar to [38], a decoder structure consisting of several elementary structures Str_i is used. At each level i, Str_i transforms the input feature maps $\mathbf{F}_i$ to $\mathbf{F}_{i+1}$ in the way that the spatial resolution of the feature maps are doubled:

$$\mathbf{F}_{i+1} = \text{Str}_i(\mathbf{F}_i) \tag{11}$$

Each elementary structure Str_i contains a sequence of operations: one convolutional layer with kernel size 3×3, one upsampling layer with scale factor 2, one non-linear activation layer (rectified linear units) ReLU6, one batch normalization layer and finally an attention module A.

$$\mathbf{F}_{i+1} = A(\text{BN}(\text{ReLU6}(\text{up}(\text{conv}(\mathbf{F}_i))))). \tag{12}$$

The convolutional and upsampling layers can be replaced by a pixel-shuffle layer to achieve more accurate reconstruction, more details being explained in Section II-E.

Note that the main parameter overhead in each block resides in the convolutional layer, which gathers both cross-channel and spatial information. The n views in a block share the same filters, except the last convolutional layer. In the last layer, $c_{\text{in}} \times c \times n$ filters are learnt, which can be considered as n independent branches of $c_{\text{in}} \times c$ filters, where $c_{in} = c_a + c_s$ is the total number of input channels of the generator Θ_G, and c is the number of color RGB channels of the output views. These distinct filters enable to generate different sub-aperture views based on the same input feature maps. It is preferable to consider this structure instead of a second ConvGRU at this stage for two reasons: 1) the dimension of the feature maps

at this last stage of generation is much higher than the one of the input latent codes. Using a second ConvGRU would cause memory issues; 2) within each light field block, the angular variation across views is very limited, thus it is reasonable to only disentangle the filters of the last layer.

The upsampling layer performs bi-linear interpolation without weight parameters, whereas BatchNorm layers and attention modules are very light-weight. We use ReLU6 which clips the maximal activation at the value of 6, instead of the conventional ReLU layer. We have observed in our experiments that ReLU6 limits the dynamics of the activation, and thus enables to reduce quantization errors, especially when the quantization is coarse.

D. Attention mechanism

Three dimensional feature volumes $\mathbf{F}_i \in \mathbb{R}^{c_i \times h_i \times w_i}$ are generated at the output of each intermediate convolutional layer. Modulating features in order to favor the most relevant ones for the targeted task has been found to be useful to improve network efficiency [48], [65]–[69]. Instead of directly processing the 3D feature volumes, which involves much more computational and parameter overhead, we chose to use the strategy presented in [48] which consists in sequentially computing the modulation weights for the cross-channel 2D feature maps by the channel attention module A_c, and then computing those for the spatial 2D feature maps by the spatial attention module A_s, making the CNN aware of "where" to focus in each feature map. The obtained attention maps are element-wise multiplied with the feature maps. The overall process is summarized as follows:

$$\begin{aligned} \mathbf{F'}_i &= A_c(\mathbf{F}_i) \odot \mathbf{F}_i \\ \mathbf{F''}_i &= A_s(\mathbf{F'}_i) \odot \mathbf{F'}_i, \end{aligned} \tag{13}$$

with $\mathbf{F'}_i$ and $\mathbf{F''}_i$ being the resulting modulated feature volumes at layer i.

The channel attention module performs average pooling and max pooling to aggregate spatial information for each feature map and to obtain a vector of c_i values. In order to extract meaningful information, this vector should be further compressed to a vector of smaller dimension c'_i, with $c'_i = \frac{c_i}{r}$, $r > 1$ being the reduction ratio. To achieve this, a simple multi-layer perceptron (MLP) structure with two fully connected layers is applied, with c_i being the number of input and output neurons, and c'_i being the number of hidden neurons. At the end, we obtain c_i modulation weights, one for each feature map, which are used to modulate the different feature maps. The function performed by the channel attention module can be expressed as:

$$A_c(\mathbf{F}_i) = \sigma(\text{MLP}(\text{MaxPool}^s(\mathbf{F}_i)) + \text{MLP}(\text{AvgPool}^s(\mathbf{F}_i))) \tag{14}$$

with σ being sigmoid activation used to keep the modulation weights between 0 and 1. MaxPool^s and AvgPool^s are respectively max pooling and average pooling operations across the spatial dimension.

A similar processing is performed by the spatial attention module. Pixel-wise pooling operations are performed across the channel dimension, and convolutions are performed on the resulting feature map to obtain the final spatial attention map of dimension $h_i \times w_i$.

$$A_s(\mathbf{F}_i) = \sigma(\text{conv}(\text{cat}[\text{MaxPool}^c(\mathbf{F}_i), \text{AvgPool}^c(\mathbf{F}_i)])) \tag{15}$$

The symbols MaxPool^c and AvgPool^c represent max and average pooling operations across channels, whereas $\text{cat}[.,.]$ denotes a concatenation along the channel dimension.

E. Upsampling and pixel-shuffle

In the generator network, each elementary structure Str_i contains a convolutional layer followed by a parameter-free upsampling layer performing a $2\times$ bi-linear interpolation of the resulting feature maps. In a compression context, this design yields a satisfying reconstruction quality while maintaining a relatively low bit-rate. However, for higher bit-rates, in order to obtain more accurate reconstructed images, it is possible to learn the interpolation operation in a similar way as proposed in [70]. Instead of a convolutional layer of size $c_i \times c_i \times k \times k$ with c_i the number of input and output channels at level i, and k the kernel size, followed by a handcrafted upsampling operation, one can use a pixel-shuffle layer containing $c_i \times 4c_i \times k \times k$ kernel parameters and a pixel re-arrangement operation, which remaps the elements of a $4c_i \times h_i \times w_i$ tensor to a tensor of dimension $c_i \times 2h_i \times 2w_i$.

Note that, when using the handcrafted upsampling operation, increasing the depth of feature maps can also improve the reconstruction quality. However, we will demonstrate in Section IV-C3 that the pixel-shuffle layer is more effective with an equivalent parameter overload. Moreover, the pixel-shuffle scheme is more memory-friendly for back-propagation since despite increasing the network size, it keeps the quantity of feature maps unchanged at each level.

III. Quantization-aware learning

Having a light-weight neural structure is essential for our goal of finding a compact representation of light fields. As an example, let us suppose that we use $c_a = 15$ latent maps to generate angular information, and $c_s = 30$ latent maps to generate spatial and static information, and each light field block contains $n = 9$ views. The corresponding structure details and the number of parameters of the network are depicted in Table I, where the layers without parameters are omitted. The network contains 109427 weights. If we reconstruct a light field of 81 views of 512×512 pixels via the network, the network is indeed a highly under-parameterized model. We obtain a compression ratio of approximately 0.005 weight per pixel.

These weights can be further quantized to reduce the size of the model. Instead of using fixed-point scalar quantization which maps the weights to uniformly spaced codewords, we learn the codebook using the k-means algorithm with a fixed number γ of codewords, as classically done in image or network compression schemes, such as e.g. in [71]. Since k-means clustering can be computationally expensive compared to one iteration of back-propagation, this clustering is only performed once before training. The codebook of each layer

	k	*s*	*in/out*	*# params*
Θ_A				
$conv_1$	3	1	35/40	12600
$conv_2$	3	1	35/20	6300
$conv_3$	3	1	20/15	2700
Total Θ_G	-	-	-	21600
Θ_G				
$conv_1$	3	1	45/45	18225
BN_1	-	-	-	90
mlp_A_{c1}	-	-	45/9, 9/45	810
$conv\_A_{s1}$	7	1	2/1	98
$conv_2$	3	1	45/45	18225
BN_2	-	-	-	90
mlp_A_{c2}	-	-	45/9, 9/45	810
$conv\_A_{s2}$	7	1	2/1	98
$conv_3$	3	1	45/45	18225
BN_3	-	-	-	90
mlp_A_{c3}	-	-	45/9, 9/45	810
$conv\_A_{s3}$	7	1	2/1	98
$conv_4$	3	1	45/45	18225
BN_4	-	-	-	90
mlp_A_{c4}	-	-	45/9, 9/45	810
$conv\_A_{s4}$	7	1	2/1	98
$conv_5$	3	1	45/27	10935
Total Θ_G	-	-	-	87827
Total Θ	-	-	-	109427

TABLE I
THE DETAIL OF PARAMETER NUMBERS OF A NETWORK EXAMPLE (LAYERS WITHOUT PARAMETERS ARE OMITTED). k, s AND *in/out* REPRESENT THE KERNEL SIZE, THE STRIDE AND THE NUMBER OF INPUT/OUTPUT CHANNELS. THIS EXAMPLE IS GIVEN FOR THE FOLLOWING CONFIGURATION: $c_a = 15$, $c_s = 30$ AND EACH LIGHT FIELD BLOCK CONTAINS $n = 9$ SUB-APERTURE VIEWS.

is then updated by averaging the gradients of their assigned elements with gradient steps as

$$\mathbf{c} \leftarrow \mathbf{c} - \eta \frac{1}{|J_{\mathbf{c}}|} \sum_{w \in J_{\mathbf{c}}} \frac{\partial E}{\partial w}. \tag{16}$$

where E is the energy function to minimize, which computes the light field reconstruction error, and η is learning rate. $J_{\mathbf{c}}$ is the set of weights w which are assigned to the codeword $\mathbf{c}$.

Achieving good reconstruction performance despite quantization can be challenging especially when the number γ of codewords per layer is small. Indeed, the quantization error can be accumulated in a neural network, since subsequent layers take as input the activation of the preceding layers, which are corrupted by quantization noise. In order to alleviate this issue, as proposed in [72], we sequentially quantize layers from the lowest to the highest, and finetune the upper layers after the lower layers have been quantized. Finally, we apply entropy coding, e.g. Huffman coding, on quantized weights to achieve further compression of the model.

Overall, for each neural layer, we transmit γ codewords, with a cost of $\gamma \times 32$ bits, each codeword being encoded in 32 bits. We also transmit the entropy $\mathcal{H}$ of the indices which associate each weight to its corresponding codeword. The entropy is computed as $\mathcal{H} = -\sum_j \frac{|J_{\mathbf{c}_j}|}{N_i} \log_2 \frac{|J_{\mathbf{c}_j}|}{N_i}$, with N_i the total number of parameters in a certain layer i.

IV. EXPERIMENTAL RESULTS

A. Settings

The encoding of a light field proceeds in three steps. First, we train an uncompressed model from scratch. We begin with a learning rate of 0.01, and after every 8000 epochs, the learning rate is decreased by a factor of 0.6. Second, we sequentially quantize weights and finetune the model layer per layer with a fixed number γ of codewords per layer. In this work, we fix $\gamma_A = 64$ for ConvGRU network and $\gamma_G = 256$ for the generator network, as they have been found to meet a good rate-distortion trade-off. The same learning rate schedule is applied as in the first step. We have observed in the experiments that following the "Train-Finetune/Quantize" process can give better image quality than directly performing the quantization-aware training from scratch. Third, entropy coding is applied to further compress the quantized weights. In our experiments, Huffman coding is used.

The test light fields are real world light fields captured by a Lytro Illum camera, which are widely used by the light field compression research community. We compare our compression scheme against

- Solutions based on video coding standards, such as the HEVC coding standard [52], [53]. The HEVC version used in the tests is HM-16.10. The base QPs are set to 20, 25, 30, 35 and a GOP of 4 is used.
- Learning-based video compression methods (HLVC [54], RLVC [55] and OpenDVC [56]). We use the code and the PSNR-tuned and MS-SSIM-tuned models provided by the authors, with their default settings. For PSNR-tuned models, BPG [73] is used to compress I-frames, whereas for MS-SSIM-tuned models, the method in [74] is used to compress I-frames. For RLVC, 6 P-frames are encoded both in the forward and backward directions, which corresponds to GOP $=$ 13 (bi-IPPP). HLVC predicts images with three hierarchical quality layers and the defaut GOP$=$ 10 is used. For OpenDVC, the default GOP is also set to 10, and an inter-coded image is predicted from the previous decoded image. To obtain different bitrates with the OpenDVC, RLVC and HLVC methods, λ =8, 16, 32 and 64 and λ =256, 512, 1024 and 2048 are chosen for the MS-SSIM and PSNR models respectively, λ being the hyperparameter controlling the trade-off between distortion and bit-rate.
- Solutions specifically designed for light field compression, i.e., JPEG pleno [51], and FDL [23] with hierarchical scheme. For JPEG Pleno, the software version used is the JPEG Pleno Verification Model 2.0. The prediction mode with WaSP is used.

We also compare our scheme against a quantized version of the NeRF model originally proposed in [32], as well as a version of reduced dimension that we developed and called NeRF-Slim. The original NeRF model [32] has 8 fully connected layers, with a layer width of 256, and each pixel is synthesized based on 128 samplings along the ray. NeRF-Slim is also composed of 8 fully connected layers, but with layers of width 134 and the same sampling along the rays. The quantized versions of these two models are denoted as

NeRF-Quant and NeRF-Slim-Quant. The models are quantized layer by layer, with 32, 64, 128 or 256 centroids per layer, to obtain NeRF-Quant and NeRF-Slim-Quant models. The same quantization-aware learning as used in our model is applied.

B. Performance

Rate-distortion: The rate-distortion curves are shown in Fig.2 and Fig.3. The image quality is evaluated in terms of PSNR (Fig.2) and Multi-Scale SSIM (MS-SSIM) (Fig. 3) respectively. The bit-rate is computed in bits per pixel (bpp). The PSNR curves show that, at moderate and high bit-rates, averaged on the test light fields, our compression scheme achieves the best light field image quality among all compared methods, whereas at low bit-rate, HEVC obtains the best performance. Note also that our PSNR-tuned models achieve better quality than other learning-based reference methods (OpenDVC, HLVC, RLVC, NeRF-Quant and NeRF-Slim-Quant) for the entire bit-rate range. Table II shows BD-PSNR gains (using the Bjontegaard measure) with respect to the HEVC baseline. Our method outperforms the non-learning methods HEVC and FDL, as well as the learning-based methods HLVC, RLVC, OpenDVC and NeRF-Quant, and is comparable with JPEG Pleno.

In terms of MS-SSIM, Fig. 3 shows that for most of the test scenes, our MS-SSIM-tuned models reach the best rate-distortion trade-off for the entire bit-rate range. Note that both PSNR-tuned and MS-SSIM-tuned models of RLVC, HLVC and OpenDVC are publicly available.

TABLE II
BD-PSNR GAINS WITH RESPECT TO HEVC BASELINE. THE GAINS ARE SHOWN FOR THE NON-LEARNING METHOD JPEG PLENO [51], FDL [23] AND FOR LEARNING-BASED METHODS HLVC [54], RLVC [55], OPENDVC [56], NERF [32]-QUANT AS WELL AS OUR PROPOSED METHOD. TEST LIGHT FIELDS ARE BIKES, DDM (DANGER_DE_MORT), SPO (STONE_PILLARS_OUTSIDE), FV2 (FOUNTAIN_&_VINCENT_2), FRIENDS AND VESPA.

LF	JPEG Pleno	HLVC	RLVC	OpenDVC	NeRF -Quant	FDL	Ours
Bikes	-0.36	-0.83	-0.68	-1.62	**0.45**	-0.45	0.03
DdM	**0.67**	-0.74	-0.52	-1.52	0.04	0.50	0.55
SPO	0.16	-0.71	-0.42	-1.18	-0.92	**0.51**	-0.32
FV2	-0.08	-0.82	-0.53	-1.42	-0.49	-1.90	**0.24**
Friends	-0.24	-1.23	-0.95	-1.98	-1.68	**0.56**	-0.62
Vespa	**1.25**	0.36	0.64	-0.26	0.97	1.17	0.92
Avg.	**0.23**	-0.66	-0.41	-1.33	-0.27	0.06	0.13

Visual comparison: Visual comparisons are given both for PSNR-tuned models and MS-SSIM-tuned models. In Fig. 4, we show the reconstruction error maps computed between the ground truth and the decompressed images when using OpenDVC, RLVC, HLVC, JPEG Pleno, NeRF-Slim-Quant and our PSNR-tuned models. Error maps are summed up over all viewpoints. One can observe that for a similar bit-rate, our model generates less error especially on object contours. In Fig. 5, we compare the visual quality of the decompressed images by using NeRF-Slim-Quant, JPEG Pleno, HEVC, and MS-SSIM-tuned OpenDVC and our models. It can be observed that the views generated by our model are less blurry than those by NeRF-Slim-Quant, HEVC and OpenDVC, and our model generates less artifact than JPEG Pleno.

Fig. 6 shows the reconstructed EPIs which are the slices in the sx- and yt-planes depicted below and on the right of the reconstructed center view. One can observe patterns with consistent slopes, which means the light field parallex is well preserved after compression.

Consistency across views: In Fig. 7, we visualize the variation of PSNR values for each sub-aperture view of the reconstructed light field. The corresponding bitrate is around 0.1 bpp for all the methods. The periodic variations observed with the methods HEVC, RLVC, HLVC and OpenDVC correspond to the GOP sizes used (4, 13, 10 and 10 respectively). For learning-based methods, i.e. RLVC, HLVC and OpenDVC, one can observe significant degradation of the PSNRs for inter coded views compared to intra coded ones. Similar observation can be made for JPEG Pleno: the quality of the views which are encoded at a higher hierarchical level with a low texture rate degrades rapidly compared to those at a lower coding level, for example, the central view. In fact, the error "propagates" with these methods, where the encoding and decoding of the current view depends on the previous reference views. On the contrary, our model consistently generates views across different viewpoints, since they are all supervised by the same loss function (image reconstruction error) during model learning, and hierarchical quality (I,P,B frames) used in conventional codecs is not imposed with our method. Apart from the views that are coded in intra mode using RLVC, HLVC and OpenDVC, and some views coded at a low hierarchical level using JPEG Pleno, our model gives a higher reconstruction quality than the other reference methods.

C. Ablation studies

1) Attention module: To put in evidence the utility of feature map modulation, in Fig. 8, we show the learning curves of our proposed network with and without attention modules respectively. After convergence, a gain of up to 1dB can be observed on the reconstructed light field images when using the feature map modulation. The parameter overhead is relatively limited, since the corresponding attention module only represents 2.5% of the total parameter load.

2) Compactness versus reconstruction quality using ConvGRU: We evaluate the efficiency of ConvGRU to learn the angular prior of light field views, and whether the use of ConvGRU yields a good trade-off between model compactness and reconstruction quality. Table III compares our model using ConvGRU against other approaches that can be considered to model the angular correlation between the light field views. We first compare with an adapted version of the deep decoder (Ada-DD) with $3v$ output channels, v being the number of sub-aperture views, while the original deep decoder [38] used for single 2D images with RGB channels has only 3 output channels. We also consider a solution in which the angular coordinates of the views are padded with the generator input, following the principle of CoordConv [62], to make the model aware of the view angular positions. Finally, we also compare

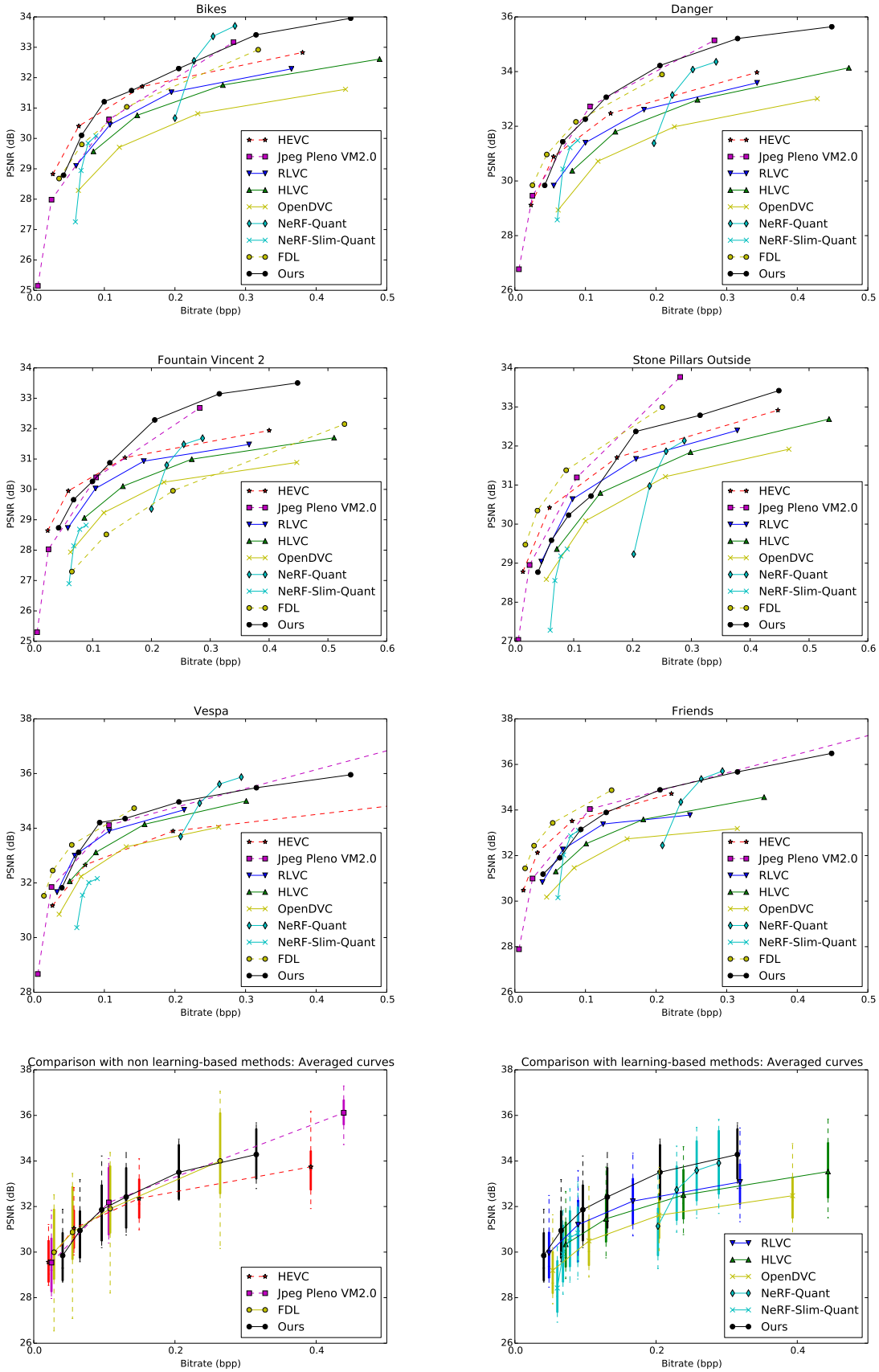

Fig. 2. PSNR vs. bitrate compression performance, with real world light fields from the JPEG Pleno dataset [75] (Bikes, Danger_de_Mort, Fountain_Vincent_2 and Stone_Pillars_Outside) and EPFL light field dataset [76] (Vespa and Friends). Averaged curves over all test data, with standard deviations, are also shown.

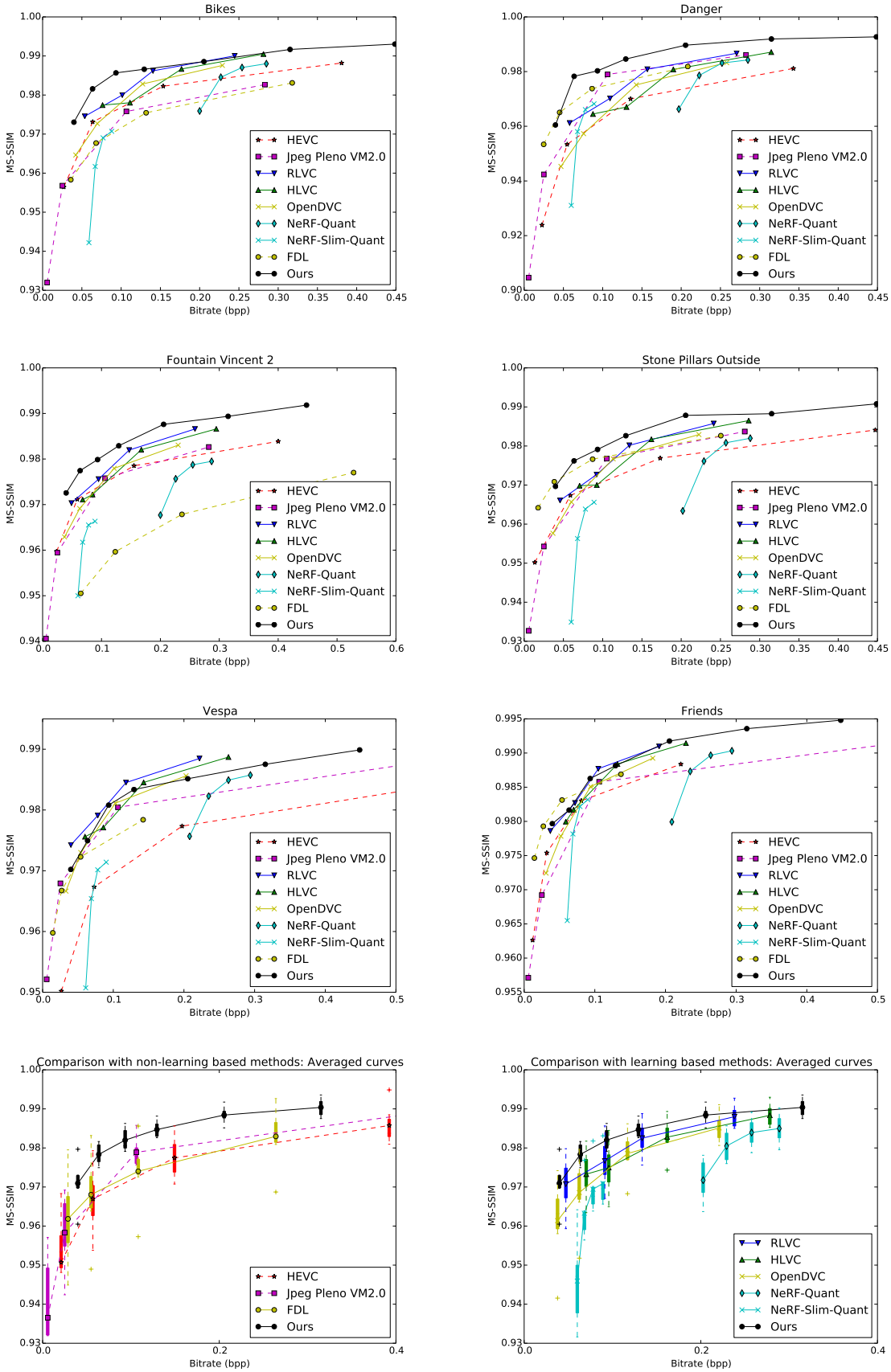

Fig. 3. Compression performance measured in terms of MS-SSIM vs. bitrate, using real world light fields from the JPEG Pleno dataset [75] (Bikes, Danger_de_Mort, Fountain_Vincent_2, Stone_Pillars_Outside and EPFL light field dataset [76] (Vespa and Friends). Averaged curves over all test data, with standard deviations, are also shown.

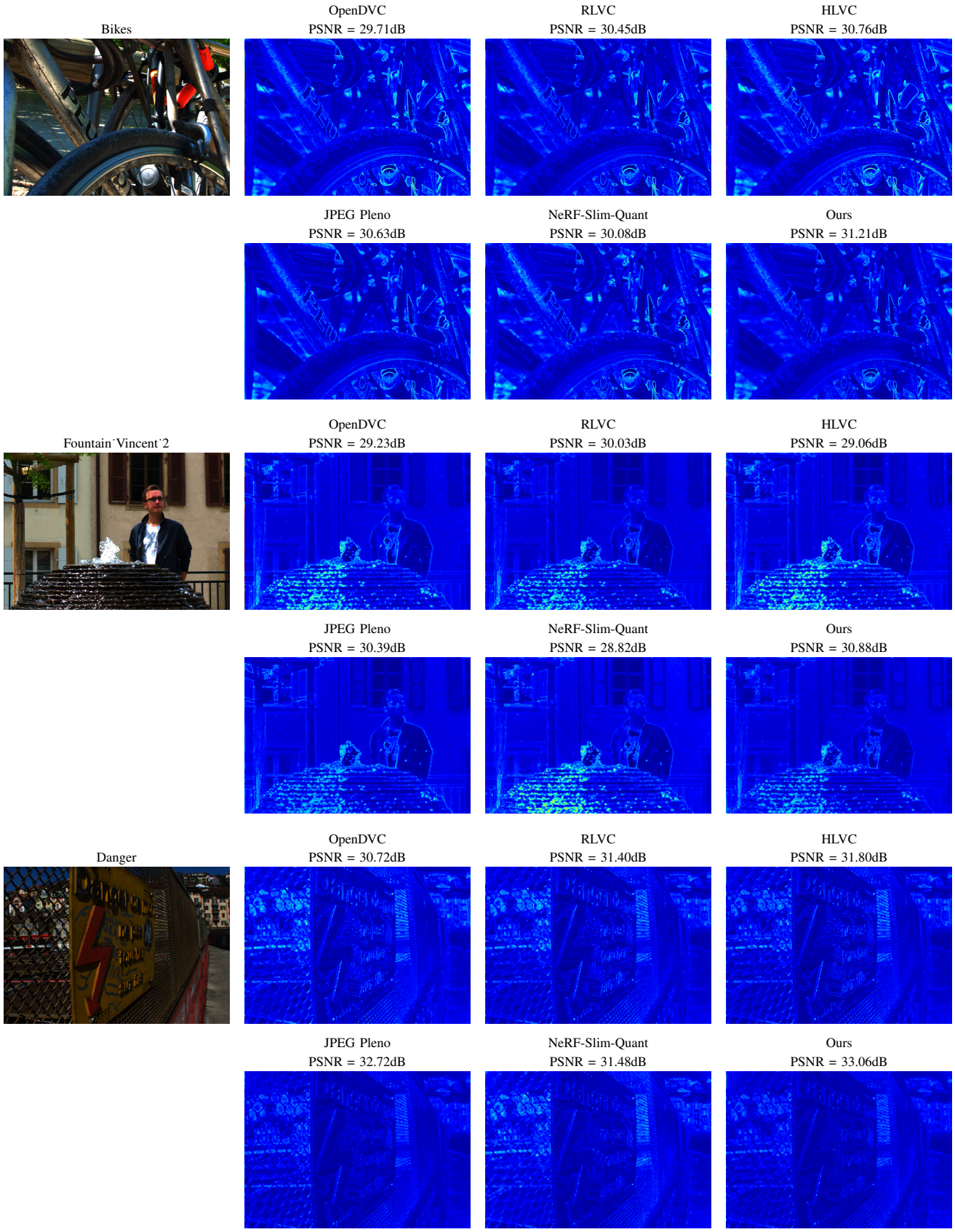

Fig. 4. The reconstruction error of the decompressed views obtained with PSNR-tuned models of OpenDVC, RLVC, HLVC and our method, as well as JPEG Pleno and NeRF-Slim-Quant. The bitrates are around 0.1 bpp.

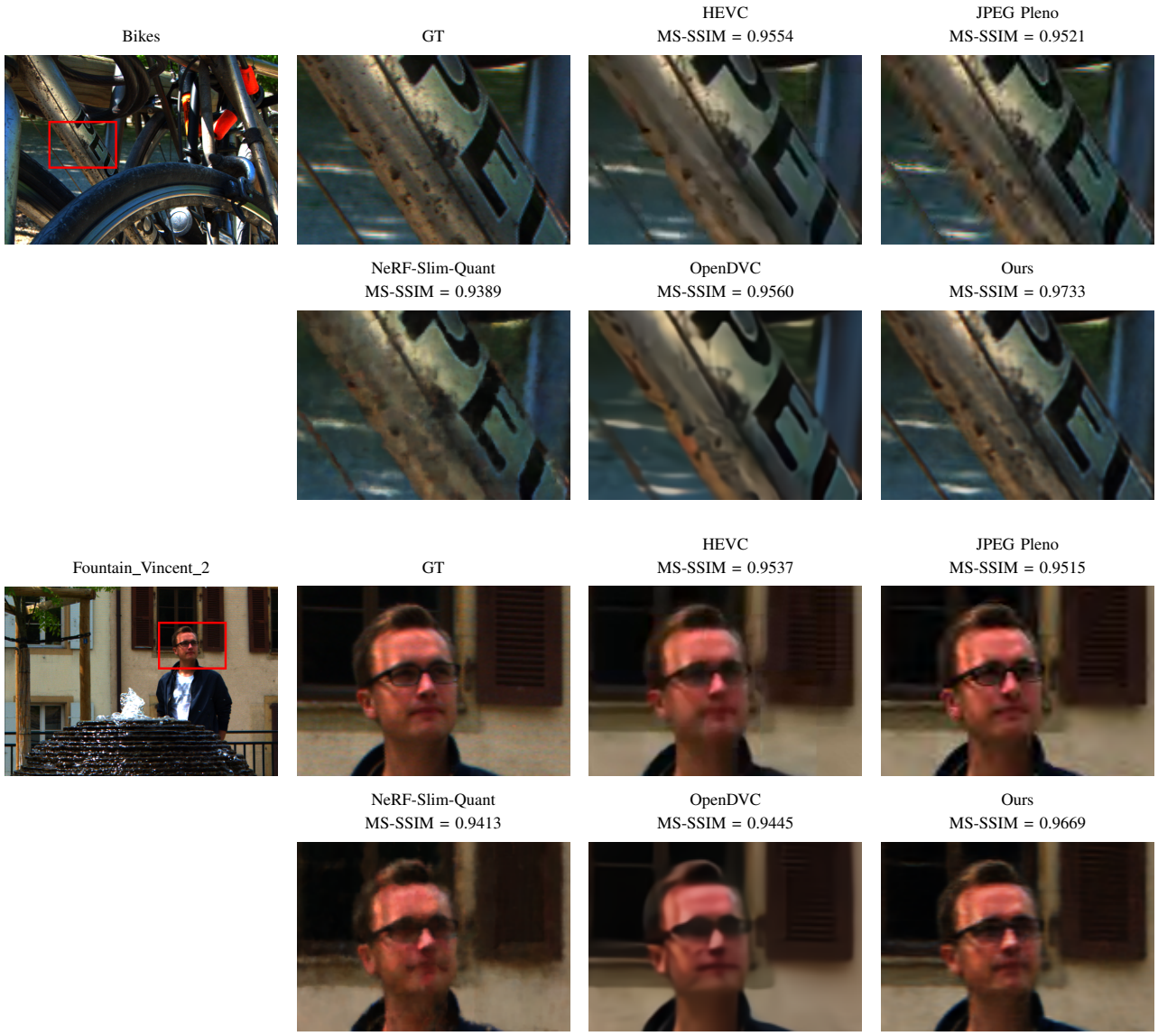

Fig. 5. Visual comparison of the decompressed view obtained with HEVC, JPEG Pleno, NeRF-Slim-Quant, MS-SSIM-tuned models of OpenDVC and our method. Comparable bitrates are around 0.03 bbp.

our model learning an angular prior based on the ConvGRU against NeRF models which map continous 5D vectors (3D coordinates plus 2D viewing directions) to volume density and view-dependent radiance. The original NeRF model of [32] has 595844 parameters. In order to be comparable with other models in terms of numbers of parameters, in Table III we consider a NeRF-Slim model with 172931 parameters, which has exactly the same architecture as a NeRF model, but with less parameters in each MLP layer. The CoordConv model is also dimensioned to have a comparable number of network parameters.

Table III shows that, in comparison with Ada-DD, CoordConv and NeRF-Slim, our model achieves the best reconstruction quality while using fewer parameters or a comparable number of parameters. Note that the consistent and high image quality across different viewpoints shown in Fig. 7 also demonstrates the effectiveness of ConvGRU module to learn a light field angular prior.

TABLE III
PSNR (DB), NUMBERS OF PARAMETERS FOR DIFFERENT MODELS GENERATING LIGHT FIELDS. VALUES ARE AVERAGED OVER THE TEST LIGHT FIELDS. †: THE COORDCONV PRINCIPLE IS USED TO MAKE OUR MODEL AWARE OF THE VIEW ANGULAR COORDINATES. ∗: THESE RESULTS HAVE BEEN OBTAINED WITH A NERF MODEL OF REDUCED DIMENSION (THAT WE CALLED NERF-SLIM) COMPARED WITH THE ORIGINAL MODEL OF [32].

	Our model	Ada-DD	CoordConv†	NeRF-Slim*
# params	173968	261300	178302	172931
# PSNR (dB)	**30.87**	29.98	30.19	29.99

3) Parameter-free upsampling vs. pixel-shuffle: In the decoder structure proposed in deep image generative models such as DIP [39] and deep decoder [38], handcrafted parameter-

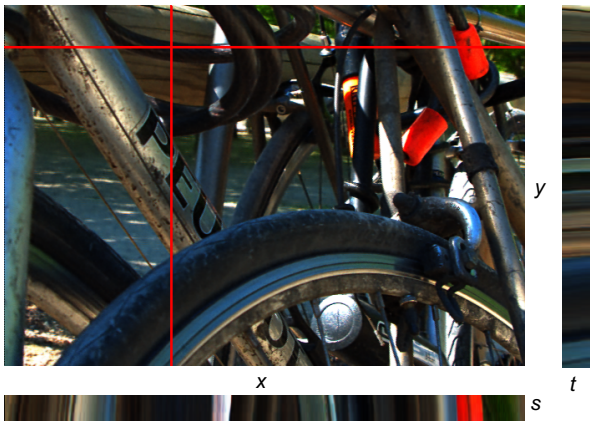

Fig. 6. Reconstructed epipolar images (EPIs) are the slices in the sx- and yt-planes shown below and on the right of the reconstructed center view. Test light field: Bikes.

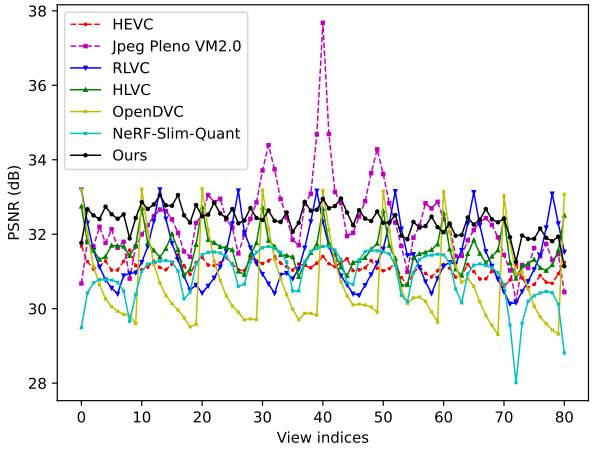

Fig. 7. The variation of average PSNR over all the test light fields, for each reconstructed sub-aperture view. The corresponding bitrate is around 0.1 bpp.

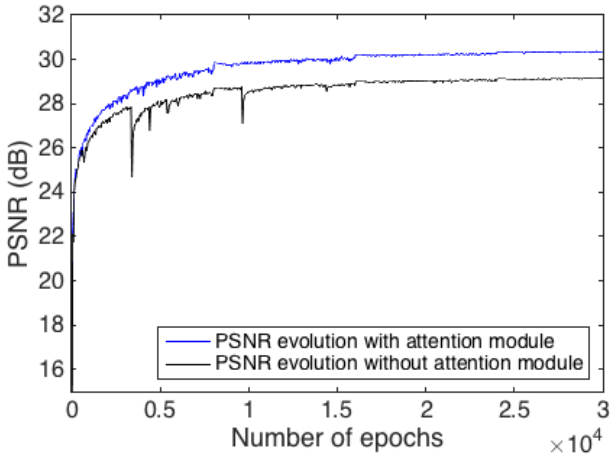

Fig. 8. Optimization curves of the proposed network with attention modules and without attention modules respectively for the light field reconstruction task.

TABLE IV
COMPARISON BETWEEN HANDCRAFTED PARAMETER-FREE UPSAMPLING AND LEARNED UPSAMPLING WITH PIXEL-SHUFFLE. THE PSNR VALUES OF THE GENERATED LIGHT FIELDS, AS WELL AS THE CORRESPONDING NETWORK PARAMETER NUMBERS ARE LISTED. THE PSNR VALUES ARE AVERAGED OVER THE TEST LIGHT FIELDS.

	(c_a,c_s)	(15,30)	(20,40)	(25,50)	(30,60)
Upsampling	#params	108270	173968	256346	355404
	PSNR (dB)	29.65	30.87	31.63	32.16
Pixel-shuffle	# params	326970	562768	863846	1230204
	PSNR(dB)	**31.52**	**33.24**	**34.11**	**34.79**

free upsampling operations are used to increase feature map resolutions from a lower layer to its immediate upper layer. Moreover, in [38] along with 1×1 convolutions, the relationships between nearby pixels of reconstructed images are barely imposed by upsampling layers. In Table IV, we compare the reconstruction performance in terms of PSNR between our model with handcrafted upsampling and its counterpart with learned upsampling by pixel-shuffle layers. The values are averaged over the test light fields. If taking the same configuration of the (c_a, c_s) pair, the number of features maps on each layer dedicated to angular information and shared spatial information, the model using pixel-shuffle significantly outperforms its counterpart using handcrafted upsampling by a large margin (a gain of approximately 2.5 dB can be observed). However, to obtain smaller models using pixel-shuffle, one has to decrease the number of feature maps per layer. For example, a model using pixel-shuffle with small number of feature maps $(c_a, c_s) = (15, 30)$ obtains lower PSNR than its counterpart using handcrafted upsampling with $(c_a, c_s) = (30, 60)$, whereas the two models have roughly the same number of parameters. In this work, we constantly search for the tradeoff between the model compactness and the generative capacity. Therefore, when a compact enough model is needed, it is more advantageous to maintain a moderate number of feature maps that contribute to image reconstruction, rather than using additional parameters to learn interpolation and decreasing feature map numbers. On the contrary, upsampling with learned pixel-shuffle layers allows a better modeling of the relationships between nearby pixels, both on the feature level and the image level, which yields more accurate reconstruction when the model compactness is less demanded.

4) Quantization: In Table V, we compare different quantization schemes in terms of light field reconstruction performance. Four networks with parameters $(c_a, c_s) = (15, 30), (20, 40), (25, 50), (30, 60)$ are tested. We take pre-trained light field generative models, which corresponds to the line "Without quantization" in Table V, and then apply different quantization schemes. A posteriori quantization without any finetuning yields about 3-4dB of PSNR loss compared to the non-compressed model. Schemes using quantization-aware finetuning (QAF) update quantized weights with respect to the light field reconstruction quality. We observe that the scheme "Non-uniform QAF" (c.f. Section III) obtains the best PSNR values among all tested quantization schemes, and achieves

TABLE V
RECONSTRUCTION QUALITY MEASURED IN PSNR(DB) BY USING DIFFERENT QUANTIZATION SCHEMES ON THE TEST LIGHT FIELD "DANGER_DE_MORTS"

(c_a,c_s)	(15,30)	(20,40)	(25,50)	(30,60)
A posteriori quantization	27.57	28.65	29.12	30.17
Uniform QAF	27.68	29.16	31.35	31.80
Non-uniform QAF	**29.84**	**31.44**	**32.26**	**33.06**
QAT from scratch	19.17	20.69	20.74	21.16
Without quantization	30.57	32.29	33.05	33.71

TABLE VI
MODEL MEMORY USAGE (IN MEGABYTES) COMPARISON BETWEEN QUANTIZED AND NON-QUANTIZED NETWORKS. THE TEST LIGHT FIELD IS "DANGER_DE_MORTS" WITH AN ORIGINAL SIZE OF 40.4 MEGABYTES.

(c_a,c_s)	(15,30)	(20,40)	(25,50)	(30,60)
Non-uniform QAF	0.11	0.18	0.27	0.35
Without quantization	0.43	0.70	1.03	1.42

TABLE VII
DECODING TIME OF TESTED METHODS

HEVC	JPEG Pleno	RLVC	HLVC	OpenDVC
13.4s	18.8s	2587.0s	491.2s	43.7s
NeRF	NeRF-Slim	CoordConv	Ours	
580.8s	353.9s	**0.250s**	0.262s	

significant gain compared against the "Uniform QAF" using equal widths quantization bins. Finally, "QAT from scratch" corresponds to the scheme applying quantization-aware training from randomly-initialized weights. The results demonstrate the necessity of optimizing network weights with respect to the reconstruction quality before quantization. In Table VI, memory usage of the models applying non-uniform QAF and those without quantification is shown.

5) Decoding time: Table VII gives the decoding times of the different methods, for a light field of 81 views of resolution 432×624. We used a GeForce RTX 2080 Ti GPU for testing learning-based methods. Our method, using the convGRU or the CoordConv technique to model angular variations, gives the shortest decoding time. Both methods use the same generator structure. When using CoordConv, instead of using ConvGRU to exploit angular correlation, view angular positions are padded with the generator input, following the principle of [62]. When using HEVC, JPEG Pleno, the RLVC, HLVC and OpenDVC methods, the light field views are decoded following the order used for encoding, the decoding of the current view depending on the decoding of a reference views. NeRF and NeRF-Slim proceeding view by view could be parallelized in theory, however at the cost of a very high memory usage, which, for some GPUs, not always be practical. The CoordConv solution decodes all the light field views at the same time, without any dependency to any of the other views. Our model can be seen as a hybrid solution, in which a short sequence carrying inter-block angular dependency information is predicted by the ConvGRU, and all the views are then generated by the generator at the same time. The simplicity of our network structure and the relative lack of reference dependency can explain the short decoding time compared against other methods. Finally, our solution with ConvGRU is only slightly slower than the use of CoordConv in terms of decoding time, which shows that the ConvGRU does not penalize much the method in terms of decoding efficiency.

Note that for each light field, its corresponding network is quantized and transmitted to the decoder. On the decoder side, it takes the input noise, being fixed for all the light fields and being known on the decoder side, and the quantized network. The decoder performs a simple feedforward inference to obtain the decoded light field. The change of the network configuration, such as number of blocks and number of views per block, does not impact the efficiency of this inference process on the decoder side.

V. CONCLUSION

In this paper, we proposed a deep generative model for light fields that does not require any training data other than the light field itself. We show that the proposed model gives a compact representation of the input light field, and can lead, with quantization-aware learning, to convincing compression performance with high image quality. In future work, we will investigate the use of this untrained compact model for solving inverse light field problems. We will also further explore the use of the fast evolving NeRF concept in the context of light field compression, considering methods such as in [35], [36], [37], [77] for learning the NeRF models.

VI. ACKNOWLEDGEMENT

The authors would like to thank Elian Dib for having provided the experimental results for Jpeg Pleno compression scheme.

REFERENCES

[1] H. E. Ives, "Parallax panoramagrams made with a large diameter lens," *Journal of the Optical Society of America*, vol. 20, no. 6, pp. 332–342, June 1930.
[2] G. Lippmann, "La photographie intégrale," *Comptes-Rendus, Académie des Sciences*, vol. 146, pp. 446–551, 1908.
[3] E. H. Adelson and J. Y. A. Wang, "Single Lens Stereo with a Plenoptic Camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 99–106, Feb. 1992.
[4] R. Ng, M. Levoy, M. Bredif, G. Duval, M. Horowitz, and P. Hanrahan, "Light Field Photography with a Handheld Plenoptic Camera," Computer Science Technical Report CSTR 2(11), Stanford University, 2005.
[5] J. C. Yang, M. Everett, C. Buehler, and L. McMillan, "A Real-Time Distributed Light Field Camera," in *Eurographics Workshop on Rendering (EGSR)*, 2002, pp. 77–86.
[6] B. Wilburn, N. Joshi, V. Vaish, E.-V. Talvala, E. Antunez, a. Barth, A. Adams, M. Horowitz, and M. Levoy, "High Performance Imaging using Large Camera Arrays," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 765–776, July 2005.
[7] S. D. Babacan, R. Ansorge, M. Luessi, P. R. Mataran, R. Molina, and A. K. Katsaggelos, "Compressive Light Field Sensing," *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4746–4757, Dec. 2012.
[8] K. Marwah, G. Wetzstein, Y. Bando, and R. Raskar, "Compressive Light Field Photography using Overcomplete Dictionaries and Optimized Projections," *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, vol. 32, no. 4, pp. 46:1–46:12, 2013.

[9] E. Miandji, J. Unger, and C. Guillemot, "Multi-shot single sensor light field camera using a color coded mask," in *European Signal Processing Conference (EUSIPCO)*, June 2018, pp. 226–230.

[10] O. Nabati, D. Mendlovic, and R. Giryes, "Fast and accurate reconstruction of compressed color light field," *2018 IEEE International Conference on Computational Photography (ICCP)*, pp. 1–11, 2018.

[11] G. Le Guludec, E. Miandji, and C. Guillemot, "Deep light field acquisition using learned coded mask distributions for color filter array sensors," *IEEE Transactions on Computational Imaging*, vol. 7, pp. 475–488, 2021.

[12] C. Conti, P. Nunes, and L. D. Soares, "HEVC-based light field image coding with bi-predicted self-similarity compensation," in *IEEE Int. Conf. Multimed. Expo Workshops (ICMEW)*, Jul. 2016.

[13] 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 *IEEE Int. Conf. Multimed. Expo Workshops (ICMEW)*. IEEE, 2016, pp. 1–4.

[14] D. Liu, L. Wang, L. Li, Z. Xiong, F. Wu, and W. Zeng, "Pseudo-sequence-based light field image compression," in *IEEE Int. Conf. Multimed. Expo Workshops (ICMEW)*. IEEE, 2016, pp. 1–4.

[15] W. Ahmad, R. Olsson, and M. Sjostrom, "Interpreting plenoptic images as multiview sequences for improved compression," in *IEEE Int. Conf. Image Process. (ICIP)*, 2017.

[16] I. Tabus, P. Helin, and P. Astola, "Lossy compression of lenslet images from plenoptic cameras combining sparse predictive coding and jpeg 2000," in *IEEE Int. Conf. Image Process. (ICIP)*. IEEE, 2017, pp. 4567–4571.

[17] C. L. Chang and X. Zhu, "Light field compression using disparity-compensated wavelet decomposition," *IEEE Trans. on Image Proc.*, vol. 15, 2002.

[18] C.-L. Chang, X. Zhu, P. Ramanathan, and B. Girod, "Light field compression using disparity-compensated lifting and shape adaptation," *IEEE Trans. on Image Proc.*, vol. 14, no. 4, pp. 793–806, 2006.

[19] B. Girod, C.-L. Chang, P. Ramanathan, and X. Zhu, "Light field compression using disparity-compensated lifting," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03).*, 2003, vol. 4, pp. IV–760.

[20] D. Barina, M. Solony, T. Chlubna, D. Dlabaja, O. Klima, and P. Zemcik, "Comparison of light field compression methods," *Multimedia Tools Appl.*, vol. 81, no. 2, pp. 2517–2528, jan 2022.

[21] X. Jiang, M. L. Pendu, and C. Guillemot, "Light fields compression using depth image based view synthesis," in *Hot3D workshop held jointly with IEEE Int. Conf. Multimed. Expo, ICME*, Jul. 2017.

[22] S. Zhao and Z. Chen, "Light field image coding via linear approximation prior," in *IEEE Int. Conf. on Image Processing, ICIP*, 2017.

[23] E. Dib, M. L. Pendu, and C. Guillemot, "Light field compression using fourier disparity layers," *IEEE International Conf. on Image Processing (ICIP)*, pp. 3751–3755, 2019.

[24] F. Hawary, C. Guillemot, D. Thoreau, and G. Boisson, "Scalable light field compression scheme using sparse reconstruction and restoration," *IEEE International Conf. on Image Processing (ICIP)*, pp. 3250–3254, 2017.

[25] M. L. Pendu, C. Guillemot, and A. Smolic, "A fourier disparity layer representation for light fields," *IEEE Trans. on Image Processing*, vol. 28, pp. 5740–5753, 2019.

[26] L. Shi, H. Hassanieh, A. Davis, D. Katabi, and F. Durand, "Light field reconstruction using sparsity in the continuous fourier domain," *ACM Trans. on Graphics (TOG)*, vol. 34, pp. 1 – 13, 2014.

[27] M. Volino, A. Mustafa, J.-Y. Guillemaut, and A. Hilton, "Light field compression using eigen textures," in *International Conf. on 3D Vision (3DV)*, 2019.

[28] C. Conti, P. Nunes, and L. D. Soares, "Inter-Layer Prediction Scheme for Scalable 3-D Holoscopic Video Coding," *IEEE Signal Processing Letters*, vol. 20, no. 8, pp. 819–822, 2013.

[29] Y. Li, M. Sjöström, R. Olsson, and U. Jennehag, "Scalable Coding of Plenoptic Images by Using a Sparse Set and Disparities," *IEEE Transactions on Image Processing*, vol. 25, pp. 80–91, 2016.

[30] D. Rüefenacht, A. T. Naman, R. Mathew, and D. Taubman, "Base-Anchored Model for Highly Scalable and Accessible Compression of Multiview Imagery," *IEEE Transactions on Image Processing*, vol. 28, no. 7, pp. 3205–3218, 2019.

[31] X. Hu, J. Shan, Y. Liu, and L. Zhang, "Adaptive two-layer light field compression scheme based on sparse reconstruction," *Proceedings of the 10th ACM Multimedia Systems Conference*, 2019.

[32] B. Mildenhall, P. Srinivasan, M. Tancik, J. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Eur. Conf. on Computer Vision (ECCV)*, 2020, pp. 405–421.

[33] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "pixelneRF: Neural radiance fields from one or few images," *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 4576–4585, 2021.

[34] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su, "MVSNeRF: Fast generalizable radiance field reconstruction from multi-view stereo," in *IEEE Int. Conf. on Computer Vision (ICCV)*, 2021.

[35] B. Y. Feng and A. Varshney, "Signet: Efficient neural representations for light fields," in *Proceedings of the International Conference on Computer Vision (ICCV 2021)*, 2021.

[36] M. Bemana, K. Myszkowski, H.-P. Seidel, and T. Ritschel, "X-fields: Implicit neural view-, light- and time-image interpolation," *ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2020)*, vol. 39, no. 6, 2020.

[37] Z. Li, L. Song, C. Liu, J. Yuan, and Y. Xu, "Neulf: Efficient novel view synthesis with neural 4d light field," in *arXiv:2105.07112v6*, Dec. 2021.

[38] R. Heckel and P. Hand, "Deep decoder: Concise image representations from untrained non-convolutional networks," *CoRR*, vol. abs/1810.03982, 2018.

[39] V. Lempitsky, A. Vedaldi, and D. Ulyanov, "Deep image prior," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9446–9454.

[40] G. Toderici, D. Vincent, N. Johnston, S. J. Hwang, D. Minnen, J. Shor, and M. Covell, "Full resolution image compression with recurrent neural networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5435–5443.

[41] K. Gregor, F. Besse, D. Jimenez Rezende, I. Danihelka, and D. Wierstra, "Towards conceptual compression," in *Advances in Neural Information Processing Systems*, 2016, vol. 29.

[42] T. Dumas, A. Roumy, and C. Guillemot, "Image compression with stochastic winner-take-all auto-encoder," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 1512–1516.

[43] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," in *International Conference on Learning Representations (ICLR)*, 2017.

[44] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *International Conference on Learning Representations (ICLR)*, 2017.

[45] N. Johnston, E. Eban, A. Gordon, and J. Ballé, "Computationally efficient neural image compression," 2019.

[46] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *International Conference on Learning Representations (ICLR)*, 2018.

[47] D. Minnen, G. Toderici, S. Singh, S. J. Hwang, and M. Covell, "Image-dependent local entropy models for learned image compression," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 430–434.

[48] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 333–348.

[49] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee, "Decomposing motion and content for natural video sequence prediction," *International Conference on Learning Representations (ICLR)*, 2017.

[50] A. Aich, A. Gupta, R. Panda, R. Hyder, M. S. Asif, and A. K. Roy-Chowdhury, "Non-adversarial video synthesis with learned priors," in *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020, pp. 6090–6099.

[51] "Jpeg pleno, url = https://jpeg.org/jpegpleno/,," .

[52] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[53] ISO/IEC JTC 1/SC29, "High Efficiency Coding and Media Delivery in Heterogeneous Environments – Part 2: High Efficiency Video Coding. ISO/IEC 23008-2:2017," Tech. Rep., 2017.

[54] R. Yang, F. Mentzer, L. Van Gool, and R. Timofte, "Learning for video compression with hierarchical quality and recurrent enhancement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[55] R. Yang, F. Mentzer, L. Van Gool, and R. Timofte, "Learning for video compression with recurrent auto-encoder and recurrent probability model," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 2, pp. 388–401, 2021.

[56] R. Yang, L. Van Gool, and R. Timofte, "OpenDVC: An open source implementation of the DVC video compression method," *CoRR*, vol. abs/2006.15862, 2020.
[57] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, and Z. Gao, "Dvc: An end-to-end deep video compression framework," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 10998–11007.
[58] J. Lin, D. Liu, H. Li, and F. Wu, "M-LVC: Multiple frames prediction for learned video compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 3546–3554.
[59] Z. Hu, Z. Chen, D. Xu, G. Lu, W. Ouyang, and S. Gu, "Improving deep video compression by resolution-adaptive flow coding," in *European Conference on Computer Vision (ECCV)*, 2020.
[60] N. Zou, H. Zhang, F. Cricri, H. R. Tavakoli, J. Lainema, E. Aksu, M. Hannuksela, and E. Rahtu, "End-to-end learning for video frame compression with self-attention," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020.
[61] M. A. Yilmaz and A. M. Tekalp, "End-to-end rate-distortion optimization for bi-directional learned video compression," in *IEEE International Conference on Image Processing (ICIP)*, 2020.
[62] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski, "An intriguing failing of convolutional neural networks and the coordconv solution," in *NeurIPS*, 2018.
[63] M. Levoy and P. Hanrahan, "Light field rendering," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Techn.(CCGIT)*, 1996, pp. 31–42.
[64] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," in *23rd Annual Conf. on Computer Graphics and Interactive Techniques*. 1996, pp. 43–54, ACM.
[65] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.
[66] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua, "SCA-CNN: Spatial and channel-wise attention in convolutional networks for image captioning," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[67] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 286–301.
[68] R. Hou, H. Chang, B. Ma, S. Shan, and X. Chen, "Cross attention network for few-shot classification," in *Advances in Neural Information Processing Systems*, 2019, vol. 32.
[69] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
[70] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
[71] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," *arXiv: Computer Vision and Pattern Recognition*, 2016.
[72] A. Fan, P. Stock, B. Graham, E. Grave, R. Gribonval, H. Jegou, and A. Joulin, "Training with quantization noise for extreme model compression," in *International Conference on Learning Representations (ICLR)*, 2021.
[73] F. Bellard, "BPG image format," https://bellard.org/bpg/.
[74] J. Lee, S. Cho, and S.-K. Beack, "Context-adaptive entropy model for end-to-end optimized image compression," in *International Conference on Learning Representations (ICLR)*, 2019.
[75] M. Rerabek and T. Ebrahimi, "New light field image dataset," in *8th International Conference on Quality of Multimedia Experience (QoMEX)*, 2016.
[76] M. Rerabek and T. Ebrahimi, "New light field image dataset," in *Int. Conf. on Quality of Multimedia Experience (QoMEX)*, 2016, number EPFL-CONF-218363.
[77] B. Attal, J.-B. Huang, M. Zollhöfer, J. Kopf, and C. Kim, "Learning neural light fields with ray-space embedding networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Xiaoran Jiang received the Engineering degree in telecommunications and the Ph.D. degree in computer science from IMT Atlantique (Télécom Bretagne), France, in 2010 and 2014, respectively. From 2016 to 2021, he worked as a researcher with INRIA (Institut National de Recherche en Informatique et en Automatique), Rennes, France. He is currently an associate professor with INSA (Institut National des Sciences Appliquées) Rennes, and IETR (Institut d'Électronique et des Technologies du numéRique), Rennes, France. His current research interests include signal and image processing, computer vision, computational photography and deep learning.

Jinglei Shi is currently a lecturer in Nankai University, Tianjin, China. He received Bachelor's degree in Electronic Information Engineering from UESTC (University of Electronic Science and Technology of China) in 2015. Then he received Engineer's degree and Master's degree in Image Processing from IMT (Institut Mines-Telecom) Atlantique, France, in 2017, and PhD degree from University Rennes 1, France, in 2021. Then he worked as a post-doctoral researcher at INRIA (Institut National de Recherche en Informatique et en Automatique) in France. His research interests concern learning-based light field depth estimation, view synthesis, video interpolation and compression.

Christine Guillemot IEEE Fellow, is Director of Research at INRIA. She holds a Ph.D. degree from ENST (Ecole Nationale Superieure des Telecommunications) Paris, and a Habilitation for Research Direction from the University of Rennes. From 1985 to Oct. 1997, she has been with FRANCE TELECOM, where she has been involved in various projects in the areas of image and video coding and processing for TV, HDTV and multimedia. From Jan. 1990 to mid 1991, she has worked at Bellcore, NJ, USA, as a visiting scientist. Her research interests include: signal and image processing, and computer vision. She has served as Associate Editor for IEEE Trans. on Image Processing (from 2000 to 2003, as well as 2014-2016), for IEEE Trans. on Circuits and Systems for Video Technology (from 2004 to 2006), and for IEEE Trans. on Signal Processing (2007-2009). She has served as senior member of the editorial board of the IEEE journal on selected topics in signal processing (2013-2015) and has been senior area editor of IEEE Trans. on Image Processing (2016-2020).