Learning Fused Pixel and Feature-based View Reconstructions for Light Fields

Jinglei Shi* Xiaoran Jiang* Christine Guillemot
INRIA Rennes - Bretagne Atlantique, France
{firstname.lastname}@inria.fr

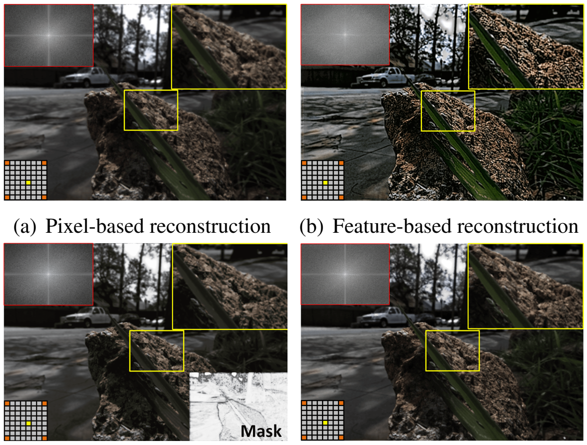
Abstract

In this paper, we present a learning-based framework for light field view synthesis from a subset of input views. Building upon a light-weight optical flow estimation network to obtain depth maps, our method employs two reconstruction modules in pixel and feature domains respectively. For the pixel-wise reconstruction, occlusions are explicitly handled by a disparity-dependent interpolation filter, whereas inpainting on disoccluded areas is learned by convolutional layers. Due to disparity inconsistencies, the pixel-based reconstruction may lead to blurriness in highly textured areas as well as on object contours. On the contrary, the feature-based reconstruction well performs on high frequencies, making the reconstruction in the two domains complementary. End-to-end learning is finally performed including a fusion module merging pixel and feature-based reconstructions. Experimental results show that our method achieves state-of-the-art performance on both synthetic and real-world datasets, moreover, it is even able to extend light fields' baseline by extrapolating high quality views without additional training.

1. Introduction

Light field imaging has recently attracted a lot of attention due to the emergence of commercial cameras and the numerous applications, going from computational photography to realistic rendering in augmented and virtual reality applications, and light field microscopy. Acquisition devices have been designed either based on camera arrays [1], on moving gantries[2], or on micro-lens arrays used in plenoptic cameras. Single hand-held 2D camera (*e.g*. cell phones) paired with pose estimation techniques [3] can enable the capture of light fields with high spatial resolution but limited angular resolution (or large baselines).

The problems of enhancing the light field angular resolution can be tackled from different perspectives, *i.e*. as a problem of light field reconstruction from a subset of views using signal priors (*e.g*. sparsity in the continuous 4D Fourier domain [4]), of angular super-resolution [5, 6] or of view synthesis. While Image Based Rendering (IBR) techniques have been predominant over the past years in the field of view synthesis (see *e.g*. [7, 8]), this field has significantly evolved thanks to the emergence of learning-based approaches.



(a) Pixel-based reconstruction (b) Feature-based reconstruction

(c) Fused view (d) Reference view

Figure 1. Visualization of the outputs (both in pixel domain and frequency domain) at different stages of our framework.

Kalantari *et al*. [9] have been among the first proposing a learning-based solution for view synthesis, by sequentially connecting two convolutional neural networks (CNNs) dedicated respectively to depth estimation and color fusion. However, due to imprecision of depth estimation, the method tends to synthesize views with blurriness, tearing and ghosting effects. In addition, it fails in occluded regions for sparse light fields (with large baselines). Inspired by the Multi-Plane Image (MPI) representation[10], Mildenhall *et al*. [3] construct a framework that renders novel views from irregularly sampled views. They apply a 3D CNN to learn a MPI representation for each input view from plane sweep volumes (PSVs). The MPIs are then warped and merged to synthesize the target view. The approach generates blurry results in the case of depth uncertainty, and the use of PSV is computationally expensive.

[*] equal contribution

Wu *et al*. [11] instead reconstruct light fields by fusing a set of sheared epipolar plane images (EPIs) scored by a CNN. Boundary artifacts appear when the target views are far from the source views.

In this paper, we propose a novel learning-based framework to synthesize light field views from a sparse set of input views. We design an end-to-end learning framework combining two reconstruction strategies, one in the pixel domain and the other in the feature space. A CNN is first used to estimate disparity from the input views. Using the estimated disparity, we project the input color views and their features to the target view position. The features are extracted using the lower layers of the VGG19 classification network[12]. For the pixel-wise reconstruction, occlusions are explicitly handled by a disparity-dependent interpolation filter, and the target view is predicted based on warped views using convolutional layers. For the feature-based reconstruction, multi-scale features at the target viewpoint are successively reconstructed based on warped features from the input views, and the reconstructed view is inferred from the feature maps of the finest scale. Finally, a soft mask is learned to merge the results of the pixel-wise and feature-based reconstructions. The entire framework is trained in an end-to-end fashion. Fig. 1 shows the reconstruction results at different stages of the network.

Experimental results with both synthetic and real-world light fields, with a large range of disparities between input views, demonstrate that our proposed framework significantly outperforms the state-of-the-art methods. Our approach gives excellent reconstruction quality on fine textures and object boundaries, despite the fact that intermediate disparity outputs may not be accurate and consistent across input views. Furthermore, we show that our network can also achieve competitive performances for light field view extrapolation without additional training.

2. Related work

View synthesis has been a very active field of research for many years. The methods have evolved from techniques making an explicit use of geometry, such as depth image based rendering (DIBR) techniques [13, 14], towards solutions based on plane sweep volumes (PSV) and not requiring explicit depth information [10]. End-to-end learning methods have also been considered using depth learned in an unsupervised manner and specifically for the view synthesis task [9]. More recently, deep neural networks have been proposed for learning Multi-Plane Images (MPI) representations, first for stereo views [10]. The MPIs can be interpolated for generating novel views by exploiting notions of visibility or transparency with alpha blending maps [10, 15]. This method has been extended to unstructured light fields in [3] leading to state-of-the-art view synthesis results. In parallel, light field reconstruction methods exploiting signal priors, *e.g*. sparsity priors in the 4D Fourier domain [4, 16], sparsity in the shearlet transform domain [6], smoothness on Epipolar Plane Images (EPI) [5] have also been proposed for view synthesis. In this section, we focus on methods that are recent and the most closely related to the proposed one, *i.e*. learning-based solutions as well as those using the concepts of PSV and MPI that are used as benchmarks in the experimental section.

2.1. Depth image based rendering w/wo learning

Traditional image based rendering techniques proceed in two steps. They first estimate the geometry (the depth) and then warp the source views into the target positions. This is the case for example of [14]. However, the quality of the results very much depends on the accuracy of the depth maps, and estimating accurate depth maps remains a challenging problem especially in presence of transparency, or gloss.

Layered representations decomposing the reflective parts of a scene into a transmitted and a reflected layer have been proposed in [17] to cope with the above difficulty. The layers rendered with their own geometry are then blended using some opaque mixing. This idea has been further developed with the concept of PSV constructed by warping a given image into a target viewpoint using different depth levels. The PSV, which can be seen as sampling the scene in the depth direction, leading to depth planes, is now often used as the input of view synthesis algorithms. This is the case in [8], where the source images are blended per sampled depth with weights based on consensus and visibility scores computed for each pixel and depth plane.

Kalantari *et al*. [9] adapt the conventional DIBR approach into an end-to-end learning framework. The authors propose an architecture based on two CNNs, the first estimates depth in each target viewpoint from the input views, while the second predicts the color. The second CNN, thanks to end-to-end learning, can correct warping errors resulting from depth inaccuracies. They train the network by minimizing the error between the synthesized and the ground truth views. In the same vein, Srinivasan *et al*. [18] propose to synthesize a light field from one single view, using a 2-stage learning process, estimating geometry first, and then estimating occluded rays. The two methods above are however limited to light fields with small baselines.

2.2. View synthesis with learned EPI interpolation

While the above methods are applied on light field views, there also exist methods operating on EPIs, in particular for angular interpolation or super-resolution [4, 5, 6, 11, 16, 19, 20]. Focusing on learning-based solutions, Wu *et al*. [19] model the light field reconstruction as a learning-based detail restoration in the EPIs. They first apply a bi-cubic angular interpolation on input EPIs from which spatial high frequencies have been removed and use a CNN to restore details in the angular domain of the interpolated EPIs. The spatial details are then recovered by a non-blind deblur operation. This "blur-restoration-deblur" framework does not require depth estimation. Wang *et al.* [20] instead apply 3D convolutions on EPI-volumes (stacked EPIs along rows or columns of the light field) to restore high-frequency details, which allows better using the correlation within the light field data. Wu *et al.* [11] train a CNN to evaluate sheared EPIs, and output a reference score which is then used for fusing the sheared EPIs. However, the methods relying on EPI structures work well only if the baseline is small.

2.3. View synthesis with learned representations

Using PSVs constructed from warped input views, Flynn *et al.* [15] train two parallel CNNs, one for predicting the color in each depth plane and the second one to predict the probability that a pixel belongs to a particular depth plane. The novel view is synthesized by element-wise multiplication of the outputs of both CNNs and then by summing up over the depth planes. Zhou *et al.* [10] train a deep network to predict a MPI representation from a narrow-baseline stereo image pair. Mildenhall *et al.* [3] extend this idea to larger-baseline view interpolation from unstructured light fields. As MPIs contain 3D information of the scene, it can be used for view extrapolation as well. This learned MPI-based solution gives state-of-the-art results in view synthesis, in particular in the difficult case of large baselines and unstructured light fields. The MPI representation has similarity with LDI representation in [21], where the authors also propose a differentiable interpolation technique based on disparity values. Choi *et al.* [22] propose a view extrapolation method with large baselines using learned depth probability volumes together with an image refinement network. Meng *et al.* [23] develop a learning framework based on a two-stage restoration with a 4-dimensional convolutional rsidual network for light field spatio-angular super-resolution. Yeung *et al.* [24] follow a two-step approach based on view synthesis network that first generates the whole set of novel views, and a view refinement network that retrieves spatial texture details.

While, in the same vein as [9], the proposed method includes learning depth information for view synthesis, it significantly differs from [9] first by exploiting information of warped features in addition to warped views, and second by the methods used for disparity estimation and warping, which limit the method in [9] to light fields with small baselines. Unlike methods extracting structures from EPI for view interpolation, the proposed method exploits features extracted from views using the VGG19-Net [12]. We show that the warped features bring complementary information to warped views to better deal with fine textures and with occluded regions.

3. Methodology

3.1. Overview

Let us denote a light field by a 4-dimensional function $L(x, y, u, v)$, where $(x, y) \in [\![1; X]\!] \times [\![1; Y]\!]$ and $(u, v) \in [\![1; U]\!] \times [\![1; V]\!]$ are respectively spatial and angular coordinates. The sub-aperture image $L(x, y, u_i, v_i)$ at the angular position $\mathbf{i} = (u_i, v_i)$ is referred to as $L_{\mathbf{i}}$.

We aim at reconstructing the novel view at the target position $\mathbf{t}$ from a set of input views $\mathcal{I} = \{L_{\mathbf{i_1}}, ..., L_{\mathbf{i_N}}\}$. In this work, the set $\mathcal{I}$ contains sparsely sampled 2×2 views. For convenience, these views are also denoted by L_{tl} (top left), L_{tr} (top right), L_{bl} (bottom left) and L_{br} (bottom right).

Fig. 2 shows our learning-based framework. First, a lightweight disparity estimator module (in blue) estimates one disparity map for each input view. Two parallel reconstruction schemes are then applied. The target view is synthesized either by PixRNet (Pixel-wise Reconstruction Network) or FeatRNet (Feature-based Reconstruction Network). For the pixel-wise scheme, given the disparity estimates, the input views are first projected to the target position by applying forward warping. PixRNet takes as input the projected views as well as their occlusion masks to synthesize the novel view. PixRNet provides accurate pixel values in lowly textured areas. However, due to inconsistencies between disparity values estimated for the different input views, a simple fusion in the pixel space of the projected views may lead to blurriness in highly textured areas as well as on object contours. FeatRNet is thus designed to compensate for this drawback. In FeatRNet, the reconstruction is based on low-level features inferred for the novel viewpoint. Extracted from lower layers of VGG19-Net [12], the features of the input views are warped to the target position at different resolution scales to generate the corresponding target view features, from which the decoder reconstructs the color view. Finally, a learned soft combination mask merges the outputs of both PixRNet and FeatRNet.

3.2. Disparity estimation

Commonly, disparity refers to the distance between two corresponding points in the left and right view of a stereo pair. Assuming the light field views are well rectified and regularly spaced, it is convenient to use "disparity" to refer to pixel-wise distance between views (by abuse of language, two points in a vertical image pair are separated by "vertical disparity").

Thus, given the input set $\mathcal{I} = \{L_{tl}, L_{tr}, L_{bl}, L_{br}\}$, two disparity maps can be computed at each input viewpoint. Let us take the view on the top left L_{tl} as an example, disparity can be computed either between the horizontal pair (L_{tl}, L_{tr}) or between the vertical pair (L_{tl}, L_{bl}) as

$$d_1 = \text{DNet}(L_{tl}, L_{tr}), \tag{1}$$

$$d_2 = \mathcal{R}^{-1} \circ \text{DNet}(\mathcal{R}(L_{tl}), \mathcal{R}(L_{bl})). \tag{2}$$

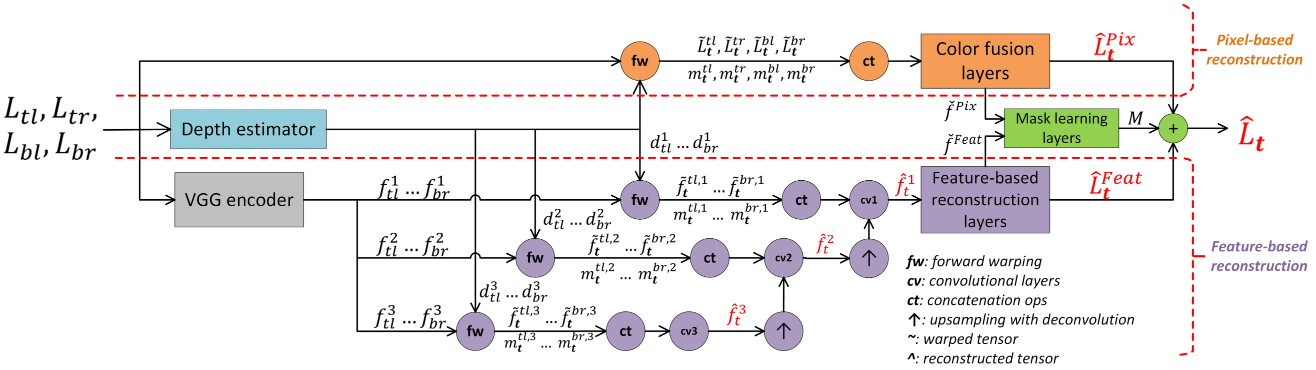

$L_{tl}, L_{tr},$
L_{bl}, L_{br}
Depth estimator
VGG encoder
fw
$\tilde{L}_t^{tl}, \tilde{L}_t^{tr}, \tilde{L}_t^{bl}, \tilde{L}_t^{br}$
$m_t^{tl}, m_t^{tr}, m_t^{bl}, m_t^{br}$
ct
Color fusion layers
$\hat{L}_t^{Pix}$
Pixel-based reconstruction
$\check{f}^{Pix}$
$\check{f}^{Feat}$
Mask learning layers
M
+
$\hat{L}_t$
$d_{tl}^1 \dots d_{br}^1$
$f_{tl}^1 \dots f_{br}^1$
$\tilde{f}_t^{tl,1} \dots \tilde{f}_t^{br,1}$
$m_t^{tl,1} \dots m_t^{br,1}$
cv1
$\hat{f}_t^1$
Feature-based reconstruction layers
$\hat{L}_t^{Feat}$
$d_{tl}^2 \dots d_{br}^2$
$f_{tl}^2 \dots f_{br}^2$
$\tilde{f}_t^{tl,2} \dots \tilde{f}_t^{br,2}$
$m_t^{tl,2} \dots m_t^{br,2}$
cv2
$\hat{f}_t^2$
$d_{tl}^3 \dots d_{br}^3$
$f_{tl}^3 \dots f_{br}^3$
$\tilde{f}_t^{tl,3} \dots \tilde{f}_t^{br,3}$
$m_t^{tl,3} \dots m_t^{br,3}$
cv3
$\hat{f}_t^3$
↑
fw: forward warping
cv: convolutional layers
ct: concatenation ops
↑: upsampling with deconvolution
~: warped tensor
^: reconstructed tensor
Feature-based reconstruction

Figure 2. Overview of our end-to-end framework. Given the corner views $\{L_{tl}, L_{tr}, L_{bl}, L_{br}\}$ as input, the depth estimator (blue) predicts depths $\{d_{tl}, d_{tr}, d_{bl}, d_{br}\}$. PixRNet (orange) reconstructs the target view $\hat{L}_{\mathbf{t}}^{\text{Pix}}$ based on the warped views $\{\widetilde{L}_{\mathbf{t}}^{tl}, \widetilde{L}_{\mathbf{t}}^{tr}, \widetilde{L}_{\mathbf{t}}^{bl}, \widetilde{L}_{\mathbf{t}}^{br}\}$. Parallelly, FeatRNet (purple) infers the features of the target view based on warped multi-scale input features, extracted from bottom layers of VGG network. The view $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ is then reconstructed. Finally, in the fusion module (green), $\hat{L}_{\mathbf{t}}^{\text{Pix}}$ and $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ are merged using a soft mask M, which is learnt from $\check{f}^{\text{Pix}}$ and $\check{f}^{\text{Feat}}$, denoting input feature maps of the last layer of PixRNet and FeatRNet, respectively.

DNet is a convolutional neural network which estimates disparity between two stereo views. In this work, we employ a pre-trained PWC-Net model which is then finetuned by light field image pairs on the same row. The symbols $\mathcal{R}(\cdot)$ and $\mathcal{R}^{-1}(\cdot)$ are counterclockwise and clockwise rotation of 90°, which enables to treat vertical pairs in the same way as horizontal ones.

A better map can be obtained by applying a simple pixel-wise fusion of d_1 and d_2. Using either d_1 or d_2, L_{tr}, L_{bl} and L_{br} are projected to the top left position (the position of L_{tl}). The corresponding warping error e_1 (warping using d_1) or e_2 (warping using d_2), is computed by summing on the three RGB color channels of the three warped views. Finally, for each pixel $\mathbf{p}$, the disparity value is selected as:

$$k' = \arg\min_k e_k(\mathbf{p}), d(\mathbf{p}) = d_{k'}(\mathbf{p}). \tag{3}$$

This part of the work has been inspired from Jiang *et al.* [25], which uses FlowNet2 [26] as disparity estimation module for corner views. Instead, we use the lightweight PWC-Net architecture which makes possible the end-to-end learning including other modules.

3.3. Pixel-wise reconstruction

The pixel-wise reconstruction module (PixRNet) follows the conventional DIBR approach to generate novel views. In particular, based on the estimated disparity maps, input views are warped to the target position and then fused to generate the final view. Similar design can be found in [9]. Apart from the fact that [9] first infers the disparity map at the target position and then employs backward projection, and our scheme applies forward projection, the main advantage of our scheme is the use of disparity-dependent interpolation which handles occlusion.

Interpolation with occlusion handling. Let us project the pixel $\mathbf{p} = (x_p, y_p)$ from the input viewpoint $\mathbf{i}$ to the target viewpoint $\mathbf{t}$, at a position with non-integer coordinates $\tilde{\mathbf{p}} = (x_{\tilde{p}}, y_{\tilde{p}})$, with the disparity value $d_{\mathbf{i}}(\mathbf{p})$:

$$\tilde{\mathbf{p}} = \mathbf{p} + (\mathbf{t} - \mathbf{i})d_{\mathbf{i}}(\mathbf{p}). \tag{4}$$

The pixel value $\widetilde{L}_{\mathbf{t}}(\mathbf{q})$ at integer coordinates $\mathbf{q} = (x_q, y_q)$ is interpolated from nearby values $L_{\mathbf{i}}(\mathbf{p})$ as

$$\widetilde{L}_{\mathbf{t}}(\mathbf{q}) = \frac{\sum_{\mathbf{p}} L_{\mathbf{i}}(\mathbf{p})W(\mathbf{p}, \mathbf{q})}{\sum_{\mathbf{p}} W(\mathbf{p}, \mathbf{q})}. \tag{5}$$

The computation of the weights $W(\mathbf{p}, \mathbf{q})$ is crucial for end-to-end learning performance. Three concerns should be addressed: 1/-the weight computation should be differentiable; 2/-as in traditional interpolation, the distance separating two pixels should be reflected in the weight; 3/-occlusions should be handled. Therefore, we propose

$$W(\mathbf{p}, \mathbf{q}) = w_D(\mathbf{p}, \mathbf{q}) w_d(\mathbf{p}), \tag{6}$$

with w_D being a coordinate distance metrics

$$w_D(\mathbf{p}, \mathbf{q}) = l(x_{\tilde{p}}, x_q) l(y_{\tilde{p}}, y_q), \tag{7}$$

where

$$l(x_1, x_2) = \begin{cases} (1 - |x_1 - x_2|), & \text{if } |x_1 - x_2| < 1; \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

and w_d a term handling occlusions defined as

$$w_d(\mathbf{p}) = \exp(-\lambda d_{\mathbf{i}}^*(\mathbf{p})). \tag{9}$$

The disparity map $d_{\mathbf{i}}$ is normalized between 0 and 1 to become $d_{\mathbf{i}}^*$. By taking the exponential function, w_d gives more importance to the foreground pixels (small normalized disparity values) rather than background ones (large normalized disparity values). Disparity normalization also avoids weight saturation at large disparity values.

Disocclusion handling. We concatenate thereby four warped views $\{\widetilde{L}^{tl}_{\mathbf{t}}, \widetilde{L}^{tr}_{\mathbf{t}}, \widetilde{L}^{bl}_{\mathbf{t}}, \widetilde{L}^{br}_{\mathbf{t}}\}$ and the corresponding disocclusion masks $\{m^{tl}_{\mathbf{t}}, m^{tr}_{\mathbf{t}}, m^{bl}_{\mathbf{t}}, m^{br}_{\mathbf{t}}\}$. The detection of the disocclusion mask is straightforward with forward warping, which identifies spatial positions having no projected pixels in their neighborhood, *i.e*. Eq.(7) equals to zero for all $\mathbf{p}$. The inpainting on the disoccluded areas is then handled by a small network of 4 convolutional layers to obtain the reconstructed view $\hat{L}^{\text{Pix}}_t$. The loss function is computed as the mean of absolute differences (MAD) between the reconstructed view and the ground truth:

$$\mathcal{L}_1 = \mathrm{MAD}(\hat{L}^{\text{Pix}}_{\mathbf{t}}, L_{\mathbf{t}}). \tag{10}$$

3.4. Feature-based reconstruction

Due to inconsistencies between disparity estimates for the different input views, a simple fusion of the projected views in the pixel space may lead to blurriness in highly textured zones as well as on object contours. Thus, we propose the feature-based reconstruction module (FeatRNet) as a complementary module of PixRNet.

For each input view $L_{\mathbf{i}}$, we extract low-level features:

$$\forall L_{\mathbf{i}} \in \mathcal{I}, \{f^1_{\mathbf{i}}, f^2_{\mathbf{i}}, f^3_{\mathbf{i}}\} = \mathrm{FeatExt}(L_{\mathbf{i}}), \tag{11}$$

with $\mathrm{FeatExt}(\cdot)$ being the operator that extracts features from the layers *relu1_2,relu2_2* and *relu3_4* of a pre-trained VGG19-Net, and $f^s_{\mathbf{i}}$ being feature volumes at scale s (the resolution of the feature maps in $f^{s+1}_{\mathbf{i}}$ is half of that in $f^s_{\mathbf{i}}$). These features are then warped to the target position in a similar manner as described in Section 3.3 for the pixels:

$$\forall \mathbf{i}, \forall s, \{\widetilde{f}^{\mathbf{i},s}_{\mathbf{t}}, m^{\mathbf{i},s}_{\mathbf{t}}\} = \mathrm{Warp}(f^s_{\mathbf{i}}, \mathbf{t}). \tag{12}$$

The warped features are input to convolutional layers to infer a feature volume of the target view at each scale s:

$$\hat{f}^s_{\mathbf{t}} = \begin{cases} \mathrm{Conv}(\{\widetilde{f}^{\mathbf{i},s}_{\mathbf{t}}, m^{\mathbf{i},s}_{\mathbf{t}}, \forall \mathbf{i}\}), & \text{if } s = 3; \\ \mathrm{Conv}(\{\widetilde{f}^{\mathbf{i},s}_{\mathbf{t}}, m^{\mathbf{i},s}_{\mathbf{t}}, \forall \mathbf{i}\}, \uparrow \hat{f}^{s+1}_{\mathbf{t}}), & \text{if } s = 1, 2. \end{cases} \tag{13}$$

At scale s=1 and 2, the inferred features at the previous scale $s+1$ are upsampled by 2 and fed into the network as well. The upsampling operator $\uparrow$ is implemented by a deconvolution layer. Finally, the target view $\hat{L}^{\text{Feat}}_{\mathbf{t}}$ is reconstructed based on features at the finest scale $\hat{f}^1_{\mathbf{t}}$. The reconstruction is supervised both in color and feature space by computing

$$\mathcal{L}_2 = \mathrm{MAD}(\hat{L}^{\text{Feat}}_t, L_t) + \sum_{s=1}^{3} \gamma_i \mathrm{MAD}(\hat{f}^s_{\mathbf{t}}, f^s_{\mathbf{t}}), \tag{14}$$

where the second term in Eq.(14) represents the difference between the inferred features and those of the ground truth target view.

Note that the use of features has been exploited in recent works for light field reconstruction. However, in most of these works [3, 10], the reconstruction is supervised by a perceptual loss minimizing the distance between features computed on the reconstructed view and those of the reference view. Here, we propose instead a bottom-up approach. We first compute the target view features by warping the warped source view VGG features. The target view is then inferred from the generated target view features. This is motivated by the intuition that VGG features optimized for object recognition can be good texture generative models.

3.5. End-to-end learning with fusion

End-to-end learning is finally performed including a fusion module to merge $\hat{L}^{\text{Pix}}_{\mathbf{t}}$ and $\hat{L}^{\text{Feat}}_{\mathbf{t}}$, making the final reconstruction $\hat{L}_{\mathbf{t}}$ well perform in both highly textured and textureless areas. Based on input feature maps of the last layer of PixRNet and FeatRNet, the fusion module learns a soft mask M with values between 0 and 1 (forced by sigmoid activation), which minimizes the pixel-wise reconstruction error:

$$\mathcal{L} = \mathrm{MAD}(\hat{L}_{\mathbf{t}}, L_{\mathbf{t}}), \tag{15}$$

with

$$\hat{L}_{\mathbf{t}} = M\hat{L}^{\text{Pix}}_{\mathbf{t}} + (1 - M)\hat{L}^{\text{Feat}}_{\mathbf{t}}. \tag{16}$$

4. Training details

We provide in the supplementary materials the structure details for layers of PixRNet, FeatRNet and the fusion module. The structure of PWC-Net, which is used as the disparity estimation module, can be found in [27].

Training schedule. End-to-end learning from scratch for such a network containing multiple modules can be intractable. In order to make sure that each module converges well and the final view inference is correctly learned, we follow a specific training schedule. We first finetune a pretrained PWC-Net with stereo pairs of light field views to make it adapt to disparity estimation. Then, PixRNet and FeatRNet are trained separately using the loss functions of Eq.(10) and Eq.(14) respectively. At this stage, the weights of PWC-Net are fixed for two reasons. The first is to accelerate the model convergence. The second reason is that, to reduce the size of the complete model, we constrain the two reconstruction schemes to use the same disparity. Finally, an end-to-end training including PWC-Net, PixRNet, FeatRNet and the fusion module is performed. Note that at this final stage, as our purpose is to minimize the pixel-wise reconstruction error (Eq.(15)), the training is no longer supervised by the feature-level reconstruction errors.

Our training data includes 94 synthetic light field scenes [28, 29] and 100 real world scenes captured by a Lytro Illum camera [9]. The model is first trained on synthetic light fields, and then further finetuned with real ones. For both training and finetuning, we work on light field patches of size 160×160 and the batch size is 5. The model is trained with a fixed learning rate of 0.00001 and the hyperparameters are set as $\lambda = 10, \gamma_1 = 1/64, \gamma_2 = 1/32, \gamma_3 = 1/4$. The training takes approximately 5 days on a GPU Tesla V100 with 32GB of memory. Our work is implemented with the *tensorflow* package.

5. Experimental results

5.1. Synthetic data

We evaluate our framework with test light fields from several synthetic datasets [28, 29, 30]. Our approach is compared against four state-of-the-art view synthesis methods for light fields which represent well the recent trend in the domain: conventional DIBR in a deep learning framework (DeepBW [9]), synthesis via multi-layer scene representation with learning (LLFF [3]) or without learning (Soft3D [8]), and view interpolation based on learned EPI structure (EPI [11]).

All reference methods except LLFF take 2×2 corner views to generate intermediate views. The released model of LLFF requires at least 5 input views. Thus, for comparison purposes, LLFF is tested with 2×2 views together with a fifth view which is the horizontal immediate neighbor to the view L_{tl}. As the performance of learning-based methods can be highly impacted by the training data, for a fair comparison, all the pre-trained models are finetuned with the same datasets that our model is trained on. We also replace the estimated camera poses required by LLFF with ground truth ones, in order to make sure that the error is merely due to the reconstruction pipeline.

Table 1 compares the reconstruction quality of the center view in terms of PSNR. The vanilla version of our model is named FPFR. FPFR* refers to “test-time augmentation” of our model: the test scene is rotated and flipped before being processed by the network, and the final reconstructed image is computed as the average of eight reconstructed views (after inverse rotation and flipping operation) based on different versions of the same scene. Note that for comparison purposes, PixRNet and FeatRNet can be trained separately with the disparity estimation module to become two fully independent view synthesis models on their own. We denote these two models “PurePix” and “PureFeat”, which are also trained in an end-to-end fashion. Test light fields are arranged according to their disparity range (from dense to sparse). On average, our method (FPFR and FPFR*) significantly outperforms other methods: a gain of nearly 2dB is observed against the best reference method. Our method performs especially well for highly textured scenes *e.g. stillife*. Reconstruction error maps are shown in Fig. 3. One can observe that our method generates less error on the object contours and thin structures in textured regions (*e.g.* the tablecloth in *stilllife* and the wallpaper in *sideboard*).

Furthermore, Fig. 4(a) shows that FPFR consistently generates high quality views across different viewpoints, whereas the reconstruction quality decreases for other methods when the target view is distant from the input views.

Note that the single mode schemes PurePix and PureFeat also obtain competitive results in Table 1. In the same vein as DeepBW [9], PurePix is on average 3.7dB better than DeepBW, especially for sparse scenes. We believe that it is mainly due to the occlusion and disocclusion handling in our PixRNet design.

5.2. Real-world data

For real-world experiments, we use the same training and test sets as used in [9]. For a fair comparison, all learning-based models are finetuned with the same dataset. Table 2 shows that our approach achieves the highest PSNRs. Note that a gain of 4.5dB is obtained for the scene *leaves*. As for synthetic data, similar observations can be made in Fig. 3 for real-world scenes: we obtain more accurate contours and well preserved textures.

5.3. Ablation study

Pixel vs. feature To put in evidence the different tasks the pixel-wise and feature-based reconstruction carry out, in Fig. 5 we show examples of feature maps taken from the last layers in PixRNet and FeatRNet, from which the color view is reconstructed. We observe highly enhanced textures and clear line structures in FeatRNet feature maps, whereas PixRNet feature maps can provide information such as brightness, color and contrast, etc.

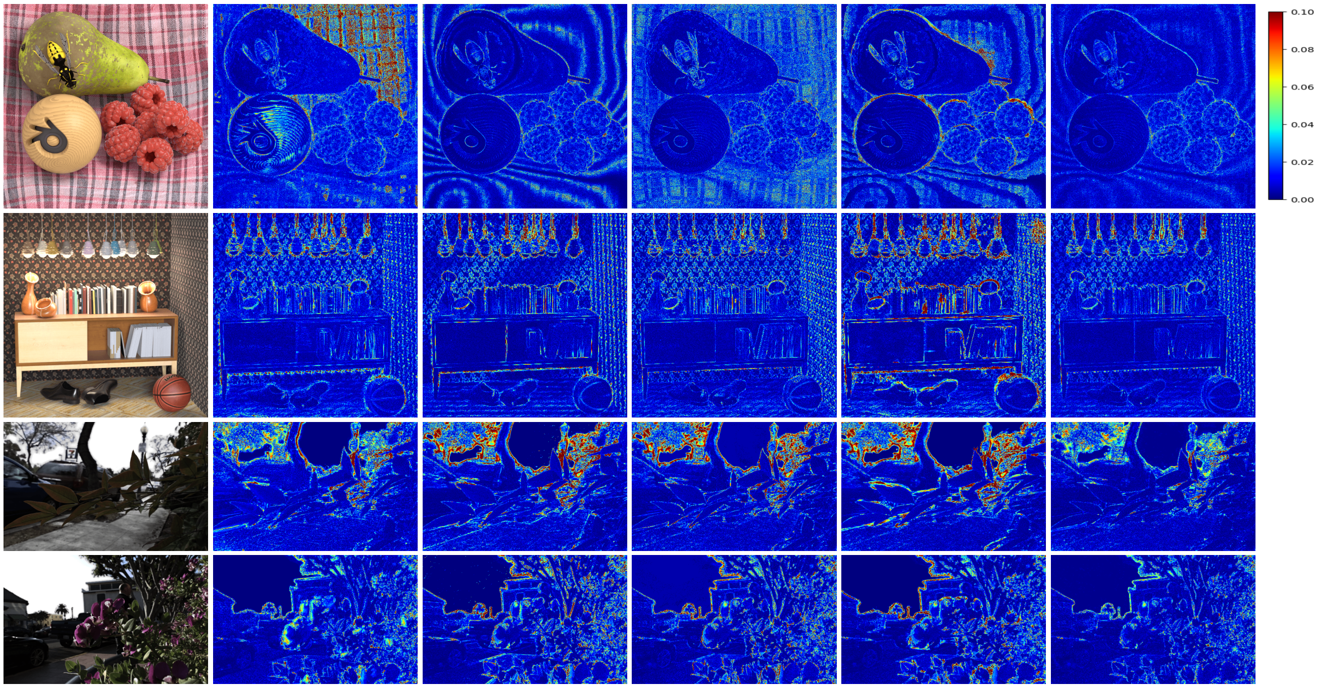
Feature-based reconstruction vs. perceptual loss A common practice to optimize the view reconstruction quality via the feature space is to apply the so-called perceptual loss [32] when performing end-to-end learning. To valid our concept of merging the pixel-wise reconstruction with the feature-based one, we compare FPFR with the single mode scheme PurePix learned end-to-end with a perceptual loss. In experiments, we observe that FPFR is about 0.7dB better.

Fusion vs. single mode With Fig. 6, we first analyse the convergence behavior at different stages of the network (the output of PixRNet $\hat{L}_{\mathbf{t}}^{\text{Pix}}$ in cyan, that of FeatRNet $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ in red, and the final output $\hat{L}_{\mathbf{t}}$ in blue). The fusion pushes each mode to excel in its domain: the image $\hat{L}_{\mathbf{t}}^{\text{Pix}}$ is more accurate on colors and low frequencies, whereas $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ contains a higher level of texture than the reference image (this explains the clear degradation of the PSNR for $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ during training). Therefore, the final image $\hat{L}_{\mathbf{t}}$ (blue curve) obtains better quality both in low and high frequencies (see Fig. 1).

In Fig. 6, we also compare FPFR with single mode schemes PurePix (in green) and PureFeat (in yellow). A clear advantage is observed for FPFR.

LFs	Disparity range	DeepBW[9]	Soft3D[8]	LLFF[3]	EPI[11]	PurePix	PureFeat	FPFR	FPFR*
mona†	[-5,5] (10)	38.90	40.92	41.20	37.54	39.90	40.35	42.47	**42.86**
butterfly†	[-6,8] (14)	40.68	42.75	41.35	39.61	41.64	39.33	42.69	**42.96**
buddha†	[-10,6] (16)	41.08	41.86	40.66	40.05	41.24	40.99	42.78	**43.06**
cotton⋆	[-9,9] (18)	47.24	**48.95**	47.07	47.97	48.18	46.45	48.58	48.76
boxes⋆	[-7,13] (20)	33.64	32.14	**34.97**	31.65	33.44	33.90	33.86	34.46
dino⋆	[-10,10] (20)	38.41	41.69	41.26	38.44	40.63	39.38	42.66	**42.98**
sideboard⋆	[-10,12] (22)	30.91	30.23	**32.33**	27.30	29.43	30.79	31.85	32.18
Toy_bricks◇	[-1,22] (23)	28.90	36.58	37.98	31.46	36.55	36.01	38.84	**39.35**
Elec_devices◇	[-10,17] (27)	34.09	36.24	36.76	31.55	35.53	34.87	37.63	**38.03**
stilllife†	[-16,16] (32)	26.29	34.73	32.73	32.02	33.96	32.77	36.39	**37.05**
Lion◇	[-5,29] (34)	28.05	35.18	35.22	33.91	35.10	34.99	35.47	**35.59**
Two_vases◇	[-5,39] (44)	25.65	32.49	35.82	29.09	33.46	34.78	35.56	**35.99**
Sculpture◇	[-26,34] (60)	22.31	29.15	29.68	26.22	28.56	29.51	30.09	**30.30**
Bear◇	[-38,53] (91)	18.36	28.00	33.22	23.40	29.00	32.64	31.87	**33.84**
Average	-	32.49	36.49	36.43	33.59	36.19	36.20	37.92	**38.39**

Table 1. Quantitative results (PSNR) for the reconstructed central view on the synthetic test light fields. The corresponding datasets are indicated by symbols: ⋆ [28], ◇ [29] and † [30].



Light field view DeepBW [9] Soft3D [8] LLFF [3] EPI [11] FPFR

Figure 3. Visual comparison of reconstruction error maps for different methods. More results are provided in supplementary materials.

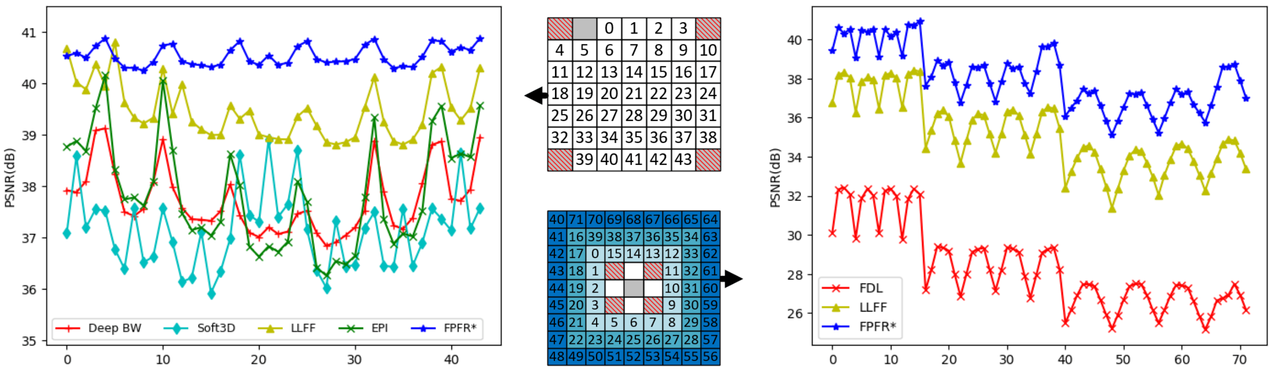
LFs	DeepBW[9]	Soft3D[8]	LLFF[3]	EPI[11]	FPFR*
Cars	31.53	27.68	29.06	28.17	**32.25**
Flower1	33.13	30.29	30.00	30.44	**34.49**
Flower2	31.95	30.52	28.90	29.26	**34.19**
Rock	34.32	32.67	32.60	32.46	**36.75**
Leaves	27.97	27.34	27.74	26.48	**32.53**
Seahorse	32.03	30.41	28.50	26.62	**34.97**
Average	31.82	29.82	29.47	28.90	**33.91**

Table 2. Quantitative results (PSNR) for the reconstructed view (5,5) on the real-world data (8 × 8 views) [9].

Single-scale vs. multi-scale A single-scale architecture ($s = 1$) for feature-based reconstruction is compared with its multi-scale counterparts ($s = 1, 2, 3$). In experiments, a gain of about 0.5dB is observed in favor of the multi-scale architecture.

5.4. Extrapolation

Extrapolation of light fields with plausible disocclusions can be a more difficult task than interpolation, since less information of the target view is known in the input views. In Fig. 4(c), we evaluate the inherent capacity of our frame-



▨		0	1	2	3	▨
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31
32	33	34	35	36	37	38
▨	39	40	41	42	43	▨

40	71	70	69	68	67	66	65	64
41	16	39	38	37	36	35	34	63
42	17	0	15	14	13	12	33	62
43	18	1	▨		▨	11	32	61
44	19	2				10	31	60
45	20	3	▨		▨	9	30	59
46	21	4	5	6	7	8	29	58
47	22	23	24	25	26	27	28	57
48	49	50	51	52	53	54	55	56

(a) PSNR for **interpolated** viewpoints (b) View indices (c) PSNR for **extrapolated** viewpoints

Figure 4. Average PSNR over 8 synthetic scenes ([28, 30]) for each novel viewpoint. (a) Interpolation. (c) Extrapolation. (b) View indices for interpolation and extrapolation. 4 input views (red slash) are used for FPFR, DeepBW[9], EPI[11] or FDL[31], whereas 5 input views (grey) are used for LLFF[3].

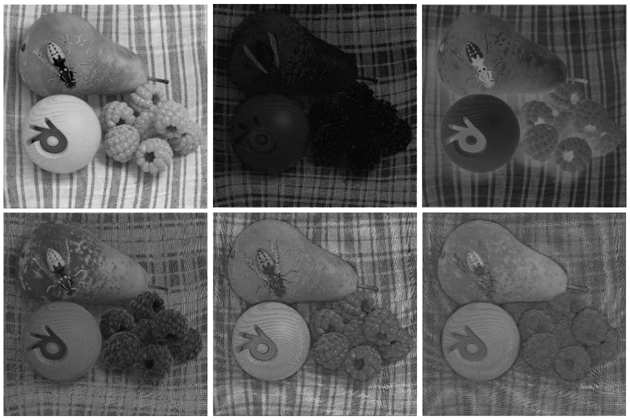


Figure 5. Feature maps taken from the last latent layers of PixRNet and FeatRNet.

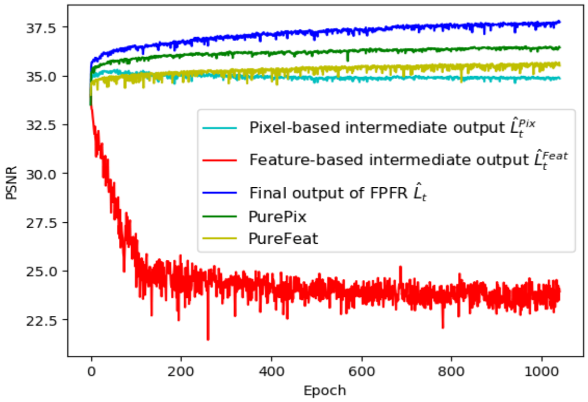


Figure 6. Learning curves of different end-to-end schemes (FPFR, PurePix and PureFeat). For FPFR, the curves for intermediate outputs ($\hat{L}_t^{\text{Pix}}$ and $\hat{L}_t^{\text{Feat}}$) are also shown.

work to extrapolate against two reference methods FDL[31] and LLFF [3]. As input, our method and FDL take 4 corner views (red slashes in Fig. 4(b)) of a subset of 3 × 3 views with narrow baseline. Since LLFF requires at least 5 views, the central view is also included in input views (grey in Fig. 4(b)). The output is an extended light field with 9×9 views (4× baseline). One can observe that our method outperforms reference methods by a large margin. Wider the extended baseline, more important is our gain. Note that both LLFF and our model are trained for interpolation tasks, here we evaluate the inherent capacity to extrapolate without any further training.

5.5. Limitations

Relying on disparity estimation, our method can be subject to errors for non-Lambertian surfaces. Moreover, even though our method is demonstrated to be efficient for structured light fields, for unstructured ones, future works would be needed by coupling the method with appropriate pose estimation methods.

6. Conclusion

We have presented a novel learning-based model for light field view synthesis. In order to obtain high quality reconstruction both in low and high frequencies, end-to-end learning is performed including a pixel-wise reconstruction module and a feature-based reconstruction module. Experiments have demonstrated that the proposed model achieves state-of-the-art performance both for synthetic and real-world light fields.

7. Acknowledgement

The work has been funded by the EU H2020 Research and Innovation Program under grant agreement No.694122 (ERC advanced grant CLIM). We would also like to show our gratitude to Ben Mildenhall for providing us with LLFF code and to Dr. Zhaolin Xiao for inspiring discussions.

References

[1] Bennett Wilburn, Neel Joshi, Vaibhav Vaish, Eino-Ville Talvala, Emilio Antunez, Adam Barth, Andrew Adams, Mark Horowitz, and Marc Levoy. High performance imaging using large camera arrays. *ACM Trans. on Graphics*, 24(3):765–776, Jul. 2005. 1

[2] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Techn.(CCGIT)*, pages 31–42, 1996. 1

[3] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Trans. on Graphics*, 2019. 1, 2, 3, 5, 6, 7, 8

[4] Lixin Shi, Haitham Hassanieh, Abe Davis, Dina Katabi, and Fredo Durand. Light field reconstruction using sparsity in the continuous fourier domain. *ACM Trans. on Graphics*, 34(1):12, 2014. 1, 2

[5] Sven Wanner and Bastian Goldluecke. Variational light field analysis for disparity estimation and super-resolution. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(3):606–619, Aug. 2013. 1, 2

[6] Suren Vagharshakyan, Robert Bregovic, and Atanas Gotchev. Light field reconstruction using shearlet transform. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(1):133–147, 2018. 1, 2

[7] Zhoutong Zhang, Yebin Liu, and Qionghai Dai. Light field from micro-baseline image pair. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3800–3809, 2015. 1

[8] Eric Penner and Li Zhang. Soft 3D reconstruction for view synthesis. *ACM Trans. on Graphics*, 36(6):235:1–235:11, 2017. 1, 2, 6, 7

[9] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Trans. on Graphics*, 35(6):193:1–193:10, 2016. 1, 2, 3, 4, 5, 6, 7, 8

[10] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. on Graphics*, 37(4):65:1–65:12, 2018. 1, 2, 3, 5

[11] Gaochang Wu, Yebin Liu, Qionghai Dai, and Tianyou Chai. Learning sheared epi structure for light field reconstruction. *IEEE Trans. Image Process.*, 28(7):3261–3273, 2019. 2, 3, 6, 7, 8

[12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 3

[13] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Siggraph*, pages 43–54, 1996. 2

[14] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Trans. on Graphics*, 32(3):1–12, 2013. 2

[15] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Learning to predict new views from the world's imagery. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5515–5524, 2016. 2, 3

[16] Anat Levin and Fredo Durand. Linear view synthesis using a dimensionality gap light field prior. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1831–1838, 2010. 2

[17] Sudipta N Sinha, Johannes Kopf, Michael Goesele, Daniel Scharstein, and Richard Szeliski. Image-based rendering for scenes with reflections. *ACM Trans. on Graphics*, 31(4):100–1, 2012. 2

[18] Srinivasan Pratul, P., Tongzhou Wang, Ashwin Sreelal, Ravi Ramamoorthi, and Ren Ng. Learning to synthesize a 4D RGBD light field from a single image. In *IEEE Int. Conf. on Computer Vision (ICCV)*, pages 2262–2270, 2017. 2

[19] Gaochang Wu, Mandan Zhao, Liangyong Wang, Qionghai Dai, Tianyou Chai, and Yebin Liu. Light field reconstruction using deep convolutional network on EPI. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1638–1646, 2017. 2

[20] Yunlong Wang, Fei Liu, Zilei Wang, Guangqi Hou, Zhenan Sun, and Tieniu Tan. End-to-end view synthesis for light field imaging with pseudo 4DCNN. In *Eur. Conf. on Computer Vision (ECCV)*, pages 333–348, 2018. 2, 3

[21] Shubham Tulsiani, Richard Tucker, and Noah Snavely. Layer-structured 3D scene inference via view synthesis. In *Eur. Conf. on Computer Vision (ECCV)*, pages 302–317, 2018. 3

[22] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H. Kim, and Jan Kautz. Extreme view synthesis. In *IEEE Int. Conf. on Computer Vision (ICCV)*, pages 7781–7790, 2019. 3

[23] Nan Meng, Hayden Kwok-Hay So, Xing Sun, and Edmund Lam. High-dimensional dense residual convolutional neural network for light field reconstruction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019. 3

[24] Henry Wing Fung Yeung, Junhui Hou, Jie Chen, Yuk Ying Chung, and Xiaoming Chen. Fast light field reconstruction with deep coarse-to-fine modeling of spatial-angular clues. In *Eur. Conf. on Computer Vision (ECCV)*, pages 137–152, 2018. 3

[25] Xiaoran Jiang, Jinglei Shi, and Christine Guillemot. A learning based depth estimation framework for 4D densely and sparsely sampled light fields. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2257–2261, 2019. 4

[26] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1647 – 1655, 2017. 4

[27] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 8934–8943, 2018. 5

[28] Katrin Honauer, Ole Johannsen, Daniel Kondermann, and Bastian Goldluecke. A dataset and evaluation methodology for depth estimation on 4D light fields. In *Asian Conf. on Computer Vision (ACCV)*, pages 19–34, 2016. 5, 6, 7, 8

[29] Jinglei Shi, Xiaoran Jiang, and Christine Guillemot. A framework for learning depth from a flexible subset of dense and sparse light field views. *IEEE Trans. Image Process.*, 28(12):5867–5880, Dec 2019. 5, 6, 7

[30] Sven Wanner, Stephan Meister, and Bastian Goldluecke. Datasets and benchmarks for densely sampled 4D light fields. In *Conf. on Vision, Modeling & Visualization (VMV)*, pages 225–226, 2013. 6, 7, 8

[31] Mikael Le Pendu, Christine Guillemot, and Aljosa Smolic. A fourier disparity layer representation for light fields. *IEEE Trans. Image Process.*, 28(11):5740–5753, Nov 2019. 8

[32] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Eur. Conf. on Computer Vision (ECCV)*, pages 694–711, 2016. 6