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.

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.
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.

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 \( i = (u_i, v_i) \) is referred to as \( L_i \).

We aim at reconstructing the novel view at the target position \( t \) from a set of input views \( I = \{L_1, ..., L_{16}\} \). In this work, the set \( I \) contains sparsely sampled \( 2 \times 2 \) 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

\[
\begin{align*}
&d_1 = DNet(L_{tl}, L_{tr}), \\
&d_2 = \mathcal{R}^{-1} \circ DNet(\mathcal{R}(L_{tl}), \mathcal{R}(L_{bl})).
\end{align*}
\]
This part of the work has been inspired from Jiang et al. Finally, for each pixel the three RGB color channels of the three warped views.

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 $90^\circ$, 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 $p$, the disparity value is selected as:

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

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 $p = (x_p, y_p)$ from the input viewpoint $i$ to the target viewpoint $t$, at a position with non-integer coordinates $\tilde{p} = (x_\tilde{p}, y_\tilde{p})$, with the disparity value $d_i(p)$:

$$\tilde{p} = p + (t - i)d_i(p).$$

The pixel value $\tilde{L}_t(q)$ at integer coordinates $q = (x_q, y_q)$ is interpolated from nearby values $L_i(p)$ as

$$\tilde{L}_t(q) = \frac{\sum_p W(p,q) L_i(p)}{\sum_p W(p,q)}.$$  \hspace{1cm} (5)

The computation of the weights $W(p,q)$ 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

$$W(p,q) = w_D(p,q) w_d(p),$$

with $w_D$ being a coordinate distance metrics

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

where

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

and $w_d$ a term handling occlusions defined as

$$w_d(p) = \exp(-\lambda d_i^2(p)).$$

The disparity map $d_i$ is normalized between 0 and 1 to become $\hat{d}_i$. 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 \( \{ L_k^{tl}, L_k^{tr}, L_k^{bl}, L_k^{br} \} \) and the corresponding disocclusion masks \( \{ m_k^{tl}, m_k^{tr}, m_k^{bl}, m_k^{br} \} \). The detection of the disocclusion mask is straightforward with forward resolution of the feature maps in the target view.

The warped views in the pixel space may lead to blurriness in highly textured zones as well as on object contours. Thus, we propose a feature-based reconstruction module (FeatRNet) as a complementary module of PixRNet.

For each input view \( L_i \), we extract low-level features:

\[
\forall L_i \in I, \{ f_i^1, f_i^2, f_i^3 \} = \text{FeatExt}(L_i),
\]

with \( \text{FeatExt}() \) being the operator that extracts features from the layers \( \text{relu}_2, \text{relu}_2, \text{relu}_3 \) of a pre-trained VGG19-Net, and \( f_i^s \) being feature volumes at scale \( s \) (the resolution of the feature maps in \( f_i^{s+2} \) is half of that in \( f_i^s \)). These features are then warped to the target position in a similar manner as described in Section 3.3 for the pixels:

\[
\forall i, \forall s, \{ \tilde{m}_k^s, m_k^s \} = \text{Warp}(f_i^s, t).
\]

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

\[
\tilde{f}_k^s = \begin{cases} 
\text{Conv}(\{ f_i^s, m_i^s, \forall i \}), & \text{if } s = 3; \\
\text{Conv}(\{ f_i^{s+1}, m_i^{s+1}, \forall i \}, \uparrow f_k^{s+1}), & \text{if } s = 1, 2.
\end{cases}
\]

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 \( \hat{L}_k^{\text{Feat}} \) is reconstructed based on features at the finest scale \( \hat{f}_k^1 \). The reconstruction is supervised both in color and feature space by computing

\[
\mathcal{L}_2 = \text{MAD}(\hat{L}_k^{\text{Feat}}, L_k) + \sum_{s=1}^{3} \gamma_s \text{MAD}(\tilde{f}_k^s, f_k^s),
\]

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 \( \hat{L}_k^{\text{Pix}} \) and \( \hat{L}_k^{\text{Feat}} \), making the final reconstruction \( \hat{L}_k \) well perform in both highly textured and textureless areas. Based on input feature maps of the last layer of PixRNet andFeatRNet, 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:

\[
\mathcal{L} = \text{MAD}(\hat{L}_k, L_k),
\]

with

\[
\hat{L}_k = MF_k^{\text{Pix}} + (1 - M)\hat{L}_k^{\text{Feat}}.
\]

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 pre-trained 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 $L_{i:j}$. 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 andFeatRNet 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. stilllife. 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 validate 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 $\hat{L}^{\text{Pix}}_{i:k}$ in cyan, that of FeatRNet $\hat{L}^{\text{Feat}}_{i:k}$ in red, and the final output $\hat{L}_i$ in blue). The fusion pushes each mode to excel in its domain: the image $\hat{L}^{\text{Pix}}_{i:k}$ is more accurate on colors and low frequencies, whereas $\hat{L}^{\text{Feat}}_{i:k}$ contains a higher level of texture than the reference image (this explains the clear degradation of the PSNR for $\hat{L}^{\text{Feat}}_{i:k}$ during training). Therefore, the final image $\hat{L}_i$ (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.
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].

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>mona</td>
<td>38.90</td>
<td>40.92</td>
<td>41.20</td>
<td>37.54</td>
<td>39.90</td>
<td>40.35</td>
<td>42.47</td>
<td>42.86</td>
</tr>
<tr>
<td>butterfly</td>
<td>40.68</td>
<td>42.75</td>
<td>41.35</td>
<td>39.61</td>
<td>41.64</td>
<td>39.33</td>
<td>42.69</td>
<td>42.96</td>
</tr>
<tr>
<td>buddha</td>
<td>41.08</td>
<td>41.86</td>
<td>40.66</td>
<td>40.05</td>
<td>41.24</td>
<td>40.99</td>
<td>42.78</td>
<td>43.06</td>
</tr>
<tr>
<td>cotton*</td>
<td>47.24</td>
<td>48.95</td>
<td>47.07</td>
<td>47.97</td>
<td>48.18</td>
<td>46.45</td>
<td>48.58</td>
<td>48.76</td>
</tr>
<tr>
<td>boxes*</td>
<td>33.64</td>
<td>32.14</td>
<td>34.97</td>
<td>31.65</td>
<td>33.44</td>
<td>33.90</td>
<td>33.86</td>
<td>34.46</td>
</tr>
<tr>
<td>dino*</td>
<td>38.41</td>
<td>41.69</td>
<td>41.26</td>
<td>38.44</td>
<td>40.63</td>
<td>39.38</td>
<td>42.66</td>
<td>42.98</td>
</tr>
<tr>
<td>sideboard*</td>
<td>30.91</td>
<td>30.23</td>
<td>32.33</td>
<td>29.30</td>
<td>30.79</td>
<td>31.85</td>
<td>32.18</td>
<td>39.35</td>
</tr>
<tr>
<td>Toy_bricks*</td>
<td>28.90</td>
<td>36.58</td>
<td>37.98</td>
<td>36.55</td>
<td>36.01</td>
<td>38.84</td>
<td>39.31</td>
<td>35.59</td>
</tr>
<tr>
<td>Elec_devices*</td>
<td>34.09</td>
<td>36.24</td>
<td>36.76</td>
<td>31.55</td>
<td>35.53</td>
<td>34.87</td>
<td>37.63</td>
<td>38.03</td>
</tr>
<tr>
<td>stilllife*</td>
<td>26.29</td>
<td>34.73</td>
<td>32.73</td>
<td>32.02</td>
<td>33.96</td>
<td>32.77</td>
<td>36.39</td>
<td>37.05</td>
</tr>
<tr>
<td>Lion*</td>
<td>28.05</td>
<td>35.18</td>
<td>35.22</td>
<td>33.91</td>
<td>35.10</td>
<td>34.99</td>
<td>35.47</td>
<td>35.59</td>
</tr>
<tr>
<td>Two_vases*</td>
<td>25.65</td>
<td>32.49</td>
<td>35.82</td>
<td>29.09</td>
<td>33.46</td>
<td>34.78</td>
<td>35.56</td>
<td>35.99</td>
</tr>
<tr>
<td>Sculpture*</td>
<td>22.31</td>
<td>29.15</td>
<td>29.68</td>
<td>26.22</td>
<td>28.56</td>
<td>29.51</td>
<td>30.09</td>
<td>30.30</td>
</tr>
<tr>
<td>Bear*</td>
<td>18.56</td>
<td>28.00</td>
<td>33.22</td>
<td>23.40</td>
<td>29.00</td>
<td>32.64</td>
<td>31.87</td>
<td>33.84</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>32.49</strong></td>
<td><strong>36.49</strong></td>
<td><strong>36.43</strong></td>
<td><strong>33.59</strong></td>
<td><strong>36.19</strong></td>
<td><strong>36.20</strong></td>
<td><strong>37.92</strong></td>
<td><strong>38.39</strong></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Cars</td>
<td>31.53</td>
<td>27.68</td>
<td>29.06</td>
<td>28.17</td>
<td>32.25</td>
</tr>
<tr>
<td>Flower1</td>
<td>33.13</td>
<td>30.29</td>
<td>30.00</td>
<td>30.44</td>
<td><strong>34.49</strong></td>
</tr>
<tr>
<td>Flower2</td>
<td>31.95</td>
<td>30.52</td>
<td>28.90</td>
<td>29.26</td>
<td><strong>34.19</strong></td>
</tr>
<tr>
<td>Rock</td>
<td>34.32</td>
<td>32.67</td>
<td>32.60</td>
<td>32.46</td>
<td><strong>36.75</strong></td>
</tr>
<tr>
<td>Leaves</td>
<td>27.97</td>
<td>27.34</td>
<td>27.74</td>
<td>26.48</td>
<td><strong>32.53</strong></td>
</tr>
<tr>
<td>Seahorse</td>
<td>32.03</td>
<td>30.41</td>
<td>28.50</td>
<td>26.62</td>
<td><strong>34.97</strong></td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>31.82</strong></td>
<td><strong>29.82</strong></td>
<td><strong>29.47</strong></td>
<td><strong>28.90</strong></td>
<td><strong>33.91</strong></td>
</tr>
</tbody>
</table>

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-
and LLFF [3]. As input, our method and FDL take 4 corner views (red slashes in Fig. 4(b)) of a subset of $3 \times 3$ 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 $9 \times 9$ views ($4 \times$ 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


