Light Field View Synthesis via Aperture Flow and Propagation Confidence Map
Nan Meng, Kai Li, Jianzhuang Liu, and Edmund Y. Lam, Fellow, IEEE

Abstract—This paper presents a learning-based approach to synthesize the view from an arbitrary camera position given a sparse set of images. A key challenge for this novel view synthesis arises from the reconstruction process, when the views from different input images may not be consistent due to obstruction in the light path. We overcome this by jointly modeling the epipolar property and occlusion in designing a convolutional neural network. We start by defining and computing an aperture flow map, which approximates the parallax and measures the pixel-wise shift between two views. While this relates to free-space rendering and can fail near the object boundaries, we further develop a propagation confidence map to address pixel occlusion in these challenging regions. The proposed method is evaluated on diverse real-world and synthetic light field scenes, and it shows outstanding performance over several state-of-the-art techniques.

Index Terms—View synthesis, image-based rendering, light field, aperture flow, epipolar property, confidence map.

I. INTRODUCTION

Perceiving the 3-dimensional (3D) nature of an object is a basic human instinct, but it can be very difficult for a computer. One reason is that the optical system cannot easily capture the geometric information of the scene, and therefore it often has difficulty recreating the visual perception [1]. Ordinarily, an image is not informative about the differences among light rays coming from various directions, as they are combined together to form the intensity at a pixel location. These differences, however, are crucial for us to perceive the world [2].

With advanced techniques and imaging setups, recording the 3D information of an object is becoming feasible [3]. One of the most promising techniques is light field photography, which uses a plenoptic camera to capture both the directions and radiance of the incident light rays [4]. The additional directional information allows a wider range of vision applications, such as depth estimation [5], [6], rendering [7], [8], refocusing [9], super-resolution [10], [11] etc. However, there is a limit to the density of the pixels that one can capture, necessitating a trade-off between spatial and angular resolution [12].

One way to relieve such a trade-off is to produce the intermediate views from the captured images. Ordinarily, approaches for view synthesis require geometric knowledge as priors in the recovery of a dense light field from the sparsely-sampled inputs [13], [14]. The new view can be reconstructed from any acquired images by projecting the pixels to proper 3D locations and re-projecting them onto the target picture [7]. However, such methods struggle with complex regions such as occlusions, as well as transparent and semi-transparent surfaces, where the depth information is difficult to calculate or estimate. Without accurate geometry information, the generated results often contain jarring artifacts [15], [16].

An alternative approach is image-based rendering (IBR), which directly reuses the pixels from the available images to produce the new views. For light field, the depth information is not necessary for the rendering process [17]. The spatio-angular redundancy makes it possible to infer the novel view from neighboring sub-aperture images (SAIs). One advantage of the IBR techniques is that they can avoid explicitly modeling the realistic geometry of the scene. However, such approaches usually require more samples to counter the undesirable aliasing effects in the outputs [7], [18].

These difficulties recently have led to investigations using the learning-based approach, which forgoes the explicit modeling of the problem and instead makes use of deep learning to approximate the ground truth with many training samples [15], [19]. The powerful representation ability of convolutional neural network (CNN) and its widespread success in vision tasks make this a promising direction [20], [21]. Generally, the CNN-based view synthesis algorithms are able to achieve relatively high-quality reconstruction, but often require a densely-sampled light field as the ground truth to acquire the accurate supervision signal [12], [22], which restricts the generalization or capacity of the algorithms. For instance, Meng et al. [23] and Yeung et al. [24] both adopt the learning framework to directly approximate the ground truth. However, the drawback is that they can only generate the views that are recorded in the dense samples.

In this work, we propose a depth-free approach to produce the novel views from arbitrary camera positions. Rather than directly synthesizing the image results, we exploit the pixel-wise epipolar relations between the given views. We first define an aperture flow map (AFM) in terms of the epipolar properties to describe the pixel-wise shift between a pair of images. One attractive characteristic of AFM is that the flow map is linearly related to the changes in camera positions, which helps in estimating the intermediate AFMs for the target view synthesis. To address the issue of occlusion, we further estimate the propagation confidence maps (PCMs), which equilibrate the radiance information from different input views to produce the final image. Finally, the coarse results
are further refined by a four-dimensional (4D) CNN with alternating filters [24].

In summary, the contributions of this paper are as follows:

- We propose a novel flow model, which we called AFM, that is tailored to the light field images to measure the pixel-wise shift distance between a given pair of images. Experimental results show that it works well on both real-world and synthetic scenes.
- The proposed approach can synthesize images from arbitrary camera positions without any experimented or estimated disparity information. Therefore, it performs consistently well on light fields with thin structures where the depth information is hard to estimate.
- We introduce the PCM to combine the pixel values from different views for the target image. It can efficiently handle occluded pixels, and therefore reduce the artifacts near the object boundaries.

II. RELATED WORK

View synthesis and IBR are closely related. Generally, the explicit geometry models are not necessary for IBR when generating new images in a light field, while view synthesis usually requires both geometry information and a few images to provide the virtual views. In this section, we review the literature of IBR algorithms, view synthesis, and recent learning-based approaches in the context of light field imaging.

Early IBR algorithms for light field rely on the characterization of the plenoptic function and treat the creation of new views as resampling [25], [26]. This approach ignores occlusion, and thus is only feasible for free space rendering or for producing the views reasonably close to the original ones. It gradually becomes clear that interpolation of plausible views in high quality requires either intensive sampling or knowledge about the scene [7], [27]. Therefore, a different set of approaches to light field rendering attempt to infer at least some geometry information, which include methods that rely on image registration [28], prior knowledge [29], or image correspondence and warping [30], to name a few.

View synthesis methods, on the other hand, focus on making use of explicit geometric knowledge that assists in the recovery of a dense light field. Some enforce explicit priors on the light field itself, such as sparsity in the Fourier domain [31] or shearlet transform domain [32], a patch-based Gaussian mixture model [33], or Lambertian surfaces with modest depth discontinuities [34]. However, these methods require either a specific sampling pattern or a large number of views, which limit their practical uses. Other techniques involve partial reconstruction of the scene geometry, such as a global 3D reconstruction [35] or a soft model of the geometrical relationships [36]. Some methods infer the geometry by estimating the disparity for a single view [37] or for each input view [14]. Given that an accurate depth estimation is hard to obtain, such approaches often struggle with complex scenes.

Another group of synthesis algorithms do not require explicit geometric models but rely on the feature correspondence between images. The classical approaches of this kind interpolate the intermediate views by exploiting the optical flow [38], [39]. However, given that non-Lambertian surfaces and occlusions are still challenging to the flow estimation methods [40], [41], the interpolated views tend to have artifacts near the object boundaries. Wang et al. [42] make use of the images taken from another standard camera as references to generate plausible frames for the light field video, which also increases the complexity of imaging system.

In recent years, there are several demonstrations of the effectiveness of CNN for view synthesis. The first conducted by Yoon et al. [43] investigates an end-to-end CNN to approximate roughly the angular correlations by only considering two adjacent views. Although its performance is limited, their method inspires a new branch of learning-based approaches. Later on, Flynn et al. [19] adopt the deep network to interpolate between views with wide baselines. Kalantari et al. [16] are the first to integrate sequential CNNs into the traditional pipeline. The entire model consists of disparity and color estimation modules, and each uses a CNN to realize its function. In contrast, some recent methods specially designed for light field attempt to learn directly the mapping between the sparsely-sampled inputs and labels. For instance, Wu et al. [22] exploit the epipolar property of the light field, and produce the views by recovering the epipolar plane image (EPI). Yeung et al. [24] and Meng et al. [12] apply high-dimensional convolutions to learn spatio-angular representations for light field reconstruction.

Often, the learning-based methods produce more plausible visual results due to the powerful representation-learning ability of the deep neural networks and the effective use of the supervised signals directly from the labels. However, there are also several drawbacks in the prevailing methods. For example, the rigid learning strategy heavily relies on the training data and therefore restricts the generalization of the model [16]. In addition, the end-to-end training pattern makes the model hard to reconstruct an image from a viewpoint that has not been recorded in the ground truth.

III. METHOD

We adopt the two-plane parametrization [44] to represent the 4D light field. Each light ray is represented by the intersections with two parallel planes transmitting from the spatial coordinate $x = (x, y)$ to the angular coordinate $u = (u, v)$, and thus denoted by $L := L(x, u)$. We assume that all the coordinate variables in the plenoptic function $L(x, u)$ are continuous. Therefore, the goal of IBR is to reconstruct such a function based on a set of discrete samples $L(x_i, u_j) \ (i, j \in \mathbb{N})$.

$$L(x_i, u_j) \xrightarrow{\theta} L(x, u),$$

where $\theta(\cdot)$ denotes the reconstruction algorithms.

A. Spatio-Angular Relation

For any given point $P$ in the free space, the two-plane parametrization is depicted in Fig. 1, where $F$ is the distance between the two parallel planes, and $Z$ is the disparity value of the focused point. The pixel shifts in different views can be
inferred using similar triangles. If we vary \( x \) with a distance \( \Delta x \), the angular coordinate has to change according to

\[
\Delta x = \frac{Z - F}{Z} \Delta u = \frac{\alpha F - F}{\alpha F} \Delta u = \left(1 - \frac{1}{\alpha}\right) \Delta u,
\]

where \( \alpha = \frac{Z}{F} \) denotes the disparity ratio. \( \Delta u \) is the distance between two viewpoints located at the camera plane \( II \). In the following, we pay more attention to the appearance of the view, and for the sake of illustration, we use the symbol \( L_u(\cdot) \) to denote the plenoptic function obtained from the viewpoint at location \( u \) as

\[
L_u(x) := L(x, u).
\]

An equivalent expression of Eq. (3) is

\[
L_u(x) = L_{u+\Delta u}(x + \Delta x),
\]

if we assume the pixels in an image shift by \( \Delta x \) when the viewpoint changes by \( \Delta u \).

Substituting Eq. (2) into Eq. (4), we obtain

\[
L_u(x) = L_{u+\Delta u}(x + \Delta x)
= L_{u+\Delta u}\left(x + \left(1 - \frac{1}{\alpha}\right) \Delta u\right).
\]

B. Photo-Consistency

The photo-consistency assumption is commonly adopted in the multi-view vision tasks. It assumes that all light rays coming from the same focus point in the scene should result in the same photometric values. Since the rays from different directions are recorded separately in a light field, this assumption means that the value of the recorded pixels in different SAI regions corresponding to the same point of the scene should be identical. Nevertheless, it does not always hold, and usually fails when there are occlusions along the path of light ray transmission.

C. Free-Space Intermediate Radiance Inference

We first establish a model to describe the radiance of a light ray impinging on an arbitrary position in the camera plane. Assume that all the light rays are transmitted in free space without any occlusion, as illustrated in Fig. 2a. For simplicity, in the following we will retain only one angular coordinate \( u \), and derive \( L(x, u) \) for a collection of light rays traveling from the position \( u \) to the position \( x \) in the respective planes. Such collection of light rays form the view image from the viewpoint \( u \). We consider the continuous change of viewpoint \( u \) and set its range to be \([0, 1]\). The two boundary viewpoint positions are when \( u = 0 \) and \( u = 1 \).

Given two collections of light rays \( L_0(x) \) and \( L_1(x) \) from two different viewpoints, and a factor \( k \in (0, 1) \), our goal is to infer the intermediate rays \( L_k(x) \), as is demonstrated in Fig. 2b. According to Eq. (2), the coordinates \( u \) and \( x \) are linearly related, which can be depicted more clearly in the EPI pattern. For instance, Fig. 2c shows the EPI pattern of two points at different depth. The red line corresponds to the point \( F_1 \) while the blue line corresponds to \( F_2 \).

The photo-consistency assumption ensures that each line should have a uniform color, i.e. projections of the same point in different views should have the same intensity value. This allows us to approximate the radiance \( L_k(x) \) by shifting the pixels corresponding to \( L_0(x) \) and \( L_1(x) \) properly. Also, in terms of Eq. (5), we have

\[
\tilde{L}_k(x) = (1 - k) \cdot L_k(x) + k \cdot L_k(x)
= (1 - k) \cdot L_0\left(x - \left(1 - \frac{1}{\alpha}\right) k\right) + k \cdot L_1\left(x + \left(1 - \frac{1}{\alpha}\right) (1 - k)\right),
\]

where \( \tilde{L}_k(x) \) denotes the estimation of \( L_k(x) \). Eq. (6) provides a more general expression. Practically, \( L_0(x) \) and \( L_1(x) \) are usually close but not the same due to the noise and illumination. Therefore, the coefficient \( k \) can also be regarded as a weighting factor to balance the information from \( L_0(x) \) and \( L_1(x) \) for the estimate. However, since the depth information of the radiance is unknown, the ratio \( \alpha \) cannot be computed.

To mitigate such a problem, we develop a way to estimate the pixel shift \( \Delta x \) directly. An AM is learned by a dense CNN, which provides the pixel-wise shift information between a pair of view images. The aperture flow is a special case of optical flow, but with values linearly associated with the changes in the angular coordinate \( \Delta u \). Here, we use the aperture flow to obtain the radiance inference. A more detailed demonstration will be presented in Section III-E.

Mathematically, \( A_{k-0}(x) \) and \( A_{k-1}(x) \) denote the AMs from \( L_0(x) \) to \( L_k(x) \) and \( L_1(x) \) to \( L_k(x) \), respectively. Following Eq. (6), we have

\[
\tilde{L}_k(x) = (1 - k)P\left(L_0(x), A_{k-0}(x)\right) + kP\left(L_1(x), A_{k-1}(x)\right),
\]

where \( P(\cdot, \cdot) \) is the pixel-wise propagation operation which translates each pixel of the input image to its correspondence in the target image in terms of the aperture flow between the two images. Given the input image \( L_i(x) (i \in [0, 1]) \) and the AM \( A_{\alpha-\alpha}(x) (\alpha \in [0, 1]) \), the output image of the function \( P(\cdot, \cdot) \) is

\[
P\left(L_i(x), A_{\alpha-\alpha}(x)\right) = L_\alpha(x) = L_i\left(x + A_{\alpha-\alpha}(x)\right).
\]

Eq. (7) implies that the closer the viewpoint position \( k \) is to 0, the more contribution \( L_0(x) \) will make to \( L_k(x) \).
D. Inference with Occlusions

We next discuss the situation where there exist occlusions along the path of the light ray, violating the photo-consistency assumption. Given the direction of light rays recorded by a plenoptic camera, an important property of the light field is that the occluded pixel in the synthesized image always appears in either the leftmost SAI or the rightmost SAI (i.e., the boundary images). This is illustrated in Fig. 3, which presents the EPI pattern of the dashed line region of two objects placed at different distances from the camera. The near object (named “occluder”) with depth $z_2$ partially occludes the further object (named “background”) with distance $z_1$. On the background, the region $CD$ is totally occluded in all views. The regions $BC$ and $DE$ are partially occluded and the other places are totally exposed. In the EPI, each slope line corresponds to a point. For example, the line $A'A''$ with a small slope on EPI corresponds to the point $A$ of the background, while the line $O'O''$ is projected from point $O$ on the occluder with a larger slope on the EPI pattern.

With such a configuration, we now illustrate how to modify the free-space inference model to fit scenarios with occlusions. As the occlusion appears near the boundary of the object, we shade the corresponding EPI in Fig. 3 to highlight these regions with occlusions, i.e. $B'O'O''$ and $R'D''R''$. We first focus on a certain pixel located at point $P$ on the background that is also at the boundary of the occluder when $u = k$. Its EPI pattern is marked with a green line $P'MP''$. This point is occluded when $u \in (0, k)$, but is exposed when $u \in (k, 1)$. On the other hand, another point $Q$ is projected to the line $Q'NQ''$, which is occluded when $u \in (k, 1)$ but is exposed when $u \in (0, k)$.

Nevertheless, both $P$ and $Q$ can be inferred from one of the boundary images, i.e. $L_0(x)$ for $Q$ and $L_1(x)$ for $P$. As a consequence, when there are occlusions along the light ray, the missing pixel information can be inferred from one of the boundary images. Based on this observation, we design another network to estimate the propagation confidence maps, known as PCMs, to assist in the inference of the pixel value. The estimated PCMs, $O_{k-0}(x)$ and $O_{k-1}(x)$, denote the confidence level that a pixel value of $L_k(x)$ can be inferred from $L_0(x)$ and $L_1(x)$, respectively.

As a result, Eq. 7 can be modified as

$$
\bar{L}_k(x) = \Phi^{-1} \circ \left[ (1-k)O_{k-0}(x) \circ P(L_0(x),A_{k-0}(x)) + kO_{k-1}(x) \circ P(L_1(x),A_{k-1}(x)) \right],
$$

where $\Phi = kO_{k-0}(x) + (1-k)O_{k-1}(x)$ denotes a normalization factor. The symbol $\circ$ denotes the Hadamard product between two matrices. The values of the confidence map should fall within $[0, 1]$. Take, as an example, the point $P$ (with coordinate $p$), which is occluded when $u = 0$ and exposed when $u = 1$. We should have $O_{k-0}(P) = 0$, which means that $L_0(P)$ has no contribution to $\bar{L}_k(P)$, and $O_{k-1}(P) = 1$, which means the value is fully contributed by $L_1(P)$. Similarly, for point $Q$ (with coordinate $q$), the value of two confidence maps should satisfy $O_{k-0}(q) = 1$ and $O_{k-1}(q) = 0$. For the points not in the partially occluded region, the value is contributed from both boundary light rays. Consequently, these two maps should satisfy the constraint

$$
O_{k-0}(x) + O_{k-1}(x) = 1.
$$
the auxiliary variable, i.e., has shifted from the original viewpoint (0 auxiliary variable such epipolar property of the aperture flow, we define an related to the changes between the viewpoints. To demonstrate corresponding pixel (radiance), and such a distance is linearly aperture flow. As discussed earlier in Section III-C, each value E. Aperture Flow Estimation

After we obtain the occlusion-aware inference expression in Eq. (9), the next crucial problem is how to estimate the aperture flow. As discussed earlier in Section III-C, each value of the aperture flow map represents the spatial shift of the corresponding pixel (radiance), and such a distance is linearly related to the changes between the viewpoints. To demonstrate such epipolar property of the aperture flow, we define an auxiliary variable \( A(x, u) \), which denotes the distance \( x \) has shifted from the original viewpoint (0) to viewpoint \( u \). Consequently, the aperture flow can also be expressed using the auxiliary variable, i.e., \( A(x, u) = A_{u=0}(x) \).

According to Eq. (2), one can easily obtain the proportional relation between \( A(\cdot, \cdot) \) and coordinate \( u \), as demonstrated in Fig. 4. Therefore, for any \( x \) and \( k \), the aperture flow satisfies

\[
A(x, 0) = 0 \tag{11}
\]

\[
A_{k=0}(x) = A(x, k) - A(x, 0) = -A_{0=k}(x) \tag{12}
\]

\[
A_{k-1}(x) = A(x, k) - A(x, 1) = -A_{1=k}(x) \tag{13}
\]

\[
A_{k=0}(x) = k \cdot A_{k=0}(x) = -k \cdot A_{0=1}(x) \tag{14}
\]

\[
A_{k=1}(x) = (k - 1) \cdot A_{k=0}(x) = (1 - k) \cdot A_{0=1}(x) \tag{15}
\]

\[
A_{1=0}(x) = A(x, 1) - A(x, 0) = A_{1=k}(x) + A_{k=0}(x) \tag{16}
\]

Eqs. (11) to (15) are derived directly from the epipolar property of light field described by Eq. (2) and Eq. (5). In addition, Eq. (16) can be deduced by combining Eq. (12) and Eq. (13). Therefore, the aperture flow \( A_{k=1}(x) \) and \( A_{k=0}(x) \) can be calculated in terms of \( A_{1=0}(x) \) and \( A_{0=1}(x) \) as

\[
A_{k=0}(x) = \frac{k + 1}{2} A_{1=0}(x) + \frac{k - 1}{2} A_{0=1}(x) \tag{17}
\]

\[
A_{k=1}(x) = \frac{k}{2} A_{1=0}(x) + \left( 1 - \frac{k}{2} \right) A_{0=1}(x), \tag{18}
\]

if we assume the summation of the weights in Eq. 17 and Eq. 18 to be 1. The intermediate radiance inference model derived in Sections III-C and III-D can be easily extended to the other angular \( (\nu) \) coordinate.

AFM and PCM are estimated sequentially using a dense network. We adopt the architecture proposed in [45], which is a fully convolutional neural network with dense connections within and across the residual blocks. We use two dense residual blocks, and each includes 4 convolutional layers and 3 ReLU layers. At the end of each block, the learned dense features are concatenated and fed into a convolutional layer with a 1 x 1 kernel to learn more effective features adaptively. In addition, each block allows direct connections from the previous blocks to extract the hierarchical features for flow and confidence map estimation.

The overview of the proposed framework is summarized in Fig. 5, which mainly consists of three stages (excluding the input). In flow estimation, our model adopts a dense network to estimate the AFMs between two boundary images. They are further used to approximate the intermediate flow maps, \( A_{k=1}(\cdot) \) and \( A_{k=0}(\cdot) \). Then, in PCM estimation, both the intermediate flow maps and images are fed into a subsequent dense network to obtain the confidence maps that indicate the pixel-wise contribution from the two input SAIs. In radiance inference, the target image at \( k \) is computed according to Eq. (9). The obtained images are further refined by exploiting the parallax information of all the SAIs with the RefineNet. This module is a simple fully-convolutional framework with 5 alternating convolutional filters [24]. The alternating filter is a type of 4D convolution filter which can fully exploit the parallax information of all the views and reduce large amount of computation. By utilizing the alternating filter, this fully-convolutional module can efficiently handle the input light fields with changeable angular size.

F. Loss Function

All modules and calculations in our framework are differentiable, which enables us to train different parts of our model synchronously. Given the input images \( L_0(x) \) and \( L_1(x) \), and a set of intermediate images \( \{L_{k_i}(x)\}^N_{i=1} \), our loss function consists of four parts. First is a reconstruction loss that directly provides a supervision signal for the synthesized results by calculating the absolute residual error \( \ell_r \) between the intermediate images and the corresponding labels, i.e.,

\[
\ell_r = \frac{1}{N} \sum_{i=1}^{N} \|L_{k_i}(x) - \tilde{L}_{k_i}(x)\|_1. \tag{19}
\]

To reconstruct the high-frequency spatial details [46], [47], we also add the content perceptual loss component \( \ell_c \) given by

\[
\ell_c = \frac{1}{N} \sum_{i=1}^{N} \|\phi(L_{k_i}(x)) - \phi(\tilde{L}_{k_i}(x))\|^2, \tag{20}
\]

where \( \phi(\cdot) \) maps the input images into high-level feature vectors extracted from the ImageNet pretrained VGG16 model (conv4_3 layer) [48].

The third part is a propagation loss \( \ell_p \), which models the quality of the four estimated AFMs as shown in Fig. 5. It includes four terms, and each measures the absolute difference
we use the ADAM optimizer with $\beta_2 = 0.224$ and an Intel Xeon(R)@2.20HGz CPU and a Titan X GPU. The minimization model is trained on a Ubuntu 16.04.4 computer with 9 views. Due to hardware limitation of plenoptic cameras, many corner angular samples are outside the field of view. Therefore, for each scene, we select the center $\times$ views in the experiments, and evaluate the signals from the labels are directly used to reconstruct the image relations or correspondences, which increase the model robustness to different types of light field images.

To demonstrate this, we train our networks using 100 real-world light field images captured with a Lytro Illum camera provided by the Stanford Lytro Archive [50]. Due to hardware limitation of plenoptic cameras, many corner angular samples are outside the field of view. Therefore, for each scene, we select the center $9 \times 9$ views in the experiments, and evaluate the approach using both real-world and synthetic scenes.

Our project is implemented using PyTorch, and the optimization model is trained on a Ubuntu 16.04.4 computer with an Intel Xeon(R)@2.20HGz CPU and a Titan X GPU. The input images are randomly cropped to $224 \times 224$. Moreover, we use the ADAM optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

Finally, we smooth the estimated aperture flow with [49]

$$\ell_s = \|\nabla_x A_{0-i-1}(x)\|_1 + \|\nabla_y A_{0-i-1}(x)\|_1 + \|\nabla_x A_{1-i-0}(x)\|_1 + \|\nabla_y A_{1-i-0}(x)\|_1,$$

where the notation $\nabla$ denotes the differential operation.

Our overall objective is a summation of these loss functions, i.e.,

$$\ell = \lambda_1 \ell_r + \lambda_2 \ell_c + \lambda_3 \ell_p + \lambda_4 \ell_s,$$

where $\lambda_1, \lambda_2, \lambda_3$ and $\lambda_4$ are the respective weights.

IV. EXPERIMENTS

A. Datasets and Implementation Details

The effectiveness of data-driven methods often depends significantly on the quality of the training data. Compared with many view synthesis approaches that directly learn to produce the target image, our model attempts to approximate the image relations or correspondences, which increase the model robustness to different types of light field images. To demonstrate this, we train our networks using 100 real-world light field images captured with a Lytro Illum camera provided by the Stanford Lytro Archive [50]. Due to hardware limitation of plenoptic cameras, many corner angular samples are outside the field of view. Therefore, for each scene, we select the center $9 \times 9$ views in the experiments, and evaluate the approach using both real-world and synthetic scenes.

Our project is implemented using PyTorch, and the optimization model is trained on a Ubuntu 16.04.4 computer with an Intel Xeon(R)@2.20HGz CPU and a Titan X GPU. The input images are randomly cropped to $224 \times 224$. Moreover, we use the ADAM optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

Figure 5: Overview of our proposed framework. Symbol $\oplus$ is the pixel-wise propagation operation, $\odot$ is the Hadamard product, and $\oplus$ refers to the pixel-wise addition.

between the corresponding propagated views and the ground truth, i.e.,

$$\ell_p = \|\mathcal{L}_0(x) - P(\mathcal{L}_1(x), A_{0-i-1}(x))\|_1 + \|\mathcal{L}_1(x) - P(\mathcal{L}_0(x), A_{1-i-0}(x))\|_1 + \frac{1}{N} \sum_{i=1}^{N} \|\mathcal{L}_{k_i}(x) - P(\mathcal{L}_1(x), A_{k_i+1}(x))\|_1 + \frac{1}{N} \sum_{i=1}^{N} \|\mathcal{L}_{k_i}(x) - P(\mathcal{L}_0(x), A_{k_i+0}(x))\|_1.$$ (21)

Finally, we smooth the estimated aperture flow with [49]

$$\ell_s = \|\nabla_x A_{0-i-1}(x)\|_1 + \|\nabla_y A_{0-i-1}(x)\|_1 + \|\nabla_x A_{1-i-0}(x)\|_1 + \|\nabla_y A_{1-i-0}(x)\|_1,$$ (22)

where the notation $\nabla$ denotes the differential operation.

Our overall objective is a summation of these loss functions, i.e.,

$$\ell = \lambda_1 \ell_r + \lambda_2 \ell_c + \lambda_3 \ell_p + \lambda_4 \ell_s,$$ (23)

where $\lambda_1, \lambda_2, \lambda_3$ and $\lambda_4$ are the respective weights.

IV. EXPERIMENTS

A. Datasets and Implementation Details

The effectiveness of data-driven methods often depends significantly on the quality of the training data. Compared with many view synthesis approaches that directly learn to produce the target image, our model attempts to approximate the image relations or correspondences, which increase the model robustness to different types of light field images. To demonstrate this, we train our networks using 100 real-world light field images captured with a Lytro Illum camera provided by the Stanford Lytro Archive [50]. Due to hardware limitation of plenoptic cameras, many corner angular samples are outside the field of view. Therefore, for each scene, we select the center $9 \times 9$ views in the experiments, and evaluate the approach using both real-world and synthetic scenes.

Our project is implemented using PyTorch, and the optimization model is trained on a Ubuntu 16.04.4 computer with an Intel Xeon(R)@2.20HGz CPU and a Titan X GPU. The input images are randomly cropped to $224 \times 224$. Moreover, we use the ADAM optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

The learning rate is set to $10^{-4}$ and reduced by a factor of 0.1 for every 5 epochs.

B. Aperture Flow

One crucial component of our synthesis algorithm is the AFM. Each is estimated in an unsupervised manner, where the signals from the labels are directly used to reconstruct images. Fig. 6 gives an example of the estimated aperture flows between the boundary images ($\mathcal{L}_0(x)$ and $\mathcal{L}_1(x)$) and different intermediate images. We use the center $5 \times 5$ views to visualize the estimated intermediate AFMs between the

Figure 6: Illustration of the aperture flow maps (AFMs) from boundary images (denoted by red square) to the target image (denoted by blue square). Actual flows are visualized using the color coding shown in the subfigure in the bottom-left corner of each scene. Colors represent the directions of vectors and the lighter colors mean the smaller vectors.
images with different intervals. The input images are indicated by red squares while the AFMs are denoted by blue squares. The figure shows the AFMs between different horizontal and vertical views from both real-world and synthetic scenes. Both the estimated AFMs are learned in an unsupervised manner. According to the color-coding panel, the pixels of foreground and background are shifted in opposite directions, and the colors tend to be darker when the interval between the two views is larger.

Another issue we need to tackle is a quantitative assessment of the quality of the computed flows. Since there is no ground truth for AFMs, it is not possible to find a reliable measurement indicator. Nevertheless, the aperture flow relations (listed in Eqs. (11) to (16)) offer an alternative approach to address the problem. One can instead evaluate the linearity of the estimated AFMs in terms of the viewpoint distance between the input SAIs. To do this, we feed a sequence of image pairs with different aperture intervals into the flow estimation network and analyze the output flow maps $A_{1\leftarrow 0}(x)$ and $A_{0\leftarrow 1}(x)$.

The aperture interval controls how different the two input images are. For example, if the two SAIs are with angular coordinates $(0, 0)$ and $(0, 2)$ respectively, their aperture interval is 2. In the experiment, we track several feature points of an input image and record the shift distance of each according to the output flow maps. These feature points are selected using corner detection [51]. Fig. 7 shows the results of 9 randomly picked feature points throughout the image on both the foreground objects and the background wall. The distribution of these points in a synthetic scene (Buddha) are shown in Fig. 7a, while the other plots (Figs. 7b to 7j) show the relationship between the point shift distance and the aperture interval.

To further demonstrate the linearity between these two
Table I: Measurement of the linear dependence between the feature point shift distance and the aperture interval.

| Feature Points | Statistical measurements |
|----------------|--------------------------|
|                | $R^2$ | Adjusted $R^2$ | PCC  |
| P1             | 0.996 | 0.996         | 0.998 |
| P2             | 0.991 | 0.989         | 0.996 |
| P3             | 0.989 | 0.986         | 0.994 |
| P4             | 0.998 | 0.997         | 0.999 |
| P5             | 0.994 | 0.993         | 0.997 |
| P6             | 0.986 | 0.984         | 0.993 |
| P7             | 0.984 | 0.981         | 0.992 |
| P8             | 0.994 | 0.993         | 0.997 |
| P9             | 0.991 | 0.990         | 0.996 |
| Average        | 0.991 | 0.990         | 0.996 |

Figure 8: Illustration of the propagation confidence map (PCM) of different scenes. All PCMs have $k = 0.5$, i.e., they correspond to the image midway between two input images, taken from various datasets.

variables, we also compute their linear regression, and plot the regression line and 95% confidence intervals in the Figs. 7b to 7j. Table I presents some statistics to measure the linear correlation between the shift distance and aperture interval, including the coefficient of determination ($R^2$), adjusted $R^2$, and Pearson correlation coefficient (PCC). The average values of all three measurements are over 0.99, which suggests that the two variables are highly linearly related.

C. Propagation Confidence Map

The AFM aims at estimating the displacement of each pixel in an image, but it can fail in regions with occlusions. To handle this, we introduce the PCM in order to consolidate the information from multiple views to predict a pixel value. Fig. 8 presents the PCMs of images midway between two input images, using both synthetic and real-world scenes. For each PCM, pixels in red denote that at these positions, the source image contributes more; the darker color in red, the higher contribution it makes. The case is opposite for blue. For both real-world and synthetic scenes, the object boundary regions tend to be darker in red or blue. As discussed in Section III-D, the occlusions always appear at the boundary regions which magnify the distinction of the contributions (on the occluded pixels) from two input images. Take the “buddha” scene (the first row in Fig. 8) as an example. In PCMs $O'k←0$ and $O'k←1$, there are different colors near the left and right boundaries of the pillar. This demonstrates that the left input image ($L_0$) contributes more on the left boundary region while the right input image ($L_1$) contributes more on the right boundary region, as interpreted in Fig. 3 (Line $P'MO''$ and Line $R'NQ''$).

D. RefineNet

The RefineNet module is designed to fully exploit the structure information within the light field. The 4D convolution is a straightforward choice but requires a lot of computations. Instead, the pseudo filters and separable filters, which reduce the model complexity, have been proved to be efficient for light field super-resolution [52] and view synthesis [24], [53] problems. Therefore, we adopt the fully-convolutional framework using alternating convolution filters. The module contains two 4D filtering steps and each one is approximated with two alternating filters with kernel size $3 \times 3 \times 1 \times 1$ and $1 \times 1 \times 3 \times 3$, respectively. The learned features are concatenated and then fed into a 4D filter with kernel size $1 \times 1 \times 1 \times 1$ to obtain the refined results. This module further improves the reconstructed images and the effectiveness of this module is illustrated in Section V-C.

V. RESULTS AND DISCUSSIONS

To validate the effectiveness of our proposed framework, we conduct experiments on both synthetic and real-world light fields. We make use of the classical quantitative metrics, namely peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) to access the performance of the algorithms.

A. Comparison with Continuous Synthesis Methods

We first evaluate the performance of our method against the recent techniques for continuous view synthesis. Algorithms of this kind attempts to learn a continuous representation of the plenoptic function from a sparsely-sampled input light field. The comparisons are conducted against three recent learning-based methods, namely Kalantari et al. [16], Soft3D [36] and Shi et al. [54]. Table II presents the quantitative results of different algorithms on the Lytro images for angular $2 \times 2 \rightarrow 8 \times 8$ resolution enhancement. All three compared methods rely on the quality of the estimated depth, which however is difficult to approximate especially based on a sparse set
of views. Our method achieves the best quantitative results compared with all the discussed methods.

Fig. 9 compares the visual performance of different algorithms. As shown, the approach of Kalantari et al. tends to generate artifacts near the object boundary, such as, the leaves and the petal boundary in “Flower 1”. Soft3D [36] gives relatively smooth results near the petal boundary. As for “Cars”, all of the other three methods lose the information of the thin structures. In this picture, three regions (the colored boxes) with thin structures are selected and zoomed in to highlight the differences. Because the depth information of the thin structure is hard to estimate, this is likely the cause of the missing or aliasing of these structures in the reconstructed images. In comparison, our depth-free method produces consistently better results for thin objects and the boundary regions. In both examples, our results are closer to the ground truth. In addition, our approach takes about 0.23s per synthesized view on average to reconstruct a $8 \times 8$ light field from its $2 \times 2$ views at a spatial resolution of $376 \times 541$, which is nearly 40 times faster than Kalantari et al. (about 9s per view) and 3.7 times faster than FPFR [54] (about 0.85s per view).

### B. Comparison with Other Learning-Based Methods

Next, we compare the performance of our method against several recent learning-based methods, including Wu et al. [55] and FPFR [54]. Table II shows the quantitative comparisons (PSNR/SSIM) of the proposed approach with continuous view synthesis algorithms for angular super-resolution $2 \times 2 \rightarrow 8 \times 8$. The input light fields for each algorithm are sampled at four corners.

#### Table II: Quantitative comparisons (PSNR/SSIM) of the proposed approach with continuous view synthesis algorithms for angular super-resolution $2 \times 2 \rightarrow 8 \times 8$.

| Scenes    | Methods          | Kalantari et al. [16] | Soft3D [36] | FPFR [54] | Ours  |
|-----------|------------------|------------------------|-------------|-----------|-------|
| Flowers1  | PSNR(db):        | 33.32 / 0.960          | 32.84 / 0.960 | 34.51     | 34.64 |
|           | SSIM:            | 0.960                  | 0.956       | 0.972     | 0.972 |
| Flowers2  | PSNR(db):        | 31.94 / 0.960          | 32.74 / 0.961 | 34.20     | 34.03 |
|           | SSIM:            | 0.960                  | 0.961       | 0.972     | 0.965 |
| Cars      | PSNR(db):        | 31.65 / 0.966          | 30.93 / 0.963 | 32.28     | 33.02 |
|           | SSIM:            | 0.966                  | 0.970       | 0.967     | 0.973 |
| Seahorse  | PSNR(db):        | 31.87 / 0.970          | 31.22 / 0.966 | 34.36     | 33.61 |
|           | SSIM:            | 0.970                  | 0.961       | 0.970     | 0.972 |
| StoneLion | PSNR(db):        | 40.57 / 0.979          | 38.53 / 0.973 | 40.02     | 40.84 |
|           | SSIM:            | 0.979                  | 0.973       | 0.975     | 0.981 |
| Leaves1   | PSNR(db):        | 35.84 / 0.973          | 31.74 / 0.947 | 34.76     | 36.95 |
|           | SSIM:            | 0.973                  | 0.970       | 0.966     | 0.978 |
| Leaves2   | PSNR(db):        | 34.17 / 0.963          | 32.47 / 0.965 | 32.86     | 34.28 |
|           | SSIM:            | 0.963                  | 0.965       | 0.957     | 0.966 |
| Average   | PSNR(db):        | 34.19 / 0.967          | 32.92 / 0.964 | 34.71     | 35.33 |
|           | SSIM:            | 0.967                  | 0.969       | 0.969     | 0.972 |

*Note that the methods Soft3D [36] and FPFR [54] did not release their codes. For FPFR, we used the code provided by the authors. For Soft3D, we used the code reimplemented by the authors of the work [55].
Figure 10: Visual comparisons of different algorithms on the (5, 5) synthesized view for the task $2 \times 2 \rightarrow 8 \times 8$ on both real-world and synthetic scenes. We select two regions (red and blue boxes) to compare the spatial results of different algorithms. For each reconstructed light field, the EPIs at the position highlighted by the cyan line are also visualized. We zoom in one selected region of each EPI for better comparison.
Table III: Quantitative comparisons (PSNR/SSIM) of the proposed approach with learning-based methods.

| Scenes      | Data Type | Kalantari et al. [16] | Wu et al. [22] | Yeung et al. [24] | HDDRNet [12] | Ours     |
|-------------|-----------|------------------------|----------------|-------------------|-------------|----------|
| Bedroom     | Synthetic | 34.87 / 0.914          | 32.52 / 0.890  | 34.26 / 0.895     | 34.29 / 0.892 | 34.48 / 0.906   |
| Cotton      | Synthetic | 41.98 / 0.964          | 38.49 / 0.944  | 41.47 / 0.956     | 42.32 / 0.961 | 43.63 / 0.973   |
| Dino        | Synthetic | 36.88 / 0.951          | 34.21 / 0.904  | 40.38 / 0.951     | 40.34 / 0.952 | 40.78 / 0.957   |
| Origami     | Synthetic | 30.91 / 0.903          | 28.26 / 0.898  | 31.54 / 0.912     | 32.18 / 0.912 | 33.59 / 0.919   |
| Average     | —         | 36.16 / 0.933          | 33.37 / 0.909  | 36.91 / 0.929     | 37.28 / 0.929 | 38.12 / 0.939   |

Table III and Fig. 10 present the quantitative and visual results, respectively. Wu et al. [22] design a “blur-restoration-deblur” pipeline to overcome angular aliasing, but the algorithm requires at least three views to generate acceptable results. For $2 \times 2 \rightarrow 8 \times 8$ task, because only two views are available for inputs, the insufficient angular information leads to aliasing effects in their reconstructed EPIs, especially in the EPIs of the two Flower scenes in Fig. 10. Yeung et al. [24] and HDDRNet [12] achieve the state-of-the-art performance. They both adopt the 4D convolution to fully exploit the spatio-angular information, which result in their superior results.

However, for the synthetic scenes, their results tend to have more errors near the object boundaries. Another drawback for these end-to-end methods is that they can only generate the light fields with fixed angular dimensions. This is due to the reshape operation (between the channel and angular dimensions) when upsampling the angular resolution. As a result, when the number of input (or output) views varies, both have to train a new model to fit for such changes. In comparison, we increase the number of views by utilizing the continuous representation among the views and adopt the fully-convolutional framework as the refinement module. Such design enables our approach to handle the input light field with various angular dimension, and the resulting approach achieves the best quantitative results and can produce images close to the ground truth.

C. Ablation Study

We conduct two ablation studies as follows. First, to demonstrate the effectiveness of the RefineNet, we quantitatively compare the light fields generated by our approach with and without this module. Table IV shows the quantitative results on both real-world and synthetic scenes. We use 20 real-world scenes randomly selected from the “Flower” dataset [56]. Each light field contains a flower in the foreground, which has a relatively larger baseline than the background. The synthetic scenes are randomly selected from HCI [57] and we use 10 light fields. As shown, the RefineNet module improves the reconstruction results by over 1.3dB on PSNR, and it also shows improvement in terms of the SSIM. Fig. 11 illustrates the visual improvements by adding this module. The reconstructed errors are further reduced and some spatial details can be recovered better.

Second, our ablation studies also evaluate the functions of different loss terms. The reconstruction loss term $\ell_r$ is essential for the network training, and therefore we conduct the ablation studies to evaluate the effectiveness of the other three loss terms. Table V shows the quantitative results of the proposed network (without the RefineNet module) trained with different loss terms on “Flower” dataset. As shown, the full model achieves the best performance.
Table V: Ablation framework on the Flowers dataset [56].

| Settings                  | PSNR  | SSIM   |
|---------------------------|-------|--------|
| without perceptual loss   | 35.32 | 0.943  |
| without propagation loss  | 35.29 | 0.920  |
| without smoothness loss   | 35.33 | 0.944  |
| Full loss                 | 35.33 | 0.945  |

Fig. 12 visualizes the $(9, 9)^{th}$ SAI of the reconstructed light field images. As shown, our method can produce more realistic results with relatively sharp textures, and construct the EPIs with clear linear structure patterns on both synthetic and real-world scenes.

VI. Conclusions

In this paper, we propose a depth-free algorithm for the reconstruction of arbitrary intermediate views of the light fields. To efficiently describe the parallax between any two given views, we define the AFM based on the epipolar property of the light field. By incorporating also the PCM, our method can efficiently address the occlusions near the object boundaries. Both AFM and PCM are approximated using a dense network. In addition, we further adopt a 4D CNN in the refinement stage to improve the quality of synthesized images. Experimental results have demonstrated that the proposed model achieves state-of-the-art performance for both synthetic and real-world light fields.

VII. Acknowledgments

This work is supported in part by the Research Grants Council of Hong Kong (GRF 17203217, 17201818, 17200019) and the University of Hong Kong (104005009, 104005438). We thank Dr. Xiaoran Jiang, the author of [55], for helping us with the experiments of Soft3D. We also thank the strong support from Dr. Jinglei Shi, the author of [54], and his group.

REFERENCES

[1] E. Y. Lam, “Computational photography with plenoptic camera and light field capture: tutorial,” Journal of the Optical Society of America A, vol. 32, no. 11, pp. 2021–2032, November 2015.

[2] R. Ng, “Digital light field photography,” in Ph.D. Dissertation, Dept. Computer Science. Stanford University California, CA, USA, 2006.

[3] N. Chen, C. Zuo, E. Y. Lam, and B. Lee, “3D imaging based on depth measurement technologies,” Sensors, vol. 18, no. 11, pp. 3711(1–38), November 2018.

[4] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan, “Light field photography with a hand-held plenoptic camera,” Computer Science Technical Report, vol. 2, no. 11, pp. 1–11, 2005.

[5] X. Sun, Z. Xu, N. Meng, E. Y. Lam, and H. K.-H. So, “Data-driven light field depth estimation using deep convolutional neural networks,” in IEEE International Joint Conference on Neural Networks, November 2016, pp. 367–374.

[6] H. Schilling, M. Diebold, C. Rother, and B. Jähne, “Trust your model: Light field depth estimation with inline occlusion handling,” in IEEE Conference on Computer Vision and Pattern Recognition, December 2018, pp. 4530–4538.

[7] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum, “Plenoptic sampling,” in ACM Conference on Computer Graphics and Interactive Techniques, July 2000, pp. 307–318.

[8] N. Meng, X. Sun, H. K.-H. So, and E. Y. Lam, “Computational light field generation using deep nonparametric bayesian learning,” IEEE Access, vol. 7, pp. 24990–25000, February 2019.

[9] N. Viganò, P. M. Gil, C. Herzog, O. de la Rocheforoud, R. van Liere, and K. J. Batenburg, “Advanced light-field refocusing through tomographic modeling of the photographed scene,” Optics Express, vol. 27, no. 6, pp. 7834–7856, 2019.

[10] N. Meng, T. Zeng, and E. Y. Lam, “Spatial and angular reconstruction of light field based on deep generative networks,” in IEEE International Conference on Image Processing, August 2019, pp. 4659–4663.

[11] Y. Wang, L. Wang, J. Yang, W. An, J. Yu, and Y. Guo, “Spatial-angular interaction for light field image super-resolution,” European Conference on Computer Vision, September 2020.
