Neural Light Field 3D Printing

QUAN ZHENG, Max Planck Institute for Informatics, Germany
VAHID BABAEI, Max Planck Institute for Informatics, Germany
GORDON WETZSTEIN, Stanford University, USA
HANS-PETER SEIDEL, Max Planck Institute for Informatics, Germany
MATTHIAS ZWICKER, University of Maryland, College Park, USA
GURPRIT SINGH, Max Planck Institute for Informatics, Germany

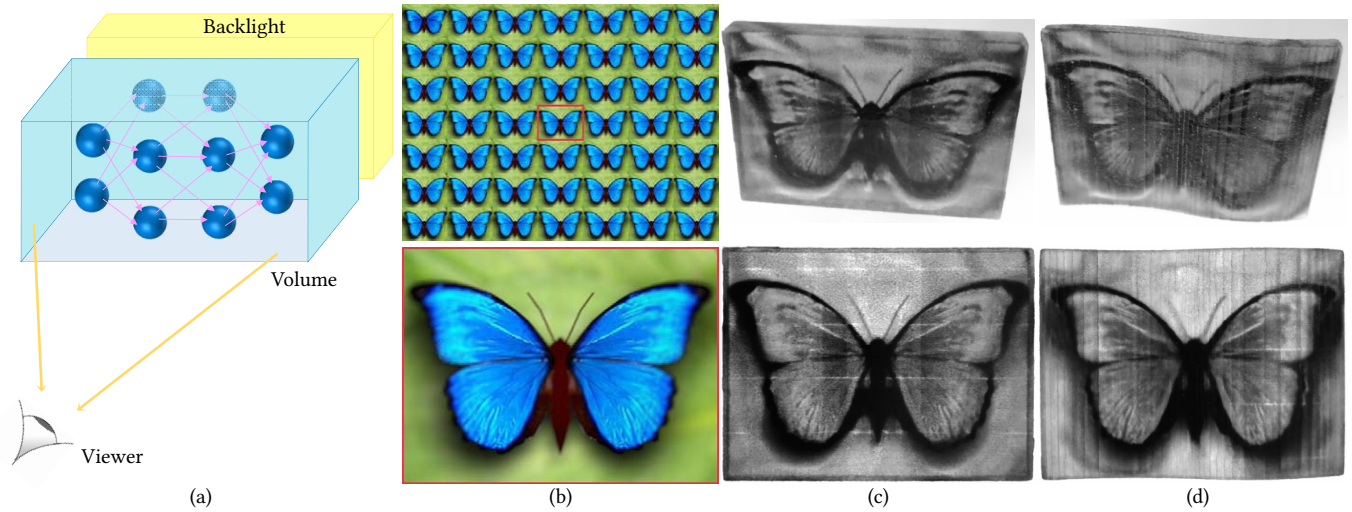


Fig. 1. Our approach encodes input light field imagery (b) as a continuous representation of the volumetric attenuator using neural networks (a). Top row shows the fabricated prototypes using our approach that optimizes both planar (c) and non-planar displays (d). Bottom row compares the central view (red inset) of the input light field (b) with front-view photographs of printed prototypes (c, d).

Modern 3D printers are capable of printing large-size light-field displays at high-resolutions. However, optimizing such displays in full 3D volume for a given light-field imagery is still a challenging task. Existing light field displays optimize over relatively small resolutions using a few co-planar layers in a 2.5D fashion to keep the problem tractable. In this paper, we propose a novel end-to-end optimization approach that encodes input light field imagery as a continuous-space implicit representation in a neural network. This allows fabricating high-resolution, attenuation-based volumetric displays that exhibit the target light fields. In addition, we incorporate the physical constraints of the material to the optimization such that the result can be printed in practice. Our simulation experiments demonstrate that our approach brings significant visual quality improvement compared to the multilayer and uniform grid-based approaches. We validate our simulations with fabricated prototypes and demonstrate that our pipeline is flexible enough to allow fabrications of both planar and non-planar displays.

CCS Concepts: • **Applied computing** → **Computer-aided manufacturing**; • **Computing methodologies** → *Neural networks*; Volumetric models.

Additional Key Words and Phrases: Volumetric display, light field, neural networks, 3D printing, computational fabrication

ACM Reference Format:
Quan Zheng, Vahid Babaei, Gordon Wetzstein, Hans-Peter Seidel, Matthias Zwicker, and Gurprit Singh. 2020. Neural Light Field 3D Printing . *ACM Trans. Graph.* 39, 6, Article 207 (December 2020), 12 pages. https://doi.org/10.1145/3414685.3417879

Authors' addresses: Quan Zheng, Max Planck Institute for Informatics, Saarbrücken, Germany, qzheng@mpi-inf.mpg.de; Vahid Babaei, Max Planck Institute for Informatics, Saarbrücken, Germany; Gordon Wetzstein, Stanford University, USA; Hans-Peter Seidel, Max Planck Institute for Informatics, Saarbrücken, Germany; Matthias Zwicker, University of Maryland, College Park, USA; Gurprit Singh, Max Planck Institute for Informatics, Saarbrücken, Germany.

0730-0301/2020/12-ART207 $15.00
https://doi.org/10.1145/3414685.3417879

1 INTRODUCTION

Light fields are an important visual medium capable of capturing the visual information of a scene in a compact manner. The generation of light fields has been made possible by either synthetic rendering approaches [Gortler et al. 1996; Levoy and Hanrahan 1996] or directly capturing with light field cameras [Marwah et al. 2013; Ng et al. 2005]. Up to now, light field data are still commonly appreciated via arrays of 2D images or bundles of video frames.

Modern 3D printers have made it possible to appreciate the experience of light field on fabricated displays at high-resolution. However, optimizing such light-field displays in a full-volumetric setting at high-resolution is still a challenging task. Previous methods [Wetzstein et al. 2011, 2012] have addressed this problem in a 2.5D setting by discretizing the 3D optimization-space as multiple 2D discrete layers. These approaches may provide an *optimal* solution using linear solvers, but they are highly restrictive in display size and resolution. To address this issue, we propose an end-to-end optimization using a neural network which encodes the target light field as an implicit continuous volume.

For a given light field, we optimize for the volumetric attenuation of light by learning to predict the absorption coefficients in a continuous 3D space. This training is facilitated by a volumetric ray-casting process which is fully differentiable. Batches of rays are traced through the implicit volume and their radiance is attenuated by the medium. Before the printing, a voxel-query can be performed from this learnt implicit representation of absorption coefficients according to the target resolution. The resulting 3D grid is then halftoned [Ostromoukhov 2001] and sent to the 3D printer.

We validate our approach by demonstrating the improvements over existing mutlilayer approaches with respect to visual quality, numerical errors, and scalability. We also develop a grid-based approach, that optimizes on a voxel grid with a fixed resolution, as the baseline for comparisons. Unlike our neural network-based optimization, the grid-based approach quickly exceeds the available memory with increasing display size and resolution. We demonstrate the practicality of our approach by fabricating a prototype (Figure 1) using an inkjet 3D printer. We go beyond established current trends (planar displays) and, for the first time to the best of our knowledge, showcase non-planar light-field displays. To summarize, our neural light field printing approach makes the following contributions:

(1) We propose an end-to-end 3D fabrication pipeline that optimizes implicit volumetric representation using a neural network.
(2) The resultant neural representation is memory efficient and can be deployed to fabricate displays at different scale and resolutions.
(3) Our design benefits from an existing neural network architecture [Mildenhall et al. 2020] and naturally extends it for fabricating both planar and non-planar volumetric light-field displays.

2 RELATED WORK

Multilayer Displays. As uncontrolled scattering of light within a physical volume will undermine 3D effects, multilayer autostereoscopic displays are generally designed based on attenuation within volumes. Wetzstein et al. [2011] demonstrated a successful prototype of a light field display by stacking multiple printed 2D attenuation layers. They computed multiple discrete attenuation layers with well-established tomographic techniques. Recently, tomography has also been extended to design large scale volumetric projectors [Jo et al. 2019] for use cases like theaters and classrooms.

With the encouraging progress of LCD technologies, mutlilayer dynamic LCD displays have been intensively studied. Following the concept of parallax barriers, Lanman et al. [2010] designed a dual-layer LCD display by stacking a pair of LCD panels and adapting each panel to multi-view content. Instead of using a layered attenuation design, the polarization field display [Lanman et al. 2011] constructed multiple polarized rotating liquid crystal layers to emit dynamic light fields. Wetzstein et al. [2012] further proposed tensor displays and generalized the design problem of multilayer LCD displays. These displays are designed according to predefined resolutions and they are limited to planar surfaces. In contrast, our display supports resolution-free designs and we can fabricate light fields on non-planar surface.

The multilayered fabrication fashion has also been adopted in other different fields. Holroyd et al. [2011] fabricated mutli-layer acrylic model to reproduce the appearance of target 3D models by warping them onto multiple layers. Along the same line, layered attenutator was also exploited to generate shadow projections of images under prescribed incident light [Baran et al. 2012].

Volumetric Displays. Blundell et al. [1994] proposed an early volumetric display called a cathode ray sphere. They also defined volumetric displays as devices that emit light fields based on emission, absorption, or scattering of light within a physical volume [Blundell et al. 1993]. Favalora [2005] summarised several common approaches to achieve volumetric displays. Swept volume displays use rotating LCD panels [Maeda et al. 2003] or parallax barriers [Yendo et al. 2005] to deflect light rays. Based on this principle, Cossairt et al. [2007] designed a horizontal parallax-only swept volume display that can handle view-dependent occlusions. Jones et al. [2007] further introduced a high-speed video projector and a spinning holographic diffuser to achieve both horizontal and vertical parallax. Projection media other than LCD or LED were also explored. Nayar and Anand [2007] devised a display by making use of a cloud of passive optical scatters in a glass cube, and water drops [Barnum et al. 2010] were also used as voxels for constructing a multilayer display.

For attenuation-based volumetric displays, an analysis [Gotoda 2010; Wetzstein et al. 2011] has been developed to convert continuous volumes to separated masks, where masks are optimized by solving a linear equation system. Using a similar volumetric approach, Mitra and Pauly [2009] proposed to use a shadow volume to generate shadow art. Loukianitsa and Putilin [2002] presented an early design of a 3D display by stacking two-layer LCD panels. They used a neural network in their design to compute pixel values to be shown on each 2D LCD panel. In contrast, we directly operate on continuous 3D volumes instead of a few separated layers. Our neural network learns the continuous 3D volume representation from input light fields. This allows us to fabricate the high-resolution light field using 3D printers.

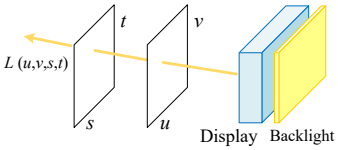
Neural Networks & Light Fields. Deep neural networks [Bengio et al. 2012; Krizhevsky et al. 2012; LeCun et al. 2015] have shown remarkable achievements in encoding highly complex functions in a non-linear fashion. Recent works [Lombardi et al. 2019; Mildenhall et al. 2019; Sitzmann et al. 2019] used neural networks to learn implicit representations from imagery without requiring any explicit geometry. Simple MLP-based architectures have also shown remarkable improvements in both view synthesis [Mildenhall et al. 2020] and in encoding complex light interactions [Kallweit et al.

2017]. In this paper, we employ a neural network to encode a full 3D representation of light field imagery in a scene-dependent setup. Our model builds upon the recent MLP-based architecture [Mildenhall et al. 2020] that extends a vanilla MLP with *positional encoding* to encode the underlying light field imagery for view synthesis.

Appearance 3D Printing. Recent high-resolution multi-color inkjet 3D printers [Stratasys 2020] have enabled creating photorealistic 3D prints. Brunton et al. [2015] proposed an algorithm for managing the color reproduction workflow. Babaei et al. [2017] introduced a color reproduction algorithm, called *contoning*, for 3D printing without the need for halftoning. Shi et al. [2018] extended the contoning framework to reproduce the spectra of oil paintings using an inkjet 3D printer. In response to the significant volume scattering of some 3D printing inks, Elek et al. [2017] proposed a color reproduction technique that preserves the texture by simulating the subsurface cross-talk between neighboring voxels. Sumin et al. [2019] extended this *scattering-aware* approach to arbitrary 3D shapes with potentially thin geometric features. In the context of light fields, Tompkin et al. [2013] and Saberpour et al. [2020] used an inkjet 3D printer to build all-in-one lenticular displays on flat and curved surfaces, respectively. Similar to their device, we use a two-material (black and clear) inkjet 3D printer to fabricate attenuation based volumetric displays.

3 BACKGROUND

We aim to fabricate static 4D light fields. To understand the process of light field formation through a volumetric attenuator, we use a standard two-plane light field parameterization [Levoy and Hanrahan 1996]. As shown in the side figure, light rays from a uniform backlight traverse a volumetric attenuator and form a light field. This process is commonly referred to as the *forward* imaging model. Let the initial radiance carried by a ray(u, v, s, t) from the backlight be $L_i(u, v, s, t)$. Our notation implies scalar radiance (gray scale), but the formulation can trivially be applied to any type of spectral color sampling by replacing scalar radiance with a vector. The initial ray's radiance is attenuated after it traverses the volume by the material in the blue box. Given the light field is temporally static, the attenuated radiance can be written as



$$L_o(u, v, s, t) = L_i(u, v, s, t) \cdot T(u, v, s, t), \tag{1}$$

where $T(u, v, s, t)$ gives the *transmittance* of the medium. It aggregates the comprehensive interactions between light rays and the volume, such as emission, scattering and absorption. We assume our printing materials have no self illumination (e.g., fluorescence) and their scattering is negligible. We further assume the volumetric attenuator has a continuously varying medium density. Hence, we approximate the modulation in the light according to the Beer-Lambert law,

$$T(u, v, s, t) = e^{-\int_0^z \mu(z)dz}, \tag{2}$$

where $\mu(\cdot)$ is the absorption coefficient and z represents the distance travelled along the ray. With Equation 1, we can compute the 4D light field emitted by the volumetric attenuator.

4 NEURAL VOLUMETRIC DISPLAY

For the forward imaging model in Section 3, the goal is to compute the light field emitted by the volumetric attenuator. By contrast, we only have the target light field and we aim to find the volumetric attenuator for printing such that the final result provides us with a light field that is close to the target. This requires an *inverse* imaging model. In this section, we firstly describe the formulation of the inverse problem and then describe our approach to learn a neural representation of the desired volumetric attenuator.

4.1 Formulation

We consider an outgoing ray $r = (u, v, s, t)$ from the volumetric display and denote its radiance as $L_o(r)$. Based on Equation 1, we can relate it to the incoming radiance $L_i(r)$ from the backlight as $L_o(r) = L_i(r) \cdot T(r)$. By plugging Equation 2 and taking the logarithm, we obtain the accumulated material absorbance along r as

$$\alpha(r) = -\ln \frac{L_o(r)}{L_i(r)} = \int_{p_i(r)}^{p_o(r)} \mu(z)dz. \tag{3}$$

Here z denotes a location along the ray r, and $p_i(r)$ and $p_o(r)$ are the points where r enters and exits the volume, respectively. Denoting the target light field as $L_{gt}(r)$, Equation 3 immediately determines the ground truth absorbance $\alpha_{gt}(r) = -\ln(L_{gt}(r)/L_i(r))$.

To determine the unknown volumetric absorption function that will reproduce $L_{gt}(r)$, let us assume that it is parameterized by a set of variables $\Theta = \{\theta_i\}$. We write the absorption coefficient μ in Equation 3 as μ_Θ. In addition, we write the accumulated absorbance corresponding to μ_Θ as $\alpha_\Theta(\mathrm{r})$ along r. Now we can express the problem of finding μ_Θ as the following optimization problem over the unknown parameters Θ,

$$\arg\min_{\Theta} \sum_{r \in F} \mathcal{D}\left(\alpha_{gt}(r), \alpha_\Theta(r)\right), \tag{4}$$

where r is a ray sampled from the target light field F, and $\mathcal{D}$ is a loss function.

4.2 Grid-based Volumetric Representation

Our inkjet 3D printers are *raster* devices, i.e., they print a multi-material grid of physical voxels. This motivates us to discretize the volume of the desired volumetric attenuator as a grid of virtual voxels. In this case the optimization variables Θ are the voxel values, and we can directly determine them by solving Equation 4.

We construct a virtual voxel grid which has the same resolution as the physical version for 3D printing. Each voxel has a size equal to the physical voxel and is associated with an absorption coefficient, which corresponds to one optimization variable $\theta_i \in \Theta$. Formally, the continuous function μ_Θ is given by the continuous interpolation of the voxel values. In our implementation we use nearest neighbor interpolation, which could easily be replaced with other, higher order interpolation schemes.

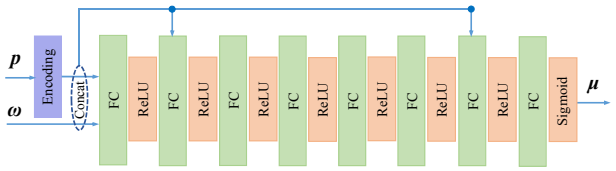


Fig. 2. Architecture of our neural network. FC stands for fully connected layers and ReLU means the rectified-linear-unit activation.

To print 3D models in high precision, a prohibitive number of small voxels are needed. Since the grid-based approach has an absorption coefficient for each voxel, it will have to optimize a high number of variables and, therefore, requires a large amount of data to optimize with. This motivates us to seek another representation that is not limited by the final printing precision during the optimization stage.

4.3 Neural Network-based Volumetric Representation

Neural networks are known to be general function approximators for many problems. In theory, neural networks with adequate capacity can approximate any non-linear function. Therefore, we consider representing the function μ_Θ as a fully-connected neural network, also known as a multilayer perceptron (MLP). Here, the optimization variables Θ represent the trainable parameters of the neural network. Since μ in Equation 3 is a function of a position $p = (x, y, z)$ defined in Euclidean space, we consider using a position vector as the input to our neural network. The output of our neural network is the absorption coefficient μ.

Our MLP architecture is shown in Figure 2. Each hidden layer has 512 neurons. Each neuron accepts the outputs of the preceding layer, conducts a multiply-add computation, and applies a nonlinear ReLU activation before sending an output to next layer. Initially, we input coordinates of position p directly to the neural network after scaling them to $[-1, 1]$. The MLP merely using original coordinates as input, however, has difficulty in learning lighting and geometric details from the target light field. In a recent work, Mildenhall et al. [2020] show that *positional encoding* [Rahaman et al. 2018] helps learn high frequency details for radiance fields. It transforms each coordinate of p to a higher dimensional space using sinusoidal functions: $p \rightarrow (p, \sin 2^0 p, \cos 2^0 p, \ldots, \sin 2^{M-1} p, \cos 2^{M-1} p)$. A related theoretical analysis of *positional encoding*, as one of the Fourier feature mappings, can be found in Tancik et al. [2020]. We leverage *positional encoding* to get an encoded vector of the position. In addition, we append the original three-dimensional direction ω to the encoded vector.

To better transport the input data to the later layers of the network, we use skip connections [He et al. 2016] which concatenate the encoded input to the following two layers. We experimentally find this makes our deep neural network training more efficient and slightly improves the quality.

4.4 Attenuation-based Volumetric Ray Casting

To solve the minimization problem formulated in Equation 4, we extract ray samples from a target light field F and cast rays into the volume which is now represented as a neural network. Our ray casting process is different from the standard volume rendering technique [Drebin et al. 1988], which is adopted in recent research work [Lombardi et al. 2019; Mildenhall et al. 2020]. Traditional volume rendering computes an additive blending of volumes of color and opacity, whereas our rendering process simulates the physical process of light field formation by attenuating light in the volumetric display.

As defined in Equation 3, we compute $\alpha_\Theta(r)$ for each ray. Note that the absorbance depends on the parameters Θ of our absorption function. We use the parametric form to express a ray as $p = r_o + zr_\omega$, where r_o denotes the origin, r_ω is the direction and z is the parametric distance along ray r. Hence, α_Θ can be rewritten as

$$\alpha_\Theta = -\int_{z_i}^{z_o} \mu_\Theta(r_o + zr_\omega)\mathrm{d}z. \tag{5}$$

Here, z_i and z_o are parametric distances from the origin of r to the intersection points on the near surface and far surface of the volume, respectively.

Note that our optimization over the volume of a display can accommodate any display geometry. This benefit allows us to optimize and print light fields onto displays with non-planar geometries. For planar surfaces, we can compute parametric distances, z_i and z_o, by ray-plane intersections. For non-planar surfaces, we find them by solving two ray-surface intersections. Note that we ignore reflections of rays on surfaces as reflected rays do not enter the volume and we simulate refraction of rays when entering and exiting a volume with the index of refraction being 1.5.

Our neural network learns a density field of absorption coefficients. Given a trained neural network, μ_Θ of every location in the volume is readily available. To solve the integral of μ_Θ, we consider ray segments that are located in the volume. By using the parametric form of a ray, the segment is defined by $[z_i, z_o]$. As configured in the parallel beam tomography approach [Wetzstein et al. 2011], we conduct parallel volumetric ray casting into the volume. For each ray, we firstly find the range that is within the volume and segment it into S bins with equal length. We then use stratified sampling to find intermediate S samples from these bins. With these samples, we can compute the integral with a numerical integration approach, the rectangle rule. For each interval delimited by these samples, we use the right sample of each interval and compute α_Θ as

$$\alpha_\Theta = -\sum_{j=0}^{S+1} \mu_\Theta(r_o + z_j r_\omega) \left\| z_{j+1} - z_j \right\|_2, \tag{6}$$

where z_j denotes sample locations generated using stratified sampling in the interval $[z_i, z_o]$ and $\|\cdot\|_2$ calculates the $\mathcal{L}^2$ distance along a ray between the two points defined by adjacent two samples of parametric distance. This is the place where the actual thickness of our final volumetric display comes in. Larger thickness allows longer average traversing distance for rays entering the volume.

4.5 End-to-end Optimization

Our volumetric ray casting computes α_Θ (Equation 6) as a linear combination of weighted samples of μ_Θ, thus the derivatives of μ_Θ

with respect to Θ can be easily computed. We implement the volumetric ray casting process along with the neural network in a deep learning library that supports automatic differentiation. This enables an end-to-end training of our neural network using a stochastic gradient descent (SGD) based optimizer.

During optimization, we sample rays from the target light field to solve the minimization problem (Equation 4). For the difference function $\mathcal{D}$, we use an $\mathcal{L}^2$ loss along with a regularization in training,

$$\mathcal{D}\left(\alpha_{gt}(r), \alpha_{\Theta}(r)\right) = \frac{1}{N}\sum_{r\in F}\left(\alpha_{gt}(r) - \alpha_{\Theta}(r)\right)^2 + \lambda\left\|\Theta\right\|_2^2, \quad (7)$$

where N is the number of rays, r is a ray sampled from the target light field F, $\alpha_{\Theta}(r)$ is the predicted absorbance computed via volume ray casting, and $\alpha_{gt}(r)$ is the ground truth absorbance. Hyperparameter λ is assigned with 0.0001. We also tried $\mathcal{L}^1$ loss function for training, but $\mathcal{L}^2$ performs better in terms of overall visual quality across different test scenes.

5 FABRICATION

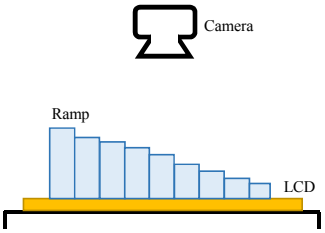
Our neural network learns continuous, spatially-varying volumetric absorptions that generate desired light fields. For fabricating the light field, we first sample the learned density of absorption coefficients according to the target print resolution. In practice, we regularly sample nine directions for the inference stage and find a mean of the output density. We then halftone the continuous density values slice-by-slice using an error diffusion algorithm [Ostromoukhov 2001]. For printing, we utilize a two-material (black and clear) inkjet 3D printer (Objet260 Connex) to print the display. The highest precision of the printer is 84 µm × 84 µm laterally, with 30 µm slice thickness.

Since our printer supports printing with a combination of a black and a clear material, we print our prototypes in gray. With the recent introduction of colored, transparent materials to inkjet 3D printing, e.g., Vero Vivid from Stratasys, our approach could be extended to color printing in a straightforward manner.

Unfortunately, the printer software only supports meshes with a limited number of triangles (about 10 millions). This limits the number of voxels we can use for printing and therefore the size and/or resolution of the display. We show printed prototypes with resolutions $512 \times 384 \times 24$ (Figure 1). Simulated results with high resolution grids spanning the full thickness range of a display can be found in the supplemental.

5.1 Material Calibration

In our two-material setup, we use a clear material, *VeroClear*, to fabricate transparent parts and a black material, *TangoBlack*, for the dark parts. We assume the scattering is insignificant and the clear material has an absorption coefficient of zero. For measuring the absorption coefficient of the black material, we take a dictionary-based approach [Gkioulekas et al. 2013] where we compare a printed model to many of its rendered counterparts with varying absorption coefficients. We search for the absorption coefficient that leads to the best match to the print.



We start by printing a ramp-shaped model in *TangoBlack*. This model is composed of 15 rectangular columns with varying heights. The illumination, provided by a LCD, comes from behind the ramp. Discarding the color information, we take a single channel HDR photo of the illuminated print. By dividing the photo of the print by the background illumination image, we obtain a map of transmittance. After averaging, we further obtain a single transmittance value for each rectangle leading to a transmission profile for the print. For creating the rendered dictionary, we handcraft a virtual scene with the same setup as in reality, and calculate a transmission profile per scene. We regularly sample the absorption coefficient $\mu \in [0, 10]\,(\mathrm{mm}^{-1})$ with a step size 0.05 and render an image for the scene per step. Finally, we find the nearest neighbor to the transmission profile of our print from within the rendered dictionary using an acceleration data structure. With a similar setup, Elek et al. [2017] measures scattering parameters of material samples, whereas our acquisition focuses on the absorption coefficients.

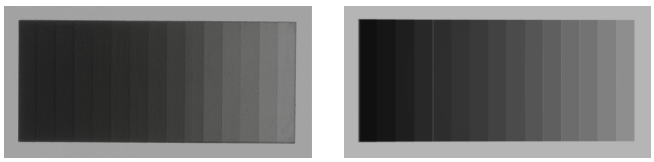


Photo Simulation

Fig. 3. Captured gray-scale photo of the printed ramp model (left) and simulated result (right) of the corresponding virtual scene.

6 EXPERIMENTS

6.1 Training Settings

We implement our neural network based on the Tensorflow library [Abadi et al. 2015] and run it on a Nvidia RTX Titan GPU. Specifically, we use the Adam [Kingma and Ba 2014] optimizer with a learning rate 1×10^{-4} and keep other parameters as their default values. The learning rate is exponentially decayed every 1000 epochs with a rate 0.8 and then is kept constant after it drops below 1×10^{-5}. Before training, we initialize kernel weights with Glorot uniform initialization [2010] and all kernel biases with zeros. When training the neural network, the mini-batch size is set to 1200 and a batch of data are randomly shuffled before feeding into the neural network. For each ray, we sample 64 points using stratified sampling as described in Section 4.4. The encoding length M for position is set to 10. Note that we sample batch data from all training views instead of sequentially sampling data from individual views.

6.2 Evaluation Setup

Dataset. In our experiments, we adopt synthetic light field data from off-the-shelf renderers. Synthetic data provides readily available camera poses which are required by our learning algorithm. We use light fields of five scenes from publicly available datasets [Wetzstein et al. 2011, 2012]. Each light field has 7×7 views and every

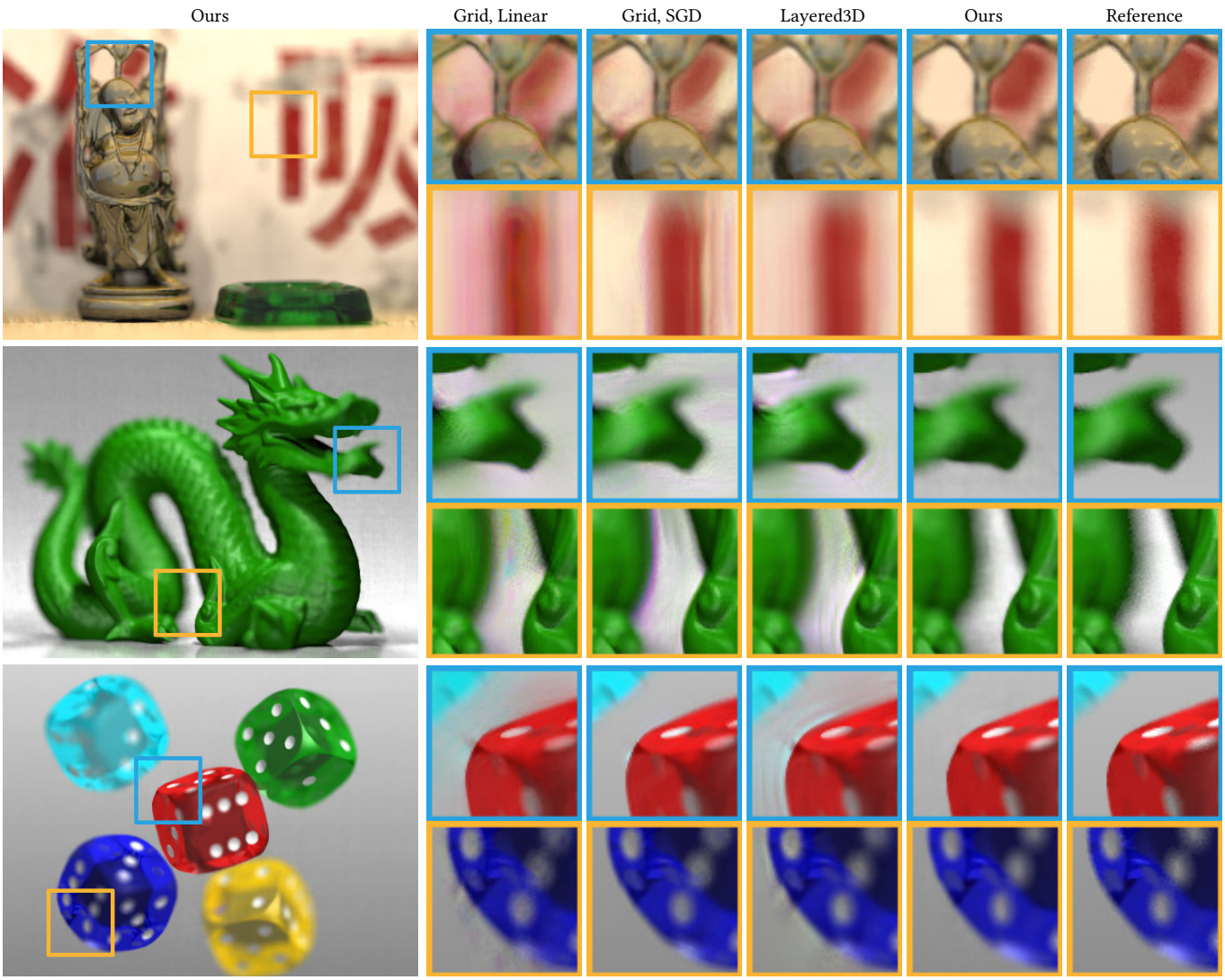


Fig. 4. Quality comparison between our, Layered3D and grid-based approach on the *Buddha*, *Green Dragon*, and *Dice* scene. Note that, for *Ours*, we demonstrate a test view that is not part of the training dataset. For the grid-based approach, we show results obtained by the linear and the SGD solver, respectively. More details are in Section 6.3. See also the supplemental for more results with GIF animations and rendered high-resolution views.

view has a resolution of 512×384. Also, we render new light fields of the *Red Dragon* scene and the *Book* scene using the PBRT [Pharr et al. 2016] renderer. Our light field dataset covers a wide range of variations. The *Butterfly* and *Buddha* scenes are abundant in geometry details, while the *Green Dragon*, *Dice* and *Car* scenes feature depth-of-field and glossy or specular highlights. The *Red Dragon* and *Book* scenes are configured for analysis of form-factors and depth-of-field.

For each light field, we hold out one view for test and use the other views in training. After optimization, we test each method on the same test view to ensure fair comparisons. Note that test views are not used in the optimization processes for all methods.

Evaluated methods. For the rendering simulation, we mainly compare our approach with Layered3D [Wetzstein et al. 2011]. We use its original Matlab implemention and set the number of layers to five according to the specifications in the original work. We also implement the grid-based approach (Section 4.2) and use it as the baseline. The grid-based approach firstly discretizes the volume into a grid of voxels, where the resolution of the grid is pre-defined based on the target resolution for printing and each voxel is associated with a three-channel absorption coefficient. We can directly optimize absorption coefficients on the voxel grid with the Adam optimizer using default parameter settings and a batch size $500K$. In addition, we leverage a constrained large-scale trust-region reflective linear solver [Coleman and Li 1996], which was adopted in Layered3D, to

Table 1. Numerical performance of all evaluated methods on the *Buddha*, *Green Dragon* and *Dice* scenes. This evaluation is conducted on the rendered test views. The image quality is measured with PSNR and SSIM (higher is better), and MAPE (lower is better).

Scene	PSNR				SSIM				MAPE			
	Grid, Linear	Grid, SGD	Layered3D	Ours	Grid, Linear	Grid, SGD	Layered3D	Ours	Grid, Linear	Grid, SGD	Layered3D	Ours
Buddha	22.7003	24.9623	25.0305	**36.1927**	0.8597	0.8648	0.8779	**0.9542**	0.0939	0.0721	0.0773	**0.0252**
Dragon	24.5019	24.4379	25.3789	**34.3736**	0.9332	0.9077	0.9361	**0.9578**	0.1127	0.1182	0.1093	**0.0537**
Dice	27.6937	30.1179	30.2448	**36.6209**	0.9419	0.9557	0.9535	**0.9727**	0.0569	0.0404	0.0464	**0.0221**

Fig. 5. Front view of the printed planar prototype of the *Butterfly* scene. The prototype is printed with 24 slices. We show a visualization of absorption coefficients of slice 1, 9, 17, and 21.

optimize for the grid-based approach. Our method, however, performs continuous-space optimization and does not presume any fixed-resolution voxel-grid structure.

Evaluation metrics. We evaluate the quantitative quality of simulation with three metrics: Peak-Signal-Noise-Ratio (PSNR), Structural Similarity Index (SSIM) [2004] and Mean Absolute Percentage Error (MAPE). MAPE is computed over all K pixels as $1/K \sum_{i=1}^{K} |p_i - g_i|/(g_i + \epsilon)$, where p_i and g_i are the pixel values of the re-rendered image and the reference image respectively, ϵ is set to 0.02.

6.3 Results

Simulation. In Figure 4, we compare our approach with Layered3D and the baseline grid-based methods on the *Buddha*, *Green Dragon* and *Dice* light fields. All methods share the same display thickness 12.5 mm. For the grid-based method, we present its results from both the linear solver and the SGD solver as described in Section 6.2. Given the specified display thickness, the grid-based approach can hardly extend to high-resolution voxel grids due to high storage cost. In practice, we find that the highest resolution of the grid that can be trained on our GPU using the SGD solver is $512 \times 384 \times 60$. However, a grid resolution higher than $512 \times 384 \times 46$ leads to an under-determined optimization which is not supported by the linear solver. We, therefore, set the resolution of the grid for the baseline to $512 \times 384 \times 45$.

In the test stage, we run all methods to render the same test view (the 41-st view) of each scene to allow a fair comparison. As shown in Figure 4, both solutions to the grid-based approach show blurriness near boundaries (see insets of the *Buddha* scene and the *Dice* scene) and exhibit objectionable color artifacts (see insets of the *Dragon* scene), whereas Layered3D is plagued with the ghosting artifacts at the rims of objects. Our approach generally provides rendered views with higher visual quality than the compared methods. We tabulate the numerical evaluation results in Table 1. For all scenes, our method achieves lower errors than other methods, which is consistent with the visual quality advantages of our approach.

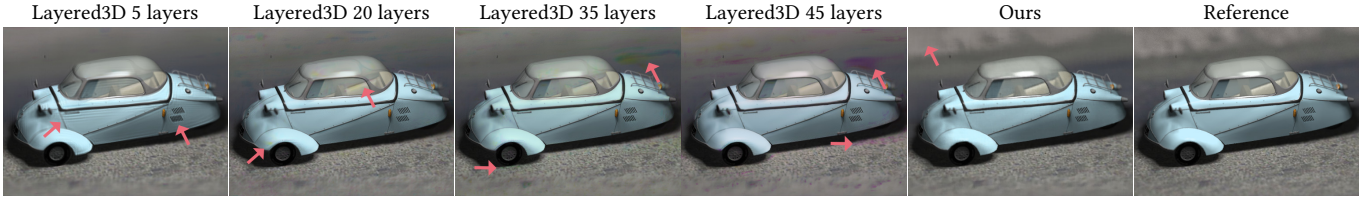
Prototyping. In Figure 1, we demonstrate our approach with 3D printed prototypes of the light field of the *Butterfly*. As the printing workflow of Section 5, we sample absorption coefficients according to the target resolution and halftone them for 3D printing. Figure 5 shows a visualization of optimized absorption coefficients from the planar prototype. Going beyond the planar prototype, we fabricate a prototype with a non-planar serpentine shape (Figure 1(d)).

After encoding a light field with a neural network, we can print prototype with an arbitrary precision supported by the printer. However, the software of our Objet260 3D printer has a limit on the size of the input meshes. This constraint limits us to printing small-scale models. For the prototype of the *Butterfly* scene, we use the highest printing precision 84 µm for the height and width dimension. For the depth dimension, we discretize the learned representation into 24 slices and each slice has a depth 0.21 mm. Finally, the prototype has a size $43.0 \times 32.3 \times 5.1$ mm.

7 DISCUSSION

Comparisons with multilayer approaches. Layered3D optimizes multiple discrete 2D layers, while our approach optimizes on a continuous volume, in which it integrates the transmittance along rays. Figure 6 investigates the capacity of our continuous representation and the layer-based representation by comparing rendered test views, where we vary the number of layers of Layered3D from 5 to 45. As indicated by red arrows, Layered3D suffers from ghosting and color artifacts.

The statistics of unknowns and run-time memory footprints are listed in Table 2. Layered3D formulates the problem as a constrained linear system. It uses a fixed number of discrete 2D layers and pre-computes a ray propagation matrix which it solves efficiently with a linear solver. In our case, the training time comprehensively hinges on hyperparameters like training epochs and mini-batch size.



PSNR 29.4445	30.0561	29.0419	28.4675	34.2322
SSIM 0.8793	0.9109	0.8911	0.8821	0.9240

Fig. 6. Comparisons between our method and Layered3D using different number of layers. The maximum iterations of Layered3D is 15, that is the same as in the original paper. Our neural network is trained for 3000 epochs and each epoch has 20 batches. The display thickness is 12.5 mm.

PSNR 22.9072	23.4009	23.3177	23.3769	35.9354
SSIM 0.8096	0.8300	0.8194	0.8287	0.9500

Fig. 7. Comparisons of the grid-based method, the layer-based method and our approach on the 9-th test view. The display thickness is 12.5 mm.

Table 2. Comparisons of parameter counts, optimization timings and running memory. Our neural network has eight hidden layers, each with 512 neurons. CPU denotes the RAM of a workstation and GPU denotes the VRAM of a graphics card.

Solutions	Parameters	Timings (h)	Memory (GB)
Layered3D-5 (CPU)	2949120	0.171	2.794
Layered3D-20 (CPU)	11796480	0.699	11.245
Layered3D-25 (CPU)	14745600	0.995	14.925
Layered3D-35 (CPU)	20643840	1.109	20.298
Layered3D-45 (CPU)	26542080	1.320	24.712
Ours (GPU)	1942019	1.453	2.659

Furthermore, our approach finds the absorbance via a numerical integration (Equation 6), where the number of samples on a ray also affects the training time. Note that the peak usage of main memory by Layered3D with 45 layers is higher than the VRAM size of our GPU.

Additionally, we show the optimization time cost and quality measurement in Figure 10, which tells that increasing the number of layers for Layered3D does not lead to a consistent quality improvement. Instead, more layers lead to higher time cost and inferior quality, although 45 layers possess 13 times more parameters than our neural network.

Comparisons using equal number of unknowns. To demonstrate that our neural network based implicit representation is more compact, we conduct a comparison with the grid-based representation and the layer-based representation [2011] under the condition of equal number of unknowns. The compared methods are configured with four RGB layers and each has a resolution of 512×384, which leads to $2,359,296$ parameters. Accordingly, we employ a neural network with eight hidden layers and 565 neurons per layer, which amounts to $2,352,663$ parameters. For the two compared methods, we solve them with both the trust-region reflective linear solver (Linear) and an SGD algorithm as used in our approach. The maximum iteration count of the linear solver is set to 15.

Table 3. Optimization timings and running memory of the equal-unknowns comparisons. Our neural network uses eight hidden layers and 565 neurons per layer.

Solutions	Timings (h)	Memory (GB)
Grid-based, Linear (CPU)	0.196	2.359
Grid-based, SGD (GPU)	2.440	1.362
Layer-based, Linear (CPU)	0.164	2.541
Layer-based, SGD (GPU)	2.320	1.238
Ours (GPU)	2.162	4.429

From images of the grid-based representation and the layer-based representation in Figure 7, ghosting and blurriness artifacts (indicated with cyan arrows) can be observed on the statue and the background character. The SGD solver performs slightly better than the linear solver in terms of numerical measurement but the advantage is relatively small. Our result presents fewer visual artifacts compared to others. Also, Table 3 lists the optimization timings and the running memory footprints. By formulating the grid-based approach and the layered-based approach as linear systems, the linear solver can efficiently find solutions but it does not lead to visually pleasing results. On the other hand, the time cost of the SGD solver depends on epochs, number of batches per epoch, and mini-batch size. For the SGD solver, we run 3000 epochs with 20 batches per epoch. Our batch size is 1200, whereas it is 500K for the *compared* methods.

Transmission maps. In Figure 8, we investigate the optimization evolution of our approach compared to Layered3D. Layered3D is assigned with five discrete layers which the method optimizes directly whereas our approach learns absorption coefficients in a continuous

Fig. 8. Comparisons of transmission maps from Layered3D (upper four rows) and our approach (lower four rows). The thickness of the display is 12.5 mm. The rightmost column shows the rendered test views.

space. We, therefore, get the approximate five transmission maps by integrating learned absorption coefficients. For Layered3D, we visualize its transmission maps and the corresponding rendering of the test view at iteration 2, 4, 7, and 15. Accordingly, we present our integrated transmission maps at training epoch 100, 500, 1500, and 3000. In the transmission maps of Layered3D, ringing and blurriness around the contour of the butterfly can be clearly observed. This explains the ringing artifacts in the rendered image. On the contrary, our transmission maps gradually get sharper along the training and the details in local regions are better preserved. The

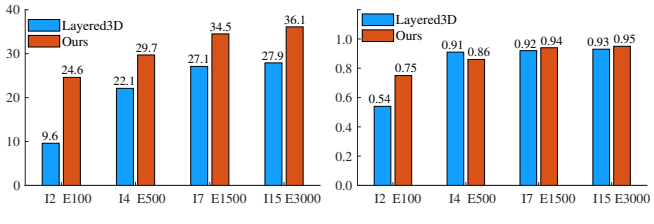


Fig. 9. PSNR (left) and SSIM (right) of Layered3D and our approach on the test view. Layered3D is measured at iteration (I) 2, 4, 7, and 15, and our approach is measured at epoch (E) 100, 500, 1500, and 3000.

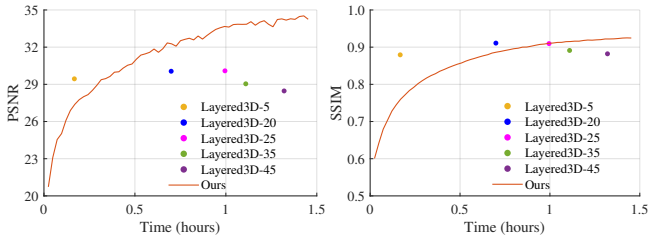


Fig. 10. Image quality comparisons on the test view of the *Car* scene. Dots of Layered3D present its overall optimization time and final quality. For our method, we also show the quality measurement along the training.

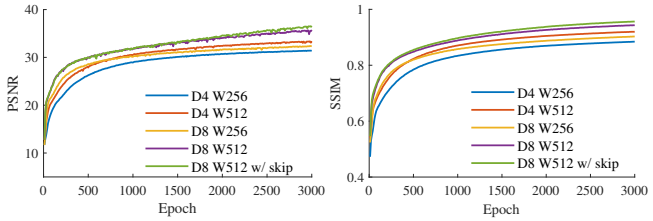


Fig. 11. Evaluations of different neural network architectures by training on the *Butterfly* scene. Both PSNR (left) and SSIM (right) are measured on the test view.

numerical quality measurements of rendered images are presented in Figure 9.

Architecture. Figure 11 compares the performance of different architectures. Here, we control the complexity of a neural network by the number of hidden layers D and the number of neurons W per layer. For the D8W512 architecture, we include its variant with skip connections (Figure 2) in the comparison. For other compared architectures, skip connections are by default not used.

For each case, we train the neural networks for 3000 epochs, test them every 10 epochs, and evaluate rendered images in terms of PSNR and SSIM. As shown, deep and wide architecture generally provides better numerical performance. In addition, skip connections bring slight extra improvement. For the D8W512 architecture, we present its training dynamics in the supplemental to measure the quality improvement during training. Also, its average time cost for rendering a view with resolution 512×384 using 64 samples on each ray is 11.2 s.

Field of view. In Figure 12, we investigate the capability of our neural network to learn light fields with large fields of view (FOV). Since volumetric displays allow only finite depth of field, we render the target light field of the *Red Dragon* scene with defocus blur. Three light fields of the scene are rendered by setting the FOV angle to 10, 20, and 40 degrees. To adapt to a higher FOV, we accordingly provide more views to the neural network. Specifically, we render 7×7, 9×9, and 11×11 views of the three light fields respectively. For each light field, we reserve its central view for test and use the other views in training.

We separately train a D8W512 neural network on each light field for 3000 epochs. As can be observed, the neural network trained on FOV 10° has the closest appearance to the reference. As the FOV increases, the neural network will gradually compromise the spatial resolution to learn wider angular changes. Note the head of the dragon in the case of FOV 40° gets blurrier than others. The trade-off between angular resolution and spatial resolution confirms the findings reported in the previous work [Tompkin et al. 2013].

Thickness of display. The number of layers and display thickness are two crucial factors for designing multilayer displays. Since our approach optimizes on continuous volume instead of discrete layers, we consider the impact of display thickness. Figure 13 shows comparisons of our approach using different thickness for the *Dice* scene. The edge length of a die is approximately 15 mm. As can be seen, when encoding the light field within a display of very small thickness, e.g., 0.8 mm, the rendered test view significantly lose contrast. Note the bright dots on dices are washed out. This issue can be alleviated by increasing the thickness. With a thickness of 6.5 mm, the rendered result achieves the appearance and contrast which are close to the reference.

7.1 Limitations and Future Work

One limitation of our approach is that the printed volume can only support finite depth of field. We investigate this issue by training our neural network on a sharply focused *Book* scene. We render a 7×7 light field of the scene with the FOV as 20° and set the display thickness to 12.5 mm. It can be observed from the comparison in Figure 14 that only finite depth of field is rendered in the test view. The pink rabbit and rear part of the book get blurred because they are located outside of the finite depth-of-field range.

Another practical issue is our training time cost compared to Layered3D using only a few layers. Since our approach adopts the mini-batch Adam optimizer, its training time is majorly affected by mini-batch size, training epochs, and batches per epoch. In addition, our approach learns the volume density within a continuous space and the number of samples on rays used for evaluating absorbance also functions as an affecting factor of the training time. As demonstrated in multiple results, our method brings improvements in both visual quality and numerical performance. In practice, the training process does not hinder the application of our approach, as the training is only conducted once before the printing.

Although the MLP serves well to our needs, it is possible to use other sparse representations like wavelets [Mallat 1999] to have some theoretical guarantees. Future work can also benefit from

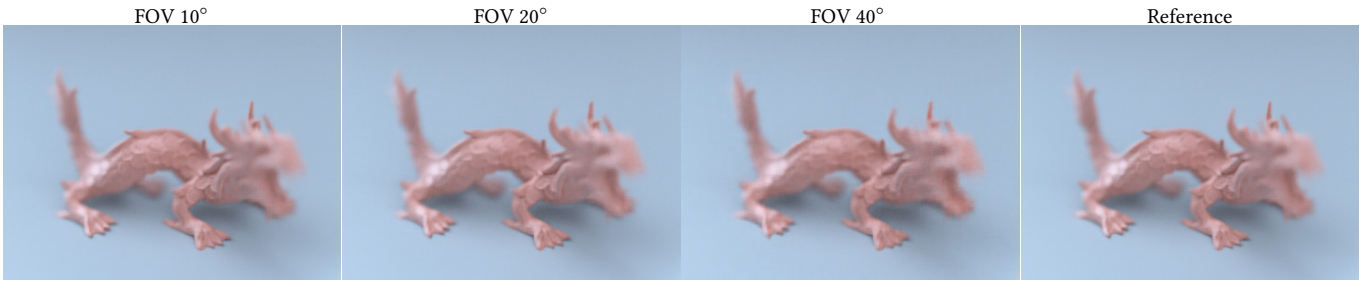


Fig. 12. The capability of neural networks to encode light fields with different field of views. With the FOV increasing, the neural network tends to compromise its spatial resolution. Note how the head of the dragon in the FOV 40° case gets blurrier than others.

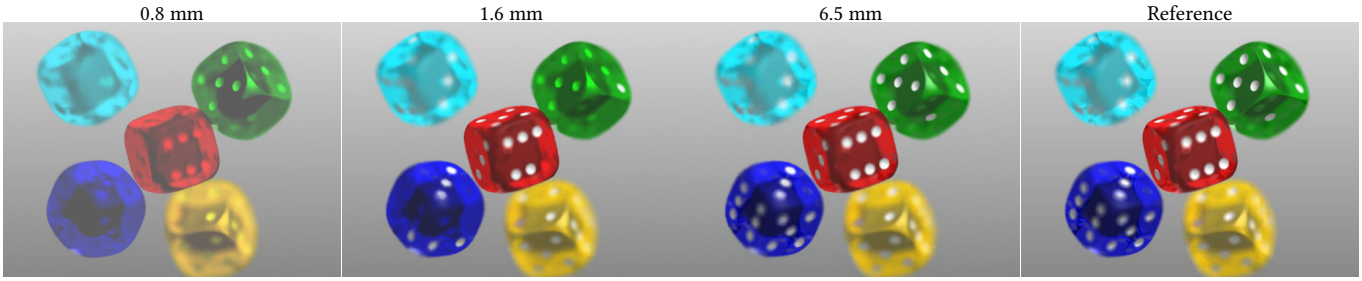


Fig. 13. Investigation of display thickness. We train neural networks to encode a light field of the *Dice* scene using different thickness of the display. Small thickness leads to contrast loss. Note the bright dots on blue and red dices are missing for thickness 0.8 mm and 1.6 mm.

Fig. 14. Comparison of a rendered test view (left) with its reference (right) on the *Book* scene. The reference image has infinite depth of field. When encoding the light field onto a display of 12.5 mm thickness, only limited depth of field (left) is exhibited.

using improved ray casting procedures that can leverage space-partitioning data-structures (e. g., Octrees) in a multi-resolution manner [Liu et al. 2020; Srinivasan et al. 2020].

In this work, we assume no scattering occurs in the volume. For the printed prototypes, the effects of scattering is rarely noticeable so the assumption holds here. For printing materials with significant scattering, the scattering effects need to be addressed along with absorption at the same time and we leave it as a future step. In addition, we use uniform back-lighting in the design and this can be extended to directional lighting.

8 CONCLUSION

We have proposed a novel and practical approach for printing input light field imagery as an attenuation-based physical volumetric display. We utilize a neural network that encodes the input light field data as a continuous representation of the medium density. With our implementation of a volumetric ray casting process, we can train our neural network end-to-end. We have demonstrated the effectiveness of our approach in comprehensive simulation experiments. Our approach significantly outperforms, both qualitatively and quantitatively, the existing multilayer approach and the grid-based approach with a similar number of unknowns. Essentially, our approach takes advantage of the fact that MLPs are more memory-efficient function approximators compared to uniformly sampled grids, which allows us to achieve higher resolution and higher quality light fields compared to the baseline techniques. We validate our simulation with fabricated prototypes. Our approach also allows printing light fields as either planar or non-planar volumetric displays.

ACKNOWLEDGMENTS

We would like to thank all anonymous reviewers for their valuable feedback. The first author is funded by the Research Executive Agency 739578.

REFERENCES

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, and et al. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/ Software available from tensorflow.org.

Vahid Babaei, Kiril Vidimče, Michael Foshey, Alexandre Kaspar, Piotr Didyk, and Wojciech Matusik. 2017. Color Contoning for 3D Printing. *ACM Trans. Graph.* 36, 4 (July 2017), 124:1–124:15. https://doi.org/10.1145/3072959.3073605

Ilya Baran, Philipp Keller, Derek Bradley, Stelian Coros, Wojciech Jarosz, Derek Nowrouzezahrai, and Markus Gross. 2012. Manufacturing layered attenuators for multiple prescribed shadow images. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 603–610.

Peter C Barnum, Srinivasa G Narasimhan, and Takeo Kanade. 2010. A multi-layered display with water drops. In *ACM SIGGRAPH 2010 papers*. 1–7.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2012. Representation Learning: A Review and New Perspectives. arXiv:1206.5538 [cs.LG]

BG Blundell, AJ Schwarz, and DK Horrell. 1993. Volumetric threedimensional display systems: their past present and future. *Engineering Science & Education Journal* 2, 5 (1993), 196–200.

Barry G Blundell, Adam J Schwarz, and Damon K Horrell. 1994. Cathode ray sphere: a prototype system to display volumetric three-dimensional images. *Optical Engineering* 33, 1 (1994), 180–187.

Alan Brunton, Can Ates Arikan, and Philipp Urban. 2015. Pushing the Limits of 3D Color Printing: Error Diffusion with Translucent Materials. *ACM Trans. Graph.* 35, 1 (Dec. 2015), 4:1–4:13. https://doi.org/10.1145/2832905

Thomas F Coleman and Yuying Li. 1996. A reflective Newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM Journal on Optimization* 6, 4 (1996), 1040–1058.

Oliver S Cossairt, Joshua Napoli, Samuel L Hill, Rick K Dorval, and Gregg E Favalora. 2007. Occlusion-capable multiview volumetric three-dimensional display. *Applied optics* 46, 8 (2007), 1244–1250.

Robert A Drebin, Loren Carpenter, and Pat Hanrahan. 1988. Volume rendering. *ACM Siggraph Computer Graphics* 22, 4 (1988), 65–74.

Oskar Elek, Denis Sumin, Ran Zhang, Tim Weyrich, Karol Myszkowski, Bernd Bickel, Alexander Wilkie, and Jaroslav Křivánek. 2017. Scattering-aware Texture Reproduction for 3D Printing. *ACM Trans. Graph.* 36, 6, Article 241 (Nov. 2017), 15 pages.

Gregg E Favalora. 2005. Volumetric 3D displays and application infrastructure. *Computer* 38, 8 (2005), 37–44.

Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. 2013. Inverse volume rendering with material dictionaries. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 1–13.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 249–256.

Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. 1996. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 43–54.

Hironobu Gotoda. 2010. A multilayer liquid crystal display for autostereoscopic 3D viewing. In *Stereoscopic Displays and Applications XXI*, Vol. 7524. International Society for Optics and Photonics, 75240P.

K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778.

Michael Holroyd, Ilya Baran, Jason Lawrence, and Wojciech Matusik. 2011. Computing and fabricating multilayer models. In *Proceedings of the 2011 SIGGRAPH Asia Conference*. 1–8.

Youngjin Jo, Seungjae Lee, Dongheon Yoo, Suyeon Choi, Dongyeon Kim, and Byoungho Lee. 2019. Tomographic projector: large scale volumetric display with uniform viewing experiences. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–13.

Andrew Jones, Ian McDowall, Hideshi Yamada, Mark Bolas, and Paul Debevec. 2007. Rendering for an interactive 360 light field display. In *ACM SIGGRAPH 2007 papers*. 40–es.

Simon Kallweit, Thomas Müller, Brian Mcwilliams, Markus Gross, and Jan Novák. 2017. Deep Scattering: Rendering Atmospheric Clouds with Radiance-Predicting Neural Networks. *ACM Trans. Graph.* 36, 6, Article 231 (Nov. 2017), 11 pages. https://doi.org/10.1145/3130800.3130880

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*.

Douglas Lanman, Matthew Hirsch, Yunhee Kim, and Ramesh Raskar. 2010. Content-adaptive parallax barriers: optimizing dual-layer 3D displays using low-rank light field factorization. In *ACM SIGGRAPH Asia 2010 papers*. 1–10.

Douglas Lanman, Gordon Wetzstein, Matthew Hirsch, Wolfgang Heidrich, and Ramesh Raskar. 2011. Polarization fields: dynamic light field display using multi-layer LCDs. In *Proceedings of the 2011 SIGGRAPH Asia Conference*. 1–10.

Yann LeCun, Y. Bengio, and Geoffrey Hinton. 2015. Deep Learning. *Nature* 521 (05 2015), 436–44. https://doi.org/10.1038/nature14539

Marc Levoy and Pat Hanrahan. 1996. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 31–42.

Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. 2020. Neural Sparse Voxel Fields. arXiv:2007.11571

Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. 2019. Neural Volumes: Learning Dynamic Renderable Volumes from Images. *ACM Trans. Graph.* 38, 4, Article 65 (July 2019), 14 pages. https://doi.org/10.1145/3306346.3323020

A Loukianitsa and Andrey N Putilin. 2002. Stereodisplay with neural network image processing. In *Stereoscopic Displays and Virtual Reality Systems IX*, Vol. 4660. International Society for Optics and Photonics, 207–211.

Hiroyuki Maeda, Kazuhiko Hirose, Jun Yamashita, Koichi Hirota, and Michitaka Hirose. 2003. All-around display for video avatar in real world. In *The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003. Proceedings.* IEEE, 288–289.

Stéphane Mallat. 1999. *A wavelet tour of signal processing*. Elsevier.

Kshitij Marwah, Gordon Wetzstein, Yosuke Bando, and Ramesh Raskar. 2013. Compressive light field photography using overcomplete dictionaries and optimized projections. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–12.

Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. 2019. Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines. *ACM Trans. Graph.* 38, 4, Article 29 (July 2019), 14 pages. https://doi.org/10.1145/3306346.3322980

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. arXiv:2003.08934 [cs.CV]

Niloy J Mitra and Mark Pauly. 2009. Shadow art. *ACM Transactions on Graphics (TOG)* 28, 5 (2009), 1–7.

Shree K Nayar and Vijay N Anand. 2007. 3D display using passive optical scatterers. *Computer* 40, 7 (2007), 54–63.

Ren Ng, Marc Levoy, Mathieu Brédif, Gene Duval, Mark Horowitz, Pat Hanrahan, et al. 2005. Light field photography with a hand-held plenoptic camera. *Computer Science Technical Report CSTR* 2, 11 (2005), 1–11.

Victor Ostromoukhov. 2001. A simple and efficient error-diffusion algorithm. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 567–572.

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2016. *Physically Based Rendering: From Theory to Implementation* (3 ed.). Morgan Kaufmann.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. 2018. On the Spectral Bias of Neural Networks. arXiv:1806.08734

Artin Saberpour, Roger D Hersch, Jiajing Fang, Rhaleb Zayer, Hans-Peter Seidel, and Vahid Babaei. 2020. Fabrication of moiré on curved surfaces. *Optics Express* 28, 13 (2020), 19413–19427.

Liang Shi, Vahid Babaei, Changil Kim, Michael Foshey, Yuanming Hu, Pitchaya Sitthi-Amorn, Szymon Rusinkiewicz, and Wojciech Matusik. 2018. Deep multispectral painting reproduction via multi-layer, custom-ink printing. *ACM Transactions on Graphics* 37, 6 (Dec. 2018), 271:1–271:15. https://doi.org/10.1145/3272127.3275057

Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. 2019. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*. 1121–1132.

Pratul P. Srinivasan, Ben Mildenhall, Matthew Tancik, Jonathan T. Barron, Richard Tucker, and Noah Snavely. 2020. Lighthouse: Predicting Lighting Volumes for Spatially-Coherent Illumination. In *CVPR*.

Stratasys. 2020. Stratasys ultimate full-color multi-material 3D printer. https://www.stratasys.com/3d-printers/j8-series. [Online; Accessed 15-05-2020].

Denis Sumin, Tobias Rittig, Vahid Babaei, Thomas Nindel, Alexander Wilkie, Piotr Didyk, Bernd Bickel, Jaroslav Křivánek, Karol Myszkowski, and Tim Weyrich. 2019. Geometry-Aware Scattering Compensation for 3D Printing. *ACM Trans. Graph.* 38, 4, Article 111 (July 2019), 14 pages.

Matthew Tancik, Pratul P Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T Barron, and Ren Ng. 2020. Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739* (2020).

James Tompkin, Simon Heinzle, Jan Kautz, and Wojciech Matusik. 2013. Content-adaptive lenticular prints. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–10.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.

Gordon Wetzstein, Douglas Lanman, Wolfgang Heidrich, and Ramesh Raskar. 2011. Layered 3D: tomographic image synthesis for attenuation-based light field and high dynamic range displays. In *ACM SIGGRAPH 2011 papers*. 1–12.

Gordon Wetzstein, Douglas Lanman, Matthew Hirsch, and Ramesh Raskar. 2012. Tensor displays: compressive light field synthesis using multilayer displays with directional backlighting. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–11.

Tomohiro Yendo, Naoki Kawakami, and Susumu Tachi. 2005. Seelinder: the cylindrical lightfield display. In *ACM SIGGRAPH 2005 Emerging technologies*. 16–es.