Progressive Multi-Scale Light Field Networks

David Li* Amitabh Varshney
University of Maryland, College Park

Abstract

Neural representations have shown great promise in their ability to represent radiance and light fields while being very compact compared to the image set representation. However, current representations are not well suited for streaming as decoding can only be done at a single level of detail and requires downloading the entire neural network model. Furthermore, high-resolution light field networks can exhibit flickering and aliasing as neural networks are sampled without appropriate filtering. To resolve these issues, we present a progressive multi-scale light field network that encodes a light field with multiple levels of detail. Lower levels of detail are encoded using fewer neural network weights enabling progressive streaming and reducing rendering time. Our progressive multi-scale light field network addresses aliasing by encoding smaller anti-aliased representations at its lower levels of detail. Additionally, per-pixel level of detail enables our representation to support dithered transitions and foveated rendering.

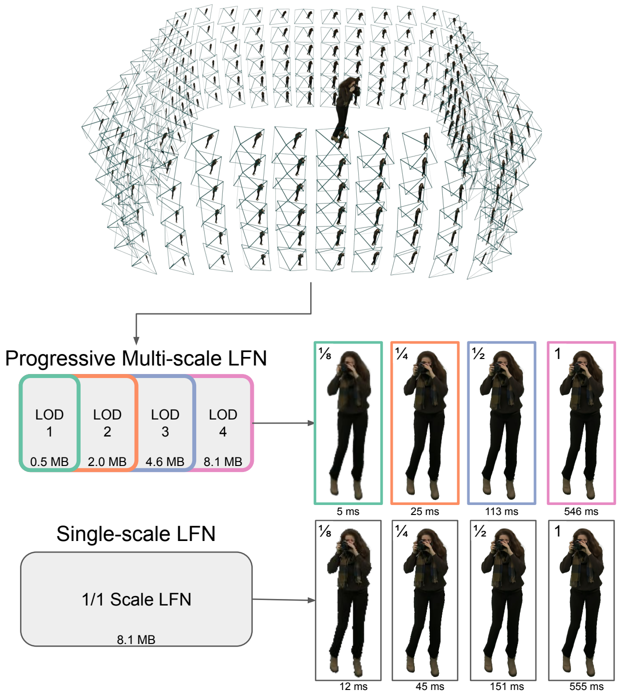


Figure 1: Our progressive multi-scale light field network is suited for streaming, reduces aliasing and flicking, and renders more efficiently at lower resolutions.

1. Introduction

Volumetric images and video allow users to view 3D scenes from novel viewpoints with a full 6 degrees of freedom (6DOF). Compared to conventional 2D and 360 images and videos, volumetric content enables additional immersion with motion parallax and binocular stereo while also allowing users to freely explore the full scene.

Traditionally, scenes generated with 3D modeling have been commonly represented as meshes and materials. These classical representations are suitable for real-time rendering as well as streaming for 3D games and AR effects. However, real captured scenes are challenging to represent using the mesh and material representation, requiring complex reconstruction and texture fusion techniques [10, 11]. To produce better photo-realistic renders, existing view-synthesis techniques have attempted to adapt these representations with multi-plane images (MPI) [69], layered-depth images (LDI) [49], and multi-sphere images (MSI) [2]. These representations allow captured scenes to be accurately represented with 6DOF but only within a limited area.

Radiance fields and light fields realistically represent captured scenes from an arbitrarily large set of viewpoints. Recently, neural representations such as neural radiance fields (NeRFs) [31] and light field networks (LFNs) [51] have been found to compactly represent 3D scenes and perform view-synthesis allowing re-rendering from arbitrary viewpoints. Given a dozen or more photographs capturing a scene from different viewpoints from a conventional camera, a NeRF or LFN can be optimized to accurately reproduce the original images and stored in less than 10 MB. Once trained, NeRFs will also produce images from different viewpoints with photo-realistic quality.

*Email: dli7319@umd.edu
Project page: https://augmentariumlab.github.io/multiscale-lfn/

While NeRFs are able to encode and reproduce real-life 3D scenes from arbitrary viewpoints with photorealistic accuracy, they suffer from several drawbacks which limit their use in a full on-demand video streaming pipeline. Some notable drawbacks include slow rendering, aliasing at smaller scales, and a lack of progressive decoding. While some of these drawbacks have been independently addressed in follow-up work [3, 66], approaches to resolving them cannot always be easily combined and may also introduce additional limitations.

In this paper, we extend light field networks with multiple levels of detail and anti-aliasing at lower scales making them more suitable for on-demand streaming. This is achieved by encoding multiple reduced-resolution representations into a single network. With multiple levels of details, light field networks can be progressively streamed and rendered based on the desired scale or resource limitations. For free-viewpoint rendering and dynamic content, anti-aliasing is essential to reducing flickering when rendering at smaller scales. In summary, the contributions of our progressive multi-scale light field network are:

1. Composing multiple levels of detail within a single network takes up less space than storing all the levels independently.
2. The progressive nature of our model requires us to only download the parameters necessary up to the desired level of detail.
3. Our model allows rendering among multiple levels of detail simultaneously. This makes for a smooth transition between multiple levels of detail as well as spatially adaptive rendering (including foveated rendering) without requiring independent models at multiple levels of detail on the GPU.
4. Our model allows for faster inference of lower levels of detail by using fewer parameters.
5. We are able to encode light fields at multiple scales to reduce aliasing when rendering at lower resolutions.

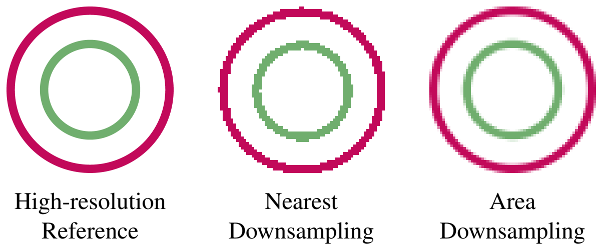


Figure 2: A high-resolution image resized using nearest and area downsampling. Rendering a low-resolution image from a high-resolution light field is similar to performing nearest downsampling, i.e. subsampling a single value. Area downsampling, subsampling with a box filter, generates an anti-aliased image but cannot be done efficiently on an LFN.

Table 1: Neural 3D Representations Comparison

Model	Real-time	Model Size	Multi-scale	Progressive
NeRF	✗	$\leq$ 10 MB	✗	✗
mip-NeRF	✗	$\leq$ 10 MB	✓	✗
Plenoctree	✓	$\geq$ 30 MB	✗	✗
LFN	✓	$\leq$ 10 MB	✗	✗
Ours	✓	$\leq$ 10 MB	✓	✓

2. Background and Related Works

Our work builds upon existing work in neural 3D representations, adaptive neural network inference, and levels of detail. In this section, we provide some background on neural representations and focus on prior work most related to ours. For a more comprehensive overview of neural rendering, we refer interested readers to a recent survey [54].

2.1. Neural 3D Representations

Traditionally, 3D content has been encoded in a wide variety of formats such as meshes, point clouds, voxels, multiplane images, and even side-by-side video. To represent the full range of visual effects from arbitrary viewpoints, researchers have considered using radiance fields and light fields which can be encoded using neural networks. With neural networks, 3D data such as signed distance fields, radiance fields, and light fields can be efficiently encoded as coordinate functions without explicit limitations on the resolution.

2.1.1 Neural Radiance Field (NeRF)

Early neural representations focused on signed distance functions and radiance fields. Neural Radiance Fields (NeRF) [31] use differentiable volume rendering to encode multiple images of a scene into a radiance field encoded as a neural network. With a sufficiently dense set of images, NeRF is able to synthesize new views by using the neural network to interpolate radiance values across different positions and viewing directions. Since the introduction of NeRF, followup works have added support for deformable scenes [40, 37, 38], real-time inference [17, 42, 33, 65], improved quality [50], generalization across scenes [67, 41], generative modelling [48, 35, 61], videos [24, 58, 56], sparse views [8, 34, 19], fast training [32, 65], and even large-scale scenes [43, 59, 55].

Among NeRF research, Mip-NeRF [3, 4] and BACON [27] share a similar goal to our work in offering a multi-scale representation to reduce aliasing. Mip-NeRF approximates the radiance across conical regions along the ray, cone-tracing, using integrated positional encoding (IPE). With cone-tracing, Mip-NeRF reduces aliasing while also slightly reducing training time. BACON [27] provide a multi-scale scene representation through band-limited networks using Multiplicative Filter Networks [13] and multiple exit points. Each scale in BACON has band-limited outputs in Fourier space.

2.1.2 Light Field Network (LFN)

Our work builds upon Light Field Networks (LFNs) [51, 16, 25, 6]. Light field networks encode light fields [21] using a coordinate-wise representation, where for each ray $\mathbf{r}$, a neural network f is used to directly encode the color $c = f(\mathbf{r})$. To represent rays, Feng and Varshney [16] pair the traditional two-plane parameterization with Gegenbauer polynomials while Sitzmann *et al.* [51] use Plücker coordinates which combine the ray direction $\mathbf{r}_d$ and ray origin $\mathbf{r}_o$ into a 6-dimensional vector $(\mathbf{r}_d, \mathbf{r}_o \times \mathbf{r}_d)$. Light fields can also be combined with explicit representations [36]. In our work, we adopt the Plücker coordinate parameterization which can represent rays in all directions.

Compared to NeRF, LFNs are faster to render as they only require a single forward pass per ray, enabling real-time rendering. However, LFNs are worse at view synthesis as they lack the self-supervision that volume rendering provides with explicit 3D coordinates. As a result, LFNs are best trained from very dense camera arrays or from a NeRF teacher model such as in [1]. Similar to NeRFs, LFNs are encoded as multi-layer perceptrons, hence they remain compact compared to voxel and octree NeRF representations [66, 18, 17, 65] which use upwards of 50 MB. Therefore LFNs strike a balance between fast rendering times and storage compactness which could make them suitable for streaming 3D scenes over the internet.

2.2. Levels of Detail

Level of detail methods are commonly used in computer graphics to improve performance and quality. For 2D textures, one of the most common techniques is mipmapping [57] which both reduces aliasing and improves performance with cache coherency by encoding textures at multiple resolutions. For neural representations, techniques for multiple levels of detail have also been proposed for signed distance functions as well as neural radiance fields.

For signed distance fields, Takikawa *et al.* [53] propose storing learned features within nodes of an octree to represent a signed distance function with multiple levels of detail. Building upon octree features, Chen *et al.* [7] develop Multiresolution Deep Implicit Functions (MDIF) to represent 3D shapes which combines a global SDF representation with local residuals. For radiance fields, Yang *et al.* [62] develop Recursive-NeRF which grows a neural representation with multi-stage training which is able to reduce NeRF rendering time. BACON [27] offers levels of detail for various scene representations including 2D images and radiance fields with different scales encoding different bandwidths in Fourier space. PINs [20] uses a progressive fourier encoding to improve reconstruction for scenes with a wide frequency spectrum. For more explicit representations, Variable Bitrate Neural Fields [52] use a vector-quantized auto-decoder to compress feature grids into lookup indices and a learned codebook. Levels of detail are obtained by learning compressed feature grids at multiple resolutions in a sparse octree. MINER [45] learns neural representations at different scales of a Laplacian pyramid. In parallel, Streamable Neural Fields [9] propose using progressive neural networks [44] to represent spectrally, spatially, or temporally growing neural representations.

2.3. Adaptive Inference

Current methods use adaptive neural network inference based on available computational resources or the difficulty of each input example. Bolukbasi *et al.* [5] propose two schemes for adaptive inference. Early exiting allows intermediate layers to route outputs directly to an exit head while network selection dynamically routes features to different networks. Both methods allow easy examples to be classified by a subset of neural network layers. Yeo *et al.* [64] apply early exiting to upscale streamed video. By routing intermediate CNN features to exit layers, a single super-resolution network can be partially downloaded and applied on a client device based on available compute. Similar ideas have also been used to adapt NLP to input complexity [70, 28]. Yang *et al.* [63] develop Resolution Adaptive Network (RANet) which extracts features and performs classification of an image progressively from low-resolution to high-resolution. Doing so sequentially allows efficient early exiting as many images can be classified with only coarse features. Slimmable neural networks [68] train a model at different widths with a switchable batch normalization and observe better performance than individual models at detection and segmentation tasks.

3. Method

Our method consists of a multi-scale light field encoded using a progressive neural network to reduce aliasing and enable adaptive streaming and rendering.

3.1. Multi-scale Light Field

We propose encoding multiple representations of the light field which produce anti-aliased representations of a light field, similar to mipmaps for conventional images. For this, we generate an image pyramid for each image view with area downsampling and encode each resolution as a separate light field representation. In contrast to bilinear or nearest pixel downsampling, area downsampling creates an anti-aliased view as shown in Figure 2 where each pixel in the downsampled image is an average over a corresponding area in the full-resolution image.

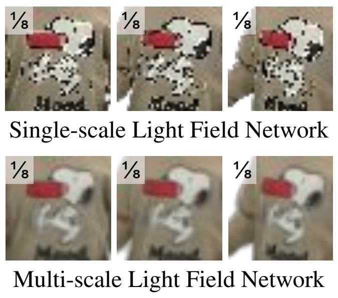

1/8
1/8
1/8
Single-scale Light Field Network
1/8
1/8
1/8
Multi-scale Light Field Network

Figure 3: Rendering a single full-scale LFN at a lower (1/8) resolution results in aliasing, unlike the multi-scale LFN at the same (1/8) resolution. When changing viewpoints (3 shown above), the aliasing results in flickering.

As with mipmaps, lower-resolution representations trade-off sharpness for less aliasing. In this case, where a light field is rendered into an image, sharpness may be preferable over some aliasing. However, in the case of videos where the light field is viewed from different poses or the light field is dynamic, sampling from a high-resolution light field leads to flickering. With a multi-scale representation, rendering light fields at smaller resolutions can be done at a lower scale to reduce flickering.

3.2. Progressive Model

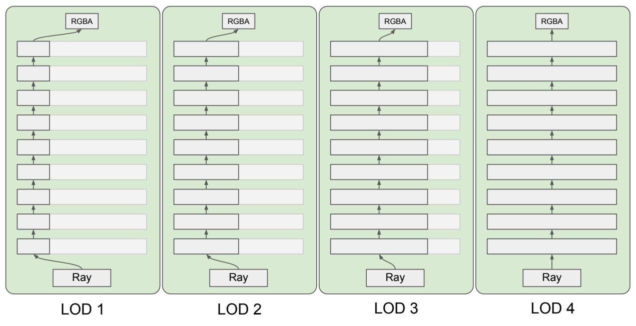


Figure 4: Using subsets of neurons to encode different levels of detail within a single model.

For light field streaming, we wish to have a compact representation that can simultaneously encode multiple levels of detail into a single progressive model. Specifically, we seek a model with the following properties: Given specific subsets of network weights, we should be able to render a complete image with reduced details. Progressive levels of details should share weights with lower levels to eliminate encoding redundancy.

With the above properties, our model offers several benefits over a standard full-scale network or encoding multi-scale light fields using multiple networks. First, our model composes all scales within a single network hence requiring less storage space than storing four separate models. Second, our model is progressive, thus only parameters necessary for the desired level of detail are needed in a streaming scenario. Third, our model allows rendering different levels of detail across pixels for dithered transitions and foveation. Fourth, our model allows for faster inference of lower levels of detail by utilizing fewer parameters. Model sizes and training times appear in Table 4.

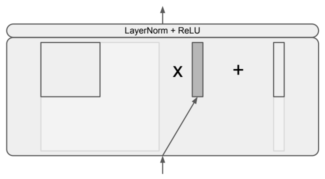


Figure 5: Illustration of a single layer where a subset of the weight matrix and bias vector are used, evaluating half of the neurons and reducing the operations down to approximately a quarter.

3.2.1 Subset of neurons

To encode multiple levels of detail within a unified representation, we propose using subsets of neurons as shown in Figure 4. For each level of detail, we assign a fixed number of neurons to evaluate, with increasing levels of detail using an increasing proportion of the neurons in the neural network. An illustration of the matrix subset operation applied to each layer is shown in Figure 5. For example, a single linear layer with 512 neurons will have a 512×512 weight matrix and a bias vector of size 512. For a low level of detail, we can assign a neuron subset size of 25% or 128 neurons. Then each linear layer would use the top-left 128×128 submatrix for the weights and the top 128 subvector for the bias. To keep the input and output dimensions the same, the first hidden layer uses every column of the weight matrix and the final output layer uses every row of the weight matrix. Unlike using a subset of layers, as is done with early exiting [5, 64], using a subset of neurons preserves the number of sequential non-linear activations between all levels of details. Since layer widths are typically larger than model depths, using subsets of neurons also allows more granular control over the quantity and quality of levels.

3.2.2 Training

With multiple levels of detail at varying resolutions, a modified training scheme is required to efficiently encode all levels of detail together. On one hand, generating all levels of detail at each iteration heavily slows down training as each iteration requires an independent forward pass through the network. On the other hand, selecting a single level of detail at each iteration or applying progressive training compromises the highest level of detail whose full weights would only get updated a fraction of the time.

We adopt an intermediate approach that focuses on training the highest level of detail. During training, each batch does a full forward pass through the entire network, producing pixels at the highest level of detail along with a forward pass at a randomly selected lower level of detail. The squared L2 losses for both the full detail and reduced detail are added together for a combined backward pass. By training at the highest level of detail, we ensure that every weight gets updated at each batch and that the highest level of detail is trained to the same number of iterations as a standard model. Specifically, given a batch of rays $\{\mathbf{r}_i\}_{i=1}^{b}$, corresponding ground truth colors $\{\{\mathbf{y}_i^c\}_{i=1}^{b}\}_{c=1}^{4}$ for four levels of details, and a random lower level of detail $k \in \{1, 2, 3\}$, our loss function is:

$$L = \frac{1}{b}\sum_{i=1}^{b}\left[\left\|f_4(\mathbf{r}_i) - \mathbf{y}_i^4\right\|_2^2 + \left\|f_k(\mathbf{r}_i) - \mathbf{y}_i^k\right\|_2^2\right]$$

As rays are sampled from the highest-resolution representation, each ray will not correspond to a unique pixel in the lower-resolution images in the image pyramid. Thus for lower levels of detail, corresponding colors are sampled using bilinear interpolation from the area downsampled training images.

3.3. Adaptive Rendering

Our proposed model allows dynamic levels of detail on a per-ray/per-pixel level or on a per-object level. As aliasing appears primarily at smaller scales, selecting the level of detail based on the distance to the viewer reduces aliasing and flickering for distant objects.

3.3.1 Distance-based Level of Detail

When rendering light fields from different distances, the space between rays in object space is proportional to the distance between the object and the camera. Due to this, aliasing and flickering artifacts can occur at large distances when the object appears small. By selecting the level of detail based on the distance of the object from the camera or viewer, aliasing and flickering can be reduced. With fewer pixels to render, rendering smaller objects involves fewer forward passes through the light field network. Hence, the amount of operations is typically proportional to the number of pixels. By rendering smaller representations at lower LODs, we further improve the performance as pixels from lower LODs only perform a forward pass using a subset of weights.

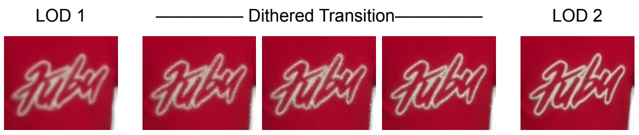


Figure 6: With per-pixel level of detail, dithering can be applied to transition between levels of detail.

3.3.2 Dithered Transitions

With per-pixel LOD, transitioning between levels can be achieved with dithered rendering. By increasing the fraction of pixels rendered at the new level of detail in each frame, dithering creates the appearance of a fading transition between levels as shown in Figure 6.

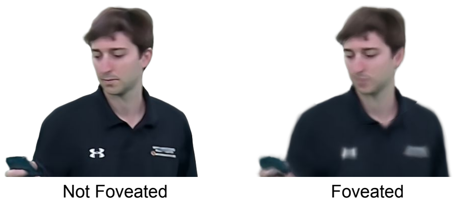


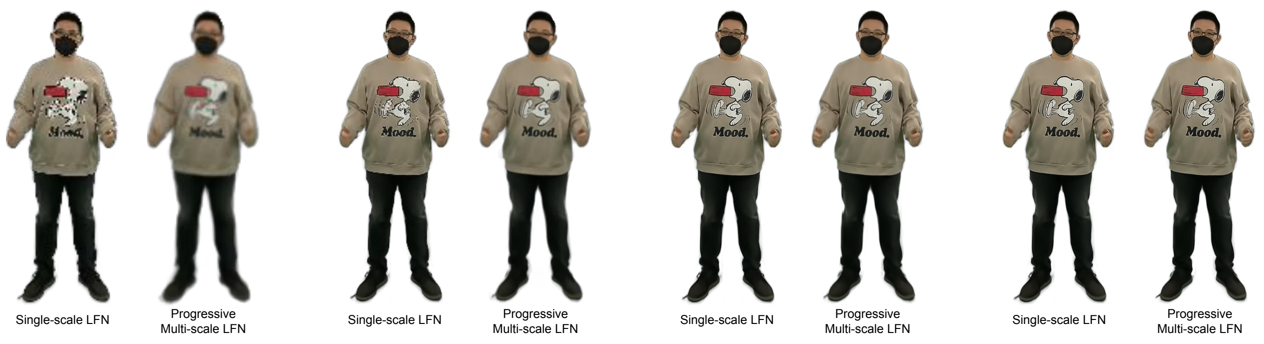
Figure 7: An example of foveated rendering from our progressive multi-scale LFN. Foveated rendering generates peripheral pixels at lower levels of detail to further improve performance in eye-tracked environments. In this example, foveation is centered over the left eye of the subject.

3.3.3 Foveated Rendering

In virtual and augmented reality applications where real-time gaze information is available through an eye-tracker, foveated rendering [29, 30, 22] can be applied to better focus rendering compute. As each pixel can be rendered at a separate level of detail, peripheral pixels can be rendered at a lower level of detail to improve the performance. Figure 7 shows an example of foveation from our progressive multi-scale LFN.

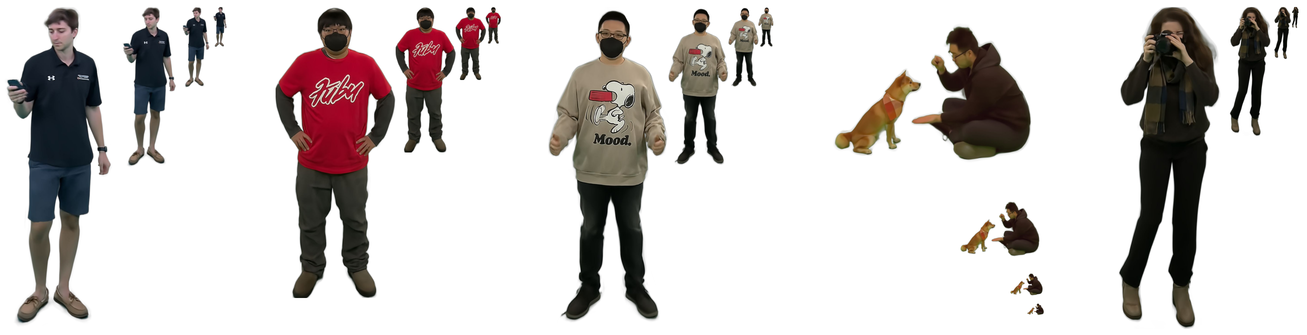
4. Experiments

To evaluate our progressive multi-scale LFN, we conduct experiments with four levels of detail. We first compare the quality when rendering at different scales from a standard single-scale LFN and our multi-scale LFN. We then perform an ablation comparing multiple LFNs trained at separate scales to our progressive network. Finally, we benchmark our multi-scale LFN to evaluate the speedup gained by utilizing fewer neurons when rendering at lower levels of detail. Additional details are in the supplementary material.


Single-scale LFN
Progressive Multi-scale LFN
Single-scale LFN
Progressive Multi-scale LFN
Single-scale LFN
Progressive Multi-scale LFN
Single-scale LFN
Progressive Multi-scale LFN

(a) Dataset C at 1/8 scale (b) Dataset C at 1/4 scale (c) Dataset C at 1/2 scale (d) Dataset C at full scale

Figure 8: Qualitative results of one of our datasets at four scales. Single-scale LFNs (left) trained at the full-resolution exhibit aliasing and flickering at lower scales. Our progressive multi-scale LFNs (right) encode all four scales into a single model and have reduced aliasing and flickering at lower scales.



(a) Dataset A (b) Dataset B (c) Dataset C (d) Dataset D (e) Dataset E

Figure 9: Scaled qualitative results from our progressive multi-scale LFNs at four levels of detail.

4.1. Experimental Setup

For our evaluation, we use datasets of real people consisting of 240 synchronized images from our capture studio. Images are captured at 4032×3040 resolution. Our datasets are processed with COLMAP [46, 47] to extract camera parameters and background matting [26] to focus on the foreground subject. In each dataset, images are split into groups of 216 training poses, 12 validation poses, and 12 test poses.

We quantitatively evaluate the encoding quality by computing the PSNR and SSIM metrics. Both metrics are computed on cropped images that tightly bound the subject to reduce the contribution of empty background pixels. We also measure the render time to determine the relative speedup at different levels of detail. Metrics are averaged across all images in the test split of each dataset.

When training LFNs, we use a batch size of 8192 rays with 67% of rays sampled from the foreground and 33% sampled from the background. For each epoch, we iterate through training images in batches of two images. In each image batch, we train on 50% of all foreground rays sampling rays across the two viewpoints. For our multi-scale progressive model and single-scale model, we train using 100 epochs. For our ablation LFNs at lower resolutions, we train until the validation PSNR matches that of our progressive multi-scale LFN with a limit of 1000 epochs. Each LFN is trained using a single NVIDIA RTX 2080 TI GPU.

Table 2: Model Parameters at Different Levels of Detail.

Level of Detail	1	2	3	4
Model Layers	10	10	10	10
Layer Width	128	256	384	512
Parameters	136,812	533,764	1,193,860	2,116,100
Full Size (MB)	0.518	2.036	4.554	8.072
Target Scale	1/8	1/4	1/2	1
Train Width	504	1008	2016	4032
Train Height	380	760	1520	3040

4.2. Rendering Quality

We first evaluate the rendering quality by comparing a standard single-scale LFN trained at the highest resolution with our multi-scale progressive LFN. For the single-scale LFN, we train a model with 10 layers and 512 neurons per hidden layer on each of our datasets and render them at multiple scales: $1/8$, $1/4$, $1/2$, and full scale. For our multi-scale progressive LFN, we train an equivalently sized model with four LODs corresponding to each of the target scales. The lowest LOD targets the $1/8$ scale and utilizes 128 neurons from each hidden layer. Each increasing LOD uses an additional 128 neurons. Model parameters for each LOD are laid out in Table 2. Note that our qualitative results are cropped to the subject and hence have a smaller resolution. Qualitative results are shown in Figure 8 with renders from both models placed side-by-side. Quantitative results are shown in Table 3.

At the full scale, both single-scale and multi-scale models are trained to produce full-resolution images. As a result, the quality of the renders from both models is visually similar as shown in Figure 8. At $1/2$ scale, the resolution is still sufficiently high so little to no aliasing can be seen in renders from either model. At $1/4$ scale, we begin to see significant aliasing around figure outlines and shape borders in the single-scale model. This is especially evident in Figure 8b where the outlines in the cartoon graphic have an inconsistent thickness. Our multi-scale model has much less aliasing with the trade-off of having more blurriness. At the lowest level of detail, we see severe aliasing along the fine textures as well as along the borders of the object in the single-scale model. With a moving camera, the aliasing seen at the two lowest scales appears as flickering artifacts.

Table 3: Average Rendering Quality Over All Datasets.

Model	LOD 1	LOD 2	LOD 3	LOD 4
Single-scale LFN	26.95	28.05	28.21	27.75
Multiple LFNs	29.13	29.88	29.27	27.75
Multi-scale LFN	29.37	29.88	29.01	28.12

(a) PSNR (dB) at 1/8, 1/4, 1/2, and 1/1 scale.

Model	LOD 1	LOD 2	LOD 3	LOD 4
Single-scale LFN	0.8584	0.8662	0.8527	0.8480
Multiple LFNs	0.8133	0.8572	0.8532	0.8480
Multi-scale LFN	0.8834	0.8819	0.8626	0.8570

(b) SSIM at 1/8, 1/4, 1/2, and 1/1 scale.

4.3. Progressive Model Ablation

We next perform an ablation experiment to determine the benefits and drawbacks of our progressive representation. For a non-progressive comparison, we represent each scale of a multi-scale light field using a separate LFN. Each LFN is trained using images downscaled to the appropriate resolution. This ablation helps determine the training overhead our progressive LFNs encounter by compressing four resolutions into a single model and how the rendering quality compares to having separate models.

Table 4: Multiple LFNs *vs* Progressive Multi-scale LFN.

Model	LOD 1	LOD 2	LOD 3	LOD 4	Total
Multiple LFNs	0.518	2.036	4.554	8.072	15.180
Multi-scale LFN	8.072				8.072

(a) Model Size (MB).

Model	LOD 1	LOD 2	LOD 3	LOD 4	Total
Multiple LFNs	3.51	6.36	4.09	11.35	25.31
Multi-scale LFN	17.78				17.78

(b) Average Training Time Over All Datasets (hours).

Progressive and non-progressive model sizes and training times are shown in Table 4. In terms of model size, using multiple LFNs adds up to 15 MB compared to 8 MB for our progressive model. This 47% savings could allow storing or streaming of more light fields where model size is a concern. For the training time, we observe that training a multi-scale network takes on average 17.78 hours. Additional offline training time for our multi-scale network compared to training a single standard LFN is expected since each training iteration involves two forward passes. Encoding a multi-scale light field using multiple independent LFNs requires training each network separately, totaling 25.31 hours. Despite the higher compression achieved by our progressive multi-scale LFN, we observe little to no training overhead. On average, training our multi-scale progressive LFN saves 30% of training time compared to naively training multiple light field networks at different resolutions.

Quantitative PSNR and SSIM quality metrics are shown in Table 3. We observe that utilizing multiple LFNs to encode each scale of each light field yields better PSNR results compared to rendering from a single-scale LFN. However, utilizing multiple LFNs increases the total model size. Our multi-scale LFN achieves the superior PSNR and SSIM results between these two setups without increasing the total model size. In an on-demand streaming scenario, using only a single-scale model would incur visual artifacts and unnecessary computation at lower scales. Streaming separate models resolves these issues but requires more bandwidth. Neither option is ideal for battery life in mobile devices. Our multi-scale model alleviates the aliasing and flickering artifacts without incurring the drawbacks of storing and transmitting multiple models.

4.4. Training Ablation

To evaluate our training strategy, we perform two ablation experiments. First, we compare our strategy to a

Table 5: Training Ablation Average Rendering Quality Results.

Ablation	LOD 1	LOD 2	LOD 3	LOD 4
Progressive Training	26.30	30.33	29.08	28.10
108 Training Views	29.04	29.63	28.93	27.98
Ours	29.37	29.88	29.01	28.12

(a) PSNR (dB) at 1/8, 1/4, 1/2, and 1/1 scale.

Model	LOD 1	LOD 2	LOD 3	LOD 4
Progressive Training	0.7947	0.8719	0.8447	0.8390
108 Training Views	0.8765	0.8771	0.8604	0.8566
Ours	0.8834	0.8819	0.8626	0.8570

(b) SSIM at 1/8, 1/4, 1/2, and 1/1 scale.

coarse-to-fine progressive training strategy where lower levels of detail are trained and then frozen as higher levels are trained. Progressive training takes on average 24.20 hours to train. Second, we use half the number of training views to evaluate how the quality degrades with fewer training views. Both experiments use our progressive multi-scale model architecture. Our results are shown in Table 5 with additional details in the supplementary material.

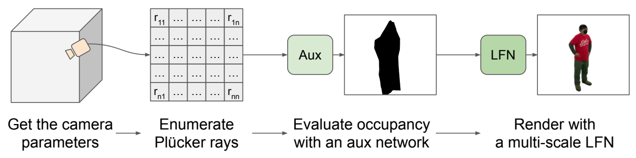

r_{11}
r_{1n}
r_{n1}
r_{nn}
Aux
LFN
Get the camera parameters
Enumerate Plücker rays
Evaluate occupancy with an aux network
Render with a multi-scale LFN

Figure 10: Overview of our rendering pipeline which utilizes an auxiliary network to skip evaluation of empty rays.

Table 6: Average Model Rendering Performance for our Multi-scale LFN (milliseconds per frame).

LOD 1	LOD 2	LOD 3	LOD 4
3.7	3.9	4.7	5.9

(a) Rendering each LOD at 1/8 scale.

LOD 1	LOD 2	LOD 3	LOD 4
4	11	58	305

(b) Rendering at 1/8, 1/4, 1/2, and 1/1 scale.

4.5. Level of Detail Rendering Speedup

We evaluate the rendering speed of our progressive model at different levels of detail to determine the observable speedup from rendering with the appropriate level of detail. For optimal rendering performance, we render using half-precision floating-point values and skip empty rays using an auxiliary neural network encoding only the ray occupancy as shown in Figure 10. Our auxiliary network is made up of three layers with a width of 16 neurons per hidden layer. Evaluation is performed by rendering rays from all training poses with intrinsics scaled to the corresponding resolution as shown in Table 2. Average rendering time results are shown in Table 6. Rendering times for lower LODs (1/8 scale) from our multi-scale network are reduced from ~ 5.9 to ~ 3.7 msec.

5. Discussion

The primary contribution of our paper is the identification of a representation more suitable for on-demand streaming. This is achieved by enhancing light field networks, which support real-time rendering, to progressively support multiple levels of detail at different resolutions. Our representation inherits some of the limitations of LFNs such as long training times and worse view-synthesis quality compared to NeRF-based models. Since our method does not support continuous LODs, we require the user to decide the LODs ahead of time. For intermediate resolutions, rendering to the nearest LOD is sufficient to avoid aliasing and flickering. While our light field datasets could also be rendered with classical techniques [21, 60, 39], doing so over the internet would not be as bandwidth-efficient. Future work may combine our multi-scale light field networks with PRIF [15] to encode color and geometry simultaneously or with VIINTER [14] for dynamic light fields.

6. Conclusion

In this paper, we propose a method to encode multi-scale light field networks targeted for streaming. Multi-scale LFNs encode multiple levels of details at different scales to reduce aliasing and flickering at lower resolutions. Our multi-scale LFN features a progressive representation that encodes all levels of details into a single network using subsets of network weights. With our method, light field networks can be progressively downloaded and rendered based on the desired level of detail for real-time streaming of 6DOF content. We hope progressive multi-scale LFNs will help improve the real-time streaming of photorealistic characters and objects to real-time 3D desktop, AR, and VR applications [23, 12].

Acknowledgments We would like to thank Jon Heagerty, Sida Li, Barbara Brawn-Cinani, and Maria Herd for developing our light field datasets as well as the anonymous reviewers for the valuable comments on the manuscript. This work has been supported in part by the NSF Grants 18-23321 and 21-37229, and the State of Maryland's MPower initiative. Any opinions, findings, conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the research sponsors.

References

[1] Benjamin Attal, Jia-Bin Huang, Michael Zollhöfer, Johannes Kopf, and Changil Kim. Learning neural light fields with ray-space embedding networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[2] Benjamin Attal, Selena Ling, Aaron Gokaslan, Christian Richardt, and James Tompkin. MatryODShka: Real-time 6DoF video view synthesis using multi-sphere images. In *European Conference on Computer Vision (ECCV)*, Aug. 2020. 1

[3] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields, 2021. 2

[4] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 2

[5] Tolga Bolukbasi, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama. Adaptive neural networks for efficient inference. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 527–536. PMLR, 06–11 Aug 2017. 3, 4

[6] Paramanand Chandramouli, Hendrik Sommerhoff, and Andreas Kolb. Light field implicit representation for flexible resolution reconstruction, 2021. 3

[7] Zhang Chen, Yinda Zhang, Kyle Genova, Sean Fanello, Sofien Bouaziz, Christian Häne, Ruofei Du, Cem Keskin, Thomas Funkhouser, and Danhang Tang. Multiresolution deep implicit functions for 3d shape representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13087–13096, October 2021. 3

[8] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7911–7920, June 2021. 2

[9] Junwoo Cho, Seungtae Nam, Daniel Rho, Jong Hwan Ko, and Eunbyung Park. Streamable neural fields. In *ECCV*, 2022. 3

[10] Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi. Fusion4D: Real-Time Performance Capture of Challenging Scenes. *ACM Trans. Graph.*, 35(4), jul 2016. 1

[11] Ruofei Du, Ming Chuang, Wayne Chang, Hugues Hoppe, and Amitabh Varshney. Montage4D: Interactive seamless fusion of multiview video textures. In *Proceedings of ACM Interactive 3D Graphics (I3D) 2018*, 2018. 1

[12] Ruofei Du, David Li, and Amitabh Varshney. Geollery: A mixed reality social media platform. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–13, New York, NY, USA, 2019. Association for Computing Machinery. 8

[13] Rizal Fathony, Anit Kumar Sahu, Devin Willmott, and J. Zico Kolter. Multiplicative filter networks. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 3

[14] Brandon Feng, Susmija Jabbireddy, and Amitabh Varshney. Viinter: View interpolation with implicit neural representations of images. In *SIGGRAPH ASIA*, 2022. 8

[15] Brandon Feng, Yinda Zhang, Danhang Tang, Ruofei Du, and Amitabh Varshney. PRIF: Primary Ray-Based Implicit Function. In *European Conference on Computer Vision*, ECCV. Springer, 2022. 8

[16] Brandon Yushan Feng and Amitabh Varshney. SIGNET: Efficient neural representation for light fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14224–14233, October 2021. 3

[17] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. *arXiv preprint arXiv:2103.10380*, 2021. 2, 3

[18] Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. *ICCV*, 2021. 3

[19] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5885–5894, October 2021. 2

[20] Zoe Landgraf, Alexander Sorkine Hornung, and Ricardo Silveira Cabral. PINs: Progressive Implicit Networks for Multi-Scale Neural Representations. In *International Conference on Machine Learning (ICML 2022)*, 2022. 3

[21] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, page 31–42, New York, NY, USA, 1996. Association for Computing Machinery. 3, 8

[22] David Li, Ruofei Du, Adharsh Babu, Camelia D. Brumar, and Amitabh Varshney. A log-rectilinear transformation for foveated 360-degree video streaming. *IEEE Transactions on Visualization and Computer Graphics*, 27(5):2638–2647, 2021. 5

[23] David Li, Eric Lee, Elijah Schwelling, Mason G. Quick, Patrick Meyers, Ruofei Du, and Amitabh Varshney. Meteovis: Visualizing meteorological events in virtual reality. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI EA '20, page 1–9, New York, NY, USA, 2020. Association for Computing Machinery. 8

[24] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 2

[25] Zhong Li, Liangchen Song, Celong Liu, Junsong Yuan, and Yi Xu. NeuLF: Efficient Novel View Synthesis with Neural 4D Light Field. In Abhijeet Ghosh and Li-Yi Wei, editors,

Eurographics Symposium on Rendering. The Eurographics Association, 2022. 3

[26] Shanchuan Lin, Andrey Ryabtsev, Soumyadip Sengupta, Brian Curless, Steve Seitz, and Ira Kemelmacher-Shlizerman. Real-time high-resolution background matting. *arXiv*, pages arXiv–2012, 2020. 6

[27] David B. Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. BACON: Band-limited coordinate networks for multiscale scene representation. *arXiv preprint arXiv:2112.04645*, 2021. 2, 3

[28] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Haotang Deng, and Qi Ju. FastBERT: a self-distilling bert with adaptive inference time. In *Proceedings of ACL 2020*, 2020. 3

[29] Xiaoxu Meng, Ruofei Du, Joseph JaJa, and Amitabh Varshney. 3D-Kernel Foveated Rendering for Light Fields. *IEEE Transactions on Visualization and Computer Graphics*, 27(8):3350–3360, Mar. 2020. 5

[30] Xiaoxu Meng, Ruofei Du, Matthias Zwicker, and Amitabh Varshney. Kernel foveated rendering. *Proc. ACM Comput. Graph. Interact. Tech.*, 1(1):5:1–5:20, July 2018. 5

[31] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2

[32] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv:2201.05989*, Jan. 2022. 2

[33] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H. Mueller, Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, and Markus Steinberger. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum*, 40(4), 2021. 2

[34] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs, 2021. 2

[35] Michael Niemeyer and Andreas Geiger. Campari: Camera-aware decomposed generative neural radiance fields. In *2021 International Conference on 3D Vision (3DV)*, pages 951–961, 2021. 2

[36] Julian Ost, Issam Laradji, Alejandro Newell, Yuval Bahat, and Felix Heide. Neural point light fields. *CoRR*, abs/2112.01473, 2021. 3

[37] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021. 2

[38] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Trans. Graph.*, 40(6), dec 2021. 2

[39] Ingmar Peter and Wolfgang Straßer. The wavelet stream: Interactive multi resolution light field rendering. In Steven J. Gortler and Karol Myszkowski, editors, *Rendering Techniques 2001*, pages 127–138, Vienna, 2001. Springer Vienna. 8

[40] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 2

[41] Daniel Rebain, Mark Matthews, Kwang Moo Yi, Dmitry Lagun, and Andrea Tagliasacchi. Lolnerf: Learn from one look, 2021. 2

[42] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. KiloNeRF: Speeding up neural radiance fields with thousands of tiny mlps. *arXiv preprint arXiv:2103.13744*, 2021. 2

[43] Konstantinos Rematas, Andrew Liu, Pratul P. Srinivasan, Jonathan T. Barron, Andrea Tagliasacchi, Thomas Funkhouser, and Vittorio Ferrari. Urban radiance fields, 2021. 2

[44] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks, 2016. 3

[45] Vishwanath Saragadam, Jasper Tan, Guha Balakrishnan, Richard G. Baraniuk, and Ashok Veeraraghavan. MINER: Multiscale Implicit Neural Representations. In *European Conference on Computer Vision*, ECCV. Springer, 2022. 3

[46] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6

[47] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 6

[48] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *arXiv preprint arXiv:2007.02442*, 2020. 2

[49] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, page 231–242, New York, NY, USA, 1998. Association for Computing Machinery. 1

[50] Jianxiong Shen, Adria Ruiz, Antonio Agudo, and Francesc Moreno-Noguer. Stochastic neural radiance fields: Quantifying uncertainty in implicit 3d representations. In *2021 International Conference on 3D Vision (3DV)*, pages 972–981, 2021. 2

[51] Vincent Sitzmann, Semon Rezchikov, William T. Freeman, Joshua B. Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. In *arXiv*, 2021. 1, 3

[52] Towaki Takikawa, Alex Evans, Jonathan Tremblay, Thomas Müller, Morgan McGuire, Alec Jacobson, and Sanja Fidler. Variable bitrate neual fields. In *SIGGRAPH 2022*. ACM, 2022. 3

[53] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vi-*

sion and Pattern Recognition (CVPR), pages 11358–11367, June 2021. 3

[54] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, Tomas Simon, Christian Theobalt, Matthias Niessner, Jonathan T. Barron, Gordon Wetzstein, Michael Zollhoefer, and Vladislav Golyanik. Advances in neural rendering, 2021. 2

[55] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs, 2021. 2

[56] Liao Wang, Jiakai Zhang, Xinhang Liu, Fuqiang Zhao, Yanshun Zhang, Yingliang Zhang, Minye Wu, Lan Xu, and Jingyi Yu. Fourier plenoctrees for dynamic radiance field rendering in real-time, 2022. 2

[57] Lance Williams. Pyramidal parametrics. *SIGGRAPH Comput. Graph.*, 17(3):1–11, jul 1983. 3

[58] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9421–9431, 2021. 2

[59] Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering. In *The European Conference on Computer Vision (ECCV)*, 2022. 2

[60] Zhaolin Xiao, Qing Wang, Guoqing Zhou, and Jingyi Yu. Aliasing detection and reduction in plenoptic imaging. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 8

[61] Christopher Xie, Keunhong Park, Ricardo Martin-Brualla, and Matthew Brown. Fig-nerf: Figure-ground neural radiance fields for 3d object category modelling. In *2021 International Conference on 3D Vision (3DV)*, pages 962–971, 2021. 2

[62] Guo-Wei Yang, Wen-Yang Zhou, Hao-Yang Peng, Dun Liang, Tai-Jiang Mu, and Shi-Min Hu. Recursive-nerf: An efficient and dynamically growing nerf, 2021. 3

[63] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution adaptive networks for efficient inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[64] Hyunho Yeo, Youngmok Jung, Jaehong Kim, Jinwoo Shin, and Dongsu Han. Neural adaptive content-aware internet video delivery. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 645–661, Carlsbad, CA, Oct. 2018. USENIX Association. 3, 4

[65] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks, 2021. 2, 3

[66] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *ICCV*, 2021. 2, 3

[67] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images, 2020. 2

[68] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. In *International Conference on Learning Representations*, 2019. 3

[69] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018. 1

[70] Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. Bert loses patience: Fast and robust inference with early exit. In *Advances in Neural Information Processing Systems*, volume 33, pages 18330–18341. Curran Associates, Inc., 2020. 3