Neural Subspaces for Light Fields

Brandon Yushan Feng and Amitabh Varshney, *Fellow, IEEE*

Abstract—We introduce a framework for compactly representing light field content with the novel concept of neural subspaces. While the recently proposed neural light field representation achieves great compression results by encoding a light field into a single neural network, the unified design is not optimized for the composite structures exhibited in light fields. Moreover, encoding every part of the light field into one network is not ideal for applications that require rapid transmission and decoding. We recognize this problem's connection to subspace learning. We present a method that uses several small neural networks, specializing in learning the neural subspace for a particular light field segment. Moreover, we propose an adaptive weight sharing strategy among those small networks, improving parameter efficiency. In effect, this strategy enables a concerted way to track the similarity among nearby neural subspaces by leveraging the layered structure of neural networks. Furthermore, we develop a soft-classification technique to enhance the color prediction accuracy of neural representations. Our experimental results show that our method better reconstructs the light field than previous methods on various light field scenes. We further demonstrate its successful deployment on encoding light fields with irregular viewpoint layout and dynamic scene content.

Index Terms—Light fields, volumetric videos, neural network, compression.

1 Introduction

Light field content can display scenes at unprecedented levels of realism and immersion. However, the vast amount of data in high quality light fields exceeds the storage, transmission, and interactive rendering capabilities of current graphics pipelines. Previous efforts to compactly represent light field content were mostly built on established concepts in image and video compression. Nonetheless, they still fall short of enabling large-scale, real-time dissemination of high-resolution light field content.

Recently, neural representation of light fields [1] has emerged as a viable alternative to conventional compression methods for efficiently storing light fields with high fidelity. For a given light field, a neural network is trained to learn the mapping function between each light field pixel's coordinates and color values. The key to achieving high accuracy of this neural network is to transform the input coordinates with a set of basis functions so that the network can more easily capture the high-frequency details. For instance, Fourier basis and Gegenbauer polynomials have been shown to be effective in transforming light field coordinates for a more accurate network [1], [2], [3], [4]. After training, multiple viewpoints from a light field scene (hundreds of megabytes) are compactly encoded into such a neural network's weights (a few megabytes). These weights are the only necessary information for storage and streaming. Moreover, this trained neural network can be evaluated at arbitrary viewpoints, thus quickly enabling interpolation for a smooth viewing experience.

This unified design of training a single neural network to cover the entire light field has simplistic appeal as a concept. However, it is not ideal in practical scenarios that emphasize efficient transmission and rendering. Such scenarios generally involve only rendering a subset of light field viewpoints. Therefore we incur unnecessary costs to transmit and evaluate a single neural network that contains other unrelated views. This question naturally arises: how can we improve the neural representation of light fields to accelerate decoding/rendering at a particular viewpoint without sacrificing visual quality?

In signal processing research, subspace learning is a well-known concept for dimensionality reduction of high-dimensional data. Its most notable example involves using principal component analysis (PCA) to find a small number of orthogonal basis vectors (principal components) with most data variability. Naively constructing a single subspace without adequately considering the structural attributes of the data would be suboptimal. If the data contains patterns that evolve over a long period, or the data are simultaneously captured by multiple sensors at different locations, constructing a global subspace may require many parameters to adequately represent the data, which would defeat the purpose of dimensionality reduction. It would be more sensible to divide such inherently composite data into multiple local subspaces such that each corresponds to a small segment or region of the data. The subsequent problem of tracking the changing subspace of data with such composite structures is often referred to as *subspace tracking*.

We first recognize the conceptual connection between the neural representation of light fields and the construct of subspace learning. When a neural network is trained to encode a given light field scene, it reduces dimensionality on the high-dimensional light field data. Specifically, the neural network succinctly describes the pixel coordinate to color mapping relationship. The fact that basis functions such as the Fourier basis or Gegenbauer polynomials improve network accuracy is deeply connected to how an appropriate basis function forms a more suitable subspace to analyze the data. Effectively, this network is trained to parametrize through its weights a *neural subspace* of the light field data.

Moreover, in the initial neural representation of light fields, one global neural subspace is learned for every part

• *B. Feng and A. Varshney are with the Department of Computer Science, University of Maryland, College Park, MD, 20770. E-mail: yfeng97@umd.edu, varshney@umd.edu*

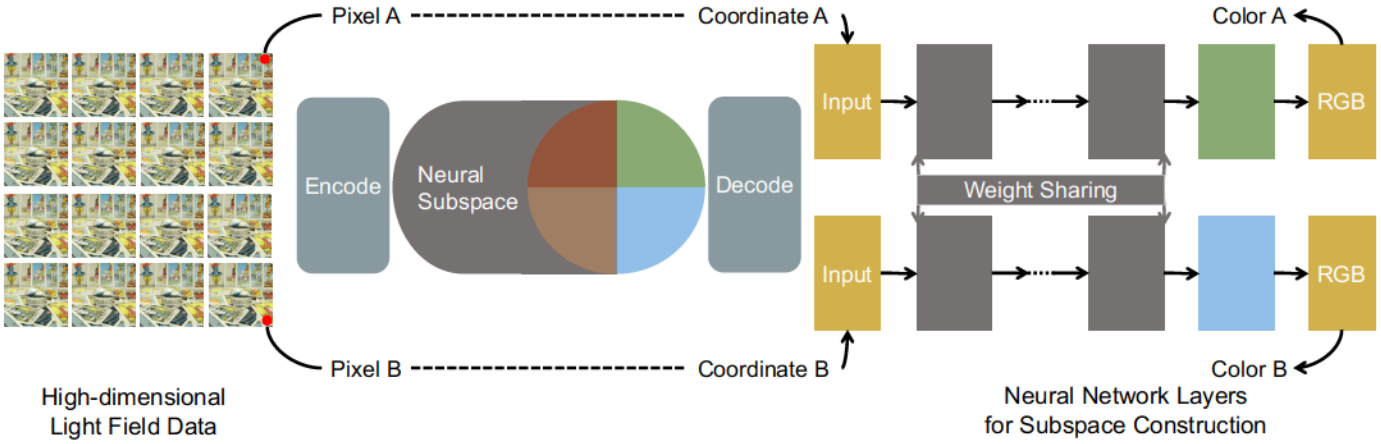

Fig. 1: We illustrate our proposed concept of neural subspaces for light fields. Given a light field scene, we divide it into multiple local segments and construct a neural subspace for each segment. This figure shows an example of a light field partitioned into four segments in space; the same concept could be similarly applied to light field videos with segments across different periods. Each neural subspace of a light field is parameterized by a multi-layer neural network, which outputs the color value given its coordinates as input. The neural subspace construction is equivalent to training the network parameters to learn accurate coordinate-to-color mappings. We further introduce an adaptive weight sharing strategy, which implicitly utilizes the similarity among nearby subspaces and reduces the total number of parameters to represent the entire light field. Compared to previous methods, this new framework achieves a higher light field compression rate for storage and streaming without sacrificing reconstruction accuracy.

of a light field scene, leading to potential inefficiencies in scenarios that concern only a subset of the scene. This is similar to the issues that motivate subspace learning for composite data, prompting us to construct multiple local neural subspaces for light field data. Indeed, it is not technically necessary to treat light fields as one unified entity and encode the viewpoints altogether - light fields can be regarded as a composite collection of local segments. Such a perspective is meaningful in practice since only a subset of the light field might be relevant at a particular moment for streaming or rendering. Moreover, it is plausible that networks specialized in local segments could represent the data more accurately than a global network while using fewer parameters overall.

This paper presents a novel approach to constructing neural subspaces for light fields. With experimental results, we demonstrate improving parameter efficiency and reconstruction accuracy under this novel perspective. Unlike the initial attempt that trains one global network for the entire light field, we train a set of local neural networks that encode only a subset of viewpoints. As each local network specializes in a particular region, this specialization permits smaller networks without sacrificing accuracy, if not further boosting accuracy. Next, recognizing the similarity among nearby subspaces, we propose a weight-sharing strategy for those local networks to enhance overall parameter efficiency while maintaining network capacity within each subspace. Effectively, this proposed strategy achieves the tracking of implicit neural subspaces. Furthermore, we introduced a new soft-classification technique for neural representations to produce the RGB color predictions, which further improves the reconstruction accuracy. Experimental results on various light field scenes indicate that the proposed framework leads to better efficiency and accuracy than the original neural representation of light fields and a range of previous methods.

In summary, our contributions are as follows:

- We introduce neural subspaces for light fields, a general framework for improving the compactness of neural representation of light fields via subspace learning.
- We propose a soft-classification technique to predict the pixel color values, which further improves the reconstruction quality.
- We propose an adaptive weight sharing strategy to improve parameter efficiency by leveraging the layered structure of neural networks.
- We achieve results superior to the state-of-the-art compression on various light fields with our proposed light field encoding framework.

1.1 Scope

The work is focused on accurately encoding and decoding pixels captured in a light field. The related but orthogonal tasks of novel view synthesis or 3D volume reconstruction is beyond the scope of this paper.

2 Related Work

2.1 Light Field Compression for Streaming

Traditional light field compression relies on classic coding strategies that typically involve analytical basis functions such as the Fourier basis and wavelets. Prior research has augmented this analytical approach with disparity [5], [6], [7] and geometry information [8]. Some sophisticated applications of light field video [9], [10] also integrate motion prediction [11] and build on existing video codec algorithms

such as HEVC (H.265) [12] and VP9 [13]. More recently, Le Pendu *et al.* [14] propose a Fourier Disparity Layer representation specific for sub-aperture light fields, which allows upsampling [15] and compression [16], [17] in the Fourier domain.

A different approach towards light field compression involves learning a dictionary of basis functions. This idea is inspired by progress in sparse coding from machine learning, where dictionaries learned with data-driven algorithms have been shown to outperform analytical basis functions [18], [19], [20], [21]. However, the dictionaries learned with conventional algorithms such as K-SVD [22] still contain too much redundancy and have a high storage cost. The state-of-the-art dictionary-based method [23], [24] for light field compression improves this approach by learning an ensemble of orthogonal dictionaries with a novel pre-clustering strategy.

Recently, a hierarchical tree structure of light fields has been introduced [25], where a root node stores a representative key view image for a given light field, and the child nodes store the sparse residuals (difference from their parent node's image) for each remaining image. Compression is achieved by only encoding the sparse residuals of those children's views instead of their full-color values. Specifically, they use the standard JPEG2000 to encode the representative key view and Bounded Integer Sequence Encoding [26] for the sparse residual views. Pratapa *et al.* [11] later extend this hierarchical method to incorporate the motion prediction and compensation concept widely used in traditional video encoding techniques.

While previous research has made substantial progress, most methods remain conceptual extensions to traditional image compression methods, which independently treat each image patch or camera view. Such methods essentially achieve compression by taking advantage of the redundancy within each patch or viewpoint without exploring potentially more optimal strategies that further utilize the redundancy between views or patches and between different time steps in the case of dynamic videos.

This paper presents a method based on the neural subspace that improves the compression rate through an adaptive weight sharing strategy. Specifically, we utilize the implicit redundancy between viewpoints or time steps to explicitly reduce the number of parameters to represent the entire light field scene.

2.2 Neural Light Field

Recent research [2], [3], [4] has shown the potential of using coordinate-input MLP networks to represent multidimensional data such as audio signals, images, videos, volumes, signed distance functions, and wave equations. Fourier-inspired techniques are the key to their breakthroughs in accurately approximating content with high-frequency details. The neural radiance field (NeRF) network [4] achieves state-of-the-art free viewpoint synthesis on static scenes by transforming the input coordinates with cosine and sine functions. Tancik *et al.* [3] further analyze this transform strategy, which is closely related to the Fourier basis functions, and show its effectiveness on 2D images. Concurrently, SIREN networks [2] replace the ReLU activation in conventional MLPs with a sine activation, also achieving accurate coordinate-to-color mappings for images and videos.

While it is straightforward to re-purpose those Fourier-inspired MLPs into light field representation by setting light field coordinates as the network input, they achieve less than ideal results [1]. Feng *et al.* introduce SIGNET, an MLP that encodes light field with high fidelity through a novel coordinate transformation strategy based on the Gegenbauer polynomials. They further demonstrate the compression capability of such a neural representation of light fields and show SIGNET's superior performance over the state-of-the-art compression methods based on sparse dictionaries.

This paper extends the initial success of neural representation of light fields and shows that its formation generalizes to learning a concept called neural subspace. Moreover, we show how the concept of subspace learning reveals the inefficiencies of not adapting to the different compositions of light field data. We then present how our techniques significantly optimize the parameter efficiency compared to using the original architectures of SIGNET.

Several recent works closely relate to light field representations with neural networks, but they focus on different problems than this paper. Kalantari et al. [27] and Bemana et al. [28] use neural networks to interpolate between nearby light field viewpoints, but the networks do not represent the known light field images themselves. Recent developments of neural light fields have also achieve successes in modeling appearance [29] and geometry [30] in unbounded 3D scenes, as well as view interpolation without geometric structures [31]. In the context of video compression without considering multi-view redundancies, implicit neural representations [32] have also been shown to outperform established neural video codecs.

2.3 Subspace Learning and Tracking

Subspace learning and tracking have been instrumental for signal processing and computer vision applications. In the traditional settings for learning subspaces from multidimensional data, the purpose is to detect meaningful patterns or correlations using a few parameters constructing a reduced number of dimensions. This technique is useful for applications such as object tracking from radar or video [33], [34], image denoising [35], and video background removal [36]. Since these tasks generally only require a coarse representation of the data points, the traditional subspace learning or dimension reduction methods are sufficient and have led to proven successes. However, the problem of constructing neural light fields requires decoding each light field pixel as accurately as possible, which is hard to achieve by traditional subspace learning methods, which only provide a rough estimation of the general patterns in the data.

We propose a framework that utilizes the expressive power of neural networks to construct a highly accurate and condensed representation of the subspace of our light field data. Compared to previous methods for light field compression, we tackle this problem from a different perspective based on the newly proposed neural representation paradigm. We break away from previous methods that use various basis functions to construct an analytically defined subspace. Instead, we propose learning a neural subspace,

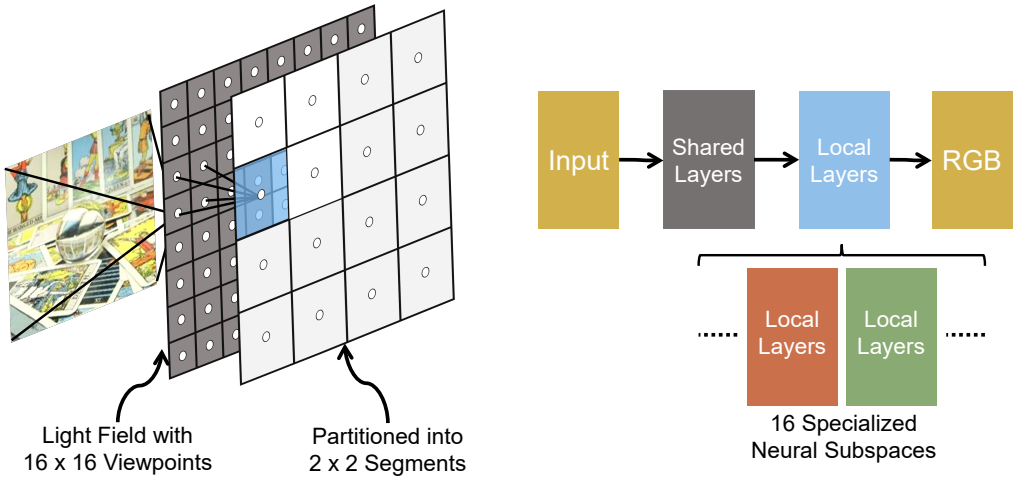

Fig. 2: Illustration of our proposed weight sharing strategy for a classic 4D light field. The light field is partitioned into segments, in this example 16 segments, each containing 2×2 viewpoints. We construct a neural subspace for each segment that summarizes its pixel-to-color mapping relationship. Each subspace shares a set of network layers while possessing its local layers that enable network specialization on its corresponding data segment.

which is realized by multiplying the input basis functions through a sequence of matrices with non-linear activations (MLP network), whose parameters (network weights) are adaptively learned for each light field scene. More importantly, we introduce a strategy to divide the neural subspace into segments properly. Then, starting from the learned neural subspace at one segment, we devise a method to update it for another segment without re-learning every parameter from scratch. This specialization allows us to downsize the total representation and the memory required to decode each pixel.

3 METHOD

3.1 Neural Light Fields with MLP

A neural light field representation [1] uses an N-layer neural network (MLP) to encode a light field (sub-aperture in the original case). Formally, it learns the following function $\mathcal{F}$:

$$\mathcal{F}(u,v,x,y) = \phi_L \circ \phi_{L-1} \circ ... \circ \phi_1([\Theta_i(u,v,x,y)]_{i=1}^{N}) \quad (1)$$

Here, (u,v,x,y) stands for the coordinate of a pixel within a sub-aperture light field, Θ_i stands for the orthogonal basis functions used to transform the coordinates, ϕ_l stands for the l-th layer of the neural network with a weight matrix W_l, bias vector b_l, and an activation function $\sigma(x) = sin(x)$. The output from each layer is $\phi_l(x) = \sigma(W_l x + b_l)$.

The MLP predicts the pixel color separately as $\mathcal{F}(u,v,x,y) = \tilde{\mathbf{c}}_p$. The training objective directly minimizes the regression error between the predicted color $\mathbf{c}_p^{Pred}$ and ground truth $\mathbf{c}_p$ across all known pixels in a light field:

$$\mathcal{L}_{Regress} = \sum_p \|\tilde{\mathbf{c}}_p - \mathbf{c}_p\|^2. \quad (2)$$

3.2 Constructing Light Field Segments

A global MLP is responsible for learning the coordinate-to-color mapping for all the pixels in a sub-aperture light field in the original neural representations for light fields. As discussed in Section 1, such a unified construct would be suboptimal in other camera configurations. Instead, we first partition a given light field into different segments in our method. For each segment's content, we then train an MLP to encode its coordinate-to-color mapping $\mathcal{F}$.

There are two aspects in which we could partition a given light field: space and time. To partition a light field in space means that we group camera viewpoints into separate clusters. To partition a light field in time means dividing a light field video into separate time intervals. In prior subspace learning and tracking methods, this partitioning strategy is key to achieving efficient dimension reduction of high-dimensional data. In our case of light field encoding, we conjecture that this partitioning strategy would significantly improve the parameter efficiency of light fields compared to SIGNET.

For example, consider a sub-aperture static light field with 16×16 viewpoints. Suppose we train an MLP that specializes in the particular segment of the top-left 4×4 viewpoints. In that case, it could require much smaller weight matrices to achieve the same reconstruction accuracy compared to training a global MLP for all 16×16 camera viewpoints. In the case of light field videos, we could similarly extend the partition strategy to divide the temporal dimension into several time intervals.

3.3 Adaptive Weight Sharing in MLP

It is not hard to imagine that if we partition a large light field into segments, we could use a much smaller MLP to encode each segment because it now needs to encode a simpler neural subspace. Nonetheless, is that the best we could do to make the neural subspaces as compact as possible?

A distinctive feature of the parametrization of the neural subspace is that each neural network contains a series of layers. Suppose every network contains N layers, and we partition the light field into D segments. Then if we naively

Scene	Method	PSNR(↑)	SSIM(↑)	EncMB(↓)	DecMB(↓)
Lego 909 MB	AMDE	40.90	0.973	29.3	29.3
	KSVD	38.39	0.960	29.3	29.3
	SIGNET	41.26	0.976	9.0	9.0
	Ours	41.95	0.982	22.9	2.2
Tarot 909 MB	AMDE	38.54	0.973	44.2	44.2
	KSVD	38.81	0.980	44.3	44.3
	SIGNET	37.47	0.975	9.0	9.0
	Ours	38.21	0.975	22.9	2.2
Bracelet 568 MB	AMDE	39.90	0.980	18.1	18.1
	KSVD	36.73	0.973	18.1	18.1
	SIGNET	38.70	0.973	12.0	12.0
	Ours	39.64	0.985	22.9	2.2

TABLE 1: **Results on Sub-aperture Light Fields**. We compare to previous methods AMDE [23], KSVD [22], and SIGNET [1]. *DecMB* is the memory (in MB) required in streaming to decode any frame, and *EncMB* is the memory (in MB) required to encode the entire light field in storage.

train an MLP for each segment, we would need to learn and store ND weight matrices for those layers.

We observe that this approach leads to possible parameter redundancy. More concretely, although each MLP is constructing a different neural subspace, nearby subspaces inevitably contain similar patterns. The redundancy occurs when we use too many parameters to represent those similar patterns. Therefore, it makes sense to let MLPs covering nearby segments share a portion of their parameters. For instance, we may let the D segments share the first K matrices in their neural subspace construction, and thus we could only learn and store $(N-K)D+K$ weight matrices instead of ND matrices. In Fig. 2, we illustrate our modification to the MLP layers in the classic case of a sub-aperture light field. These modifications could be easily extended to light field videos by sharing weights among adjacent time intervals instead of nearby viewpoints. Initially, we train the network at Level 0, optimizing all N layers for all light-field pixels. After the first round of training, we pick the $(N-K)$-th layer as the starting point of fine-tuning. Specifically, when fine-tuning at Level 1, we freeze the first $N-K$ layers trained at Level 0 and only optimize the last K layers. In the example shown in Fig. 2, this step is equivalent to creating four copies of the last three matrices and then separately fine-tuning each copy on one patch of pixels at Level 1. When finetuning Level 2, we additionally freeze the $(N-K+1)$-th layer (green matrix in Fig. 2) and train the remaining $K-1$ layers.

3.4 Soft-Classification for RGB Prediction

RGB color values are the common network output in neural representations of light fields, images, videos, and radiance fields. Visual information is often stored as 8-bit color with 256 possible discretized values, but are usually normalized to be within 0 to 1, before being used to train those neural representations. As a result, the final layer of the neural network is often designed to directly output three scalar values corresponding to the normalized RGB values. Effectively, the network weights, along with the final activation function (such as linear or sigmoid), perform a direct regression on the color values.

While this standard direct regression approach is sensible and works well in practice, it is not the only valid approach. A naive alternative would be to let the network classify, out of all possible colors, which specific color the input pixel belongs to. However, the 8-bit RGB format essentially covers $(2^8)^3 = 2^{24}$ individual colors, which would require the final network layer to have $M \times 2^{24}$ parameters and defeat the purpose of using this network for compression.

Instead, we let the network classify each of the three color channels separately. Specifically, its final layer outputs a vector with 768 values, which we then reshape into a 3×256 matrix. We then apply the softmax function separately for each of the three rows. As a result, the 256 values in each row represent the probability of those 256 possible colors being the true color of this channel.

Common machine learning applications adopt hard classification, where the final prediction is determined as the class with the highest corresponding probability. This is appropriate when the classes are discrete and have no ordered relationship (e.g. predicting dog/cat/chicken). However, in our setting, the possible classes are by definition ordered, and thus we do not have to adopt hard classification in practice. Instead, we multiply the predicted probability with their actual value (from 0 to 255), and then sum them up as the final output. As shown in later experimental results, our soft-classification approach enhances the reconstruction quality of our neural representation. Moreover, compared to hard classification, it does not restrict the output to be discrete color levels; it has the same flexibility as direct regression to produce any floating numbers with comparable numeric precision.

4 Experiments

In this section, we provide detailed analysis of our method through quantitative experiments. First, we confirm its overall validity by comparing its compression performance with several prior methods on traditional sub-aperture light fields. Second, we validate our use of the adaptive weight-sharing strategy, juxtaposing it with an alternative approach that seems potentially more effective but achieves inferior results in practice. Third, we demonstrate the benefit of predicting RGB color using our proposed soft-classification strategy. Fourth, we show the efficacy of our framework on dynamic light field scenes where its streaming-friendly property flourishes. Finally, we analyze the impact of the varying the hyper-parameters of the network architecture.

4.1 Training Setup

We train the MLPs in PyTorch using the Adam optimizer with default parameters in all our experiments. We supervise the network based on the mean squared error loss between reconstructed and ground truth colors. We train and test the networks under the automatic mixed precision mode in PyTorch using 16-bit floats. We adopt the cosine-annealing learning rate scheduler, with the initial learning rate as 1×10^{-5} and the final learning rate as 1×10^{-8}. We randomly shuffled all the training samples and divided them into batches of size equal to 2048. To ensure reproducibility, we set the random seed as 0.

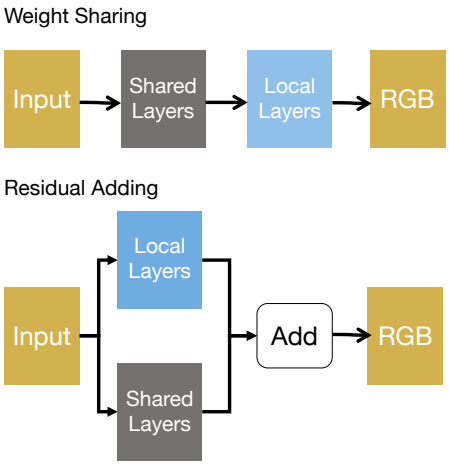

Fig. 3: **Two possible strategies to incorporate local subspaces into the network architecture**. Weight Sharing finetunes the later network layers, while Residual Adding adjusts the features through addition for different subspaces. Our empirical results show Weight Sharing achieves better quality while using fewer parameters.

Method	PSNR(↑)	SSIM(↑)	DecMB(↓)
SIGNET	39.14	0.975	9.0
+ Residual Adding	35.30	0.943	2.3
+ Weight Sharing	38.91	0.972	1.8
+ Hard Classification	39.26	0.979	2.2
+ Soft Classification	39.93	0.982	2.2

TABLE 2: **Quantitative Comparison of Different Strategies**. Weight Sharing achieves subspace specialization with better quality than Residual Adding, despite using fewer parameters. Soft Classification achieves better performance in improving prediction accuracy over Hard Classification and Direct Regression (used by SIGNET).

Unless otherwise noted, we set all networks to have ten layers with residual connections, where the intermediate layers have a dimension size of 512×512. We apply layer normalization after all MLP layers except the final one.

For networks with sine activation functions, we follow the initialization method suggested by Sitzmann *et al.* [2].

4.2 Quantitative Metrics

To evaluate the reconstruction accuracy, we compute PSNR and SSIM, two commonly used metrics for measuring the fidelity of image data. To holistically evaluate the parameter efficiency, we compute two metrics: *DecMB*, the memory required to decode any frame in megabytes completely, and *EncMB*, the memory needed to encode the entire light field for storage in megabytes. In the context of streaming applications, *DecMB* better reflects the cost for real-time transmission and on-device decoding and rendering. In contrast, *EncMB* captures the total storage cost of the server and is less relevant to the cost of large-scale, repetitive content transmissions.

4.3 Sub-aperture Light Fields

We first deploy our framework on the classic sub-aperture light fields from the Stanford Light Field Archive [37]. We report the performance of our methods compared with previous methods in Table 1, using the same set of scenes selected by Miandji et al. [23] for a direct comparison. Results indicate that our method achieves a significant (ranging from 5.2x to 12.5x) reduction in the decoding bandwidth cost with improved reconstruction accuracy. Moreover, to comprehensively evaluate the improvement of our method over SIGNET, we present results on the rest of the Stanford Light Field Archive scenes in Table **??**.

4.3.1 Comparison with the Residual Approach

While the above preliminary results may serve as a proof of concept of the overall framework, we now demonstrate the specific importance of our weight-sharing strategy in enabling subspace specialization. We present an alternative approach with a residual adding setup, shown in Fig. 3. For this approach, instead of finetuning the later layers for a local segment, we train a parallel path of layers that learns to produce the residual error of the original path. This residual adding approach is similar to many conventional modifications to neural network architectures [38], [39], [40], and on the surface, it might seem more promising than the weight sharing setup. However, as the results in Table 2 and **??** and Fig. **??** show, it is much less effective in accurately constructing a local neural subspace, even if we allow it to use more parameters than the weight sharing approach.

4.3.2 Improved Accuracy with Soft Classification

Using the sub-aperture scenes, we also analyze the impact of using different strategies to obtain the final color prediction. We train our network with Soft Regression, Hard Classification, and Soft Classification as discussed in Section 3. Soft Regression directly outputs the predicted color values as done in [1]. Hard Classification uses the color level with the highest predicted probability as the output for each channel. Soft Classification aggregates the predicted probability with multiply-then-sum approach introduced in Section 3. We measure the performance on the scenes as in Table 1, and the averaged results are provided in Table 2.

4.4 Volumetric Light Fields

The results thus far validate our method's usefulness on classic light field scenes, and they also approve the specific implementation of our weight-sharing strategy. Nonetheless, the neural subspace method truly shines in the case of large-scale light fields with irregular camera layout and even dynamic content. In this scenario, it is reasonable to assume that only a small portion of the light field scene must be streamed and displayed in real-time. It makes more sense to encode each local subspace separately to reduce the streaming bandwidth.

Therefore, we try to relax the assumption of only focusing on classic light fields with the two-plane parameterization. We create our dataset of room-scale light field scenes captured using 64 synchronized motion cameras at the resolution of 1640×1232 and 30 frames per second. Unlike the classic light fields where all the viewpoints exist

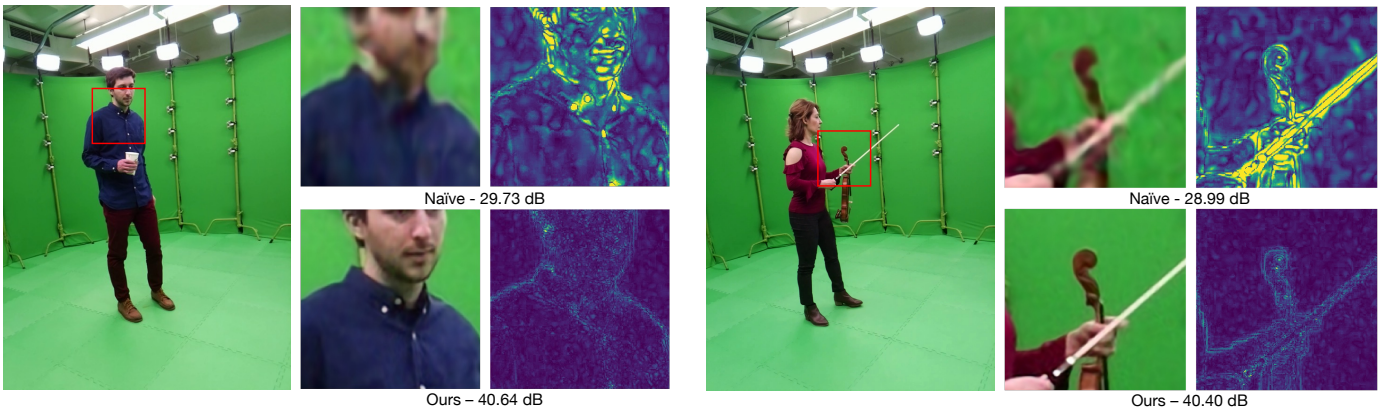

Fig. 4: Comparisons from the cylindrical light field scenes. We show the original image (left), reconstructed images using the Naive method (top right), and our method (bottom right). The Naive method only uses one MLP network to encode the content (similar to a slim SIGNET), but without the subspace specialization and weight sharing described in Section 3. We design the MLP such that the Naive method incurs a similar streaming bandwidth cost. The residual errors are visualized in yellow. Our subspace setup achieves superior visual quality.

Scene	Method	PSNR(↑)	SSIM(↑)	EncMB(↓)	DecMB(↓)
Cup (370MB)	SIGNET	38.04	0.955	9.0	9.0
	Naive	29.73	0.898	0.38	0.38
	Ours	40.64	0.967	5.2	0.46
Violin (370MB)	SIGNET	37.69	0.953	9.0	9.0
	Naive	28.99	0.891	0.38	0.38
	Ours	40.40	0.966	5.2	0.46
Training (370MB)	SIGNET	39.29	0.970	9.0	9.0
	Naive	30.38	0.911	0.38	0.38
	Ours	42.81	0.982	5.2	0.46

TABLE 3: **Results on Volumetric Light Fields**. Naive refers to naively reducing the width of the MLP networks without specializing on subspace segments as described in Section 3. Our results achieves higher accuracy and better parameter efficiency than SIGNET. Compared to the Naive method, our subspace-based method better balances the trade-off between streaming and storage efficiency.

Scene	Method	PSNR(↑)	SSIM(↑)	MB/s(↓)	Δf(↓)
Vid-Cup	MPEG	37.12	0.959	0.15	44.89
	Ours	38.21	0.968	0.15	45.25
Vid-Violin	MPEG	37.54	0.957	0.15	38.39
	Ours	38.11	0.962	0.15	38.31
Vid-Training	MPEG	37.61	0.955	0.15	39.95
	Ours	38.14	0.962	0.15	40.12

TABLE 4: **Comparison to MPEG on Dynamic Light Field Videos**. Controlling for the same level of bitrates, our method achieves higher fidelity without the blocking artifacts commonly seen (e.g. Fig. 5) in traditional, patch-based methods represented by MPEG. Δf measures the level of flickering as discussed in Sec. 4.4.2.

on a single 2D plane and are restricted to the same front-facing orientation, our 64-camera setup contains viewpoints distributed on a cylindrical surface. It provides a 360° all-around coverage of the scene. For each scene, we record 210 frames and capture various human movements.

Compressing high-resolution scenes with such a structural layout is not straightforward for prior methods. Many prior methods for light field compression are tailored to light fields with viewpoints restricted to a 2D planar grid [11], [25], [41] . The non-planar layout of the cameras results in both translation and rotation among the viewpoints, which is ill-posed for these compression methods that utilize translation between viewpoints to represent repetitive patterns compactly. On the other hand, some other methods [22], [23] explicitly construct a dictionary for encoding a new set of content, and they require sending this sizable dictionary for decoding, which is not ideal for streaming applications. On the contrary, neural-network-based methods like SIGNET and ours are flexible towards different camera layouts without a dictionary requirement. Considering their superior compression performance over previous methods already shown in Table 1, we thus primarily focus on the neural network-based methods in this section.

Since the light field scenes, in this case, cannot be parametrized by two planes, we label the cameras using cylindrical coordinates, which make up the angular part of each light field pixel's input coordinate to the MLP network.

Let us consider encoding a static moment captured by the volumetric light field. First, we directly deploy the original SIGNET design on our dataset by training one MLP to encode all 64 images of each scene. Then, to achieve a streaming-friendly file size, we train a slim SIGNET version on our data by reducing SIGNET's network width by four times, which effectively cuts the number of parameters by sixteen times. We call this the Naive method since it merely reduces the network's representation capacity compared to a full SIGNET without adjusting to compensate for its representation quality.

Finally, we apply our neural subspace framework on these light field scenes. Each scene is divided into 16 segments. Each segment contains four viewpoints vertically aligned. We construct the neural subspaces across these 16 segments by training their corresponding MLP network weights. The MLP network's architecture is specifically designed to incur a streaming bandwidth cost similar to the

Fig. 5: Results on Dynamic Volumetric Light Field scenes. For each example frame, we show zoomed-in comparisons between MPEG (top) and our method (bottom). At a similar bitrate level, our method avoids the blocking artifacts commonly seen in MPEG-compressed content.

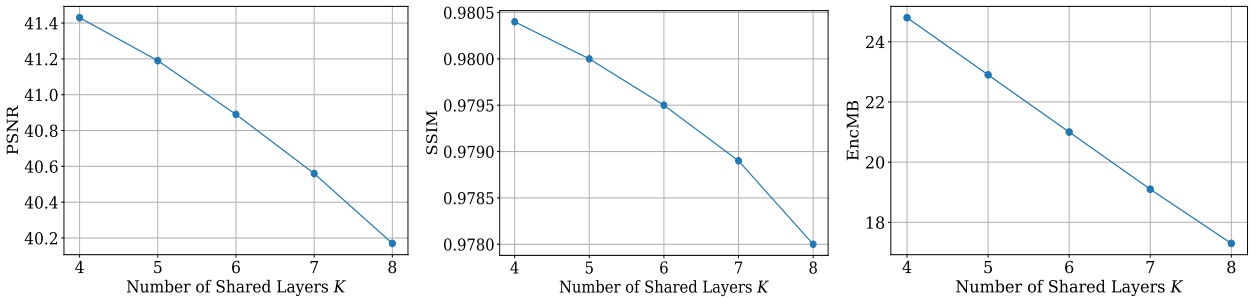

Fig. 6: Impact of Changing the Number of Shared Layers K. As the number of shared layers increases, there are fewer subspace-specific layers as we maintain the total number of layers, reducing the benefit of subspace specialization. On the other hand, having more shared layers reduces the total storage cost *EncMB*.

Naive method. Specifically, the first four layers are shared across the subspace, while the remaining layers are adapted for each subspace. More details of the network architecture are listed in Table **??**.

In Table 3 and Fig. 4 we show the quantitative and qualitative results. Compared to SIGNET, our method considerably reduces storage and streaming costs while improving the reconstruction accuracy due to the local subspace specialization. The Naive method, while achieving a lightweight file size suitable for streaming, fails to accurately encode the data, which shows the importance of our proposed strategy of subspace specialization.

4.4.1 *Extension to Dynamic Content*

We further apply our framework on dynamic sequences captured with our volumetric setup. We partition each light

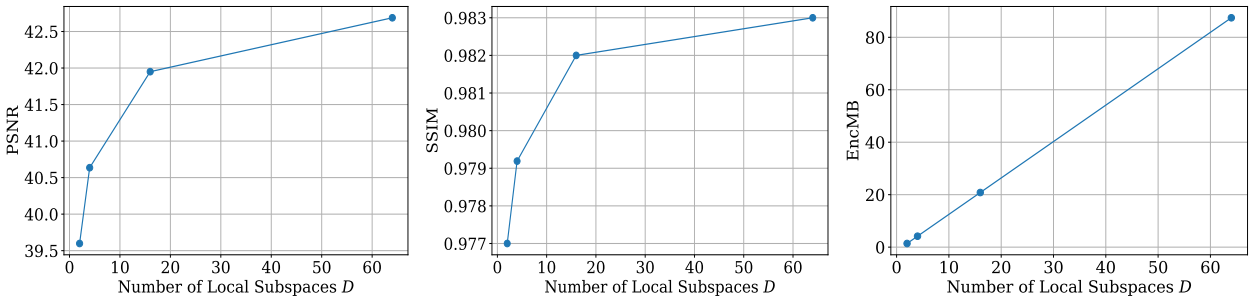

Fig. 7: Impact of Changing the Number of Local Subspaces D. As the number of local subspaces increases, each subspace covers a smaller region in the light field, therefore reducing the network fitting difficulty and hence higher reconstruction quality. On the other hand, having more individual subspaces entails higher storage cost *EncMB*.

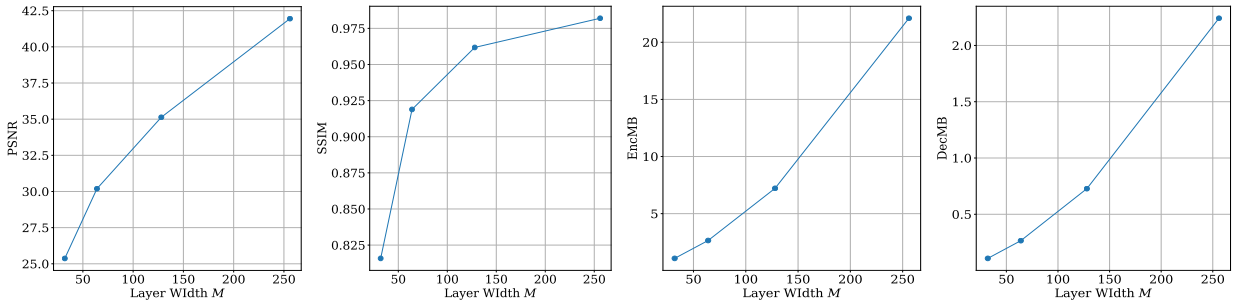

Fig. 8: Impact of Changing the Layer Width M. As the width of the intermediate layers increases, the network has more capacity to fit the data and achieves higher reconstruction quality. However, the number of parameters required in storage *(EncMB)* and decoding *(DecMB)* increases at the same time.

field scene into 14 temporal subspaces (each encompassing 0.5 seconds of content) and train the neural network weights for each subspace with the same layer allocation as in the static case. We provide the quantitative results in Table 4, and in Fig. 5 we show example frames reconstructed from our network juxtaposed with frames processed using the conventional MPEG codec. We control the MPEG bitrates to be as low as possible without sacrificing the frame rate. Quantitatively, our method achieves higher fidelity and avoids the typical blocking artifacts of MPEG as in Fig. 5.

4.4.2 Possible Flickering Across Subspaces

To quantitatively measure the possible flickering when moving across subspaces, we adopt the Flicker metric [42], [43], which measures the temporal flickering across consecutive frames. Unlike previous works [42], [43] that measure the flickering across entire videos, we are interested in how the subspace division strategy affects the level of flickering. Specifically, assuming the previous subspace ends at time $t-1$ and the next subspace starts at time t, we compute the flicker $f_{t-1\to t+1}$. For all subspace initial time steps t, we record the difference in flickering between the reconstructed frames and original ground truth frames

$$\Delta f_t = |f^{Recon}_{t-1\to t+1} - f^{True}_{t-1\to t+1}|, \tag{3}$$

and report the average Δf in Table 4. The level of flickering exhibited between subspaces is similar to using MPEG.

4.5 Hyper-parameter Analysis

Our strategy to achieve subspace specializations through the network design involved several hyper-parameters, such as the number of shared layers K among the subspaces, the number of subspaces D to divide the light field, and the width M of the network layers. In Figs. 6, 7, and 8, we present the results of varying each of those hyper-parameters while fixing all others, using the same dataset (*Lego*) and the same training batch size.

Changing K or D does not change the number of parameters required to decode any view, *DecMB* , while changing M affects both the storage cost *EncMB* and *DecMB*. When changing the number of subspaces D, $D=2$ refers to separating all viewpoints into the left and right halves, $D=4$ divides the viewpoints into quadrants, $D=16$ is our adopted setup dividing the two viewpoint axes by 4, and $D=64$ divides the two axes by 8.

5 Discussions

5.1 End-to-end v.s. Two-stage Learning

Our method essentially performs a two-stage learning, with the global stage of training the network to represent the entire light field, followed by a local stage where we fine-tune the layers specific to each subspace. A conceptually viable alternative is to combine the two separate stages and directly train the method end-to-end. Specifically, at each training iteration, we may dynamically select the subspace-specific layers for the current training batch.

However, we observe inferior reconstruction accuracy from such an end-to-end learned network. We believe a major reason is that, under the end-to-end learning scenario, each training batch comes from the same subspace. In con-

trast, in the implementation of this paper and SIGNET [1], all light field pixels are **randomly shuffled** and divided into batches at the first (global) stage. Compared to just sampling 2048 pixels randomly **from one randomly selected image**, this completely randomized data preparation was adopted because it led to a better global accuracy. Therefore, when we use the end-to-end scheme with batches from the same subspace, it is not surprising that it learns an inferior global solution than training with completely randomized batches. If we use global randomized batches while dynamically selecting the subspace layers at each training iteration, we would need to split up the batch on-the-fly in every forward pass, sort the pixels by their subspaces, and then separately infer them using their own local subspace layers. This leads to approximately $20\times$ slower run-time using PyTorch. In other words, this potentially helpful approach is unfortunately not computationally efficient enough under the current deep learning framework.

5.2 Limitations

Our method's encoding and decoding speed are inherently dependent on neural networks' training and inference speed on the available hardware. We train the MLP network weights from scratch to encode a new scene, which takes over an hour to achieve acceptable quality. Decoding a 1024×1024 image, the network takes around 0.258 seconds to complete the forward pass on an NVIDIA A4000 GPU. Another limitation is that our method currently only achieves lossy compression. It remains uncertain whether such neural-network-based methods could ever come close to lossless encoding under a prolonged training regime or if they could theoretically establish an error bound.

5.3 Future directions

It would be worthwhile for a future study to explore incorporating meta-learning or few-shot learning [44], [45] to speed up the encoding, potentially utilizing the prior knowledge or re-using the weights learned from earlier light field scenes. Moreover, since the implicit neural representation naturally allows random access to individual pixels, another interesting direction is to combine it with foveated [46] or view-dependent rendering techniques [47] to enable more efficient rendering. Considering the lossy nature of neural representation, it would therefore be a promising research direction to explore advancing such a neural-network-based method towards lossless compression.

6 Conclusion

As the intersection among neural networks, graphics, and multimedia processing is rapidly expanding, content creation and application possibilities based on high-resolution light field data are on the horizon. Compactly representing such densely sampled data is crucial for the light field's wider dissemination and impact on people's activities. Our work builds on the recent progress of neural computing. It invokes the classic idea of subspace learning, taking neural light field representations to the next step by making them more compact and streaming-friendly. We hope the new framework of neural subspace for light fields inspires more progress in bringing light field content closer to a ubiquitous component of our daily lives.

Acknowledgments

This work has been supported in part by the NSF Grant 18-23321 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] B. Y. Feng and A. Varshney, "SIGNET: Efficient Neural Representation for Light Fields," *International Conference on Computer Vision (ICCV) 2021*, 2021.

[2] V. Sitzmann, J. N. P. Martel, A. Bergman, D. B. Lindell, and G. Wetzstein, "Implicit Neural Representations with Periodic Activation Functions," in *Advances in Neural Information Processing Systems*, ser. NeurIPS 2020, 2020.

[3] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, "Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains," in *Advances in Neural Information Processing Systems*, ser. NeurIPS 2020, 2020.

[4] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," in *Proceedings of the European Conference on Computer Vision (ECCV 2020)*, 2020, pp. 405–421.

[5] M. Magnor and B. Girod, "Data compression for light-field rendering," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 338–343, 2000.

[6] C. Chang, X. Zhu, P. Ramanathan, and B. Girod, "Light Field Compression Using Disparity-compensated Lifting and Shape Adaptation," *IEEE Transactions on Image Processing*, vol. 15, pp. 793–806, 2006.

[7] A. Jagmohan, A. Sehgal, and N. Ahuja, "Compression of lightfield rendered images using coset codes," *The Thirty-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, vol. 1, pp. 830–834, 2003.

[8] X. Zhu, A. Aaron, and B. Girod, "Distributed Compression for Large Camera Arrays," *IEEE Workshop on Statistical Signal Processing, 2003*, pp. 30–33, 2003.

[9] M. Broxton, J. Flynn, R. Overbeck, D. Erickson, P. Hedman, M. DuVall, J. Dourgarian, J. Busch, M. Whalen, and P. Debevec, "Immersive light field video with a layered mesh representation," *ACM Transactions on Graphics (TOG)*, vol. 39, pp. 86:1 – 86:15, 2020.

[10] X. Jiang, M. L. Pendu, R. A. Farrugia, and C. Guillemot, "Light Field Compression With Homography-Based Low-Rank Approximation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, pp. 1132–1145, 2017.

[11] S. Pratapa and D. Manocha, "HMLFC: Hierarchical Motion-Compensated Light Field Compression for Interactive Rendering," *Computer Graphics Forum*, vol. 38, November 2019.

[12] G. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, pp. 1649–1668, 2012.

[13] D. Mukherjee, J. Bankoski, A. Grange, J. Han, J. Koleszar, P. Wilkins, Y. Xu, and R. Bultje, "The Latest Open-source Video Codec VP9 - An Overview and Preliminary Results," *2013 Picture Coding Symposium (PCS)*, pp. 390–393, 2013.

[14] M. L. Pendu, C. Guillemot, and A. Smolic, "A Fourier Disparity Layer Representation for Light Fields," *IEEE Transactions on Image Processing*, vol. 28, pp. 5740–5753, 2019.

[15] M. L. Pendu and A. Smolic, "High Resolution Light Field Recovery with Fourier Disparity Layer Completion, Demosaicing, and Super-Resolution," *2020 IEEE International Conference on Computational Photography (ICCP)*, pp. 1–12, 2020.

[16] E. Dib, M. L. Pendu, and C. Guillemot, "Light Field Compression Using Fourier Disparity Layers," *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 3751–3755, 2019.

[17] M. L. Pendu, C. Ozcinar, and A. Smolic, "Hierarchical Fourier Disparity Layer Transmission For Light Field Streaming," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 2606–2610.

[18] J. Mairal, F. R. Bach, J. Ponce, and G. Sapiro, "Online Dictionary Learning for Sparse Coding," in *Proceedings of the 26th annual International Conference on Machine Learning (ICML '09)*, 2009, pp. 689–696.

[19] E. Miandji, J. Kronander, and J. Unger, "Learning-Based Compression of Surface Light Fields for Real-Time Rendering of Global Illumination Scenes," in *SIGGRAPH Asia 2013 Technical Briefs*, ser. SA '13, 2013.

[20] J. Sulam, V. Papyan, Y. Romano, and M. Elad, "Multilayer Convolutional Sparse Modeling: Pursuit and Dictionary Learning," *IEEE Transactions on Signal Processing*, vol. 66, pp. 4090–4104, 2018.

[21] A. Abdi, A. Payani, and F. Fekri, "Learning Dictionary for Efficient Signal Compression," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 3689–3693.

[22] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[23] E. Miandji, S. Hajisharif, and J. Unger, "A Unified Framework for Compression and Compressed Sensing of Light Fields and Light Field Videos," *ACM Transactions on Graphics (TOG)*, vol. 38, pp. 1 – 18, 2019.

[24] S. Hajisharif, E. Miandji, P. Larsson, K. Tran, and J. Unger, "Light Field Video Compression and Real Time Rendering," *Computer Graphics Forum*, vol. 38, 2019.

[25] S. Pratapa and D. Manocha, "RLFC: Random Access Light Field Compression Using Key Views and Bounded Integer Sequence Encoding," *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, May 2019.

[26] J. Nystad, A. Lassen, A. Pomianowski, S. Ellis, and T. Olson, "Adaptive Scalable Texture Compression," in *EGGH-HPG'12*, 2012.

[27] N. K. Kalantari, T. Wang, and R. Ramamoorthi, "Learning-Based View Synthesis for Light Field Cameras," *ACM Transactions on Graphics (TOG)*, vol. 35, pp. 1 – 10, 2016.

[28] M. Bemana, K. Myszkowski, H.-P. Seidel, and T. Ritschel, "X-fields: Implicit neural view-, light-and time-image interpolation," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–15, 2020.

[29] V. Sitzmann, S. Rezchikov, W. Freeman, J. Tenenbaum, and F. Durand, "Light Field Networks: Neural Scene Representations with Single-Evaluation Rendering," vol. 34, 2021.

[30] B. Y. Feng, Y. Zhang, D. Tang, R. Du, and A. Varshney, "PRIF: Primary Ray-based Implicit Function," in *Proceedings of the European Conference on Computer Vision (ECCV 2022)*, 2022.

[31] B. Y. Feng, S. Jabbireddy, and A. Varshney, "VIINTER: View Interpolation With Implicit Neural Representations of Images," in *ACM SIGGRAPH Asia 2022 Conference Proceedings*, 2022.

[32] Y. Zhang, T. van Rozendaal, J. Brehmer, M. Nagel, and T. Cohen, "Implicit Neural Video Compression," *arXiv preprint arXiv:2112.11312*, 2021.

[33] G. W. Stewart, "An Updating Algorithm for Subspace Tracking," *IEEE Trans. Signal Processing*, vol. 40, pp. 1535–1541, 1992.

[34] F. D. L. Torre and M. J. Black, "A Framework for Robust Subspace Learning," *International Journal of Computer Vision*, vol. 54, pp. 117–142, 2004.

[35] B. Wang and Z. Tu, "Sparse Subspace Denoising for Image Manifolds," *2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 468–475, 2013.

[36] N. Vaswani, T. Bouwmans, S. Javed, and P. Narayanamurthy, "Robust Subspace Learning: Robust PCA, Robust Subspace Tracking, and Robust Subspace Recovery," *IEEE Signal Processing Magazine*, vol. 35, pp. 32–55, 2018.

[37] A. Adams, "The Stanford Light Field Archive," 2008, http://lightfield.stanford.edu/lfs.html.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[39] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual Dense Network for Image Super-Resolution," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2472–2481, 2018.

[40] J. Flynn, M. Broxton, P. Debevec, M. DuVall, G. Fyffe, R. Overbeck, N. Snavely, and R. Tucker, "DeepView: View Synthesis With Learned Gradient Descent," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2362–2371, 2019.

[41] G. Wu, M. Zhao, L. Wang, Q. Dai, T. Chai, and Y. Liu, "Light Field Reconstruction Using Deep Convolutional Network on EPI," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1638–1646, 2017.

[42] S. Winkler, E. D. Gelasca, and T. Ebrahimi, "Toward Perceptual Metrics for Video Watermark Evaluation," in *Applications of Digital Image Processing XXVI*, vol. 5203. SPIE, 2003, pp. 371–378.

[43] D. Li, R. Du, A. Babu, C. D. Brumar, and A. Varshney, "A Log-Rectilinear Transformation for Foveated 360-degree Video Streaming," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 5, pp. 2638–2647, 2021.

[44] M. Tancik, B. Mildenhall, T. Wang, D. Schmidt, P. P. Srinivasan, J. T. Barron, and R. Ng, "Learned Initializations for Optimizing Coordinate-based Neural Representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2846–2855.

[45] A. Bergman, P. Kellnhofer, and G. Wetzstein, "Fast Training of Neural Lumigraph Representations Using Meta Learning," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[46] X. Meng, R. Du, J. F. JaJa, and A. Varshney, "3D-Kernel Foveated Rendering for Light Fields," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 8, pp. 3350–3360, 2021.

[47] C. Koniaris, M. Kosek, D. Sinclair, and K. Mitchell, "Compressed Animated Light Fields with Real-time View-dependent Reconstruction," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 4, pp. 1666–1680, 2018.

Brandon Yushan Feng Brandon Yushan Feng is a Ph.D. candidate and a research assistant in the Computer Science Department at the University of Maryland, College Park, advised by Prof. Amitabh Varshney. He holds a M.S. in Statistics and a B.A. in Computer Science and Statistics from the University of Virginia. His research focuses on leveraging machine learning for vision and graphics applications.

Amitabh Varshney Amitabh Varshney is Dean of the College of Computer, Mathematical, and Natural Sciences and a Professor of Computer Science at the University of Maryland, College Park. In his research, Varshney explores applications of high-performance computing and visualization in engineering, science, and medicine. He has worked on a number of research areas including visual saliency, summarization of large visual datasets, and visual computing for big data. He is currently exploring applications of virtual and augmented reality in several applications, including healthcare and telemedicine. He has served in various roles in the IEEE Visualization and Graphics Technical Committee, including as its Chair (2008–2012). He received the IEEE Visualization Technical Achievement Award in 2004. He is a Fellow of IEEE and a member of the IEEE Visualization Academy.