Masked Wavelet Representation for Compact Neural Radiance Fields

Daniel Rho[1*]   Byeonghyeon Lee[2*]   Seungtae Nam[2]   Joo Chan Lee[2]   Jong Hwan Ko[2,3†]

Eunbyung Park[2,3†]

[1]AI2XL, KT

[2]Department of Artificial Intelligence, Sungkyunkwan University

[3]Department of Electrical and Computer Engineering, Sungkyunkwan University

Abstract

Neural radiance fields (NeRF) have demonstrated the potential of coordinate-based neural representation (neural fields or implicit neural representation) in neural rendering. However, using a multi-layer perceptron (MLP) to represent a 3D scene or object requires enormous computational resources and time. There have been recent studies on how to reduce these computational inefficiencies by using additional data structures, such as grids or trees. Despite the promising performance, the explicit data structure necessitates a substantial amount of memory. In this work, we present a method to reduce the size without compromising the advantages of having additional data structures. In detail, we propose using the wavelet transform on grid-based neural fields. Grid-based neural fields are for fast convergence, and the wavelet transform, whose efficiency has been demonstrated in high-performance standard codecs, is to improve the parameter efficiency of grids. Furthermore, in order to achieve a higher sparsity of grid coefficients while maintaining reconstruction quality, we present a novel trainable masking approach. Experimental results demonstrate that non-spatial grid coefficients, such as wavelet coefficients, are capable of attaining a higher level of sparsity than spatial grid coefficients, resulting in a more compact representation. With our proposed mask and compression pipeline, we achieved state-of-the-art performance within a memory budget of 2 MB. Our code is available at https://github.com/daniel03c1/masked_wavelet_nerf.

1. Introduction

Recent advances in coordinate-based neural representation (neural fields or implicit neural representation) have demonstrated remarkable performance in many applications. In particular, neural radiance fields (NeRF) have

---

*Equal contribution
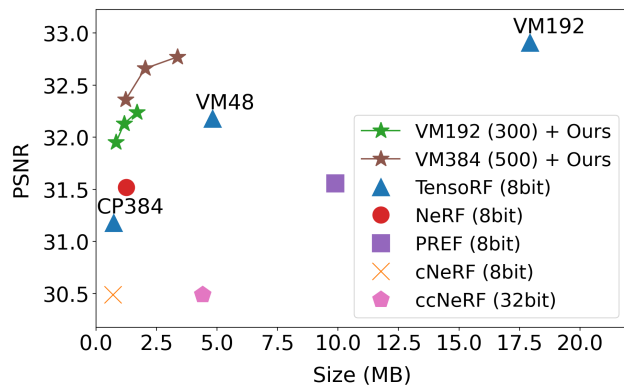
†Corresponding authors

Figure 1. Rate-distortion curves on the NeRF synthetic dataset. The numbers inside parenthesis denote the axis resolution of grids.

sparked interest by synthesizing high-quality images from novel viewpoints. It uses a multi-layer perceptron (MLP) with positional encoding to map coordinates to corresponding colors and opacities. Combined with the differentiable volumetric rendering and the neural network's architectural priors, it has shown great potential to be a new representation paradigm. However, the high computational costs (in both training and inference) have been a significant bottleneck, often taking several days to converge.

Several follow-up studies have been proposed to accelerate training and inference times [5, 9, 11, 14, 16, 31, 32, 40, 44]. To speed up inference, KiloNeRF [32] proposed splitting a 3D scene into thousands of partial scenes, each of which is assigned a tiny, distinct neural network. While achieving impressive speed-up, it requires a massive amount of memory storage. On the other hand, Fast-NeRF [14] suggested caching and factorizing the NeRF network to reduce computational costs. Meanwhile, meta-learning algorithms [12, 26, 29] have also been applied to accelerate the training time, and they have shown faster convergence with the learned initialization [34, 40]. However, it requires well-organized large-scale datasets and a pre-training process. Furthermore, the rendering costs re-

main unchanged since they use the same network architecture [40], and meta-learning algorithms often suffer from poor out-of-distribution generalization performance.

Alternatively, there has been a surge of recent interest in incorporating classical data structures, such as grids or trees, into the NeRF framework [13, 22, 24, 33, 38, 39, 49]. Incorporating additional data structures has significantly reduced training and inference time (from days to a few minutes) without compromising reconstruction quality. However, the overall size dramatically increases due to these dense and volumetric structures. In order to reduce the spatial complexity, several methods have been proposed, including pruning areas [33, 37, 49], encodings [24, 38], and tensor decomposition [3, 4, 17].

This paper aims to further improve the spatial complexity while maintaining the rendering quality. Leveraging the decades of research on standard compression algorithms [35, 36, 46], we propose compressing grid-based neural fields using frequency-based transformations. In the frequency domain, a large portion of the coefficients can be discarded without considerably degrading the reconstruction quality, and most standard compression algorithms have exploited frequency domain representations. Thus, we propose using this property on grid-based neural fields to maximize parameter efficiency. Among other alternatives, we employ the discrete wavelet transform (DWT) due to its compactness and ability to efficiently capture both global and local information.

Once we obtain sparse representations via frequency domain representations, we can take advantage of the existing compression techniques. Unlike conventional media data (e.g., image and audio), however, no off-the-shelf compression tools exist that we can leverage without complicated engineering efforts. In addition, since NeRF, or neural rendering networks in general, is a relatively new data format, the characteristics or patterns of their coefficients have not been thoroughly investigated. Thus, we present a compression pipeline for our purposes. To automatically filter out unnecessary coefficients, we propose a trainable binary mask. For each 3D scene, we jointly optimize grid parameters and their corresponding masks. This per-scene optimization strategy can be more optimal than the global quantization table used in standard image compression codecs [43].

To compress sparse grid representations, we first merge masks with wavelet coefficients to zero out coefficients and then apply standard compression algorithms to masked coefficients. We utilize the run-length encoding to encode binary information about which coefficients are non-zero. For further compression, we apply one of the entropy coding algorithms, Huffman encoding [19], to these encoded outputs.

Our method incurs negligible computational costs at test time, requiring only one inverse DWT (IDWT) per grid. After the IDWT, the wavelet grids are transformed into spatial grids, eliminating the need for additional IDWT during rendering. As a result, the computing time and costs (including memory costs) are identical to the original spatial grid-based NeRF.

In summary, our contributions are as follows:

- We propose using wavelet coefficients to improve parameter sparsity and reconstruction quality. Through experiments, we show that the wavelet coefficients can be more compact than the spatial domain coefficients in neural radiance fields.

- We propose a trainable mask that can be applied to any grid-based neural representation. Experimental results demonstrate that our proposed masking method can zero out more than 95% of the total grid parameters while maintaining high reconstruction quality.

- We achieve state-of-the-art performance in novel view synthesis under a memory budget of 2 MB.

2. Related Works

2.1. Neural rendering

Neural radiance fields. NeRF [23] has demonstrated that 3D scenes and objects can be successfully represented using coordinate-based neural representation. NeRF renders a scene by casting a ray per pixel and sampling colors and opacity from points that lie on the ray. However, since this approach relies on the dense, point-wise sampling of color values and opacity, the computational costs are significant.

Due to this inefficiency, there has been considerable interest in improving computational efficiency. A plethora of methods propose to adopt data structures such as trees [45, 49], point clouds [27, 28, 48, 50], and grids [4, 13, 15, 24, 33, 37, 38] to efficiently represent 3D scenes and objects. PlenOctrees [49] use an octree structure for real-time rendering. Despite the reduced rendering costs, the increased size of more than ten times the size of standard MLP-based neural fields, such as NeRF [23], limits their wide application. Point cloud-based approaches [27, 48] reduce the number of points sampled per ray by utilizing the scene geometry of a point cloud. Lastly, a number of studies have demonstrated the efficiency of using grid structures, including vectors, matrices, and tensors, on neural fields [4, 13, 24, 37]. With the use of these grid structures, training iterations that once required tens of hours can now be completed in a matter of minutes. Furthermore, real-time rendering at inference time is also possible. However, dense 3D grid structures require substantial amounts of memory, often exceeding 1 GB. To alleviate this memory requirement, several methods have been proposed to reduce the size of grids [4, 24, 38, 42].

To achieve near-instant rendering with reasonable memory requirements, Instant-NGP [24] proposed using hash-based multi-resolution grids. Using hash functions, each grid maps input coordinates to corresponding feature vectors. Similarly, VQAD [38] proposed using codebooks and vector quantization rather than the hash function. This enables control over the overall size of the neural fields. However, during training, a sizable amount of memory is required to learn which code from the codebook to use for each point on the grid.

In order to alleviate the memory requirement of using 3D grids, another line of work decomposes 3D grids into lower dimensional representations, such as planes and vectors [3, 4, 17, 42]. EG3D [3] proposes a tri-plane approach, which represents a 3D scene with three perpendicular planes and uses feature vectors, extracted separately from each plane, as inputs for the following MLPs. TensoRF [4] is another parameter-efficient yet expressive method for decomposing dense three-dimensional grids into a smaller set of parameters, such as planes and lines. Although these methods dramatically reduce the time and space complexity of the 3D scene and object representation, their overall sizes are larger than those of MLP-only methods.

Frequency-based representation. Several studies have explored frequency-based parameterization for efficient neural field representations [17, 41, 45, 47]. Fourier PlenOctree [45] has demonstrated that the Fourier transform can improve both parameter efficiency and training speed of the PlenOctree structure [49]. Nevertheless, the overall size is larger than MLP-only methods. PREF [17] has also exploited the Fourier transform in 3D scene representation. It applies the Fourier transform to a tensor decomposition-based representation. However, it has not shown notable improvements in both the representation quality and the parameter efficiency compared to spatial grid representations.

Besides the Fourier transform, there is another line of signal representations in the frequency domain: the wavelet transform. The wavelet transform decomposes a signal using a set of basis functions called wavelets. The wavelet transform can extract both frequency and temporal information, in contrast to the Fourier transform, which can only capture frequency information. Because each coefficient covers a different frequency and period in time, it is known that the wavelet transform can represent transient signals more compactly than Fourier-based transforms. This compactness contributes to standard compression codecs, such as JPEG 2000 [35]. Motivated by the success of the standard codecs, we aim to mimic their compression pipeline and apply it to the neural radiance fields. One disadvantage of the wavelet transform is that it can be less efficient than Fourier-based transforms for smooth signals. However, we believe that most situations in grid-based neural fields

are not the case, considering that each grid's resolution is constrained to accurately depict detailed scenes and objects with limited memory budgets.

2.2. Model compression

In recent years, many works have studied to downsize neural networks through various techniques such as weight pruning and quantization.

Pruning. For efficient representations, pruning methods have been explored in neural fields. Point-NeRF [48] proposes a pruning strategy for point cloud representation by imposing a sparsity loss on point confidence. Re:NeRF [8] has validated that simply applying a general pruning technique, which gradually removes coefficients throughout the course of training, to grid-based neural representations results in a significant performance drop. It proposed iterative parameter removal and conditional parameter inclusion and demonstrated comparable compression performance. However, it has not shown desirable performance under limited memory budgets. Similarly, KiloNeRF [32] prunes out parameters for unused areas using a fixed threshold. Instead, we propose a trainable mask that can remove a large number of coefficients while causing only minor degradation in the reconstruction quality.

Quantization. In terms of neural network quantization, there are two main approaches: post-quantization (PTQ) [21, 25] and quantization-aware training (QAT) [7, 18, 30, 51]. Post-quantization methods quantize network parameters after training is finished. PTQ has the advantage that it does not require additional training iterations or datasets. Furthermore, it supports arbitrary-bit quantization. Performance, however, might degrade because the network parameters were not optimized for the target bit precision. Unlike post-quantization, QAT optimizes neural networks for a certain bit precision during training iterations. Since quantization is not differentiable, it usually relies on the straight-through estimator (STE) [1] for network optimization [7]. QAT has been validated for its effectiveness in neural representation models. However, PTQ has the disadvantage of requiring additional training samples and iterations when quantizing the weights of a pretrained network. Since we train neural fields from scratch, we use PTQ for weight quantization.

3. Methods

3.1. Neural radiance fields

We consider a neural radiance field leveraging grid representations. It takes an input coordinate $x \in \mathbb{R}^3$ and viewing direction, $d \in \mathbb{R}^2$, generating a four-dimensional vector consisting of a density and three-channel RGB colors, de-
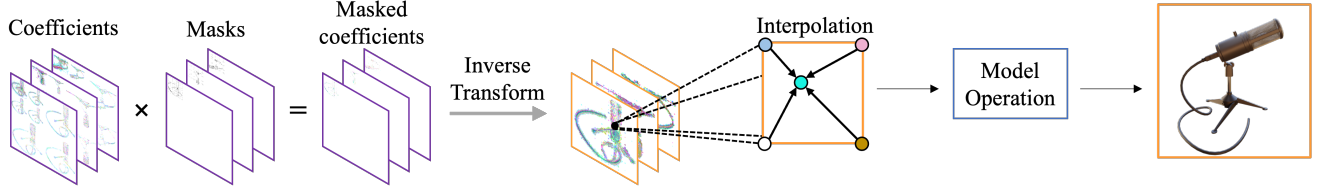
Figure 2. The overall architecture of our model. Images with purple borders illustrate wavelet coefficients, while those with orange borders visualize spatial coefficients. The wavelet coefficients in each 2D grid are multiplied with a binarized mask to form a masked wavelet coefficient grid. Masked wavelet coefficients are then inverse-transformed to spatial features. We sample feature vectors for input coordinates using bilinear interpolation on the grids. Then, opacity and color for each input coordinate are estimated using the sampled feature vectors following the defined model.

fined as follows.

$$\sigma(x) = f_\sigma(x; \gamma_\sigma, \mathcal{M}_\sigma), \tag{1}$$

$$c(x, d) = f_c(x, d; \theta, \gamma_c, \mathcal{M}_c), \tag{2}$$

where θ is the parameter of an MLP, $\gamma = \{\gamma_\sigma, \gamma_c\}$ is a set of grid parameters, and $\mathcal{M} = \{\mathcal{M}_\sigma, \mathcal{M}_c\}$ is a set of masks for grid parameters, which will be described shortly (Sec. 3.3). The following volumetric rendering equation is used to synthesize novel views:

$$C(r) = \int_{t_n}^{t_f} T(t)\sigma(r(t))c(r(t), d)dt, \tag{3}$$

$$T(t) = \exp(-\int_{t_n}^{t} \sigma(r(s))ds), \tag{4}$$

where $r(t)$ is a ray from a camera viewpoint, and $C(r)$ is the expected color of the ray $r(t)$. Two integral bounds (near and far) are denoted as t_n and t_f, respectively. Please refer to NeRF [23] for more details.

3.2. Wavelet transform on decomposed tensors

Frequency-based algorithms, such as discrete cosine transform (DCT) or discrete wavelet transform (DWT), have been developed and improved over the decades. For high compression performance, standard codecs have used them to make sparse representations. However, in the case of 3D grid representation, their computational complexity increases cubically with grid resolution, making high-resolution 3D scene and object representations impractical. In order to use frequency-based algorithms for compact representation, we need lower-dimensional data, such as 2D planes or 1D lines.

Recent studies have explored various decomposition methods to reduce the spatial complexity of 3D grid-based representations. Among them, plane-based representations have achieved remarkable success in reducing the number of parameters while maintaining the rendering performance [3, 4, 17]. They propose decomposing a 3D tensor into a set of 2D planes or 1D vectors.

To combine the best of both worlds, we propose using the wavelet transform on 2D plane-based neural fields [4]. We use the wavelet transform because of its compactness, especially for non-repeating, non-smooth signals.

For efficient 3D object and scene representation, recent studies proposed using a lower-dimensional grid (2D planes or 1D lines) [3, 4]. For example, EG3D [3] utilizes three 2D planes (tri-plane) and TensoRF [4] employs three sets, each consisting of a plane and a vector. We describe our method based on TensoRF. In detail, we use a set of 2D matrices and 1D vectors for grid representation, $\gamma_\sigma = \{\mathcal{W}_{\sigma,r}^x, \mathcal{W}_{\sigma,r}^y, \mathcal{W}_{\sigma,r}^z, v_{\sigma,r}^x, v_{\sigma,r}^y, v_{\sigma,r}^z\}_{r=1}^{N_{\sigma,r}}$ (σ denotes density, and we will omit the subscript σ for brevity from now on) as proposed in TensoRF [4]. N_r is the number of ranks in matrix-vector decomposition and $\mathcal{W}_r^x \in \mathbb{R}^{H \times W}$, $\mathcal{W}_r^y \in \mathbb{R}^{W \times D}$, $\mathcal{W}_r^z \in \mathbb{R}^{H \times D}$ are matrices, $v_r^x \in \mathbb{R}^D$, $v_r^y \in \mathbb{R}^H$, $v_r^z \in \mathbb{R}^W$, are vectors in x, y, z directions, respectively. H, W, D are the resolution of the grid. More formally, a 3D grid representation G can be defined as follows.

$$G = \sum_{r=1}^{N_r} \sum_{d \in \{x,y,z\}} v_r^d \otimes \texttt{idwt}(\mathcal{W}_r^d), \tag{5}$$

where $\otimes$ denotes the outer product and $\texttt{idwt}(\cdot)$ is a two-dimensional inverse discrete wavelet transform (IDWT).

With a slight abuse of notation, a single level IDWT of a matrix $\mathcal{W} \in \mathcal{R}^{m \times n}$ can be written as follows.

$$\begin{aligned}\texttt{idwt}(\mathcal{W}) = \Phi \star (\uparrow 2)(\mathcal{W}_{LL}) + \Psi_{HL} \star (\uparrow 2)(\mathcal{W}_{HL}) \\ + \Psi_{LH} \star (\uparrow 2)(\mathcal{W}_{LH}) + \Psi_{HH} \star (\uparrow 2)(\mathcal{W}_{HH}),\end{aligned} \tag{6}$$

where $\mathcal{W}_{LL}, \mathcal{W}_{HL}, \mathcal{W}_{LH}, \mathcal{W}_{HH} \in \mathbb{R}^{m/2 \times n/2}$ denote approximation, horizontal, vertical, and diagonal coefficients, respectively. $\star$ denotes 2D cross correlation and $(\uparrow 2): \mathbb{R}^{m/2 \times n/2} \to \mathbb{R}^{m \times n}$ denotes (nearest neighbor) upscaling. Ψ and Φ are wavelet function and scaling function, respectively. We use the biorthogonal 4.4 wavelet function and its corresponding scaling function [6]. Even though we follow the grid representation of TensoRF, our method is not limited to the existing 2D plane-based representations but can

be applied to any other approaches that adopt 2D planes to represent higher-dimensional structures.

As shown in Eq. 5, we only apply IDWT over matrices $\mathcal{W}$ since we observed that IDWT to vectors resulted in significant quality degradation. As this transformation is differentiable, the model can be trained end-to-end. We can also obtain the 3D grid representation for appearance through the same process and provide all formulations in the supplementary material.

Our method incurs insignificant costs during inference. Once training is finished, only one IDWT per grid is needed to transform wavelet grids into spatial grids. Thus, the remaining computational costs and time are exactly the same as the original spatial grid-based neural fields.

3.2.1 Multi-level wavelet transform

To further improve parameter sparsity and reconstruction quality, we propose using a multi-level wavelet transform. We experimentally found that higher-level wavelet transformations result in higher sparsity in grids (Sec. 4.3.1). However, naively using a multi-level wavelet transform on grid parameters degrades the representation quality (Tab. 1). We hypothesize that this is because the range of wavelet coefficients and the range of their gradients vary depending on the level of decomposition. In more detail, high-pass coefficients typically have a larger gradient scale with a smaller weight range. Different weight scales and gradients for different frequencies seem to hinder the optimization process. As such, we propose multiplying wavelet coefficients by the scales proportional to the inverse of frequency. With our proposed scaling factors s, Eq. 5 can be rewritten as follows.

$$G = \sum_{r=1}^{N_r} \sum_{d \in \{x,y,z\}} v_r^d \otimes \mathrm{idwt}(\mathcal{W}_r^d \odot s_r^d), \qquad (7)$$

where $\odot$ denotes the Hadamard product. For example, a two-level DWT, as shown in Fig. 3, generates a total of seven groups of coefficients (LL2, HL2, LH2, HH2, HL1, LH1, HH1). The scaling factors s for HL2, LH2, and HH2 will be set to 1/2, and for HL1, LH1, and HH1 to 1/3. This scaling method enhances the quality of the reconstruction; more experiments are described in Sec. 4.3.2.

3.3. Learning to mask

Even though Wavelet coefficients can be sparse, we need additional methods to attain higher sparsity. Thus, we propose using element-wise masks to increase the portion of zero elements in grids. By jointly optimizing binary masks and grid parameters, we aim to zero out the majority of coefficients without significantly degrading the rendering quality. As we defined earlier,
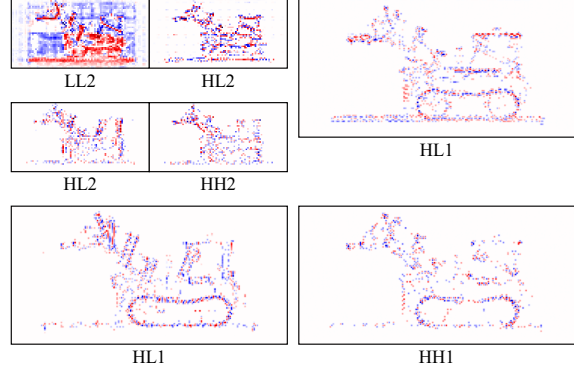

Figure 3. Multi-level discrete wavelet coefficients (2-level). LL, HL, LH, and HH denote approximation, horizontal, vertical, and diagonal coefficients, respectively. The numbers on the right denote the level of decomposition.

we have a set of trainable element-wise masks, $\mathcal{M} = \{\mathcal{M}_r^x, \mathcal{M}_r^y, \mathcal{M}_r^z, m_r^x, m_r^y, m_r^z\}_{r=1}^{N_r}$ (note that we omitted the subscript σ for brevity). During training, the grid parameters and corresponding binarized masks are multiplied element by element. Since calculating gradients directly from binarized masks is not feasible, we used the straight-through-estimator technique [1] to train and use masks. Formally, the masked matrix parameters $\widehat{\mathcal{W}}_r$ for $\mathcal{W}_r$ can be expressed as follows,

$$\widehat{\mathcal{W}}_r = \mathrm{sg}((\mathcal{H}(\mathcal{M}_r) - \sigma(\mathcal{M}_r)) \odot \mathcal{W}_r) + \sigma(\mathcal{M}_r) \odot \mathcal{W}_r, \qquad (8)$$

where $\mathrm{sg}(\cdot)$ is the stop-gradient operator. $\sigma(\cdot)$ and $\mathcal{H}(\cdot)$ denote the element-wise sigmoid and Heaviside step function, respectively. We can also compute the masked vector parameters $\widehat{v}_r$ similarly. By replacing grid parameters ($\mathcal{W}_r$, v_r) from Eq. (7) with masked grid parameters ($\widehat{\mathcal{W}}_r, \widehat{v}_r$), we can represent a masked 3D grid representation as follows:

$$G = \sum_{r=1}^{N_r} \sum_{d \in \{x,y,z\}} \widehat{v}_r^d \otimes \mathrm{idwt}(\widehat{\mathcal{W}}_r^d \odot s_r^d) \qquad (9)$$

To make binarized mask values sparse, we use the sum of all mask values as the additional loss term $\mathcal{L}_m$. The overall loss function is a sum of rendering loss $\mathcal{L}_r$ and the mask regularization term $\mathcal{L}_m$.

$$\mathcal{L} = \mathcal{L}_r + \lambda_m \mathcal{L}_m. \qquad (10)$$

We use λ_m to control the sparsity of the parameters of the grid representation.

3.4. Compression pipeline

With our proposed masking method and multi-level DWT, the ratio of zeros in grids can go up to more than
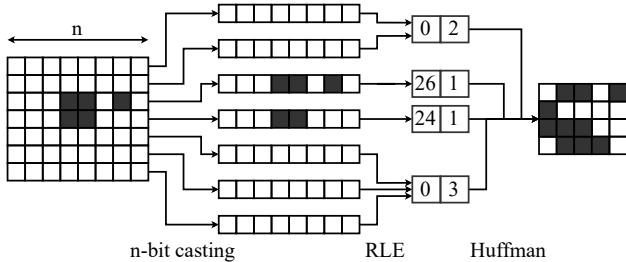
Figure 4. Proposed bitmap compression pipeline. RLE and Huffman denote run-length encoding and Huffman encoding, respectively.

95%. However, sparse representations themselves do not reduce the total size. In this section, we describe our proposed compression pipeline for sparse grid parameters. Instead of storing grids as they are, we separately store non-zero coefficients and bitmaps (or masks) that indicate which coefficients are non-zero. Despite using 1-bit bitmaps, the overall bitmap size is large due to the large number of parameters. To reduce the bitmap sizes, we propose a compression pipeline with the following three stages: n-bit casting, run-length encoding (RLE), and the Huffman encoding [19]. The compression pipeline is illustrated in Fig. 4.

Before applying the compression pipeline, we split wavelet coefficient masks by level of decomposition. For the 2-level wavelet transform, for instance, we group HL1, LH1, and HH1, then HL2, LH2, and HH2, and finally LL2 (Fig. 3). This is based on our experimental results that wavelet coefficients with the same levels of sparsity have similar levels of sparsity (Fig. 6). We found that grouping coefficients with similar sparsity results in better compression performance. In addition, we found that directly applying RLE to 1-bit streams causes inefficient bit allocation due to the numerous repeating zeros. Therefore, we first cast the binary mask values to n-bit unsigned integers, and perform RLE afterward. In our experiments, we set n to 8. Finally, we apply the Huffman encoding algorithm to the RLE-encoded streams to map values with a high probability to shorter bits.

4. Experiments

4.1. Experimental settings

We compared our proposed method with other baselines in four datasets; NeRF synthetic dataset [23], Neural Sparse Voxel Fields (NSVF) [22] dataset, TanksAndTemples dataset [20], and LLFF dataset [10].

We used TensoRF VM-192 as the baseline model and applied the proposed method. To control the size of our method, we adjusted the mask regularization weight λ_m, the number of grid channels (from VM-192 to VM-384),

and the resolution of grids. Unless otherwise specified, we used 8-bit quantization for every method. We compared our method with quantized NeRF, TensoRF models (CP and VM), cNeRF [2] and CCNeRF [42]. We also tried to compare our approach with VQAD [38], which efficiently compresses 3D scene representations using codebooks and vector quantization. However, VQAD requires depth maps to prune out vacant areas in advance. Otherwise, the required memory exceeds the memory capacity of the Tesla A100 equipped with 40 GB of memory. Since the datasets we used do not provide depth maps, we did not compare VQAD with ours.

We followed the experimental settings of TensoRF [4]. We trained models for 30,000 iterations, each of which is a minibatch of 4,096 rays. We used the Adam optimizer and an exponential learning rate decay scheduler. Following TensoRF, the initial learning rates of grid-related parameters and MLP-related parameters were set to 0.02 and 0.001, respectively. Final learning rates were set at 1/10 of the initial learning rate. We updated the alpha masks at the 2000th, 4000th, 6000th, 11000th, 16000th, 21000th, and 26000th iterations. We set λ_m in Eq. (10) to 1e-10 for high parameter sparsity and 5e-11 and 2.5e-11 for relatively lower sparsity. We set the initial values of masks $\mathcal{M}$ to one, and set their initial learning rate to the same learning rate as grid parameters ϕ. As a reconstruction quality measurement, we used PSNR.

4.2. Results

Figs. 1 and 5 show the quantitative performances of our method and the baselines on various datasets. Each graph displays the average PSNR and size of methods. More detailed numbers are available in the supplementary materials.

As the rate-distortion curves show, our method achieves higher efficiency than the baseline models and other methods. Even with the base network (VM-192), our method consistently outperforms other baselines on all novel view synthesis datasets, under a small memory budget of 2 MB. We adjusted both λ_m and the network size to increase the output size. By adjusting $lambda_m$, we can control both size and quality while having no effect on computational costs, time, or memory requirements. Increasing network size (doubling the number of channels and increasing the resolution) can push the boundary even further. It is also worth mentioning that applying our method to a larger backbone model (VM-384) outperforms models with a smaller backbone model (VM-192), even with the same value of λ_m or a similar model size. However, as the network grows in size, the computational costs and time required for training and testing increase. Therefore, it is still reasonable to use a small backbone model when using our proposed method.

Fig. 7 shows the qualitative results with and without our proposed masking method. Trainable masks removed more
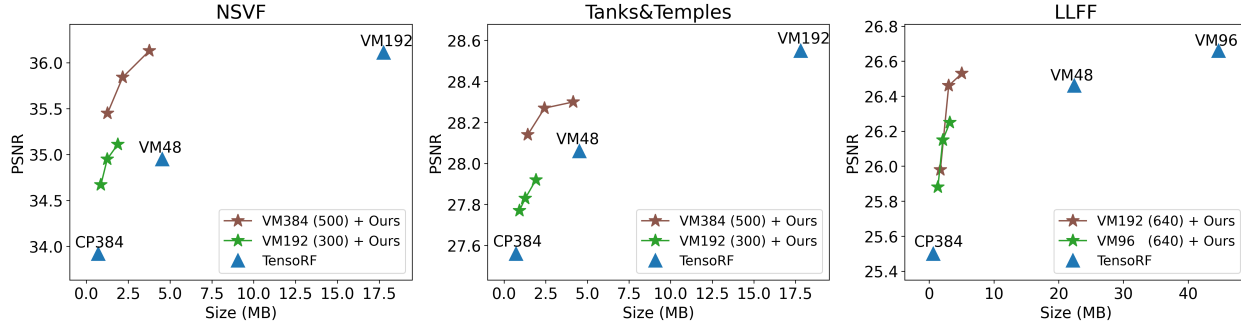
Figure 5. Rate distortion curves on the NSVF synthetic, TanksAndTemples, and LLFF dataset. We used 8-bit precision for every method.
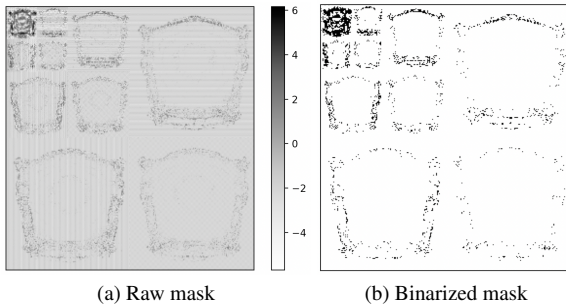

(a) Raw mask   (b) Binarized mask

Figure 6. Qualitative analysis on mask $\mathcal{M}$ (Chair in the NeRF synthetic dataset).

than 97% of the grid parameters; nevertheless, the rendered results are still accurate, and the qualitative difference is almost imperceptible. This demonstrates that our proposed method efficiently eliminates unnecessary wavelet coefficients by leveraging trainable masks.

Fig. 6 illustrates both raw and binarized masks after training. First of all, sparsity varies depending on the level of the wavelet transform. As shown in Fig. 6b, coefficients with lower frequencies have lower sparsity, whereas coefficients with higher frequencies have higher sparsity. What is interesting is that the raw mask values seem to reflect the characteristics of corresponding wavelet coefficients. Vertical, horizontal, and diagonal patterns can be found in the raw mask values for the vertical, horizontal, and diagonal coefficients, respectively.

4.3. Ablation studies

In this section, we analyze each component of our proposed method. Every experiment was conducted on the whole NeRF synthetic dataset, and we used the average PSNR as a measurement for representation quality. We used 4-level wavelet transform and set λ_m to 1e-10, unless otherwise specified.

Methods	PSNR
DWT	**31.949**
DWT w/o level-wise scaling	31.145
DCT	31.325

Table 1. Ablation study on level-wise scaling. Without our proposed scaling method, the performance of DWT is worse than DCT.

4.3.1 Discrete wavelet transform

In this section, we evaluate the compactness of wavelet coefficients and the performance improvements brought by the multi-level wavelet transform. Fig. 8 shows the rate-distortion curves of four different levels of wavelet transform and non-wavelet coefficients. As shown in the figure, wavelet-based grid representation outperforms the grid in the spatial domain, especially when the sparsity is high. Furthermore, rate-distortion curves demonstrate that higher level of wavelet transform improves visual quality and sparsity.

4.3.2 Level-wise wavelet coefficient scaling

As mentioned in Sec. 3.2.1, we proposed multiplying wavelet coefficients by the scales proportional to the inverse of frequency. Thus, we compared two versions of wavelet transform; one with our proposed weight scaling (Sec. 3.2.1) and the other without it. As Tab. 1 shows, scaling wavelet coefficients improves reconstruction performance.

4.3.3 Wavelet functions

In this section, we analyze how different wavelet functions affect reconstruction quality. For comparison, we selected several wavelet functions: Haar, Coiflet 1, Daubechies 4, and reverse biorthogonal 4.4. As shown in Tab. 2, the type of wavelet function has little effect on reconstruction quality. Even Haar, the simplest wavelet function, performs
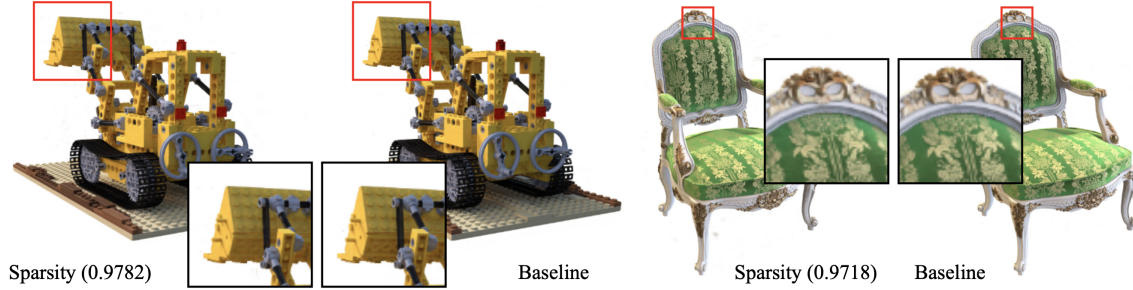
Figure 7. A qualitative analysis of grid sparsity. The sparsity refers to the ratio of zero elements in grids. With the help of trainable masks, removing more than 97% of grid parameters only results in an imperceptible difference.
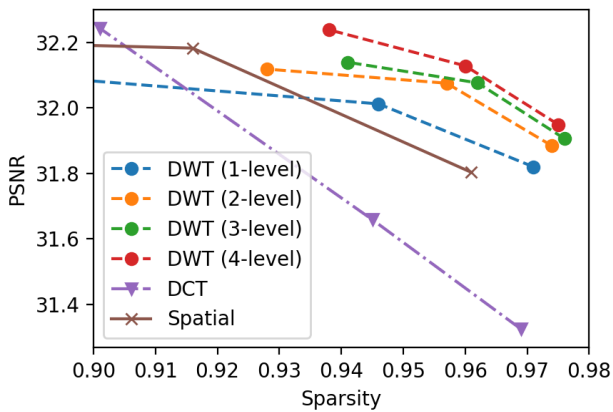

Figure 8. The rate distortion curves of different signal representation schemes (spatial, DWT, and DCT). Sparsity on the x axis refers to the ratio of zeros in grid parameters. The grid sparsity was controlled by λ_m. The numbers inside the parenthesis indicate the levels of the wavelet transform.

Wavelet Function	PSNR
Haar	31.889
Coiflets 1	31.846
Daubechies 4	31.734
biorthogonal 4.4 (default)	**31.949**
reverse biorthogonal 4.4	31.727

Table 2. Reconstruction performances of different wavelet functions.

fairly well. Still, the biorthogonal 4.4 that we used shows the best performance among selected wavelet functions.

4.3.4 Discrete cosine transform

In this section, we compare the performance of DWT with that of DCT, which represents a signal with respect to the sum of cosine functions with different frequencies. DCT, similar to DWT, has an energy compaction property. That is, a small number of non-zero DCT coefficients are sufficient to represent a given signal. Because of its energy compactness, DCT or its variants are another widely used signal transformation for image, audio, and audio compression. However, as shown in Fig. 8 and Tab. 1, DWT shows superior compression and representation performance to DCT. We believe this is due to the non-repeating, non-smooth information in grids, for which DWT is more appropriate.

5. Conclusion

We propose a compact representation for grid-based neural fields, enabled by a novel masking strategy and multi-level wavelet transform. We demonstrate that these two components made it possible to remove more than 95% of grid parameters without a significant loss of visual quality. With our proposed compression pipeline and the sparse wavelet coefficients, we achieved state-of-the-art performance under a memory budget of 2 MB. We believe we can reduce the size further with more sophisticated compression pipelines.

Our proposed method is naturally constrained by the limitations of grid-based neural fields that are developed for bounded scenes or objects. We believe that expanding this grid-based representation to encompass unbounded or large scenes would be an intriguing direction, given that we can now compactly represent 3D objects with our proposed representation scheme.

Acknowledgments

References

[1] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 3, 5

[2] Thomas Bird, Johannes Ballé, Saurabh Singh, and Philip A Chou. 3d scene compression through entropy penalized neural representation functions. In *2021 Picture Coding Symposium (PCS)*, pages 1–5. IEEE, 2021. 6

[3] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J. Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16123–16133, June 2022. 2, 3, 4

[4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European conference on computer vision*. Springer, 2022. 2, 3, 4, 6

[5] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021. 1

[6] Albert Cohen, Ingrid Daubechies, and J-C Feauveau. Biorthogonal bases of compactly supported wavelets. *Communications on pure and applied mathematics*, 45(5):485–560, 1992. 4

[7] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. 3

[8] Chenxi Lola Deng and Enzo Tartaglione. Compressing explicit voxel grid representations: fast nerfs become also small. *arXiv preprint arXiv:2210.12782*, 2022. 3

[9] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891, 2022. 1

[10] Ben Mildenhall et al. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019. 6

[11] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, SA '22, New York, NY, USA, 2022. Association for Computing Machinery. 1

[12] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. 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 1126–1135. PMLR, 06–11 Aug 2017. 1

[13] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5501–5510, June 2022. 2

[14] Stephan J. Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14346–14355, October 2021. 1

[15] Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. pages 5875–5884, October 2021. 2

[16] Tao Hu, Shu Liu, Yilun Chen, Tiancheng Shen, and Jiaya Jia. Efficientnerf efficient neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12902–12911, June 2022. 1

[17] Binbin Huang, Xinhao Yan, Anpei Chen, Shenghua Gao, and Jingyi Yu. Pref: Phasorial embedding fields for compact neural representations. *arXiv preprint arXiv:2205.13524*, 2022. 2, 3, 4

[18] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. 3

[19] David A Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952. 2, 6

[20] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4), jul 2017. 6

[21] Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. Brecq: Pushing the limit of post-training quantization by block reconstruction. *arXiv preprint arXiv:2102.05426*, 2021. 3

[22] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15651–15663. Curran Associates, Inc., 2020. 2, 6

[23] 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 *European conference on computer vision*, pages 405–421. Springer, 2020. 2, 4, 6

[24] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4), jul 2022. 2, 3

[25] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? Adaptive

rounding for post-training quantization. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7197–7206. PMLR, 13–18 Jul 2020. 3

[26] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018. 1

[27] Julian Ost, Issam Laradji, Alejandro Newell, Yuval Bahat, and Felix Heide. Neural point light fields. pages 18419–18429, June 2022. 2

[28] Songyou Peng, Chiyu Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. Shape as points: A differentiable poisson solver. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 13032–13044. Curran Associates, Inc., 2021. 2

[29] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of maml. *arXiv preprint arXiv:1909.09157*, 2019. 1

[30] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 525–542, Cham, 2016. Springer International Publishing. 3

[31] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14153–14161, 2021. 1

[32] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14335–14345, October 2021. 1, 3

[33] Katja Schwarz, Axel Sauer, Michael Niemeyer, Yiyi Liao, and Andreas Geiger. VoxGRAF: Fast 3d-aware image synthesis with sparse voxel grids. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. 2

[34] Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 10136–10147. Curran Associates, Inc., 2020. 1

[35] Athanassios Skodras, Charilaos Christopoulos, and Touradj Ebrahimi. The jpeg 2000 still image compression standard. *IEEE Signal processing magazine*, 18(5):36–58, 2001. 2, 3

[36] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012. 2

[37] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5459–5469, June 2022. 2

[38] Towaki Takikawa, Alex Evans, Jonathan Tremblay, Thomas Müller, Morgan McGuire, Alec Jacobson, and Sanja Fidler. Variable bitrate neural fields. In *ACM SIGGRAPH 2022 Conference Proceedings*, SIGGRAPH '22, New York, NY, USA, 2022. Association for Computing Machinery. 2, 3, 6

[39] 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. *arXiv preprint arXiv:2101.10994*, 2021. 2

[40] Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P. Srinivasan, Jonathan T. Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations. In *CVPR*, 2021. 1, 2

[41] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020. 3

[42] Jiaxiang Tang, Xiaokang Chen, Jingbo Wang, and Gang Zeng. Compressible-composable neRF via rank-residual decomposition. In *Advances in Neural Information Processing Systems*, 2022. 2, 3, 6

[43] Gregory K. Wallace. The jpeg still picture compression standard. *Commun. ACM*, 34(4):30–44, apr 1991. 2

[44] Huan Wang, Jian Ren, Zeng Huang, Kyle Olszewski, Menglei Chai, Yun Fu, and Sergey Tulyakov. R2l: Distilling neural radiance field to neural light field for efficient novel view synthesis. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 612–629, Cham, 2022. Springer Nature Switzerland. 1

[45] Liao Wang, Jiakai Zhang, Xinhang Liu, Fuqiang Zhao, Yanshun Zhang, Yingliang Zhang, Minye Wu, Jingyi Yu, and Lan Xu. Fourier plenoctrees for dynamic radiance field rendering in real-time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13524–13534, June 2022. 2, 3

[46] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the h. 264/avc video coding standard. *IEEE Transactions on circuits and systems for video technology*, 13(7):560–576, 2003. 2

[47] Zhijie Wu, Yuhe Jin, and Kwang Moo Yi. Neural fourier filter bank. *arXiv e-prints*, pages arXiv–2212, 2022. 3

[48] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022. 2, 3

[49] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5752–5761, October 2021. 2, 3

[50] Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. Differentiable point-based radiance fields for efficient view synthesis. *arXiv preprint arXiv:2205.14330*, 2022. 2

[51] Chenzhuo Zhu, Song Han, Huizi Mao, and William J. Dally. Trained ternary quantization. In *International Conference on Learning Representations*, 2017. 3