SHACIRA: Scalable HAsh-grid Compression for Implicit Neural Representations

Sharath Girish
University of Maryland
sgirish@cs.umd.edu

Abhinav Shrivastava
University of Maryland
abhinav@cs.umd.edu

Kamal Gupta
University of Maryland
kampta@cs.umd.edu

Abstract

Implicit Neural Representations (INR) or neural fields have emerged as a popular framework to encode multimedia signals such as images and radiance fields while retaining high-quality. Recently, learnable feature grids proposed by Müller et al. [1] have allowed significant speed-up in the training as well as the sampling of INRs by replacing a large neural network with a multi-resolution look-up table of feature vectors and a much smaller neural network. However, these feature grids come at the expense of large memory consumption which can be a bottleneck for storage and streaming applications. In this work, we propose SHACIRA, a simple yet effective task-agnostic framework for compressing such feature grids with no additional post-hoc pruning/quantization stages. We reparameterize feature grids with quantized latent weights and apply entropy regularization in the latent space to achieve high levels of compression across various domains. Quantitative and qualitative results on diverse datasets consisting of images, videos, and radiance fields, show that our approach outperforms existing INR approaches without the need for any large datasets or domain-specific heuristics. Our project page is available at https://shacira.github.io.

1. Introduction

In today's digital age, large quantities of data in different modalities (images, audio, video, 3D) is created and transmitted every day. Compressing this data with minimal loss of information is hence an important problem and a number of techniques have been developed in the last few decades to address this challenging problem. While the conventional methods such as JPEG [5] for images, HEVC [6] for videos excel at encoding signals in their respective domains, coordinate-based implicit neural representations (INR) or Neural Fields [7] have emerged as a popular alternative for representing complex signals because of their ability to capture high frequency details, and adaptability for diverse domains. INRs are typically multi layer perceptrons (MLPs) optimized to learn a scalar or vector field. They

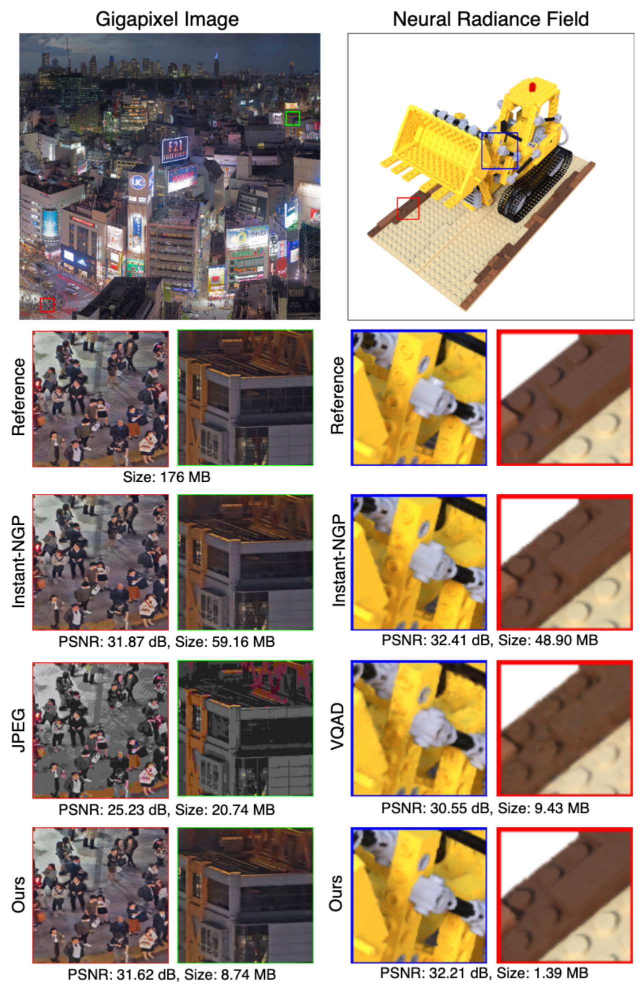

Figure 1: We demonstrate the effectiveness of SHACIRA for two tasks. The left column shows a gigapixel image at 21450×56718 resolution (cropped for visualization) encoded using Instant-NGP [1], JPEG2000 [2], and SHACIRA (ours). The right column reconstructs NeRF [3] from 2D images and their camera poses using Instant-NGP [1], VQAD [4], and SHACIRA. For each example, we zoom into two crops to compare different methods. We show overall PSNR and size required by each method. SHACIRA can capture high-resolution details with a smaller storage size in a task-agnostic way (only 2D/3D examples shown here).

take a coordinate (location and/or time) as input and predict a continuous signal value(s) as output (such as pixel color/occupancy). Recently, various works have adapted INRs to represent a variety of signals such as audio [8], images [9–12], videos [13, 14], shapes [15, 16], and radiance fields [3, 17]. Unsurprisingly, several methods have been proposed to compress INRs using quantization [10, 14, 18], pruning, or a combination of both [13]. The focus of these works is to compress the weights of the MLP, which often leads to either a big drop in the reconstruction quality, or slow convergence for high resolution signals.

In this work, we consider a different class of INR approaches that employ learnable multi-resolution feature grids [1, 19]. These feature grids store feature vectors at different coordinate locations with varying Level-Of-Detail (LOD). The features from different levels (or resolutions) are concatenated and passed through a tiny MLP to reconstruct the required signal. This shifts the burden of representing the signal to the feature grid instead of the MLP. Such methods have shown to be effective in approximating complex signals such as 3D scenes and gigapixel images with high fidelity [1] and fast training time (since the cost of lookup is very small). However, the size of the feature grids can be very large which is not memory efficient and impractical for many real-world applications with network bandwidth or storage constraints.

We propose an end-to-end learning framework for compressing such feature grids without any loss in reconstruction performance. Our feature grid consists of quantized feature vectors and parameterized decoders which transform the feature vectors into continuous values before passing them to MLP. We use an entropy regularization loss on the latent features to reduce the size of the discrete latents without significantly affecting the reconstruction performance. To address the discretization gap inherent to this discrete optimization problem, we employ an annealing approach to the discrete latents which improves the training stability of the latents, converging to better minima. Both entropy regularization and reconsutruction objective can be trained jointly in an end-to-end manner without requiring post-hoc quantization, pruning or finetuning stages. Further, the hierarchical nature of feature grids allows scaling to high dimensional signals unlike pure MLP-based implicit methods.

As seen in Figure 1, the proposed approach is able to compress feature-grid methods such as Instant-NGP [1] with almost an order of magnitude while retaining the performance in terms of PSNR for gigapixel images and 3D scenes from the RTMV dataset [20]. We also conduct extensive quantitative experiments and show results on standard image compression benchmarks such as the Kodak dataset outperforming the classic JPEG codec as well as other implicit methods in the high compression regime. Our approach can even be trivially extend to videos, performing competitively with video-specific INR methods such as NeRV [13], without explicitly exploiting the inherent temporal redundancy present in videos. The key contribution of our work is to directly compress the learnable feature grid with proposed entropy regularization loss and highlight its adaptibility to diverse signals. We summarize our contributions below:

- We introduce an end-to-end trainable compression framework for implicit feature grids by maintaining discrete latent representations and parameterized decoders.
- We provide extensive experiments on compression benchmarks for a variety of domains such as images, videos, and 3D scenes showing the generalizability of our approach outperforming a variety of INR works.

2. Related work

Learned image/video compression: A large number of neural compression works for images consist of an autoencoder framework [21, 22] which transform/encode a data point to a latent code and decode the quantized latent to obtain a reconstruction of the data signal. The autoencoder is typically trained in an end-to-end fashion on a large training dataset by minimizing a rate-distortion objective. Numerous extensions to these works introduce other improvements such as hyperpriors [23], autoregressive modeling [24], Gaussian mixture likelihoods and attention modules [25], improved inference [26]. Another set of works extends this framework for videos as well [27–30] exploiting the inherent temporal redundancy. These approaches achieve impressive compression results outperforming classical codecs in their respective domains. In contrast, we focus on a different class of works involving implicit neural representations (INRs) which overfit a network to each datapoint and store only the network weights to achieve data compression.

INRs and application to data compression: INRs [31] are a rapidly growing field popularized for representing 3D geometry and appearance [3, 15, 32, 33] and have since been applied to a wide variety of fields such as GANs [34, 35], image/video compression [9, 10, 13, 14, 18], robotics [36] and so on. Meta-learning on auxiliary datasets has been shown to provide good initializations and improvements in reconstruction performance while also greatly increasing convergence speed for INRs [11, 16, 18]. Our approach can similarly benefit from such meta-learning stages but we do not focus on it and rather highlight the effectiveness of our approach to compress feature-grid based INRs and its advantages over MLP-based INRs. Perhaps the closest work to our approach is that of VQAD [4] which performs

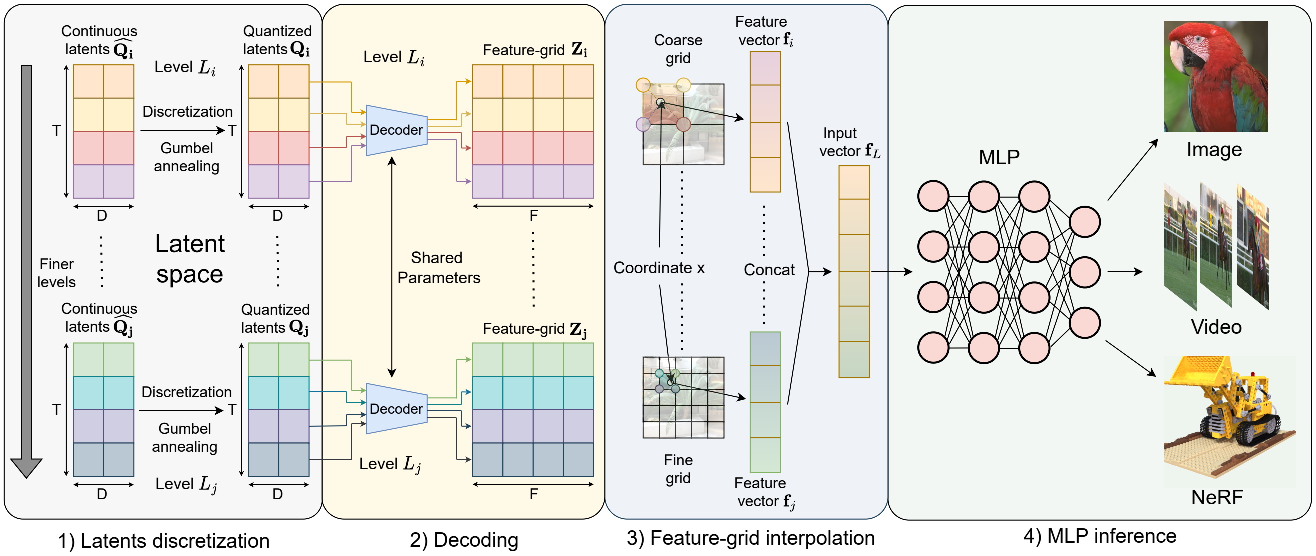

Figure 2: Overview of our approach: We maintain latent representations which are quantized and decoded using parameterized decoders to obtain a hash table/feature-grid at different levels. We then index the input coordinate into the hash table to obtain feature vectors. The feature vectors are then concatenated and passed through an MLP to obtain the output signal.

a vector quantization of these feature grids learning a codebook/dictionary and its mapping to the feature grid in 3D domain. They however learn a fixed-size codebook without any regularization loss and do not perform well for high-fidelity reconstructions as we discuss in Section 4.

Model compression: As INRs represent data as neural networks, they transform the data compression problem to a model compression one. Many works exist for model compression involving pruning for achieving high levels of sparsity [37–40] or quantization for reducing the number of bits necessary [41–43]. Another line of works [44, 45] perform compression similar to [23] using quantized latents with entropy regularization losses. These methods, however, are specific to convolutional networks and are not trivially extensible to compress INRs.

3. Approach

Our goal is to simultaneously train and compress feature-grid based implicit neural networks. Section 3.1 provides a brief overview of feature-grid INRs proposed in [1]. Section 3.2 describes various components of our approach while Section 3.3 discusses compressing feature grids. Our approach for end-to-end feature grid compression is outlined in Section 3.4 and also illustrated in Figure 2.

3.1. Feature-grid INRs

INRs or neural fields [7] typically learn a mapping $g_\phi(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}^c$ where g is an MLP with parameters ϕ. Input $\mathbf{x} \in \mathbb{R}^d$ to the MLP are d dimensional coordinates, where $d = 2$ in the case of images, 3 in the case of videos, or 5 in the case of radiance fields. c-dimensional output can represent RGB colors or occupancy in space-time for the given input coordinates. Such methods are able to achieve high-quality reconstructions, however, suffer from long training times and scalability to high-resolution signals. A few works have suggested utilizing fixed positional encodings of input coordinates [46] to be able to reconstruct complex high-resolution signals but the training time of these networks still poses a big challenge. To alleviate this issue, [1, 19] proposed storing the input encodings in the form of a learnable feature grid. The feature grid allows for significant speed-up in the training time by replacing a large neural network with a multi-resolution look-up table of feature vectors and a much smaller neural network. We now provide a brief overview of Instant-NGP [1].

For ease of explanation, for the rest of this section, we will assume input is a 2D signal $\mathbf{x} \in \mathbb{R}^2$. However, all the techniques we discuss can be directly applied to the 3D case. In this framework, we represent the feature grid by a set of parametric embeddings $\mathbf{Z}$. $\mathbf{Z}$ is arranged into L levels representing varying resolutions or Levels-Of-Detail (LOD). More formerly, each level has its own embedding matrix $\mathbf{Z}_l$, and hence $\mathbf{Z} = \{\mathbf{Z}_1, \ldots, \mathbf{Z}_l\}$. The number of feature vectors in the embedding matrix $\mathbf{Z}_l$ depends on the resolution of the level R_l. For coarser resolutions, we allow $\mathbf{Z}_l$ to consist of R_l^2 rows, but for finer resolutions, we cap the maximum number of rows in the matrix to T. Hence for

$\mathbf{x} \in \mathbb{R}^2$,

$$\mathbf{Z}_l \in \mathbb{R}^{T_l \times F}, \text{ where } T_l = \min(R_l^2, T) \quad (1)$$

Here F is the dimension of feature vectors and is kept fixed for all levels. For a given input $\mathbf{x}$, we can obtain the 4 closest corner indices $\{tl, tr, bl, br\}$ within the $R_l \times R_l$ grid. Each of the corner index maps to an entry in $\mathbf{Z}_l$. This mapping is direct when $R_l^2 \leq T$ and uses a hashing function [47] otherwise. The feature vector $\mathbf{f}_l$ at level l for the input $\mathbf{x}$ is then obtained by a simple bilinear interpolation of feature vectors of the corner indices, *i.e*.

$$\mathbf{f}_l(\mathbf{x}) = \text{interp}(\mathbf{Z}_l[tl], \mathbf{Z}_l[tr], \mathbf{Z}_l[bl], \mathbf{Z}_l[br]) \quad (2)$$

Note that in the case of 3D input, we consider 8 closest corner indices, and perform a trilinear interpolation. $\mathbf{f}_l(\mathbf{x})$ is concatenated across different levels to obtain the overall feature vector $\mathbf{f}(\mathbf{x}) \in \mathbb{R}^{FL}$, which is passed as input to the neural network g_ϕ.

$$\widehat{\mathbf{y}} = g_\phi(\text{concat}[\mathbf{f}_1(\mathbf{x}), \ldots, \mathbf{f}_L(\mathbf{x})]) \quad (3)$$

Here $\widehat{\mathbf{y}}$ is the final prediction of INR for the input $\mathbf{x}$. Since the concatenation, indexing, and interpolation operations are differentiable, parameters $\{\phi, \mathbf{Z}\}$ can be optimized using any gradient based optimizer by minimizing a loss function $\mathcal{L}(\widehat{\mathbf{y}}, \mathbf{y})$ between the predicted $\widehat{\mathbf{y}}$ and ground-truth signal $\mathbf{y}$. $\mathcal{L}$ can be any differentiable loss function such as the Mean Squared Error (MSE). The MLP is typically very small in comparison to MLP-based INRs. Thus, ϕ consists of far fewer parameters than $\mathbf{Z}$. Such an INR design allows for much faster training and inference as the cost of indexing into the feature grid is quite small.

3.2. Feature-grid reparameterization

While feature-grid INRs can converge faster than pure-MLP approaches, the space required to store all the parameters at different levels can rise very rapidly if we want high-fidelity reconstructions for high-resolution inputs. This makes them unsuitable for resource-constrained applications. We thus aim to reduce the storage size of these feature grids. To this effect, we propose to maintain discrete or quantized latent representations $\mathbf{Q}_l \in \mathbb{R}^{T_l \times D}$ for each embedding $\mathbf{Z}_l \in \mathbb{R}^{T_l \times F}$. The latents, consisting of only integer values, can be of any dimension D with a larger D allowing for greater representation power at the cost of storage size.

In order to map these discrete latent features $\mathbf{Q}_l$ to the continuous features in the embedding table $\mathbf{Z}_l$, we propose a parameterized decoder $d_\theta : \mathbb{R}^D \to \mathbb{R}^F$. While it is possible to use separate complex decoders for each level, in practice, we observed that a single shared decoder parameterized as a linear transform across all L levels works pretty well and has a minimal impact on the training/inference times and fidelity of reconstructions.

Note that the quantized latents $\mathbf{Q}$ are no more differentiable (here we dropped the subscript l without loss of generality for notational simplicity). In order to optimize these quantized latents, we maintain a continuous proxy parameters $\widehat{\mathbf{Q}}$ of the same size as $\mathbf{Q}$. $\mathbf{Q}$ is obtained by rounding $\widehat{\mathbf{Q}}$ to the nearest integer. To make this operation differentiable, we utilize the Straight-Through Estimator [48](STE). In STE, we use quantized weights $\mathbf{Q}$ during the forward pass, but use the continuous proxies to propagate the gradients from $\mathbf{Q}$ to $\widehat{\mathbf{Q}}$ during the backward pass.

STE serves as a simple differentiable approximation to the rounding operation but leads to a large rounding error $\|\mathbf{Q} - \widehat{\mathbf{Q}}\|$. To overcome this issue, we utilize an annealing approach [26] to perform a soft rounding operation. We represent $\mathbf{Q}$ by either rounding up (denoted as $\lceil . \rceil$) or down (denoted as $\lfloor . \rfloor$) $\widehat{\mathbf{Q}}$ to the nearest integer. Using one-hot gates $b \in \{0, 1\}$, where $b = 0$ corresponds to rounding up, $b = 1$ corresponds to rounding down, we can represent $\mathbf{Q} = b\lfloor\widehat{\mathbf{Q}}\rfloor + (1-b)\lceil\widehat{\mathbf{Q}}\rceil$. The gate b is sampled from a soft relaxed 2-class distribution

$$\begin{aligned}
\text{Prob}(b=0) &\propto \exp\left\{-\tanh^{-1}\left(\widehat{\mathbf{Q}} - \lfloor\widehat{\mathbf{Q}}\rfloor\right)/\tau\right\} \\
\text{Prob}(b=1) &\propto \exp\left\{-\tanh^{-1}\left(\lceil\widehat{\mathbf{Q}}\rceil - \widehat{\mathbf{Q}}\right)/\tau\right\}
\end{aligned} \quad (4)$$

where $\tau > 0$ represents the temperature parameter. $\widehat{\mathbf{Q}}$ approaching either $\lfloor\widehat{\mathbf{Q}}\rfloor$ or $\lceil\widehat{\mathbf{Q}}\rceil$ thus increases the likelihood of sampling to that respective value. In the beginning of the training, $\tau = 1$ and is a poor approximator of rounding operation but provides more stable gradients. As training progresses, τ is annealed towards zero, so that the random distribution converges to a deterministic one at the end of training. The gradients are propagated through the samples using the Gumbel reparameterization trick [49].

3.3. Feature-grid compression

To further improve compression levels, we minimize the entropy of the latents using learnable probability models [23]. We note that our learned latents $\mathbf{Q}$ or $\widehat{\mathbf{Q}}$ are of dimensions $T \times D$. We can interpret these latents as consisting of T samples from a discrete D dimensional probability distribution. We introduce D probability models one for each latent dimension $d \in \{1, \ldots, D\}$, $P_d : \mathbb{R} \to [0, 1]$. We discuss the exact form of P_d in the supplementary material. With these probability models, we can minimize the length of a bit sequence encoding $\widehat{\mathbf{Q}}$ by minimizing the self-information loss or the entropy of $\widehat{\mathbf{Q}}$ [50]

$$\mathcal{L}_I(\widehat{\mathbf{Q}}) = -\frac{1}{T}\sum_{d=1}^{D}\sum_{i=1}^{T}\log_2\left(P_d\left(\widehat{\mathbf{Q}}[i,d] + n\right)\right) \quad (5)$$

where $n \sim \mathcal{U}[-1,1]$ represents the uniform random distribution to approximate the effects of quantization, and $\mathcal{L}_I$ is the self-information loss.

3.4. End-to-end optimization

We provide an overview of our approach in Fig. 2. We maintain continuous learnable latents $\widehat{\mathbf{Q}}$. The proposed annealing approach (Sec. 3.1) progressively converges the continuous $\widehat{\mathbf{Q}}$ to the discrete $\mathbf{Q}$. The approach is made differentiable using the gumbel reparameterization trick and straight-through estimator. $\mathbf{Q}$ is passed through the decoder d_θ with parameters θ to obtain the feature grid table $\mathbf{Z}$. We then index $\mathbf{Z}$ at different levels/resolutions using the coordinates $\mathbf{x}$ to obtain a concatenated feature vector $\mathbf{f}$ which is passed through an MLP g_ϕ to obtain the predicted signal $\widehat{\mathbf{y}}$.

$$\mathbf{Q} = \text{discretize}(\widehat{\mathbf{Q}}) \tag{6}$$
$$\mathbf{Z} = d_\theta(\mathbf{Q}) \tag{7}$$
$$\widehat{\mathbf{y}} = g_\phi\left(\text{concat}\left[\text{interp}(\mathbf{Z}, \mathbf{x})\right]\right) \tag{8}$$

Our framework is thus fully differentiable in the forward and backward pass. For a given signal $\mathbf{y}$ and its corresponding input coordinate grid $\mathbf{x}$, we optimize the parameters ϕ of MLP g_ϕ, discrete feature grid $\widehat{\mathbf{Q}}$, discrete to continuous decoder θ, and the probability models $\{P_d\}$ in an end-to-end manner by minimizing the following rate distortion objective

$$\mathcal{L}_{\text{MSE}}(\widehat{\mathbf{y}}, \mathbf{y}) + \lambda_I \mathcal{L}_I(\widehat{\mathbf{Q}}) \tag{9}$$

where λ_I controls the rate-distortion optimization trade-off. Post training, the quantized latents are stored losslessly using algorithms such as arithmetic coding [51] utilizing the probability tables from the density models $\{P_d\}$.

4. Experiments

We apply our INR framework to images, videos, and radiance fields. We outline our experimental setup in Sec. 4.1. Sec. 4.2, Sec. 4.3, and Sec. 4.4 provide results on compression for images, radiance fields, and videos respectively. Sec. 4.5 illustrates the application of our approach for progressive streaming. Sec. 4.6 discusses the convergence speeds of our approach. Sec. 4.7 ablates the effect of entropy regularization and annealing. Additional experiments and ablations are provided in the supplementary material.

4.1. Experimental details and setup

We show image compression performance on the Kodak dataset [52] consisting of 24 images of resolution 512×768. To show our scalability to larger images, we provide results on higher resolution images varying from 2M pixels to 1G pixels. We use the Kaolin-Wisp library [53] for our experiments on radiance fields. In addition to the Lego scene

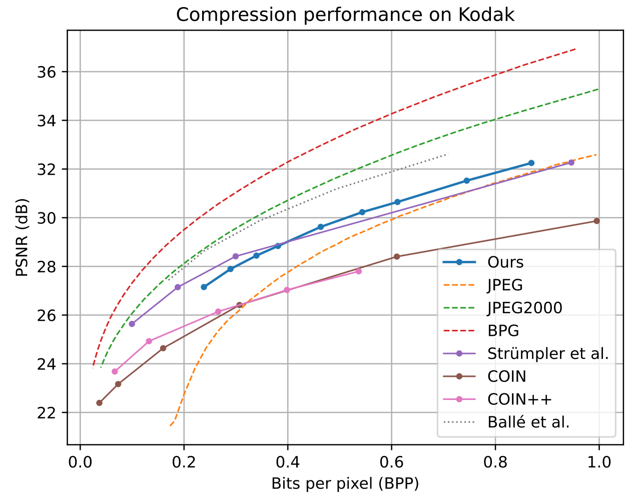

Figure 3: Comparison of our approach on the Kodak image dataset with classical (dashed), RDAE (dotted), INR (solid) methods. We outperform state-of-the-art INR approaches bridging the gap to classical and RDAE methods.

in Fig. 1, we provide results on 10 bricks scenes from the RTMV dataset [20] which contains a wide variety of complex 3D scenes. For videos, we benchmark on the UVG dataset [54] consisting of seven 1080p resolution videos consisting of 600 frames each at 120fps. Additionally, we extract the first frame from these 7 videos to create a 7-image dataset, UVG-F, for benchmarking image compression at 1080×1920 resolution against other popular INR compression approaches. We primarily measure distortion in terms of the PNSR (dB) and rate in terms of Bits-Per-Pixel (BPP) (or model size for neural fields).

We fix the minimum grid resolution to 16 and vary the number of per-level entries T, the number of levels L, and the maximum grid resolution for various modalities to control the rate/distortion trade-off. We fix the number of hidden MLP layers to 2 with the ReLU activation. We fix the batch size (number of coordinates per iteration) to be 2^{18} for all our experiments except Kodak where we pass the full coordinate grid. The entropy regularization parameter is set to $1.0e^{-4}$ for all our experiments unless specified otherwise. The parameters are optimized using the Adam optimizer [55]. We provide additional experimental details in the supplementary material.

4.2. Scalable image compression

We visualize results on the Kodak benchmark in Fig. 3. We outperform the MLP-based INR approaches of COIN [9], COIN++ [10] by a significant margin at all bitrates. We outperform [18], which utilizes positional encodings and weight quantization, at higher bitrates while also having much lesser encoding times (as we show in

Table 1: Image compression at varying resolutions. We compare against implicit network methods of INGP [1], Positional [18], SIREN [8], and the auto-encoder based work, RDAE [23]. We achieve high values of PSNR, maintaining similar quality reconstructions as INGP while requiring far fewer bits ($4 - 9\times$).

Image	Method	PSNR ↑	SSIM ↑	BPP ↓
UVG-F (1920×1080)	Positional [18]	33.17	0.86	1.52
	JPEG [5]	36.98	0.91	0.76
	RDAE [23]	34.23	**0.93**	0.76
	Ours	**37.74**	0.92	**0.76**
SMACS (4630×4537)	INGP [1]	34.61	0.86	0.18
	JPEG [5]	34.77	0.86	0.18
	RDAE [23]	34.06	**0.89**	0.40
	Ours	**34.90**	0.86	**0.04**
Cosmic-Cliffs (8441×14575)	INGP [1]	38.78	0.96	0.63
	SIREN [8]	27.32	0.90	0.14
	JPEG [5]	38.38	0.95	0.29
	RDAE [23]	37.90	**0.97**	0.38
	Ours	**38.79**	0.96	**0.11**
Pearl (23466×20000)	INGP [1]	**29.62**	0.84	1.00
	JPEG [5]	29.10	0.84	0.29
	Ours	29.44	**0.84**	**0.12**
Tokyo (21450×56718)	INGP [1]	**31.87**	**0.82**	0.39
	JPEG [5]	31.16	0.82	0.18
	Ours	31.62	0.82	**0.06**

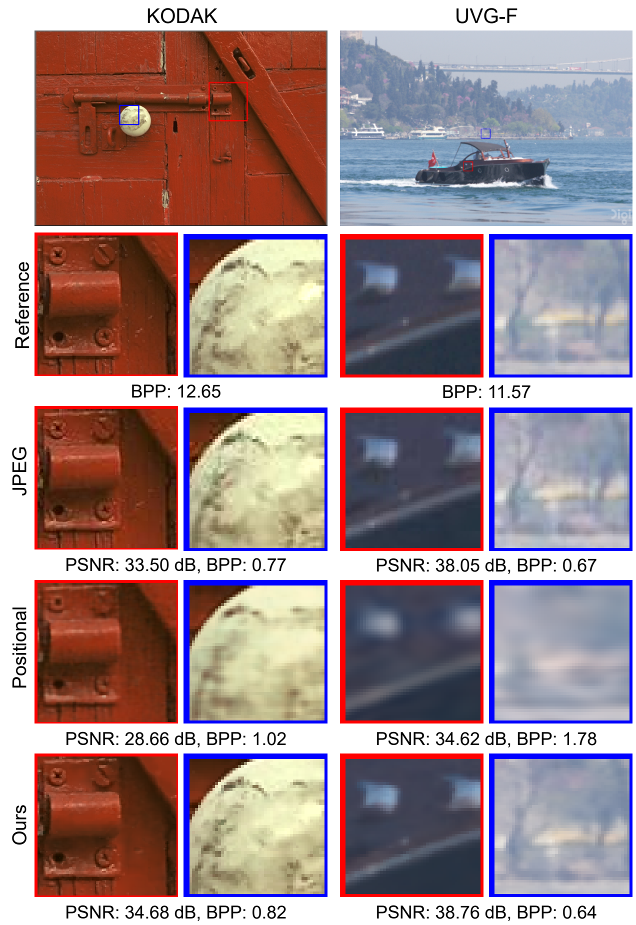

Figure 4: Qualitative results on Kodak and UVG-F: We obtain much higher quality reconstructions capturing finer detail compared to [18] or JPEG. Notice the blocking and discoloration artifacts present in JPEG which are significant at lower BPP values.

Sec. 4.6) We show qualitative results on one of the images from the KODAK dataset in Fig. 4. We obtain higher quality reconstructions ($28.66 \rightarrow 34.68$ PSNR) capturing fine details while also requiring fewer bits for storage ($1.02 \rightarrow 0.88$ BPP) as compared to [18]. We however, observe slightly lower performance in the low BPP regime where uncompressed MLP weights represent a significant fraction($\sim$25%) of the total size. We hypothesize that at lower bit rates, compression of MLPs can achieve further reduction in model size. Our work focuses only on the compression of feature grids and can potentially benefit from the complimentary works on MLP compression.

We additionally outperform JPEG for the full range of BPP with a much larger gap at lower bitrates. Notice the blocking artifacts in Fig. 4 for JPEG while we achieve consistent reconstructions. However, the classical methods of JPEG2000, BPG and the autoencoder based work of [23] continue to outperform our method, especially in the higher BPP regime. Nevertheless, we reduce the gap between INRs and autoencoder in the low-dimensional image space.

In order to understand the scalability of various image compression methods with image resolution, we compress images in the UVG-F dataset (1920×1080) and 4 images with increasing resolution. Results are summarized in Table 1. We outperform [18] by a large margin on the UVG-F dataset with an improvement of over 4.5dB while also requiring $2\times$ fewer bits. This is also observed in Fig. 4 (right column) where we capture much finer high frequency details compared to the positional encoding approach of [18]. We marginally outperform JPEG as well in the high BPP regime which again exhibits minor blocking artifacts. Perhaps, the most surprising result is the performance of the Rate-Distortion AutoEncoder (RDAE) based approach [23] which does not scale very well to large dimensions. We obtain a 3.5dB improvement in PSNR while maintaining a similar BPP albeit with slightly lower SSIM score.

For higher image resolutions, we continue to observe negligible drops in performance compared to INGP [1] while achieving $4-9\times$ smaller bitrates. We qualitatively visualize results on the Cosmic-cliffs image (with a resolution of 8441×14575) in Fig. 5. We achieve a similar PSNR as Instant-NGP (38.78 dB) with $\sim 6\times$ reduction in bitrates. MLP-based INRs such as SIRENs [8] are unable to cap-

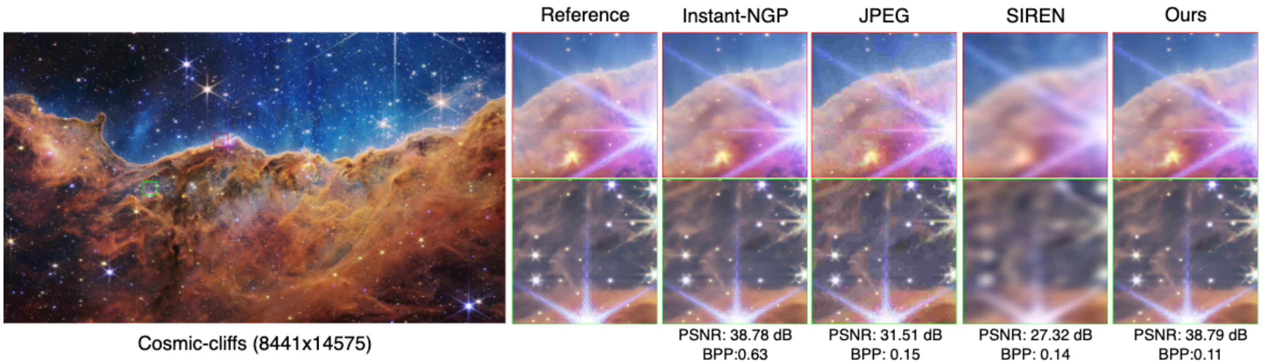

Figure 5: Compression result visualization on the Cosmic-Cliffs image for 4 methods. We obtain similar PSNR and reconstruction quality as Instant-NGP while ∼6× smaller. SIREN fails to fit high frequency information leading to blurry patches as seen. JPEG on the other hand suffers from blocking artifacts and discoloration leading to drop in reconstruction quality.

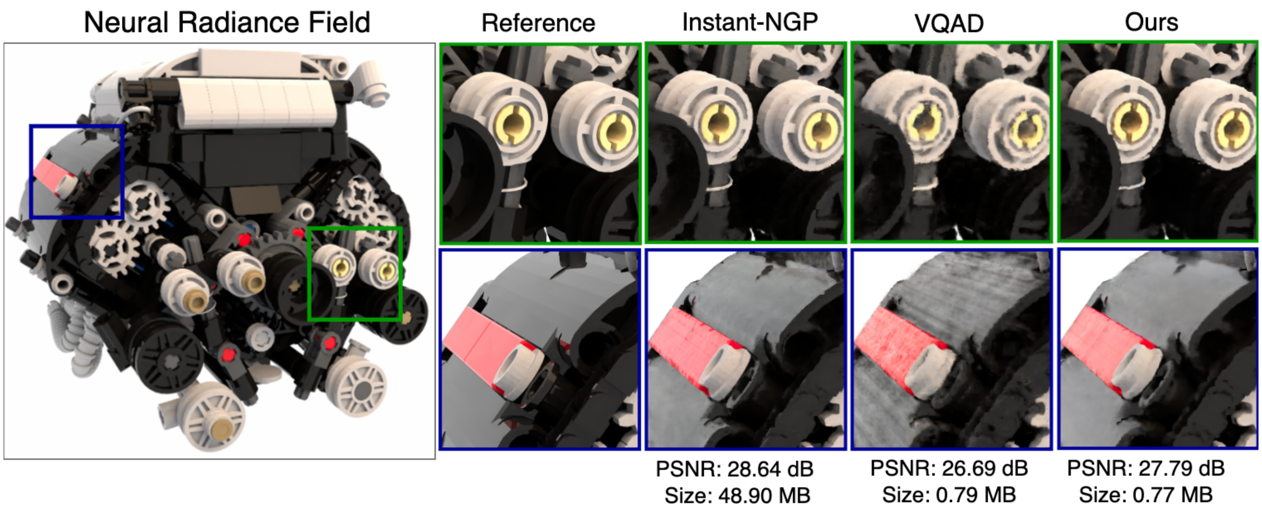

Figure 6: Evaluation on V8 from the RTMV dataset (1600 × 1600 resolution). We obtain ∼60× compression compared to Instant-NGP but with a PSNR drop of $1dB$. We outperform VQAD obtaining higher PSNR at similar model size. We also obtain much finer reconstructions when compared with VQAD as shown in the zoomed patches.

ture the high frequency detail resulting in blurry images and achieve low PSNR (27.32 dB). JPEGs also lead to a large drop in performance in the similar low BPP regime(∼0.15).

4.3. Radiance fields compression

We now turn to the application of SHACIRA to neural radiance fields or NeRFs. We compare our approach against the baseline Instant-NGP [1], mip-NERF [56] and VQAD [4], a codebook based feature-grid compression approach. We evaluate on 10 brick scenes from the RTMV dataset by training each approach for 600 epochs and summarize the results in Table 3. We marginally outperform the baseline INGP in terms of all the reconstruction metrics of PSNR, SSIM and LPIPS while also achieving ∼48× compression requiring an average of only 1MB per scene. We also outperform mip-NERF performing better on PSNR and SSIM while reducing the storage size. For a better comparison with VQAD (based off of NGLOD-NERF [19]), we scale down T, the maximum number of hashes. We see that we obtain $> 20\%$ lower model size at 0.43 MB compared to 0.55 MB of VQAD while slightly better in the reconstruction metrics. We also see a clear improvement over VQAD for the LEGO NeRF scene visualized in Figure 1. VQAD has around 1.5dB PSNR drop while still being ∼7× larger in terms of storage size. Additionally, VQAD fails to scale for higher bitwidth due to memory constraints even on an NVIDIA RTX A100 GPU with 40GB GPU RAM.

To better illustrate the 3 approaches of INGP, VQAD and SHACIRA, we train on the full resolution V8 scene (1600 × 1600) for 600 epochs, visualizing the results in Fig. 6. We outperform VQAD (+1dB) at similar model size and obtain ∼60× compression compared to INGP, but with 1dB drop in PSNR. Nevertheless, we reconstruct the scene with fewer artifacts and consistency in the intricate

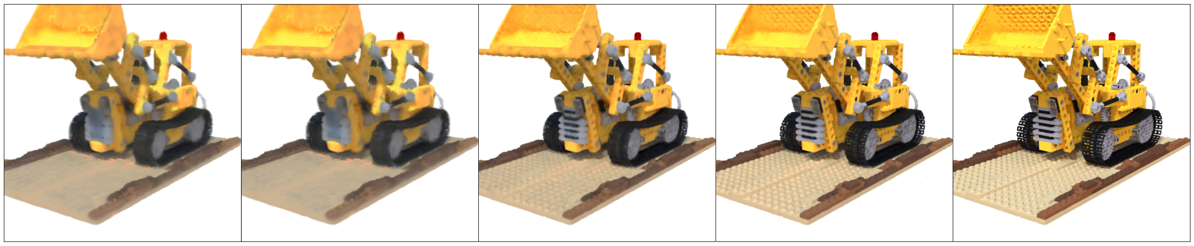

Figure 7: Our multiresolution compressed representations can be transmitted at varying LODs at inference time (without any retraining) making it suitable for applications with progressive streaming.

Table 2: Comparison against various video INR approaches on the UVG dataset. We outperform NIRVANA with higher PSNR and lower BPP while obtaining slightly lower PSNR than NeRV at 3× reduction in model size.

Method	Encoding Time ↓	PSNR ↑	BPP ↓
SIREN	15 hours	27.20	0.28
NeRV	3.5 hours	**35.54**	0.66
NIRVANA	4 hours	34.71	0.32
Ours	6.5 hours	35.01	**0.21**

Table 3: Comparison of various methods on scenes in the RTMV dataset. We marginally outperform the baseline Instant-NGP in terms of all metrics while also achieving a 48× compression.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	Storage ↓
NeRF	28.28	0.9398	0.0410	2.5MB
mip-NERF	**31.61**	**0.9582**	**0.0214**	**1.2MB**
NGLOD-NERF	**32.72**	**0.9700**	**0.0379**	≈20MB
VQAD	31.45	0.9638	0.0468	0.55MB
Ours	31.46	0.9657	0.0428	**0.43MB**
Instant-NGP	31.88	0.9690	0.0381	48.9MB
Ours	**32.14**	**0.9704**	**0.0348**	**1.03MB**

shapes, compared to VQAD as highlighted in the patches.

4.4. Video compression

Next, we apply our approach to video compression as well. As a baseline, we compare against SIREN a coordinate-based INR. We also compare against NeRV [13], a video INR based approach which takes a positional encoding as input and predicts frame-wise outputs for a video. We also compare against another video INR, NIRVANA [14], which is an autoregressive patch-wise prediction framework for compressing videos. We run the baselines on the UVG dataset, with the results shown in Table 2.

We outperform SIREN by a significant margin with an almost $+7dB$ gain in PSNR and 25% lesser BPP and shorter encoding times. This is to be expected as SIREN fails to scale to higher dimensional signals such as videos usually with more than 10^9 pixels. We also outperform NIRVANA achieving higher PSNR and lower BPP albeit at longer encoding times. Compared to NeRV, we obtain a 0.5dB PSNR drop but achieve 3× compression in model size. We would like to add that our current implementation utilizes PyTorch and the encoding time can be reduced significantly using fully fused CUDA implementation ([1] demonstrated that efficient lookup of the hashtables can be an order of magnitude faster than a vanilla Pytorch version). Additionally, our approach is orthogonal to both baselines and provides room for potential improvements. For instance, compressed multi-resolution feature grids can be used to replace positional embedding for NeRV as well as coordinate-based SIREN, and provide faster convergence and better reconstruction for high dimensional signals.

4.5. Streaming LOD

[4] show the advantage of feature-grid based INRs for progressive streaming at inference time due to their multi-resolution representation capabilities. Since our compression framework consists of latents at different LODs as well, they can be progressively compressed with varying LOD yielding reconstructions with different resolution. Formally, for $\mathbf{Q} = \{\mathbf{Q}_1, \mathbf{Q}_2, ...\mathbf{Q}_L\}$ we can reconstruct the signal at LOD l by only passing the first l latents while masking out the finer resolutions. This can be applied directly during inference without any additional retraining. We visualize the effect of such progressive streaming in Fig. 7. We obtain better reconstruction with increasing bitrates or the latent size. This is especially beneficial for streaming applications as well as for easily scaling the model size based on the storage or bandwidth constraints.

4.6. Convergence speeds

We now compare the convergence speeds of our feature-grid based approach with that of [18] which is an MLP-based INR with positional encoding. We summarize the results for an image in the Kodak dataset in Fig.8. We measure encoding times on an NVIDIA RTX A6000 GPU for the full length of the training for both approaches. We obtain higher PSNR at a much faster rate than [18] at a similar BPP range (hue value in color bar). While [18] reduces BPP (0.84 at 600s) with higher encoding times, their PSNR remains stagnant at 32.5dB. In contrast, our approach

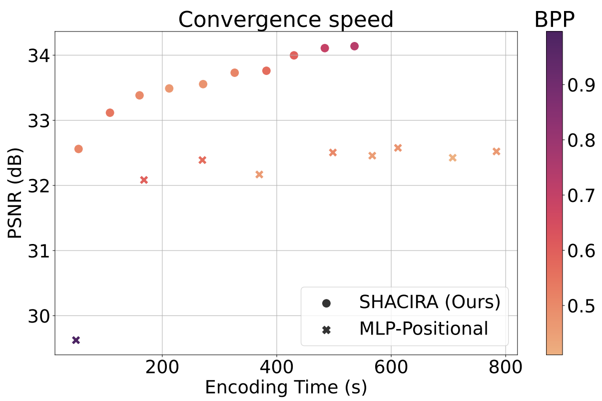

Figure 8: Comparison of convergence speeds of our feature-grid approach with [18] utilizing MLP-based INRs. We obtain higher PSNR at a much faster rate for a similar BPP range.

achieves this PSNR and BPP (0.85) within just 53s achieving more than a $10\times$ speedup in convergence. Additionally, we achieve higher PSNR with longer encoding times reaching 34dB in 430s while maintaining a similar BPP (0.87).

4.7. Effect of entropy regularization and annealing

In this section, we analyze the effect of entropy regularization and annealing. We pick the Jockey image from UVG-F (1080×1920) for our analysis. We set the default values of latent and feature dimensions to 1. For the analysis, we compare trade-off curves by increasing the number of entries from 2^{13} to 2^{17} in multiples of 2 which naturally increases the number of parameters and also the PSNR and BPP. Note that better trade-off curves indicate shifting upwards (higher PSNR) and to the left (lower BPP).

Figure 9 shows the effect of entropy regularization. The absence of it, corresponding to a value of 0.0 shows a drop in performance compared to values of $1.0e^{-4}$ and higher. We see that the network performance is fairly stable in this range with much higher values of $4.0e^{-4}$ showing small drops. This shows that entropy regularization using the defined probability models helps in improving the PSNR-BPP trade-off curve by decreasing entropy (or model size/BPP) with no drop in network performance (PSNR).

Figure 10 shows the effect of annealing. We vary the duration of annealing as a fraction of the total training duration. No annealing corresponds to the value 0.0 and has a large drop in the trade-off curve compared to higher values. This shows that annealing performs better than standard STE (Straight Through Estimator) alone and is an important component of our approach for quantizing the feature grid parameters. Increasing the period of annealing shows consistent improvements in the trade-off curve.

5. Conclusion

We proposed SHACIRA, a general purpose, end-to-end trainable framework for learning compressed INRs or neural fields for images, videos, and NeRFs. Our method adds a minimal overhead over feature-grid based methods of training INRs. We make this possible by parameterizing the feature grid of INRs with discrete latents and a decoder. Applying entropy regularization on discrete latents ensures that we can learn and maintain compressed quantized weights. A tiny decoder allows us to still operate in the continuous space for training and sampling with high fidelity. Our method requires no posthoc training or quantization and can be easily added to existing INR pipelines. We conduct an extensive evaluation on images, videos, and 3D datasets to show the efficacy of our approach in these domains. Compared to MLP-based INRs, we scale well for high resolution signals capturing high frequency details and converge faster. An additional benefit of our approach is that it allows reconstruction of the signal at different levels-of-details without retraining, which is especially beneficial for streaming applications. While the compressed network is memory efficient, it does not offer any inference speedups compared to the uncompressed network and is an interesting avenue for further research. Meta learning the feature grid for even faster convergence is another direction for future work.

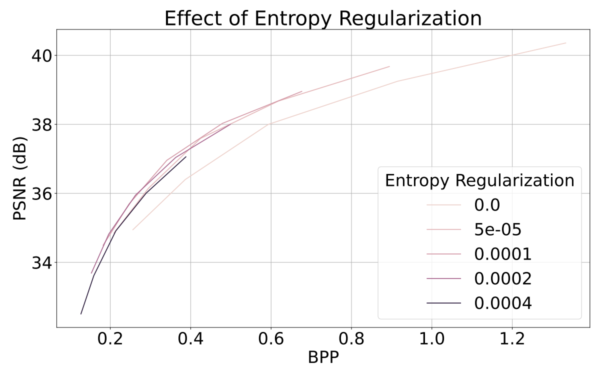

Figure 9: Effect of entropy regularization. In absence of entropy regularization, corresponding to the value 0.0, there is a drop in the PSNR-BPP trade-off curve.

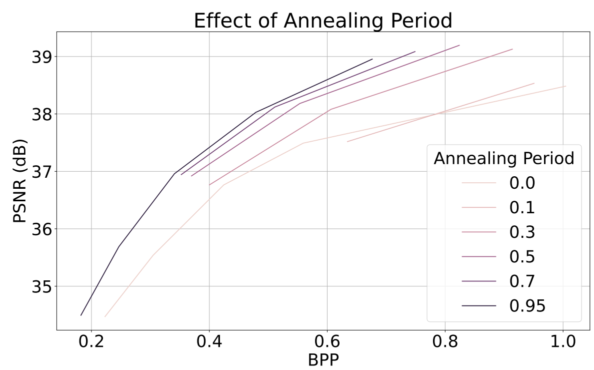

Figure 10: Effect of annealing. Increasing the fractional period of annealing improves the trade-off curve highlighting its importance for quantization approximation compared to stand-alone STE (Straight Through Estimator) with an annealing period of 0.0.

References

[1] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 1, 2, 3, 6, 7, 8

[2] David S Taubman and Michael W Marcellin. Jpeg2000: Standard for interactive imaging. *Proceedings of the IEEE*, 90(8):1336–1357, 2002. 1

[3] 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. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2

[4] 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*, pages 1–9, 2022. 1, 2, 7, 8

[5] Gregory K Wallace. The jpeg still picture compression standard. *IEEE transactions on consumer electronics*, 38(1):xviii–xxxiv, 1992. 1, 6

[6] Vivienne Sze, Madhukar Budagavi, and Gary J Sullivan. High efficiency video coding (hevc). In *Integrated circuit and systems, algorithms and architectures*, volume 39, page 40. Springer, 2014. 1

[7] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *Computer Graphics Forum*, 2022. 1, 3

[8] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020. 2, 6

[9] Emilien Dupont, Adam Goliński, Milad Alizadeh, Yee Whye Teh, and Arnaud Doucet. Coin: Compression with implicit neural representations. *arXiv preprint arXiv:2103.03123*, 2021. 2, 5

[10] Emilien Dupont, Hrushikesh Loya, Milad Alizadeh, Adam Goliński, Yee Whye Teh, and Arnaud Doucet. Coin++: Data agnostic neural compression. *arXiv preprint arXiv:2201.12904*, 2022. 2, 5

[11] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2846–2855, 2021. 2

[12] Ishit Mehta, Michaël Gharbi, Connelly Barnes, Eli Shechtman, Ravi Ramamoorthi, and Manmohan Chandraker. Modulated periodic activations for generalizable local functional representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14214–14223, 2021. 2

[13] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. *Advances in Neural Information Processing Systems*, 34:21557–21568, 2021. 2, 8

[14] Shishira R Maiya, Sharath Girish, Max Ehrlich, Hanyu Wang, Kwot Sin Lee, Patrick Poirson, Pengxiang Wu, Chen Wang, and Abhinav Shrivastava. Nirvana: Neural implicit representations of videos with adaptive networks and autoregressive patch-wise modeling. *arXiv preprint arXiv:2212.14593*, 2022. 2, 8

[15] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 2

[16] Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *Advances in Neural Information Processing Systems*, 33:10136–10147, 2020. 2

[17] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. 2

[18] Yannick Strümpler, Janis Postels, Ren Yang, Luc Van Gool, and Federico Tombari. Implicit neural representations for image compression. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*, pages 74–91. Springer, 2022. 2, 5, 6, 8, 9

[19] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11358–11367, 2021. 2, 3, 7

[20] Jonathan Tremblay, Moustafa Meshry, Alex Evans, Jan Kautz, Alexander Keller, Sameh Khamis, Charles Loop, Nathan Morrical, Koki Nagano, Towaki Takikawa, and Stan Birchfield. Rtmv: A ray-traced multi-view synthetic dataset for novel view synthesis. *IEEE/CVF European Conference on Computer Vision Workshop (Learn3DG ECCVW), 2022*, 2022. 2, 5

[21] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. *arXiv preprint arXiv:1611.01704*, 2016. 2

[22] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*, 2017. 2

[23] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018. 2, 3, 4, 6

[24] David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. *Advances in neural information processing systems*, 31, 2018. 2

[25] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7939–7948, 2020. 2

[26] Yibo Yang, Robert Bamler, and Stephan Mandt. Improving inference for neural image compression. *Advances in Neural*

Information Processing Systems, 33:573–584, 2020. 2, 4

[27] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. Dvc: An end-to-end deep video compression framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11006–11015, 2019. 2

[28] Amirhossein Habibian, Ties van Rozendaal, Jakub M Tomczak, and Taco S Cohen. Video compression with rate-distortion autoencoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7033–7042, 2019.

[29] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Balle, Sung Jin Hwang, and George Toderici. Scale-space flow for end-to-end optimized video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2020.

[30] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for video compression with hierarchical quality and recurrent enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6628–6637, 2020. 2

[31] Kenneth O Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8:131–162, 2007. 2

[32] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 2

[33] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 2

[34] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8628–8638, 2021. 2

[35] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021. 2

[36] Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, Joshua B Tenenbaum, Alberto Rodriguez, Pulkit Agrawal, and Vincent Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6394–6400. IEEE, 2022. 2

[37] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018. 3

[38] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*, 2019.

[39] Sharath Girish, Shishira R Maiya, Kamal Gupta, Hao Chen, Larry S Davis, and Abhinav Shrivastava. The lottery ticket hypothesis for object recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 762–771, 2021.

[40] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019. 3

[41] Xiaofan Lin, Cong Zhao, and Wei Pan. Towards accurate binary convolutional neural network. *Advances in neural information processing systems*, 30, 2017. 3

[42] Frederick Tung and Greg Mori. Clip-q: Deep network compression learning by in-parallel pruning-quantization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7873–7882, 2018.

[43] Angela Fan, Pierre Stock, Benjamin Graham, Edouard Grave, Rémi Gribonval, Herve Jegou, and Armand Joulin. Training with quantization noise for extreme model compression. *arXiv preprint arXiv:2004.07320*, 2020. 3

[44] D. Oktay et al. Scalable model compression by entropy penalized reparameterization. In *ICLR*, 2020. 3

[45] Sharath Girish, Kamal Gupta, Saurabh Singh, and Abhinav Shrivastava. Lilnetx: Lightweight networks with extreme model compression and structured sparsification. *ArXiv*, abs/2204.02965, 2022. 3

[46] 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

[47] Matthias Teschner, Bruno Heidelberger, Matthias Müller, Danat Pomerantes, and Markus H Gross. Optimized spatial hashing for collision detection of deformable objects. In *Vmv*, volume 3, pages 47–54, 2003. 4

[48] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 4

[49] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 4

[50] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948. 4

[51] Ian H Witten, Radford M Neal, and John G Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987. 5

[52] Kodak lossless true color image suite. `http://r0k.us/graphics/kodak/8`. 5

[53] Towaki Takikawa, Or Perel, Clement Fuji Tsang, Charles Loop, Joey Litalien, Jonathan Tremblay, Sanja Fidler, and Maria Shugrina. Kaolin wisp: A pytorch library and engine for neural fields research. `https://github.com/NVIDIAGameWorks/kaolin-wisp`, 2022. 5

[54] Alexandre Mercat, Marko Viitanen, and Jarno Vanne. Uvg dataset: 50/120fps 4k sequences for video codec analysis and development. In *Proceedings of the 11th ACM Multimedia Systems Conference*, pages 297–302, 2020. 5

[55] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[56] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan.

Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 7