Contents

LIFT: Latent Implicit Functions for Task- and Data-Agnostic Encoding

Supplementary Material

1. More of LIFT

1.1. Qualitative Comparison: LIFT vs. GASP

We generated 10 samples from GASP [11] using the pre-trained model available on their GitHub repository. Although GASP achieves a competitive FID score of 13.5, its adversarial approach often leads to noticeable artifacts in the generated samples, a common issue with adversarial methods. These artifacts suggest limitations in capturing fine-grained details and spatial coherence. In contrast, our model's samples demonstrate greater visual fidelity, with fewer artifacts and a smoother, more coherent structure across generated outputs as shown in Figure 2 in the paper.

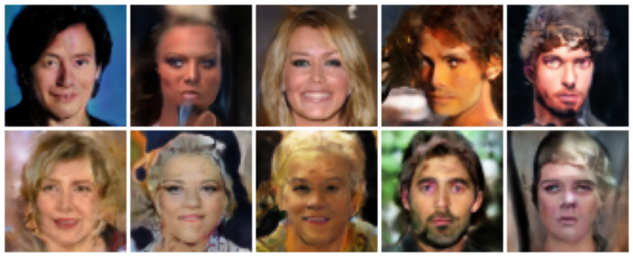


Figure 1. Generation result of GASP [11].

1.2. Ablation on Scale Levels of HLG

To assess the contribution of each scale in our HLG framework, we conduct an ablation study on the CelebA-HQ dataset by selectively removing individual scales and measuring the resulting performance. Specifically, we evaluate four configurations: (1) Global only, (2) Local only, (3) Global + Local, and (4) Global + Intermediate + Local (our full model). Table 1 summarizes the corresponding PSNR values on the test set. The Global-only configuration yields a PSNR of 24.82 dB, underscoring the importance of fine-scale information. Conversely, using the Local scale alone significantly improves performance to 39.28 dB, highlighting the role of detailed spatial features. Combining the Global and Local scales further raises the PSNR to 39.62 dB, indicating that multi-scale cues complement each other. Finally, incorporating the Intermediate scale and the Global and Local scales produces the highest PSNR of 40.91 dB, demonstrating that each scale contributes unique and essential information. These results confirm that a multi-scale approach is crucial for capturing both coarse global context and finer local details, thereby improving overall reconstruction quality.

Table 1. Comparison of test PSNR across different scales.

Scale	**Global**	**Local**	**Global-Local**	**Global-Intermediate-Local**
Test PSNR	24.82	39.28	39.62	40.91

1.3. Different Configurations of LIFT on the ShapeNet dataset

In Table 2, we evaluate the impact of channel sizes in the latent representations on reconstruction quality using PSNR for the ShapeNet 64^3 dataset. Our main configuration employs a global latent $Z^\dagger$ with 256 channels, an intermediate latent $Z^\star$ with 128 channels, and a local latent Z with 64 channels, achieving a PSNR of 35.15 dB with a compression ratio (CR) of 8× for Z^α. Halving the channel dimensions leads to a PSNR of 33.91 dB (CR of 16×). Further reduction to 64, 32, and 16 channels yields a PSNR of 32.21 dB (CR of 32×). Additional configurations with even smaller channel sizes—32, 16, 8 channels and 32, 16, 2 channels, result in PSNR values of 30.78 dB (CR of 64×) and 24.01 dB (CR of 256×), respectively. These experiments demonstrate the trade-off between compression and reconstruction fidelity that higher channel capacities in the latent spaces contribute to better reconstruction fidelity, while more compressed representations, despite reducing PSNR, can still maintain competitive performance relative to other methods.

Table 2. Impact of channel size in local, intermediate, and global latents on PSNR. CR denotes the compression ratio based on an input shape of 64^3. Higher CR values indicate greater compression.

Input shape			Test PSNR ↑	CR
$Z^\dagger$	$Z^\star$	Z		
$1 \times 1 \times 1 \times 256$	$4 \times 4 \times 4 \times 128$	$8 \times 8 \times 8 \times 64$	35.15	8×
$1 \times 1 \times 1 \times 128$	$4 \times 4 \times 4 \times 64$	$8 \times 8 \times 8 \times 32$	33.91	16×
$1 \times 1 \times 1 \times 64$	$4 \times 4 \times 4 \times 32$	$8 \times 8 \times 8 \times 16$	32.21	32×
$1 \times 1 \times 1 \times 32$	$4 \times 4 \times 4 \times 16$	$8 \times 8 \times 8 \times 8$	30.78	64×
$1 \times 1 \times 1 \times 32$	$4 \times 4 \times 4 \times 16$	$8 \times 8 \times 8 \times 2$	24.01	256×

1.4. Scaling LIFT to Higher Resolutions

To evaluate the scalability of LIFT, we apply it to high-resolution data using CelebA-HQ at 384×384 resolution. Each image is encoded into a three-level latent hierarchy consisting of 1×1×256, 8×8×128, and 16×16×64 tensors. For downstream tasks such as generative modeling, we retain only the local latent (16×16×64), resulting in a latent that is 27× smaller than the original input in terms of total dimensionality. We train LIFT for 500 k iterations with

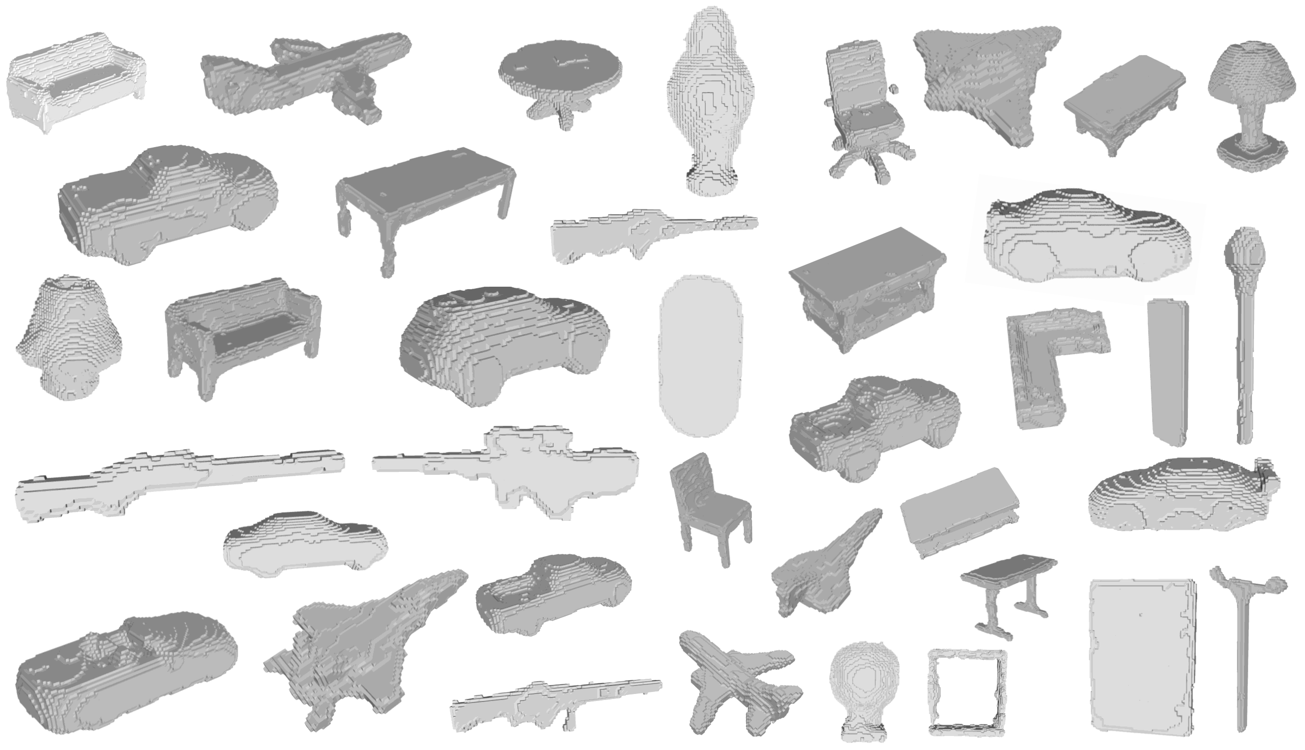


Figure 2. LIFT Shapenet voxel generation results.

a per-device batch size of 15 on four A100-80G GPUs, and subsequently train an ablated diffusion model (ADM) [7] on the resulting latent codes. As shown in Table 3, the ADM model achieves an FID of 9.71, improving over the DDIM baseline 10.44, while the reconstructor obtains 31.21 and 30.56 dB PSNR on the training and test sets, respectively. Qualitative reconstruction results on the test set are presented in Figure 3. These results demonstrate that LIFT maintains strong reconstruction fidelity even at significantly increased input resolutions.

Table 3. Reconstruction (PSNR) and generation (FID) performance on CelebA-HQ 384^2.

Method	Training PSNR ↑	Test PSNR ↑	FID (Generation) ↓
DDMI [24]	–	–	10.44
LIFT	31.21	30.56	**9.71**

1.5. Inference Time Efficiency and Benchmarks

We benchmark the inference-time performance of LIFT on both 2D (CelebA-HQ 64^2) and 3D (ShapeNet 64^3) datasets, reporting frames per second (FPS) and peak memory usage during reconstruction. All models are evaluated on an NVIDIA Quadro RTX 8000 GPU, with results summarized in Table 4. Since all methods are evaluated on the same device, we report baseline performance as provided in the mNIF [36] paper. On CelebA-HQ, LIFT achieves an inference speed of 5507.9 FPS, more than 2.14× faster than the next best method (mNIF-S at 2985.6 FPS), while also consuming less memory (8.2 MB vs. 10.2 MB). On ShapeNet, LIFT maintains its advantage, reaching **410.4 FPS** with a memory footprint of only 406 MB, representing a 1.86× speedup over mNIF-S (191.5 FPS), and a substantial reduction in memory usage compared to GEM (4010 MB) and GASP (763 MB). These results demonstrate that LIFT is highly efficient, offering faster inference and lower memory consumption across diverse input modalities. Its compact latent representation and localized decoding enable scalable performance, making it well-suited for deployment in real-time and resource-constrained applications.

Table 4. Inference speed (FPS) and memory usage for different methods evaluated on CelebA-HQ 64^2 and ShapeNet 64^3, executed on an NVIDIA Quadro RTX 8000.

Method	CelebA-HQ 64^2		ShapeNet 64^3	
	FPS ↑	Mem (MB) ↓	FPS ↑	Mem (MB) ↓
Functa [10]	332.9	144.1	–	–
GEM [9]	559.6	70.3	16.7	4010.0
GASP [11]	1949.3	16.4	180.9	763.1
DPF [39]	–	–	–	–
mNIF-S [36]	2985.6	10.2	191.5	642.1
mNIF-L [36]	891.3	24.4	69.6	1513.3
LIFT	5507.9	8.2	410.4	406.0

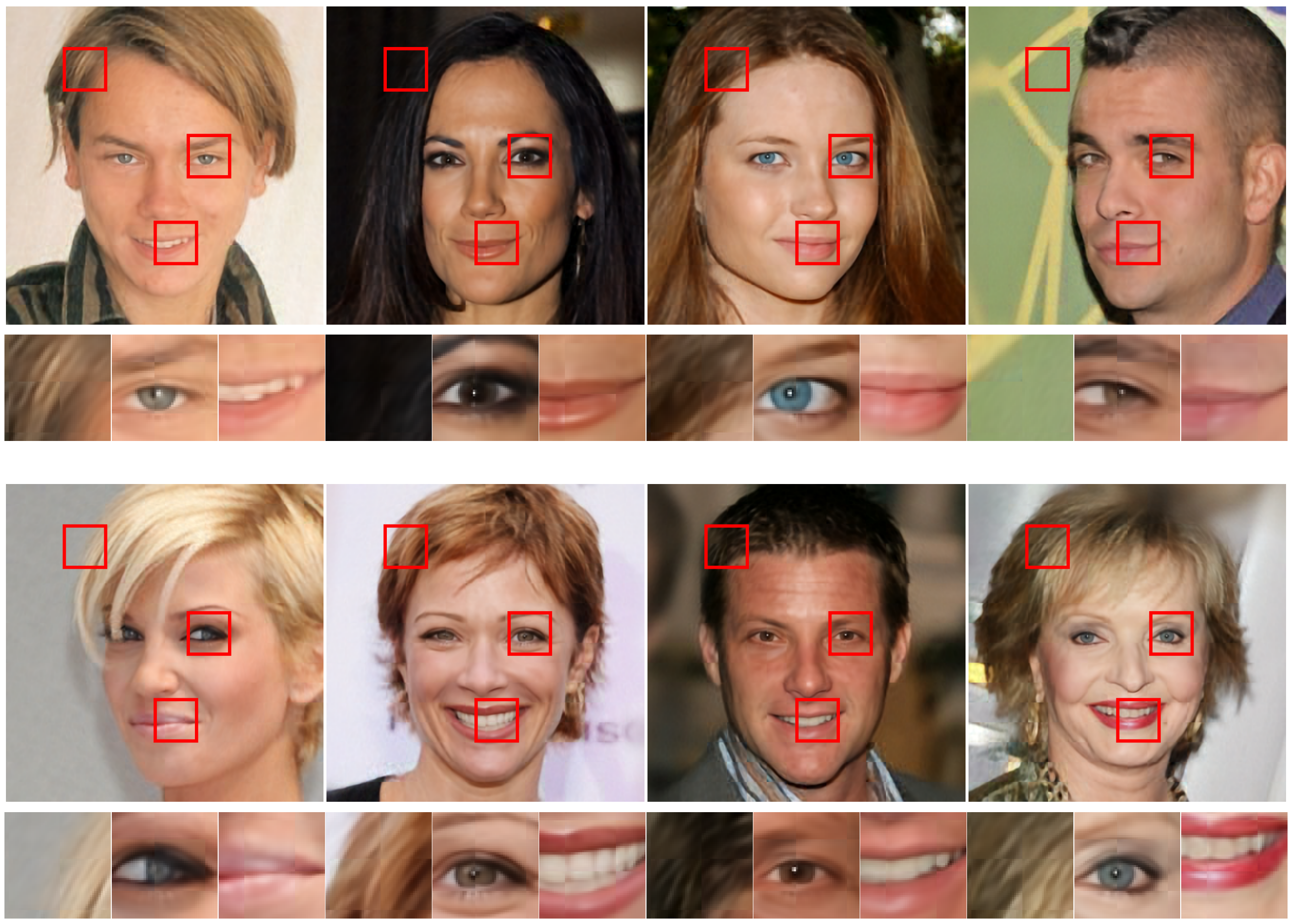


Figure 3. LIFT CelebA-HQ 384^2 reconstruction results.

2. Implementation Details

2.1. Datasets

2.1.1 Images

We use the CelebA-HQ 64 × 64 dataset provided by Functa [10], as well as the CelebA-HQ 384 × 384 dataset [16], each partitioned into 27K training and 3K testing images. For reconstruction tasks, performance is evaluated using the Peak Signal-to-Noise Ratio (PSNR) and reconstruction Fréchet Inception Distance (rFID). Additionally, the FID metric [14] is computed alongside precision, recall, and F1 score metrics to assess image generation quality [23, 28]. To further demonstrate scalability, we use the ImageNet-100 256 × 256 dataset [27] for the reconstruction task. ImageNet-100 [2, 27], a subset of the ImageNet-1k dataset, consists of 100 classes with 130K training and 5K test samples. For the classification task, we utilize the CIFAR-10 dataset [18], which contains 60K 32×32 color images distributed across 10 classes, with 6K images per class. The dataset is divided into 50K training images and 10K test images.

2.1.2 Voxels

We employ the ShapeNet dataset [5], which includes 35,019 training and 8,762 testing samples for 3D voxel data. Each voxel is represented at a 64^3 resolution with 16,384 surface points. We evaluate reconstruction tasks using Mean Squared Error (MSE) and PSNR metrics. For generation tasks, following IM-Net [6], we generate 8,762 shapes and extract 2,048-dimensional mesh features. Performance is measured using coverage, maximum mean discrepancy (MMD) [1], and Chamfer distance, adhering to the GEM [9] and mNIF [36] protocols.

2.2. Stage 1: Context Adaptation

For stage 1, we implement all models in JAX [4] using Haiku [13] and JAXline for training, with Functa [10] as the baseline framework for our coding setup. This stage involves adapting models for image and voxel data through a meta-learning framework, as detailed in the subsections below.

Table 5. Training configurations and model specifications for Stage 1 across CelebA-HQ, ImageNet, and ShapeNet datasets, detailing hyperparameters optimized for each dataset.

Input shape			P-MLP Configs			Meta-learning Configs					
					CelebA-HQ 64^2						
$Z^\dagger$	$Z^\star$	Z	# Hidden Layers	Width	W_0	Inner lr	Inner Opt	Outer lr	Outer Opt	T_{inner}	Meta-SGD
$1\times1\times64$	$4\times4\times32$	$8\times8\times16$	1	256	15	1.0	SGD	1.5e-5	Adam	3	✓
$1\times1\times128$	$4\times4\times64$	$8\times8\times32$	1	256	20	1.0	SGD	5e-5	Adam	3	✓
$1\times1\times256$	$4\times4\times128$	$8\times8\times64$	1	256	20	1.0	SGD	5e-5	Adam	3	✓
$1\times1\times512$	$4\times4\times256$	$8\times8\times128$	1	256	20	1.0	SGD	3e-5	Adam	3	✓
$1\times1\times256$	$4\times4\times128$	$8\times8\times64$	1	256	20	1.0	SGD	5e-5	Adam	3	✓
$1\times1\times256$	$4\times4\times128$	$8\times8\times64$	3	64	20	1.0	SGD	5e-5	Adam	3	✓
$1\times1\times256$	$4\times4\times128$	$8\times8\times64$	4	64	20	1.0	SGD	5e-5	Adam	3	✓
$1\times1\times256$	$4\times4\times128$	$8\times8\times64$	8	32	20	1.0	SGD	3e-5	Adam	3	✓
$1\times1\times256$	$4\times4\times128$	$8\times8\times64$	16	16	20	1.0	SGD	2e-5	Adam	3	✓
$1\times1\times256$	$2\times2\times128$	$8\times8\times64$	1	256	20	1.0	SGD	5e-5	Adam	3	✓
$1\times1\times256$	$2\times2\times128$	$4\times4\times64$	1	256	20	1.0	SGD	5e-5	Adam	3	✓
$1\times1\times256$	$4\times4\times128$	$8\times8\times128$	1	256	20	1.0	SGD	4e-5	Adam	3	✓
$1\times1\times64$	$4\times4\times64$	$8\times8\times64$	1	256	20	1.0	SGD	5e-5	Adam	3	✓
$1\times1\times128$	$4\times4\times128$	$8\times8\times128$	1	256	20	1.0	SGD	4e-5	Adam	3	✓
					CelebA-HQ 384^2						
$1\times1\times256$	$8\times8\times128$	$16\times16\times64$	2	256	15	1.0	SGD	1.5e-5	Adam	3	✓
					ImageNet-100 256^2						
$1\times1\times256$	$8\times8\times256$	$16\times16\times256$	2	256	20	1.0	SGD	2e-5	Adam	3	✓
$1\times1\times512$	$8\times8\times256$	$16\times16\times128$	2	256	20	1.0	SGD	2e-5	Adam	3	✓
					CIFAR-10 32^2						
$1\times1\times64$	$4\times4\times32$	$8\times8\times16$	2	256	10	1.0	SGD	3e-5	Adam	3	✓
					ShapeNet 64^3						
$1\times1\times1\times256$	$4\times4\times4\times128$	$8\times8\times8\times64$	4	64	20	1.0	SGD	4e-5	Adam	3	✓
$1\times1\times1\times256$	$4\times4\times4\times128$	$8\times8\times8\times64$	8	32	20	1.0	SGD	4e-5	Adam	3	✓
$1\times1\times1\times128$	$4\times4\times4\times64$	$8\times8\times8\times32$	4	64	20	1.0	SGD	4e-5	Adam	3	✓
$1\times1\times1\times64$	$4\times4\times4\times32$	$8\times8\times8\times16$	4	64	20	1.0	SGD	4e-5	Adam	3	✓
$1\times1\times1\times32$	$4\times4\times4\times16$	$8\times8\times8\times8$	4	64	20	1.0	SGD	4e-5	Adam	3	✓
$1\times1\times1\times32$	$4\times4\times4\times16$	$8\times8\times8\times2$	4	64	20	1.0	SGD	2e-5	Adam	3	✓

2.2.1 Images

We train on images from the train split of the CIFAR-10 32^2, CelebA-HQ 64^2, and ImageNet-100 256^2 datasets. The model takes local 2D input coordinates $(x_{\text{local}}, y_{\text{local}})$, with each P-MLP processing its corresponding local coordinate and returning 3D RGB values (y_R, y_G, y_B) representing the local patch. After processing, we merge these patches to reconstruct the full image. We use local dense sampling, querying all local input coordinates in parallel for all P-MLPs. For instance, with a grid size of 8, we generate 64 query coordinates $(64/8 \times 64/8)$ for 8×8 P-MLPs to input into the network for image benchmarking. In addition, the model is trained using a meta-learning approach for 200K iterations with a per-device batch size of 128 on four A40 GPUs (each with 48 GB of memory) using the CelebA-HQ dataset. We used a per-device batch size of 512 for CIFAR-10 and 16 for ImageNet-100, trained on 4 A40 GPUs. All details regarding different model configurations and meta-learning setups are provided in Table 5. To create the modulation datasets during inference, we freeze the network weights and optimize the zero-initialized latents in 3 steps using SGD. We then save the resulting Z^α latent vectors for both training and test data. Additionally, we set $K = 8$ in our $\mathcal{L}_{\text{smoothness}}$ loss function with the $\lambda = (1/100)^2$. For the ablation studies, we use the highlighted pink in Table 5. Our results in the main Table 1 use the highlighted orange.

2.2.2 Voxels

We train on the ShapeNet 64^3 dataset, using 35,019 training shapes at $64\times64\times64$ resolution and testing on 8,762 shapes at the same resolution. The model takes local 3D input coordinates $(x_{\text{local}}, y_{\text{local}}, z_{\text{local}})$, with each P-MLP processing its corresponding local coordinate, and returns a scalar output y_o, indicating whether the queried coordinate is inside or outside of the object. After processing, we merge these local outputs to reconstruct the entire voxel grid representation of the shape. The model is trained using a meta-learning approach, configured for 120K iterations with a per-device batch size of 8, on four A40 GPUs. Further details on the configurations and meta-learning setups are provided in Table 5. We use the approach described for **Images** (subsubsection 2.2.1) to create the modulation datasets during inference. We set $K = 16$ in our $\mathcal{L}_{\text{smoothness}}$ loss function with the $\lambda = (1/100)^2$.

2.3. Stage 2: Task-Driven Generalization

2.3.1 Images

Generation: We base our image generation implementation on the ablated diffusion model (ADM) [7]. The training pipeline begins with the creation of our modulation dataset using the LIFT framework. In this setup, the latent representations of Z^α are derived using the configurations highlighted in orange in Table 5. For the diffusion model, we configure an image size of 8 and set the number of channels to 320, employing channel multipliers of 1, 2, and 4. Each layer consists of two residual blocks with a dropout rate of 0.1. Attention mechanisms are applied at a resolution of 4, and both `resblock_updown` and `use_scale_shift_norm` are enabled to enhance feature normalization and spatial resolution handling. The diffusion process is conducted over 2000 steps for 500K iterations using a cosine noise schedule. Training is performed with a learning rate of 1×10^{-4} and a batch size of 256. We utilize a loss-second-moment schedule sampler to optimize the training dynamics. To mitigate the low variance observed in the learned modulations, we standardize the latent representations across spatial dimensions ($\frac{Z^\alpha - \mu}{\tau.\sigma}$), ensuring that each feature dimension is centered and scaled independently. This normalization simplifies the generative process. Additionally, we apply a scaling factor (τ) of 2.5 to our standardization to enhance training stability. During the sampling phase, we employ the DDIM sampler [32] with a timestep respacing of 200.
Classification: In our implementation, the VMamba [38] model is configured with a batch size of 512 and an embedding dimension of 128. Label smoothing with a factor of 0.1 is applied to enhance generalization, and a drop path rate of 0.1 is employed for regularization. The model architecture comprises two stages with depths of 9 and 2 layers, respectively. Latents are normalized across examples using a scaling normalization factor of 2.0. The learning rate schedule includes a base learning rate of 5e-3, a minimum learning rate of 5e-6, and a warmup phase of 10 epochs starting at 5e-5.

2.3.2 Voxels

Generation: For our base diffusion model architecture, we use ADM [7], which features a U-Net architecture initially created for 2D image synthesis. Since the original design was intended for 2D, we adapted all operations to function in 3D. We trained our diffusion model for 500K iterations with a learning rate of 1×10^{-4} and a batch size of 64. The training process employs the "loss-second-moment" schedule sampler, and we standardize the latents like images by scaling them with a factor $\tau = 2.5$. The model operates on $8 \times 8 \times 8$ image inputs, with an initial channel size of 128 and channel multipliers set to $\{1, 2, 4\}$. Each resolution includes two residual blocks. A dropout rate of 0.1 is applied. In addition, `resblock_updown` and `use_scale_shift_norm` are enbaled. The diffusion process comprises 1000 steps and follows a cosine noise schedule for smooth and stable noise transitions. For sampling, DDIM is used with 200 timesteps, achieving a balance between computational efficiency and sample quality.

3. ReLIFT vs. SIREN

3.1. SIREN Pipeline

Given a set of signals $\{(\mathbf{r}_i, \mathbf{y}_i)\}_{i=1}^N$, where $\mathbf{r}_i \in \mathbb{R}^d$ represents spatial coordinates and $\mathbf{y}_i \in \mathbb{R}^m$ denotes the corresponding attributes, SIREN aims to approximate a continuous function $f(\mathbf{r}; \theta)$ that maps coordinates to their associated values with high fidelity. This function is parameterized as a neural network, where each layer l operates according to the following equations:

$$\begin{aligned}
\mathbf{z}^{(0)} &= \sin\left(\omega_0(W^{(0)}\mathbf{r} + \mathbf{b}^{(0)})\right) \\
\mathbf{z}^{(l)} &= \sin\left(\omega_0(W^{(l)}\mathbf{z}^{(l-1)} + \mathbf{b}^{(l)})\right), \quad l = 1, \ldots, L-2, \\
f(\mathbf{r}; \theta) &= W^{(L)}\mathbf{z}^{(L-1)} + \mathbf{b}^{(L)},
\end{aligned} \tag{1}$$

where $\mathbf{z}^{(l)}$ is the output of the l-th layer, $\theta = \{W^{(l)}, \mathbf{b}^{(l)} \mid l = 1, \ldots, L\}$ denotes the learnable parameters, L is the total number of layers, and ω_0 is a fixed hyperparameter that scales the input to the sinusoidal activation, controlling the frequency response of each layer. Also, they showed that rescaling initialization by ω_0 adjusts SIRENs' spectral bias, with higher ω_0 favoring higher frequencies.

3.2. Revisiting Input Layer Transformation

As highlighted in [37], the initial layer of SIREN serves as a frequency encoding mechanism. Specifically, the output of the first layer can be expressed as:

$$\mathbf{z}^{(0)} = \sin\left(\Omega \mathbf{r}\right), \tag{2}$$

where $\Omega = \omega_0 W^{(0)} \in \mathbb{R}^T$. Considering a three-layer SIREN, the network can be defined as:

$$f(\mathbf{r}; \theta) = \mathbf{w}^{(2)\top} \sin\left(\mathbf{W}^{(1)} \sin\left(\Omega \mathbf{r}\right)\right), \tag{3}$$

with $\mathbf{W}^{(1)} \in \mathbb{R}^{F \times T}$ and $\mathbf{w}^{(2)} \in \mathbb{R}^F$. The input to each neuron in the second layer is a linear combination of sinusoids at frequencies determined by Ω. The output of a neuron $z_m^{(1)}$ in the second layer can be written as:

$$z_m^{(1)} = \sin\left(\mathbf{W}_{m:}^{(1)} \sin(\Omega \mathbf{r})\right) = \sin\left(\sum_{t=0}^{T-1} W_{m,t}^{(1)} \sin\left(\boldsymbol{\omega}_t^\top \mathbf{r}\right)\right), \tag{4}$$

where $\boldsymbol{\omega}_t^\top$ denotes the t-th row of Ω. To analyze the frequency components without considering the bias term, we expand the sine of a sum using Bessel function identities. Using the property:

$$\sin\left(\sum_k \phi_k\right) = \operatorname{Im}\left\{\exp\left(j \sum_k \phi_k\right)\right\}, \tag{5}$$

and the exponential identity:

$$\exp\left(j \sum_k \phi_k\right) = \prod_k \exp(j\phi_k), \tag{6}$$

we can write:

$$\exp\left(j \sum_{t=0}^{T-1} W_{m,t}^{(1)} \sin\left(\boldsymbol{\omega}_t^\top \mathbf{r}\right)\right) = \prod_{t=0}^{T-1} \exp\left(j W_{m,t}^{(1)} \sin\left(\boldsymbol{\omega}_t^\top \mathbf{r}\right)\right). \tag{7}$$

Using the expansion of the complex exponential of a sine function:

$$\exp(j\beta \sin(\theta)) = \sum_{n=-\infty}^{\infty} J_n(\beta) e^{jn\theta}, \tag{8}$$

where $J_n(\beta)$ is the Bessel function of the first kind of order n, we have:

$$\exp\left(j W_{m,t}^{(1)} \sin\left(\boldsymbol{\omega}_t^\top \mathbf{r}\right)\right) = \sum_{s_t=-\infty}^{\infty} J_{s_t}\left(W_{m,t}^{(1)}\right) e^{j s_t \boldsymbol{\omega}_t^\top \mathbf{r}}. \tag{9}$$

Therefore, the product over t becomes:

$$\exp\left(j \sum_{t=0}^{T-1} W_{m,t}^{(1)} \sin\left(\boldsymbol{\omega}_t^\top \mathbf{r}\right)\right) = \prod_{t=0}^{T-1} \sum_{s_t=-\infty}^{\infty} J_{s_t}\left(W_{m,t}^{(1)}\right) e^{j s_t \boldsymbol{\omega}_t^\top \mathbf{r}} = \sum_{\mathbf{s}\in\mathbb{Z}^T} \left(\prod_{t=0}^{T-1} J_{s_t}\left(W_{m,t}^{(1)}\right) e^{j s_t \boldsymbol{\omega}_t^\top \mathbf{r}}\right), \tag{10}$$

where $\mathbf{s} = (s_0, s_1, \ldots, s_{T-1})$. Since $\sin\left(\sum_t \phi_t\right) = \operatorname{Im}\left\{\exp\left(j\sum_t \phi_t\right)\right\}$, we have:

$$\begin{aligned} z_m^{(1)} &= \operatorname{Im}\left\{\exp\left(j \sum_{t=0}^{T-1} W_{m,t}^{(1)} \sin\left(\boldsymbol{\omega}_t^\top \mathbf{r}\right)\right)\right\} \\ &= \operatorname{Im}\left\{\sum_{\mathbf{s}\in\mathbb{Z}^T} \left(\prod_{t=0}^{T-1} J_{s_t}\left(W_{m,t}^{(1)}\right) e^{j s_t \boldsymbol{\omega}_t^\top \mathbf{r}}\right)\right\} \\ &= \sum_{\mathbf{s}\in\mathbb{Z}^T} \left(\prod_{t=0}^{T-1} J_{s_t}\left(W_{m,t}^{(1)}\right)\right) \sin\left(\sum_{t=0}^{T-1} s_t \boldsymbol{\omega}_t^\top \mathbf{r}\right). \end{aligned} \tag{11}$$

Thus, the output of the network can be expressed as:

$$\begin{aligned} f(\mathbf{r};\theta) = \sum_{m=0}^{F-1} w_m^{(2)} z_m^{(1)} = \sum_{m=0}^{F-1} w_m^{(2)} \\ \sum_{\mathbf{s}\in\mathbb{Z}^T} \left(\prod_{t=0}^{T-1} J_{s_t}\left(W_{m,t}^{(1)}\right)\right) \sin\left(\sum_{t=0}^{T-1} s_t \boldsymbol{\omega}_t^\top \mathbf{r}\right). \end{aligned} \tag{12}$$

This expression indicates that the network output is a sum of sinusoidal functions at frequencies $\sum_{t=0}^{T-1} s_t \boldsymbol{\omega}_t$, where $s_t \in \mathbb{Z}$. This implies that by scaling $\sum_{t=0}^{T-1} s_t \boldsymbol{\omega}_t$, we can increase the network's capacity to learn higher-frequency components. In addition, the coefficients of these sinusoids are determined by the products of Bessel functions $J_{s_t}\left(W_{m,t}^{(1)}\right)$ and the weights $w_m^{(2)}$. Due to the properties of Bessel functions, which generally decrease in magnitude with increasing order $|s_t|$ when the argument $W_{m,t}^{(1)}$ is small, higher-order harmonics (those with larger $|s_t|$) tend to have smaller coefficients. This results in an implicit bias towards lower-frequency components, concentrating most of the output signal's energy around the fundamental frequencies $\boldsymbol{\omega}_t$. However, increasing the scale of inner layer coefficients, like $W^{(1)}$, boosts higher-order harmonics, allowing the network to capture a wider frequency range. Therefore, we begin by examining how scaling the input layer frequency affects the network's capacity. Next, we introduce a residual connection term, which intuitively directs the network's focus towards adjusting $W^{(1)}$ to amplify higher-order harmonics while preserving the fundamental components.

3.3. Effect of Input Scaling on Frequency Representation

To investigate the effect of scaling the input frequencies, we introduce a scaling factor $\gamma > 1$ such that:

$$\mathbf{z}^{(0)} = \sin(\gamma \Omega \mathbf{r}). \tag{13}$$

Applying the same analysis, the output of the second layer becomes:

$$
\begin{aligned}
z_m^{(1)} &= \operatorname{Im}\left\{\exp\left(j\sum_{t=0}^{T-1} W_{m,t}^{(1)} \sin\left(\gamma\boldsymbol{\omega}_t^\top \mathbf{r}\right)\right)\right\} \\
&= \operatorname{Im}\left\{\prod_{t=0}^{T-1} \exp\left(jW_{m,t}^{(1)} \sin\left(\gamma\boldsymbol{\omega}_t^\top \mathbf{r}\right)\right)\right\} \\
&= \operatorname{Im}\left\{\prod_{t=0}^{T-1} \left(\sum_{s_t=-\infty}^{\infty} J_{s_t}\left(W_{m,t}^{(1)}\right) e^{js_t\gamma\boldsymbol{\omega}_t^\top \mathbf{r}}\right)\right\} \\
&= \sum_{\mathbf{s}\in\mathbb{Z}^T} \left(\prod_{t=0}^{T-1} J_{s_t}\left(W_{m,t}^{(1)}\right)\right) \sin\left(\gamma\sum_{t=0}^{T-1} s_t\boldsymbol{\omega}_t^\top \mathbf{r}\right).
\end{aligned}
\tag{14}
$$

By introducing the scaling factor γ, we effectively scale the frequencies of the sinusoidal components in the output by γ. When γ increases, the frequencies $\gamma\sum_{t=0}^{T-1} s_t\boldsymbol{\omega}_t$ increase proportionally, extending the network's capacity to represent higher-frequency components. This adjustment enhances the network's ability to represent high-frequency information by broadening the range of frequencies it can model.

3.4. ReLIFT and Residual Connections

SIRENs have shown impressive abilities in representing complex signals using sinusoidal activations [31]. However, they face a capacity-convergence gap when modeling high-frequency components. Although scaling the first layer ω_0 can improve the network's capacity for higher frequencies, it may not fully bridge this gap (see Figure 4). To overcome these limitations, we propose adding residual connections to our architecture, aiming to enhance both expressiveness and convergence.

In our proposed architecture, ReLIFT, we integrate residual connections into SIREN, where each layer adds its input to its output. This approach facilitates gradient flow, preserves low-frequency representations, and enables better modeling of higher frequencies. The network layers are defined as follows:

$$
\begin{aligned}
&\mathbf{z}^{(0)} = \sin\left(\gamma\omega_0(W^{(0)}\mathbf{r} + \mathbf{b}^{(0)})\right) \\
&\mathbf{z}^{(l)} = \sin\left(\omega_0(W^{(l)}\mathbf{z}^{(l-1)} + \mathbf{b}^{(l)})\right) + \mathbf{z}^{(l-1)},\ l = 1, \dots, L-2, \\
&f(\mathbf{r};\theta) = W^{(L)}\mathbf{z}^{(L-1)} + \mathbf{b}^{(L)}.
\end{aligned}
\tag{15}
$$

Let's consider the first layer outputs:

$$
\mathbf{z}^{(0)} = \sin\left(\gamma\Omega\mathbf{r}\right). \tag{16}
$$

The second layer with a residual connection is given by:

$$
\mathbf{z}^{(1)} = \sin\left(\mathbf{W}^{(1)} \sin\left(\gamma\Omega\mathbf{r}\right)\right) + \mathbf{z}^{(0)}. \tag{17}
$$

Considering the frequency component analysis using Bessel function identities (see subsection 3.2), the output of the second layer becomes:

$$
\begin{aligned}
z_m^{(1)} = &\sum_{\mathbf{s}\in\mathbb{Z}^T} \left(\prod_{t=0}^{T-1} J_{s_t}\left(W_{m,t}^{(1)}\right)\right) \sin\left(\gamma\sum_{t=0}^{T-1} s_t\boldsymbol{\omega}_t^\top \mathbf{r}\right) \\
&+ \sin(\gamma\omega_m^T r).
\end{aligned}
\tag{18}
$$

Intuitively, the residual connection in our network plays a crucial role in ensuring that lower-order harmonics (fundamental frequencies) are robustly represented. Although the Bessel expansion contributes to higher-order harmonics, the residual connection allows the network to avoid depending solely on the Bessel functions to capture lower frequencies. By preserving fundamental frequencies through the residual connection, the network can adjust the weights $W^{(1)}$ to enhance higher-order harmonics without risking the attenuation of fundamental components. Furthermore, residual connections provide shortcut paths for gradients, which significantly improves gradient flow during backpropagation [12], mitigating issues related to vanishing or exploding gradients.

We conducted experiments comparing the standard SIREN model and our ReLIFT model, including ablations, on a single-image representation task with similar model configurations (a 5-layer MLP with a width of 256). As illustrated in Figure 4, ReLIFT achieved faster convergence and higher PSNR, demonstrating its effectiveness in reducing the gap in convergence and capacity. Additionally, we extend our method to various tasks, which are discussed in the following sections. We also present the activation statistics for ReLIFT and SIREN in Figure 5. As shown, the maximum frequency increases with additional layers, allowing our model to capture a broader range of frequencies effectively.

4. ReLIFT in the LIFT Framework

We integrate ReLIFT into our LIFT framework, modifying the modulation approach of SIREN-based activations (as detailed in Section 3.2). Specifically, we introduce a residual connection to the output of the activation function and apply a scaling factor γ to the input layer frequency ω_0. To evaluate the impact of this approach, we perform a reconstruction task on the CelebA-HQ 64^2 dataset. Both the baseline and ReLIFT networks are trained using the configurations specified in Table 5, highlighted in pink. We also set $\gamma = 2$, and trained both models for 120K iterations.

Our experiments reveal a marked improvement in convergence rate and reconstruction quality during the early stages of training with the ReLIFT model compared to the baseline LIFT configuration. Notably, at just 10K iterations, ReLIFT achieves a PSNR that is 2.3 points higher than LIFT, and by 20K iterations, ReLIFT reaches a PSNR of 38.21, outper-

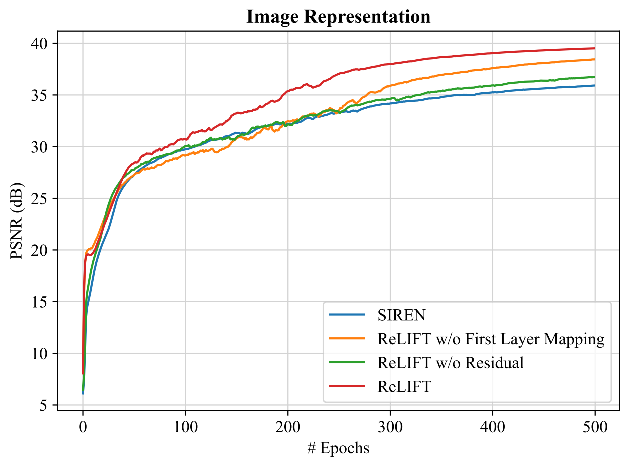


Figure 4. Comparison of convergence rates between the standard SIREN and ReLIFT on an image representation task. ReLIFT exhibits faster convergence and higher capacity. γ is set to 2.

forming all baselines, which typically requires 200K iterations to approach comparable results. Regarding computational efficiency, ReLIFT reaches 10K iterations in a wall time of 68 minutes and 20K in approximately 134 minutes, with a per-device batch size of 128 on 4 A40 GPUs. This efficiency demonstrates that, within only 134 minutes, ReLIFT can achieve high-fidelity reconstructions that surpass all current SOTA methods, making it a compelling model for rapid, high-quality image reconstruction.

5. Single Data Task Analysis

To evaluate the effectiveness of our straightforward approach, ReLIFT, we extend our experiments to single data tasks, including **signal representation** and **inverse problems**. Our results demonstrate that ReLIFT can significantly reduce the convergence-capacity gap and achieve SOTA performance across all tasks, without introducing additional learnable parameters or requiring new activation functions. These findings highlight ReLIFT's ability to surpass existing SOTA methods with a simple yet powerful adjustment, making it highly applicable across a range of tasks.

We evaluate `ReLIFT` against six SOTA INRs: ReLU with positional encoding (**ReLU+P.E**) [33], Fourier feature embedding (**FFN**) [33], **SIREN** [31], Gaussian-based activation functions (**Gauss**) [26], wavelet activation functions (**WIRE**) [29], **FINER** [21]. To ensure a fair comparison, each INR is configured with the same network architecture, consisting of 3 hidden layers with 256 neurons per layer, and is trained using the Adam optimizer [17] and an L2 loss function between the network output and the ground truth. Other hyperparameters follow the specifications provided in the authors' open-source code and WIRE [29]. Experiments run for 500 epochs, except for audio (1000 epochs) and occupancy (200 epochs). We also set $\gamma = 2$ for ReLIFT.

5.1. Signal Representations

5.1.1 Image

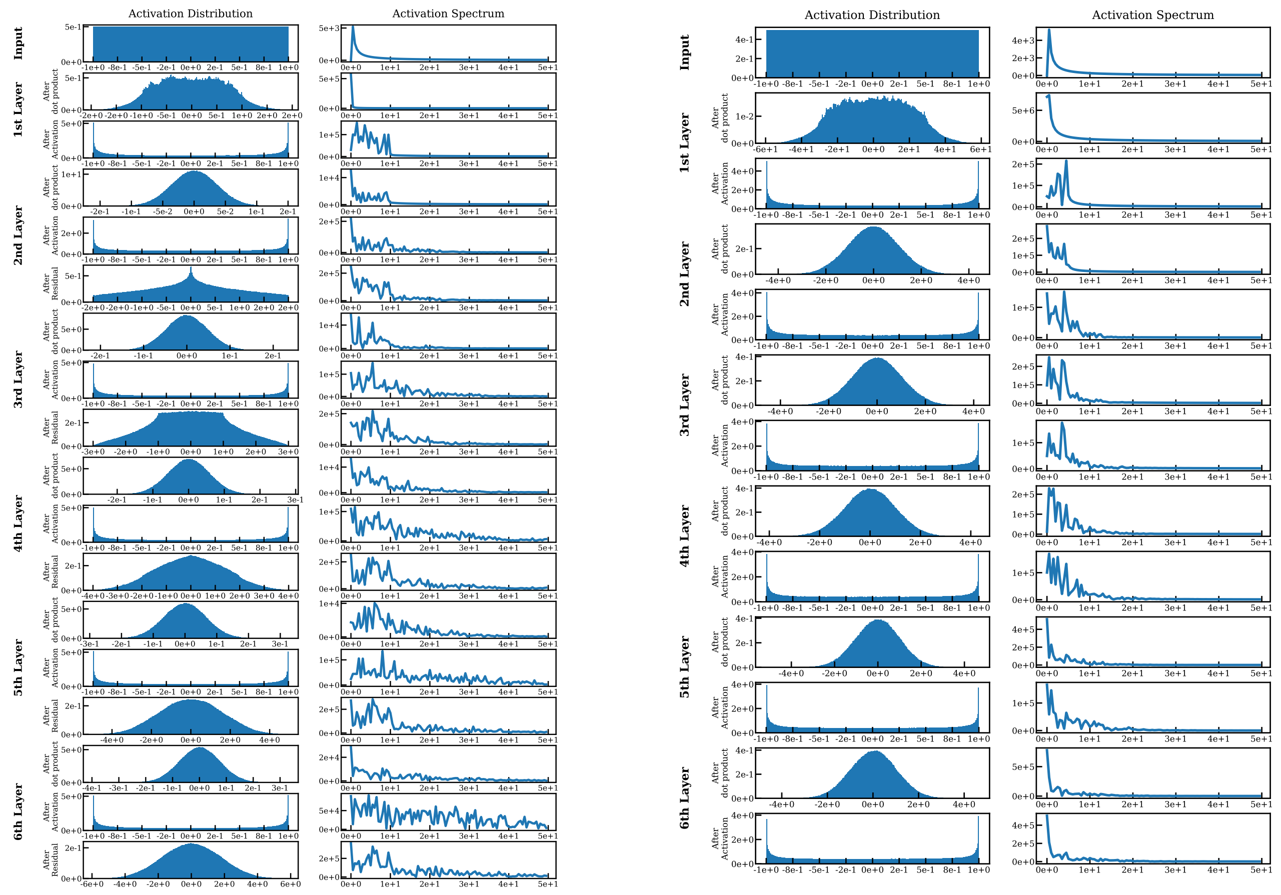
Data. For the image representation task, we use the DIV2K dataset [34], with images downscaled to 1/4 of their original size. In Figure 7, the first image (octopus) is resized from $1404 \times 2040 \times 3$ to $351 \times 510 \times 3$. The second image is trained at a resolution of $411 \times 510 \times 3$, and the third at $435 \times 510 \times 3$.

Analysis. Given a 2D point (x, y), the INR learns a mapping function $f : \mathbb{R}^2 \to \mathbb{R}^3$ that outputs the RGB values. The results in Figure 7 highlight that ReLIFT consistently surpasses other INR methods in PSNR across various images, showcasing its superior reconstruction capability. For the first image (octopus), ReLIFT delivers sharper reconstructions, particularly in the highlighted area. In contrast, competing methods such as ReLU+P.E., FFN, and WIRE produce noticeably blurrier outputs, and Gauss introduces color artifacts. ReLIFT improves the PSNR by 2.59 and 2.64 over SIREN and FINER, the second and third-best methods, respectively. Similarly, for the second image (tiger), ReLIFT achieves a PSNR increase of 3.61 and 4.41 over SIREN and FINER, demonstrating the method's reliability across different images. In the final, higher-resolution image, ReLIFT achieves a PSNR of 40.11, surpassing its closest competitor by a margin of 4.12, further reinforcing its capability for high-fidelity reconstructions.

5.1.2 Occupancy Volume

Data. We evaluate our approach using 4 shapes from a public dataset [19, 22]. For each shape, we create occupancy volumes by point sampling on a $512 \times 512 \times 512$ grid, assigning a value of 1 to voxels within the shape and 0 to those outside.

Analysis. Given a 3D point (x, y, z), the INR learns a mapping function $f : \mathbb{R}^3 \to \mathbb{R}$ that outputs the signed distance field (SDF) values. The quantitative comparisons in Table 6 demonstrate the effectiveness of our approach, ReLIFT, in representing SDF across various shapes. In terms of Intersection over Union (IOU), ReLIFT consistently outperforms other methods on all tested shapes: Armadillo, Dragon, Lucy, and Thai Statue. On average, ReLIFT achieves the highest IOU (0.9963), surpassing the second-best method, FINER, with an average of 0.9944. The qualitative comparison is shown in Figure 8 for the Thai statue. ReLIFT achieves the best overall performance, capturing both high-frequency details and smooth transitions, closely aligning with the ground truth. FINER retains broader structural features but sacrifices finer details, leading to coarser outputs. SIREN performs well in smooth



(a) ReLIFT

(b) SIREN

Figure 5. Activation statistics comparison between ReLIFT and SIREN.

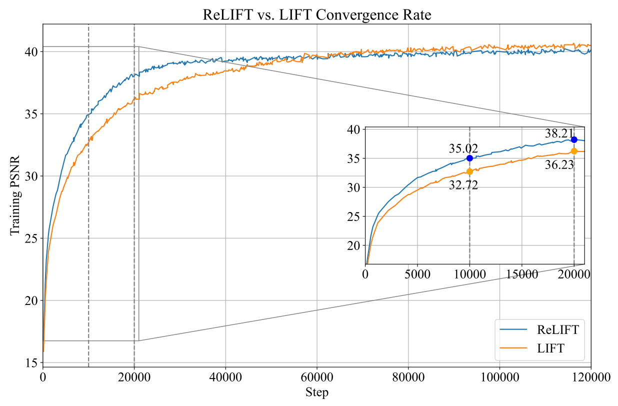


Figure 6. Convergence rate comparison of ReLIFT and LIFT.

regions but introduces artifacts and struggles with intricate features, resulting in less accurate reconstructions. WIRE produces overly smoothed approximations with significant detail loss, making it suitable only for coarse representations. ReLU+P.E. benefits from positional encoding to improve spatial structure over WIRE and SIREN but does not reach the fidelity and precision of ReLIFT and FINER.

Table 6. Quantitative comparisons of SDF representations.

	Methods	Armadillo	Dragon	Lucy	Thai Statue	Avg.
IOU ↑	ReLU+P.E.	0.9966	0.9963	0.9919	0.9906	0.9939
	SIREN	0.9968	0.9969	0.9881	0.9934	0.9938
	WIRE	0.9677	0.9724	0.9705	0.9484	0.9648
	FINER	0.9958	0.9945	0.9955	0.9919	0.9944
	ReLIFT	0.9974	0.9975	0.9960	0.9943	0.9963

5.1.3 Audio Representations

Data. For our audio representation task, we use the initial 7 seconds of Bach's Cello Suite No. 1: Prelude [31], sampled at a rate of 44100 Hz.

Analysis. We evaluate ReLIFT's performance against other methods to assess its effectiveness in audio signal represen-

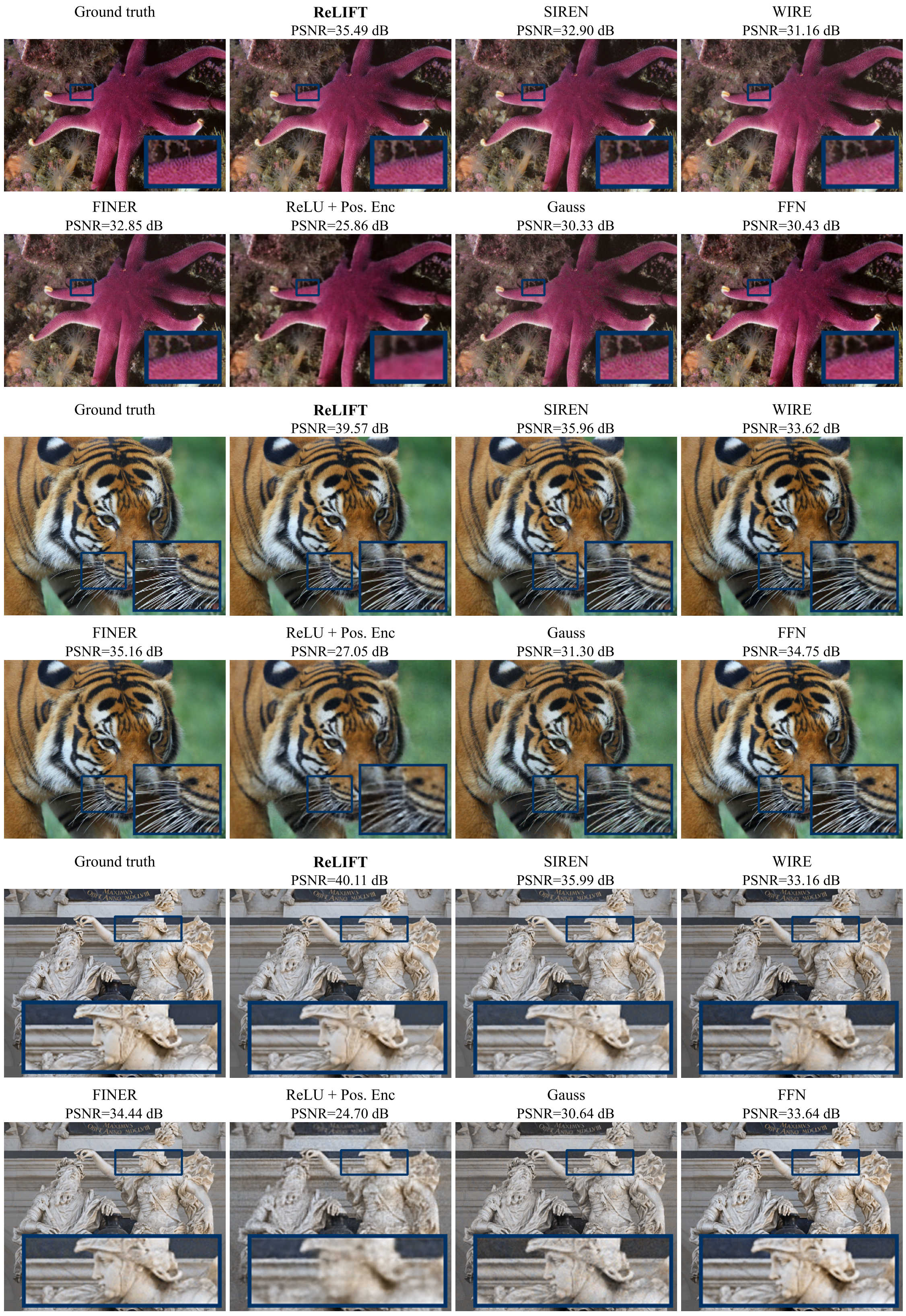


Figure 7. **Image representation:** PSNR comparisons of ReLIFT with SOTA models.

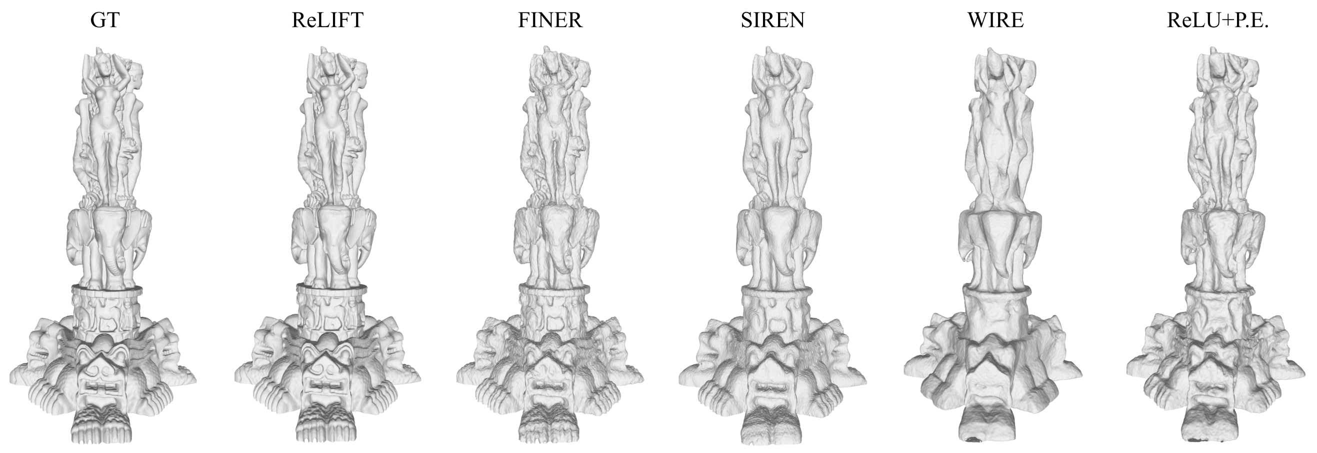


Figure 8. **Shape representation:** Qualitative comparisons of ReLIFT with SOTA models.

tation. Given a 1D point, the INR learns a mapping function $f : \mathbb{R} \to \mathbb{R}$. ReLIFT leverages its residual capabilities and scaling factor, allowing us to use a higher frequency scaling value w_0. For ReLIFT, we set the first layer ω_0 to 10000, scaled by $\gamma = 2$, and use hidden layers with $\omega_0 = 90$. In contrast, the original SIREN architecture suffers when high values of ω_0 are used. According to Neural Tangent Kernel (NTK) analysis [15] in [37], excessively large ω_0 values can lead to a poorly conditioned NTK, where certain eigenvalues become too small. This issue hinders effective learning, resulting in SIREN's poor performance on high-frequency representations. By addressing this challenge, ReLIFT achieves a PSNR of 54.99 dB, outperforming SIREN and other methods, with FINER as the next best at 46.56 dB—a difference of +8.43 dB (see Figure 9). The periodic nature of audio signals allows ReLIFT to produce a clear and accurate representation, similar to SIREN, but without the background noise issues. ReLIFT quickly reaches a low-error representation, while methods like Gauss, WIRE, and ReLU+P.E. introduce more noticeable distortion during playback. SIREN and FINER reduce this problem up to a point, but background noise is still present. Overall, ReLIFT performs best in minimizing error, as reflected in its PSNR value and reconstruction error.

5.2. Inverse Problems

5.2.1 Image Super-resolution

Data. An image from the DIV2K dataset [34] is used, with downsampling applied from an original resolution of 1356 × 2040 × 3 by scaling factors of 1/2, 1/4, and 1/6.

In super-resolution. INRs serve as effective interpolants, using their natural strengths and inherent biases to enhance performance. To test this idea, we performed $1\times$, $2\times$, $4\times$, and $6\times$ super-resolution experiments on an image. The results in Table 7 show that ReLIFT achieves the highest PSNR and SSIM values at every super-resolution level, outperforming other top methods. For instance, ReLIFT reaches a PSNR of 34.30 and an SSIM of 0.94 at $1\times$ resolution and keeps a strong performance up to $6\times$ resolution with a PSNR of 27.28 and an SSIM of 0.85. Visual comparisons Figure 10 also show that ReLIFT preserves sharper details, while other methods tend to produce blurrier results, highlighting ReLIFT's ability to maintain high-quality reconstruction.

Table 7. ReLIFT vs. SOTAs in super-resolution.

Methods	1×		2×		4×		6×	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Gauss	30.32	0.79	29.15	0.86	26.92	0.83	25.17	0.81
FFN	32.83	0.90	29.28	0.85	29.03	0.86	27.05	0.84
ReLU P.E.	32.46	0.87	30.41	0.88	26.10	0.83	24.61	0.81
WIRE	31.63	0.85	31.28	0.86	28.61	0.83	24.76	0.70
SIREN	32.02	0.87	31.55	0.89	28.95	0.87	26.37	0.84
FINER	34.35	0.89	32.45	0.90	29.31	0.87	26.94	0.83
ReLIFT	34.30	0.94	33.07	0.90	30.23	0.89	27.28	0.85

5.2.2 Image Denoising

Data. We use an image from the DIV2K dataset [34], downsampled by a factor of 1/4 from an original resolution of $1356 \times 2040 \times 3$ to $339 \times 510 \times 3$. To simulate realistic sensor noise, we apply photon and readout noise, where each pixel is affected by independent Poisson random variables. The mean photon count (τ) is set to 40, and the readout count is fixed at 2.

Analysis. We demonstrate ReLIFT's effectiveness in tackling inverse problems, especially in image denoising, by leveraging its inductive bias and robustness to noise. To manage high-frequency noise patterns in noisy images, we set the first layer scaling parameter to $\omega_0 = 10$, which helps ReLIFT maintain a balance between low- and high-frequency information—a configuration we also apply to SIREN and WIRE for comparison. As shown in Fig-

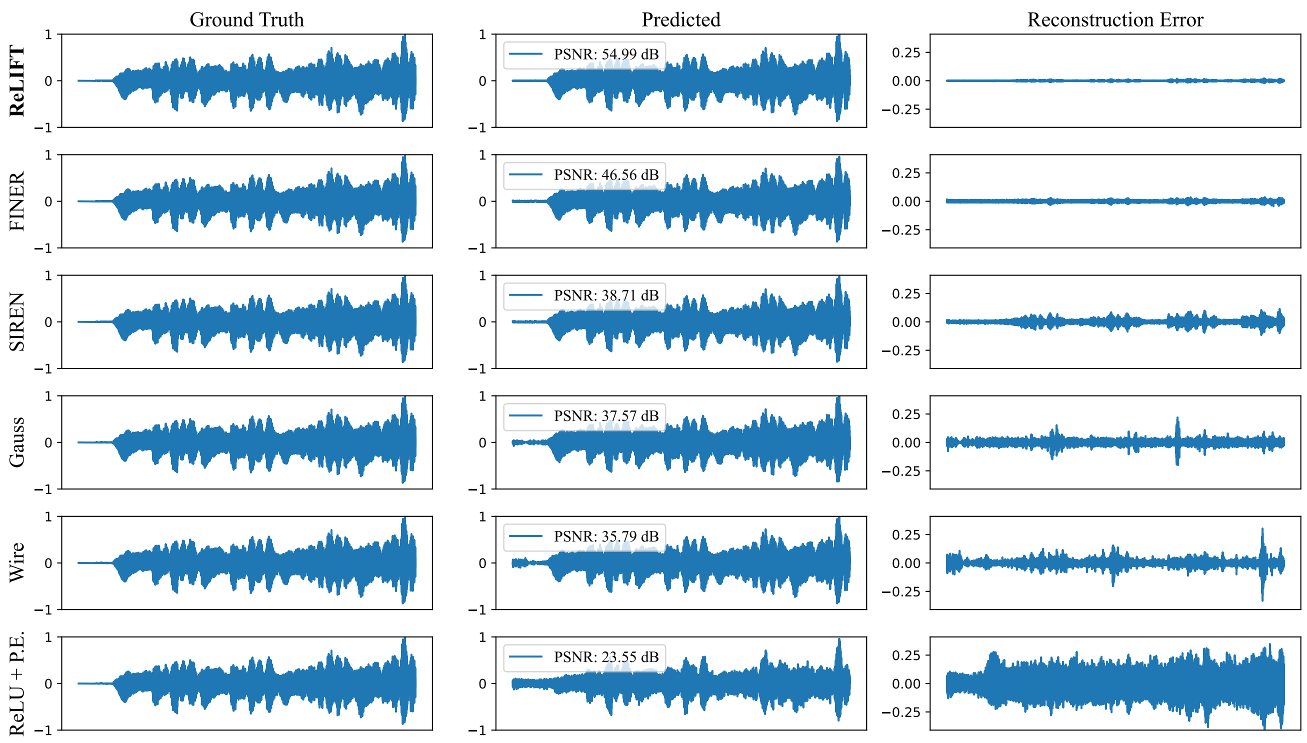


Figure 9. **Audio representation:** PSNR and reconstruction error comparisons of ReLIFT with SOTA models.

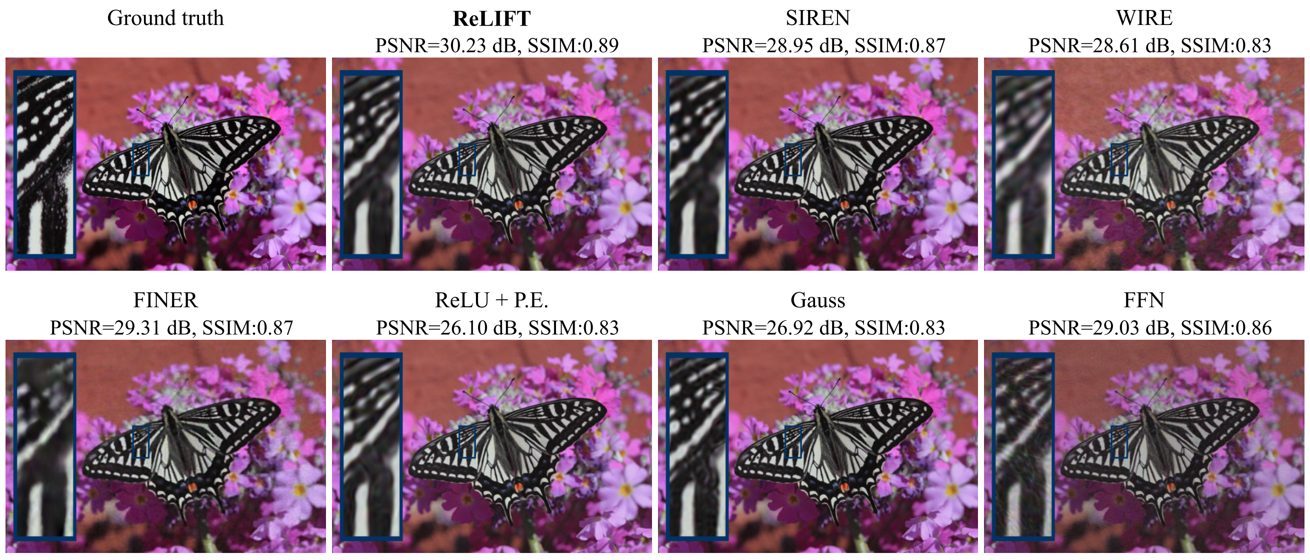


Figure 10. **Image Super-resolution:** PSNR and SSIM comparisons of a 4× single image super-resolution between ReLIFT and SOTA models.

ure 11, ReLIFT achieves significant improvements, including a PSNR gain of +10.84 dB and an SSIM increase of 0.45 over the original noisy image. ReLIFT effectively preserves image details while reducing noise artifacts, as evident in the zoomed-in areas, where both SIREN and WIRE exhibit over-smoothed details. While ReLU-based networks primarily capture low-frequency information, ReLIFT excels at balancing low- and high-frequency features benefiting from its optimized scaling and residual connections.

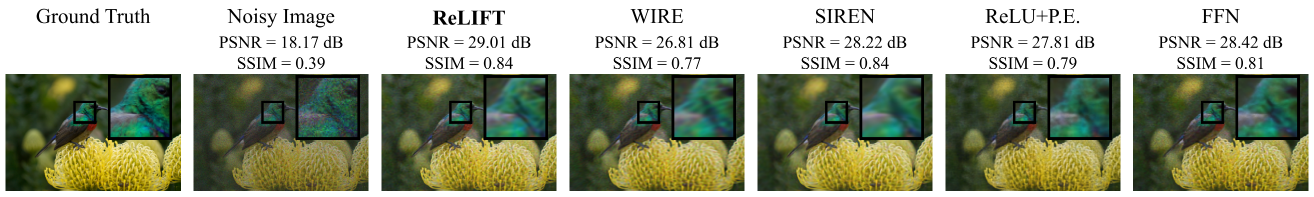


Figure 11. **Image Denoising:** PSNR and SSIM comparisons between ReLIFT and SOTA models.

5.2.3 Inpainting

Data. For the inpainting experiment, we use a $572 \times 582 \times 3$ image, where 25% of the pixels are masked as shown in the ”Training Data” column in Figure 12. The masked pixels serve as missing data points that the models aim to reconstruct, while the remaining pixels provide context for the inpainting process. This setup tests each model’s ability to restore missing information while preserving image quality.
Analysis. In Figure 12, we compare the performance of ReLIFT with other SOTA methods, including FINER, SIREN, and ReLU+P.E., on the inpainting task. ReLIFT achieves the highest PSNR at 22.40 dB, outperforming FINER (22.17 dB), SIREN (22.02 dB), and ReLU+P.E. (21.43 dB). This improvement demonstrates ReLIFT’s superior capability to restore missing details with greater accuracy. In the zoomed-in regions, we observe that ReLIFT maintains sharper edges and textures compared to other methods, which tend to produce more blurred or smoothed reconstructions. FINER follows closely behind ReLIFT in terms of PSNR, showing reasonable inpainting quality but slightly softer details. SIREN and ReLU+P.E. further lag in performance, with more pronounced blurring in the reconstructed areas, suggesting less effective handling of fine textures and edges. ReLIFT’s advantage stems from its design, which effectively balances high- and low-frequency features, allowing it to capture both broad structures and finer details in the inpainting process.

6. ReLIFT Spectral Bias

Previous work by Rahaman et al. [25] has shown that MLP-based networks exhibit a spectral bias, where lower frequency components are learned more rapidly than higher ones. To investigate this spectral bias within our network, we adopted [30] experimental framework using a 1D periodic function composed of four primary frequencies, as defined in Equation 19. The function $f(x)$ was sampled at 300 points over the interval $[-1, 1]$.

$$f(x) = 2R\left(\frac{\sin(3\pi x) + \sin(5\pi x) + \sin(7\pi x) + \sin(9\pi x)}{2}\right) \tag{19}$$

In this equation, R represents a rounding function that introduces discontinuities, thereby increasing the complexity of the training process. Our network architecture consists of a multilayer perceptron (MLP) with three hidden layers, each containing 128 neurons. We set the initial frequency parameter $\omega_0 = 5$ for both the SIREN model and our proposed method, ReLIFT, with a scaling factor $\gamma = 2$.
To evaluate the effectiveness of our approach, we trained both SIREN and ReLIFT on the defined function and compared their frequency learning dynamics. As illustrated in Figure 13, SIREN demonstrates a spectral bias by quickly learning the low-frequency components while struggling to capture the high-frequency ones. In contrast, our ReLIFT method successfully balances the learning of both low and high-frequency components, thereby mitigating the spectral bias inherent in standard MLP-based networks.

7. ReLIFT Layer Visualization

The layer visualization for ReLIFT, SIREN, FINER, and WIRE are depicted in Figure 14. We analyze each method below.

7.1. ReLIFT

ReLIFT displays structured and coherent spatial patterns across all layers, indicating strong feature retention. The patterns remain intricate even in the deeper layers, suggesting that ReLIFT’s design allows it to capture and retain fine-grained details across the network depth. The PSNR of 37.11 dB aligns with the visual clarity of the patterns, as ReLIFT seems to retain more information with minimal degradation in detail.

7.2. SIREN:

SIREN starts with sharp, detailed patterns in the initial layers, but these become progressively noisier and less defined in the later layers. While the initial layers retain significant variance, later layers lose high-frequency details. This could be due to the sinusoidal activation potentially introducing noise or leading to over-smooth representations in deeper layers. The PSNR of 34.10 dB reflects this slight degradation in detail. Although SIREN can capture complex features initially, it may struggle to maintain high fidelity across multiple layers, making it somewhat less effective than ReLIFT for fine-detail learning.

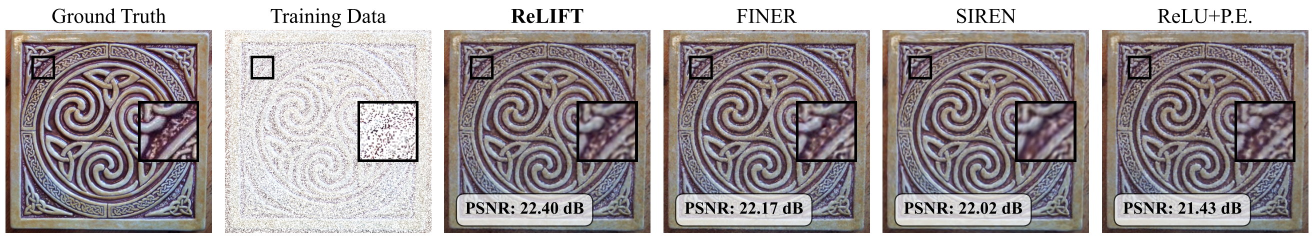


Figure 12. **Inpainting:** PSNR comparison between ReLIFT and SOTA models on 25% of the pixels in a $572 \times 582 \times 3$ image.

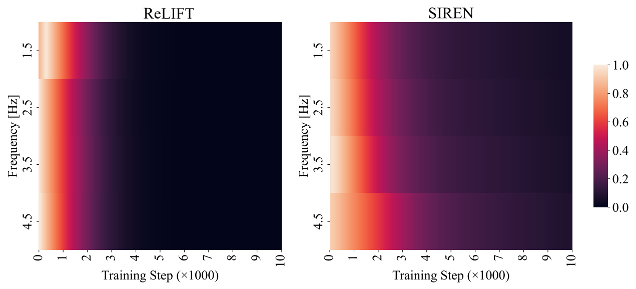


Figure 13. Frequency learning comparison between SIREN and ReLIFT. The x-axis shows training steps, the y-axis indicates frequency, and the color represents relative approximation error.

7.3. FINER:

FINER shows larger, blocky patterns that lose resolution as layers deepen, indicating a shift towards lower frequencies and a loss of spatial detail. This pattern suggests that FINER might prioritize coarse information, sacrificing fine-grained structure in favor of more generalized features. Due to the blocky representations, the filters in FINER likely have less variation in high frequencies, which is why the visualizations appear more uniform and less detailed compared to ReLIFT. With a PSNR of 33.49 dB, FINER generates an output that is less detailed and slightly blurry.

7.4. WIRE:

WIRE quickly loses visible structure across layers, with activations fading into dark, nearly uniform representations by the later layers. This rapid loss of detail suggests that WIRE lacks mechanisms for retaining spatial structure across depth, perhaps due to an architecture that does not prioritize high-frequency detail retention. The lack of high-variance components leads to flatter, less informative patterns, which contribute to lower-quality visualizations and limited information retention. The PSNR of 31.06 dB reflects significant degradation in detail, resulting in a final output that is overly smooth and lacking in fidelity.

8. Limitations and Discussion

While our approach delivers promising results, certain limitations are worth discussing. In image-generation experiments, the generated outputs exhibit slight blurriness, which we hypothesize arises from the use of the sinusoidal activation function, potentially leading to over-smoothed representations. For 3D generation, we believe that fine-tuning the hyperparameters of the ADM model could further improve sample quality and evaluation scores. Compared to global-based methods that struggle with high-resolution generation, our hierarchical design effectively integrates global context with fine-grained local details. This multiscale representation is crucial for downstream tasks. For example, Functa [10] achieves CIFAR-10 classification accuracies of 68.3%, 68.3%, and 66.7% for latent sizes of 256, 512, and 1024, respectively, indicating that relying solely on larger global latents does not capture task-specific features effectively. Moreover, our method achieves significantly better classification performance compared to local-based approaches [3] and pixel-based methods [8, 12, 20, 35], demonstrating that effectively leveraging both local and global information is crucial for downstream tasks. In addition to its superior accuracy, our approach is also highly efficient. For instance, our higher-resolution CelebA-HQ (64^2) model exhibits superior computational performance and scalability compared to the lower-resolution CIFAR-10 (32^2) dataset used in Spatial Functa. The high computational demands of Spatial Functa make it challenging to extend to 3D data, whereas our framework remains computationally feasible. These results reinforce the importance of balancing global context with local details to obtain more expressive and robust latent representations (see Experiments).

We also explored alternative designs, such as using different activation functions for varying latent resolutions. This resulted in a modulated activation function expressed as:

$$\Sigma_{s=1}^{M} \lambda_s \sin\left(W_0(Wx + m_{s\times s})\right),$$

where $m_{s\times s}$ is the modulation latent with a size of $s \times s$, combined with a scaling factor λ_s to emphasize local latents. However, this modification led to a marginal reduc-

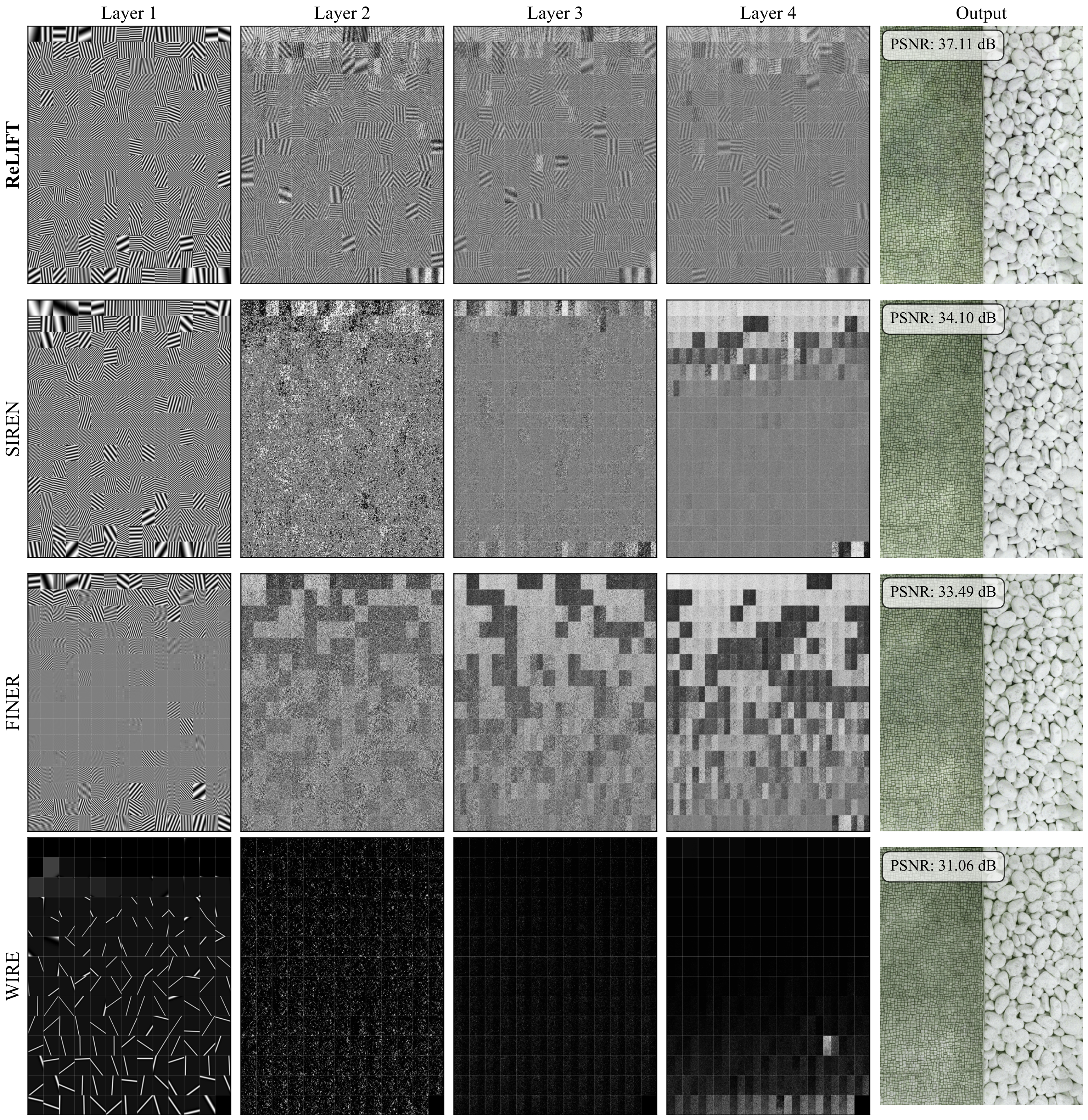


Figure 14. Visualization of hidden layer outputs.

tion in reconstruction performance compared to the hierarchical design. Furthermore, we tested removing the hierarchical modulation module and instead introduced a learnable scaling modulation factor in addition to the shift modulations. While this approach simplifies the architecture, it resulted in a noticeable drop in performance, underscoring the importance of hierarchical modulation in achieving robust and high-quality representations.

Overall, our reconstruction model, LIFT, is distinguished by its speed and scalability across various resolutions, making it highly effective for both low and high-resolution data. Moreover, LIFT converges rapidly within just a few iterations, significantly shortening training durations. Notably, the ReLIFT variant accelerates convergence even further, proving especially beneficial when training time is constrained. Consequently, our framework is designed to tackle the challenges of large-scale datasets with LIFT, as well as address single-data scenarios using ReLIFT.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018. 4

[2] Ambityga. Imagenet-100 dataset, 2024. Accessed: 2024-11-18. 4

[3] Matthias Bauer, Emilien Dupont, Andy Brock, Dan Rosenbaum, Jonathan Richard Schwarz, and Hyunjik Kim. Spatial functa: Scaling functa to imagenet classification and generation. *arXiv preprint arXiv:2302.03130*, 2023. 15

[4] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, et al. JAX: composable transformations of Python+ NumPy programs. 2018. 4

[5] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 4

[6] 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. 4

[7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 3, 6

[8] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 15

[9] Yilun Du, Katie Collins, Josh Tenenbaum, and Vincent Sitzmann. Learning signal-agnostic manifolds of neural fields. *Advances in Neural Information Processing Systems*, 34: 8320–8331, 2021. 3, 4

[10] Emilien Dupont, Hyunjik Kim, SM Ali Eslami, Danilo Jimenez Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one. In *International Conference on Machine Learning*, pages 5694–5725. PMLR, 2022. 3, 4, 15

[11] Emilien Dupont, Yee Whye Teh, and Arnaud Doucet. Generative models as distributions of functions. In *International Conference on Artificial Intelligence and Statistics*, pages 2989–3015. PMLR, 2022. 2, 3

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 8, 15

[13] Tom Hennigan, Trevor Cai, and Peter Norman. Haiku: Sonnet for JAX, 2020. 4

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 4

[15] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018. 12

[16] Tero Karras. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 4

[17] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 9

[18] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). 4

[19] Stanford Computer Graphics Laboratory. The stanford 3d scanning repository, 2014. Accessed: [date of access, 2024-11-09]. 9

[20] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022. 15

[21] Zhen Liu, Hao Zhu, Qi Zhang, Jingde Fu, Weibing Deng, Zhan Ma, Yanwen Guo, and Xun Cao. Finer: Flexible spectral-bias tuning in implicit neural representation by variable-periodic activation functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2713–2722, 2024. 9

[22] 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. 9

[23] Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models. In *International Conference on Machine Learning*, pages 7176–7185. PMLR, 2020. 4

[24] Dogyun Park, Sihyeon Kim, Sojin Lee, and Hyunwoo J. Kim. DDMI: Domain-agnostic latent diffusion models for synthesizing high-quality implicit neural representations. In *The Twelfth International Conference on Learning Representations*, 2024. 3

[25] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International conference on machine learning*, pages 5301–5310. PMLR, 2019. 14

[26] Sameera Ramasinghe and Simon Lucey. Beyond periodicity: Towards a unifying framework for activations in coordinate-mlps. In *European Conference on Computer Vision*, pages 142–158. Springer, 2022. 9

[27] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015. 4

[28] Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. *Advances in neural information processing systems*, 31, 2018. 4

[29] Vishwanath Saragadam, Daniel LeJeune, Jasper Tan, Guha Balakrishnan, Ashok Veeraraghavan, and Richard G Baraniuk. Wire: Wavelet implicit neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18507–18516, 2023. 9

[30] Kexuan Shi, Xingyu Zhou, and Shuhang Gu. Improved implicit neural representation with fourier reparameterized training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25985–25994, 2024. 14

[31] 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. 8, 9, 10

[32] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 6

[33] 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. 9

[34] Radu Timofte, Shuhang Gu, Jiqing Wu, Luc Van Gool, Lei Zhang, Ming-Hsuan Yang, Muhammad Haris, Greg Shakhnarovich, Norimichi Ukita, Shijia Hu, Yijie Bei, Zheng Hui, Xiao Jiang, Yanan Gu, Jie Liu, Yifan Wang, Federico Perazzi, Brian McWilliams, Alexander Sorkine-Hornung, Olga Sorkine-Hornung, Christopher Schroers, Jiahui Yu, Yuchen Fan, Jianchao Yang, Ning Xu, Zhaowen Wang, Xinchao Wang, Thomas S. Huang, Xintao Wang, Ke Yu, Tak-Wai Hui, Chao Dong, Liang Lin, Chen Change Loy, Dongwon Park, Kwanyoung Kim, Se Young Chun, Kai Zhang, Pengjv Liu, Wangmeng Zuo, Shi Guo, Jiye Liu, Jinchang Xu, Yijiao Liu, Fengye Xiong, Yuan Dong, Hongliang Bai, Alexandru Damian, Nikhil Ravi, Sachit Menon, Cynthia Rudin, Junghoon Seo, Taegyun Jeon, Jamyoung Koo, Seunghyun Jeon, Soo Ye Kim, Jae-Seok Choi, Sehwan Ki, Soomin Seo, Hyeonjun Sim, Saehun Kim, Munchurl Kim, Rong Chen, Kun Zeng, Jinkang Guo, Yanyun Qu, Cuihua Li, Namhyuk Ahn, Byungkon Kang, Kyung-Ah Sohn, Yuan Yuan, Jiawei Zhang, Jiahao Pang, Xiangyu Xu, Yan Zhao, Wei Deng, Sibt Ul Hussain, Muneeb Aadil, Rafia Rahim, Xiaowang Cai, Fang Huang, Yueshu Xu, Pablo Navarrete Michelini, Dan Zhu, Hanwen Liu, Jun-Hyuk Kim, Jong-Seok Lee, Yiwen Huang, Ming Qiu, Liting Jing, Jiehang Zeng, Ying Wang, Manoj Sharma, Rudrabha Mukhopadhyay, Avinash Upadhyay, Sriharsha Koundinya, Ankit Shukla, Santanu Chaudhury, Zhe Zhang, Yu Hen Hu, and Lingzhi Fu. Ntire 2018 challenge on single image super-resolution: Methods and results. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 965–96511, 2018. 9, 12

[35] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 15

[36] Tackgeun You, Mijeong Kim, Jungtaek Kim, and Bohyung Han. Generative neural fields by mixtures of neural implicit functions. *Advances in Neural Information Processing Systems*, 36, 2024. 3, 4

[37] Gizem Yüce, Guillermo Ortiz-Jiménez, Beril Besbinar, and Pascal Frossard. A structured dictionary perspective on implicit neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19228–19238, 2022. 6, 12

[38] Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. In *Forty-first International Conference on Machine Learning*, 2024. 6

[39] Peiye Zhuang, Samira Abnar, Jiatao Gu, Alex Schwing, Joshua M Susskind, and Miguel Angel Bautista. Diffusion probabilistic fields. In *The Eleventh International Conference on Learning Representations*, 2023. 3