

# Recurrent Sinusoidal INRs for Efficient High-Fidelity Representation

Hyunmin Cho<sup>1</sup>, Jaejun Yoo<sup>2</sup>, and Kyong Hwan Jin<sup>1\*</sup>

<sup>1</sup> Department of Electrical Engineering, Korea University, Seoul, South Korea  
{hyun\_cho,kyong\_jin}@korea.ac.kr

<sup>2</sup> Graduate School of Artificial Intelligence, UNIST, Ulsan, South Korea  
jaejun.yoo@unist.ac.kr

 Project Page: <https://hyeon-cho.github.io/Harmonic-line-Spectrum/>

**Abstract.** We study sinusoidal recurrence as an iterative mechanism for harmonic spectral enrichment in implicit neural representations (INRs). Our analysis reveals that sinusoidal activations induce a harmonic line spectrum, providing a spectral account of how recurrent unrolling enriches the effective spectral support. We realize this principle with a shared sinusoidal block that iteratively refines the latent representation. We empirically validate the resulting spectral behavior against feed-forward INRs, non-sinusoidal recurrent variants, and equilibrium-style sinusoidal models. Complementing this analysis, we evaluate the proposed architecture across image and 3D representation tasks. On RGB image benchmarks, our method achieves higher fidelity than feed-forward baselines with fewer parameters and fewer optimization steps, and it further transfers favorably to super-resolution, NeRF, and SDF tasks.

**Keywords:** Recurrent network · Implicit signal representation

## 1 Introduction

We study implicit neural representations (INRs) [18, 22, 25, 29, 31], which parameterize a signal as a neural function  $f$ , typically a coordinate-based MLP,

$$f : \mathbb{R}^d \rightarrow \mathbb{R}^c.$$

By treating the input as spatial or spatio-temporal coordinates, INRs represent continuous signals and can interpolate values at unseen locations. This property has made them a standard tool in applications such as 3D reconstruction and novel view synthesis [20, 22, 25].

A central challenge of coordinate-based MLPs is *spectral bias*: neural networks tend to fit low-frequency structure more readily than high-frequency structure, making faithful recovery of fine detail difficult [26, 32]. A large body of work has addressed this issue through activation design, input encodings, multiresolution parameterizations, and optimization strategies [12, 18, 19, 23, 31]. While

---

\* Correspondence to: Kyong Hwan Jin <kyong\_jin@korea.ac.kr>.

effective, many of these approaches improve fidelity by increasing model size, auxiliary parameters, or training complexity.

To reason about where high-frequency information is gained or lost, it is useful to decompose a typical INR into three stages. For a target signal  $\mathbf{s}$  evaluated at coordinates  $\mathbf{x}$ , we write

$$\mathbf{s} = \mathcal{D}(\mathcal{F}(\mathcal{E}(\mathbf{x}))), \quad (1)$$

where  $\mathcal{E}$  lifts low-dimensional coordinates into a higher-dimensional representation [23, 32],  $\mathcal{F}$  transforms this representation into a latent feature, and  $\mathcal{D}$  maps the latent feature to the output space. Under this view, high-fidelity reconstruction depends not only on the coordinate features introduced by  $\mathcal{E}$ , but also on how  $\mathcal{F}$  composes and refines them into representations capable of resolving fine-scale structure.

A straightforward way to improve the ability of  $\mathcal{F}$  to resolve fine-scale structure is to increase its representational capacity through deeper or wider latent transformations. While effective, this approach introduces additional independently parameterized layers and increased computation. This leaves open a complementary question: *can  $\mathcal{F}$  be made more effective at resolving fine-scale structure without increasing independently parameterized depth?*

Recurrence provides a natural mechanism for such refinement. Unrolling a recurrent update for  $T$  steps increases effective depth while reusing the same parameters, offering a parameter-efficient alternative to independently parameterized depth [3, 8, 30]. Related INR formulations have also explored iterative latent computation through equilibrium networks, whose fixed-point formulation enables constant-memory backpropagation with respect to effective depth [3, 13]. However, these equilibrium formulations are motivated primarily by memory-efficient training and amortized fixed-point solving, while how repeated latent transformations can be made more effective for fine-scale reconstruction remains a separate question.

To this end, we interpret sinusoidal recurrence as an iterative mechanism for harmonic spectral enrichment. We show that sinusoidal activations induce harmonic line spectra, providing a spectral rationale for how repeated application of a shared sinusoidal block can enrich effective spectral support without adding independently parameterized depth. In practice, this formulation achieves higher-fidelity image reconstruction with fewer parameters and optimization steps, while transferring favorably to super-resolution, NeRF, and SDF tasks.

- We formulate weight-tied sinusoidal refinement for INRs, increasing effective depth through finite recurrent unrolling without adding independently parameterized layers.
- We provide a harmonic line-spectrum analysis of sinusoidal transformations, offering a spectral interpretation of how repeated sinusoidal refinement can enrich effective spectral support.
- We show that this formulation achieves higher-fidelity image reconstruction with fewer parameters and optimization steps than feed-forward baselines, while its decoder transfers favorably to continuous representation tasks.

## 2 Preliminaries: Design Strategies for High-Fidelity Implicit Signal Representation

***Implicit Neural Representations and Spectral Bias.*** Implicit neural representations model signals as continuous coordinate-based functions parameterized by neural networks (e.g.,  $f_\theta : (x, y) \in \mathbb{R}^2 \mapsto (r, g, b) \in \mathbb{R}^3$ ). This formulation supports continuous sampling, interpolation, and resolution-independent evaluation [7, 22]. Despite their versatility, a central challenge is spectral bias. Coordinate MLPs tend to fit low-frequency structure before fine-scale detail [6, 26], motivating architectural choices for high-frequency representation.

***Enhancing Representational Capacity through Activation Functions.*** Prior work improves high-frequency representation by modifying the activation functions of coordinate MLPs. SIREN replaces standard nonlinearities with periodic sine activations, yielding coordinate networks with strong capacity for representing complex signals [31]. FINER further adapts sinusoidal activations to cover a broader range of frequencies and improve reconstruction stability [18]. Beyond periodic functions, Gaussian activations provide spatially localized alternatives [27], while WIRE employs complex Gabor wavelets that combine sinusoidal oscillation with Gaussian localization [29].

***Enriching Coordinate Features with Multi-Frequency Encodings.*** Rather than modifying the activation function, another line of work enriches the input coordinates with multi-frequency encodings. NeRF applies positional encoding to map coordinates to a multi-frequency sinusoidal basis [22], while Random Fourier Features (RFF) use sampled sinusoidal projections to expose diverse input frequencies [32]. Multiresolution encodings further augment coordinate features through learned grid or hash-based representations [23]. By providing richer frequency content at the input, these methods improve the ability of INRs to fit fine-scale signals.

## 3 Latent Refinement in INRs

Beyond modifying activations or enriching coordinate inputs, another design choice concerns how latent transformations are parameterized and evaluated across depth. We focus on this design axis through the latent transformation  $\mathcal{F}$  in Eq. 1.

A feed-forward latent transformation applies a sequence of independently parameterized affine maps and nonlinearities,

$$\mathbf{h}^{(\ell+1)} = \sigma_\ell \left( \mathbf{W}_\ell \mathbf{h}^{(\ell)} + \mathbf{b}_\ell \right), \quad \ell = 0, \dots, L-1, \quad (2)$$

where  $\mathbf{h}^{(0)} = \mathcal{E}(\mathbf{x})$  and each layer has its own parameters  $(\mathbf{W}_\ell, \mathbf{b}_\ell)$ . Increasing  $L$  can enrich the latent transformation, but generally requires additional independently parameterized layers.

For high-fidelity signal representation, the structure of such latent transformations can also be designed to control how frequency content is processed across depth. BACON incorporates band-limited sinusoidal filtering, while FourierNet and GaborNet construct frequency-aware feed-forward transformations [9, 17].

A separate line of work introduces iterative latent computation through an equilibrium formulation. Specifically, in equilibrium-style INR proposed by *Huang et al.* [13], the latent representation is defined implicitly as a fixed point,

$$\mathbf{h}^* = \mathcal{G}_\theta(\mathbf{h}^*; \mathbf{h}^{(0)}), \quad \mathbf{h}^{(0)} = \mathcal{E}(\mathbf{x}), \quad (3)$$

enabling constant-memory backpropagation via implicit differentiation [3] and amortized fixed-point solving.

Fixed-point formulations, however, characterize the latent computation only at equilibrium, making the evolution induced by successive applications of the shared map less directly visible. We therefore consider a finite, explicitly unrolled refinement trajectory, whose intermediate states allow us to examine how repeated transformations alter the latent representation.

## 4 Harmonic Enrichment by Sinusoidal Recurrence

To analyze this stepwise refinement, we characterize the spectral structure induced by sinusoidal transformations. We show that sinusoidal layers induce a *harmonic line-spectrum* in intermediate representations, providing a Fourier-series interpretation of both depth and weight-tied unrolling.

### 4.1 Input layer as a learnable positional embedder

Following *Sitzmann et al.* [31], we define the sinusoidal coordinate lifting as

$$\begin{aligned} \mathbf{h}_0(\mathbf{x}) &\triangleq \mathcal{E}(\mathbf{x}) \\ &= \sin(\omega_{\text{in}} \mathbf{W}_{\text{in}} \mathbf{x} + \mathbf{b}_{\text{in}}), \end{aligned} \quad (4)$$

where  $\omega_{\text{in}}$  controls the bandwidth of the embedding. Writing the  $i$ -th channel explicitly,

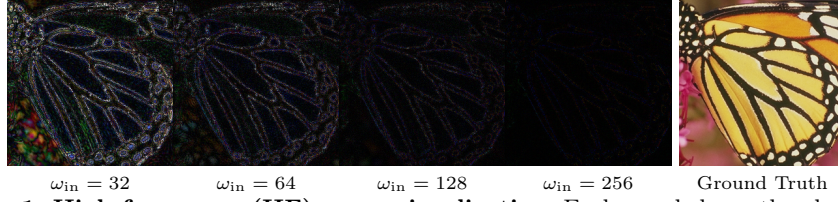
$$[\mathbf{h}_0(\mathbf{x})]_i = \sin(\boldsymbol{\Omega}_i^\top \mathbf{x} + \phi_i), \quad \boldsymbol{\Omega}_i \triangleq \omega_{\text{in}} \mathbf{w}_i, \quad \phi_i \triangleq b_i, \quad (5)$$

makes clear that the encoder maps the input coordinates to a collection of sinusoidal basis functions, each parameterized by a learnable frequency vector  $\{\boldsymbol{\Omega}_i\}_{i=1}^m$  and phase  $\{\phi_i\}$ . Accordingly,  $\mathcal{E}$  can be understood as a *learnable* Fourier feature map, which is consistent with the Fourier-series interpretation of sinusoidal implicit neural representations studied by *Benbarka et al.* [4]. In contrast

**Table 1:** Frequency-separated reconstruction quality over training steps. Each cell reports PSNR in dB as  $\text{Full}_{\text{LF}}^{\text{HF}}$ , where Full is the PSNR on the full image, LF is the PSNR on the low-pass component, and HF is the PSNR on the high-pass component.

| Inp. freq. ( $\omega_{\text{in}}$ ) | #Iteration                              |                                         |                                         |                                         |                                         |
|-------------------------------------|-----------------------------------------|-----------------------------------------|-----------------------------------------|-----------------------------------------|-----------------------------------------|
|                                     | 200                                     | 400                                     | 600                                     | 800                                     | 1000                                    |
| 32                                  | 22.73 <sub>37.29</sub> <sup>23.51</sup> | 24.64 <sub>40.50</sub> <sup>25.10</sup> | 25.56 <sub>39.01</sub> <sup>26.04</sup> | 26.00 <sub>37.33</sub> <sup>26.61</sup> | 26.46 <sub>37.30</sub> <sup>27.09</sup> |
| 64                                  | 26.35 <sub>45.19</sub> <sup>26.60</sup> | 28.91 <sub>48.97</sub> <sup>29.05</sup> | 30.41 <sub>50.26</sub> <sup>30.51</sup> | 31.42 <sub>50.10</sub> <sup>31.53</sup> | 32.04 <sub>46.75</sub> <sup>32.26</sup> |
| 128                                 | 30.22 <sub>51.84</sub> <sup>30.31</sup> | 33.89 <sub>55.74</sub> <sup>33.94</sup> | 36.11 <sub>56.00</sub> <sup>36.18</sup> | 37.62 <sub>54.83</sub> <sup>37.74</sup> | 38.74 <sub>53.98</sub> <sup>38.91</sup> |
| 256                                 | 36.38 <sub>56.47</sub> <sup>36.45</sup> | 40.52 <sub>56.45</sub> <sup>40.68</sup> | 43.32 <sub>56.46</sub> <sup>43.62</sup> | 45.39 <sub>56.54</sub> <sup>45.87</sup> | 47.03 <sub>56.64</sub> <sup>47.71</sup> |





**Fig. 1: High-frequency (HF) error visualization.** Each panel shows the absolute difference between the predicted and ground-truth *high-pass* components,  $|\text{HP}(\hat{I}) - \text{HP}(I_{\text{GT}})|$ , where  $\text{HP}(I) = I - \text{GaussianBlur}(I)$ . From left to right we vary the frequency scale  $\omega_{\text{in}} \in \{32, 64, 128, 256\}$  in the sinusoidal lifting (Eq. 4), which sets the overall bandwidth of the learnable Fourier-feature embedding.

to Random Fourier Features [32], whose frequencies are fixed after initialization, the spectral components here are optimized end-to-end, while  $\omega_{\text{in}}$  directly determines the overall frequency scale available to the model.

This interpretation is supported empirically by the frequency-separated reconstruction results in Fig. 1 and Table 1. As  $\omega_{\text{in}}$  increases, the model consistently achieves better reconstruction on the high-pass component across all training stages, while also improving full-image PSNR. A similar trend is already visible at early optimization stages (e.g.,  $23.51 \rightarrow 36.45$  dB at 200 iterations).

## 4.2 Hidden sinusoidal layers as structured harmonic expansion

We now show how hidden sinusoidal layers transform the encoder’s learnable sine bases into richer line spectra. Recall from Sec. 4.1 that the sinusoidal encoder produces a finite collection of basis tones:

$$\mathbf{h}_0(\mathbf{x}) = \sin(\omega_{\text{in}} \mathbf{W}_{\text{in}} \mathbf{x} + \mathbf{b}_{\text{in}}) \in \mathbb{R}^m, \quad [\mathbf{h}_0(\mathbf{x})]_i = \sin(\boldsymbol{\Omega}_i^\top \mathbf{x} + \phi_i). \quad (6)$$

Let  $\theta_i(\mathbf{x}) \triangleq \boldsymbol{\Omega}_i^\top \mathbf{x} + \phi_i$ . For clarity, we first analyze the *first* hidden sinusoidal layer exactly:

$$\mathbf{h}_1(\mathbf{x}) = \sin(\omega \mathbf{W} \mathbf{h}_0(\mathbf{x}) + \mathbf{b}), \quad \mathbf{W} \in \mathbb{R}^{n \times m}, \mathbf{b} \in \mathbb{R}^n, \quad (7)$$

where  $\sin(\cdot)$  is applied elementwise.<sup>3</sup> It suffices to examine one scalar pre-activation, corresponding to the  $j$ -th channel:

$$u_j(\mathbf{x}) \triangleq [\omega \mathbf{W} \mathbf{h}_0(\mathbf{x}) + \mathbf{b}]_j = \omega \mathbf{w}_j^\top \mathbf{h}_0(\mathbf{x}) + b_j, \quad \mathbf{w}_j^\top := \mathbf{W}_{j,:}. \quad (8)$$

Substituting the encoder output from Eq. (6) into the pre-activation Eq. (8), and expanding  $\mathbf{w}_j^\top \mathbf{h}_0(\mathbf{x}) = \sum_{i=1}^m w_{j,i} [\mathbf{h}_0(\mathbf{x})]_i$ , yields the finite multi-tone form:

$$u_j(\mathbf{x}) = b_j + \omega \sum_{i=1}^m w_{j,i} \sin(\theta_i(\mathbf{x})) = \beta_j + \sum_{i=1}^m \alpha_{j,i} \sin(\theta_i(\mathbf{x})), \quad (9)$$

where  $\beta_j := b_j$  and  $\alpha_{j,i} := \omega w_{j,i}$ .

<sup>3</sup> Although the formulation retains a bias term for generality, our implementation employs bias-free recurrent layers; Sec. 4.3 provides the motivation for this choice.

Applying the sinusoidal nonlinearity to this multi-tone input yields an explicit generalized Fourier expansion. Using Euler’s identity ( $\sin(u_j(\mathbf{x})) = \text{Im}(e^{iu_j(\mathbf{x})})$ ) together with the Jacobi–Anger identity [1], we obtain

$$\exp(iu_j(\mathbf{x})) = e^{i\beta_j} \sum_{\mathbf{k} \in \mathbb{Z}^m} \left( \prod_{i=1}^m J_{k_i}(\alpha_{j,i}) \right) \exp\left(i \sum_{i=1}^m k_i \theta_i(\mathbf{x})\right), \quad (10)$$

and therefore

$$\sin(u_j(\mathbf{x})) = \sum_{\mathbf{k} \in \mathbb{Z}^m} c_{j,\mathbf{k}} \sin\left(\beta_j + \sum_{i=1}^m k_i \theta_i(\mathbf{x})\right), \quad c_{j,\mathbf{k}} = \prod_{i=1}^m J_{k_i}(\alpha_{j,i}). \quad (11)$$

Eq. (11) makes the sinusoid-specific effect explicit: in other words, a hidden sine layer creates new spectral lines at integer combinations of the encoder frequencies—including sums, differences, and higher-order harmonics—rather than merely reweighting existing ones [28]. These frequencies take the form

$$\boldsymbol{\Omega}' = \sum_{i=1}^m k_i \boldsymbol{\Omega}_i, \quad \mathbf{k} \in \mathbb{Z}^m, \quad (12)$$

that is, integer combinations of the encoder frequencies, as also characterized in the harmonic analysis of sinusoidal networks by *Novello et al.* [24].

This exact calculation motivates the deeper-layer view. Once a hidden representation is itself regarded as a line spectrum, or a finite truncation of its dominant tones, applying another sinusoidal layer yields the same type of integer-combination closure. Thus, if  $\Omega^{(\ell)}$  denotes the set of prominent frequency vectors after layer  $\ell$ , then in this idealized sense

$$\Omega^{(\ell+1)} \subseteq \text{span}_{\mathbb{Z}}(\Omega^{(\ell)}). \quad (13)$$

Hence, increasing depth enables progressively higher-order harmonic interactions from the same coordinate lifting. The same viewpoint also motivates weight-tied unrolling: repeatedly applying a shared sinusoidal block realizes iterative harmonic refinement under a fixed parameter budget.

**Table 2:** Reconstruction quality at optimization step 500 as a function of the number of recurrent unrolling steps. ‘Feed-forward’ corresponds to a single pass, and #2–#5 denote recurrent steps  $R=2$ –5. Each entry is formatted as **PSNR** $_{\Delta}$ , where  $\Delta$  is the PSNR gain (dB) relative to the feed-forward baseline.

|             | #Recurrent Unrolling Steps |                        |                         |                         |                                |
|-------------|----------------------------|------------------------|-------------------------|-------------------------|--------------------------------|
|             | Feed-forward               | #2                     | #3                      | #4                      | #5                             |
| <b>PSNR</b> | 39.724 <sub>+0.0</sub>     | 46.362 <sub>+6.6</sub> | 54.772 <sub>+15.0</sub> | 60.843 <sub>+21.1</sub> | <b>63.378</b> <sub>+23.7</sub> |
| #Param.     | 593.7K                     | 593.7K                 | 593.7K                  | 593.7K                  | 593.7K                         |

Table 2 quantifies the practical effect of finite recurrent refinement under a fixed parameter budget. At optimization step 500, increasing the number of recurrent unrolling steps from a single feed-forward pass to  $R=5$  improves PSNR from 39.724 dB to 63.378 dB, with monotonic gains at every intermediate step under the same parameter count. We next examine whether this improvement is accompanied by systematic changes in the intermediate spectral structure.

On a regular  $H \times H$  coordinate grid, we analyze the intermediate feature maps  $\mathbf{h}^{(\ell)}$  by computing the per-channel 2-D DFT  $\mathbf{F}_c^{(\ell)} = \text{DFT}(\mathbf{h}_{:,c}^{(\ell)})$ , where  $|\mathbf{F}_c^{(\ell)}[u, v]|$  is the strength of channel  $c$  at frequency  $(u, v)$ . After channel averaging and peak

**Table 3: Spectral support  $S_\ell^\tau$  and radial HF $_\ell$ .**

| Architecture                                                      | Recurrent | Mode            | $S_\ell^\tau$ per depth/recurrent $\ell$ (%) |      |      |      |            |      |            |      |             |       | HF $_\ell$ at $\ell=0,8$ |       | PSNR<br>for ref. |
|-------------------------------------------------------------------|-----------|-----------------|----------------------------------------------|------|------|------|------------|------|------------|------|-------------|-------|--------------------------|-------|------------------|
|                                                                   |           |                 | 0                                            | 1    | 2    | 3    | 4          | 5    | 6          | 7    | 8           | 0     | 8                        |       |                  |
| <i>(a) Sinusoidal recurrent, untrained</i>                        |           |                 |                                              |      |      |      |            |      |            |      |             |       |                          |       |                  |
| SIREN                                                             | ✓         | $\omega=256/45$ | 30.9                                         | 26.5 | 34.3 | 43.7 | 52.0       | 61.3 | 72.3       | 85.0 | <b>94.6</b> | 0.022 | 0.101                    | —     |                  |
| FINER                                                             | ✓         | $\omega=256/45$ | 51.4                                         | 53.6 | 72.4 | 89.3 | <b>100</b> | 100  | 100        | 100  | 100         | 0.048 | 0.232                    | —     |                  |
| <i>(b) Sinusoidal recurrent, trained on Set 5</i>                 |           |                 |                                              |      |      |      |            |      |            |      |             |       |                          |       |                  |
| SIREN                                                             | ✓         | Float           | 31.0                                         | 17.1 | 17.2 | 19.1 | 20.5       | 12.9 | 16.4       | 28.1 | 61.7        | 0.020 | 0.011                    | 78.35 |                  |
| FINER                                                             | ✓         | Float           | 53.9                                         | 41.4 | 46.6 | 46.0 | 37.4       | 16.4 | 41.6       | 100  | 100         | 0.044 | 0.016                    | 85.62 |                  |
| ► <i>Sinusoidal-based Spectral methods</i>                        |           |                 |                                              |      |      |      |            |      |            |      |             |       |                          |       |                  |
| iSIREN [13]                                                       | ✓         | DEQ             | 32.4                                         | 33.5 | 33.5 | 33.5 | 33.5       | 33.5 | 33.5       | 33.5 | 33.5        | 0.008 | 0.008                    | 47.16 |                  |
| GaborNet [9]                                                      | ✗         | $L=3$           | 40.6                                         | 58.2 | 73.0 | 35.2 | —          | —    | —          | —    | —           | 0.038 | 0.008                    | 44.48 |                  |
| FourierNet [9]                                                    | ✗         | $L=3$           | 41.4                                         | 58.3 | 66.6 | 36.0 | —          | —    | —          | —    | —           | 0.050 | 0.008                    | 38.38 |                  |
| BACON [17]                                                        | ✗         | freq=128        | 0.9                                          | 6.3  | 14.0 | 15.1 | —          | —    | —          | —    | —           | 0.059 | 3e-4                     | 27.88 |                  |
| <i>(c) Non-sinusoidal recurrent, trained — spectrum collapses</i> |           |                 |                                              |      |      |      |            |      |            |      |             |       |                          |       |                  |
| Gauss                                                             | ✓         | $\sigma=30$     | 1.0                                          | 1.9  | 14.0 | 1.2  | <b>0.0</b> | 0.0  | 0.0        | 0.0  | 0.0         | 2e-4  | 7e-3                     | 12.02 |                  |
| PEMLP                                                             | ✓         | $N_f=10$        | 0.8                                          | 0.9  | 0.8  | 0.5  | 0.1        | 0.1  | <b>0.0</b> | 0.0  | 0.0         | 6e-4  | 2e-5                     | 24.09 |                  |

**Table 4: Effectiveness of Recurrent Connections in Non-sinusoidal and Sinusoidal Architectures.** PSNR (dB) $\uparrow$  is reported in the capacity-limited regime.

| Non-sinusoidal  | Rec. | ~100K        | ~200K        | ~400K        | ~600K        | Sinusoidal      | Rec. | ~100K        | ~200K        | ~400K        | ~600K        |
|-----------------|------|--------------|--------------|--------------|--------------|-----------------|------|--------------|--------------|--------------|--------------|
| PEMLP (1e-3)    | ✗    | <b>28.89</b> | <b>31.04</b> | 31.80        | <b>34.20</b> | SIREN (1e-3)    | ✗    | 33.60        | 39.57        | 46.23        | 48.64        |
| + Rec. (1e-3)   | ✓    | 27.48        | 29.64        | <b>33.82</b> | 31.04        | + Rec. (1.5e-4) | ✓    | <b>37.26</b> | <b>46.26</b> | <b>61.36</b> | <b>78.35</b> |
| Gauss (1.5e-4)  | ✗    | <b>20.48</b> | <b>26.49</b> | <b>35.33</b> | <b>60.31</b> | FINER (5e-4)    | ✗    | <b>38.01</b> | 40.59        | 45.56        | 48.36        |
| + Rec. (1.5e-4) | ✓    | 11.74        | 11.77        | 11.92        | 12.02        | + Rec. (1.5e-4) | ✓    | 34.21        | <b>44.45</b> | <b>68.06</b> | <b>85.62</b> |

normalization in dB,

$$M_\ell^{\text{dB}}[u, v] = 20 \log_{10} \frac{(1/C) \sum_c |\mathbf{F}_c^{(\ell)}[u, v]|}{\max_{u', v'}(\cdot)}, \quad (14)$$

we summarize each refinement step using spectral support  $S_\ell^\tau$  (which measures spectral breadth), and DC-normalized upper-band spectral content HF $_\ell$  (which measures the average spectral magnitude over the upper radial band  $\mathcal{U}=\{H/4, \dots, H/2-1\}$  relative to the DC component):

$$S_\ell^\tau \triangleq H^{-2} |\{(u, v) : M_\ell^{\text{dB}} > \tau\}|, \quad \text{HF}_\ell \triangleq \frac{1}{|\mathcal{U}|} \sum_{k \in \mathcal{U}} \frac{R_\ell(k)}{R_\ell(0)}, \quad (15)$$

with  $R_\ell(k)$  denoting the radial profile of the channel-averaged linear magnitude spectrum at radius  $k$  from the zero-frequency (DC) component.

At random initialization, repeated sinusoidal transformations rapidly expand the measured spectral support and upper-band content across refinement steps (Tab. 3(a)). After training, its trajectory reflects task-dependent reweighting. In contrast, the equilibrium-style iSIREN baseline [13] exhibits a stationary spectral profile, while non-sinusoidal recurrent controls lose support (Tab. 3(b,c)). This distinction is mirrored in fitting: under matched capacity, recurrence consistently improves SIREN [31] and benefits FINER [18] beyond ~200K parameters, but degrades Gaussian models and yields no persistent gain for PEMLP (Tab. 4).

#### 4.3 On the Use of Bias Terms in Recurrent Layer

To examine the role of bias in weight-tied sinusoidal recurrence, we first consider the general recurrent update with an additive bias term. Let

$$\mathbf{h}^{(r+1)} = \sin\left(\omega(\mathbf{W}_{\text{rec}}\mathbf{h}^{(r)} + \mathbf{b}_{\text{rec}})\right), \quad \mathbf{z}^{(r)} \triangleq \omega\mathbf{W}_{\text{rec}}\mathbf{h}^{(r)}, \quad \beta \triangleq \omega\mathbf{b}_{\text{rec}}. \quad (16)$$

By applying the sine addition formula element-wise across channels, the recurrent update can be written as

$$\mathbf{h}^{(r+1)} = \sin(\mathbf{z}^{(r)} + \beta) = \sin(\mathbf{z}^{(r)}) \odot \cos(\beta) + \cos(\mathbf{z}^{(r)}) \odot \sin(\beta). \quad (17)$$

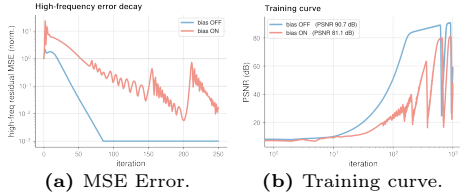
Here,  $\mathbf{b}_{\text{rec}}$  induces the same phase shift whenever the tied nonlinear map is reused. This shift changes the phase-dependent derivative gate through which variations in the current hidden state are propagated to the next step,

$$\frac{\partial \mathbf{h}^{(r+1)}}{\partial \mathbf{h}^{(r)}} = \text{diag}[\cos(\mathbf{z}^{(r)} + \beta)] \omega \mathbf{W}_{\text{rec}}. \quad (18)$$

That is,  $\beta$  shifts each channel to a different operating point on the sinusoidal response, replacing the bias-free local gate  $\cos(\mathbf{z}^{(r)})$  with  $\cos(\mathbf{z}^{(r)} + \beta)$ . Because this modulation is reapplied at every unrolled step, its effect can accumulate along the recurrent trajectory. Empirically, models with and without  $(\mathbf{b}_{\text{rec}})$  behave similarly for a single recurrent application ( $R = 1$ ), whereas enabling the recurrent bias leads to a substantial degradation in fitting once the same map is reused across multiple steps ( $R \geq 2$ ; Table 5). We further observe less stable high-frequency residual dynamics when recurrent bias is enabled (Fig. 2).

**Table 5:** Mean peak PSNR (dB) over 1K iterations on Kodak24. We use  $\omega_0 = 256/45$  (first/hidden layers). OFF and ON denote recurrent bias disabled and enabled, respectively. Both settings use 792K parameters.

| #Rec. $\rightarrow$ | 1            | 2            | 3            | 4            | 5            |
|---------------------|--------------|--------------|--------------|--------------|--------------|
| Bias ON             | 51.16        | 60.35        | 51.97        | 63.57        | 78.10        |
| Bias OFF            | <b>51.41</b> | <b>83.49</b> | <b>87.54</b> | <b>89.59</b> | <b>90.28</b> |
| $\Delta\text{PSNR}$ | +0.25        | +23.14       | +35.57       | +26.02       | +12.18       |



**Fig. 2:** Effectiveness of  $\mathbf{b}_{\text{rec}}$  for training trajectory.

## 5 Method

As discussed in Sec. 4, repeated sinusoidal transformations progressively enrich the reachable spectrum through structured harmonic interactions. This observation motivates a recurrent, weight-tied design: instead of increasing the number of independently parameterized layers, we repeatedly apply a shared sinusoidal block to iteratively refine the latent representation under a fixed parameter budget. The overall pipeline is illustrated in Fig. 3.

Given a coordinate  $\mathbf{x}$ , we initialize a latent state by a sinusoidal input projection and refine it for  $R$  recurrent steps using a shared transformation:

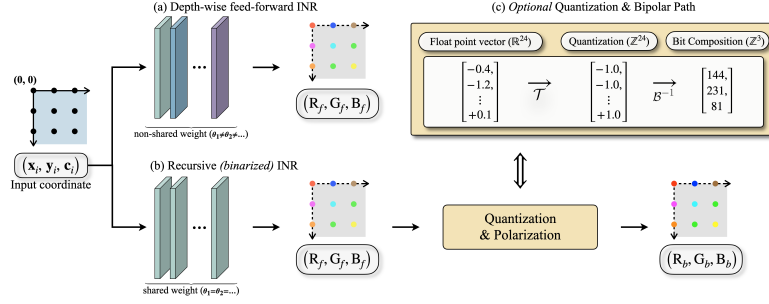
$$\mathbf{h}^{(0)} = \sigma(W^{(\text{in})}(\mathbf{x})), \quad \mathbf{h}^{(r)} = \sigma(W^{(\text{rec})}(\mathbf{h}^{(r-1)})), \quad r = 1, \dots, R, \quad (19)$$

followed by the output projection

$$\hat{\mathbf{c}} = W^{(\text{out})}(\mathbf{h}^{(R)}). \quad (20)$$

Here,  $W^{(\text{in})}$ ,  $W^{(\text{rec})}$ , and  $W^{(\text{out})}$  are bias-free linear maps, with  $W^{(\text{rec})}$  shared across all recurrent steps, and  $\sigma(\cdot)$  denotes the elementwise sinusoidal activation. Equivalently, the model can be written in compact compositional form as

$$f_{\psi}(\mathbf{x}) = W^{(\text{out})} \circ \underbrace{(\sigma \circ W^{(\text{rec})}) \circ \dots \circ (\sigma \circ W^{(\text{rec})})}_{R \text{ recurrent steps}} \circ \sigma \circ W^{(\text{in})}(\mathbf{x}). \quad (21)$$



**Fig. 3: Feed-forward vs. recurrent INRs under a fixed parameter budget.** (a) A depthwise feed-forward INR baseline uses non-shared weights across layers to map coordinates  $(x_i, y_i)$  to RGB outputs  $(R_f, G_f, B_f)$  in FP32. (b) recurrent INR reuses a single shared module and iteratively refines hidden features for  $T$  steps, increasing effective depth via unrolling and training with backpropagation through time. Optionally, we apply a threshold/sign operator  $\mathcal{T}$  to obtain bipolar outputs and an inverse bipolar mapping  $\mathcal{B}^{-1}$  to convert bipolar values to UINT8 for exact discrete recovery.

This recurrence is the architectural counterpart of the spectral analysis in Sec. 4: each application of the shared sinusoidal block expands harmonic interactions while preserving the parameter count.

### 5.1 Binarized supervision for exact discrete reconstruction

For lossless representation, we seek to reproduce the quantized values observed at sampled coordinates exactly, rather than merely approximate them through real-valued regression. This motivates supervising the INR in a binarized code space instead of directly regressing the corresponding intensity or amplitude values. In this sense, our formulation is related to recent INR-based approaches that cast reconstruction in a digital representation space [12].

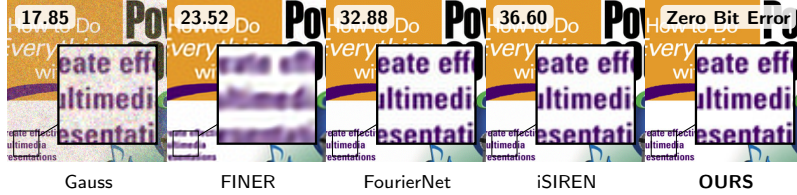
Let  $y_p \in \{0, \dots, 2^B - 1\}$  be the quantized target value at coordinate  $p$ , and let  $\Gamma(\cdot)$  be a  $B$ -bit encoder. We define the supervision target as a bipolar code

$$\mathbf{c}_p = 2\Gamma(y_p) - 1 \in \{-1, +1\}^B. \quad (22)$$

We use a bipolar code rather than a standard  $\{0, 1\}$  code for two reasons. First, bipolar targets are zero-centered, which is better matched to the symmetric range of sinusoidal activations. Second, all bipolar codewords have identical norm, i.e.,  $\|\mathbf{c}_p\|_2^2 = B$ , which yields a particularly simple alignment-based training objective. Specifically, the network predicts a real-valued code vector  $\hat{\mathbf{c}}_p = f_\psi(\mathbf{x}_p) \in \mathbb{R}^B$ , and we optimize it using cosine similarity:

$$\max_{\psi} \mathcal{L}_{\text{align}}(\psi) \quad \text{with} \quad \mathcal{L}_{\text{align}} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \frac{\hat{\mathbf{c}}_p^\top \mathbf{c}_p}{\|\hat{\mathbf{c}}_p\|_2 \|\mathbf{c}_p\|_2}, \quad \hat{\mathbf{c}}_p = f_\psi(\mathbf{x}_p). \quad (23)$$

where  $\mathcal{P}$  denotes the set of sampled coordinates. This choice is also well motivated from the squared-error perspective. Expanding the Euclidean distance



**Fig. 4: Qualitative comparison after 125 optimization iterations** on an early-converging example that reaches zero bit error at the 125-iteration checkpoint. The boxes show PSNR values, and inset crops visualize fine-detail recovery. Compared to baselines, our method reconstructs sharper edges and text-like structures with fewer artifacts, consistent with the quantitative gains in Table 6.

between prediction and target gives

$$\|\hat{\mathbf{c}}_p - \mathbf{c}_p\|_2^2 = \|\hat{\mathbf{c}}_p\|_2^2 + \|\mathbf{c}_p\|_2^2 - 2\hat{\mathbf{c}}_p^\top \mathbf{c}_p. \quad (24)$$

Since  $\mathbf{c}_p \in \{-1, +1\}^B$ , we have  $\|\mathbf{c}_p\|_2^2 = B$ , which is constant across all targets. Therefore, the target-dependent term is entirely determined by the alignment between  $\hat{\mathbf{c}}_p$  and  $\mathbf{c}_p$ . Cosine similarity makes this alignment explicit while additionally removing sensitivity to the scale of  $\hat{\mathbf{c}}_p$ .

*Gray-binarization.* In our implementation,  $\Gamma(\cdot)$  is chosen as a Gray encoder [11]. Compared with standard binary coding, Gray coding preserves local adjacency in the quantized signal space, since neighboring quantization levels differ by only one bit. We keep this design choice simple in the method description and return to its implications in the Sec. 7.

## 6 Experiments

We first evaluate quantized 2D image fitting, where exactness is directly measurable under a fixed parameter budget. We then assess transferability by integrating the same decoder into super-resolution and 3D NeRF pipelines.

### 6.1 Image representation

**Datasets and Evaluation Protocol.** We evaluate on four widely used image datasets: Set5 [5], Kodak24 [10], DIV2K [2], and FFHQ [14]. For DIV2K and FFHQ, we randomly sample 100 and 600 images, respectively, and resize all images to  $256 \times 256$ . Unless otherwise noted, we use five recurrent steps. For fair comparison, we preserve the original macro-architecture of the baselines and adjust only the hidden width to obtain comparable parameter counts.

**Implementation.** The decoder output is bounded by  $\mathcal{G}(z) = \sin(\arctan z)$ , which can be written as  $z/\sqrt{1+z^2}$ . This function is approximately linear around the origin while smoothly saturating to  $[-1, 1]$ , making it well matched to the bipolar target space and empirically stabilizing optimization. To keep training fully differentiable, the transformations  $\mathcal{T}$  and  $\mathcal{B}^{-1}$  are applied only at inference time. We train all models with AdamW using a learning rate of  $1.5 \times 10^{-4}$ , without a scheduler, on NVIDIA RTX 3090 GPUs.

**Table 6:** We report the number of optimization iterations, reconstruction fidelity, and bit error, along with parameter count. Baselines are trained for 1000 iterations with 791K parameters. Our model uses fewer parameters (609K) and achieves substantially higher fidelity with a fixed budget of 100 iterations (OURS (#iter=100)), and can reach exact quantized reconstruction with an adaptive number of iterations.

| Method                                               | #Param. | #Iter  | Set5                 |           | Kodak24              |           | DIV2K-100            |           | FFHQ-600             |           |
|------------------------------------------------------|---------|--------|----------------------|-----------|----------------------|-----------|----------------------|-----------|----------------------|-----------|
|                                                      |         |        | PSNR / SSIM ↑        | Bit err ↓ | PSNR / SSIM ↑        | Bit err ↓ | PSNR / SSIM ↑        | Bit err ↓ | PSNR / SSIM ↑        | Bit err ↓ |
| Standard INR Baselines                               |         |        |                      |           |                      |           |                      |           |                      |           |
| ◦ WIRE                                               | 791K    | 1000   | 26.91 / .6637        | 544.52K   | 26.83 / .6583        | 566.38K   | 26.65 / .7620        | 555.53K   | 26.43 / .5971        | 561.66K   |
| ◦ Gauss                                              |         |        | 33.97 / .8802        | 453.77K   | 34.18 / .8867        | 460.84K   | 34.06 / .9211        | 456.13K   | 34.01 / .8647        | 458.33K   |
| ◦ SIREN                                              |         |        | 38.54 / .9629        | 350.11K   | 35.01 / .9404        | 395.26K   | 32.81 / .9535        | 430.17K   | 37.68 / .9630        | 351.22K   |
| ◦ FINER                                              |         |        | 41.78 / .9775        | 309.20K   | 42.15 / .9803        | 305.60K   | 40.81 / .9876        | 325.74K   | 41.08 / .9796        | 312.68K   |
| Fixed-frequency, band-limited spectral INR Baselines |         |        |                      |           |                      |           |                      |           |                      |           |
| ◦ BACON                                              | 791K    | 1000   | 33.71 / .9488        | 390.47K   | 31.93 / .9194        | 429.31K   | 29.73 / .9177        | 462.32K   | 33.15 / .9414        | 390.39K   |
| Learnable-frequency spectral INR Baselines           |         |        |                      |           |                      |           |                      |           |                      |           |
| ◦ FourierNet                                         | 791K    | 1000   | 45.67 / .9924        | 242.83K   | 41.77 / .9882        | 293.41K   | 39.31 / .9883        | 334.63K   | 42.64 / .9898        | 266.08K   |
| ◦ GaborNet                                           |         |        | 50.31 / .9970        | 190.86K   | 51.38 / .9975        | 180.71K   | 49.51 / .9976        | 206.54K   | 49.54 / .9962        | 193.25K   |
| ◦ iSIREN                                             |         |        | 51.80 / .9976        | 172.55K   | 52.67 / .9981        | 167.44K   | 51.43 / .9983        | 183.92K   | 50.90 / .9971        | 173.83K   |
| Ours (Harmonic-Siren)                                |         |        |                      |           |                      |           |                      |           |                      |           |
| ◦ OURS                                               | 609K    | 100    | 58.16 / .9997        | 0.78K     | 53.20 / .9989        | 2.51K     | 46.28 / .9973        | 5.66K     | 61.39 / .9998        | 0.62K     |
|                                                      |         | ≤ 1000 | ∞ AT 322 ± 117 iters |           | ∞ AT 447 ± 327 iters |           | ∞ AT 537 ± 432 iters |           | ∞ AT 440 ± 350 iters |           |

**Image Fitting.** Table 6 shows that finite sinusoidal refinement accelerates high-fidelity fitting under reduced parameter and optimization budget. With 609K parameters and only 100 iterations, *ours* exceeds the 1,000-iteration results of all learnable-frequency spectral baselines [9, 13, 17] on various datasets. This advantage is obtained despite using fewer parameters than the baselines. Fig. 4 shows the same early-optimization trend: compared with learnable-frequency baselines, *ours* recovers sharper boundaries and finer structures with fewer residual artifacts, indicating that shared sinusoidal refinement improves the composition of learned harmonic components across recurrent steps.

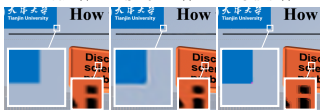
**Super-resolution.** We evaluate unseen-coordinate recovery in complementary image-specific and pretrained settings. For per-image  $\times 2$  fitting (Table 7a), *ours* preserves a sharper vertical boundary on screen content than SIREN [31] and bilinear interpolation, and attains the lowest Kodak24 LPIPS [34] with fewer optimization iterations on average. For pretrained LIIF-based arbitrary-scale SR [7] (Table 7b), we retain the EDSR-baseline encoder [16] and replace only the coordinate MLP under a matched  $\sim 50$ K-parameter decoder budget. Across Set5, Set14, and B100 at  $\times 2$ – $\times 4$ , our decoder consistently improves PSNR and SSIM, while perceptual gains are most pronounced at  $\times 2$ .

**Table 7: Super-resolution with recurrent INRs.** We evaluate our framework for (a) image-specific INR fitting and (b) pretrained LIIF-based arbitrary-scale SR. For (b), we retain the EDSR-baseline encoder and replace only LIIF’s coordinate MLP with our recurrent decoder under a matched  $\sim 50$ K-parameter decoder budget.

| Model   | #Params. | #Iter.        | PSNR $\uparrow$ | LPIPS $\downarrow$ |
|---------|----------|---------------|-----------------|--------------------|
| RFF-F10 | 630.4K   | 1000          | 25.37           | 0.4477             |
| SIREN   | 611.6K   | 1000          | <b>27.86</b>    | 0.3068             |
| Ours    | 609K     | 595 $\pm$ 285 | 26.55           | <b>0.2272</b>      |

| Dataset                                       | Scale      | LIIF MLP ( $\approx$ 50K) |                 |                    | Ours ( $\approx$ 50K) |                 |                    |
|-----------------------------------------------|------------|---------------------------|-----------------|--------------------|-----------------------|-----------------|--------------------|
|                                               |            | PSNR $\uparrow$           | SSIM $\uparrow$ | LPIPS $\downarrow$ | PSNR $\uparrow$       | SSIM $\uparrow$ | LPIPS $\downarrow$ |
| <i>Benchmark (EDSR-baseline encoder) [16]</i> |            |                           |                 |                    |                       |                 |                    |
| Set5                                          | $\times 2$ | 37.673                    | 0.9485          | 0.0625             | 37.721                | 0.9489          | 0.0574             |
|                                               | $\times 3$ | 34.036                    | 0.9104          | 0.1292             | 34.103                | 0.9111          | 0.1275             |
|                                               | $\times 4$ | 31.820                    | 0.8711          | 0.1797             | 31.850                | 0.8718          | 0.1867             |
| Set14                                         | $\times 2$ | 33.325                    | 0.9005          | 0.1022             | 33.401                | 0.9016          | 0.0941             |
|                                               | $\times 3$ | 30.116                    | 0.8243          | 0.2121             | 30.126                | 0.8249          | 0.2103             |
|                                               | $\times 4$ | 28.377                    | 0.7582          | 0.2912             | 28.389                | 0.7598          | 0.3010             |
| B100                                          | $\times 2$ | 31.968                    | 0.8979          | 0.1624             | 32.031                | 0.8996          | 0.1524             |
|                                               | $\times 3$ | 28.926                    | 0.8046          | 0.2966             | 28.976                | 0.8060          | 0.2969             |
|                                               | $\times 4$ | 27.427                    | 0.7294          | 0.3856             | 27.450                | 0.7308          | 0.3981             |





**Table 8: NVS quality on four scenes.** We compare NeRF and *Ours* under the same training budget; entries report PSNR, SSIM, and LPIPS for the rendered RGB views. Consistent gains across scenes indicate improved geometry-aware reconstruction.

| NeRF     | PSNR $\uparrow$ | SSIM $\uparrow$ | LPIPS $\downarrow$ | + Ours   | PSNR $\uparrow$ | SSIM $\uparrow$ | LPIPS $\downarrow$ |
|----------|-----------------|-----------------|--------------------|----------|-----------------|-----------------|--------------------|
| Flower   | 23.71           | 0.6316          | 0.4145             | Flower   | 24.54           | 0.7138          | 0.2934             |
| Orchids  | 17.85           | 0.4198          | 0.5172             | Orchids  | 18.79           | 0.5313          | 0.3797             |
| Fern     | 21.06           | 0.5640          | 0.4943             | Fern     | 21.60           | 0.6500          | 0.3704             |
| Fortress | 26.10           | 0.5987          | 0.4423             | Fortress | 26.36           | 0.7189          | 0.3206             |

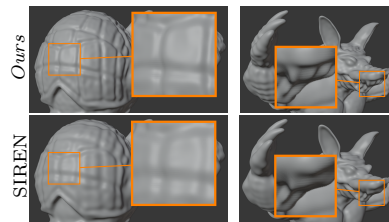


**Fig. 5: Qualitative novel view synthesis results (depth + rendering).** We compare NeRF and *Ours* on three scenes, showing predicted depth (grayscale) alongside the corresponding rendered RGB views. Under the same model budget (600K parameters) and training protocol, *Ours* yields cleaner, more coherent depth with sharper object boundaries, which translates to crisper geometry-aware renderings.

## 6.2 Extension to 3D Representation

**Neural Radiance Field.** We evaluate *Ours* in a neural radiance field setting on the forward-facing LLFF dataset [21]. We replace the standard NeRF RGB decoder with our recurrent sinusoidal decoder while keeping the rest of the rendering pipeline unchanged and maintaining a comparable parameter budget. As shown in Table 8, *Ours* consistently improves PSNR, SSIM [33], and LPIPS [34] over the baseline NeRF [22]. The lower LPIPS is consistent with better preservation of fine appearance details and local sharpness. This trend is also visually reflected in Fig. 5, where *Ours* produces visually sharper renderings and more coherent depth predictions, suggesting that the proposed decoder benefits both view synthesis quality and scene structure recovery.

**Signed Distance Function.** We evaluate *Ours* on signed distance function (SDF) reconstruction using the Stanford Armadillo dataset [15], following the setup of Sitzmann *et al.* [31]. We compare against SIREN under the same training budget of 2K epochs. Our SDF model uses a single hidden layer with 500 units and three recurrent steps. As shown in Fig. 6, *Ours* recovers finer surface structure and preserves sharper geometric details than SIREN, indicating that the proposed recurrent sinusoidal decoder transfers effectively to continuous 3D shape representation.



**Fig. 6: Ours on SDF.** Under the same training budget, *Ours* reconstructs finer surface structure than SIREN [31].



**Table 9: Effect of Gray coding on 2D image  $\times 2$  super-resolution.** Replacing natural binary coding (NBC) with Gray coding (GC) improves both reconstruction at the training resolution and generalization to unseen coordinates for the same architecture and parameter budget.

| Architecture | Coding | #Params. | Fit PSNR $\uparrow$ | SR PSNR $\times 2\uparrow$ |
|--------------|--------|----------|---------------------|----------------------------|
| RFF-F10      | NBC    | 630.4K   | 22.35               | 21.08                      |
|              | GC     |          | 26.30               | 23.98                      |
| <b>Ours</b>  | NBC    | 609K     | $\infty$            | 23.45                      |
|              | GC     |          | $\infty$            | <b>26.55</b>               |

**Table 10: Effect of Gray coding on NeRF rendering.** Results on the Flower scene from LLFF. Replacing natural binary coding (NBC) with Gray coding (GC) improves PSNR and SSIM and lowers LPIPS for both the standard NeRF decoder and **Ours**.

| Architecture  | Coding | #Params. | PSNR $\uparrow$ | SSIM $\uparrow$ | LPIPS $\downarrow$ |
|---------------|--------|----------|-----------------|-----------------|--------------------|
| Baseline NeRF | NBC    | 602.9K   | 20.79           | 0.5079          | 0.4465             |
|               | GC     |          | 23.05           | 0.6612          | 0.3437             |
| <b>Ours</b>   | NBC    | 600K     | 21.14           | 0.5263          | 0.4370             |
|               | GC     |          | <b>23.80</b>    | <b>0.7010</b>   | <b>0.3030</b>      |

## 7 Discussion and Limitations

### 7.1 Code-space continuity via Gray coding

Binary-domain supervision can represent quantized signals exactly, but its effectiveness depends critically on how quantization levels are mapped to codewords. Let  $\Gamma : \{0, \dots, 2^n - 1\} \rightarrow \{0, 1\}^n$  be an  $n$ -bit encoder, and let  $\tilde{\Gamma}(q) = 2\Gamma(q) - 1 \in \{-1, +1\}^n$  denote its bipolar form. If two codewords differ in  $h$  bits, then their bipolar cosine similarity is

$$\frac{\tilde{\Gamma}(q)^\top \tilde{\Gamma}(q')}{\|\tilde{\Gamma}(q)\|_2 \|\tilde{\Gamma}(q')\|_2} = 1 - \frac{2h}{n}. \quad (25)$$

Thus, in bipolar space, the Hamming distance directly determines how abruptly adjacent targets change under cosine supervision. To quantify this effect, we define the local code-space sensitivity for one quantization increment  $\delta$  as

$$L_\Gamma^{\text{local}} = \max_q \frac{\mathcal{H}(\Gamma(q), \Gamma(q+1))}{\delta}, \quad (26)$$

where  $\mathcal{H}$  denotes the Hamming distance and, for normalized intensities,  $\delta = 2^{-n}$  corresponds to one quantization step. Under natural binary coding (NBC), adjacent levels can differ in as many as  $n$  bits, yielding

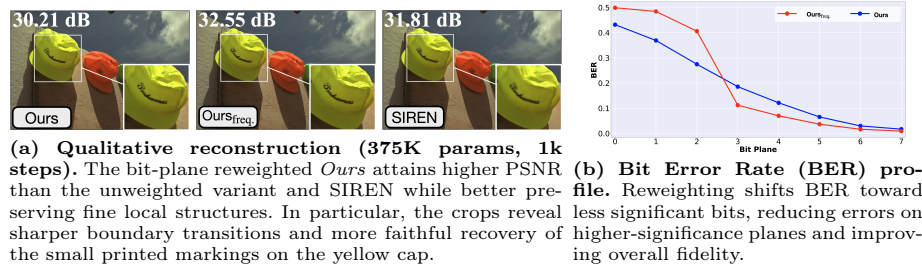
$$L_{\text{NBC}}^{\text{local}} = \frac{n}{\delta}. \quad (27)$$

In contrast, Gray coding [11] flips exactly one bit between adjacent levels by construction, so

$$L_{\text{GC}}^{\text{local}} = \frac{1}{\delta}. \quad (28)$$

Therefore, Gray coding reduces the worst-case local sensitivity by a factor of  $n$ .

We validate this effect in two settings. On Kodak image fitting and subsequent ( $\times 2$ ) super-resolution (Table 9), replacing NBC with Gray coding improves generalization to unseen coordinates for both a Fourier-feature baseline and Ours. On the Flower scene of LLFF NeRF rendering (Table 10), Gray coding consistently improves PSNR and SSIM while reducing LPIPS, with the largest gains observed for Ours.



**Fig. 7: Bit-plane reweighting improves fidelity under a fixed parameter budget.** (a) Reconstruction of a  $768 \times 512$  Kodak image using 375K trainable parameters and 1,000 optimization steps. The reweighted *Ours* achieves higher PSNR than both the unweighted model and SIREN, with sharper local details. (b) Bit-error rate (BER) across bit planes: reweighting prioritizes high-significance bits and shifts residual errors to less significant planes, improving image quality without increasing parameters.

## 7.2 Decoded-Distortion-Aware Bit-Plane Reweighting

The Gray-code analysis in Sec. 7.1 also distinguishes residual code errors by their effect after decoding. When the model or optimization budget is insufficient for exact reconstruction, residual errors need not be equally costly in the reconstructed signal. In particular, errors in lower-significance Gray planes induce bounded changes only in lower-order decoded binary digits, whereas errors in higher-significance planes can produce substantially larger intensity deviations. We therefore redistribute optimization emphasis toward the more consequential code planes in the non-exact regime. Concretely, we use a weighted loss

$$\mathcal{L}_w = \sum_{i=0}^{B-1} w_i \mathcal{L}^{(i)}, \quad (29)$$

where  $\mathcal{L}^{(i)}$  is the loss for the  $i$ -th code plane. In our implementation, with  $i = 0$  denoting the least significant plane, we set

$$w_i = 2^{-(8-i)}, \quad i \in \{0, 1, 2\}, \quad w_i = 1, \quad \text{otherwise}. \quad (30)$$

This reweighting effectively reallocates the available budget during optimization. As shown in Fig. 7, it improves reconstruction quality under the same parameter budget.

## 7.3 Iterative refinement and wall-clock efficiency

Because recurrent unrolling increases the computation performed at each optimization step, we evaluate the wall-clock efficiency of *Ours* on Kodak24. As shown in Table 11, *Ours* already reaches 42.84 dB in 6.52 seconds, surpassing the best baseline result obtained after 1,000 iterations. Within a comparable wall-clock budget, *Ours* continues to improve substantially, reaching 59.12 dB in 22.45 seconds. Exact reconstruction is achieved after 636 iterations, corresponding to 34.17 seconds. Although iterative refinement introduces additional computation per iteration, it yields a substantially better wall-clock fidelity trade-off, reaching higher reconstruction quality much earlier than feed-forward baselines.

**Table 11: Convergence speed vs. fidelity on Kodak-24 (wall-clock).** Baselines are reported at 1000 optimization iterations, while *Ours* is shown at intermediate checkpoints. PSNR=  $\infty$  indicates exact quantized reconstruction.

| Models  | Iter. | Time (s)↓     | PSNR (dB)↑   | Models | Iter. | Time (s)↓    | PSNR (dB)↑   |
|---------|-------|---------------|--------------|--------|-------|--------------|--------------|
| ○ Gauss | 1000  | 27.177        | 32.93        |        | 100   | <b>6.522</b> | <b>42.84</b> |
| ○ SIREN | 1000  | <b>21.687</b> | 33.09        | ○ Ours | 400   | 22.445       | 59.12        |
| ○ FINER | 1000  | 26.063        | <b>40.08</b> |        | 636   | 34.167       | $\infty$     |

#### 7.4 Disentangling Recurrent Refinement and Binarized Supervision

To isolate the architectural benefit of recurrence from the binarized supervision, Table 12 separates the contributions of recurrence and binarized supervision. Relative to the feed-forward baseline, recurrence yields the primary improvement in reconstruction fidelity. Binarized supervision then operates on this high-fidelity recurrent solution, eliminating the remaining mismatch.

**Table 12: Effect of recurrence and binarized supervision.** Compared with a feed-forward baseline, recurrent refinement yields a large fidelity gain. Binarized supervision removes the remaining mismatch and enables exact quantized reconstruction.

| KODAK24          | #Params.      | #Iteration  | PSNR↑        | SSIM↑         | #Bit Error↓   |
|------------------|---------------|-------------|--------------|---------------|---------------|
| FeedForward [18] | 791K          | 1000        | 42.15        | 0.9803        | 305.6K        |
| +Rec.            | <b>593.7K</b> | <b>1000</b> | <b>64.19</b> | <b>0.9998</b> | <b>49.70K</b> |
| +Rec.+Bin.       | <b>609K</b>   | 595 ± 285   | $\infty$     | <b>1.0</b>    | <b>0</b>      |

## 8 Conclusion, Limitations, and Implications

In this work, we offer a spectral perspective on how finite sinusoidal recurrence enriches the spectral support of intermediate INR features. A harmonic line-spectrum view links sinusoidal features to Fourier structure and provides a spectral interpretation of why recurrent unrolling can expand effective spectral support without increasing the number of parameters. Combined with bipolar code-space supervision, this design yields faster convergence and exact quantized reconstruction on 2D image benchmarks. The same decoder also transfers favorably to super-resolution, NeRF, and SDF tasks.

**Limitations.** While our method remains beneficial at moderate-to-large parameter budgets, its gains are not consistently observed across all model sizes, suggesting that its effectiveness may be limited under severe capacity constraints.

**Implications and Future Work.** Overall, these findings suggest that recurrence is a simple and modular alternative to increasing independently parameterized depth, while understanding and mitigating its model-size dependence remain directions for future work.

**Acknowledgments.** This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2024-00335741), and by the Culture, Sports and Tourism R&D Program through the Korea Creative Content Agency grant funded by the Ministry of Culture, Sports and Tourism (Project Name: International Collaborative Research and Global Talent Development for the Development of Copyright Management and Protection Technologies for Generative AI; Project Number: RS-2024-00345025), and by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2020-II201336, Artificial Intelligence Graduate School Program (UNIST)).

## References

1. Abramowitz, M., Stegun, I.A. (eds.): Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables. No. 55 in National Bureau of Standards Applied Mathematics Series, U.S. Government Printing Office, Washington, D.C. (1964)
2. Agustsson, E., Timofte, R.: NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (July 2017)
3. Bai, S., Kolter, J.Z., Koltun, V.: Deep equilibrium models. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019), [https://proceedings.neurips.cc/paper\\_files/paper/2019/file/01386bd6d8e091c2ab4c7c7de644d37b-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2019/file/01386bd6d8e091c2ab4c7c7de644d37b-Paper.pdf)
4. Benbarka, N., Höfer, T., Riaz, H.u.M., Zell, A.: Seeing implicit neural representations as fourier series. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 2041–2050 (January 2022)
5. Bevilacqua, M., Roumy, A., Guillemot, C., Alberi Morel, M.l.: Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In: Proceedings of the British Machine Vision Conference. pp. 135.1–135.10. BMVA Press (2012). <https://doi.org/10.5244/C.26.135>
6. Cao, Y., Fang, Z., Wu, Y., Zhou, D.X., Gu, Q.: Towards understanding the spectral bias of deep learning. In: Zhou, Z.H. (ed.) Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21. pp. 2205–2211. International Joint Conferences on Artificial Intelligence Organization (8 2021). <https://doi.org/10.24963/ijcai.2021/304>, main Track
7. Chen, Y., Liu, S., Wang, X.: Learning continuous image representation with local implicit image function. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8628–8638 (June 2021)
8. Elman, J.L.: Finding structure in time. Cognitive Science **14**(2), 179–211 (1990). [https://doi.org/10.1016/0364-0213\(90\)90002-E](https://doi.org/10.1016/0364-0213(90)90002-E)
9. Fathony, R., Sahu, A.K., Willmott, D., Kolter, J.Z.: Multiplicative filter networks. In: International Conference on Learning Representations (2021), <https://openreview.net/forum?id=OmtmcPkkhT>
10. Franzen, R.: Kodak lossless true color image suite. <https://r0k.us/graphics/kodak/> (1999), accessed: June 29, 2026
11. Gray, F.: Pulse code communication (1953), <https://patents.google.com/patent/US2632058A/en>
12. Han, W.K., Lee, B., Cho, H., Im, S., Jin, K.H.: Towards lossless implicit neural representation via bit plane decomposition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2269–2278 (June 2025)
13. Huang, Z., Bai, S., Kolter, J.Z.: (Implicit)<sup>2</sup>: Implicit layers for implicit representations. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems. vol. 34, pp. 9639–9650. Curran Associates, Inc. (2021), [https://proceedings.neurips.cc/paper\\_files/paper/2021/file/4ffbd5c8221d7c147f8363ccdc9a2a37-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/4ffbd5c8221d7c147f8363ccdc9a2a37-Paper.pdf)
14. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)

15. Krishnamurthy, V., Levoy, M.: Fitting smooth surfaces to dense polygon meshes. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. p. 313–324. SIGGRAPH '96, Association for Computing Machinery, New York, NY, USA (1996). <https://doi.org/10.1145/237170.237270>
16. Lim, B., Son, S., Kim, H., Nah, S., Lee, K.M.: Enhanced deep residual networks for single image super-resolution. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (July 2017)
17. Lindell, D.B., Van Veen, D., Park, J.J., Wetzstein, G.: BACON: Band-limited coordinate networks for multiscale scene representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16252–16262 (June 2022)
18. Liu, Z., Zhu, H., Zhang, Q., Fu, J., Deng, W., Ma, Z., Guo, Y., Cao, X.: 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 (CVPR). pp. 2713–2722 (June 2024)
19. McGinnis, J., Hölzl, F.A., Shit, S., Bieder, F., Friedrich, P., Mühlau, M., Menze, B., Rueckert, D., Wiestler, B.: Optimizing rank for high-fidelity implicit neural representations (2025), <https://arxiv.org/abs/2512.14366>
20. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
21. Mildenhall, B., Srinivasan, P.P., Ortiz-Cayon, R., Kalantari, N.K., Ramamoorthi, R., Ng, R., Kar, A.: Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (ToG)* **38**(4), 1–14 (2019)
22. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In: Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I. p. 405–421. Springer-Verlag, Berlin, Heidelberg (2020). [https://doi.org/10.1007/978-3-030-58452-8\\_24](https://doi.org/10.1007/978-3-030-58452-8_24)
23. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.* **41**(4) (Jul 2022). <https://doi.org/10.1145/3528223.3530127>
24. Novello, T., Aldana, D., Araujo, A., Velho, L.: Tuning the frequencies: Robust training for sinusoidal neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3071–3080 (June 2025)
25. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: DeepSDF: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
26. Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., Courville, A.: On the spectral bias of neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 5301–5310. PMLR (09–15 Jun 2019), <https://proceedings.mlr.press/v97/rahaman19a.html>
27. Ramasinghe, S., Lucey, S.: Beyond periodicity: Towards a unifying framework for activations in coordinate-mlps. In: Avidan, S., Brostow, G., Cissé, M., Farinella,

- G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. pp. 142–158. Springer Nature Switzerland, Cham (2022)
28. Roddenberry, T.M., Saragadam, V., de Hoop, M.V., Baraniuk, R.: Implicit neural representations and the algebra of complex wavelets. In: The Twelfth International Conference on Learning Representations (2024), <https://openreview.net/forum?id=uZfjFyPAvn>
  29. Saragadam, V., LeJeune, D., Tan, J., Balakrishnan, G., Veeraraghavan, A., Baraniuk, R.G.: WIRE: Wavelet implicit neural representations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18507–18516 (June 2023)
  30. Schwarzschild, A., Gupta, A., Ghiasi, A., Goldblum, M., Goldstein, T.: The uncanny similarity of recurrence and depth. In: International Conference on Learning Representations (2022), <https://openreview.net/forum?id=3wNcr5nq56>
  31. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 7462–7473. Curran Associates, Inc. (2020), [https://proceedings.neurips.cc/paper\\_files/paper/2020/file/53c04118df112c13a8c34b38343b9c10-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/53c04118df112c13a8c34b38343b9c10-Paper.pdf)
  32. Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., Ng, R.: Fourier features let networks learn high frequency functions in low dimensional domains. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 7537–7547. Curran Associates, Inc. (2020), [https://proceedings.neurips.cc/paper\\_files/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf)
  33. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing **13**(4), 600–612 (2004). <https://doi.org/10.1109/TIP.2003.819861>
  34. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)