EVOS: Efficient Implicit Neural Training via EVOlutionary Selector

Supplementary Material

A. 1D Audio Fitting Task

Background & Settings. Fitting 1D audio data can be formulated as $F_\theta(\mathbf{x}) : (t) \mapsto (a)$, where a represents the amplitude value at time step t. For this task, we utilized the *test.clean* split from LibriSpeech [35] dataset, with each audio sample truncated to the initial 5 seconds at a 16,000 Hz sampling rate. Following Siamese SIREN [23], we configured both ω and ω_0 to 100 in the SIREN architecture. The quality of reconstructed audio was evaluated using SI-SNR, STOI [53], PESQ [44], and Mean Square Error (MSE) metrics. Due to task incompatibility issues exhibited by EGRA [15], Expan. [63], and Soft Mining [21], we restricted our experiments to standard training, uniform sampling, INT [61], and its variants.

Results. As shown in Table 6, our method simultaneously achieves reduced training time and enhanced reconstruction quality per iteration compared to INT and its variants, consistently outperforming standard training across all metrics. Fig. 7 illustrates the reconstruction error under a fixed 30-second training duration, comparing standard training (red) with our method (purple). Our approach demonstrates notably lower error rates than standard training. The Mel spectrogram visualization in Fig. 6 further validates the effectiveness of our method.

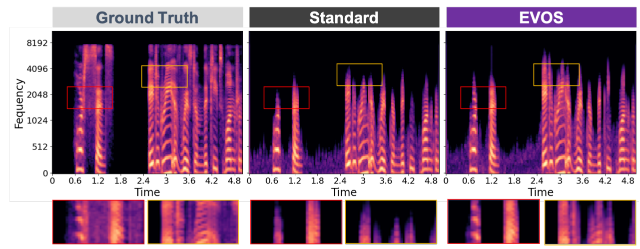

Figure 6. Visual comparison of Mel spectrogram reconstructions with 30-second training duration. Due to spectral bias, INRs exhibit lower expressiveness in high-frequency regions compared to low-frequency regions. EVOS integration can alleviate this limitation under fixed time constraints.

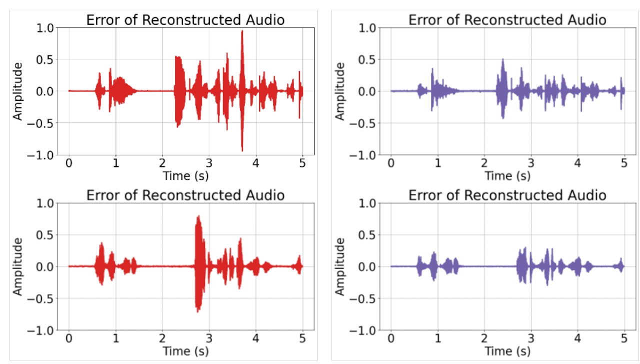

Figure 7. Visual comparison of reconstruction error for 1D audio fitting with 30-second training duration. **Red** and **Purple** lines represent standard training and our method, respectively.

Strategies	SI-SNR ↑	STOI ↑	PESQ ↓	MSE ↓ (e-4)	Time ↓ (sec)
Standard	11.21	0.896	1.387	2.835	47.62
Uniform.	9.88	0.855	1.274	4.938	28.47
INT [61]	11.62	0.904	1.307	2.410	31.36
INT [61]*	12.14	0.908	1.409	2.067	45.80
<u>INT</u> [61]†	11.79	0.905	1.394	2.233	33.42
EVOS	12.35	0.910	1.411	2.014	29.44
<u>EVOS</u>	12.95	0.921	1.449	1.660	31.63

$\star$ denotes INT (dense.)

† denotes the best-performing variant reported in INT [61].

Table 6. Comparison of sampling strategies on 1D audio fitting. Forest : the best performance.

B. 2D Text Fitting Task

Background & Settings. Fitting 2D synthesized text image data can be formulated as $F_\theta(\mathbf{x}) : (x, y) \mapsto (r, g, b)$. The dataset in this experiment was obtained from [54]. Such synthesized text data differs from natural images in its inherently imbalanced distribution and limited pixel intensity variety. For example, a synthesized text image containing three words might only have four intensity values (three text colors and one background color), unlike natural images with abundant color variations. Given the task's relative simplicity, we set the total iterations $T = 1000$ and used SIREN as the backbone, while maintaining other settings consistent with Sec. 4.2.

Results. Table 7 demonstrates that our method achieves the most significant efficiency improvements compared to existing sampling methods [15, 21, 61, 63] on 2D text fitting tasks. Table 8 verifies our method's compatibility across different architectures [28, 41, 50, 54]. Furthermore, Fig. 8 presents reconstructed texts after 20 seconds of training using various acceleration methods. Since INT† shows degraded performance in this task, we used INT (dense) for comparison. The highlighted regions (red boxes) demonstrate our method's superior reconstruction quality.

C. 2D Image Fitting Task

Settings. We evaluated EVOS on the widely used Kodak dataset [14], which comprises 24 natural images at 768 ×

Strategies	PSNR ↑	SSIM ↑	LPIPS ↓	MSE ↓ (e-3)	Time ↓ (sec)
Standard	35.15	0.986	0.022	1.527	35.89
Uniform.	33.73	0.983	0.031	2.118	20.68
EGRA [15]	34.07	0.983	0.029	1.979	21.01
Expan. [63]	36.92	0.984	0.017	1.043	20.33
Soft. [21]	37.01	0.981	0.144	1.053	22.18
INT [61]⋆	36.76	0.989	0.155	1.088	31.58
INT [61]†	35.59	0.987	0.020	1.389	24.75
EVOS	37.42	0.985	0.016	1.002	24.15

⋆ denotes INT (dense.)

† denotes the best-performing variant reported in INT [61].

Table 7. Comparison of sampling strategies on 2D synthesized text fitting. Forest : the best performance.

Strategies	PSNR ↑	SSIM ↑	LPIPS ↓	MSE ↓ (e-3)	Time ↓ (sec)
PEMLP [54]	34.64	0.977	0.036	1.944	30.67
+EVOS	37.80	0.983	0.025	1.138	18.83
SIREN [50]	35.15	0.986	0.022	1.527	35.89
+EVOS	37.42	0.985	0.016	1.002	24.15
GAUSS [41]	36.97	0.986	0.013	1.946	50.44
+EVOS	39.28	0.992	0.005	1.066	31.09
FINER [28]	41.56	0.993	0.005	0.420	46.20
+EVOS	43.82	0.990	0.008	0.349	28.01

Table 8. Quantitative comparison across different backbones on 2D text fitting task. Mint : enhance both in efficiency & quality.

512 resolution, distinct from the DIV2K dataset [2] used in Sec. 4. All experimental settings, including hyperparameters, backbones, and network architectures, strictly followed those in Sec. 4.2. We compared reconstruction quality under fixed iterations to demonstrate our method's advantages.

Results. As shown in Table 10, our method achieves state-of-the-art efficiency compared to other sampling-based acceleration methods. Specifically, under fixed iterations, our approach reduces training time by 46.79% while improving PSNR by 0.31 dB compared to standard training with constant scheduler, and achieves 29.79% time reduction with 0.91 dB PSNR gains using step-wise scheduler. Notably, training with only 50% of the data not only reduces computational cost but also improves per-iteration performance, further supporting our findings in Sec. 4.2.

D. 3D Shape Fitting Task

Backgound & Settings. We used Signed Distance Fields (SDF) to represent 3D shapes, a widely adopted approach in computer graphics [20]. The fitting task can be formulated as $F_\theta(\mathbf{x}) : (x, y, z) \mapsto (s)$, where (x, y, z) represents the coordinate of given points and s denotes the signed distance to the surface. Following INT [61], we employed an 8×256 MLP with SIREN architecture. We evaluated our method on the Asian Dragon scene from the Stanford 3D Scanning Repository [1]. The total iterations were set to $T = 10,000$, with other settings remaining consistent with Sec. 4.2. Following [25], we sampled points from the surface using coarse (Laplacian noise with variance 0.1) and fine (Laplacian noise with variance 0.001) sampling procedures, randomly selecting 50,000 points per iteration. For EVOS, due to the absence of suitable high-frequency extractors for SDF, we temporarily disabled the crossover component. Given the varying degrees of task incompatibility exhibited by EGRA [15], Expan. [63], and Soft Mining [21], we confined our experiment to standard

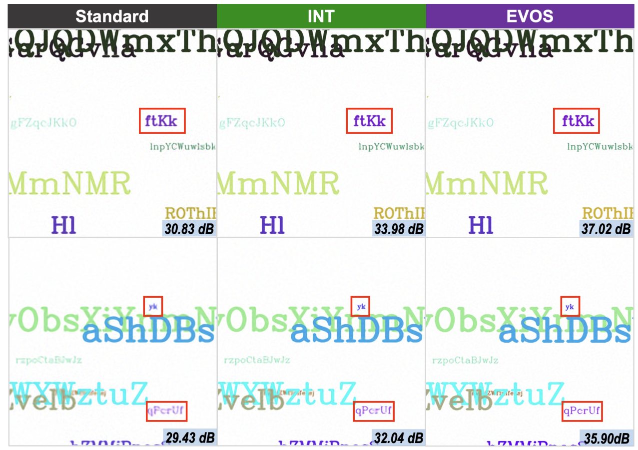

Figure 8. Visual comparison for 2D text fitting task. We employ INT (dense.) rather than INT† due to the latter's performance degradation in this task. All experiments are conducted under consistent conditions with a fixed training duration of 20 seconds.

Settings	5k IoU ↑	5k CHD ↓	10k IoU ↑	10k CHD ↓	Time (sec) ↓
Standard	0.949	1.45e-6	0.967	6.65e-7	171.26
Uniform.	0.918	3.66e-4	0.965	1.13e-3	96.07
INT [61]	0.946	2.94e-6	0.962	1.96e-6	106.75
INT [61]⋆	0.938	1.65e-5	0.956	1.52e-5	147.09
INT [61])†	0.951	1.59e-6	0.965	1.01e-6	115.28
EVOS	0.955	1.42e-6	0.965	8.92e-7	98.31
EVOS	0.955	1.44e-6	0.967	8.18e-7	106.02

⋆ denotes INT (dense.)

† denotes the best-performing variant reported in INT [61].

Table 9. Comparison of sampling strategies on 3D shape fitting. CHD: CHamfer Distance. Forest : the best performance.

Strategies	1k Iterations PSNR ↑	1k Iterations SSIM ↑	1k Iterations LPIPS↓	2k Iterations PSNR↑	2k Iterations SSIM↑	2k Iterations LPIPS↓	5k Iterations PSNR↑	5k Iterations SSIM↑	5k Iterations LPIPS↓	Time↓ (sec)
Standard	30.54	0.848	0.235	33.47	0.906	0.131	36.10	0.938	0.074	266.55
Uniform.	29.91	0.832	0.266	32.55	0.891	0.161	35.16	0.927	0.096	137.53
EGRA [15]	29.92	0.831	0.267	32.63	0.891	0.159	35.21	0.927	0.095	141.62
Expan. [63]	30.47	0.844	0.237	33.015	0.897	0.139	35.30	0.927	0.084	138.23
INT [61] (dense.)	30.92	0.853	0.229	33.61	0.906	0.129	36.08	0.936	0.074	230.76
INT [61] (incre.)	30.92	0.853	0.229	31.43	0.853	0.211	34.65	0.904	0.109	198.88
EVOS (proposed)	31.68	0.862	0.209	34.81	0.915	0.109	36.41	0.934	0.073	141.85
Uniform.	29.28	0.814	0.301	32.25	0.885	0.175	35.941	0.935	0.080	153.08
EGRA [15]	29.27	0.812	0.302	32.29	0.885	0.175	35.96	0.935	0.081	166.54
Expan. [63]	30.46	0.839	0.241	33.11	0.895	0.148	36.12	0.935	0.081	153.02
INT [61] (dense)	31.07	0.845	0.250	33.58	0.901	0.140	36.00	0.935	0.078	251.08
INT [61] (incre.)†	31.07	0.845	0.249	30.33	0.821	0.265	36.22	0.938	0.076	177.05
EVOS (proposed)	31.39	0.843	0.231	34.708	0.907	0.118	37.01	0.943	0.066	187.15

† denotes the best-performing variant reported in INT [61].

Table 10. Comparison of sampling strategies on Kodak datasets. Strategies without underlines employ constant scheduler ($\beta = 0.5$), while underlined strategies implement step-wise scheduler. Forest : the best performance; Mint : exceeds standard training.

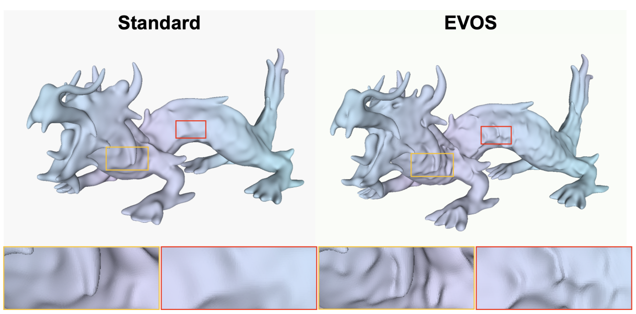

Figure 9. Visual comparison of 3D shape fitting with fixed 90-second training.

training, uniform sampling, INT, and its variants.

Results. Table 9 demonstrates that EVOS achieves significant efficiency improvements compared to standard training, INT, and its variants. Our method maintains comparable reconstructed quality while reducing training time by 38.10%. At 5,000 iterations, we achieve improvement in IoU while maintaining acceleration benefits. Visual comparisons in Fig. 9 show reconstructed shapes after 90 seconds of training, demonstrating that EVOS significantly enhances reconstructed details.

E. Implementation of Soft Mining

Background. Soft Mining is an acceleration method for Neural Radiance Field (NeRF) that utilizes Langevin Monte-Carlo sampling to form training batches during optimization. The sampling process can be formulated as:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + a\nabla \log Q\left(\mathbf{x}_t\right) + b\boldsymbol{\eta}_{t+1}, \tag{9}$$

where $\mathbf{x}_t$ represents sampled data at step t, a and b are hyperparameters, $Q(\mathbf{x})$ denotes the L1 norm of the error, and $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$ is Gaussian noise. To mitigate training bias introduced by importance sampling, they propose soft mining to regulate the loss function:

$$\mathcal{L} = \frac{1}{N}\sum_{n=1}^{N}\left[\frac{\operatorname{err}\left(\mathbf{x}_n\right)}{\operatorname{sg}\left(Q\left(\mathbf{x}_n\right)\right)^{\alpha}}\right], \quad \text{where } \alpha \in [0,1], \tag{10}$$

where α controls mining softness ($\alpha = 0$ for pure hard mining, $\alpha = 1$ for pure importance mining), $\operatorname{err}(\mathbf{x})$ is the L2 norm of the error, $\operatorname{sg}(\cdot)$ is the stop gradient operator, and N denotes the batch size. Further details are provided in the original paper [21].

Parameter Tuning & Ablation Study. The official implementation of Soft Mining showed significant performance degradation when applied to natural image fitting (Sec. 4.2), likely due to fundamental differences between image fitting and radiance field synthesis tasks. To ensure fair comparison, we conducted comprehensive parameter tuning and ablation studies. Our investigation focused on three key components: the softness parameter α, warmup iteration count, and fuzzy indexing mechanism. Parameters a and b exhibited minimal influence on performance; hence, we retained their default values. Results in Table 11 demonstrate that fuzzy indexing significantly degraded performance. Based on empirical analysis, we determined optimal settings by

Settings	1k Iterations PSNR ↑	SSIM ↑	LPIPS↓	2k Iterations PSNR↑	SSIM↑	LPIPS↓	5k Iterations PSNR↑	SSIM↑	LPIPS↓
Original	25.98	0.815	0.141	25.98	0.824	0.095	25.97	0.830	0.071
w/o Fuzzy Indexing	31.48	0.903	0.124	33.52	0.930	0.072	35.29	0.948	0.044
Hard ($\alpha = 0$)	30.42	0.877	0.159	33.07	0.920	0.071	35.25	0.944	0.046
$\alpha = 0.1$	30.67	0.884	0.148	33.21	0.923	0.069	35.27	0.946	0.045
$\alpha = 0.3$	31.15	0.896	0.131	33.45	0.928	0.067	35.27	0.948	0.048
$\alpha = 0.5^{\star}$	31.49	0.903	0.124	33.52	0.931	0.072	35.31	0.948	0.045
$\alpha = 0.7$	31.47	0.903	0.125	33.44	0.930	0.074	35.18	0.947	0.047
$\alpha = 0.9$	31.17	0.899	0.131	33.18	0.928	0.079	34.79	0.944	0.052
Important ($\alpha = 1$)	30.78	0.893	0.140	32.97	0.927	0.082	34.52	0.943	0.055
w/o warmup	30.98	0.895	0.140	33.21	0.928	0.076	35.18	0.947	0.045
warmup for 0.1k	31.34	0.901	0.131	33.32	0.928	0.076	35.15	0.946	0.047
warmup for 0.5k	31.51	0.903	0.124	33.43	0.929	0.073	35.22	0.947	0.046

$\star$ The final α in our implementation.

Table 11. Results of empirical parameter tuning for Soft Mining. All experiments except *Original* are conducted with the basic improvement of *w/o Fuzzy Indexing*. Other experimental settings follow Sec.4.2.

Strategies	1k Iterations PSNR ↑	SSIM ↑	LPIPS↓	2k Iterations PSNR↑	SSIM↑	LPIPS↓	5k Iterations PSNR↑	SSIM↑	LPIPS↓	Time↓ (min)
Uniform.	27.14	0.802	0.305	28.77	0.851	0.212	30.14	0.886	0.139	4.76
EGRA [15]	27.10	0.799	0.309	28.80	0.850	0.214	30.32	0.885	0.140	6.17
Expan. [63]	27.21	0.790	0.303	28.47	0.831	0.241	29.67	0.862	0.183	5.05
Soft Mining [21]	27.80	0.819	0.274	29.29	0.859	0.137	30.73	0.888	0.136	6.94
EVOS	26.22	0.742	0.308	29.49	0.847	0.166	29.74	0.847	0.154	6.87

Table 12. Comparison of sampling strategies under extremely ultra-low selection ratio ($\beta = 0.05$). Green : the best performance.

disabling fuzzy indexing, setting $\alpha = 0.5$, and maintaining the original 1,000 warmup iterations.

Fuzzy Indexing Issue. Fuzzy Indexing serves as an engineering preprocessing step rather than an algorithmic component of soft mining. After LMC (Eq. 9), sampled points (coordinates) are processed into bounded values $w \in [0, 1]$ and subsequently scaled to the INR coordinate space, typically through min-max normalization to [-1,1]. This process can result in sampling coordinates beyond the scope of available ground truth values. For instance, sampled coordinates (5.52, 9.27) lack corresponding ground truth, which is only available at discrete points like (5,9) or (6,10), potentially lowering supervision accuracy. This fuzzy indexing issue is particularly pronounced with limited training data, leading to significant performance degradation in image fitting tasks while maintaining effectiveness in NeRF applications.

How to Disable Fuzzy Indexing? To mitigate the performance degradation caused by fuzzy indexing issues, we disabled this step by regulating w to real coordinate value. Specifically, after obtaining sampled value w, we first transform it to coordinate space and round it to the nearest integer (corresponding to real coordinates) before applying INR's min-max normalization. As shown in Table 11, reconstruction quality improves significantly after disabling fuzzy indexing. More details can be found in our code.

Analysis for Degraded Performance. As shown in Table 1, despite our optimized implementation, Soft Mining's performance remains unsatisfactory, particularly in later training stages, falling below uniform sampling. We attribute this limitation to LMC sampling mechanism's incompatibility with fitting tasks involving smaller sampling sets. A key distinction between general INR training (e.g., audio, text, image and shape fitting task) and NeRF training lies in their training data volume. NeRF training typically requires $N \sim 10^{10}$ points, calculated as:

$$N = \underbrace{(128 + 64)}_{\text{points along a ray}} \times \underbrace{1080 \times 768}_{\text{pixels}} \times \underbrace{100}_{\text{views}} \sim 10^{10}. \quad (11)$$

In standard NeRF training, each iteration uniformly samples $4096 \times (128 + 64) \sim 10^5$ points per batch, representing 0.005% of total training data. Conversely, a 1080P im-

Strategies	5 minutes PSNR ↑	SSIM ↑	LPIPS↓	15 minutes PSNR↑	SSIM↑	LPIPS↓	25 minutes PSNR↑	SSIM↑	LPIPS↓
Soft. [21] ($\beta = 0.05$)	30.19	0.878	0.156	31.57	0.901	0.106	32.01	0.907	0.092
Soft. [21] ($\beta = 0.5$)	31.48	0.903	0.124	34.02	0.936	0.062	35.14	0.947	0.046
EVOS ($\beta = 0.5$)	32.69	0.912	0.082	36.43	0.954	0.026	37.50	0.961	0.019

Table 13. Comparison of sampling strategies under fixed time budget. Reported times represent cumulative training duration across all dataset samples. All experimental settings follow Sec. 4.2. Green : the best performance.

age contains only $N \sim 10^5$ points, allowing INR fitting to access all training data in each iteration without batch splitting. This smaller sampling space and larger sampling ratio conflict with LMC sampling's design paradigm, potentially explaining Soft Mining's degraded performance in our experiments. To validate this hypothesis, we set sampling ratio $\beta = 0.05$ to simulate batch training in image fitting. Results in Table 12 support our hypothesis, with Soft Mining outperforming other methods, including EVOS, under ultra-low sampling ratios. However, as shown in Table 13, despite this advantage, Soft Mining's performance remains inferior to EVOS under equivalent time budgets for general INR acceleration.

F. Implementation of EGRA & Expan.

We reimplemented EGRA [15] and Expansive Supervision (Expan.) [63] following their algorithmic designs and hyperparameter settings, as official implementations were unavailable. More details can be found in our released code.

G. Compatibility for Different Scheduler

Settings. We further evaluated EVOS with linear and cosine increment schedulers. The linear scheduler increases the selection ratio from 0% to 100% across iterations, with selection intensity $q = \frac{N}{T}$, where T denotes total iterations and N represents total coordinates. The cosine scheduler implementation follows [61]. All other experimental settings remain consistent with Sec. 4.2.

Results. Results for linear and cosine schedulers are presented in Table 14 and Table 15, respectively. With the linear increment scheduler, EVOS reduces training time by 36.24% while achieving a 1.26 dB gain in PSNR. Similarly, with the cosine increment scheduler, it achieves a 36.49% reduction in training time with a 1.20 dB PSNR improvement. These results demonstrate EVOS's robust performance across different scheduling strategies.

H. Compatibility for Different Selection Ratio

Settings. We investigated the impact of selection ratio β under constant scheduler by evaluating $\beta = \{0.3, 0.7\}$. Results for $\beta = 0.5$ are presented in Table 1 (without underline). All other experimental settings remain consistent with Sec. 4.2.

Results. Results for selection ratios of 30% and 70% are presented in Table 16 and Table 17, respectively. Our method maintains efficiency improvements with 70% selection ratio; however, performance degrades with a lower sampling intensity (30%), where despite greater time reduction, it fails to surpass standard training performance.

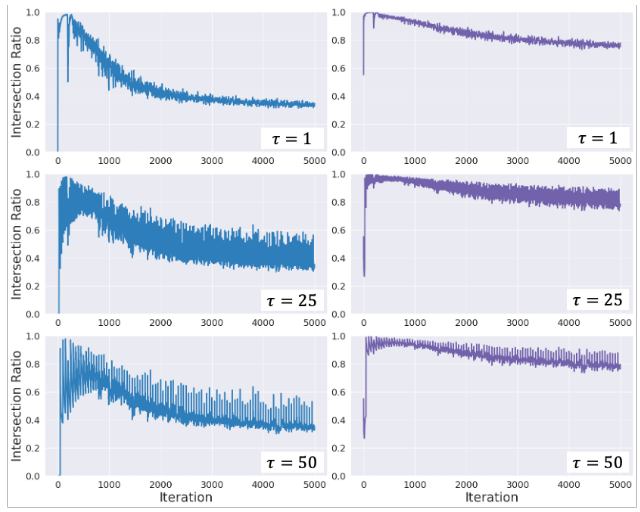

Figure 10. $\mathcal{G}_t(\tau, \sigma)$ curves across iterations with $\tau = \{1, 25, 50\}$ and $\sigma = \{0.1, 0.5\}$. **Blue** and **Purple** curves represent $\sigma = 0.1$ and $\sigma = 0.5$, respectively.

I. Empirical Study for Designing $\Gamma(t)$

In Section 3.2, we introduced a linear increasing scheduler (Eq. 2) for key iterations to balance performance and efficiency, motivated by the observed gradual linear increase in distribution changes of $\mathcal{D}(F_\theta(\mathbf{x}), \mathbf{y})$ across iterations. Here, we detail our analysis of the fitness distribution dynamics that guided the design of $\Gamma(t)$.

Measuring Distribution Changes in $\mathcal{D}(F_\theta(\mathbf{x}), \mathbf{y})$. We define the distribution change of $\mathcal{D}_t(F_\theta(\mathbf{x}), \mathbf{y})$ at step t as $\mathcal{G}_t(\tau, \sigma)$:

$$\mathcal{G}_t(\tau, \sigma) = \frac{|\tilde{\mathbf{x}}_t^\sigma \cup \tilde{\mathbf{x}}_{t-\tau}^\sigma|}{|\tilde{\mathbf{x}}_t^\sigma|}, \tag{12}$$

Strategies	1k Iterations PSNR ↑	1k Iterations SSIM ↑	1k Iterations LPIPS↓	2k Iterations PSNR↑	2k Iterations SSIM↑	2k Iterations LPIPS↓	5k Iterations PSNR↑	5k Iterations SSIM↑	5k Iterations LPIPS↓	Time↓ (min)
Standard	31.06	0.899	0.123	34.34	0.944	0.042	37.10	0.964	0.021	180.45
Uniform.	29.84	0.874	0.175	33.23	0.931	0.062	37.20	0.963	0.021	111.30
EGRA [15]	29.85	0.873	0.175	33.27	0.930	0.062	37.11	0.963	0.021	113.21
Expan. [63]	30.80	0.888	0.133	33.79	0.934	0.053	37.19	0.962	0.023	136.23
INT [61] (incre.)	31.68	0.889	0.129	31.79	0.890	0.103	37.23	0.963	0.021	129.41
INT [61] (dense.)	31.68	0.889	0.129	34.55	0.937	0.052	37.21	0.963	0.019	177.38
EVOS (proposed)	32.26	0.894	0.094	35.95	0.945	0.038	38.36	0.968	0.019	115.05

Table 14. Comparison of sampling strategies with **linear** increment scheduler. Green : the best performance.

Strategies	1k Iterations PSNR ↑	1k Iterations SSIM ↑	1k Iterations LPIPS↓	2k Iterations PSNR↑	2k Iterations SSIM↑	2k Iterations LPIPS↓	5k Iterations PSNR↑	5k Iterations SSIM↑	5k Iterations LPIPS↓	Time↓ (min)
Standard	31.06	0.899	0.123	34.34	0.944	0.042	37.10	0.964	0.021	180.45
Uniform.	29.54	0.868	0.188	33.00	0.928	0.068	37.11	0.962	0.023	110.81
EGRA [15]	29.54	0.866	0.189	33.08	0.928	0.066	37.27	0.964	0.0194	112.88
Expan. [63]	30.53	0.881	0.140	33.63	0.931	0.056	37.34	0.963	0.020	126.98
INT [61] (incre.)	31.68	0.882	0.132	30.81	0.869	0.138	37.22	0.963	0.019	128.77
INT [61] (dense.)	31.68	0.882	0.132	34.67	0.935	0.055	37.17	0.962	0.020	175.95
EVOS (proposed)	31.57	0.881	0.106	35.63	0.941	0.041	38.30	0.968	0.016	114.60

Table 15. Comparison of sampling strategies with **cosine** increment scheduler. Green : the best performance.

where

$$\tilde{\mathbf{x}}_t^{\sigma} = \underset{\{\tilde{\mathbf{x}}\}_{\sigma N} \subseteq \{\mathbf{x}\}_N}{\arg\max} \; (\underbrace{||F_\theta(\mathbf{x}) - \mathbf{y}||}_{\mathcal{D}_t(F_\theta(\mathbf{x}), \mathbf{y})}). \tag{13}$$

Here, τ represents the measurement interval, $\sigma \in (0, 1)$ denotes the sampling intensity for measurement, and N is the total number of coordinates in set $\mathbf{x}$. A higher $\mathcal{G}_t(\tau, \sigma)$ indicates a larger intersection ratio, implying minimal distribution change, and vice versa.

Settings & Results. We conducted experiments using standard training without sampling-based acceleration, following the settings in Sec. 4.2. We plotted $\mathcal{G}_t(\tau, \sigma)$ curves across iterations with $\tau = \{1, 25, 50\}$ and $\sigma = \{0.1, 0.5\}$. Results in Fig. 10 demonstrate that $\mathcal{G}_t(\tau, \sigma)$ exhibits a gradual linear decrease across various settings as iterations progress. This increasing trend in distribution changes of $\mathcal{D}(F_\theta(\mathbf{x}), \mathbf{y})$ validates our design choice for $\Gamma(t)$ in Eq. 2.

Strategies	1k Iterations PSNR ↑	SSIM ↑	LPIPS↓	2k Iterations PSNR↑	SSIM↑	LPIPS↓	5k Iterations PSNR↑	SSIM↑	LPIPS↓	Time↓ (min)
Standard	31.06	0.899	0.123	34.34	0.944	0.042	37.10	0.964	0.021	180.45
Uniform.	29.71	0.872	0.175	32.42	0.921	0.078	34.92	0.946	0.037	58.24
EGRA [15]	29.73	0.871	0.175	32.49	0.920	0.077	35.01	0.946	0.037	61.24
Expan. [63]	30.60	0.883	0.125	32.92	0.920	0.061	35.09	0.943	0.035	58.46
Soft. [21]	30.77	0.891	0.146	32.68	0.921	0.087	34.43	0.941	0.055	66.31
INT [61] (incre.)	31.74	0.890	0.121	26.62	0.738	0.289	29.45	0.832	0.162	85.13
INT [61] (dense.)	31.74	0.890	0.121	34.40	0.926	0.062	36.72	0.951	0.037	125.17
EVOS (proposed)	31.87	0.891	0.097	35.14	0.939	0.037	36.24	0.948	0.029	63.71

Table 16. Comparison of sampling strategies with **constant** increment scheduler ($\beta = 0.3$). Green : the best performance.

Strategies	1k Iterations PSNR ↑	SSIM ↑	LPIPS↓	2k Iterations PSNR↑	SSIM↑	LPIPS↓	5k Iterations PSNR↑	SSIM↑	LPIPS↓	Time↓ (min)
Standard	31.06	0.899	0.123	34.34	0.944	0.042	37.10	0.964	0.021	180.45
Uniform	30.72	0.893	0.134	33.89	0.939	0.048	36.85	0.961	0.021	126.83
EGRA [15]	30.75	0.892	0.134	33.96	0.939	0.047	36.92	0.961	0.020	129.30
Expan. [63]	31.28	0.899	0.116	34.34	0.940	0.042	37.10	0.961	0.019	127.76
Soft. [21]	31.89	0.908	0.113	33.98	0.935	0.065	35.79	0.952	0.040	139.05
INT [61] (incre.)	31.37	0.902	0.118	33.81	0.929	0.057	37.17	0.957	0.027	143.86
INT [61] (dense.)	31.37	0.902	0.118	34.42	0.941	0.0477	36.93	0.956	0.029	233.10
EVOS (proposed)	32.87	0.919	0.078	36.27	0.956	0.027	37.96	0.965	0.019	133.05

Table 17. Comparison of sampling strategies with **constant** increment scheduler ($\beta = 0.7$). Green : the best performance.

References

[1] The Stanford 3D Scanning Repository. https://graphics.stanford.edu/data/3Dscanrep/, 2007. Accessed: 2024-11. 2

[2] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 126–135, 2017. 5, 2

[3] Hmrishav Bandyopadhyay, Ayan Kumar Bhunia, Pinaki Nath Chowdhury, Aneeshan Sain, Tao Xiang, Timothy M. Hospedales, and Yi-Zhe Song. Sketchinr: A first look into sketches as implicit neural representations. *CoRR*, abs/2403.09344, 2024. 1, 2

[4] Zhicheng Cai, Hao Zhu, Qiu Shen, Xinran Wang, and Xun Cao. Batch normalization alleviates the spectral bias in coordinate networks. In *CVPR*, pages 25160–25171. IEEE, 2024. 3

[5] Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pages 608–625. Springer, 2020. 2

[6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European conference on computer vision*, pages 333–350. Springer, 2022. 1, 2, 3

[7] Chia-Hao Chen, Ying-Tian Liu, Zhifei Zhang, Yuan-Chen Guo, and Song-Hai Zhang. Joint implicit neural representation for high-fidelity and compact vector fonts. In *ICCV*, pages 5515–5525. IEEE, 2023. 2

[8] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser-Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. In *NeurIPS*, pages 21557–21568, 2021. 2

[9] Xiang Chen, Jinshan Pan, and Jiangxin Dong. Bidirectional multi-scale implicit neural representations for image deraining. In *CVPR*, pages 25627–25636. IEEE, 2024. 1

[10] Yinbo Chen and Xiaolong Wang. Transformers as meta-learners for implicit neural representations. In *ECCV (17)*, pages 170–187. Springer, 2022. 1, 2

[11] Yann N. Dauphin, Razvan Pascanu, Çaglar Gülçehre, KyungHyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *NIPS*, pages 2933–2941, 2014. 2

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

[13] Emilien Dupont, Hyunjik Kim, S. M. 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 *ICML*, pages 5694–5725. PMLR, 2022. 1, 2

[14] E.Kodak. Kodak dataset, 1999. 1

[15] Zhenbiao Gai, Zhenyang Liu, Min Tan, Jiajun Ding, Jun Yu, Mingzhao Tong, and Junqing Yuan. Egra-nerf: Edge-guided ray allocation for neural radiance fields. *Image and Vision Computing*, 134:104670, 2023. 1, 3, 5, 6, 2, 4, 7

[16] Yue He, Pengfei Tian, Renzhe Xu, Xinwei Shen, Xingxuan Zhang, and Peng Cui. Model-agnostic random weighting for out-of-distribution generalization. In *KDD*, pages 1050–1061. ACM, 2024. 7

[17] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992. 1

[18] Langwen Huang and Torsten Hoefler. Compressing multidimensional weather and climate data into neural networks. In *ICLR*. OpenReview.net, 2023. 1, 2

[19] Xin Huang, Qi Zhang, Ying Feng, Hongdong Li, and Qing Wang. Inverting the imaging process by learning an implicit camera model. In *CVPR*, pages 21456–21465. IEEE, 2023. 1, 2

[20] Mark W. Jones, J. Andreas Bærentzen, and Milos Srámek. 3d distance fields: A survey of techniques and applications. *IEEE Trans. Vis. Comput. Graph.*, 12(4):581–599, 2006. 2

[21] Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. Accelerating neural field training via soft mining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20071–20080, 2024. 2, 3, 5, 6, 1, 4, 7

[22] Subin Kim, Sihyun Yu, Jaeho Lee, and Jinwoo Shin. Scalable neural video representations with learnable positional features. In *NeurIPS*, 2022. 2

[23] Luca A. Lanzendörfer and Roger Wattenhofer. Siamese SIREN: audio compression with implicit neural representations. *CoRR*, abs/2306.12957, 2023. 1

[24] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H. Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *CVPR*, pages 8456–8465. IEEE, 2023. 2

[25] David B. Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. In *CVPR*, pages 16231–16241. IEEE, 2022. 2

[26] Ke Liu, Feng Liu, Haishuai Wang, Ning Ma, Jiajun Bu, and Bo Han. Partition speeds up learning implicit neural representations based on exponential-increase hypothesis. In *ICCV*, pages 5451–5460. IEEE, 2023. 1, 2

[27] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020. 3

[28] 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. *CoRR*, abs/2312.02434, 2023. 2, 5, 7, 1

[29] Shishira R. Maiya, Sharath Girish, Max Ehrlich, Hanyu Wang, Kwot Sin Lee, Patrick Poirson, Pengxiang Wu, Chen Wang, and Abhinav Shrivastava. NIRVANA: neural implicit representations of videos with adaptive networks and autoregressive patch-wise modeling. In *CVPR*, pages 14378–14387. IEEE, 2023. 2

[30] Julien N. P. Martel, David B. Lindell, Connor Z. Lin, Eric R. Chan, Marco Monteiro, and Gordon Wetzstein. Acorn: adaptive coordinate networks for neural scene representation. *ACM Trans. Graph.*, 40(4):58:1–58:13, 2021. 1, 2

[31] Ishit Mehta, Michaël Gharbi, Connelly Barnes, Eli Shechtman, Ravi Ramamoorthi, and Manmohan Chandraker. Modulated periodic activations for generalizable local functional representations. In *ICCV*, pages 14194–14203. IEEE, 2021. 3

[32] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV (1)*, pages 405–421. Springer, 2020. 1, 2, 3

[33] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 1, 2, 3

[34] Trung Thanh Nguyen, Shengxiang Yang, and Juergen Branke. Evolutionary dynamic optimization: A survey of the state of the art. *Swarm and Evolutionary Computation*, 6:1–24, 2012. 2

[35] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An ASR corpus based on public domain audio books. In *ICASSP*, pages 5206–5210. IEEE, 2015. 1

[36] Dogyun Park, Sihyeon Kim, Sojin Lee, and Hyunwoo J. Kim. DDMI: domain-agnostic latent diffusion models for synthesizing high-quality implicit neural representations. *CoRR*, abs/2401.12517, 2024. 2

[37] Jeong Joon Park, Peter R. Florence, Julian Straub, Richard A. Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, pages 165–174. Computer Vision Foundation / IEEE, 2019. 1, 2

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019. 5

[39] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron C. Courville. On the spectral bias of neural networks. In *ICML*, pages 5301–5310. PMLR, 2019. 1, 2

[40] Maziar Raissi, Paris Perdikaris, and George E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.*, 378:686–707, 2019. 1, 2

[41] Sameera Ramasinghe and Simon Lucey. Beyond periodicity: Towards a unifying framework for activations in coordinate-mlps. In *ECCV (33)*, pages 142–158. Springer, 2022. 2, 5, 7, 1

[42] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields. In *CVPR*, pages 14153–14161. Computer Vision Foundation / IEEE, 2021. 2

[43] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *ICCV*, pages 14315–14325. IEEE, 2021. 2

[44] Antony W. Rix, John G. Beerends, Michael P. Hollier, and Andries P. Hekstra. Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs. In *ICASSP*, pages 749–752. IEEE, 2001. 1

[45] Vishwanath Saragadam, Jasper Tan, Guha Balakrishnan, Richard G. Baraniuk, and Ashok Veeraraghavan. MINER: multiscale implicit neural representation. In *ECCV (23)*, pages 318–333. Springer, 2022. 1, 2

[46] Vishwanath Saragadam, Daniel LeJeune, Jasper Tan, Guha Balakrishnan, Ashok Veeraraghavan, and Richard G. Baraniuk. WIRE: wavelet implicit neural representations. In *CVPR*, pages 18507–18516. IEEE, 2023. 2, 5, 7

[47] Junwon Seo, Sangyoon Lee, Kwang In Kim, and Jaeho Lee. In search of a data transformation that accelerates neural field training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[48] Kexuan Shi, Hai Chen, Leheng Zhang, and Shuhang Gu. Inductive gradient adjustment for spectral bias in implicit neural representations, 2024. 3

[49] Kexuan Shi, Xingyu Zhou, and Shuhang Gu. Improved implicity neural representation with fourier bases reparameterized training. *CoRR*, abs/2401.07402, 2024. 3

[50] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020. 1, 2, 5, 7

[51] Yannick Strümpler, Janis Postels, Ren Yang, Luc Van Gool, and Federico Tombari. Implicit neural representations for image compression. In *ECCV (26)*, pages 74–91. Springer, 2022. 1, 2

[52] Filip Szatkowski, Karol J. Piczak, Przemyslaw Spurek, Jacek Tabor, and Tomasz Trzcinski. Hypernetworks build implicit neural representations of sounds. In *ECML/PKDD (4)*, pages 661–676. Springer, 2023. 1, 2

[53] Cees H. Taal, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen. An algorithm for intelligibility prediction of time-frequency weighted noisy speech. *IEEE Trans. Speech Audio Process.*, 19(7):2125–2136, 2011. 1

[54] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *NeurIPS*, 2020. 1, 2, 5, 7

[55] Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P. Srinivasan, Jonathan T. Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations. In *CVPR*, pages 2846–2855. Computer Vision Foundation / IEEE, 2021. 1, 2

[56] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility

to structural similarity. *IEEE Trans. Image Process.*, 13(4): 600–612, 2004. 5

[57] Shaowen Xie, Hao Zhu, Zhen Liu, Qi Zhang, You Zhou, Xun Cao, and Zhan Ma. DINER: disorder-invariant implicit neural representation. In *CVPR*, pages 6143–6152. IEEE, 2023. 1, 2, 3

[58] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *CVPR*, pages 5428–5438. IEEE, 2022. 3

[59] Runzhao Yang. TINC: tree-structured implicit neural compression. In *CVPR*, pages 18517–18526. IEEE, 2023. 3

[60] Zhi-Hui Zhan, Lin Shi, Kay Chen Tan, and Jun Zhang. A survey on evolutionary computation for complex continuous optimization. *Artificial Intelligence Review*, 55(1):59–110, 2022. 2

[61] Chen Zhang, Steven Tin Sui Luo, Jason Chun Lok Li, Yik-Chung Wu, and Ngai Wong. Nonparametric teaching of implicit neural representations. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. 1, 2, 3, 5, 6, 7

[62] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595. Computer Vision Foundation / IEEE Computer Society, 2018. 5

[63] Weixiang Zhang, Shuzhao Xie, Shijia Ge, Wei Yao, Chen Tang, and Zhi Wang. Expansive supervision for neural radiance field. *arXiv preprint arXiv:2409.08056*, 2024. 1, 3, 5, 6, 2, 4, 7

[64] Weixiang Zhang, Shuzhao Xie, Chengwei Ren, Shijia Ge, Mingzi Wang, and Zhi Wang. Enhancing implicit neural representations via symmetric power transformation, 2024. 3

[65] Xingxuan Zhang, Peng Cui, Renzhe Xu, Linjun Zhou, Yue He, and Zheyan Shen. Deep stable learning for out-of-distribution generalization. In *CVPR*, pages 5372–5382. Computer Vision Foundation / IEEE, 2021. 7

[66] Tianle Zhong, Jiechen Zhao, Xindi Guo, Qiang Su, and Geoffrey Fox. RINAS: training with dataset shuffling can be general and fast. *CoRR*, abs/2312.02368, 2023. 2, 5